Eduardo Pereira Borges
pereira@dcc.ufmg.br

Um modelo de medição para processos de desenvolvimento de software

Dissertação apresentada ao Departamento de Ciência da Computação do Instituto de Ciências Exatas da Universidade Federal de Minas Gerais como requisito parcial para a obtenção do grau de Mestre em Ciência da Computação.

Belo Horizonte

Março de 2003

Dedico este trabalho a meus pais, Enira e Vicente.

Agradecimentos

Ao meu orientador, professor Wilson, pela competência, bom senso e presteza demonstrados nesses três semestres em que trabalhamos juntos.

Aos professores Clarindo e Maurílio, pelo apoio oferecido quando decidi ingressar no mestrado.

À Fernanda, pelo amor, pela compreensão e pela preciosa revisão do texto.

Ao amigo Eduardo Júnior, pelo incentivo e pela colaboração na estruturação e formatação do trabalho.

Ao Valdo, pela importante ajuda durante a construção do protótipo da ferramenta.

Ao Synergia, pela bolsa concedida e pela oportunidade de vivenciar, na prática, várias das questões que procurei abordar – uma prova de que projetos de extensão produzem, sim, bons resultados de pesquisa para a Universidade. Aos companheiros de trabalho, em especial à Equipe de Requisitos e Análise, agradeço pelo apoio e pela compreensão diante das inevitáveis ausências.

Minha gratidão também àqueles que, mesmo não tendo participado diretamente do trabalho, proporcionaram-me a paz de espírito necessária ao desenvolvimento de uma empreitada como esta. Destaco nesse grupo os familiares, os amigos músicos, o *Maracatu Lua Nova*, o grupo *Pé de Saia e o Menino* e o *Estúdio de Pesquisas Musicais*.

Resumo

No desenvolvimento de produtos de software, a realização de medições diversas pode fornecer informação importante sobre o andamento dos projetos e a qualidade dos produtos. Mais do que isso, medidas quantitativas constituem uma forma de registrar precisamente a memória dos projetos de uma organização, formando uma base de dados históricos que pode ser usada para caracterizar, ajustar e melhorar progressivamente os processos em uso. O presente trabalho explora aspectos comuns dos processos de desenvolvimento mais utilizados pela indústria de software e propõe um arcabouço genérico de medição que pode ser configurado e aplicado a boa parte deles. Técnicas consagradas pela literatura são utilizadas para selecionar e definir um conjunto de medidas capaz de caracterizar quantitativamente alguns aspectos-chave do ciclo de desenvolvimento (esforço, tamanho dos produtos, defeitos, progresso, cronograma e estabilidade de requisitos) e fornecer dados que contribuam para a melhoria da qualidade dos produtos e da produtividade e previsibilidade dos processos.

Abstract

In software development, different kinds of measures provide important information about ongoing projects and the quality of products. Moreover, quantitative measures are a way of precisely registering the memory of the projects of an organization, forming a historical baseline which can be used to characterize, adapt and progressively improve the processes being used. This work exploits common aspects of development processes widely used by the software industry and proposes a generic measuring framework, which can be applied to most of them. Popular techniques are utilized to select and define a set of measures in order to quantitatively characterize some key aspects of the development cycle (cost, size of products, number of defects, progress, schedule and requirements stability) and provide data that contribute to improve the quality of products, productivity and predictability of processes.

Sumário

Lista de figuras

Lista de tabelas

Capítulo 1

Introdução

O crescimento contínuo da demanda por produtos de software cada vez maiores e mais complexos, robustos e confiáveis, tem exigido o desenvolvimento de técnicas mais precisas para controle e gestão dos projetos. No entanto, o cenário típico que geralmente se observa nas organizações produtoras de software mostra-nos que tais técnicas, quando existentes, não são usadas adequadamente em grande parte dos casos. Estimativas equivocadas de prazos e custos, ou produtos que apresentam baixa qualidade, infelizmente constituem situações comuns no desenvolvimento de software.

Grande parte dos projetos e produtos é desenvolvida e evoluída em bases artesanais, cuja única preocupação é atingir o objetivo de prazo, negligenciando-se os atributos de qualidade dos produtos. E freqüentemente nem o objetivo de prazo é atingido, resultando em atrasos, orçamentos ultrapassados e insatisfação geral. Além disso, experiências passadas não são bem aproveitadas, e é comum uma organização cometer o mesmo erro diversas vezes.

Em uma organização, o aproveitamento da experiência obtida ao longo do desenvolvimento de cada produto é de extrema importância, na medida em que vai progressivamente compondo uma base de conhecimento que contribuirá para um melhor desempenho em projetos futuros. As lições aprendidas permitem que os processos sejam continuamente ajustados, tornando-se progressivamente mais adequados, robustos, estáveis e previsíveis. O grau com que a experiência anterior pode contribuir para a maturidade de uma organização dependerá de sua capacidade de capturar e organizar informações. E a melhor forma de obter tal conhecimento é através de medições.

Em outros ramos da engenharia a realização de medições diversas constitui uma prática corriqueira, cujos benefícios já foram consagrados e extensamente explorados na literatura. O chamado *Controle Estatístico de Processos*, que consiste na aplicação de técnicas estatísticas para a análise das medições realizadas sobre os processos e produtos, tornou-se um princípio básico para a gestão e evolução dos processos em uso. A idéia é mantê-los sob um rigoroso controle quantitativo e trabalhar para que eles atinjam um certo grau de estabilidade e previsibilidade. Feito isso, é possível realizar estimativas acuradas, acompanhar o desempenho frente a essas previsões e identificar ações que podem ser tomadas para melhorá-los, de forma precisa e objetiva.

Em Engenharia de Software, no entanto, a utilização desses princípios permaneceu tímida por muitos anos. Tentativas mal sucedidas de medição, aliadas ao custo da coleta de dados quantitativos e à diversidade dos fatores humanos e subjetivos envolvidos no desenvolvimento de software (o que dificulta sua mensuração), foram fatores que contribuíram para esse cenário.

O crescimento recente do interesse nessa área é em parte devido às iniciativas do Departamento de Defesa dos Estados Unidos (DoD) e do *Software Engineering Institute* (SEI), na Universidade de Carnegie Mellon. Práticas e recomendações foram publicadas juntamente com relatos e estudos de caso, fazendo com que o termo *medidas* (ou *métricas*[1]) *de software* passasse a figurar com freqüência cada vez maior no jargão da Engenharia de Software.

São vários os benefícios trazidos pelas medições. Através delas é possível obter dados precisos relacionados ao esforço de desenvolvimento, tamanho e qualidade de produtos. Além disso, facilitam a construção de modelos capazes de produzir previsões a respeito de parâmetros do processo ou do produto, que podem ser utilizadas no planejamento e acompanhamento dos projetos. Medições são tipicamente utilizadas em dois contextos:

- No contexto de um projeto de desenvolvimento, dados quantitativos podem fornecer informações precisas quanto às características do produto em desenvolvimento, além de possibilitar mecanismos adequados de acompanhamento e controle da evolução do projeto. Medições permitem identificar, com prontidão, divergências entre comportamentos planejados e obtidos ao longo do curso do projeto, possibilitando a adoção rápida de contramedidas necessárias.
- Na melhoria dos processos, a mensuração permite que se tenha um melhor conhecimento da eficácia do processo de desenvolvimento, através do acompanhamento quantitativo de diversos aspectos como produtividade, qualidade dos produtos desenvolvidos, acurácia de estimativas, entre outros. Esse conhecimento constitui a base para a realização das atividades de melhoria de processos.

Uma prova da importância crescente das atividades de mensuração é o fato dos modelos de capacitação utilizados no mercado, como o SW-CMM [PAULK93], o CMMI [CMMI02] e os padrões e certificações ISO [ISO01] terem incluído a prática dessas atividades como um requisito indispensável a um processo de desenvolvimento maduro.

[1] Este trabalho evitará a utilização do termo *métrica* por não existir um consenso na literatura sobre seu significado. Além disso, padrões recentes como o ISO/IEC 15939 [ISO01] recomendam a utilização preferencial do termo *medida*.

1.1 Motivação

Apesar dos benefícios trazidos, a implementação de uma política de mensuração em uma organização não constitui uma tarefa fácil. Em primeiro lugar, devido à quantidade e à complexidade dos fatores envolvidos no desenvolvimento de um software de grande porte, há tantos aspectos a serem medidos que torna-se necessário selecionar apenas um subconjunto restrito, de forma a viabilizar as atividades de coleta e análise. Essa seleção das medidas a serem coletadas deve ser cuidadosamente estudada e fortemente fundamentada, pois interfere diretamente na qualidade e relevância dos resultados obtidos com a mensuração. Sem uma política bem definida para a definição desse subconjunto de medidas, a probabilidade de se escolher arbitrariamente as mais adequadas é remota.

Outra dificuldade consiste em definir e padronizar o processo de coleta e análise dos dados. É necessário criar e documentar procedimentos, padronizar formulários e planilhas de coleta e análise, e estabelecer critérios claros de medição. Caso contrário, diferentes interpretações das regras de mensuração pelas pessoas envolvidas podem influenciar os resultados obtidos, reduzindo sua confiabilidade.

Além disso, tais procedimentos envolvem geralmente uma grande quantidade de informação, principalmente no contexto dos grandes projetos de software. Nesses casos, o registro e a análise manual dos dados torna-se inviável, tanto pelo risco de ocorrência de erros decorrentes de falhas humanas, quanto pelo custo envolvido. A existência de ferramentas automatizadas que dêem suporte a essas atividades torna-se essencial.

Finalmente, as informações coletadas devem ser armazenadas de forma estruturada, preferencialmente em uma base de dados centralizada. Para isso, é necessário que os aspectos de processo e de produto que serão submetidos à medição sejam agrupados em um modelo conceitual único, que especifique com precisão as entidades envolvidas, seus atributos e relacionamentos. É esse modelo conceitual que servirá como base para a análise das medidas coletadas e permitirá a comparação entre dados de diferentes projetos.

Por esses motivos, quando uma organização opta por iniciar um programa de mensuração, várias providências devem ser tomadas antes que se dê início à coleta dos dados: as medidas a serem coletadas devem ser selecionadas, procedimentos de medição devem ser padronizados e documentados, a forma de armazenamento das medidas em uma base histórica precisa ser definida, e diretrizes para a análise desses dados devem ser estabelecidas. Esse conjunto de definições forma um *modelo de informação de medição*, que por sua vez fornece a base para a implantação de um *processo* de coleta e análise de medidas. Juntos, o modelo de informação e o processo de medição compõem uma estrutura para a implantação da mensuração em uma organização [McGarry01].

Como uma política de mensuração precisa estar adequada ao contexto em que será aplicada, vários autores defendem a idéia de que cada organização deve criar seu próprio modelo

de informação, e que não é possível estabelecer um conjunto de fixo medidas que atenda a todos os casos [DoD00][Basili94b]. Por isso, a literatura geralmente se limita à exposição de conceitos e ao estabelecimento de diretrizes e requisitos que devem ser observados na criação do modelo para que se obtenha bons resultados com a mensuração. É essa a tendência que tem sido seguida também em normas e padrões internacionais de qualidade, como o padrão ISO/IEC 15939 [ISO01].

No entanto, mesmo levando-se em consideração as particularidades de cada organização, é natural supor que muitas delas possuam características em comum. Afinal, existem alguns aspectos que são inerentes a todos os processos de desenvolvimento, tais como a preocupação com qualidade, a produtividade e a previsibilidade [Florac97]. Se fosse confeccionado um modelo de informação genérico que levasse em consideração essas semelhanças, tal modelo poderia ser potencialmente utilizado por um grande número de organizações como base para a implantação de um processo de mensuração.

A principal dificuldade para se confeccionar um modelo de medição com essas características consiste na dependência de algumas medidas em relação a detalhes do processo de desenvolvimento que está sendo medido. Como cada organização geralmente adota um processo específico e as medidas coletadas freqüentemente tratam de particularidades desses processos, a utilização do mesmo modelo de medição em contextos diferentes é dificultada: um modelo concebido para um processo específico dificilmente será aplicável a outros.

Uma possível abordagem para lidar com essa limitação seria escolher não um processo específico, mas um tipo de *arquitetura* de processo que fosse amplamente utilizada pela comunidade de desenvolvimento de software. A partir dela seria possível propor um modelo que pudesse ser parametrizado e personalizado para a utilização em diferentes processos (e personalizações) estruturados sobre tal arquitetura, o que facilitaria significativamente a incorporação de práticas de medição em organizações que adotassem esses processos.

Para utilizar essa abordagem, uma questão crucial é a escolha de uma arquitetura de processos que seja difundida o suficiente para que o modelo possa ser potencialmente aplicável a um grande grupo de organizações. Uma arquitetura que possui tal característica é aquela que divide o processo em uma estrutura matricial, como é descrito a seguir.

1.2 Processos de arquitetura matricial

Dentre os processos de desenvolvimento existentes atualmente, vários deles dividem as atividades executadas ao longo do ciclo de desenvolvimento em duas dimensões, formando uma estrutura matricial: uma dimensão relacionada ao tempo, que divide o desenvolvimento em etapas, e uma divisão em fluxos de trabalho, que classifica as atividades executadas ao longo do desenvolvimento de um produto em disciplinas de Engenharia de Software (requisitos,

implementação, testes etc.).

Dos processos que utilizam esse tipo de estrutura, podemos destacar o RUP (*Rational Unified Process*) [KRUCHTEN00], o Praxis [PAULA03] e o OOSP (*Object Oriented Software Process*) [AMBLER99]. Neste trabalho adotaremos a nomenclatura definida pelo Processo Unificado [JACOBSON98], que constitui a base dos dois primeiros processos citados.

O Processo Unificado foi proposto por Booch, Jacobson e Rumbaugh em 1998, reunindo métodos propostos anteriormente pelos três autores [Booch94][Booch96][Jacobson94][Rumbaugh91]. Nele, o desenvolvimento é visto como uma série de iterações, cada uma consistindo de atividades de levantamento de requisitos, análise, desenho, implementação e testes. Essa organização reflete as duas dimensões da estrutura matricial:

- Na dimensão do tempo, o ciclo de desenvolvimento de um produto é dividido em **fases**. Uma fase corresponde a um intervalo de tempo entre dois marcos bem definidos, cada marco correspondendo a um ponto de aceitação por parte do cliente. Por sua vez, cada fase possui divisões menores, que são denominadas **iterações**. Essas subdivisões permitem que pontos de controle intermediários sejam estabelecidos dentro das fases, facilitando o acompanhamento dos projetos.
- A segunda dimensão corresponde à divisão em **fluxos** (ou **disciplinas**). Em todas as fases, as atividades técnicas realizadas são divididas em subprocessos chamados *fluxos de trabalho*[2], cada um correspondendo a um tema técnico específico.

Originalmente, o Processo Unificado é composto por quatro fases (Concepção, Elaboração, Construção e Transição) e cinco fluxos (Requisitos, Análise, Desenho, Implementação e Testes). Cada fluxo é executado com maior ou menor intensidade ao longo das iterações. Essa estrutura pode ser observada na FIGURA 1, que mostra a distribuição típica do esforço ao longo dos fluxos, fases e iterações do processo durante o desenvolvimento de um produto.

[2] O termo *fluxo de trabalho* (ou simplesmente *fluxo*) será utilizado neste trabalho como tradução do termo em inglês *workflow*.

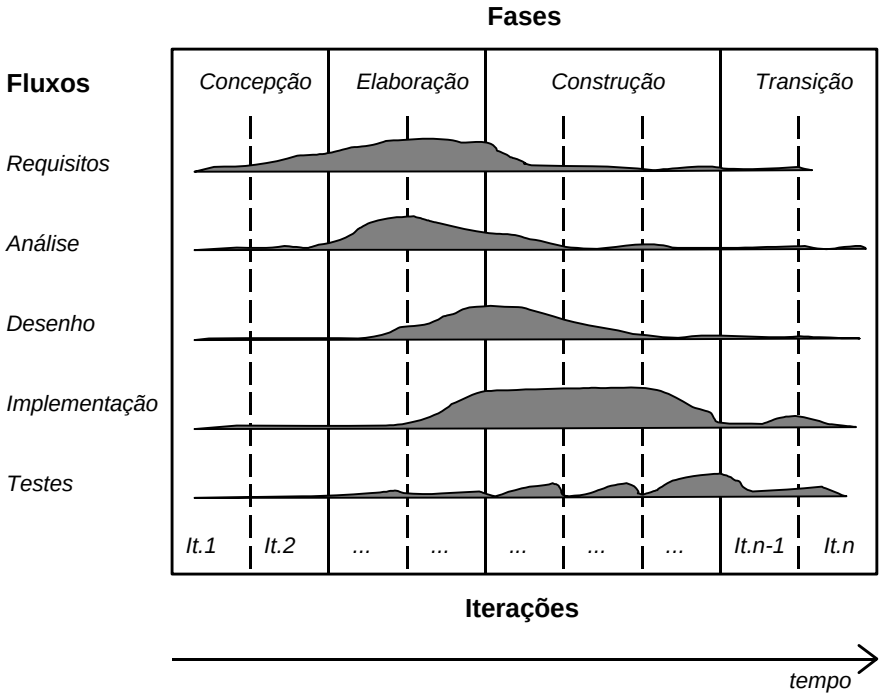


FIGURA 1 - O ciclo de desenvolvimento no Processo Unificado (adaptado de [JACOBSON98])

Essa arquitetura proposta pelo Processo Unificado tem sido amplamente utilizada em processos adotados pela indústria de software, e a tendência é que se torne a base dos processos comerciais mais usados em um futuro próximo [PAULA03].

A divisão em fases e fluxos existe também em outros processos, ainda que sob nomenclaturas diferentes. Por exemplo, no OOSP [AMBLER99] o ciclo de desenvolvimento é dividido em quatro fases (*Início*, *Construção*, *Entrega* e *Manutenção e Suporte*) e um conjunto de *papéis* (analista, desenhista, programador etc.), que podem ser vistos como uma analogia aos fluxos de trabalho. Até mesmo os processos que possuem uma única dimensão, como acontece com os processos em cascata, podem ser modelados em termos de fases e fluxos. É o caso, por exemplo, do PSP (*Personal Software Process*) [HUMPHREY95], que pode ser modelado como sendo constituído de seis fases (Planejamento, Desenho, Codificação, Compilação, Teste e Retrospectiva) e um único fluxo, já que esse processo não divide as atividades em fluxos de trabalho.

1.3 Objetivos do trabalho

A proposta principal deste trabalho consiste na confecção de um modelo de medição voltado para processos de desenvolvimento de software que sejam baseados em arquiteturas

matriciais.

Por *modelo de medição* entende-se o conjunto dos seguintes elementos:

- Definição de um conjunto de atributos cuja mensuração seja significativa para a obtenção de informações úteis sobre o andamento dos projetos e a melhoria dos processos (exemplos de atributos são o tamanho dos produtos, o esforço dedicado aos projetos e os defeitos existentes nos produtos desenvolvidos);
- Convenções e procedimentos que descrevam a forma pela qual cada atributo deve ser mensurado;
- Um modelo conceitual que una todos os atributos em uma única estrutura, formalizando os conceitos envolvidos e explicitando os relacionamentos existentes entre eles.

Além disso, a meta do trabalho é que o modelo de medição proposto atenda aos seguintes requisitos:

- Ser flexível o suficiente para que possa ser adaptado a diferentes processos de arquitetura matricial, ou a personalizações de um mesmo processo.
- Ser fundamentado em técnicas consagradas pela literatura e amplamente utilizadas pela comunidade de desenvolvimento de software, como o paradigma GQM [BASILI92], o *Goal-Driven Software Measurement* [PARK96] e os conceitos do PSM [DoD00].

Para validar o modelo e exemplificar como o mesmo pode ser aplicado, este trabalho prevê também a descrição de sua instanciação em um processo real: o Praxis 2.0 [PAULA03].

Finalmente, como o alto custo das atividades de medição exige o apoio de ferramentas automatizadas, também fazem parte deste trabalho a especificação e modelagem de uma ferramenta para esse fim.

1.4 Limites do trabalho

Tão importante quanto enumerar os objetivos do trabalho é esclarecer os limites do mesmo, visando delimitar mais precisamente o escopo que está sendo tratado.

Em primeiro lugar, é importante ressaltar que o produto deste trabalho é um *modelo*, e não um *processo* de medição. O que o trabalho propõe é uma estrutura de dados que modele aspectos mensuráveis do processo de desenvolvimento, além de alguns procedimentos e convenções para a coleta desses dados. A divisão das atividades relacionadas à mensuração em um conjunto ordenado de passos e a atribuição de responsabilidades aos agentes envolvidos, que

seriam necessários para que o modelo proposto fosse considerado um processo bem definido, estão fora do escopo deste trabalho. Utilizando a nomenclatura definida pelo modelo ISO/IEC-15939 (apresentado no Capítulo 2), podemos dizer que o foco deste trabalho está no *modelo de informação*, e não no *modelo de processo*.

A segunda ressalva é referente às medidas que compõem o modelo. Apesar deste trabalho propor um conjunto básico de medidas que permita às organizações extraírem informações relevantes para um melhor conhecimento dos produtos e processos, esse conjunto não deve ser encarado como uma proposta fixa e universal. Como será apresentado ao longo do texto, o conjunto ideal de medidas a serem coletadas em cada situação depende diretamente dos objetivos que se pretende atingir com a medição – afinal, a mensuração é apenas uma técnica usada para que tais metas possam ser atingidas de maneira controlada. Se os objetivos mudam, por conseqüência, os aspectos a serem mensurados também sofrem alteração.

O que está sendo proposto é um conjunto básico de medidas derivado a partir de três objetivos: qualidade dos produtos, produtividade e previsibilidade. Como essas constituem metas comuns a grande parte das organizações, acredita-se que as medidas propostas possam ser aplicadas diretamente a um grande número delas. No entanto, quando outros objetivos forem estabelecidos, além dos três citados, é necessário ajustar o processo de medição, acrescentando novos atributos mensuráveis. Em outros casos, pode ser interessante reduzir a quantidade de dados a coletar – por exemplo, no início da implantação da mensuração em uma organização, pode ser interessante começar coletando-se um conjunto reduzido de dados e ir ampliando aos poucos. O modelo de medição proposto mantém um rastreamento entre medidas e objetivos, que pode ser utilizado como referência nos casos em que for necessário realizar tais personalizações.

Finalmente, o foco do trabalho é voltado para processos de desenvolvimento de aplicativos interativos. Mesmo que vários conceitos possam ser aplicados também no desenvolvimento de aplicações em lote, sistemas embutidos ou de tempo real, a utilização do modelo nesses contextos possivelmente exigirá uma série de adaptações.

1.5 Organização deste documento

O trabalho de confecção do modelo de medição seguiu uma seqüência bem definida de atividades, que foi utilizada como base para a estruturação dos primeiros capítulos deste documento.

A primeira atividade consistiu no estudo de conceitos relacionados à área de mensuração em software, envolvendo aspectos teóricos, diretrizes para a seleção das medidas, diretrizes para o processo de medição como um todo, além de padrões e recomendações estabelecidos internacionalmente. O resultado desse estudo é descrito no **Capítulo 2**, e foi

utilizado como fundamentação para todo o restante do trabalho.

Em seguida, foi feita a seleção das medidas que irão compor o modelo de medição proposto. Como veremos, todo o modelo é construído sobre essas medidas, e por isso a escolha de um conjunto adequado delas é de vital importância para que a mensuração seja bem sucedida. O **Capítulo 3** descreve o processo que foi utilizado para guiar essa escolha e o resultado obtido.

Uma vez escolhidas as medidas, iniciou-se a construção do modelo. Cada medida foi detalhada e formalizada, e procedimentos de coleta foram padronizados. Todos os elementos de dados foram agrupados em um modelo conceitual único, que serviu como base para a definição de algumas informações importantes que podem ser derivadas a partir de combinações dessas medidas. A descrição completa do modelo construído encontra-se no **Capítulo 4**.

O **Capítulo 5** apresenta um exemplo de aplicação do modelo proposto. O exemplo consiste na instanciação do modelo para um processo concreto, o Praxis 2.0 [PAULA03], com a intenção de verificar sua capacidade de ser configurado e servir de guia para outras instanciações.

O **Capítulo 6** descreve a especificação de uma ferramenta capaz de oferecer apoio informatizado às atividades de medição durante a utilização do modelo. Tal ferramenta, denominada *Quantum*, foi especificada e modelada seguindo o processo Praxis, e durante sua elaboração várias das medidas propostas foram coletadas como forma de experimentar os procedimentos de medição definidos no Capítulo 4.

Finalmente, o **Capítulo 7** apresenta as contribuições e conclusões observadas ao longo do trabalho, além de apontar sugestões de melhorias e novos trabalhos que podem ser desenvolvidos com base nos resultados aqui obtidos.

O documento vem acompanhado ainda de três apêndices, com informações que complementam alguns aspectos apontados ao longo do texto. O **Apêndice A.1** apresenta a listagem completa das entidades e atributos identificados durante o processo de seleção das medidas descrito no Capítulo 2. Nesse mesmo capítulo são propostos vários indicadores (gráficos e tabelas) que podem gerar informações úteis a partir dos dados coletados. Esses indicadores foram posteriormente detalhados, e encontram-se descritos no **Apêndice A.2**. Por último, o **Apêndice A.3** apresenta sugestões de medidas derivadas que podem ser calculadas a partir dos dados existentes no modelo, complementando as definições contidas no Capítulo 4.

Capítulo 2

Medições em software

Este capítulo apresenta a descrição de alguns conceitos e recomendações existentes na literatura, os quais foram utilizados como referência para a construção do modelo de medição proposto:

- A seção 2.1 descreve conceitos importantes relacionados à teoria das medições, tais como escalas e unidades de medidas.
- A seção 2.2 apresenta técnicas e recomendações publicadas que descrevem como deve ser feita a seleção das medidas a serem coletadas. Tais técnicas foram utilizadas para guiar a escolha das medidas que compõem o modelo que está sendo proposto, como será descrito no Capítulo 3.
- A seção 2.3 apresenta alguns padrões e recomendações para o processo de medição usados como referência para a formalização do modelo, que será detalhado no Capítulo 4.

2.1 Conceitos relacionados à mensuração

Medição ou *mensuração* é o processo pelo qual números ou símbolos são associados a atributos de entidades no mundo real, com o objetivo de descrevê-la de acordo com um conjunto de regras claramente definidas [FENTON94]. Uma *entidade* pode ser uma pessoa, um objeto (como uma especificação de requisitos) ou um evento (como um dia de trabalho ou um teste executado sobre um produto). Um *atributo* é uma propriedade da entidade, como a altura ou pressão sangüínea (de uma pessoa), o tamanho ou a funcionalidade (de uma especificação), o custo ou duração (de uma atividade).

A mensuração produz como resultado um conjunto de medidas. Uma *medida* constitui um mapeamento entre um atributo empírico e uma escala matemática [KITCHENHAM95]. *Unidades de medida* são estabelecidas para convencionar como esses atributos devem ser registrados. Um mesmo atributo pode ser mensurado através de diferentes unidades de medida; por exemplo, para

o atributo “temperatura” podem ser usadas as unidades Celsius, Fahrenheit ou Kelvin.

Uma unidade de medida está geralmente associada a uma *escala*, que determina as operações e transformações que lhe podem ser aplicadas. Existem quatro tipos de escalas [PFLEEGER97][MILLS88]:

- Uma escala *nominal* divide um conjunto de itens em diferentes categorias. É a escala utilizada, por exemplo, quando classificamos produtos de software quanto à linguagem de programação utilizada em sua construção. Não permite a realização de comparações aritméticas nem a ordenação de valores; a única operação possível é a verificação de igualdade entre dois valores mensurados.
- Uma escala *ordinal* também divide os itens em categorias, como na escala nominal, mas nela as categorias representam uma ordem. Um exemplo de escala ordinal é a classificação de um defeito quanto à sua gravidade (alta, média ou baixa).
- Uma escala de *intervalo* define uma distância entre um ponto e outro, de tal forma que existam intervalos de mesmo tamanho entre números consecutivos. Esse tipo de escala permite a execução de operações de adição, subtração e o cálculo de valores médios. No entanto, não existe um zero absoluto, dificultando a comparação entre grandezas. É o caso das temperaturas medidas em graus Celsius: não se pode afirmar que 30ºC representa o “dobro” de temperatura se comparado a uma temperatura de 15ºC. No contexto de software, um exemplo de aplicação desse tipo de escala é na medida de complexidade proposta em [MCGABE76] (complexidade ciclomática).
- Uma escala de *razão* difere da escala de intervalo por possuir um zero absoluto, permitindo o cálculo da razão entre grandezas. No exemplo de medição de temperatura, a escala em Kelvin constitui uma escala desse tipo. Em desenvolvimento de software, um bom exemplo é o número de linhas de código: um código com 2000 linhas pode ser naturalmente interpretado como tendo o dobro do tamanho de outro que possua 1000 linhas.

No contexto específico do desenvolvimento de software, a medição pode ser vista como o processo de definir, coletar, analisar e agir sobre medidas que possam potencialmente melhorar tanto a qualidade do software desenvolvido por uma organização quanto o próprio processo de desenvolvimento utilizado [AMBLER99]. Diversas medidas de software já foram propostas pela literatura ([DOD00], por exemplo, apresenta 51 sugestões de medidas), e a seleção das medidas mais adequadas em cada caso depende dos objetivos que se pretende atingir com a medição.

Em linhas gerais, medidas podem ser utilizadas para atender aos seguintes objetivos [HUMPHREY89]:

- ***Conhecer***: Os dados podem ser coletados para prover um conhecimento mais preciso de um item ou processo.
- ***Avaliar***: Dados quantitativos podem ser usados para verificar se um produto ou atividade atende aos critérios de aceitação.
- ***Controlar***: Dados podem ser usados para acompanhar e controlar alguma atividade.
- ***Prever***: Os dados podem ser usados para gerar indicadores de tendências ou estimativas.

É o detalhamento desses objetivos que conduz a seleção do conjunto de medidas adequado para cada organização. Métodos para guiar esse detalhamento e a seleção das medidas serão apresentados na seção 2.2.

2.1.1 Tipos de medidas de software

Medidas de software podem ser classificadas por diferentes aspectos: a natureza do atributo que está sendo medido (medidas de produto e de processo), o relacionamento entre a medida e o atributo medido (medidas básicas e derivadas), a objetividade (medidas objetivas e subjetivas) e o momento da mensuração (medidas preditivas e explanatórias). Os itens a seguir definem cada uma dessas formas de classificação.

- **Medidas de produto *x* medidas de processo** [MILLS88]: *medidas de produto* são aquelas obtidas a partir de características de um produto ou artefato em qualquer estágio do desenvolvimento, como o código-fonte ou uma especificação de requisitos. Exemplos desse tipo de medida são o tamanho de um produto, a complexidade de um código e o número de páginas de um documento.

  Medidas de processo são obtidas a partir das atividades envolvidas no desenvolvimento dos produtos. O esforço dedicado a cada atividade e a quantidade de defeitos injetados ou detectados em cada uma são exemplos de medidas que se enquadram nesse grupo.

- **Medidas básicas *x* derivadas** [GRADY87]: *medidas básicas* (comumente chamadas também de medidas primitivas, diretas ou explícitas) são aquelas que podem ser mensuradas a partir de observação direta dos atributos envolvidos. São exemplos de medidas básicas a quantidade de linhas de comando existentes em um código-fonte, a data em que um defeito é detectado, ou o esforço dedicado a uma atividade.

  Medidas derivadas (ou indiretas) são aquelas que não podem ser mensuradas diretamente a partir da observação de um atributo, mas são calculadas a partir de combinações de outras medidas. Exemplos de medidas derivadas são aquelas

relacionadas a produtividade (como linhas de código/pessoa-mês) e qualidade (número de defeitos/ponto de função, por exemplo).

- **Medidas objetivas *x* subjetivas** [HUMPHREY89] [MILLS88]: uma *medida objetiva* consiste na contagem absoluta de atributos do produto ou processo, sendo idealmente independente do autor da mensuração: a mesma medição, realizada por duas pessoas diferentes, gera resultados idênticos. Exemplos incluem o número de linhas de código e número de defeitos detectados em um produto.

  Medidas subjetivas envolvem a classificação ou qualificação de um aspecto do produto ou processo, baseada em julgamento humano. Por isso, observadores diferentes podem medir valores diferentes para um mesmo atributo, caso possuam opiniões divergentes. Exemplos de medidas subjetivas são a classificação de um defeito quanto à gravidade e a avaliação do grau com que uma técnica ou método é utilizado por uma organização.

  Medidas objetivas contribuem para uma maior precisão e confiabilidade das informações coletadas, e devem ser usadas sempre que possível. A utilização de medidas subjetivas exige um cuidado especial de padronização e documentação dos critérios utilizados, para reduzir ao máximo a dependência em relação à opinião pessoal do observador (a não ser nos casos em que é a própria opinião pessoal que está sendo mensurada).

- **Medidas preditivas *x* explanatórias** [HUMPHREY89]: *Medidas preditivas* consistem em estimativas geradas com o objetivo de prever certos aspectos do desenvolvimento com antecedência. É o caso de todas as estimativas geradas durante o planejamento de um projeto.

  Medidas explanatórias são produzidas a partir da ocorrência dos eventos, com o intuito de caracterizá-los objetivamente. Exemplos de medidas explanatórias são o registro dos defeitos detectados ao longo do desenvolvimento, a contagem de linhas em um trecho de código concluído, ou o registro da duração de uma atividade realizada.

2.2 Diretrizes para a seleção das medidas

O primeiro passo para a definição de uma política de mensuração é a escolha das medidas a serem coletadas. Essa escolha deve ser feita com critérios bem fundamentados, devido ao grande número de opções possíveis e ao custo envolvido na coleta de cada informação.

Em razão da quantidade e da complexidade dos fatores envolvidos no desenvolvimento de um software de grande porte, o número de propriedades que podem ser

potencialmente medidas é consideravelmente alto. Exemplos vão desde o tamanho do produto em desenvolvimento e tempo gasto em determinadas atividades, até aspectos relacionados ao ambiente de trabalho, como o nível de ruído ou a disponibilidade de espaço físico. Em certo grau, todos esses fatores exercem alguma influência sobre o processo de desenvolvimento, e sua medição poderia levar a conclusões interessantes. Mas a coleta de cada informação envolve inevitavelmente um custo, e por isso torna-se necessário restringir as medições a um subconjunto limitado de elementos.

A escolha adequada desses elementos constitui um fator fundamental para o sucesso de qualquer política de medição. Deve ser feita de forma bastante cuidadosa, e seguir critérios claros. Caso contrário, a probabilidade de se escolher arbitrariamente as medidas mais adequadas é muito remota.

Vários mecanismos podem ser utilizados para guiar o processo de seleção das medidas, destacando-se: o paradigma GQM (*Goal-Question-Metric*) [BASILI92][BASILI94A], o QFD (*Quality Function Deployment Approach*) [KOGURE83] e o SQM (*Software Quality Metrics Approach*) [BOEHM76].

Em linhas gerais, o que essas propostas pregam é que a seleção das medidas deve tomar como base os objetivos que se pretende atingir com a mensuração. A idéia é que a primeira pergunta a se fazer durante a seleção de um conjunto de métricas a serem coletadas não seja “Que medidas devem ser usadas?”, mas “O que se deseja aprender?” [ROMBACH89]. Um programa de medição que tenha o foco somente na coleta dos dados, sem que haja uma visão clara de como eles podem ser aplicados para auxiliar a organização a atingir seus objetivos, dificilmente terá sucesso.

Dentre as propostas citadas, o paradigma GQM constitui a mais simples e abrangente (uma comparação entre as três abordagens pode ser vista em [Basili92]), tendo sido por isso adotada e recomendada extensivamente pela literatura.

A próxima seção descreve o GQM em maiores detalhes. Em seguida, é apresentada uma extensão desse paradigma, também bastante difundida, e que foi utilizada como base para este trabalho: o processo *Goal-Driven Software Measurement*.

2.2.1 O paradigma *Goal-Question-Metric*

O GQM (*Goal-Question-Metric*) é um paradigma bastante difundido para guiar a escolha das medidas mais adequadas para um processo de mensuração [Basili92][Basili+94]. Foi originalmente desenvolvido para a caracterização e a avaliação de defeitos em projetos desenvolvidos por um dos laboratórios de Engenharia de Software da NASA, no início da década de 80. Após alguns experimentos foi estendido para um contexto mais amplo, podendo ser aplicado a partir de então a qualquer processo de medição. Uma descrição detalhada do GQM, incluindo diretrizes práticas para sua aplicação, pode ser obtida em [SOLINGEN99].

Sua principal característica é a utilização de uma abordagem *top-down* para a definição das medidas. O GQM prega que o processo de mensuração não deve ser guiado pelas medidas em si, mas pelos objetivos que se pretende atingir com sua coleta. Nesse sentido, as medições só devem ser realizadas se estiverem fundamentadas por uma meta claramente definida.

O ponto de partida para a escolha das medidas é, portanto, a definição das metas que irão guiar o restante do processo. De forma resumida, o GQM descreve três passos que devem ser seguidos:

- Definir um conjunto de metas que se pretende atingir quanto ao processo e aos produtos com as atividades de medição. As metas geralmente estão diretamente relacionadas a questões de relevância fundamental para a organização, como a satisfação dos usuários, a qualidade dos produtos e a produtividade.
- Gerar uma série de perguntas que traduzam essas metas em aspectos quantitativos, e que possam ser alvos de medição. A idéia é gerar perguntas que, se respondidas, ajudariam a organização a atingir as metas estabelecidas.
- Especificar as medidas (métricas) que precisariam ser coletadas de forma a obter as informações necessárias para responder às perguntas geradas.

A FIGURA 2 ilustra o relacionamento existente entre metas, perguntas e métricas.

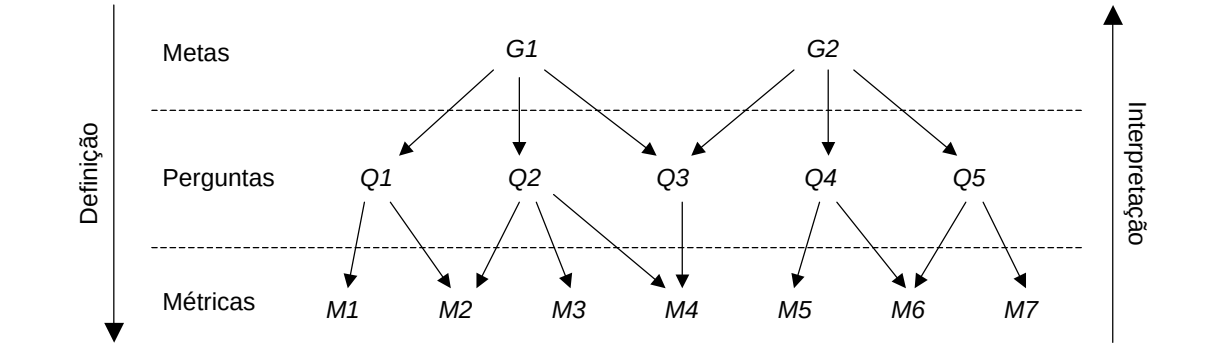


FIGURA 2 - O paradigma GQM [SOLINGEN99]

Como as metas constituem o ponto central de toda a estrutura de medição, a especificação delas recebe uma ênfase especial no processo. O GQM estabelece um conjunto de informações que a definição de uma meta deve conter:

- Uma *proposta*, que especifica o objeto alvo da medição (geralmente um processo ou um produto) e a razão da coleta. Razões típicas são a caracterização, avaliação, previsão ou melhoria do objeto especificado.
- Uma *perspectiva*, que define o aspecto a ser utilizado como referência (custo, correção e remoção de defeitos são alguns exemplos) e o ponto de vista que será

levado em consideração, que pode ser o dos usuários, dos clientes, dos desenvolvedores e dos gestores de projeto, entre outros.

- Uma descrição do *ambiente*, que descreve o contexto que está sendo levado em consideração. O objetivo dessa descrição é registrar as características do ambiente em que as medições serão realizadas: o processo utilizado, o tipo de software produzido, a infra-estrutura utilizada, características da equipe, etc. O registro dessas características é importante para que medidas coletadas em contextos similares possam ser agrupadas para comparação.

Para guiar a definição de uma meta quanto a todos esses aspectos, o GQM propõe um gabarito onde elas podem ser especificadas. Esse gabarito está ilustrado na FIGURA 3.

Proposta	Analisar o(a) _____ *(objeto: processo, produto)* com a intenção de _____ *(razão: conhecer, avaliar, prever, melhorar)*
Perspectiva	o(a) _____ *(foco: custo, correção, confiabilidade, ...)* do ponto de vista do(a) _____ *(quem: usuário, cliente, desenvolvedor, gerente, ...)*
Ambiente	no seguinte contexto: _____ *(descrição do ambiente)*.

FIGURA 3 - Gabarito para definição de meta, no GQM [BASILI92]

Um exemplo fictício de meta poderia ser *analisar os produtos desenvolvidos com a intenção de avaliar a confiabilidade sob o ponto de vista dos usuários*. Tomando-a como ponto de partida, várias perguntas poderiam ser geradas. Alguns exemplos são:

- Qual a taxa de defeitos existentes no produto final?
- Quais os tipos de defeitos mais comuns no produto final?
- Qual a gravidade desses defeitos?
- As correções estão sendo feitas à medida que os defeitos são detectados?
- Novos defeitos estão sendo introduzidos durante a correção dos problemas detectados?

Para cada uma das perguntas, é possível chegar a um conjunto de medições que precisariam ser realizadas para que as mesmas possam ser respondidas de maneira quantitativa.

Para a primeira pergunta, por exemplo, chegaria-se ao seguinte conjunto de medidas:

- Tamanho do produto final (em linhas de código, por exemplo);
- Número de defeitos detectados após a entrega do produto final ao cliente;
- Densidade de defeitos, que pode ser calculada como o número de defeitos dividido pelo tamanho do produto.

Usando a classificação apresentada na seção anterior, as duas primeiras medidas citadas são exemplos de medidas *básicas*, enquanto a terceira é uma medida *derivada* a partir das demais. Tendo essas medidas em mãos, seria possível responder à primeira das perguntas geradas, e com isso adquirir um conhecimento relevante para que a meta de medição possa ser alcançada. Seguindo o mesmo princípio para as demais perguntas, é possível obter um conjunto completo de medidas capaz de fornecer as informações necessárias ao acompanhamento do desempenho da organização quanto à meta estabelecida.

A metodologia proposta pelo GQM apresenta uma série de benefícios:

- Seguindo-se os passos corretamente, é possível garantir que as medições realizadas estejam alinhadas com as metas da organização. Com isso, o risco de coletar dados que posteriormente não mostrem utilidade prática fica bastante reduzido.
- O motivo da coleta de cada uma das medidas fica claramente estabelecido. Com isso, é mais fácil conscientizar os envolvidos da importância de coletar os dados corretamente.
- O rastreamento mantido entre metas e métricas, se devidamente documentado, serve como base para futuras adaptações e melhorias do processo de medição. Quando as metas mudam, o reflexo no conjunto de medidas pode ser facilmente percebido, e então elas podem ser adaptadas às novas necessidades.
- Esse mesmo rastreamento, se percorrido no sentido oposto (*bottom-up*), pode ser usado como um guia para a interpretação do resultado das medições. Os dados coletados são usados para responder às perguntas, e então é possível conhecer a situação atual da organização em relação às metas estabelecidas.

Como toda a estrutura de medição é derivada a partir das metas estabelecidas, a definição adequada dessas metas torna-se crucial para o sucesso do processo. Mas identificar as metas de medição mais importantes em cada caso exige uma certa experiência, e organizações que estejam iniciando-se na prática da mensuração podem ter dificuldades em utilizar o GQM com eficácia.

2.2.2 O *Goal-Driven Software Measurement*

Uma extensão do paradigma GQM foi proposta em 1996 pelo SEI (*Software Engineering Institute*), na Universidade de Carnegie Mellon: o *Goal-Driven Software Measurement* [PARK96].

O *Goal-Driven Software Measurement* praticamente transforma os princípios propostos pelo GQM em um processo completo para a definição de uma política de medição, detalhando passo a passo uma seqüência de atividades que produz ao final um conjunto de medidas adequado às necessidades de uma organização.

Além de descrever mais detalhadamente os passos a serem seguidos, o *Goal-Driven Software Measurement* traz duas importantes contribuições em relação à proposta do GQM:

- Ao invés de partir das metas específicas de medição, o processo toma como base as *metas de negócio* da organização, que são mais abrangentes e geralmente mais conhecidas. A seqüência de passos apresentada descreve como detalhar as metas de negócio até obter as metas específicas de medição, a partir de quando os princípios do GQM podem ser seguidos.
- No GQM, as medidas necessárias são deduzidas diretamente a partir das perguntas. No *Goal-Driven Software Measurement* são introduzidos os *indicadores*, que constituem um nível intermediário entre perguntas e métricas para auxiliar a identificação das métricas mais adequadas. Essa versão adaptada do GQM é freqüentemente denominada GQ(I)M – *Goal-Question-Indicator-Metric*.

A estrutura do processo é composta de dez passos, que devem ser seguidos de maneira ordenada:

1. Identificação de metas de negócio;
2. Identificação do que se deseja aprender;
3. Identificação de submetas, que refinam as metas de negócio;
4. Identificação de entidades e atributos envolvidos;
5. Formalização de metas de medição;
6. Identificação de questões quantitativas e indicadores relacionados às metas de medição;
7. Identificação de elementos de dados a serem coletados para a construção dos indicadores apontados;
8. Definição e padronização das medições a serem realizadas;
9. Identificação de ações necessárias para a implementação do processo de medição;
10. Preparação de um plano para a implementação do processo.

Os quatro primeiros passos têm como objetivo gerar uma estrutura de informações que servirá como base para a definição das metas de medição. Tendo sido definidas essas metas, os princípios do GQM são aplicados (eles correspondem aproximadamente ao quinto, sexto e sétimo passos). Finalmente, os três passos restantes procuram tornar as medidas obtidas aplicáveis operacionalmente, através da padronização do processo de coleta e do planejamento da implantação desse processo na organização.

A organização dos sete primeiros passos pode ser vista na FIGURA 4. Nela, são demonstrados os resultados obtidos em cada passo (o número dos passos está indicado entre parênteses), e como esses resultados formam insumos para os passos seguintes. Os passos que correspondem ao GQM foram destacados, para que possam ser comparados à estrutura apresentada na seção anterior (FIGURA 2).

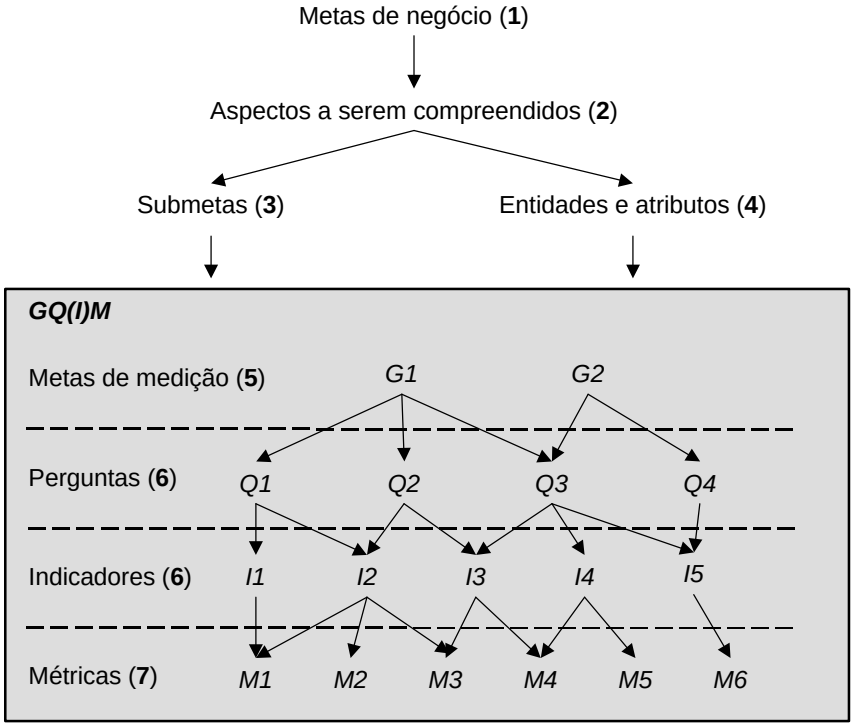


FIGURA 4 - *Goal-Driven Software Measurement*

As metas de negócio devem representar objetivos estratégicos da organização, e não precisam estar diretamente relacionadas a aspectos mensuráveis do processo. Um exemplo de meta de negócio seria o aumento da satisfação do usuário quanto aos produtos desenvolvidos. Para cada meta, é possível enumerar uma série de aspectos que devem ser compreendidos, verificados, previstos ou melhorados para que ela possa ser alcançada. Para a meta de aumento da satisfação dos usuários, por exemplo, os seguintes aspectos poderiam ser enumerados:

- A documentação fornecida ao usuário é suficientemente clara?
- Os produtos possuem níveis aceitáveis de defeitos?
- O tempo gasto para a correção de um defeito detectado atende às expectativas dos usuários?
- Os produtos são entregues em dia?
- A interface homem-máquina oferecida pelos produtos é satisfatória?

Assim como esses, vários outros aspectos poderiam ser listados. Agrupando-os em temas é possível desdobrar a meta de negócio em outras mais específicas, chamadas de *submetas*. Para a meta exemplificada, possíveis submetas seriam:

- Melhorar a legibilidade dos documentos;
- Melhorar a confiabilidade e o desempenho do produto final;
- Melhorar o acompanhamento do progresso em relação aos compromissos assumidos;
- Melhorar a eficácia do processo de manutenção;
- Melhorar a comunicação com o usuário.

Após a definição das submetas, o passo seguinte consiste na identificação de entidades e atributos envolvidos no processo que estará sendo medido. Para cada entidade (código-fonte, manual de usuário, defeitos, processo de manutenção, etc) define-se um conjunto de atributos mensuráveis, como tamanho, distribuição, freqüência, tempo e esforço.

A lista de submetas obtidas para cada meta de negócio e o conjunto de entidades enumeradas com seus respectivos atributos formam a base para a aplicação do paradigma GQM, que ocorre a partir do quinto passo do processo. A diferença entre o GQM utilizado aqui – chamado também de *Goal-Question-Indicator-Metric* ou simplesmente GQ(I)M – e o GQM padrão, descrito na seção anterior, é a utilização dos *indicadores*. Um indicador é uma representação de uma combinação de medidas em um formato específico (geralmente listas, tabelas ou gráficos), que se construído poderia fornecer informações úteis para responder às perguntas derivadas das metas de medição.

Para ilustrar a utilização de indicadores, a FIGURA 5 apresenta um exemplo. Trata-se de um gráfico que combinaria medições realizadas em cada defeito detectado após a entrega de um produto, e que poderia ser usado para responder à pergunta: “quais os tipos de defeitos mais comuns no produto final?”, mencionada como exemplo na seção 2.2.1. Uma vez idealizado esse indicador, é possível especificar as medidas que precisariam ser coletadas para possibilitar sua construção (que seria, neste caso, a classificação de cada defeito detectado quanto ao tipo).

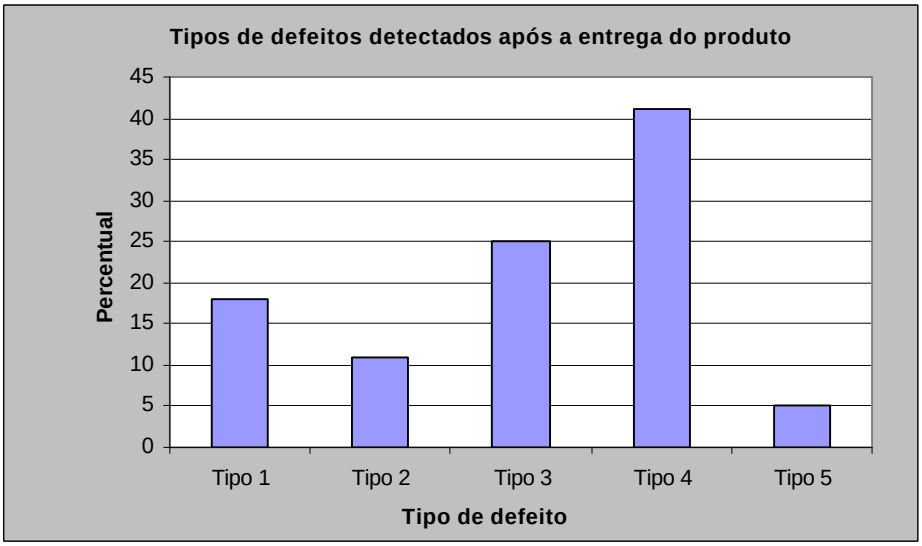


FIGURA 5 - Exemplo de indicador no GQ(I)M

Aplicando esse princípio a todas as metas de medição especificadas, obtém-se um conjunto de medidas alinhado com as metas de negócio da organização. Resta então que os procedimentos de coleta sejam padronizados e que sua implantação no processo de desenvolvimento utilizado pela organização seja planejada.

Para a padronização dos procedimentos de coleta, o *Goal-Driven Software Measurement* recomenda a confecção de listas de conferência, como proposto em [CARLETON92]. Para o planejamento da implantação das práticas de medição, [PARK96] apresenta um gabarito que pode ser utilizado para a confecção de um plano de implementação.

2.3 Padrões e recomendações para o processo de medição

2.3.1 *Practical Software Measurement* e ISO/IEC 15939

O *Practical Software Measurement Project* é um projeto patrocinado pelo Departamento de Defesa e pelo exército norte-americanos, que tem como objetivo estabelecer um conjunto de práticas, ferramentas e serviços para auxiliar os gerentes de projetos a obter informações objetivas sobre os projetos em andamento, para que estes atinjam suas metas de prazo, custo e qualidade. A iniciativa conta com a participação de órgãos do governo, universidades e empresas privadas, e sua estratégia é reunir as práticas de maior sucesso na área de mensuração em software para compor um processo único, que possa ser utilizado como guia

para a implantação de uma política de medição em organizações. Esse processo foi denominado *Practical Software Measurement*, ou simplesmente PSM.

As iniciativas tomadas pelo projeto resultaram em dois produtos principais:

- Diretrizes e recomendações técnicas para o processo de mensuração, reunidas sob a forma de um livro [DoD00];
- Uma ferramenta de apoio ao processo, denominada *PSM Insight* [PSMSC03].

A primeira versão do PSM foi publicada em 1997, e o processo atualmente se encontra em sua quarta versão. Em 2001, os princípios do PSM foram formalizados em um padrão, o ISO/IEC 15939 [ISO01], que estabeleceu algumas convenções terminológicas.

O PSM considera que as atividades de mensuração em software, para obterem sucesso, devem possuir duas características [McGarry01]:

- Um alinhamento direto das atividades de coleta, análise e divulgação de dados medidos com as **necessidades de informação** dos responsáveis pela tomada de decisões nos projetos. Nesse ponto, os princípios do PSM se assemelham aos do GQM, ressaltando a importância de se estabelecer de antemão os objetivos que se pretende atingir com a mensuração;
- A existência de um **processo de mensuração** estruturado e bem documentado, que defina com precisão as atividades de medição.

A estrutura do PSM é baseada em dois modelos, cada um voltado para a obtenção de uma dessas características:

- Um **modelo de informação**, que determina como as medidas a serem utilizadas devem ser especificadas e estruturadas para atender às necessidades de informação;
- Um **modelo de processo**, que especifica como o processo de mensuração deve ser conduzido.

O modelo de informação é um mecanismo para mapear as necessidades de informação em termos dos atributos do produto ou processo que devem ser mensurados. Segundo ele, a construção de uma medição deve descrever como os atributos serão quantificados e combinados para formar indicadores que forneçam a base para a tomada de decisões. Para isso, devem ser definidas medidas em três diferentes níveis:

- *Medidas básicas*: medidas obtidas diretamente da observação das entidades do produto ou processo que está sendo medido, onde atributos dessas entidades são mapeados em valores de uma escala através de um método de medição;
- *Medidas derivadas*: valores obtidos a partir da aplicação de algoritmos ou funções matemáticas que combinam duas ou mais medidas (que podem ser medidas

básicas ou mesmo outras medidas derivadas);

- *Indicadores*: visão de um conjunto de medidas em um formato (geralmente gráfico ou tabular) voltado para o atendimento de uma ou mais necessidades de informação.

A FIGURA 6 apresenta como esses conceitos são combinados no PSM, formando uma estrutura para a construção das medições.

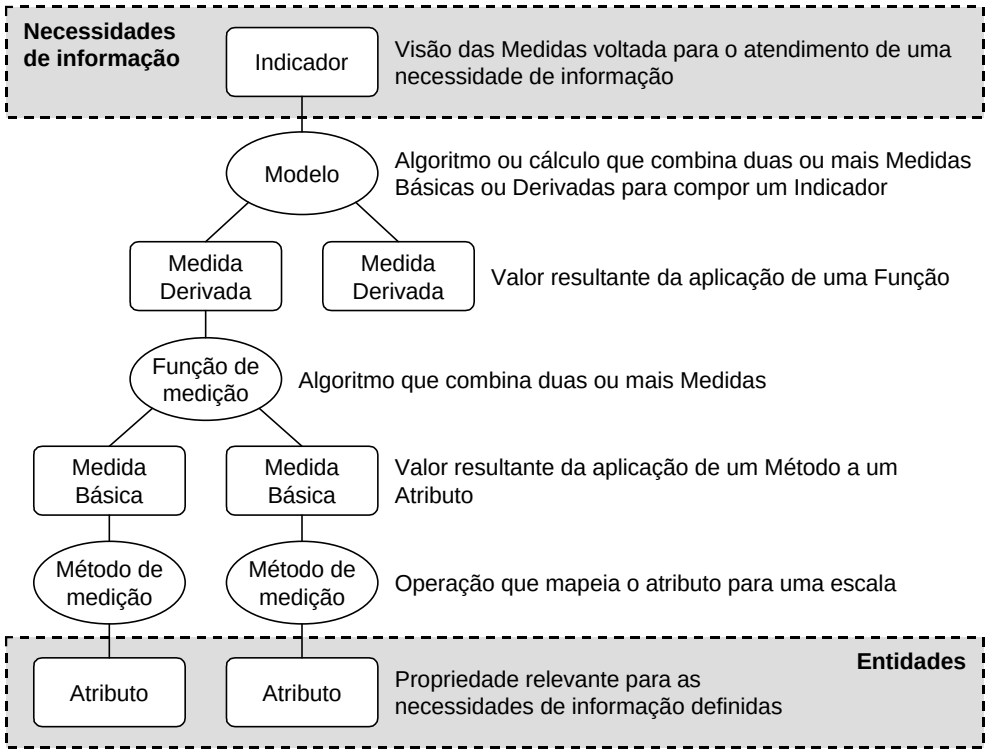


FIGURA 6 - Construção de uma medição pelo PSM

Enquanto o modelo de informação fornece uma estrutura para relacionar as necessidades de informação a um conjunto de medidas, o modelo de processo apresenta um arcabouço para a implementação da mensuração em um projeto, descrevendo as principais atividades que devem ser realizadas.

O modelo de processo é baseado em quatro atividades principais, que segundo o PSM são essenciais para o sucesso da implementação de qualquer política de medição:

- *Planejar mensuração*: envolve a identificação das necessidades de informação para um projeto e a seleção das medidas mais adequadas para atender a essas necessidades, através da instanciação do modelo de informação. Também inclui a formalização dos procedimentos de coleta, análise e divulgação dos dados, a definição de regras e convenções sobre a condução do processo de medição e o

planejamento dos recursos e tecnologias a serem disponibilizados para a mensuração.

- *Executar mensuração*: consiste na implementação das atividades previstas no plano de mensuração durante a execução do projeto, através da coleta e análise das medidas e de sua utilização no apoio à tomada de decisões pelos gestores.
- *Avaliar mensuração*: consiste em aplicar técnicas de medição e análise para avaliar o próprio processo de mensuração. O objetivo dessa atividade é identificar possibilidades de melhoria nesse processo, para garantir que ele seja continuamente atualizado para suprir novas necessidades de informação, promovendo um crescente amadurecimento da organização quanto à medição.
- *Estabelecer e sustentar comprometimento*: envolve o fornecimento de recursos e infra-estrutura necessários à implementação e à manutenção do programa de mensuração, garantindo o apoio organizacional ao processo.

A FIGURA 7 exibe como essas atividades são combinadas para formar o modelo de processo do PSM. Através dela é possível observar também como o processo de mensuração interage com os demais processos técnicos e gerenciais da organização.

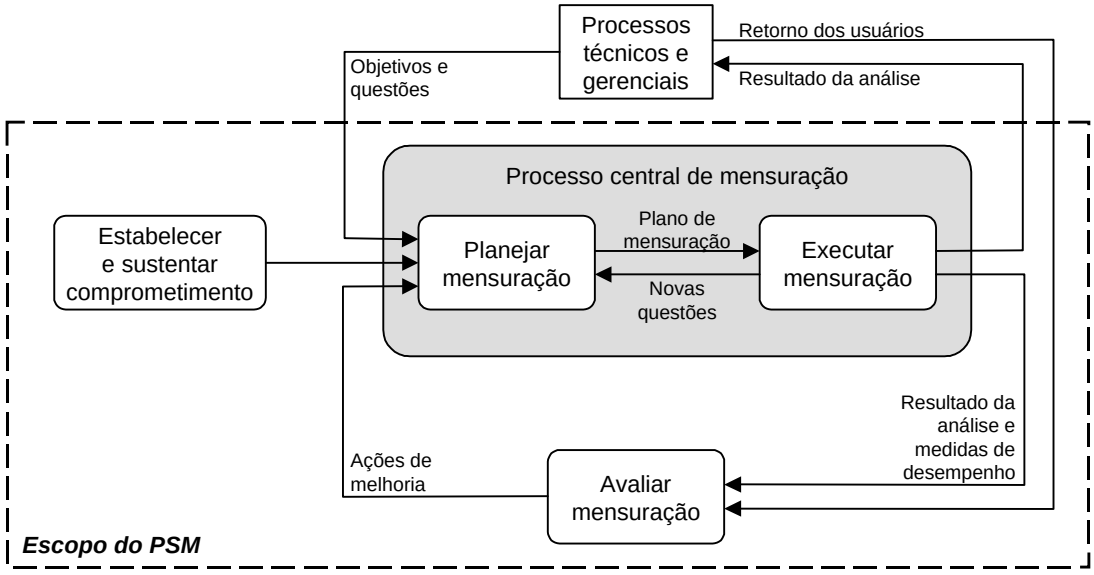


FIGURA 7 - Modelo de processo de medição, do PSM [MCGARRY01]

O material do PSM apresenta também várias sugestões de medidas (51 ao todo) [DOD00], que foram elaboradas com base na experiência vivenciada na implantação de programas de medição nas organizações participantes. Essas medidas cobrem as seguintes áreas:

- *Cronograma e progresso*: área relacionada ao cumprimento tempestivo de marcos

de projeto e à conclusão de unidades de trabalho nos prazos previstos.

- *Recursos e custo*: área que lida com a adequação entre o trabalho a ser executado e os recursos alocados ao projeto.
- *Tamanho e estabilidade de produtos*: lida com a estabilidade da funcionalidade ou da capacidade requerida do software, e com o volume de software necessário para contemplar essa capacidade.
- *Qualidade de produtos*: área relacionada à capacidade do software produzido em atender às expectativas dos usuários.
- *Desempenho de processos*: trata da capacidade do processo em atender às necessidades impostas por cada projeto.
- *Eficácia de tecnologia*: lida com a viabilidade e adequação das soluções técnicas propostas (arquiteturas, ferramentas e tecnologias), tendo em vista as características de cada projeto.
- *Satisfação dos clientes*: área relacionada ao grau com que os produtos e serviços oferecidos atendem às expectativas dos clientes.

Apesar de estarem mais voltadas para o planejamento e controle dos projetos (o que está relacionado ao segundo nível de maturidade do modelo CMMI, descrito na seção 2.3.2), as medidas propostas pelo PSM podem ser utilizadas também como fonte para a construção de uma base de dados históricos que dê suporte ao controle quantitativo dos processos e da qualidade (nível 4 do CMMI). [FLORAC97] descreve como estender o processo de medição para esse contexto.

Para auxiliar o registro e a análise das medidas, o material do PSM inclui também uma ferramenta, chamada *PSM Insight* [PSMSC03]. Disponibilizada gratuitamente, essa ferramenta permite a definição das medidas a serem coletadas, o registro dos valores medidos para cada uma delas e a construção de indicadores gráficos para auxiliar a análise dos resultados. No entanto, sua utilização em grandes organizações é comprometida pelo fato de não suportar acessos concorrentes, característica fundamental para viabilizar o registro tempestivo das medições pelos desenvolvedores. Uma alternativa para o problema pode ser a utilização de outra ferramenta para o registro dos dados, com a posterior exportação para o *PSM Insight*, que serviria apenas de suporte à análise e construção dos indicadores. A ferramenta Quantum, proposta neste trabalho (Capítulo 6), poderia ser utilizada para esse fim.

2.3.2 Medições nos modelos SW-CMM e CMMI

O *Software Capability Maturity Model*, ou simplesmente SW-CMM, é um modelo de capacitação desenvolvido pelo Software Engineering Institute (o mesmo instituto citado anteriormente na seção referente ao *Goal-Driven Software Measurement*). Lançado oficialmente

no final dos anos 1980, o modelo encontra-se atualmente na versão 1.1, descrita em [PAULK93] e [PAULK95].

Os objetivos de um modelo de capacitação são, basicamente, permitir que os processos de uma organização sejam avaliados quanto à sua maturidade e servir de guia para as atividades que visam à melhoria desses processos. Para isso, o SW-CMM segue uma arquitetura em estágios, agrupando as práticas recomendadas em diferentes níveis de maturidade. São cinco níveis ao todo, que representam estágios crescentes de maturidade de um processo:

Nível	Nome	Características
1	Inicial	Estágio inicial dos produtores de software. A organização utiliza processos informais e métodos ad-hoc (às vezes descritos como caóticos) e é incapaz de fazer estimativas técnicas de recursos e prazos dos projetos.
2	Repetitivo	A organização já adota um conjunto de práticas que permitem a ela assumir compromissos referentes a requisitos, prazos e custos, com alta probabilidade de cumpri-los. Essas práticas são adotadas no nível dos projetos, e dificuldades podem surgir quando houver mudanças na tecnologia, nos tipos de produtos desenvolvidos ou na estrutura organizacional.
3	Definido	A organização possui uma infra-estrutura de processos bem definidos, que permite a adaptação a vários tipos de mudanças. A limitação deste nível de maturidade é que o conhecimento dos processos por parte da organização ainda é basicamente qualitativo.
4	Gerido	O domínio dos processos de software evolui para uma forma quantitativa. A organização é proficiente em coletar medidas e gerir a base de dados de processo, e sabe intervir nos processos para atingir metas de qualidade dos produtos.
5	Otimizante	Estágio em que os processos estão em melhoria contínua, sendo constantemente otimizados para as necessidades de cada momento.

TABELA 1 - Níveis do SW-CMM (adaptado de [PAULA03])

Cada um desses níveis é composto por um conjunto de *áreas-chave* de processo, e cada uma delas identifica um grupo de atividades correlatas capazes de trazer benefícios importantes quando executadas em conjunto. Em cada nível, as áreas chaves correspondem a questões que devem ser resolvidas para que aquele nível possa ser atingido. Cada área-chave, por sua vez, define um conjunto de *metas* que devem ser alcançadas para que a mesma possa ser considerada como sendo dominada pela organização.

O SW-CMM deverá ser substituído futuramente pelo seu sucessor, o modelo CMMI – *Capability Maturity Model Integration* [CMMI02]. O CMMI é bastante mais complexo que o SW-CMM 1.1, abordando de forma integrada processos de engenharia de software, engenharia de sistemas, definição de produtos e aquisição. Como uma das melhorias trazidas pelo CMMI em relação ao SW-CMM diz respeito justamente às atividades de medição, este trabalho dará maior enfoque ao CMMI. Nele, essas atividades são tratadas como uma área-chave específica no nível 2 (denominada *Medição e Análise*), enquanto no SW-CMM elas ficavam diluídas nas demais áreas-chave.

A área-chave de Medição e Análise tem como proposta desenvolver e sustentar uma capacitação nas atividades de medição, que forneça informações úteis à gestão dos projetos. Por pertencer ao segundo nível de maturidade, o objetivo da mensuração aqui é simplesmente garantir que os projetos sejam devidamente controlados e acompanhados em relação aos compromissos

assumidos. Além de algumas metas genéricas, comuns a várias áreas-chave do CMMI, a área de Medição e Análise possui as seguintes metas específicas:

Meta	Descrição	Práticas associadas
Alinhar as atividades de medição e análise	Os objetivos e atividades relacionados à mensuração devem estar alinhados com as necessidades de informação e metas da organização.	• Estabelecer objetivos da mensuração; • Especificar as medidas; • Especificar os procedimentos de coleta e armazenamento dos dados; • Especificar procedimentos de análise.
Fornecer resultados das medições	Os resultados das medições são fornecidos para atender às necessidades de informação identificadas.	• Coletar as medidas; • Analisar os dados obtidos; • Armazenar os dados e os resultados; • Divulgar os resultados.

TABELA 2 - Metas específicas da área-chave de Medição e Análise do CMMI nível 2

É possível perceber semelhanças entre as metas dessa área-chave e a estrutura do PSM, apresentada na seção anterior. Podemos afirmar que a meta referente ao alinhamento das atividades de medição e análise corresponde à idéia do *modelo de informação* do PSM, e que a segunda meta corresponde ao conceito de *modelo de processo*. Na verdade, essa semelhança se deve ao fato do PSM e do padrão ISO/IEC 15939 terem sido utilizados como referências durante a definição dessa área-chave do CMMI (referências ao PSM e ao padrão citado são encontradas em [CMMI02]).

Além da área-chave de Medição e Análise existente no nível 2, o CMMI trata da mensuração com bastante ênfase no nível 4. Nesse nível, no entanto, o enfoque é diferente: como o problema do planejamento e controle de projetos já foi tratado nos estágios anteriores, o objetivo aqui é obter um conhecimento quantitativo da qualidade dos produtos e do desempenho dos processos da organização. As medições são usadas para colocar os processos sob um rigoroso controle estatístico, e os projetos são geridos em bases quantitativas. Duas áreas-chave compõem esse quarto nível do CMMI:

- *Desempenho dos Processos Organizacionais,* que trata do entendimento quantitativo dos processos;
- *Gestão Quantitativa de Projetos*, que trata da aplicação desse entendimento ao alcance dos objetivos quantitativos de desempenho e qualidade, por parte dos projetos.

A TABELA 3 apresenta as metas específicas e práticas associadas a cada uma dessas áreas-chave.

Área-chave	Meta	Descrição	Práticas associadas
Desempenho dos Processos Organizacionais	Estabelecer modelos e linhas de base de desempenho	São estabelecidos e mantidos modelos e linhas de base para caracterizar o desempenho esperado dos processos utilizados pela organização.	• Selecionar os processos e elementos a serem incluídos na análise de desempenho; • Estabelecer medidas de desempenho de processos; • Estabelecer objetivos de qualidade e de desempenho dos processos; • Estabelecer uma linha de base para o desempenho dos processos; • Estabelecer modelos de desempenho dos processos.
Gestão Quantitativa de Projetos	Gerir quantitativamente os projetos	Os projetos são geridos quantitativamente em relação a metas de qualidade e de desempenho dos processos.	• Estabelecer os objetivos do projeto quanto a qualidade e desempenho do processo; • Compor o processo a ser utilizado no projeto; • Selecionar os subprocessos a serem geridos estatisticamente; • Gerir o desempenho do projeto, identificando ações corretivas quando apropriado.
	Gerir estatisticamente o desempenho dos subprocessos	O desempenho de um conjunto selecionado de subprocessos internos ao processo utilizado em um projeto é gerido em bases estatísticas.	• Selecionar medidas e técnicas de análise; • Aplicar métodos estatísticos para compreender variações; • Controlar o desempenho dos subprocessos selecionados; • Registrar dados estatísticos no repositório de dados da organização.

TABELA 3 - Metas e práticas associadas às áreas-chave do CMMI nível 4

O conhecimento quantitativo dos processos e produtos, gerado a partir da execução dessas práticas, fornecerá a base para a implantação da melhoria contínua dos processos, que é o tema do nível 5 (estágio máximo de maturidade no CMMI).

Capítulo 3

Seleção das medidas

Como apresentado no Capítulo 2, um fator determinante do sucesso ou fracasso de um programa de mensuração é a adequação das medidas selecionadas aos objetivos que se pretende atingir com a medição. O controle quantitativo é uma prática de custo considerável, que só é justificável se trouxer benefícios concretos, com a geração de informações realmente úteis e confiáveis. A escolha do que deve ser mensurado ao longo do processo deve levar em conta os benefícios práticos que cada medição pode trazer.

Para guiar a seleção dos elementos que irão compor o modelo de medição, este trabalho seguiu a seqüência de passos proposta pelo *Goal-Driven Software Measurement*, apresentado no Capítulo 2. No entanto, como esse processo foi originalmente concebido para o contexto de uma organização específica interessada em implantar práticas de medição, algumas adaptações tiveram que ser realizadas para que seus princípios pudessem ser aplicados também no contexto deste trabalho. Foram realizadas três alterações no processo:

- No primeiro passo, onde são estabelecidas as metas de negócio, a estratégia utilizada foi a de escolher metas geralmente comuns a todas as organizações produtoras de software, já que o modelo proposto aqui pretende ser aplicável a diversas organizações.
- No quinto passo, onde são formalizadas as metas de medição, o processo original prevê que sejam fornecidas algumas descrições de ambiente, que tratam de detalhes técnicos sobre a organização e os projetos que serão submetidos à medição. Como este trabalho teve a preocupação de criar um modelo que seja independente de um ambiente específico, essas descrições foram omitidas.
- Os dois últimos passos tratariam do planejamento da implantação das práticas de medição na organização, através da identificação das ações necessárias e da confecção de um plano de implantação na organização. Novamente pelo caráter de genericidade deste trabalho, esses passos não foram realizados por estarem relacionados ao contexto de uma única organização.

As seções a seguir descrevem como os sete primeiros passos do *Goal-Driven Software Measurement* foram seguidos para selecionar os elementos e atributos que servirão como

base para o modelo de medição proposto no Capítulo 4:

- A seção 3.1 trata da definição das metas de negócio e submetas, e corresponde aos passos 1 a 4 do processo;
- A seção 3.2 apresenta a formalização das metas de medição, prevista no passo 5;
- A seção 3.3 descreve o sexto passo, que consiste na formulação de perguntas e indicadores;
- Os elementos de dados a serem coletados são identificados na seção 3.4, correspondendo ao passo 7 do *Goal-Driven Software Measurement*..

O oitavo passo, que trata da definição e padronização das medições a serem realizadas, foi tratado à parte devido a sua complexidade, e corresponde ao Capítulo 4 deste documento.

3.1 Definição das metas de negócio e submetas

A aplicação do *Goal-Driven Software Measurement* tem início com a definição de um conjunto de metas de negócio, cujo refinamento constituirá a base de todo o modelo de medição.

Como a proposta deste trabalho é criar um modelo genérico, que possa ser aplicado potencialmente a diversas organizações e processos, a definição dessas metas não pode ficar restrita ao contexto de uma organização específica. A estratégia utilizada foi a de selecionar metas geralmente comuns à maior parte das organizações, de forma a tornar o modelo potencialmente aplicável a grande parte delas. Essa estratégia teve também a intenção de tornar o modelo mais estável: mesmo para uma única organização, não convém congelar um modelo em torno de metas muito detalhadas, já que no mundo atual de negócios esse tipo de meta tende a mudar com freqüência.

Caso alguma organização interessada em aplicar o modelo tenha metas diferentes das postuladas aqui, a descrição do processo utilizado para a seleção das medidas, apresentada neste capítulo, pode servir como base para a realização das adaptações necessárias.

Como apresentado em [FLORAC97], todos os processos que envolvem o desenvolvimento de produtos possuem algumas características em comum. Para produzir determinados produtos, eles consomem tempo e recursos. Além disso, é esperado que esses produtos sejam obtidos no tempo esperado. Finalmente, os produtos desenvolvidos podem possuir defeitos ou imperfeições. Em razão dessas similaridades, há algumas questões que representam preocupações constantes na gerência dos processos:

- Qualidade dos produtos;
- Cumprimento de prazos para entrega dos produtos;
- Tempo ou duração do processo;
- Custo do processo;

As metas de negócio utilizadas como base para a construção do modelo de medição foram então baseadas nesses quatro fatores. São elas:

- Produzir software de qualidade;
- Obter boa produtividade;
- Cumprir os compromissos de prazo e custo assumidos.

Metas de negócio são objetivos estratégicos da organização, e precisam ser refinadas para que possam ser traduzidas em objetivos mensuráveis. O primeiro passo para isso, que constitui a segunda etapa do *Goal-Driven Software Measurement*, consiste na identificação do que se deseja aprender para que se possa compreender, prever ou melhorar as atividades relacionadas a cada uma das metas escolhidas.

A TABELA 4 mostra as questões levantadas para a primeira das três metas de negócio. Nelas, as perguntas encontram-se agrupadas por temas. Tal agrupamento já faz parte do terceiro passo do processo, que tem por objetivo transformar as metas de negócio em itens mais específicos, as *submetas*. Uma mesma pergunta pode aparecer em mais de um grupo, se estiver diretamente relacionada a mais de um tema.

Meta 1: Produzir software de qualidade	
Tema	**Questões**
Defeitos	• Quais os principais tipos de defeitos injetados e detectados ao longo do ciclo de desenvolvimento? • Em que iteração e em que fluxo do processo os defeitos estão sendo injetados nos artefatos? • Em que iteração e em que fluxo do processo os defeitos dos artefatos estão sendo detectados e corrigidos? • Os requisitos estão sendo especificados de forma correta?
Testes e revisões	• As atividades de revisão se mostram eficazes na detecção dos defeitos? • Está sendo disponibilizado esforço suficiente para as atividades de revisão? • Quais os tipos de defeitos detectados em atividades de revisão? • Está sendo disponibilizado esforço suficiente para as atividades de teste? • Os testes se mostram eficazes na detecção de defeitos? • Quais os tipos de defeitos detectados em atividades de testes?
Produto de software	• O produto final possui taxa de defeitos em nível aceitável? • Quais os principais tipos de defeitos encontrados no produto final?
Requisitos	• Os requisitos estão sendo especificados de forma correta? • Os requisitos especificados refletem as necessidades dos usuários?
Esforço	• Está sendo disponibilizado esforço suficiente para as atividades de revisão? • Está sendo disponibilizado esforço suficiente para as atividades de teste?

TABELA 4 - Questões identificadas para a meta de qualidade dos produtos

Em seguida, cada um dos temas sob os quais as perguntas foram agrupadas é traduzido em uma submeta de negócio, que detalha a meta geral "produzir software de qualidade". Obtém-se então as cinco submetas apresentadas na TABELA 5.

Meta 1: Produzir software de qualidade	
Submeta	**Descrição**
1.1	Minimizar o número de defeitos durante todo o processo de desenvolvimento.
1.2	Maximizar a eficácia das atividades de revisão e teste.
1.3	Maximizar a confiabilidade do produto final.
1.4	Maximizar o atendimento às necessidades do cliente.
1.5	Dimensionar adequadamente o esforço dedicado a atividades que visam melhorar a qualidade do produto.

TABELA 5 - Submetas obtidas para a meta de qualidade dos produtos

O objetivo dessas submetas é destacar os aspectos mais relevantes relacionados às metas de negócio escolhidas no início do processo, para que elas possam ser posteriormente traduzidas em objetivos mensuráveis.

As tabelas a seguir mostram as questões e submetas obtidas para as demais metas de negócio, relacionadas à produtividade e ao cumprimento de compromissos assumidos.

Meta 2: Obter boa produtividade	
Grupo	**Questões**
Defeitos	• Qual o impacto dos defeitos injetados em cada fluxo e cada iteração, no que diz respeito à produtividade? • Qual o esforço necessário para corrigir cada tipo de defeito?
Atividades	• Qual a produtividade da equipe em cada um dos fluxos do processo? • Qual a produtividade da equipe em cada iteração do processo? • Que passos do processo estão apresentando melhor produtividade? • Que passos do processo estão apresentando pior produtividade?
Estabilidade de requisitos	• Qual a taxa de ocorrência de mudanças em requisitos? • Qual o impacto das mudanças de requisitos na produtividade?
Produtos e artefatos	• Qual o tamanho do produto de software construído? • Qual foi o esforço total gasto para construí-lo? • Qual foi o tamanho de cada artefato produzido? • Qual o esforço gasto para construir cada artefato?
Atividades de revisão	• Qual o esforço investido em atividades de revisão (revisões técnicas, inspeções de código, etc)? • Qual o impacto dessas atividades na produtividade da equipe?

TABELA 6 - Questões identificadas para a meta de produtividade

Meta 2: Obter boa produtividade	
Submeta	**Descrição**
2.1	Minimizar o impacto dos defeitos na produtividade.
2.2	Maximizar a produtividade da equipe em cada passo do processo.
2.3	Minimizar o número e o impacto de mudanças de requisitos na produtividade.
2.4	Conhecer o tamanho de cada produto e artefato já produzido, e o esforço necessário para produzi-los.
2.5	Maximizar os ganhos de produtividade obtidos com as revisões.

TABELA 7 - Submetas obtidas para a meta de produtividade

Meta 3: Cumprir os compromissos de prazo e custo assumidos	
Grupo	**Questões**
Estimativas de esforço	• O esforço estimado para cada fluxo ao longo do ciclo de desenvolvimento é adequado? • A estimativa da distribuição do esforço entre as iterações do processo de desenvolvimento está adequada? • Qual o grau de confiabilidade das estimativas produzidas? • O esforço total consumido no desenvolvimento dos produtos está dentro dos limites estimados?
Planejamento de projeto	• O planejamento dos prazos e do esforço está adequado, tendo em vista as características da organização e dos produtos a serem desenvolvidos? • O número de pessoas alocadas ao projeto é adequado tendo em vista os compromissos assumidos? • O perfil dessas pessoas é adequado? • A quantidade de esforço disponibilizado para os projetos é suficiente para concluir o desenvolvimento dos produtos dentro dos prazos assumidos?
Acompanhamento de projeto	• O andamento de cada projeto tende ao cumprimento dos compromissos de prazo e custo? • Os produtos estão sendo entregues nos prazos acordados com os clientes? • Todos os artefatos produzidos foram concluídos nos prazos previstos? • O esforço total consumido no desenvolvimento dos produtos está dentro dos limites estimados?

TABELA 8 - Questões identificadas para a meta de cumprimento dos compromissos

Meta 3: Cumprir os compromissos de prazo e custo assumidos	
Submeta	**Descrição**
3.1	Maximizar a acurácia das estimativas de esforço.
3.2	Maximizar a adequação dos planejamentos dos projetos, tendo em vista as características de cada projeto e da equipe de desenvolvimento.
3.3	Controlar precisamente o andamento de cada projeto, tendo em vista o cumprimento dos objetivos assumidos.

TABELA 9 - Submetas obtidas para a meta de cumprimento dos compromissos

Uma vez obtida a lista das submetas, juntamente com as questões relacionadas a cada uma, o próximo passo do *Goal-Driven Software Measurement* consiste em identificar as entidades e atributos envolvidos. Isso servirá de base para a formalização das metas de medição, descrita na seção seguinte.

Para cada questão relacionada às submetas identificadas no passo anterior, devem ser

primeiramente identificadas as entidades nela implícitas. Em seguida, são listados os atributos associados a cada entidade. O objetivo é listar atributos que, se quantificados, contribuam para a obtenção de respostas às questões formuladas ou estabeleçam o contexto para a interpretação de tais respostas. A lista de entidades e atributos obtidos é relativamente extensa, e por esse motivo não será apresentada aqui. A relação completa foi inserida no apêndice A.1, ao final deste trabalho.

3.2 Formalização das metas de medição

Nos passos anteriores, o principal objetivo foi identificar as metas de negócio e os principais aspectos que interferem no alcance dessas metas. No quinto passo, as informações levantadas até então são utilizadas para formalizar as metas do processo de mensuração, dando início à estrutura de metas, perguntas e medidas que forma a base do paradigma GQ(I)M descrito na seção 2.2.2.

Essa formalização consiste na especificação de quatro componentes:

1. Um *objeto de interesse*, que pode ser um produto, processo, recurso, agente, artefato, medida ou ambiente. Também pode ser um conjunto de outras entidades ou objetos.
2. Uma *proposta*, que pode ser compreender, prever, planejar, controlar, comparar ou melhorar algum aspecto do objeto.
3. Uma *perspectiva*, que identifica quem está interessado nos resultados das medições (desenvolvedor, gerente de projeto, cliente, etc.).
4. Uma descrição do *ambiente*, que fornece um contexto para que os resultados das medições possam ser interpretados. Normalmente refere-se a aspectos do projeto que está sendo medido.

Considerando as metas de negócio, submetas, questões, entidades e atributos identificados nos passos anteriores, foram obtidas as 11 metas de medição apresentadas na TABELA 10. A tabela apresenta a formalização das metas em termos do objeto de interesse, proposta e perspectiva. Como foi mencionado na introdução deste capítulo, a descrição do ambiente foi omitida por estar voltada para situações em que a política de medição tem como foco uma organização específica, o que a torna incompatível com a natureza deste trabalho.

Para permitir uma maior rastreabilidade, a última coluna da tabela mostra a relação de submetas às quais cada meta de medição está associada.

Nº	Objeto de interesse	Proposta	Perspectiva	Submetas associadas
1	Defeitos	Analisar a taxa de defeitos detectados no produto como forma de aferir sua qualidade.	Avaliar quantitativamente a qualidade do produto, pelo grupo de garantia da qualidade.	1.1 1.3 1.4
2	Defeitos	Analisar a distribuição dos defeitos de naturezas diversas, injetados e detectados nos diferentes fluxos e iterações, com o objetivo de identificar oportunidades de melhoria do processo quanto à redução de defeitos.	Formar uma base de dados históricos para as atividades de melhoria de processo, identificando os pontos mais problemáticos quanto à inserção e detecção de defeitos ao longo do processo.	1.1 1.3
3	Defeitos	Analisar o esforço gasto na correção de cada defeito, tendo em vista sua natureza, complexidade, e as iterações e fluxos em que foram injetados e corrigidos, com o objetivo de conhecer o impacto da correção de defeitos na produtividade.	Fornecer uma base que permita aos gerentes de projeto conhecerem o impacto de defeitos na produtividade, e ao grupo de melhoria de processos propor melhorias para minimizar tal impacto.	2.1
4	Esforço	Analisar o esforço gasto em cada iteração e fluxo do processo, com o objetivo de conhecê-lo e identificar formas minimizá-lo.	Fornecer embasamento quantitativo para a elaboração de estimativas pelo gerente de projeto, e para a proposta de melhorias pelo grupo de melhoria de processos.	2.2 2.4 3.1 3.2 3.3
5	Tamanho e Esforço	Conhecer o tamanho de cada produto desenvolvido e o esforço necessário para produzi-lo, com o objetivo de manter uma base de dados históricos sobre produtividade e permitir a normalização dos demais dados coletados.	Fornecer embasamento quantitativo para a realização de estimativas pelos gerentes de projeto e permitir ao grupo de engenharia de processos normalizar os demais dados coletados, possibilitando a comparação de dados entre diferentes projetos.	1.5 2.4 3.1 3.2
6	Testes	Avaliar as atividades de teste quanto ao esforço dedicado e ao rendimento obtido na detecção de defeitos, com o objetivo de identificar formas de aumentar sua eficácia.	Fornecer base quantitativa para que o grupo de melhoria de processos possa propor melhorias nas atividades de teste.	1.2 1.3 1.5
7	Revisões	Avaliar as revisões quanto ao esforço dedicado e ao rendimento obtido na detecção de defeitos, com o objetivo de identificar formas de aumentar sua eficácia.	Fornecer base quantitativa para que o grupo de melhoria de processos possa propor melhorias nas atividades de revisão.	1.2 1.3 1.5 2.5
8	Requisitos	Avaliar a freqüência de alterações de requisitos ocorridas, com o objetivo de minimizá-las em projetos futuros e prever o impacto de cada uma nos projetos em andamento.	Fornecer base quantitativa para que o grupo de melhoria de processos possa propor formas de minimizar as alterações de requisitos decorrentes de deficiências no processo de especificação. Permitir ao gerente de projeto planejar o impacto dessas alterações nos prazos e custos dos projetos.	1.4 2.3
9	Requisitos	Avaliar o esforço gasto na adequação de produtos e artefatos em decorrência de alterações de requisitos, com o objetivo de conhecer o impacto dessas mudanças no esforço dedicado aos projetos.	Fornecer uma base que permita aos gerentes de projeto conhecerem o impacto de alterações de requisitos no esforço dedicado aos projetos, e ao grupo de melhoria de processos propor melhorias para minimizar tal impacto.	2.3
10	Estimativas	Analisar o erro das estimativas com o objetivo de conhecer sua acurácia e identificar oportunidades para melhorá-la.	Permitir um melhor conhecimento dos erros das estimativas pelos gerentes de projetos, possibilitando o dimensionamento dos fatores de contingência, e monitorar a previsibilidade do processo de desenvolvimento pelo grupo de melhoria de processos.	3.1 3.2
11	Acompanha-mento de projetos	Analisar a fração já concluída dos projetos, com o objetivo de compará-la com as estimativas feitas e com os compromissos assumidos.	Prover aos gerentes de projeto um controle quantitativo sobre o andamento dos projetos.	3.3

TABELA 10 - Metas de medição estipuladas

3.3 Formulação de perguntas e indicadores

Agora que as metas de medição já foram formalizadas, já se tem uma base para que o paradigma *Goal-Question-Indicator-Metric* – ou GQ(I)M – possa ser utilizado. Definidas as metas de medição, o objetivo agora é formular questões *quantitativas* e criar indicadores que dêem suporte a essas questões.

Por exemplo, para a primeira meta de medição, referente à análise da taxa de defeitos, as seguintes perguntas foram elaboradas:

- Qual a densidade de defeitos observada em cada produto desenvolvido?
- Qual tem sido a evolução da densidade de defeitos ao longo dos projetos de desenvolvimento?

Tais perguntas poderiam ser respondidas mais facilmente se houvesse um indicador gráfico como o apresentado na Figura 8, que exibe a evolução da densidade de defeitos (medida em defeitos/ponto de função) ao longo dos projetos de desenvolvimento realizados pela organização.

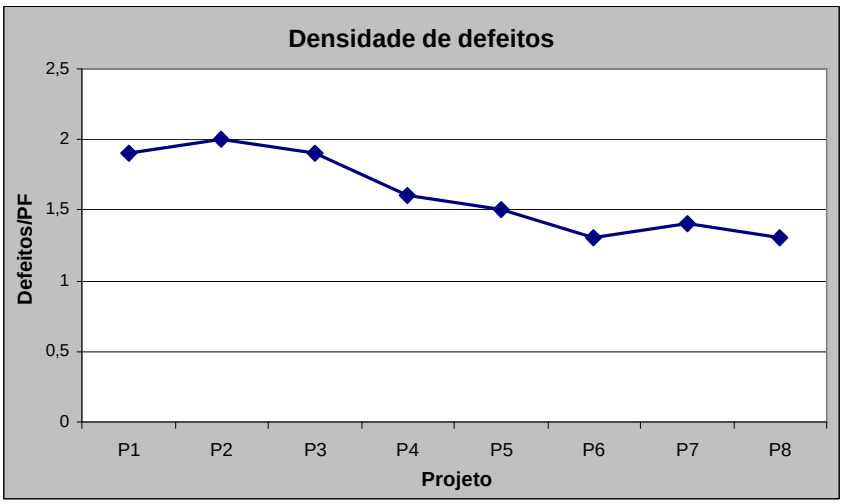


FIGURA 8 - Indicador criado para a meta de medição referente ao acompanhamento da taxa de defeitos

Indicadores como esse podem ser criados para todas as metas de medição. A TABELA 11 exibe as questões elaboradas para cada uma das metas enumeradas na seção anterior, e associa a cada grupo de questões um conjunto de indicadores. Nela, cada indicador é apenas referenciado por seu código; sua descrição completa encontra-se no apêndice A.2, ao final deste trabalho.

Meta	Perguntas	Indicadores
1	• Qual a densidade de defeitos observada em cada produto desenvolvido? • Qual tem sido a evolução da densidade de defeitos ao longo dos projetos de desenvolvimento?	I-4
2	• Quais os tipos de defeitos mais comuns em cada fluxo e iteração? • Qual a distribuição dos defeitos em relação à iteração em que foram injetados? • Qual a distribuição dos defeitos em relação ao fluxo em que foram injetados? • Qual a distribuição dos defeitos em relação à iteração em que foram detectados? • Qual a distribuição dos defeitos em relação ao fluxo em que foram detectados? • Que iterações estão gerando número maior de defeitos? Quando esses defeitos estão sendo removidos?	I-1 I-2 I-3
3	• Qual o esforço médio para corrigir cada tipo de defeito? • Qual o esforço médio para corrigir defeitos injetados em cada fluxo? • Qual o esforço médio para corrigir defeitos injetados em cada iteração? • Qual o esforço médio para corrigir defeitos em função da iteração em que a correção foi realizada?	I-5 I-9
4	• Qual o esforço gasto em cada iteração do processo? • Qual o esforço gasto em cada fluxo do processo?	I-8
5	• Qual a produtividade observada em cada projeto de desenvolvimento? • Qual a evolução da produtividade ao longo do tempo? • Qual o tamanho de cada produto desenvolvido?	I-7
6	• Qual a fração dos defeitos que corresponde aos detectados em atividades de teste? • Quais os tipos de defeitos encontrados e quais os não encontrados? • Qual o esforço dedicado às atividades de teste?	I-2 I-1 I-8
7	• Qual a fração dos defeitos que corresponde aos detectados em atividades de revisão? • Qual a taxa de detecção de defeitos nessas atividades? • Quais os tipos de defeitos encontrados e quais os não encontrados? • Qual a fração do esforço de um projeto que é empregada nas atividades de revisão?	I-13 I-12 I-1 I-14
8	• Qual a fração dos requisitos que sofre alterações após a conclusão da especificação de requisitos?	I-10
9	• Qual o esforço médio (por unidade de tamanho) gasto para adequar produtos e artefatos a cada mudança de requisitos? • Como varia esse esforço de acordo com a iteração em que as alterações ocorreram?	I-11 I-9
10	• Qual o erro médio das estimativas (esforço, prazo, defeitos, revisões, estabilidade de requisitos, tamanho) realizadas no planejamento dos projetos, comparando-se os valores estimados e obtidos? • Qual a evolução dos erros a cada projeto?	I-17
11	• Qual a fração já concluída do projeto? • Qual a fração prevista no planejamento realizado? • Qual o esforço já gasto até então? • Qual o esforço previsto no planejamento realizado? • A quantidade de defeitos está sob controle, ou representa algum risco para a conclusão tempestiva do projeto?	I-15 I-16 I-6 I-5

TABELA 11 - Perguntas e indicadores criados para cada uma das metas de medição

A partir do esboço dos indicadores, é possível definir quais as medidas que devem ser coletadas para possibilitar a construção de cada um. É esse o tema do próximo passo do processo.

3.4 Identificação dos elementos de dados

O sétimo passo do *Goal-Driven Software Measurement* consiste na confecção de uma lista enumerando todos os dados necessários à construção dos indicadores sugeridos no passo anterior.

Por exemplo, para construir o indicador I-4 exemplificado na seção anterior (Figura 8), considerando as suas possíveis variações (detalhadas no apêndice A.2), é necessário coletar as seguintes informações sobre cada defeito detectado nos projetos:

- Projeto ao qual pertence o defeito;
- Classificação do defeito quanto à espécie, caso se deseje considerar no indicador apenas defeitos de determinada espécie (o conceito de espécies de defeito é aprofundado na seção 4.2.2);
- Iteração em que o defeito foi injetado, caso seja desejável considerar apenas os defeitos injetados em uma determinada iteração;
- Fluxo em que o defeito foi injetado, caso seja desejável considerar apenas os defeitos injetados em um determinado fluxo.

Além disso, é necessário coletar informações sobre o tamanho de cada produto desenvolvido para que seja possível calcular a densidade desses defeitos.

Fazendo o mesmo para cada um dos indicadores criados, obtém-se ao final o conjunto completo dos elementos de dados que devem ser coletados ao longo do processo de desenvolvimento. Neste ponto, graças ao trabalho feito nos passos anteriores, pode-se afirmar que cada um desses elementos corresponde a pelo menos uma proposta objetiva de medição, que por sua vez está alinhada com as metas de negócio da organização. É esse o principal benefício trazido pelo *Goal-Driven Software Measurement*: o rastreamento de cada um dos elementos de dados em relação às metas que motivaram sua coleta.

A TABELA 12 apresenta todos os elementos de dados que são necessários para a construção dos indicadores enumerados na seção anterior.

Entidade	Elemento de dado a ser coletado	Indicadores relacionados
Cada projeto realizado	Número de horas trabalhadas diariamente em cada fluxo de cada projeto, por cada membro da equipe.	I-7; I-8; I-14; I-16
	Tamanho final do produto desenvolvido	I-4; I-7; I-10
	Tamanho de cada artefato produzido	I-7
	Relação dos requisitos do produto	I-10
Cada defeito detectado	Projeto ao qual pertence o defeito	I-1; I-2; I-3; I-4; I-5; I-6; I-9; I-13
	Data em que foi detectado	I-1; I-2; I-3; I-6
	Data em que foi corrigido	I-6; I-3
	Iteração em que foi injetado	I-1; I-2; I-4
	Esforço empregado na correção	I-5; I-9
	Classificação quanto à espécie	I-1; I-2; I-3; I-4; I-5; I-6; I-12; I-13
	Fluxo em que foi injetado	I-1; I-2; I-4; I-5; I-13
	Fluxo em que foi detectado	I-1; I-2
Cada revisão realizada	Projeto ao qual pertence	I-12; I-13; I-14
	Data de conclusão da revisão	I-14
	Método de verificação utilizado	I-12; I-13; I-14
	Esforço total empregado na revisão	I-12; I-14
	Fluxo ao qual pertence o material revisado	I-12; I-14
	Relação dos defeitos detectados	I-1; I-12; I-13
Cada alteração de requisito solicitada	Projeto ao qual pertence	I-9; I-10; I-11
	Data da solicitação de alteração	I-11
	Tamanho da alteração	I-10; I-11
	Esforço total dedicado	I-9; I-11
	Relação dos requisitos alterados	I-10
Cada marco de projeto atingido	Projeto ao qual se refere	I-15
	Data do marco	I-15
	Tamanho de cada requisito	I-15
	Estado de cada requisito	I-15
Cada planejamento de projeto realizado	Projeto ao qual se refere	I-7; I-8; I-9; I-16; I-17
	Tamanho estimado para o produto final	I-7; I-17
	Esforço previsto para cada fluxo, em cada iteração	I-7; I-8; I-9; I-16; I-17
	Tamanho previsto de cada artefato	I-7; I-17
	Número previsto de defeitos injetados em cada fluxo, para cada iteração	I-17
	Percentual estimado do esforço dedicado a correção de defeitos	I-17
	Esforço previsto para as atividades de revisão, dividido por tipo de revisão, fluxo e iteração	I-17
	Fração estimada dos defeitos detectados em revisões	I-17
	Percentual estimado de requisitos alterados, em termos do número de requisitos	I-17
	Percentual estimado de requisitos alterados, em termos do tamanho das alterações	I-17
	Esforço total previsto para alterações de requisitos	I-9; I-17
Cada marco de projeto previsto em um planejamento	Planejamento ao qual pertence	I-15; I-16
	Data do marco	I-15; I-16
	Tamanho de cada requisito	I-15
	Estado de cada requisito	I-15

TABELA 12 - Elementos de dados identificados

Para cada elemento de dado, a TABELA 12 exibe também a relação dos indicadores para os quais contribui. Além de proporcionar para uma melhor rastreabilidade, essa informação pode ser utilizada para que se estabeleça priorizações. Isso pode ser útil em situações em que o programa de medição for implantado gradativamente em uma organização: pode-se, por exemplo, começar a implantação pelos elementos de dados que contribuem para um subconjunto inicial de indicadores, e ir posteriormente expandindo até que se obtenha o conjunto completo.

O próximo capítulo descreve como esses elementos foram formalizados e transformados em medidas quantitativas, fornecendo a base para a criação do modelo de medição.

Capítulo 4

Confecção do modelo de medição

Feita a seleção dos atributos do processo a serem mensurados, é necessário definir melhor cada elemento a ser medido, a fim de tornar tais medidas aplicáveis operacionalmente. A preocupação com a formalização das medidas tem o objetivo de garantir duas propriedades necessárias a qualquer política de medição [CARLETON92]:

- **Comunicação**: De posse dos dados coletados, outras pessoas devem ser capazes de saber o que foi medido, como isso foi feito, e o que foi incluído e excluído nas medições realizadas;
- **"Repetibilidade"**: De posse das definições feitas, outras pessoas devem ser capazes de realizar as mesmas medições e obter essencialmente os mesmos resultados.

Se a definição das medidas não garantir a comunicação, a interpretação dos resultados torna-se inviável, pois é necessário que os critérios de medição estejam bastante claros para se obter qualquer conclusão prática. Se, por outro lado, a repetibilidade não for garantida, o valor coletado nas mensurações fica dependente da interpretação do responsável pelo seu registro, o que compromete seriamente sua confiabilidade.

Para garantir que essas propriedades sejam atingidas, um modelo de medição deve detalhar procedimentos e convenções para cada mensuração a ser realizada. Esse detalhamento deve conter os seguintes elementos:

- Esclarecimento do que deve ser incluído e o que deve ser deixado de fora das medições realizadas – mesmo para medidas aparentemente óbvias, uma definição imprecisa do que deve ser levado em conta na medição pode causar problemas. Um exemplo é a contagem do número de linhas de código-fonte de um produto, como forma de mensurar seu tamanho. É necessário estabelecer regras como a inclusão ou omissão das linhas em branco e comentários existentes no código; caso contrário, interpretações diferentes podem levar a medições diferentes, o que compromete a confiabilidade dos resultados obtidos.
- Definição precisa de todos os atributos a serem mensurados – uma imprecisão na

definição de um atributo também pode levar a diferentes interpretações e comprometer as medições realizadas. Um exemplo seria a medição do número de defeitos detectados em um produto, que constitui uma das práticas de mensuração mais comuns. O que exatamente deve-se entender por um *defeito*? Apenas os defeitos que comprometam a execução do produto, ou defeitos de documentação também devem ser considerados? E problemas de desempenho ou usabilidade, devem ser considerados como sendo defeitos?

- Escolha das escalas e unidades de medida a serem utilizadas em cada caso – em vários casos, os valores registrados para determinado atributo só trazem informação útil se vierem acompanhados das respectivas escalas e unidades de medida. É comum um mesmo atributo possuir diferentes opções de unidades de medida: a medição do esforço dedicado a determinado grupo de atividades, por exemplo, pode ser feita em horas trabalhadas, dias de trabalho, pessoas-mês, entre outros. Deve-se convencionar uma única unidade de medida para cada atributo a ser medido, visando facilitar as atividades de análise e evitar erros de conversão.
- Padronização da forma de registro dos resultados mensurados – tão importante quanto estabelecer as regras para a realização das medições é definir o formato de registro dos resultados obtidos. Para isso, a prática mais comum é a criação de formulários a serem preenchidos pelos responsáveis por cada coleta. Tais formulários podem ser confeccionados em papel ou eletronicamente (através de planilhas eletrônicas ou mesmo interfaces de sistemas informatizados específicos para o suporte ao processo de mensuração).

A definição do modelo proposto neste trabalho é apresentada em três seções. A seção 4.1 formaliza as medidas a serem coletadas, definindo convenções, unidades de medida e relatórios padronizados para o registro dos dados. Na medida em que são formalizadas, as medidas são agrupadas em um modelo conceitual único que detalha as entidades envolvidas e o relacionamento existente entre elas, através de diagramas de classes em notação UML [Rumbaugh99]. A seção 4.2 detalha os elementos do modelo que são configuráveis, para permitir que o mesmo seja aplicável a diferentes processos e organizações. Finalmente, a seção 4.3 mostra como diferentes atributos do modelo podem ser combinados entre si para formar medidas derivadas, capazes de extrair da massa de dados algumas informações úteis para a melhoria dos processos e a gestão dos projetos. São essas medidas derivadas que irão fornecer as informações necessárias para a construção dos indicadores gráficos e tabelas propostos durante a execução do *Goal-Driven Software Measurement*, descrita no Capítulo 3.

4.1 Formalização das medidas básicas

Esta seção formaliza cada um dos elementos de dados identificados ao final do processo de seleção das medidas. Para uma melhor organização, os elementos foram divididos em seis grupos:

- Medidas de tamanho;
- Medidas de estabilidade de requisitos;
- Medidas de esforço;
- Medidas de progresso e cronograma;
- Medidas de defeitos;
- Medidas de revisões.

As seções a seguir detalham cada um desses grupos. Em seguida, é definida também a forma em que devem ser registradas as estimativas produzidas durante os planejamentos dos projetos.

Cada grupo de medidas possui um diagrama de classes em notação UML que define uma estrutura para a organização das informações coletadas e exibe os relacionamentos entre as entidades envolvidas. Nesses diagramas, algumas das classes são apresentadas em uma cor diferenciada (cinza). Trata-se de classes que modelam elementos configuráveis do modelo, que devem ser instanciados por cada organização que for aplicá-lo (esses elementos serão posteriormente detalhados, na seção 4.2). Os diagramas apresentados vão, pouco a pouco, compondo o modelo conceitual único apresentado na seção 4.1.8.

4.1.1 Medidas de tamanho

A medição do tamanho dos produtos desenvolvidos é essencial para qualquer programa de mensuração, por vários motivos. Como o tamanho de um produto geralmente apresenta uma correlação com o esforço necessário para produzi-lo [ALBRECHT83], sua medição é utilizada diretamente em atividades de planejamento e acompanhamento, e na realização de estimativas diversas. São importantes também para normalizar outros indicadores, permitindo a comparação entre dados de diferentes projetos, além de serem indispensáveis para o cálculo de várias medidas derivadas, como produtividade.

Uma boa medida de tamanho deve atender aos seguintes critérios [PAULA03]:

- Ser contável através de um procedimento bem-definido;
- Ser calculável a partir da informação contida em uma especificação de requisitos de software;

- Apresentar boa correlação com o esforço de desenvolvimento.

A existência de procedimentos de medição bem definidos contribui para a confiabilidade das informações, já que a padronização procura evitar que o resultado final seja influenciado por decisões tomadas individualmente pelos autores das contagens.

A possibilidade de contagem a partir das informações contidas em um documento de especificação de requisitos é importante para que o tamanho seja conhecido desde as fases iniciais do ciclo de desenvolvimento, fornecendo assim subsídios para o planejamento e o acompanhamento dos projetos.

Finalmente, a correlação com o esforço de desenvolvimento é imprescindível para que as medidas de tamanho possam ser utilizadas como indicadores do esforço necessário à confecção de um produto.

4.1.1.1 Linhas de código x Pontos de função

As duas medidas de tamanho mais utilizadas são a contagem de **linhas de código** e a análise de **pontos de função**.

O número de linhas de código constitui um dos indicadores de tamanho mais tradicionais, sendo utilizado há décadas pela indústria de software. É obtido através da contagem do total de instruções presentes no código-fonte de um produto. Naturalmente, critérios devem ser estabelecidos para padronizar essa contagem. Recomendações para a definição desses critérios são apresentadas em [PARK92] e [HUMPHREY95].

A contagem de linhas de código possui algumas vantagens em relação a outros indicadores de tamanho:

- Simplicidade da coleta, que pode ser facilmente automatizada;
- Facilidade de compreensão, por ser baseada em elementos concretos (instruções do código-fonte);
- Boa correlação com o esforço de desenvolvimento [DOD00].

Por esses motivos, o número de linhas de código foi a medida de tamanho recomendada pelo IEEE para subsidiar o cálculo de produtividade [IEEE92]. No entanto, essa medida apresenta também problemas que merecem ser enumerados:

- É dependente da linguagem de programação utilizada, não podendo ser usado diretamente para comparar produtos desenvolvidos em linguagens diferentes.
- O número real de linhas de código de um produto só pode ser obtido com precisão após a conclusão do projeto e, portanto, tem pouco valor preditivo.

Outra alternativa para a mensuração do tamanho de um produto é a análise de Pontos de Função [ALBRECHT83]. Esse método avalia o tamanho de um produto de software a partir da complexidade de suas interfaces, através de regras padronizadas de contagem. Trata-se de uma

medida *funcional*, por ser baseada em funções e informações contempladas pelo produto, sob o ponto de vista do usuário. Podemos afirmar que essa forma de contagem está relacionada aos requisitos do produto, e não à forma pela qual o mesmo é construído.

Na literatura é possível encontrar críticas á contagem de pontos de função. [KITCHENHAM95], por exemplo, afirma que a técnica de contagem viola princípios teóricos de mensuração. Durante a contagem, cada função de dado ou de transação recebe um valor quanto à complexidade (*simples*, *médio* ou *complexa*). Dependendo da complexidade, esses elementos recebem pesos diferentes, que são então somados. A rigor, como a complexidade é representada em uma escala ordinal (segundo a nomenclatura definida na seção 2.1), esses valores não poderiam ser somados. No entanto, há estudos que comprovam a validade da utilização de pontos de função como indicadores do tamanho de um produto e como estimadores do esforço necessário para produzi-lo [ALBRECHT83]. Além disso, é uma medida que apresenta vantagens em relação à contagem de linhas de código:

- É independente da linguagem de programação utilizada, podendo ser usada para comparar projetos desenvolvidos com tecnologias diferentes;
- Por estar relacionada aos requisitos do produto do ponto de vista do usuário, pode ser mensurada nas fases iniciais do projeto, a partir de uma especificação de requisitos.

A principal desvantagem é a dificuldade de automatização, pelo fato de alguns passos exigirem um certo grau de interpretação por parte dos responsáveis pela contagem. Também por esse motivo, a contagem de pontos de função deve seguir rigorosamente um conjunto de regras padronizadas, como as apresentadas em [GARMUS01], [DREGER89] e nos manuais de contagem do IFPUG – International Function Point Users Group [IFPUG99]. Por se tratarem de regras consideravelmente complexas e já tratadas exaustivamente pela literatura, elas não serão apresentadas aqui. Detalhes sobre o assunto podem ser encontrados também em [LONGSTREET02] e [PAULA03].

No modelo de medição proposto neste trabalho, como as medidas de tamanho são utilizadas para o acompanhamento de projetos desde as fases iniciais, os pontos de função constituem a opção mais adequada. No entanto, a contagem de linhas de código pode ser feita opcionalmente, como será apresentado em seguida.

4.1.1.2 Registro de tamanho

O modelo de medição proposto neste trabalho utiliza duas formas de contagem de tamanho. Uma, obrigatória, é a contagem dos pontos de função. Outra, opcional, é a contagem do tamanho de alguns artefatos produzidos, utilizando critérios de medida pré-definidos.

A contagem de pontos de função será adotada como base para a confecção de estimativas, normalização das medidas dependentes de tamanho e acompanhamento da evolução dos projetos. Para permitir essa última aplicação, é necessário registrar o tamanho dos produtos

com uma granulosidade mais fina que a simples informação do seu total de pontos de função. A cada estágio dos projetos de desenvolvimento deve ser possível medir a fração do produto representada pelas funcionalidades já concluídas, de forma a prover um conhecimento preciso do progresso obtido até então. Para possibilitar esse nível de controle, o tamanho em pontos de função será contabilizado e registrado por *requisito*.

Para isso, primeiramente os requisitos do produto devem ser enumerados. Os processos de desenvolvimento geralmente prevêem o cadastramento de todos os requisitos em um artefato específico para esse fim. A relação desses requisitos pode ser transportada ao formulário para registro de marcos de projeto, apresentado na FIGURA 9, onde será registrado o tamanho de cada um. O registro de tamanho de cada requisito deve ser feito considerando o número de pontos de função brutos (não ajustados) [IFPUG99] por ele representado. Além disso, deve-se informar o *fator de ajuste* a ser utilizado, que consiste em um valor entre 0,65 e 1,35 utilizado para ajustar o número de pontos de função com base em um conjunto de características gerais do produto. O somatório do tamanho de todos os requisitos (que corresponde ao total de pontos de função brutos), multiplicado pelo fator de ajuste informado, determina o total de pontos de função ajustados do produto.

Requisitos não funcionais, como aqueles relacionados a desempenho e usabilidade, apesar de contribuírem para o cálculo do fator de ajuste, não são contados individualmente na análise de pontos de função. Esse tipo de requisito, quando cadastrado, deve receber tamanho zero.

É importante ressaltar que o mapeamento das funções contadas no processo de análise de pontos de função (entradas, saídas e consultas externas, arquivos lógicos internos e arquivos lógicos externos) para os requisitos do produto pode não ser direto, e dependerá dos critérios utilizados para o cadastramento desses requisitos. [PAULA03] apresenta um exemplo de mapeamento para o cadastro de requisitos previsto no processo Praxis.

Uma alternativa possível para simplificar esse mapeamento é considerar, para fins de acompanhamento de progresso, os requisitos como sendo exatamente as funções identificadas no processo de contagem de pontos de função. A decisão da abordagem mais adequada fica a critério da organização que for implementar as atividades de mensuração.

Em um projeto de desenvolvimento, o primeiro registro de tamanho deverá ser feito no momento em que for concluída a definição dos requisitos do produto. No entanto, no decorrer do projeto esses requisitos podem sofrer alterações em conseqüência de procedimentos de gestão de requisitos (por exemplo, alterações de requisitos solicitadas pelos usuários, ou correção de defeitos na especificação de requisitos). Para contemplar essa possibilidade, o registro do tamanho deverá ser feito em diferentes pontos do projeto, que aqui serão denominados *marcos de projeto* (o conceito de marcos será aprofundado na seção 4.1.4).

Para cada marco de projeto, deve-se registrar o tamanho em pontos de função de cada um dos requisitos. O registro é feito em um formulário específico, apresentado na FIGURA 9. Além

do campo *Tamanho* existente para cada requisito (que aparece destacado nesta visão do formulário), esse formulário apresenta diversos outros campos que serão detalhados na seção 4.1.4, por estarem mais relacionados ao acompanhamento do progresso do que à contagem de tamanho.

Formulário para registro de marco de projeto

Organizações ProSoft LTDA

Nome do projeto	*Quantum 1.0*
Nome do marco de projeto	*Fim da iteração LR*
Data do marco	*20/07/2002*
Natureza dos dados	*Reais*
Identificador do planejamento	*-*

Situação dos requisitos

Requisito	Descrição	Tamanho	Estado	
CUA1	*Gestão do registro de esforço*	*42*	*Levantado*	*20%*
CUA2	*Gestão do cadastro de projetos*	*45*	*Levantado*	*20%*
CUA3	*Gestão de usuários*	*19*	*Levantado*	*20%*
CUA4	*Gestão do registro de defeitos*	*40*	*Levantado*	*20%*
CUA5	*Gestão do registro de revisões*	*35*	*Levantado*	*20%*
CUA6	*Gestão do cadastro de requisitos*	*20*	*Levantado*	*20%*
CUA7	*Gestão do cadastro de planejamentos*	*25*	*Levantado*	*20%*
CUA8	*Gestão de cronograma de projeto*	*24*	*Levantado*	*20%*
CUA9	*Registro de marco de projeto*	*27*	*Levantado*	*20%*
CUA10	*Registro de métricas de artefatos*	*21*	*Levantado*	*20%*
CUA11	*Gestão de alterações em requisitos*	*26*	*Levantado*	*20%*
CUA12	*Registro de estimativa de esforço*	*18*	*Levantado*	*20%*
CUA13	*Registro de estimativa de defeitos*	*11*	*Levantado*	*20%*
CUA14	*Registro de estimativa de revisões*	*11*	*Levantado*	*20%*
CUA15	*Registro de estimativa de estabilidade de requisitos*	*7*	*Levantado*	*20%*
CUA16	*Gestão do cadastro de processos*	*22*	*Levantado*	*20%*
CUA17	*Gestão do cadastro de classificações de membros da equipe*	*19*	*Levantado*	*20%*

Tamanho total (PF brutos)	*412*
Fator de ajuste	*0,932*
Tamanho total (PF ajustados)	*383,9*

FIGURA 9 - Formulário para registro de marco de projeto, destacando a medição do tamanho

Na FIGURA 10 é apresentado o diagrama de classes utilizado no modelo de medição para representar os dados informados no formulário. A classe *Situação de requisito* vincula os *Requisitos* aos respectivos *Estados*, e o conjunto formado pelas situações de todos os requisitos compõe um *Marco de projeto*.

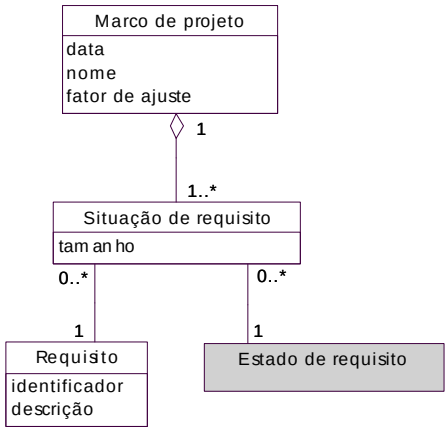


FIGURA 10 - Visão do registro de marco de projeto, sob a forma de um diagrama de classes

Além da contagem de pontos de função, outros indicadores de tamanho poderão ser opcionalmente coletados. Trata-se de medidas relacionadas a artefatos específicos do processo, que podem ser usadas como um suporte adicional à contagem de pontos de função. Apesar de não ser obrigatória, a coleta dessas medidas é fortemente recomendada por trazer dois benefícios principais:

- Fornece informações úteis sobre os artefatos produzidos permitindo, por exemplo, a comparação do tamanho de artefatos produzidos em diferentes projetos.
- Os valores medidos podem ser confrontados com o tamanho medido em pontos de função, para verificar se existe uma correlação e validar os pontos de função como indicadores de tamanho.

Exemplos de artefatos mensuráveis são documentos produzidos ao longo do projeto e o próprio código-fonte do produto. No momento da instanciação do modelo de medição a um processo de desenvolvimento deve-se estabelecer quais serão os artefatos mensurados, e quais os critérios de medição para cada um. Uma fonte importante é o conjunto de critérios estabelecidos pelo IEEE [IEEE92], dos quais citaremos alguns exemplos:

- Para documentos textuais: número de páginas, número de palavras, número de ideogramas e número de imagens.
- Para o código-fonte do produto: número de linhas de código físicas ou o número de linhas de código lógicas.

Os arquivos eletrônicos produzidos pelas ferramentas de modelagem constituem outro exemplo de artefato mensurável. Pode-se contar, por exemplo, o número de classes, atributos e métodos presentes em um modelo de classes. Nesses casos, a contagem geralmente pode ser automatizada através de facilidades providas pelas próprias ferramentas de modelagem.

O resultado das medições realizadas sobre cada artefato, expresso nas unidades de medida definidas pela organização na instanciação dos elementos configuráveis do modelo (seção 4.2.2), deve ser registrado no formulário para registro de medidas de artefatos, ilustrado na FIGURA 11.

Ao contrário da contagem de pontos de função, como as informações registradas aqui não serão utilizadas para acompanhamento dos projetos, o registro de tamanho dos artefatos será único para todo o projeto. Ele poderá ser feito à medida que os artefatos forem concluídos ou ao final do projeto.

Formulário de registro de medidas de artefatos

Organizações ProSoft LTDA

Nome do projeto	*Locadora 1.0*
Natureza dos dados	*Reais*
Identificador do planejamento	*-*

Medidas dos artefatos

Artefato	Critério de medida	Valor medido
ERSw	*Número de páginas*	*238*
ERSw	*Número de casos de uso*	*34*
MASw	*Número de classes*	*76*
MASw	*Número de métodos*	*254*
MASw	*Número de atributos*	*305*
MDSw	*Número de classes*	*142*
MDSw	*Número de métodos*	*632*
MDSw	*Número de atributos*	*402*
CFSw	*Número de linhas de código, excluindo-se comentários e linhas em branco*	*6342*

FIGURA 11 - Formulário para registro de medidas de artefatos

A FIGURA 12 mostra como o modelo de classes representa a medição dos artefatos. A classe “Realização ou planejamento de projeto” se deve ao fato das medições de artefatos serem utilizadas também no contexto do planejamento dos projetos. Essa forma de utilização das medições de artefatos será apresentada na seção 4.1.7.

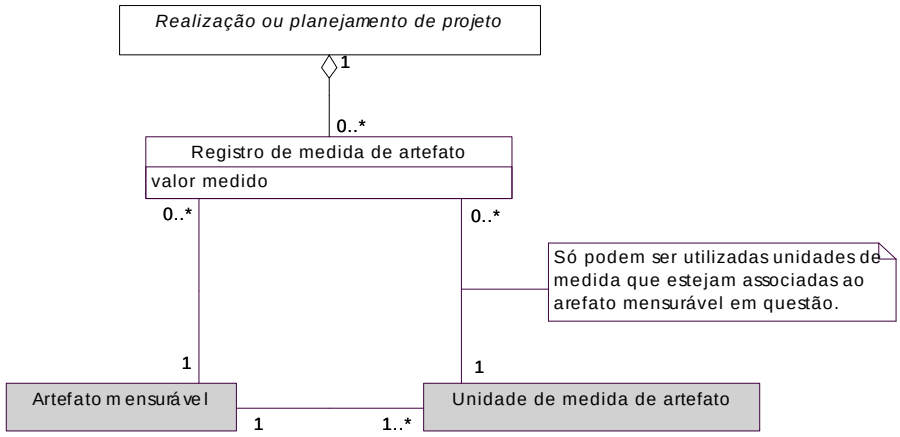


FIGURA 12 - Visão do registro de tamanho de artefatos, sob a forma de um diagrama de classes

4.1.2 Medidas de estabilidade de requisitos

Um problema comum no desenvolvimento de software é a instabilidade dos requisitos, que ocorre quando clientes e usuários trazem novos requisitos, ou mudanças de requisitos, quando o desenvolvimento já está em fase adiantada [PAULA03]. Ocorrências desse tipo costumam ter um custo bastante elevado, já que podem resultar em perda de parte do trabalho já realizado e na alteração do tamanho do produto, tendo um impacto inevitável no esforço necessário ao seu desenvolvimento.

Uma vez que os requisitos foram especificados e aprovados por clientes e desenvolvedores, qualquer alteração nessa especificação deve ser tratada por procedimentos bem definidos, que constituem a disciplina de *Gestão de Requisitos*. Apesar de parte das alterações de requisitos serem causadas por fatores externos aos projetos (como mudanças em legislações vigentes que venham a ter impacto nos requisitos, por exemplo), boa parte delas poderia ser evitada por um bom trabalho de especificação de requisitos. O registro de requisitos de forma ambígua ou incompleta constitui uma fonte significativa desse tipo de problema.

Para se ter um controle sobre a estabilidade dos requisitos durante os projetos de desenvolvimento, é necessário acompanhar quantitativamente todas as alterações em requisitos ocorridas. Esse acompanhamento quantitativo permite:

- Identificar e monitorar o risco representado por essas alterações para o sucesso dos projetos, tendo em vista o impacto em prazos e esforço;
- Avaliar a qualidade do processo de especificação de requisitos;
- Mensurar o curso das alterações, em termos do esforço empregado para fazer as alterações necessárias nos artefatos.

O acompanhamento é feito através do registro de cada alteração ocorrida nos requisitos em um formulário próprio, apresentado na FIGURA 13. Nele devem ser informados os seguintes dados:

- **Nome do projeto** – Projeto de desenvolvimento de software em andamento na organização ao qual pertence(m) o(s) requisito(s) alterado(s).
- **Identificador da alteração** – Código utilizado para identificar aquela alteração de requisitos. Deve ser único para cada alteração cadastrada em um projeto.
- **Data da solicitação** – Data em que a alteração foi formalmente solicitada pelo cliente.
- **Data da conclusão** – Data em que a alteração solicitada foi completamente concluída.
- **Descrição** – Descrição textual da alteração realizada, para fins de documentação.
- **Tamanho** – Tamanho (em pontos de função ajustados) da alteração. Para calculá-lo, devem ser utilizadas as técnicas de contagem de pontos de função para alteração de produtos [LONGSTREET02].
- **Esforço dedicado** - Total aproximado de horas trabalhadas que foram dedicadas à atualização dos artefatos que sofreram impacto com alteração dos requisitos. Geralmente essa atualização envolve um conjunto de providências a serem tomadas: identificação dos artefatos impactados, atualização desses artefatos e validação da atualização, que pode ser através de teste (no caso de alteração que implique em modificações no código-fonte, por exemplo) ou de simples verificação. O esforço dedicado a todas essas atividades deve ser contabilizado.
- **Requisitos alterados** – Listagem de todos os requisitos afetados pela alteração. Uma alteração deve estar relacionada a pelo menos um requisito. O cadastros os requisitos é detalhado na seção 4.1.1.

É importante ressaltar que devem ser registradas aqui apenas alterações em requisitos solicitadas pelo cliente. Correções de defeitos que tenham sido injetados na especificação de requisitos por erro dos desenvolvedores devem ser registradas como defeitos (seguindo-se os procedimentos definidos na seção 4.1.5), e não como alterações em requisitos. Mesmo existindo semelhanças entre o processo de correção de defeitos e o de alteração de requisitos, é gerencial e conceitualmente importante diferenciá-los: defeitos são decorrentes de erros cometidos pelo desenvolvedores, enquanto alterações de requisitos são mudanças solicitadas pelo cliente em relação ao que havia sido acordado inicialmente com os desenvolvedores.

Formulário de registro de alteração em requisitos Organizações ProSoft LTDA	
Nome do projeto	*Quantum 1.0*
Identificador da alteração	*AR_09*
Data da solicitação	*29/10/2002*
Data da conclusão	*17/11/2002*
Descrição	*Alteração das restrições aplicáveis ao valor da data de um marco de projeto.*
Tamanho (PF ajustados)	*11,3*
Esforço dedicado (h)	*33,2*

Requisitos alterados	
Identificador	**Descrição**
CUA8	*Gestão de cronograma de projeto*
CUA9	*Registro de marco de projeto*

FIGURA 13 - Formulário para registro de alteração em requisitos

O modelo de classes da FIGURA 14 apresenta a modelagem para o armazenamento dessas informações que compõem o registro de uma alteração em requisito.

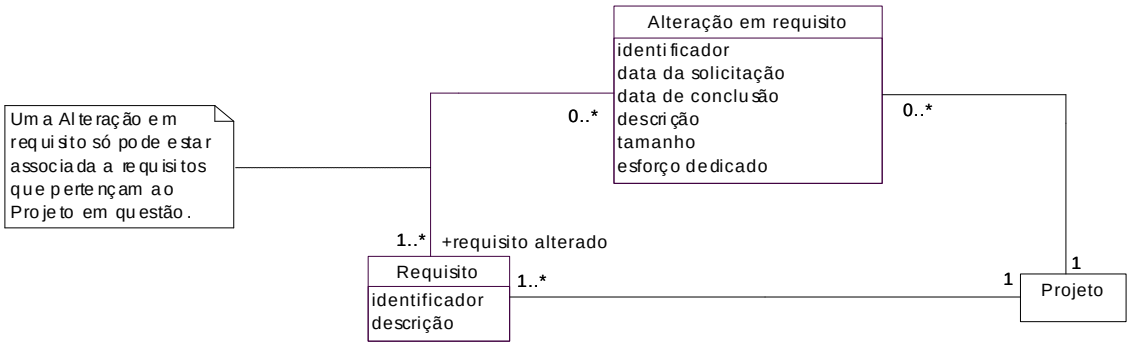


FIGURA 14 - Visão do registro de alteração em requisitos, sob a forma de um diagrama de classes

Com exceção da data de conclusão e do esforço dedicado, todos os dados deverão ser preenchidos no momento em que uma solicitação da alteração for recebida. As informações sobre a data de conclusão e o esforço dedicado devem ser preenchidas imediatamente após a conclusão das atualizações dos artefatos.

4.1.3 Medidas de esforço

O esforço é uma das medidas mais aceitas e utilizadas para compreender e gerenciar

processos e projetos de software, por vários motivos.

Em primeiro lugar, o esforço empregado em um projeto de software constitui um dos principais indicadores de seu custo, já que uma fração significativa dos orçamentos dos projetos é dedicada à remuneração da equipe envolvida. Mesmo não sendo suficientes, por si só, para determinar com grande precisão o valor investido, os indicadores de esforço seguramente possuem uma forte correlação com esse custo [DoD00].

Outro fator importante é que o conhecimento do esforço dedicado aos projetos consiste, juntamente com as informações sobre o tamanho dos produtos desenvolvidos, em um dos ingredientes indispensáveis para o cálculo da produtividade das equipes. Essa informação sobre a produtividade será essencial para o conhecimento do desempenho da organização e para a realização de estimativas de prazos e custos em projetos futuros.

O planejamento de projetos é outra aplicação direta da caracterização quantitativa do esforço. É necessário conhecer a distribuição típica do esforço ao longo dos passos do processo para que seja possível estimar os recursos humanos que provavelmente serão requeridos em cada etapa dos projetos, e com isso dimensionar as equipes de desenvolvimento. Tal conhecimento só pode ser obtido através da análise de dados históricos de esforço coletados em projetos já desenvolvidos pela organização.

O conhecimento da distribuição do esforço ao longo dos passos do processo é útil também para as atividades de melhoria do próprio processo. Os passos que consomem a maior parte dos recursos poderão ser escolhidos como os primeiros alvos das atividades de melhoria, por serem potencialmente capazes de trazer maior impacto sobre o desempenho geral da organização.

Como qualquer outra medida, a operacionalização da coleta de dados de esforço exige a definição prévia de algumas regras. Em particular, dois aspectos devem ser esclarecidos:

- A unidade de medida a ser utilizada;
- A opção pelo registro do esforço líquido ou nominal.

As seções a seguir discutem com mais detalhes cada um desses aspectos.

4.1.3.1 Escolha da unidade de medida

A unidade de medida escolhida para medição do esforço é a *hora trabalhada* (equivalente ao termo em inglês *staff-hour*). Essa é a unidade recomendada pelo IEEE para o registro de esforço, e é definida como “uma hora de trabalho exercida por um membro da equipe envolvida no projeto” [IEEE92].

Outras unidades poderiam ser utilizadas, como pessoas-mês ou semanas de trabalho. Porém, tais unidades de medida podem apresentar alguns inconvenientes [GOETHERT92]:

- O número de horas trabalhadas por uma pessoa durante um mês, ou durante uma semana, pode variar entre diferentes organizações, ou mesmo entre diferentes

projetos de uma mesma organização.

- Medidas como pessoas-mês geralmente não proporcionam um nível de granulosidade adequado para medir atividades individuais, de duração mais curta.
- Medidas como semanas de trabalho geralmente levam em consideração que cada semana contém 5 dias de trabalho. Porém, o esforço equivalente a cada dia de trabalho pode variar entre diferentes organizações. Além disso, eventos como feriados e horas extras de trabalho devem ser tratados de alguma forma específica, tornando as atividades de coleta e análise desses dados mais trabalhosas.

Utilizando-se a hora trabalhada como unidade de medida fundamental, problemas desse tipo podem ser evitados. Além disso, a partir do registro das horas trabalhadas, as demais unidades de medida podem ser facilmente obtidas, caso haja necessidade.

4.1.3.2 Horas nominais x *horas líquidas*

Para padronizar a medição do esforço, o conceito de hora trabalhada precisa ser definido com maior precisão. Em particular, deve-se deixar claro se o esforço deve ser registrado em horas líquidas ou em horas nominais.

Se o registro for baseado em horas líquidas, todo o tempo dedicado à realização de tarefas não produtivas (pequenas interrupções no decorrer do trabalho, pausas para descanso, realização e atendimento de chamadas telefônicas, etc.) deverá ser descontado do tempo total de trabalho. Se a opção for por horas nominais, tais interrupções não são levadas em consideração. Nesse caso, registra-se apenas o horário de início e término de cada atividade, e considera-se todo esse intervalo de tempo como sendo um período de trabalho. Cada uma das opções apresenta vantagens e desvantagens.

O registro de horas líquidas permite a obtenção de medidas mais precisas de produtividade, pois leva em consideração somente as horas de trabalho efetivamente dedicadas à realização das atividades produtivas. Porém, a distinção entre horas produtivas e improdutivas está sujeita interpretação pessoal de quem faz o registro, o que prejudica em parte a acurácia dos dados coletados.

Optando-se pelo registro de horas líquidas, as informações coletadas não devem ser utilizadas para fins de avaliação de desempenho individual, e isso deve ser deixado claro a todos os integrantes da equipe. Caso contrário, eles se sentiriam intimidados em registrar grandes períodos improdutivos quando estes ocorressem. Riscos desse tipo só são contornados assegurando-se a todos o anonimato das informações prestadas.

Outro aspecto importante a considerar é que a utilização de horas líquidas não dispensa a necessidade de registrar também as horas nominais, por dois motivos:

- Para garantir que todo o tempo de trabalho perdido com tarefas improdutivas seja efetivamente excluído dos valores registrados como horas líquidas de trabalho, é

necessário que esse registro esteja desvinculado dos cálculos de remuneração dos integrantes da equipe. As horas nominais teriam que ser então contabilizadas à parte, para esse fim.

- Se o registro de esforço é baseado em horas líquidas, as medidas de produtividade obtidas consideram apenas essas horas, e as estimativas de esforço realizadas com base nessa produtividade fornecem apenas o esforço líquido previsto. No entanto, esses resultados não podem ser usados diretamente para o dimensionamento de recursos, pois parte deles será inevitavelmente perdida em atividades improdutivas. Estudos mostram que apenas 50% a 75% do esforço nominal é empregado em atividades produtivas [HUMPHREY95]. Para conhecer com precisão qual seria esse fator de conversão, seria necessário contabilizar à parte as horas nominais e compará-las com o montante de horas líquidas.

Se o registro de horas líquidas não dispensa o de horas nominais, uma alternativa é registrar apenas as horas nominais. Essa opção mostra-se vantajosa por diversos motivos:

- A medição de esforço pode ser unificada com a coleta de horas de trabalho para fins de pagamento, evitando a necessidade de dois registros distintos para o mesmo trabalho realizado. Além de poupar tempo dos membros da equipe, reduz a probabilidade de ocorrência de erros.
- A garantia do anonimato das informações prestadas deixa de ser uma preocupação, já que o registro de horas nominais normalmente já é de conhecimento dos responsáveis pela remuneração da equipe.
- A obtenção dos índices de produtividade a serem aplicados na realização de estimativas em novos projetos toma como referência o esforço em horas nominais, que é a unidade mais adequada para esse fim. As estimativas obtidas podem ser aplicadas diretamente para dimensionar os recursos, pois refletem o esforço total previsto (e não só sua fração produtiva), evitando a necessidade da utilização de fatores de conversão.

Uma conseqüência da opção pelo registro único de horas nominais é que o índice de produtividade obtido será influenciado pelos períodos de improdutividade e ociosidade de cada membro da equipe. Isso significa que quanto maior a ocorrência de interrupções de trabalho e quanto mais tarefas improdutivas forem realizadas no dia-a-dia da organização, menor será o índice de produtividade. Tal conseqüência é perfeitamente aceitável se considerarmos que a produtividade está associada ao custo por unidade produzida (numa relação de proporção inversa), e que o custo não é determinado somente pelas horas líquidas de trabalho, mas também pelas horas desperdiçadas. Quanto maior o número de horas de trabalho perdidas, maior o custo por unidade produzida, e menor a produtividade.

Pelas razões enumeradas, o modelo de informação proposto neste trabalho utilizará o registro de horas nominais de trabalho como medida de esforço.

4.1.3.3 Registro das medidas de esforço

A coleta de dados sobre esforço deve ser feita por todos os desenvolvedores envolvidos em projetos de software da organização, sendo cada desenvolvedor responsável pela coleta de seus dados. Ao final de cada dia de trabalho, os desenvolvedores devem registrar o esforço empregado nas atividades desempenhadas naquele dia.

Os dados de esforço deverão ser registrados em um formulário específico, ilustrado na Figura 15. Nele, deverão ser registradas as seguintes informações sobre cada esforço registrado:

- **Desenvolvedor** – Membro da equipe de desenvolvimento a quem se refere o esforço registrado.
- **Data** – Data em que ocorreram as atividades cujo esforço está sendo registrado. Este campo é importante por dois motivos, principalmente:
  - Facilitar o controle do registro das horas pelos membros da equipe, auxiliando-os a identificar atividades omitidas e replicadas;
  - Permitir a identificação da iteração à qual o esforço deve ser associado, a partir dos marcos de início e término de cada uma delas.
- **Projeto** – Projeto de desenvolvimento de software em andamento na organização ao qual se referem as atividades realizadas. Cada esforço registrado deve estar associado a um e somente um projeto. Por isso, atividades referentes a projetos distintos, mesmo se realizadas no mesmo dia de trabalho, devem ser registradas separadamente.
- **Fluxo** – Fluxo ao qual pertencem as atividades realizadas no período de tempo que está sendo registrado. As opções possíveis são definidas pelo processo de desenvolvimento adotado no projeto em questão. Atividades pertencentes a fluxos diferentes devem ser registradas como entradas distintas no formulário.
- **Duração** – Período de tempo dedicado às atividades cujo esforço está sendo registrado. Pelos motivos já mencionados, a duração deve ser expressa em horas trabalhadas.

Todos os campos são de preenchimento obrigatório.

Formulário para registro de esforço

Organizações ProSoft LTDA

Período: 20/07/2002 a 26/07/2002

Desenvolvedor	Data	Projeto	Fluxo	Duração (h)
José Augusto Soares	*20/07/2002*	*Locadora 1.0*	*Implementação*	*5,4*
José Augusto Soares	*20/07/2002*	*Locadora 1.0*	*Desenho*	*2,6*
Maria Antônia das Dores	*20/07/2002*	*Locadora 1.0*	*Gestão de projeto*	*1,9*
José Augusto Soares	*21/07/2002*	*Locadora 1.0*	*Desenho*	*3,5*
José Augusto Soares	*21/07/2002*	*Locadora 1.0*	*Desenho*	*2,4*
Maria Antônia das Dores	*21/07/2002*	*Locadora 1.0*	*Gestão de projeto*	*4,1*
Maria Antônia das Dores	*21/07/2002*	*Pharma 2.0*	*Gestão de projeto*	*3,9*

FIGURA 15 - Formulário para registro de esforço

A FIGURA 16 apresenta a modelagem das informações coletadas nos registros de esforço sob a forma de um diagrama de classes UML.

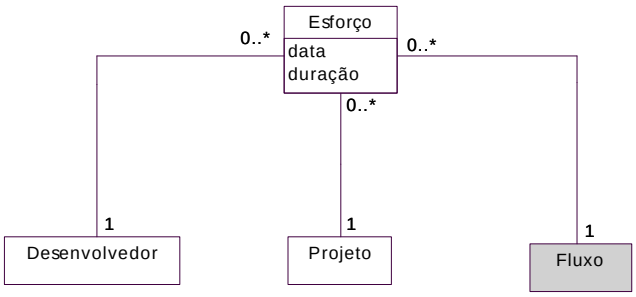


FIGURA 16 - Visão das medidas de esforço sob a forma de um diagrama de classes

As medidas de estabilidade de requisitos, defeitos e revisões (apresentadas nas seções 4.1.2, 4.1.5 e 4.1.6, respectivamente) também contêm um registro próprio de esforço: as medidas de defeitos armazenam o esforço dedicado à correção de cada defeito detectado, as medidas de estabilidade de requisitos armazenam o esforço adicional consumido por alterações em requisitos, e as medidas de revisões registram o esforço dedicado a cada atividade de revisão realizada. No entanto, esses registros são feitos para fins bastante específicos, e não substituem o preenchimento do formulário apresentado acima. Em outras palavras: o esforço dedicado a revisões, correção de defeitos e mudanças decorrentes de alterações em requisitos deverá ser registrado tanto no formulário acima (onde cada desenvolvedor envolvido registra seu próprio esforço) quanto no formulário específico para registro de defeito, revisão ou alteração em requisito (onde é registrado o esforço total dedicado àquela tarefa, considerando todos os envolvidos).

4.1.3.4 Classificação dos membros da equipe

Uma equipe de desenvolvimento, principalmente em projetos de médio e grande porte, é formada por desenvolvedores de diferentes níveis de formação e experiência. As organizações geralmente adotam classificações para esses diferentes níveis, como por exemplo: analista júnior, analista pleno, analista sênior, etc.

Mesmo que uma organização não adote oficialmente uma classificação formal para seus desenvolvedores, é possível fazer uma divisão para fins de mensuração, tomando como base o nível de remuneração.

Conhecendo a classificação de cada desenvolvedor é possível obter, além do esforço total dedicado aos projetos, a distribuição desse esforço por classes de desenvolvedores. Essa informação será útil em dois contextos:

- Permitir o planejamento de equipes em projetos futuros, levando em consideração a quantidade necessária de desenvolvedores em cada nível de experiência, reduzindo assim o risco de se alocar equipes muito inexperientes.
- Melhorar a aplicabilidade do registro de esforço como indicador de custo dos projetos, já que as classificações dos membros das equipes refletem, pelo menos aproximadamente, diferentes níveis de remuneração. É possível, assim, obter uma melhor aproximação do custo total dos projetos, no que se refere ao gasto com pessoal.

O registro da classificação dos membros da equipe deve ser feito por projeto, pois é possível que um mesmo desenvolvedor mude de nível de um projeto para outro.

No momento em que um projeto for iniciado, é necessário cadastrá-lo num formulário próprio (apresentado na Figura 17) e informar a lista dos desenvolvedores nele envolvidos. Para cada desenvolvedor, então, informa-se a classificação. Os valores possíveis para esse campo são previamente definidos pela organização.

Formulário de informações sobre projeto

Organizações ProSoft LTDA

Nome do projeto	*Locadora 1.0*
Descrição	*Sistema para informatização de locadoras.*
Processo utilizado	*Praxis padrão, versão 2.0*

Equipe alocada para o projeto

Nome do desenvolvedor	**Classificação**
José Augusto Soares	*Analista Júnior*
Maria Antônia das Dores	*Analista Júnior*
Antônio Moreira Cardoso	*Analista Júnior*
Joana Lopes Ribeiro	*Analista Sênior*
Cristina Assis de Carvalho	*Analista Pleno*
Paulo Roberto Magalhães Carvalho	*Analista Pleno*
Ana Carolina Lopes Velloso	*Analista Júnior*

FIGURA 17 - Formulário para registro de informações sobre projeto

Um desenvolvedor só pode registrar esforço para um projeto se estiver cadastrado em sua equipe, com uma classificação definida. Essa regra tem o objetivo de garantir que todo registro de esforço possa ser mapeado para uma (e exatamente uma) classificação de membro da equipe.

Outra informação importante que deve ser registrada no formulário de informações sobre projeto é a definição do processo de desenvolvimento que estará sendo utilizado. Ela será usada não só no contexto das medições de esforço, mas para quase todo o processo de mensuração, já que a maior parte das medidas contém informações sobre elementos do processo.

A Figura 18 apresenta o diagrama de classes para as informações sobre projetos e suas respectivas equipes.

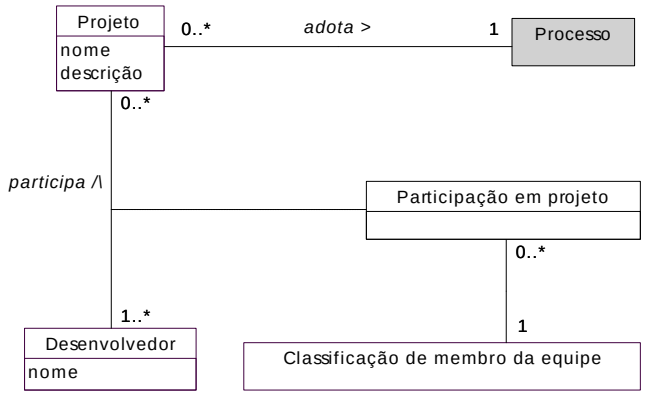


FIGURA 18 - Visão das informações sobre projetos e equipes sob a forma de um diagrama de classes

4.1.4 Medidas de progresso e cronograma

O cumprimento de prazos é certamente uma das preocupações básicas em qualquer projeto de software. Em algumas situações, a pontualidade na entrega de um produto chega a ser tão importante quanto sua funcionalidade ou qualidade.

Durante a realização de um projeto, é necessário acompanhar a evolução do progresso em pontos intermediários do ciclo de desenvolvimento, pois somente dessa forma é possível prever futuros problemas e tomar medidas para evitá-los. A idéia é detectar eventuais atrasos o mais cedo possível, de modo a viabilizar ações corretivas eficazes.

Esse acompanhamento deve ser feito utilizando-se medidas adequadas, que aqui são denominadas medidas de progresso. Elas devem permitir quantificar a fração já concluída dos projetos em pontos intermediários do ciclo de desenvolvimento, para que seja possível compará-las com os valores planejados e identificar eventuais desvios. Esses pontos intermediários de controle serão chamados *marcos de projeto*[3].

4.1.4.1 Marcos de projeto

Um marco de projeto é caracterizado por uma data e pelo progresso obtido no projeto até essa data. A maior dificuldade consiste em especificar quantitativamente esse progresso. Neste trabalho, foi utilizada a proposta apresentada em [PAULA03], onde o cálculo do progresso é baseado nos estados em que se encontram os requisitos do produto.

Um estado é representado por um nome e por um número de 0 a 100, numa escala percentual, que mede o quanto já se caminhou quanto ao oferecimento da fatia de funcionalidade

[3] Neste trabalho, o termo *marco de projeto* será utilizado com o mesmo significado do termo em inglês *milestone*).

representada pelo requisito.

A monitoração do progresso consiste, portanto, na informação sobre o estado em que se encontra cada requisito na data representada pelo marco de projeto que está sendo registrado (os estados possíveis e os respectivos valores devem ser configurados pela organização, conforme descrito na seção 4.2.2). Uma vez obtidas essas informações para todos os requisitos funcionais, é possível estimar o progresso obtido até então no desenvolvimento do produto como um todo.

O registro do estado dos requisitos será feito no mesmo formulário utilizado para o registro de tamanho, apresentado na seção 4.1.1. A Figura 19 apresenta novamente esse formulário, destacando os campos que serão utilizados agora para a caracterização do progresso.

Além do tamanho e do estado dos requisitos, esse formulário contém um cabeçalho onde é feita a identificação do marco do projeto. Nele, são informados o nome do marco, o projeto ao qual pertence, a data do marco, a natureza dos dados (reais ou planejados) e o identificador do planejamento. Os dois últimos campos mencionados estão relacionados à utilização desse formulário para o registro de valores planejados, como é descrito na seção 4.1.7.

Formulário para registro de marco de projeto

Organizações ProSoft LTDA

Nome do projeto	*Locadora 1.0*
Nome do marco de projeto	*Fim da iteração LR*
Data do marco	*20/07/2002*
Natureza dos dados	*Reais*
Identificador do planejamento	*-*

Situação dos requisitos

Requisito	**Descrição**	**Tamanho**	**Estado**	
CUA1	*Gestão do registro de esforço*	*42*	*Levantado*	*20%*
CUA2	*Gestão do cadastro de projetos*	*45*	*Levantado*	*20%*
CUA3	*Gestão de usuários*	*19*	*Levantado*	*20%*
CUA4	*Gestão do registro de defeitos*	*40*	*Levantado*	*20%*
CUA5	*Gestão do registro de revisões*	*35*	*Levantado*	*20%*
CUA6	*Gestão do cadastro de requisitos*	*20*	*Levantado*	*20%*
CUA7	*Gestão do cadastro de planejamentos*	*25*	*Levantado*	*20%*
CUA8	*Gestão de cronograma de projeto*	*24*	*Levantado*	*20%*
CUA9	*Registro de marco de projeto*	*27*	*Levantado*	*20%*
CUA10	*Registro de métricas de artefatos*	*21*	*Levantado*	*20%*
CUA11	*Gestão de alterações em requisitos*	*26*	*Levantado*	*20%*
CUA12	*Registro de estimativa de esforço*	*18*	*Levantado*	*20%*
CUA13	*Registro de estimativa de defeitos*	*11*	*Levantado*	*20%*
CUA14	*Registro de estimativa de revisões*	*11*	*Levantado*	*20%*
CUA15	*Registro de estimativa de estabilidade de requisitos*	*7*	*Levantado*	*20%*
CUA16	*Gestão do cadastro de processos*	*22*	*Levantado*	*20%*
CUA17	*Gestão do cadastro de classificações de membros da equipe*	*19*	*Levantado*	*20%*

Tamanho total (PF brutos)	*412*
Fator de ajuste	*0,932*
Tamanho total (PF ajustados)	*383,9*

FIGURA 19 - Formulário para registro de marco de projeto, destacando a medição do progresso

No exemplo apresentado, todos os requisitos estão no estado *Levantado*, que representa um progresso de 20%.

Deve ser registrado, no mínimo, um marco de projeto para cada final de iteração. Marcos adicionais também podem ser utilizados em pontos intermediários, mas são opcionais e a quantidade deles pode variar de projeto a projeto. O diagrama apresentado na FIGURA 20 ilustra essa estrutura na forma de um diagrama de classes.

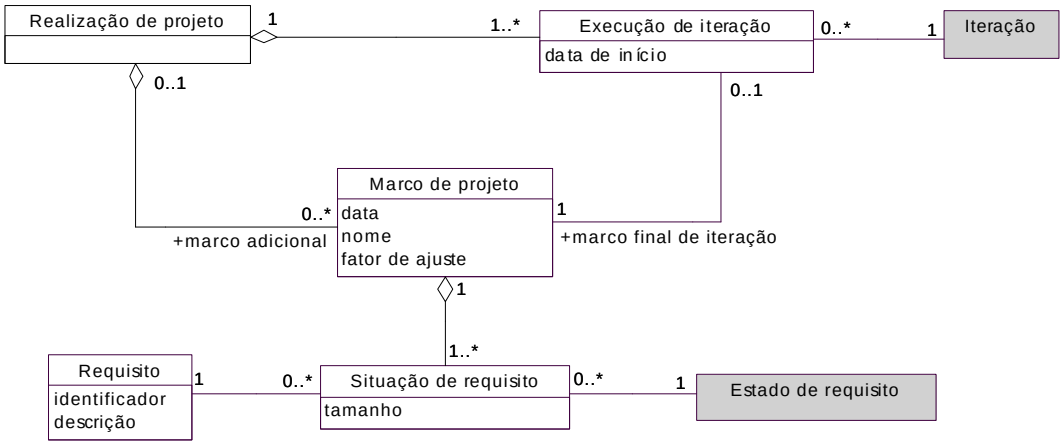


FIGURA 20 - Diagrama de classes que representa os marcos finais de iteração e a opção de criação de marcos adicionais

4.1.4.2 Registro de cronograma

Outro dado importante para o acompanhamento dos projetos é o cronograma obtido, representado pelas datas de início e fim de cada iteração. Essa informação será útil em dois contextos:

- Para permitir a comparação entre as datas realmente obtidas ao longo do projeto com o que estava previsto no cronograma planejado, identificando eventuais atrasos (o registro do cronograma planejado é detalhado na seção 4.1.7).
- Para possibilitar a divisão dos resultados das medições em iterações. Em várias outras medidas já apresentadas, são coletadas informações sobre as datas em que ocorreram os eventos registrados. Por exemplo, o registro de defeitos armazena a data de detecção de cada defeito; o registro de esforço armazena a data à qual se refere cada esforço informado. Posteriormente, quando esses dados coletados forem analisados, as datas de início e fim de cada iteração serão usadas para distribuir os valores mensurados entre as iterações do processo, permitindo que se obtenha informações sobre o desempenho do processo em cada uma delas.

As iterações de cada projeto são determinadas pelo processo de desenvolvimento adotado. O registro das datas de início e término de cada uma é feito através do formulário ilustrado na FIGURA 21. É importante ressaltar que nele devem ser registradas as datas efetivamente obtidas no decorrer do projeto, e não as datas planejadas. Por isso o registro deve ser feito ao longo do projeto, na medida em que cada iteração for iniciada ou concluída. Os campos correspondentes a marcos ainda não atingidos devem ser deixados em branco.

Formulário de registro de cronograma

Organizações ProSoft LTDA

Nome do projeto	*Quantum 1.0*	
Natureza dos dados	*Reais*	
Identificador do planejamento	-	
Iterações do projeto		
Iteração	**Data de início**	**Data de término**
AT (Ativação)	*05/06/2002*	*16/06/2002*
LR (Levantamento de Requisitos)	*17/06/2002*	*20/07/2002*
AR (Análise de Requisitos)	*21/07/2002*	*15/09/2002*
DI (Desenho Implementável)	*16/09/2002*	
L1 (Liberação 1)		
L2 (Liberação 2)		
TA (Teste Alfa)		
TB (Teste Beta)		
OP (Operação Piloto)		
Marcos adicionais de projeto		
Nome do marco		**Data**
Conclusão da ERSw para revisão técnica		*06/09/2002*

FIGURA 21 - Formulário para registro de cronograma.

A estrutura de classes criada para armazenar as informações de cronograma é a mesma utilizada para lidar com os marcos de projeto (FIGURA 20), e adota a seguinte estratégia:

- Para as iterações de um projeto, a data de início é armazenada na classe “Execução de iteração”, e a data final corresponde à data registrada no marco de projeto que representa o final daquela iteração.
- Para os marcos adicionais, a data do marco é informada na própria classe “Marco de projeto”.

4.1.5 Medidas de defeitos

A principal motivação em se coletar dados sobre defeitos detectados nos produtos de software é que essas informações constituem um importante indicador de qualidade dos produtos desenvolvidos.

Dentro de todos os aspectos relacionados ao desenvolvimento de um produto de software, a qualidade do produto é provavelmente o mais difícil de ser mensurado. O próprio conceito de qualidade é difícil de se definir com precisão, devido à quantidade de fatores envolvidos e à subjetividade inerente a boa parte deles.

A opção pela medição dos defeitos como forma de controlar quantitativamente a qualidade dos produtos se deu por um conjunto de motivos:

- Mesmo não sendo o único parâmetro que influencia o conceito de qualidade de um produto de software, a taxa de defeitos constitui certamente um dos mais significativos, por influir diretamente na percepção de qualidade pelos usuários [Florac92].
- A contagem de defeitos constitui a base de diversos outros indicadores de qualidade comumente utilizados, como confiabilidade, correção, completeza, eficiência e usabilidade [IEEE90].
- Os defeitos constituem um dos poucos critérios de qualidade de mensuração objetiva, o que torna sua coleta relativamente simples.
- Medidas de defeitos podem ser obtidas em fases intermediárias do desenvolvimento do produto. Por constituírem um importante indicador da qualidade provável do produto final, seu acompanhamento já nas fases intermediárias possibilita a tomada de ações corretivas em tempo hábil, caso o nível de qualidade se mostre abaixo do aceitável.

Além de permitir a monitoração da qualidade dos produtos, a medição dos defeitos pode ser útil para identificar possibilidades de melhoria da produtividade das equipes. Como parte do esforço consumido pelos projetos é destinada à correção de defeitos, a incidência desses defeitos influi diretamente no custo de um projeto. Ao controlá-los quantitativamente, é possível identificar os pontos do processo mais problemáticos, ou seja, as iterações e fluxos que estão inserindo o maior número de defeitos, e trabalhar para reduzi-los.

Finalmente, o registro dos defeitos é útil para o acompanhamento dos projetos. Uma alta taxa de defeitos pode significar uma menor produtividade da equipe, devido ao aumento da fração do esforço dedicada às atividades de correção. A identificação prévia desse tipo de problema permite aos gerentes de projeto tomar as medidas necessárias em tempo hábil, evitando surpresas na fase final dos projetos.

4.1.5.1 Definição do conceito de defeito

Um defeito pode ser definido como qualquer problema ou imperfeição encontrado em um produto ou artefato de software, cuja correção implique na alteração de pelo menos um artefato produzido. Uma vez concluída a especificação de requisitos, qualquer inconsistência no produto ou em um dos seus artefatos em relação ao que está especificado no documento é classificada como um defeito. Também é considerado defeito qualquer caso de inadequação a padrões e critérios de qualidade seguidos pela organização.

No entanto, serão considerados aqui apenas os problemas decorrentes de erros cometidos por membros da equipe de desenvolvimento. Alterações em requisitos (alterações

solicitadas pelo cliente em um requisito do produto após a aprovação da especificação de requisitos) não devem ser registradas como defeitos, mas em um formulário à parte, como foi descrito na seção 4.1.2.

Uma vez estabelecidos esses conceitos, é necessário ainda definir mais precisamente os critérios para a classificação de um problema qualquer como um defeito. Por exemplo, um erro cometido por um desenvolvedor ao codificar uma classe, se percebido e corrigido por ele imediatamente, deve ser registrado como um defeito? Naturalmente, devem ser estabelecidos critérios para que um problema percebido seja considerado ou não um defeito.

Um possível critério seria registrar apenas defeitos detectados em uma iteração seguinte à que foram injetados. É essa a estratégia adotada no PSP [HUMPHREY95], por exemplo. No entanto, por desconsiderar os defeitos injetados e detectados em uma mesma iteração, tal critério não se mostra adequado para o modelo de medição aqui proposto, já que a maior parte dos defeitos detectados em revisões se enquadra nessa situação.

Outro critério possível seria o proposto e adotado pelo laboratório de engenharia de software da NASA: registrar defeitos para fins de mensuração somente após a unidade de software ter sido colocada sob gestão de configuração [SEL95]. O termo “unidade de software” pode ser compreendido aqui como um artefato ou parte de um artefato produzido ao longo de um projeto. No código-fonte, cada classe ou arquivo pode ser considerado uma unidade de software. Esse critério permite que defeitos injetados e removidos em uma mesma iteração sejam mensurados, mas só é viável em organizações que adotem um processo formal de gestão de configurações interno às iterações.

Por dependerem de práticas adotadas por cada organização, esses critérios deverão ser definidos caso a caso. Abaixo são destacadas três opções possíveis:

- Registrar apenas defeitos que atendam a um dos seguintes casos:
  - Defeito injetado em uma iteração e detectado em uma iteração seguinte;
  - Defeito detectado em algum procedimento de revisão ou teste.
- Registrar todos os defeitos detectados após a unidade ter sido colocada sob gestão de configuração, como proposto em [SEL95] (possível somente caso a organização adote uma política mais formal de Gestão de Configurações interna às iterações);
- Definir um outro critério, de acordo com as práticas adotadas pela organização e seu interesse quanto à caracterização dos defeitos.

O mais importante é que, uma vez definido o critério, ele seja seguido à risca em todos os projetos realizados.

4.1.5.2 Registro de defeitos

Durante o ciclo de vida do projeto, cada defeito encontrado deve ser registrado em um

formulário próprio, ilustrado na FIGURA 22. A seguir temos uma descrição dos campos desse formulário:

- **Projeto** – Projeto de desenvolvimento de software em andamento na organização ao qual pertence o artefato em que o defeito foi detectado.
- **Número do defeito** – Número de identificação do defeito. Pode ser um número seqüencial, ou qualquer outro número de identificação de defeitos já utilizado pela organização. Deve ser único para cada defeito encontrado.
- **Descrição do defeito** – Descrição textual do defeito detectado.
- **Data da detecção** – Data em que o defeito foi detectado.
- **Data da correção** – Data em que o defeito identificado foi considerado corrigido.
- **Esforço para correção** – Total aproximado de horas trabalhadas que foram dedicadas à correção do defeito. Geralmente a correção de um defeito envolve um conjunto de providências a serem tomadas: identificação da causa do defeito, alteração do artefato para remover o defeito e validação da correção, que pode ser através de teste (no caso de defeitos encontrados no código-fonte, por exemplo) ou de simples verificação. O esforço dedicado a todas essas atividades deve ser contabilizado. Caso haja mais de um desenvolvedor envolvido na correção do defeito, o esforço dedicado por todos eles deve ser somado.
- **Fluxo em que foi injetado** – Fluxo do processo no qual o defeito provavelmente foi injetado. Apesar de envolver julgamento humano, pode ser determinado pela própria natureza do defeito encontrado. Por exemplo, defeitos de sintaxe no código-fonte são geralmente injetados no fluxo de implementação, e defeitos de arquitetura são geralmente injetados no fluxo de desenho. As opções possíveis são definidas pelo processo de desenvolvimento adotado no projeto em questão.
- **Fluxo em que foi detectado** – Fluxo ao qual pertence a atividade que estava sendo realizada quando o defeito foi detectado. As opções possíveis são definidas pelo processo de desenvolvimento adotado no projeto em questão.
- **Revisão em que foi detectado** – Caso o defeito tenha sido detectado em uma atividade de revisão, ela deve ser mencionada neste campo. As opções possíveis são as revisões realizadas para o projeto em questão, cadastradas através do *Formulário de Registro de Revisões* (FIGURA 24).
- **Iteração em que foi injetado** – Iteração do projeto em que o defeito foi injetado. Caso não seja possível determiná-la com precisão, deve-se fazer uma estimativa. As opções possíveis são as iterações definidas pelo processo de desenvolvimento adotado no projeto em questão.

As iterações em que o defeito foi detectado e corrigido podem ser deduzidas a partir

das datas de detecção e correção, respectivamente, e por isso não precisam ser explicitamente registradas.

Formulário de registro de defeito

Organizações ProSoft LTDA

Nome do projeto	*Locadora 1.0*	
Número do defeito	*263*	
Descrição do defeito	*O diagrama de componentes físicos foi omitido da DDSw.*	
Data da detecção	*20/09/2002*	
Data da correção	*23/09/2002*	
Esforço para correção (h)	*9,5*	
Fluxo em que foi injetado	*Desenho*	
Fluxo em que foi detectado	*Desenho*	
Revisão em que foi detectado	*-*	
Iteração em que foi injetado	*Desenho Implementável*	
Classificação do defeito		
Propriedade	**Taxonomia utilizada**	**Espécie de defeito**
Natureza	*Humphrey*	*Documentação*
Gravidade	*Taxonomia única*	*Menor*
Duplicidade		
Defeito duplicado?	*Sim*	
Número do defeito original	*112*	

FIGURA 22 - Formulário para registro de defeito

Além dos campos descritos, o formulário contém também informações sobre a classificação do defeito e campos para os casos em que houver defeitos duplicados.

A classificação do defeito consiste em atribuir uma espécie para cada uma de suas propriedades mensuráveis. Para cada propriedade há um conjunto de taxonomias disponíveis, cada uma definindo um conjunto de espécies. A lista de propriedades, taxonomias e espécies deve ser definida previamente pela organização, de acordo com a estrutura que será apresentada na seção 4.2.2 (nessa seção os conceitos de propriedade mensurável, taxonomia e espécie serão também apresentados com mais detalhes).

Os campos a serem preenchidos são:

- **Propriedade** – Propriedade mensurável de defeito que está sendo medida. Todas as propriedades mensuráveis previstas pela política de medição devem ser registradas, e uma mesma propriedade não pode ser registrada mais de uma vez para o mesmo defeito.
- **Taxonomia** – Taxonomia escolhida para classificar o defeito quanto à propriedade que está sendo registrada. O conjunto de taxonomias disponíveis deve ser

previamente definido pela organização no momento da instanciação da política de medição. A opção pela taxonomia mais adequada em cada caso deve ser feita pelo desenvolvedor que faz o registro do defeito.

- **Espécie de defeito** – Classificação do defeito quanto à sua espécie. As espécies possíveis são aquelas previstas pela taxonomia selecionada no campo anterior.

O modelo de medição prevê também o caso em que um mesmo defeito é detectado mais de uma vez. Por exemplo, um defeito detectado em uma atividade de revisão, enquanto não for corrigido, estará sujeito a ser novamente apontado em novas revisões, ou mesmo "descoberto" novamente por acaso, por outros desenvolvedores. Em outras situações, pode-se perceber que dois registros de defeito já informados tratam, na verdade, do mesmo defeito registrado de formas diferentes. Nesses casos, dizemos que há duplicidade de defeitos.

Uma opção simples para solucionar casos de duplicidade seria simplesmente excluir um dos registros. No entanto, para fins de mensuração, é interessante manter todas as informações, inclusive sobre os registros duplicados. Afinal, trata-se da uma nova detecção do defeito, e que terá informações próprias: uma data de detecção, fluxo detector, atividade de revisão em que foi detectado (caso tenha sido detectado em uma revisão), descrição e classificação. Para manter um registro dessas informações e ao mesmo tempo impedir que defeitos duplicados prejudiquem a confiabilidade dos indicadores de qualidade, o registro desse tipo de defeito deve mencionar que trata-se de uma duplicação. A última seção do formulário da FIGURA 22 existe para esse fim.

Um defeito pode ser registrado como sendo duplicação de no máximo um outro defeito, mas dois ou mais defeitos podem ser registrados como duplicações de um mesmo defeito. O registro de duplicidades pode também ser feito em cadeia: um defeito *a* registrado como uma duplicação de *b*, que por sua vez é colocado como o sendo de um terceiro defeito, *c*. A única situação que não é permitida é a existência de ciclos na cadeia de duplicidades, pois nesse caso não é possível distinguir qual dos registros deve ser levado em consideração pelos indicadores de qualidade.

Todas as informações contidas no formulário para registro de defeito, inclusive aquelas sobre classificação e duplicidade, encontram-se modeladas no diagrama de classes exibido na FIGURA 23.

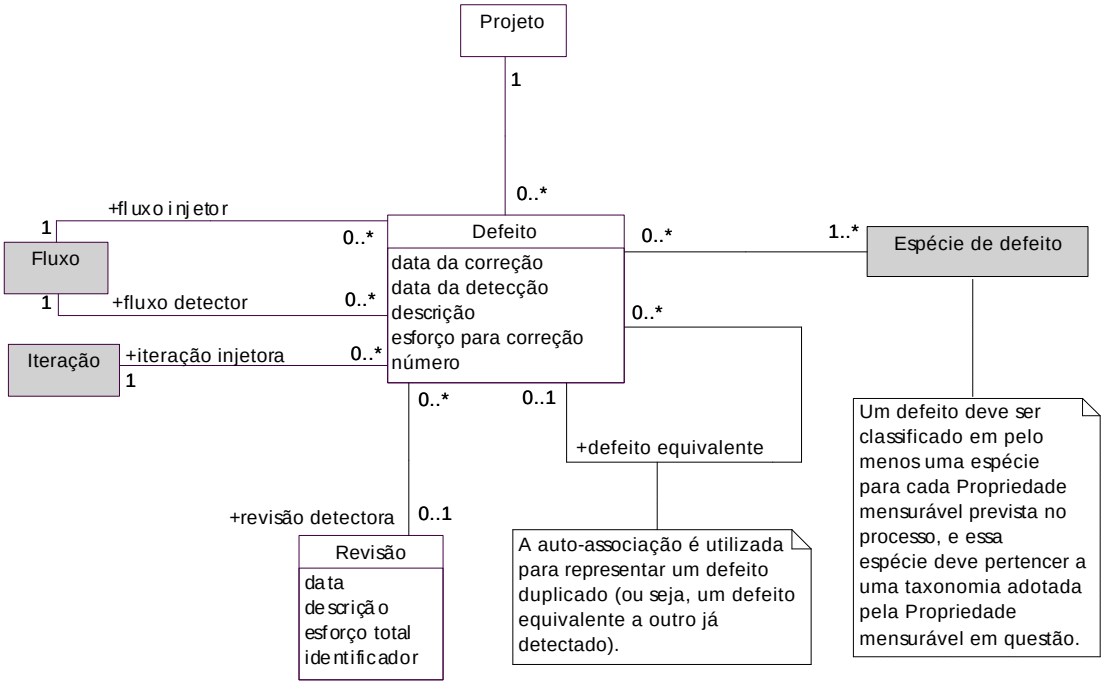


FIGURA 23 - Visão do registro de defeitos, sob a forma de um diagrama de classes

Alguns campos do formulário deverão ser preenchidos no momento em que o defeito for detectado, e outros podem ser informados posteriormente, desde que estejam preenchidos quando a correção do defeito for concluída. A TABELA 13 contém uma lista dos campos que se enquadram em cada um desses casos. Para defeitos duplicados, basta preencher os campos referentes à detecção do defeito. Para os demais defeitos, todos os campos são de preenchimento obrigatório.

Campos que devem ser preenchidos no momento da detecção	**Campos que devem estar preenchidos quando a correção for concluída**
Número do defeito	Data da correção
Projeto	Esforço para correção (h)
Descrição do defeito	Fluxo em que foi injetado
Data da detecção	Classificação
Fluxo em que foi detectado	Duplicidade
Revisão em que foi detectado	

TABELA 13 - Classificação dos campos quanto ao momento em que devem ser preenchidos

4.1.6 Medidas de revisões

Revisões de software constituem um recurso poderoso para elevar a qualidade e a produtividade do processo de software. Neste trabalho, o termo *revisão* se refere às atividades que

visam à detecção precoce de defeitos, através de verificações realizadas sobre um artefato ou conjunto de artefatos. Existem diferentes tipos de revisão, cada uma utilizando um método específico de verificação: revisões técnicas, inspeções, revisões de apresentação (*walkthrough*), revisões gerenciais, revisões informais, revisões individuais, entre outras.

Há vários registros na literatura que comprovam os benefícios trazidos pelas revisões. [HUMPHREY89] apresenta dez estudos publicados que relatam melhorias trazidas com a implantação dessa prática em organizações diversas, como a redução drástica do custo de correção de defeitos, aumento de produtividade e redução da quantidade de defeitos no produto final.

Os objetivos principais das revisões são [HUMPHREY89]:

- Detectar defeitos o mais cedo possível, durante o ciclo de desenvolvimento;
- Garantir que as partes envolvidas concordam tecnicamente com o trabalho realizado;
- Verificar se o produto sob revisão atende a um conjunto de critérios pré-definidos;
- Formalizar a conclusão de uma tarefa técnica;
- Fornecer dados a respeito do produto e do próprio processo de revisão.

Como o foco deste trabalho é a mensuração, estamos mais interessados aqui no primeiro e no último dos objetivos enumerados, por tratarem de aspectos de medição mais direta. A coleta de medidas sobre revisões se justifica por vários motivos:

- Avaliar quantitativamente a eficácia dessas atividades no que diz respeito à detecção de defeitos. Isso pode ser feito observando, do total de defeitos detectados ao longo do ciclo de desenvolvimento, o percentual que foi detectado em atividades de revisão.
- Avaliar o custo das revisões, através do controle do esforço dedicado a essas atividades.
- Comparar os diferentes tipos de revisão, verificando quais são os que apresentam melhor desempenho a um menor custo.
- Verificar se os projetos estão dedicando um montante de esforço adequado às atividades de revisão.
- Identificar problemas no próprio processo de revisão e fornecer embasamento quantitativo para a proposição de melhorias.

Para tornar possível a obtenção desses benefícios, todas as revisões realizadas ao longo dos projetos de software devem ser devidamente registradas. O formato do formulário é apresentado na FIGURA 24, e a forma de preenchimento de cada um dos campos é descrita a seguir:

- **Projeto** – Projeto de desenvolvimento de software em andamento na organização ao qual pertence(m) o(s) artefato(s) submetido(s) à revisão.

- **Identificador da revisão** – Identificador único da revisão realizada. Utilizado para permitir que a revisão possa ser referenciada do formulário para registro de defeitos, ao se registrar os defeitos nela detectados.
- **Descrição** – Descrição breve da revisão realizada.
- **Data** – Data em que foi realizada a revisão. Se a mesma for realizada em mais de um dia de trabalho, deve-se registrar a data de sua conclusão.
- **Método de verificação** – Indica o método de verificação utilizado na revisão que está sendo registrada. Exemplos de métodos são: revisão técnica, inspeção, revisão de apresentação, entre outros. As opções possíveis para esse campo são determinadas durante a instanciação do modelo de medição, como é descrito na seção 4.2.2.
- **Fluxo** – Fluxo do processo de desenvolvimento responsável pela confecção do(s) artefato(s) submetido(s) à revisão.
- **Esforço total** – Esforço total dedicado à revisão, em horas trabalhadas. Deve incluir o tempo de preparação dos revisores, o tempo de preparação da revisão (convocação dos revisores, preparação de material de apoio, etc.) e o tempo dedicado à realização das reuniões de revisão. Quando há mais de um desenvolvedor envolvido no processo, o esforço de cada um deve ser somado, e o valor registrado neste campo deve refletir o resultado dessa soma.

Todos os campos devem ser obrigatoriamente preenchidos no momento do registro da revisão.

Formulário de registro de revisão
Organizações ProSoft LTDA

Projeto	*Locadora 1.0*
Identificador da revisão	*Locadora_1_0_RTERSw*
Descrição	*Revisão técnica da Especificação de Requisitos*
Data	*28/08/2002*
Método de verificação	*Revisão Técnica*
Fluxo	*Requisitos*
Esforço total (horas)	*83*

FIGURA 24 - Formulário para registro de revisão

A FIGURA 25 mostra como as informações a serem coletadas sobre as revisões foram modeladas em um diagrama de classes. É importante notar que o modelo contém a informação sobre os defeitos detectados em uma revisão, apesar disso não ser registrado explicitamente no

formulário. Essa informação pode ser obtida a partir do registro de defeitos, apresentado na seção 4.1.5, que permite a vinculação de um defeito registrado à revisão que o detectou.

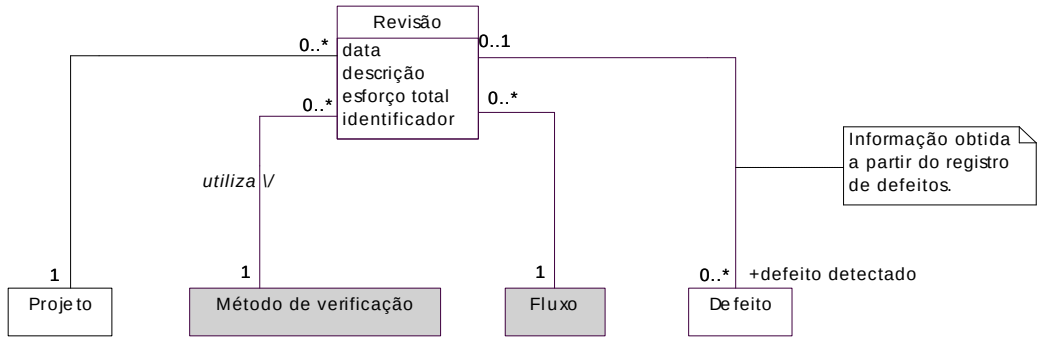


FIGURA 25 - Visão do registro de revisões, sob a forma de um diagrama de classes

4.1.7 Medidas de planejamento de projetos

As seções anteriores descrevem como devem ser realizadas as medições referentes a eventos ocorridos ao longo da execução de projetos. O objetivo dessas medidas é caracterizar quantitativamente como se deu, na prática, a execução de cada um. Por esse motivo são denominadas medidas **explanatórias** (conforme a nomenclatura definida na seção 2.1.1), sendo geradas após a ocorrência dos eventos. Medidas desse tipo contribuem para um conhecimento mais preciso do desempenho do processo de desenvolvimento ao longo dos projetos.

Um papel igualmente importante é desempenhado por outra categoria de medidas, a das medidas **preditivas.** Elas consistem em valores estimados antes da realização ou conclusão do projeto, com o objetivo de prever aspectos diversos quanto a seu andamento (por exemplo, o esforço necessário em cada etapa do desenvolvimento, prazos, etc.), subsidiando o seu planejamento. Geralmente o cálculo desses valores estimados utiliza como base dados históricos registrados em projetos similares já desenvolvidos pela organização no passado. A realização de estimativas constitui uma tarefa complexa, principalmente em casos em que o processo de desenvolvimento ainda não atingiu um certo nível de previsibilidade. Quanto menor a previsibilidade do processo, maior será o erro provável das estimativas produzidas. Para conhecer o grau de confiabilidade dessas estimativas, é necessário armazená-las e posteriormente compará-las com os valores obtidos no decorrer dos projetos.

O modelo de medição proposto neste trabalho contempla também o registro e a análise de medidas preditivas. Para cada categoria de medida apresentada anteriormente (esforço, defeitos, revisões, tamanho, progresso, cronograma e estabilidade de requisitos), foram definidas as convenções para o registro de valores estimados, possibilitando a posterior comparação com os valores obtidos, representados pelas medidas explanatórias.

No entanto, este trabalho se limita a determinar a forma como os valores estimados devem ser registrados na base de dados de medições. Não são propostas aqui técnicas específicas para a confecção dessas estimativas – para isso, já existem diversos métodos bastante difundidos na literatura ([MILLS88] apresenta um panorama dos modelos de previsão propostos na literatura). A base de dados históricos criada através das medições propostas neste trabalho poderia ser usada como entrada a qualquer um desses métodos. A escolha do método mais adequado fica a critério de cada organização.

O objetivo aqui é apenas oferecer meios para acompanhar a precisão e acurácia desses valores, independentemente do método que tenha sido usado para gerá-los, comparando-os com os valores reais. Para isso, as seções a seguir definem como deverão ser registradas as estimativas realizadas em cada planejamento de projeto.

Em primeiro lugar, cada planejamento realizado deverá ser devidamente registrado, através do formulário ilustrado na FIGURA 26. Nele, devem ser informados:

- **Projeto** – Identificador do projeto ao qual se refere o planejamento.
- **Data do planejamento** – Data em que foram calculadas as estimativas registradas no planejamento. Caso o cálculo dessas estimativas tenha sido feito em mais de um dia de trabalho, deve ser registrada a data de conclusão.
- **Identificador** – Código utilizado para identificar o planejamento. Deve ser único para cada planejamento cadastrado em um projeto.
- **Descrição** – Breve descrição textual do planejamento realizado.

Formulário de cadastro de planejamentos de projeto

Organizações ProSoft LTDA

Nome do projeto	*Locadora 1.0*	
Planejamentos cadastrados		
Data	**Identificador**	**Descrição**
12/09/2002	*PL_AR_001*	*Planejamento do projeto ao final da Elaboração*
22/10/2002	*PL_DI_001*	*Replanejamento devido a alterações em requisitos*

FIGURA 26 - Formulário para cadastro dos planejamentos realizados em um projeto

O formulário é usado apenas para descrever os planejamentos realizados e atribuir um identificador único a cada um. Após ser cadastrado, cada planejamento de projeto deverá conter informações sobre estimativas de esforço, defeitos, revisões, tamanho, progresso, cronograma e

estabilidade de requisitos. Convenções e formatos para registro dessas estimativas são apresentados nas subseções a seguir.

Em um mesmo projeto pode ser registrado mais de um planejamento, já que é comum ocorrerem replanejamentos (nesses casos, deve-se registrar o novo planejamento separadamente do anterior, mantendo este para fins de histórico). No entanto, é obrigatório o registro de pelo menos um planejamento para cada projeto.

Ao contrário das medidas explanatórias, que são registradas tipicamente por eventos (dia de trabalho, defeito detectado, revisão realizada, etc.), a maior parte das medidas preditivas é coletada apenas no nível de iterações e fluxos do processo, como veremos a seguir. A opção por esse nível de granulosidade deve-se a dois motivos:

- Iterações e fluxos constituem as menores subdivisões formais de um processo de desenvolvimento que siga uma estrutura matricial semelhante à do Processo Unificado;
- É esse o nível geralmente tratado nos planejamentos dos projetos.

A opção pelo nível de granulosidade foi tomada considerando também que as estimativas são obtidas através de métodos técnicos, e não por previsões pessoais baseadas em intuição ou critérios puramente subjetivos (casos em que seria mais interessante utilizar itens mais granulosos, já que estimativas pessoais são pouco acuradas quando utilizadas para prever grandes blocos de trabalho).

4.1.7.1 Estimativa de tamanho, progresso e cronograma

As estimativas de tamanho, progresso e cronograma são registradas de forma idêntica à utilizada para os valores reais, obtidos ao longo da execução do projeto. Aqui também é utilizado o conceito de marcos de projeto, que representam pontos intermediários de controle. A única diferença é que os marcos tratados aqui constituem marcos planejados, e não marcos reais. Isso deve ser informado no cabeçalho do formulário (é utilizado aqui o mesmo formulário apresentado nas seções 4.1.1.2 e 4.1.4.1), que deve destacar a natureza dos dados como sendo planejados, e informar o planejamento ao qual pertence a estimativa registrada. Essas informações estão destacadas no formulário reproduzido na FIGURA 27.

O preenchimento dos valores planejados para tamanho, estado e progresso dos requisitos é feito de forma idêntica ao que é feito para os dados reais, utilizando o cadastro de requisitos como base para o registro das medidas.

Deve haver, no mínimo, um marco planejado representando o final de cada iteração. Assim como nos dados reais, marcos intermediários podem ser opcionalmente criados. No entanto, vale ressaltar que o ideal é que o planejamento dos projetos divida as fases em iterações tão pequenas quanto for necessário para se obter a granulosidade desejada ao acompanhamento, dispensando a necessidade de marcos intermediários. Isso porque os marcos finais de iteração possuem critérios mais definidos que os intermediários, e conseqüentemente possuem um

significado mais forte para o acompanhamento dos projetos.

Ao longo da execução do projeto, é possível comparar os marcos planejados com os marcos reais. Essa comparação deve levar em consideração principalmente a data em que o marco foi atingido e o progresso obtido até então. Discrepâncias entre os valores reais e planejados indicam a necessidade de providências gerenciais, para evitar que os desvios comprometam o cumprimento dos compromissos assumidos.

Formulário para registro de marco de projeto

Organizações ProSoft LTDA

Nome do projeto	*Locadora 1.0*
Nome do marco de projeto	*Fim da iteração LR*
Data do marco	*15/07/2002*
Natureza dos dados	*Planejados*
Identificador do planejamento	*PL_AR_001*

Situação dos requisitos

Requisito	**Descrição**	**Tamanho**	**Estado**	
CUA1	*Gestão do registro de esforço*	*31*	*Levantado*	*20%*
CUA2	*Gestão do cadastro de projetos*	*45*	*Levantado*	*20%*
CUA3	*Gestão de usuários*	*19*	*Levantado*	*20%*
CUA4	*Gestão do registro de defeitos*	*40*	*Levantado*	*20%*
CUA5	*Gestão do registro de revisões*	*35*	*Levantado*	*20%*
CUA6	*Gestão do cadastro de requisitos*	*20*	*Levantado*	*20%*
CUA7	*Gestão do cadastro de planejamentos*	*25*	*Levantado*	*20%*
CUA8	*Gestão de cronograma de projeto*	*24*	*Levantado*	*20%*
CUA9	*Registro de marco de projeto*	*27*	*Levantado*	*20%*
CUA10	*Registro de métricas de artefatos*	*21*	*Levantado*	*20%*
CUA11	*Gestão de alterações em requisitos*	*26*	*Levantado*	*20%*
CUA12	*Registro de estimativa de esforço*	*18*	*Levantado*	*20%*
CUA13	*Registro de estimativa de defeitos*	*11*	*Levantado*	*20%*
CUA14	*Registro de estimativa de revisões*	*11*	*Levantado*	*20%*
CUA15	*Registro de estimativa de estabilidade de requisitos*	*7*	*Levantado*	*20%*
CUA16	*Gestão do cadastro de processos*	*22*	*Levantado*	*20%*
CUA17	*Gestão do cadastro de classificações de membros da equipe*	*19*	*Levantado*	*20%*

Tamanho total (PF brutos)	*401*
Fator de ajuste	*0,932*
Tamanho total (PF ajustados)	*373,7*

FIGURA 27 - Formulário para registro de marco de projeto, para um marco planejado

Além do registro dos marcos planejados de projeto, o tamanho dos artefatos também deve ser estimado para que seja posteriormente confrontado com os valores reais. Neste caso, o registro também é feito de forma idêntica à utilizada para os dados reais (FIGURA 11, apresentada na seção 4.1.1.2).

4.1.7.2 Estimativa de estabilidade de requisitos

Assim como em todos os outros grupos de medidas apresentadas, aquelas que dizem respeito à estabilidade de requisitos também devem ter valores estimados durante o planejamento dos projetos. Aqui, o mais importante é prever a quantidade esperada de requisitos alterados e o impacto previsto dessas alterações na produtividade da equipe, e acompanhar os valores obtidos ao longo da execução dos projetos.

O registro das estimativas é feito no formulário apresentado na FIGURA 28. Nele devem ser informados:

- **Projeto** – Identificador do projeto de desenvolvimento de software ao qual se refere a estimativa.
- **Identificador do planejamento** – Identificador do planejamento ao qual pertence a estimativa, dentre aqueles cadastrados no formulário para identificação de planejamento de projeto, ilustrado na FIGURA 26.
- **Percentual de requisitos alterados** – Valor estimado para a fração dos requisitos, dentre todos aqueles cadastrados para o projeto, que se espera sofrer alterações. Em outras palavras, representa o percentual estimado dos requisitos referenciados em pelo menos um registro de alteração (formulário apresentado na FIGURA 13). Deve levar em consideração apenas o número de requisitos alterados, desconsiderando o tamanho dos mesmos.
- **Percentual do esforço total dedicado a alterações em requisitos** – Representa, do total de horas dedicadas pela equipe ao projeto em questão, a fração estimada a ser consumida por atividades que tenham como objetivo adequar artefatos a alterações ocorridas em requisitos. Constitui uma estimativa do impacto das alterações de requisitos na produtividade da equipe.
- **Tamanho total de alterações em requisitos** – Valor estimado para o total de pontos de função ajustados representado pelas alterações em requisitos ocorridas ao longo do projeto. Posteriormente, pode ser confrontado com o somatório do tamanho informado em todos os registros de alteração de requisitos cadastrados para o projeto.

Formulário de registro de estimativa de estabilidade de requisitos Organizações ProSoft LTDA	
Nome do projeto	*Quantum 1.0*
Identificador do planejamento	*PL_AR_001*
Estimativa de estabilidade de requisitos	
% de requisitos alterados	*13,1%*
% do esforço total dedicado a alterações em requisitos	*11,6%*
Tamanho total de alterações em requisitos no projeto (PF)	*34,1*

FIGURA 28 - Formulário para registro de estimativa de estabilidade de requisitos

A Figura 29 apresenta a modelagem de classes utilizada para representar todas as estimativas produzidas em um planejamento de projeto.

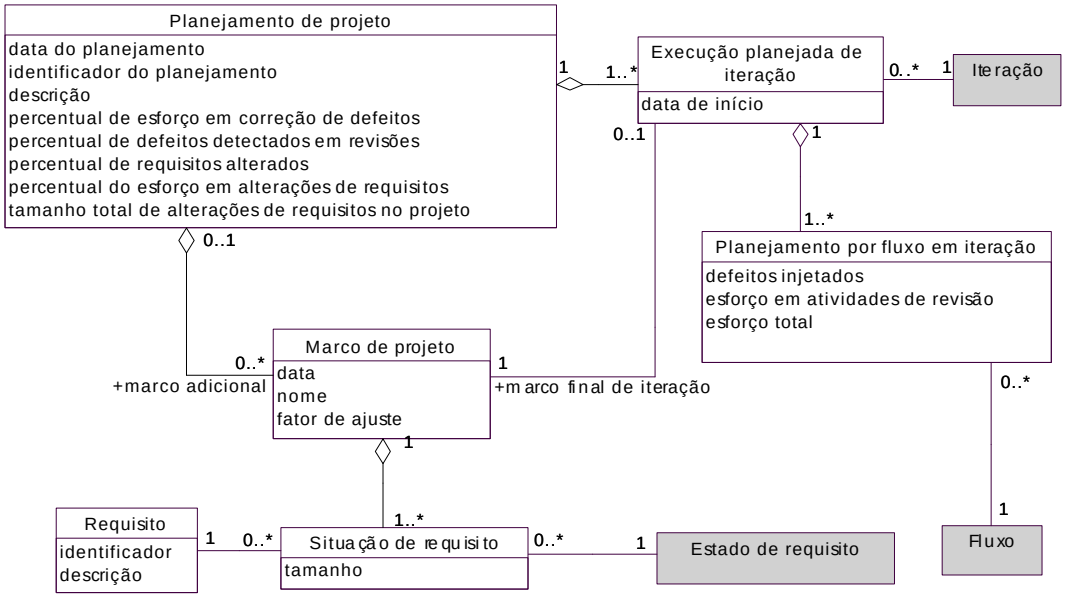


FIGURA 29 - Visão do registro das estimativas, sob a forma de um diagrama de classes

Comparando esse diagrama de classes com o da FIGURA 20, referente ao registro de cronograma durante a realização dos projetos, percebe-se que há uma grande semelhança entre ambos. Isso é natural, já que a forma proposta para os registros do cronograma real e do planejado é exatamente a mesma.

Por essa semelhança, é possível modelar a execução real e o planejamento dos projetos em um único diagrama utilizando o conceito de herança, como foi feito na FIGURA 30. A

classe “Realização ou planejamento de projeto” modela os aspectos mensuráveis comuns entre uma execução real e uma planejada. Cada projeto possui uma única realização, mas pode haver mais de um planejamento para um mesmo projeto.

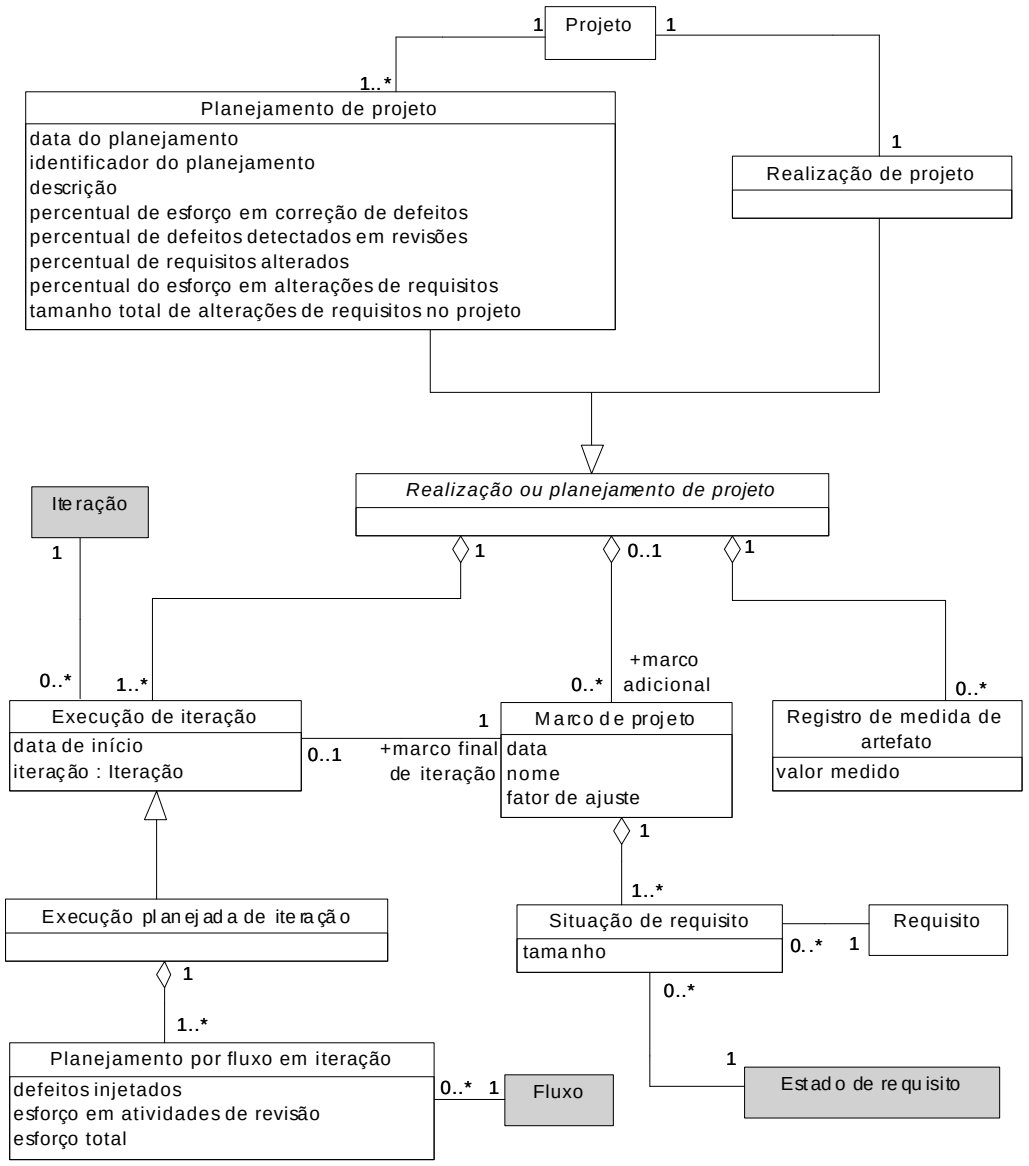


FIGURA 30 - Modelagem unificada da realização e do planejamento dos projetos

4.1.7.3 Estimativa de esforço

Durante o planejamento de um projeto, é necessário estimar não só o esforço total necessário para o desenvolvimento do produto, mas também a distribuição desse esforço pelas

fases e fluxos do processo.

A distribuição do esforço pelas iterações é necessária para permitir ao gerente de projeto dimensionar os recursos necessários para cada uma e obter, a partir dessa informação, o prazo necessário e o número de desenvolvedores a serem alocados em cada iteração.

A distribuição do esforço pelos fluxos, por sua vez, constitui um indicador da natureza do trabalho predominante em cada iteração. Isso é bastante útil para definir o perfil mais adequado de desenvolvedores para a equipe a cada momento, já que os fluxos são associados de forma razoavelmente direta a especializações de profissionais.

Os valores estimados devem ser registrados no formulário ilustrado na FIGURA 31. Nele, há uma matriz de iterações *x* fluxos, onde deve ser informada a quantidade prevista de horas trabalhadas para cada fluxo, em cada iteração. O esforço total por fluxo, por iteração e o esforço total do projeto podem ser facilmente obtidos através da soma dos valores informados em cada linha ou coluna da matriz.

Os valores estimados devem considerar todo o esforço previsto para cada fluxo e iteração, incluindo o esforço previsto para atividades de revisão.

Formulário de registro de estimativa de esforço

Organizações ProSoft LTDA

Nome do projeto	*Quantum 1.0*
Identificador do planejamento	*AR_09*

Estimativa de esforço por iteração e fluxo

Iteração/Fluxo	*RQ*	*AN*	*DS*	*IM*	*TS*	*ES*	*GP*	*GQ*	*EP*	**Total**
LR	*20*	*0*	*0*	*0*	*0*	*0*	*3*	*0*	*0*	*23*
AR	*40*	*20*	*5*	*0*	*0*	*0*	*8*	*4*	*0*	*77*
DI	*40*	*80*	*15*	*0*	*0*	*0*	*15*	*8*	*0*	*150*
L1	*5*	*10*	*80*	*50*	*10*	*10*	*8*	*10*	*0*	*183*
TA	*3*	*6*	*70*	*100*	*20*	*10*	*8*	*10*	*0*	*227*
TB	*0*	*0*	*5*	*20*	*30*	*0*	*6*	*8*	*0*	*69*
OP	*0*	*0*	*0*	*10*	*20*	*0*	*4*	*5*	*0*	*39*
Total	*0*	*0*	*0*	*10*	*20*	*0*	*4*	*3*	*0*	*37*

FIGURA 31 - Formulário para registro de estimativa de esforço

4.1.7.4 Estimativa de defeitos

Como foi apresentado na seção 4.1.5, a medição dos defeitos procura caracterizar dois aspectos do processo de desenvolvimento:

- A qualidade dos produtos desenvolvidos;
- O impacto dos defeitos na produtividade da equipe.

As estimativas de defeitos devem, portanto, refletir esses dois aspectos.

Quanto à caracterização da qualidade, o registro da estimativa de defeitos é bastante semelhante ao apresentado anteriormente para a estimativa de esforço. Deve-se registrar, para cada iteração e cada fluxo, o número esperado de defeitos injetados. Neste caso, a importância em distribuir os defeitos estimados pelas fases e fluxos do processo é permitir o acompanhamento da qualidade do produto em pontos intermediários do ciclo de desenvolvimento. Por exemplo, se o número de defeitos injetados em uma iteração se mostrar superior ao planejado, o gerente de projeto pode buscar ações para reduzir a incidência de defeitos em iterações futuras, de forma a evitar riscos quanto à qualidade do produto final. Nesse caso, o planejamento da distribuição em fluxos permite identificar quais são os fluxos responsáveis pelo índice de defeitos acima do planejado, e a partir dessa informação selecionar as ações mais adequadas.

Quanto ao impacto dos defeitos na produtividade da equipe, o indicador utilizado aqui é o percentual do esforço total que é dedicado à correção de defeitos. Nele deve-se informar, do esforço total planejado para o projeto (informado no formulário descrito na seção 4.1.7.1), qual a fração que se espera dedicar a atividades que tenham como objetivo corrigir os defeitos detectados. Quanto maior esse percentual, maior o impacto esperado dos defeitos na produtividade.

O formulário utilizado para o registro dessas estimativas está ilustrado na FIGURA 32.

Formulário de registro de estimativa de defeitos

Organizações ProSoft LTDA

Nome do projeto	*Quantum 1.0*
Identificador do planejamento	*AR_09*

Número estimado de defeitos injetados por iteração e fluxo

Iteração/Fluxo	*RQ*	*AN*	*DS*	*IM*	*TS*	*ES*	*GP*	*GQ*	*EP*	**Total**
LR	*6*	*7*	*0*	*0*	*0*	*0*	*0*	*0*	*0*	*13*
AR	*11*	*11*	*5*	*0*	*0*	*0*	*7*	*3*	*0*	*37*
DI	*3*	*4*	*11*	*12*	*9*	*5*	*3*	*1*	*0*	*48*
L1	*0*	*0*	*13*	*42*	*7*	*2*	*3*	*2*	*0*	*69*
TA	*0*	*0*	*0*	*19*	*11*	*4*	*3*	*0*	*0*	*37*
TB	*0*	*0*	*0*	*4*	*8*	*3*	*3*	*0*	*0*	*18*
OP	*0*	*0*	*0*	*3*	*0*	*3*	*3*	*0*	*0*	*9*
Total	*20*	*22*	*29*	*80*	*35*	*17*	*22*	*6*	*0*	*231*
% do esforço total dedicado a correções de defeitos					*14,3%*					

FIGURA 32 - Formulário para registro de estimativa de defeitos

4.1.7.5 Estimativa de revisões

Quanto às atividades de revisão, as estimativas devem caracterizar o esforço que se

pretende empregar nessas atividades e a eficácia que se espera obter quanto à detecção de defeitos, já que este constitui o principal objetivo das revisões.

O esforço estimado para essas atividades é registrado de forma similar ao registro da estimativa de esforço para o projeto, apresentado na seção 4.1.7.1: o registro é feito em uma matriz de fases *x* fluxos, onde é registrado o total de horas por fluxo, em cada fase. No entanto, aqui se deve registrar apenas o número de horas a ser dedicado às atividades de revisão. Como as revisões fazem parte dos fluxos de trabalho, esse valor sempre corresponderá a uma fração do que foi registrado nas estimativas de esforço da seção 4.1.7.1.

O formulário para registro das estimativas está apresentado na FIGURA 33. Preenchendo os campos da matriz, é possível obter facilmente o esforço total por fase e por fluxo (simplesmente somando as linhas e colunas). Durante a execução do projeto, é possível comparar o esforço planejado com o realizado, para verificar se as atividades de revisão estão sendo realmente realizadas conforme o que havia sido planejado. Quanto à eficácia esperada para essas atividades, basta informar, do número total de defeitos esperados para o projeto, qual a fração que se espera detectar nas revisões.

Além de ser um importante indicador do benefício trazido por essas atividades, as informações apresentas aqui podem ser avaliadas em conjunto com a estimativa do número de defeitos, permitindo a obtenção de medidas derivadas, como a produtividade estimada das revisões (em defeitos detectados/hora). É possível também comparar o esforço previsto para as revisões com o esforço total estimado para o projeto e obter medidas derivadas interessantes, como a fração do esforço total que se espera dedicar a atividades de revisão, que pode ser útil para prever o custo dessas atividades em um projeto.

Formulário de registro de estimativa de revisões

Organizações ProSoft LTDA

Nome do projeto	*Quantum 1.0*
Identificador do planejamento	*AR_09*

Esforço estimado para as atividades de revisão (em horas)

Iteração/Fluxo	*RQ*	*AN*	*DS*	*IM*	*TS*	*ES*	*GP*	*GQ*	*EP*	**Total**
LR	*8,0*	*4,0*	*1,0*	*0,0*	*0,0*	*0,0*	*2,0*	*1,0*	*0,0*	*16,0*
AR	*8,0*	*16,0*	*3,0*	*0,0*	*0,0*	*0,0*	*3,0*	*2,0*	*0,0*	*32,0*
DI	*1,0*	*2,0*	*16,0*	*10,0*	*2,0*	*2,0*	*2,0*	*2,0*	*0,0*	*37,0*
L1	*1,0*	*1,0*	*14,0*	*20,0*	*4,0*	*2,0*	*2,0*	*2,0*	*00,*	*46,0*
TA	*0,0*	*0,0*	*1,0*	*4,0*	*6,0*	*0,0*	*1,0*	*2,0*	*0,0*	*14,0*
TB	*0,0*	*0,0*	*0,0*	*2,0*	*4,0*	*0,0*	*1,0*	*1,0*	*0,0*	*8,0*
OP	*0,0*	*0,0*	*0,0*	*2,0*	*4,0*	*0,0*	*1,0*	*0,0*	*0,0*	*7,0*
Total	*19,0*	*23,0*	*35,0*	*38,0*	*20,0*	*4,0*	*13,0*	*10,0*	*00*	*162,0*
% esperado de defeitos detectados em revisões					*46,4%*					

FIGURA 33 - Formulário para registro de estimativa de revisões

4.1.8 Visão geral do modelo conceitual

Como conclusão do trabalho de definição das medidas básicas que compõem o modelo de medição, todas as medidas especificadas foram agrupadas em um diagrama que consolida os atributos identificados em uma estrutura única, explicitando os relacionamentos existentes entre as várias entidades envolvidas. É esse modelo conceitual que será utilizado como base para a definição das medidas derivadas na seção 4.3, que por sua vez permitirão a construção dos indicadores desejados.

Para viabilizar a apresentação de todas as classes envolvidas em um diagrama único sem sacrificar a clareza das informações, o diagrama não exibe explicitamente as classes correspondentes aos elementos configuráveis do modelo, que serão detalhadas na seção 4.2 (Processo, Fase, Fluxo, Espécie de Defeito, etc.). Os relacionamentos com essas classes foram modelados como atributos.

O diagrama com o modelo completo das medidas é apresentado na FIGURA 34, e consiste na consolidação de todos os diagramas parciais apresentados nas seções anteriores.

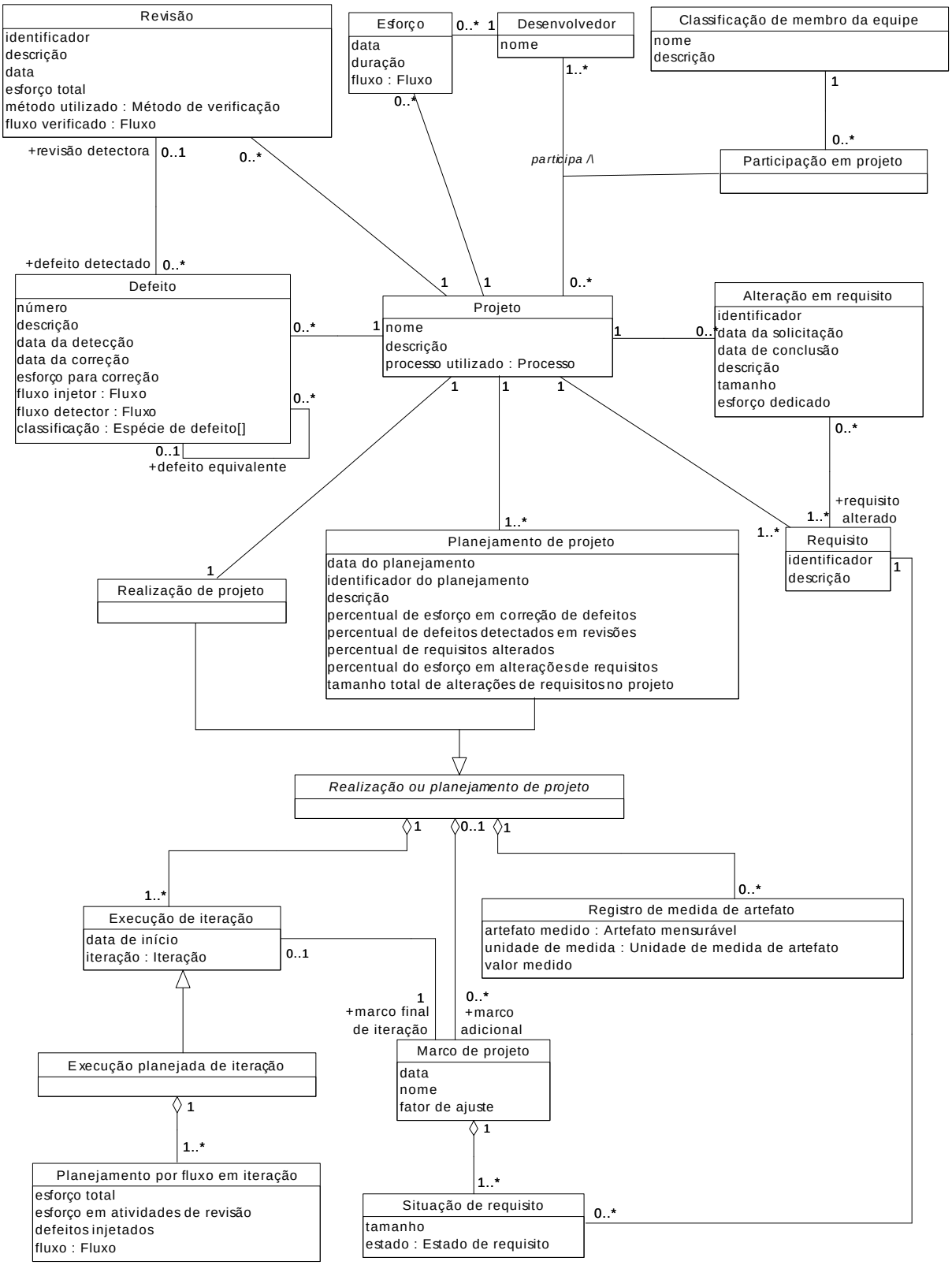


FIGURA 34 – Visão do modelo completo, contendo todas as medidas básicas a serem coletadas

Além de contribuir para que a formalização das medidas básicas tenha uma maior clareza e precisão, o diagrama apresentado pode ser utilizado também como base para a construção de uma estrutura de dados voltada para o armazenamento das medidas coletadas em uma base de dados informatizada. A ferramenta Quantum, descrita no Capítulo 6, teve sua modelagem das classes de entidade inteiramente baseada nessa estrutura.

4.2 Elementos configuráveis do modelo

Conforme colocado na seção 1.3, um dos requisitos para o modelo proposto é que seja flexível o suficiente para poder ser adaptado a diferentes processos de arquitetura matricial. Para que isso seja possível, os elementos do modelo que dependem diretamente de particularidades de cada processo devem ser configuráveis. Esta seção descreve o conjunto desses elementos, dividido em dois grupos:

- Elementos que modelam a **estrutura do processo**: a arquitetura do Processo Unificado, de onde foi retirada a nomenclatura usada neste trabalho, divide o ciclo de desenvolvimento em iterações e agrupa as atividades em fluxos. No entanto, o conjunto dessas iterações e fluxos pode variar em cada processo (ou mesmo em personalizações de um mesmo processo). Como será apresentado a seguir, o modelo apenas assume que existam fases, fluxos e iterações, e deixa a instanciação desses elementos a cargo de cada organização.
- Aspectos referentes à **estratégia de medição**: além da definição das fases e fluxos, há outros elementos importantes para a mensuração que podem variar de processo para processo. É o caso, por exemplo, das taxonomias utilizadas para classificação dos defeitos.

Em ambos os casos, os elementos foram modelados sob a forma de classes em notação UML. Quando o modelo é aplicado a um processo real, sua configuração consiste em instanciar essas classes com os valores aplicáveis em cada caso. Um exemplo de instanciação é apresentado no Capítulo 5.

4.2.1 Estrutura do processo

A estrutura do processo assumida pelo modelo de medição deriva da arquitetura definida pelo Processo Unificado, que divide o ciclo de desenvolvimento em fases (que por sua vez são divididas em iterações) e fluxos. Tal divisão é ortogonal, de forma que em uma iteração de uma fase podem ser realizadas atividades de diversos fluxos.

Como foi apresentado na seção 4.1, algumas das medidas propostas apresentam informações que as relacionam a esses elementos do processo de desenvolvimento. Afinal, um dos principais objetivos da mensuração é permitir que se tenha um conhecimento quantitativo dos processos utilizados. A categorização de medidas coletadas em fluxos e iterações permite a posterior análise comparativa do desempenho do processo em cada uma de suas subdivisões, possibilitando que sejam identificados os pontos que mais necessitam de melhorias.

A princípio, o modelo proposto pode ser aplicado a qualquer processo de desenvolvimento que seja estruturado sobre essa arquitetura matricial, independentemente de quais são as fases, fluxos e iterações existentes em cada um. A FIGURA 35 apresenta o diagrama de classes que modela essa estrutura.

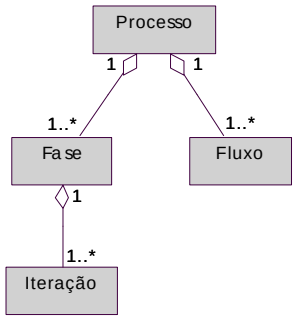


FIGURA 35 - Estrutura utilizada para modelar o processo de desenvolvimento

A Tabela 14 descreve o significado de cada classe, e apresenta exemplos de instâncias que poderiam ser criadas para cada uma.

Classe	Definição	Exemplos de instâncias
Processo	Processo de desenvolvimento em uso pela organização, e que será submetido à mensuração.	RUP, Praxis 1.0, Praxis 2.0, ou qualquer personalização desses processos.
Fase	Divisão maior de um processo, para fins gerenciais, que corresponde aos pontos principais de aceitação por parte do cliente.	Concepção, Elaboração, Construção e Transição.
Iteração	Passo constituinte de uma fase, no qual se atinge um conjunto bem definido de metas parciais de um projeto.	Para o processo Praxis 2.0 Padrão, por exemplo, as iterações seriam: Ativação, Levantamento dos Requisitos, Análise dos Requisitos, Desenho Implementável, Liberação 1, Liberação 2, Liberação 3, Testes Alfa e Operação Piloto [Paula03].
Fluxo	Subprocesso caracterizado por um tema técnico ou gerencial.	Para o RUP [KRUCHTEN00], por exemplo, as instâncias seriam Modelagem do negócio, Requisitos, Análise e Desenho, Implementação, Testes, Desdobramento, Gestão de configuração e alteração, Gestão de projeto, Ambiente.

TABELA 14 - Descrição das classes que modelam o processo de desenvolvimento

4.2.2 Aspectos referentes à política de medição

Além de assumir a estrutura apresentada na seção anterior para o processo de desenvolvimento, o modelo proposto utiliza alguns elementos genéricos que devem ser definidos para cada processo em que for aplicado. São conceitos que geralmente já existem em qualquer processo ou que podem ser definidos para fins de mensuração:

- Espécies de defeitos;
- Métodos de verificação usados em revisões;
- Estados de requisitos;
- Artefatos mensuráveis.

Para modelar esses elementos, foi criada a estrutura de classes apresentada na FIGURA 36. As seções a seguir descrevem com maiores detalhes esses conceitos.

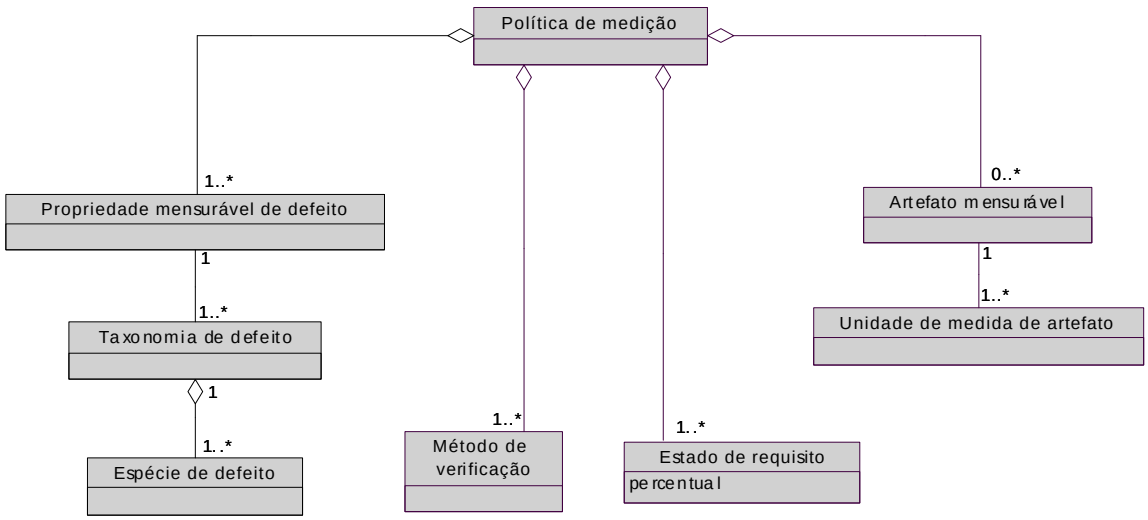


FIGURA 36 - Elementos gerais da estrutura de medição, que devem ser instanciados para cada processo de desenvolvimento

4.2.2.1 Espécies de defeitos

Para caracterizar os defeitos detectados ao longo de um projeto de desenvolvimento, é interessante classificá-los não só quanto às fases e fluxos em que são injetados, detectados e removidos, mas também quanto a outras características como gravidade (relacionada ao impacto causado pelo defeito) e natureza (que o classifica quanto à causa). Essa classificação mais detalhada permite conhecer os tipos mais comuns de defeitos em cada subdivisão do processo, o que auxilia a identificação de ações específicas que poderiam potencialmente reduzir o número de defeitos em projetos futuros.

Para permitir uma certa flexibilidade, o modelo não fixa as propriedades dos defeitos a serem mensuradas, tampouco as taxonomias a serem utilizadas para classificar os defeitos quanto a essas propriedades. Foi criada apenas uma estrutura a partir da qual esses elementos possam ser instanciados, composta pelos seguintes elementos:

- **Propriedade mensurável de defeito**: característica que deve ser observada nos defeitos detectados, para fins de mensuração. Exemplos de propriedades mensuráveis podem ser:
  - *Complexidade*, relacionada ao esforço necessário para a correção;
  - *Gravidade*, associada ao impacto na utilização do produto;
  - *Natureza*, relacionada à origem do defeito (ex: documentação, sintaxe, atribuição, etc.);
  - *Visibilidade*, referente à visibilidade do defeito para o cliente.
- **Taxonomia**: Definição de um conjunto de *espécies* a serem utilizadas para classificar um defeito quanto a uma propriedade mensurável.
- **Espécie de defeito**: Classificação possível para uma propriedade mensurável de defeito, dentro de uma taxonomia.

Por exemplo, para a *propriedade mensurável* "natureza", uma *taxonomia* aplicável seria a proposta em [HUMPHREY95], que classifica os defeitos em dez *espécies*: Documentação, Sintaxe, Versão, Atribuição, Interface, Checagem, Dado, Função, Sistema e Ambiente.

A associação de um defeito à espécie que o caracteriza quanto a uma propriedade mensurável faz parte do procedimento de registro de defeitos, descrito na seção 4.1.5.

4.2.2.2 Métodos de verificação usados em revisões

Como já foi colocado, um dos benefícios da coleta de dados quantitativos sobre revisões é a possibilidade de comparar diferentes métodos de verificação quanto à produtividade e à eficácia obtidas na detecção de defeitos. Para que isso seja possível, na seção 4.1.6 foi definido um procedimento para registro de revisões, onde ficou estabelecido que cada revisão realizada deve ser registrada e classificada quanto ao método de verificação utilizado.

Como os métodos de verificação podem variar de acordo com o processo utilizado pela organização, a estrutura de medição permite que eles sejam cadastrados e inseridos no modelo como instâncias da classe "Método de verificação". Inspeção, revisão técnica, revisão de apresentação e verificação automatizada são exemplos de possíveis instâncias. Informações sobre diferentes métodos de verificação utilizados em revisões podem ser obtidas em [PAULA03] e [HUMPHREY89].

4.2.2.3 Estados de requisitos

A medida utilizada para medir o progresso de um projeto em uma determinada data baseia-se na situação em que se encontram os requisitos, como foi descrito na seção 4.1.4. Para isso, é necessário que esses estados sejam cadastrados, informando-se o percentual de progresso representado por cada um.

Cada estado cadastrado corresponde a um estágio de desenvolvimento que cada requisito pode atingir durante o projeto e representa a fração já concluída de um requisito quando o mesmo se encontra naquele estado. Para o processo Praxis, por exemplo, os estados seriam: Identificado (10%), Detalhado (20%), Analisado (30%), Desenhado (40%), Especificado (50%), Realizado (60%), Implementado (70%), Verificado (80%), Validado (90%) e Completo (100%) [PAULA03].

4.2.2.4 Artefatos mensuráveis

O modelo de medição contempla também a possibilidade de se coletar algumas medidas sobre artefatos que são produzidos ao longo do ciclo de desenvolvimento. Medidas desse tipo podem ser usadas para caracterizá-los quanto a propriedades como tamanho (conforme descrito na seção 4.1.1) e complexidade.

O modelo já prevê uma contagem de tamanho expressa em termos do número de pontos de função dos produtos, mas outros tipos de contagem poderiam ser realizados, com base nos artefatos gerados. Por exemplo, é possível contar o número de páginas de uma especificação de requisitos, a quantidade de classes existentes em um modelo, ou a quantidade de linhas de um código-fonte.

Diferentes razões podem motivar a mensuração de artefatos:

- Identificação de problemas: medições de propriedades de artefatos podem ser indicadores de problemas específicos, identificados através de comparações com os valores publicados na literatura. Por exemplo, um número de LC/PF (linhas de código por ponto de função) em um código-fonte significativamente superior ao que a literatura prevê para a linguagem adotada pode indicar problemas de implementação, como um domínio insuficiente da filosofia orientada a objetos. Em uma especificação de requisitos, um número muito baixo de páginas por ponto de função pode indicar documentação insuficiente dos requisitos identificados.
- Conformidade com padrões: existem alguns padrões internacionais que procuram uniformizar medidas usadas pela indústria de software, com o objetivo de possibilitar a comparação dos dados de diferentes organizações. Alguns desses padrões utilizam medições de propriedades de artefatos. É o caso, por exemplo, do padrão para a medição de produtividade proposto pelo IEEE [IEEE92], que adota a medição do tamanho de alguns artefatos: o código-fonte é medido pela quantidade de linhas de código, e documentos textuais são medidos pelo número de páginas e

pela quantidade de palavras, gráficos e ideogramas.

- Validação: medidas de artefatos podem ser comparadas a outras medições como forma de validá-las. Por exemplo, a contagem do número de linhas de código dos produtos pode ser comparada com as respectivas contagens de pontos de função, para verificar se a correlação esperada entre elas realmente existe.

A configuração da medição dos artefatos consiste na definição de quais deles serão submetidos à revisão (que no modelo correspondem a objetos da classe “Artefato mensurável”) e, para cada um, as medidas que serão coletadas (representadas por instâncias da classe “Unidade de medida de artefato”).

4.3 Definição de medidas derivadas

Os elementos de dados formalizados na seção 4.1 constituem em sua maioria *medidas básicas*, ou seja, atributos que podem ser mensurados a partir de observação direta. Dados desse tipo, se analisados individualmente, não trazem muita informação útil para o conhecimento dos produtos ou para a melhoria dos processos de uma organização, pois apenas caracterizam um aspecto isolado do desenvolvimento de um produto.

Para que sejam transformados em informações úteis, diferentes elementos devem ser combinados entre si, formando as chamadas *medidas derivadas*. Estas, por sua vez, serão utilizadas como base para a construção de *indicadores*, como aqueles apresentados no Apêndice A.2. Essa construção de medidas derivadas e indicadores é baseada na estrutura proposta pelo PSM e pelo padrão ISO/IEC 15939, descrita na seção 2.3.1.

Por exemplo, uma medida básica que faz parte do modelo proposto na seção 4.1 é a quantidade de horas dedicada por cada membro da equipe a cada projeto, em cada dia de trabalho. Combinando-se todos os registros de esforço, é possível obter então o esforço total dedicado ao projeto (o que já pode ser considerado uma medida derivada, já que foi calculado a partir da combinação de vários registros de esforço). O esforço total, por sua vez, pode ser dividido pelo tamanho do produto, dando origem à medida derivada *produtividade*. Essa medida pode ser calculada para todos os projetos e, combinada com a produtividade planejada (outra medida derivada), permite a construção do indicador *I-7 - Evolução da produtividade*, descrito no Apêndice A.2.

O mesmo princípio pode ser usado para criar inúmeras combinações, cada uma gerando diferentes medidas derivadas. Aqui, essas medidas não serão tratadas com o mesmo nível de detalhes e formalismo com que foram descritas as medidas básicas na seção 4.1. Isso porque não existe um conjunto fixo delas a ser gerado: medidas básicas podem ser combinadas entre si de diversas maneiras, e cada uma delas pode ser útil para satisfazer determinadas necessidades de

informação. Aqui serão apenas sugeridas algumas medidas derivadas importantes para se construir os indicadores descritos no Apêndice A.2. São elas:

- Tamanho do produto em um marco de projeto;
- Esforço acumulado até um marco de projeto;
- Progresso obtido em um marco de projeto;
- Valor obtido em um marco de projeto (que difere do progresso por só considerar os requisitos que obtiveram 100% de progresso);
- Esforço total dedicado a uma subdivisão do processo;
- Densidade de defeitos injetados em uma subdivisão do processo;
- Percentual de defeitos de determinada espécie injetados em uma subdivisão do processo;
- Percentual de defeitos de determinada espécie detectados em uma subdivisão do processo;
- Produtividade obtida em uma subdivisão do processo;
- Percentual de retrabalho representado pelas correções de defeitos;
- Percentual de retrabalho representado pelas alterações em requisitos;
- Percentual total de retrabalho;
- Percentual de defeitos detectados em revisões que utilizam determinado método de verificação, em uma determinada subdivisão do processo;
- Percentual de requisitos alterados;
- Percentual representado pelo tamanho total de todas as alterações, comparado ao tamanho total do produto;
- Esforço médio para correção de defeitos de determinada espécie;
- Esforço médio para a implantação de alterações em requisitos injetados em determinada iteração do processo;
- Erro de determinada estimativa, calculado a partir do valor estimado e do valor obtido na prática.

Uma descrição de como cada uma dessas medidas derivadas pode ser calculada, bem como a associação delas com os indicadores, é apresentada no Apêndice A.3.

Capítulo 5

Aplicação do modelo ao processo Praxis

Este capítulo exemplifica como o modelo de medição proposto pode ser aplicado a um processo de desenvolvimento concreto. Isso é feito através da sua instanciação para o processo Praxis 2.0, descrito em [PAULA03].

Dois foram os objetivos que motivaram a instanciação do modelo para o processo Praxis:

- Exemplificar como os elementos configuráveis do modelo, apresentados na seção 4.1, podem ser definidos para adequá-lo ao contexto de um processo real;
- Fornecer os subsídios necessários para a elaboração de uma personalização do Praxis que contemple as práticas de mensuração, já que o processo atualmente não possui uma política de medição bem definida.

A seção 5.1 descreve brevemente o Praxis, e a aplicação do modelo de medição a esse processo é apresentada na seção 5.2.

5.1 O processo Praxis

O Praxis (PRocesso para Aplicativos eXtensíveis e InterativoS) constitui um processo de desenvolvimento de software desenhado para projetos com duração de seis meses a um ano, realizados individualmente ou por pequenas equipes [PAULA03]. É voltado para o desenvolvimento de aplicativos gráficos interativos, baseados na tecnologia orientada a objetos.

Seguindo a arquitetura definida no Processo Unificado [JACOBSON98], o Praxis propõe um ciclo de vida composto por fases, divididas em uma ou mais iterações, e fluxos, que correspondem a disciplinas da Engenharia de Software. A TABELA 15 apresenta a divisão em fases e iterações, enquanto a organização em fluxos é descrita na TABELA 16.

Fase	Iteração	Descrição
Concepção	Ativação	Levantamento e análise das necessidades dos usuários e conceitos da aplicação, em nível de detalhe suficiente para justificar a especificação de um produto de software.
Elaboração	Levantamento de Requisitos	Levantamento das funções, interfaces e requisitos não-funcionais desejados para o produto.
	Análise de Requisitos	Modelagem conceitual dos elementos relevantes do domínio do problema e uso desse modelo para validação dos requisitos e planejamento detalhado da fase de Construção.
Construção	Desenho Implementável	Definições interna e externa dos componentes de um produto de software, em nível suficiente para decidir as principais questões de arquitetura e tecnologia e para permitir o planejamento detalhado das liberações.
	Liberação 1	Implementação de um subconjunto de funções do produto que será avaliado pelos usuários.
	Liberação ...	Idem.
	Testes Alfa	Realização dos testes de aceitação, no ambiente dos desenvolvedores, juntamente com elaboração da documentação de usuário e possíveis planos de Transição.
Transição	Testes Beta	Realização dos testes de aceitação no ambiente dos usuários.
	Operação Piloto	Operação experimental do produto em instalação piloto do cliente, com a resolução de eventuais problemas através de processo de manutenção.

TABELA 15 - Fases e iterações do Praxis 2.0 [PAULA03]

Natureza	Fluxo	Descrição
Técnicos	Requisitos	Fluxo que visa a obter um conjunto de requisitos de um produto, acordado entre cliente e fornecedor.
	Análise	Fluxo que visa a detalhar, estruturar e validar os requisitos de um produto, em termos de um modelo conceitual do problema, de forma que eles possam ser usados como base para o planejamento e controle detalhados do respectivo projeto de desenvolvimento.
	Desenho	Fluxo que visa a formular um modelo estrutural do produto que sirva de base para a implementação, definindo os componentes a desenvolver e a reutilizar, assim como as interfaces entre si e com o contexto do produto.
	Implementação	Fluxo que visa a detalhar e implantar o desenho através de componentes de código e de documentação associada.
	Testes	Fluxo que visa a verificar os resultados da implementação, através do planejamento, desenho e realização de baterias de testes.
	Engenharia de sistemas	Fluxo que abrange atividades relativas ao desenvolvimento do sistema no qual o produto de software está contido; por exemplo, modelagem de processos de negócio, implantação, usabilidade e criação de conteúdo.
Gerenciais	Gestão de projetos	Fluxo que visa a planejar e controlar os projetos de software.
	Gestão da qualidade	Fluxo que visa a verificar e assegurar a qualidade dos produtos e processos de software.
	Engenharia de processos	Fluxo que visa a manter, dar suporte e promover melhorias nos próprios processos de software.

TABELA 16 - Fluxos técnicos e gerenciais do Praxis 2.0 [PAULA03]

Apesar de ter sido originalmente destinado à utilização em projetos didáticos em disciplinas de Engenharia de Software, o processo vem sendo usado com sucesso também em projetos maiores, alguns envolvendo equipes de até dezenas de pessoas.

5.1.1 Medições no Praxis

O Praxis define algumas medições que devem ser realizadas ao longo dos projetos. Elas estão presentes de forma mais explícita em dois dos fluxos gerenciais:

- **Gestão de projetos** – medições de tamanho e esforço são utilizadas para gerar informações sobre produtividade e subsidiar o planejamento dos projetos. Para o controle dos projetos há também um acompanhamento quantitativo do progresso e da estabilidade dos requisitos.
- **Engenharia de processos** – esse fluxo, adicionado ao processo em sua segunda versão, prevê atividades de mensuração como parte das atividades de melhoria de processo. Aqui, o objetivo não é mais controlar os projetos, mas conhecer quantitativamente o processo visando às atividades de melhoria.

Dentro da classificação estabelecida pelo CMMI, apresentada no Capítulo 2, a mensuração prevista no fluxo de Gestão de Projetos está relacionada ao nível 2, enquanto na Engenharia de Processos as atividades de medição já se voltam mais às características do nível 4.

Muitas das medições propostas, no entanto, não são completamente definidas pelo processo. É o caso, por exemplo, das medidas de esforço: o Praxis prevê que o esforço dedicado a cada fase e fluxo seja contabilizado, mas não define como a coleta deve ser feita no curso dos projetos. Além disso, o processo recomenda que seja criada uma base de dados históricos de projetos, mas não entra em detalhes sobre como essa base deve ser construída.

A ausência dessas características é aceitável, uma vez que a proposta original do Praxis estava voltada para utilização em projetos menores, de caráter didático. No entanto, à medida que o processo vem sendo utilizado em projetos maiores, tais limitações começam a se tornar mais significativas. A seção seguinte descreve como o modelo de medição proposto neste trabalho pode ser utilizado para incorporar ao Praxis um tratamento mais completo das práticas de medição.

5.2 Instanciação do modelo de medição

Para adequar o modelo de medição ao contexto do Praxis, a primeira providência necessária é instanciar os elementos que modelam a arquitetura do processo, a partir da estrutura apresentada na seção 4.2.1. Essa é uma tarefa bastante simples, já que o Praxis herdou do Processo Unificado a estrutura matricial que já divide o ciclo de desenvolvimento em uma estrutura composta por fases e fluxos. Podemos definir as instâncias das classes que descrevem a estrutura do processo diretamente a partir da definição do Praxis descrita na seção anterior. A TABELA 17 enumera o resultado obtido.

Classe	Instâncias no Praxis 2.0
Processo	Apenas uma instância, que corresponde ao próprio Praxis 2.0 padrão.
Fase	Concepção, Elaboração, Construção e Transição.
Iteração	Ativação, Levantamento dos Requisitos, Análise dos Requisitos, Desenho Implementável, Liberação 1, Liberação 2, Liberação n, Testes Alfa e Operação Piloto.
Fluxo	Requisitos, Análise, Desenho, Implementação, Testes, Engenharia de sistemas, Gestão de projetos, Gestão da qualidade, Engenharia de processos.

TABELA 17 - Instanciação das classes que modelam a estrutura do processo, para o Praxis 2.0

Vale ressaltar que as instâncias apresentadas se referem ao processo Praxis 2.0 padrão e que diferenças podem surgir em personalizações desse processo. As regras de personalização do Praxis permitem a alteração dos fluxos e iterações que compõem o processo, e alterações desse tipo devem ser refletidas no modelo de medição.

Feita a instanciação das classes que tratam da estrutura do processo, é necessário definir alguns detalhes da coleta dos diferentes tipos de medida. As seções a seguir tratam dessas definições, e mostram relacionamentos entre algumas medidas previstas no modelo e artefatos do Praxis a partir dos quais podem ser diretamente coletadas. Na medida em que as definições são descritas, os elementos configuráveis do modelo (espécies de defeitos, métodos de verificação, estados de requisitos e artefatos mensuráveis) vão sendo instanciados.

5.2.1 Tamanho

Poucos ajustes são necessários para implementar a medição de tamanho no Praxis, uma vez que a estratégia adotada nesse processo para acompanhamento do tamanho constituiu a principal referência para o modelo de medição proposto.

O Cadastro de Requisitos de Software (CRSw), que constitui um dos artefatos gerenciais do Praxis, já contém uma lista completa dos requisitos. Essa lista pode ser diretamente transposta para o formulário de registro de marco de projeto, descrito na seção 4.1.1. Outro artefato, a MPPSw (Memória de Planejamento de Projeto de Software), contém as informações sobre o tamanho de cada requisito em pontos de função não ajustados e o valor do fator de ajuste a ser utilizado.

Resta definir, então, as medições de tamanho a serem realizadas diretamente sobre os artefatos, já que o Praxis padrão não define medidas desse tipo. Como uma das características desse processo é a conformidade com padrões internacionais, como os padrões do IEEE, uma alternativa é definir a mensuração de artefatos com base em padrões desse tipo. Uma referência aplicável é o padrão para medidas de produtividade do IEEE [IEEE92], que define algumas medições em artefatos para subsidiar o cálculo da produtividade. Tais medições se baseiam principalmente na contagem de linhas de código e na medição do tamanho de documentos, através de critérios como o número de páginas e a quantidade de palavras.

A TABELA 18 apresenta o conjunto de medidas a ser coletado para o Praxis para documentos e modelos, com base no padrão citado. O que esse padrão não define é a medição de outros modelos além do próprio código-fonte, como os modelos de análise ou desenho. Como esse tipo de artefato compreende alguns dos principais resultados do processo, a tabela propõe também algumas medidas que podem ser extraídas de modelos.

Tipo de artefato	Artefato mensurável	Unidade de medida
Documento	ERSw (Especificação de Requisitos de Software)	Número de páginas.
		Número de palavras, segundo as convenções definidas em [IEEE92].
		Número de figuras, segundo as convenções definidas em [IEEE92].
	DDSw (Descrição de Desenho de Software)	Número de páginas.
		Número de palavras, segundo as convenções definidas em [IEEE92].
		Número de figuras, segundo as convenções definidas em [IEEE92].
	DTSw (Descrição de Testes de Software)	Número de páginas.
		Número de palavras, segundo as convenções definidas em [IEEE92].
		Número de figuras, segundo as convenções definidas em [IEEE92].
Modelo	CFSw (Código-fonte)	Número de linhas de código (excluindo comentários e linhas em branco)
		Número de linhas de comentário (excluindo linhas em branco)
	MASw (Modelo de Análise de Software)	Número de classes.
		Número de atributos.
		Número de operações.
	MDSw (Modelo de Desenho de Software)	Número de classes.
		Número de atributos.
		Número de operações.
	BTRSw (Bateria de Testes de Regressão)	Número de casos de teste.

TABELA 18 - Artefatos mensuráveis e respectivas unidades de medida propostos para o processo Praxis

Apesar do grande número de medidas, o custo para a coleta é bastante baixo porque normalmente as ferramentas utilizadas para a confecção dos artefatos já oferecem recursos para a coleta automatizada desses dados.

Há ainda um terceiro tipo de artefato no Praxis: os relatórios. Eles não foram considerados na mensuração porque seu principal objetivo é simplesmente relatar as conclusões de determinadas atividades no projeto. A medição do tamanho desse tipo de artefato não oferece muita informação útil.

5.2.2 Estabilidade de requisitos

Uma das atividades do subfluxo de Controle de Projetos do Praxis (pertencente ao fluxo de Gestão de Projetos) é a *monitoração do escopo*, que tem como principal objetivo o registro das alterações feitas nos requisitos e a medição da estabilidade destes. No processo

padrão, essa atividade já prevê que todas as alterações ocorridas nos requisitos após o término da fase de Elaboração sejam registradas nos relatórios de acompanhamento dos projetos (RAPSw). Neles, entre outras informações, são registrados a data da realização de cada alteração, a relação de requisitos que sofreram impacto por cada uma e o esforço adicional empregado na realização das alterações necessárias nos artefatos.

Para completar as informações previstas no modelo de medição, descritas na seção 4.1.2, basta que se colete também, para cada alteração, a data de solicitação pelo cliente e o tamanho da mesma, em pontos de função ajustados. Vale lembrar que, nesse caso, a contagem do tamanho deve utilizar a técnica de contagem de pontos de função específica para alterações [LONGSTREET02].

5.2.3 Esforço

O Praxis padrão prevê um acompanhamento quantitativo do esforço nos relatórios de acompanhamento de projeto (RAPSw), emitidos ao final de cada iteração. Porém, não define precisamente como essas medidas devem ser coletadas ao longo dos projetos. Os procedimentos de medição de esforço propostos no modelo aqui apresentado (seção 4.1.3) podem ser usados como entrada para a emissão desses relatórios.

Como o registro de esforço não utiliza outro elemento configurável do processo além da definição dos fluxos, descrita na seção anterior, não há outras classes do modelo a serem instanciadas para esse tipo de medição.

Uma ressalva merece ser colocada sobre a classificação do esforço quanto ao fluxo, quando se tratar do esforço dedicado a atividades de revisão: no Praxis padrão todas as revisões são atividades dos respectivos fluxos, e o esforço deve ser registrado como tal. Apenas a *gestão* das revisões (planejamento, convocação, pós-processamento) é que faz parte do fluxo de Gestão da Qualidade.

5.2.4 Progresso e cronograma

Assim como a medição do tamanho, o acompanhamento de progresso e cronograma previsto no modelo proposto (descrito na seção 4.1.4) foi inteiramente baseado na estratégia adotada pelo Praxis. O processo inclusive já define os estados possíveis para os requisitos: *Identificado* (10%), *Detalhado* (20%), *Analisado* (30%), *Desenhado* (40%), *Especificado* (50%), *Realizado* (60%), *Implementado* (70%), *Verificado* (80%), *Validado* (90%) e *Completo* (100%). A descrição do significado de cada um desses estados é apresentada em [PAULA03].

Como o único dado necessário ao acompanhamento do progresso além da medição do tamanho é a informação sobre os estados dos requisitos, podemos considerar que o Praxis padrão já implementa essa parte do modelo. Basta transferir as informações do CRSw para a base de

dados de medição (o que pode até ser feito de forma automatizada, com o auxílio de alguma ferramenta).

Quanto ao registro de cronograma, as datas de início e conclusão de cada iteração podem ser retiradas dos relatórios de acompanhamento de projeto (RAPSw), ou do relatório final de projeto (RFPSw). O Praxis padrão não adota marcos de projeto internos às iterações.

5.2.5 Defeitos

Para implantar a medição dos defeitos, a primeira providência necessária é definir precisamente quando uma imperfeição detectada em algum artefato será considerada um defeito. Dos critérios apresentados na seção 4.1.5 podemos afirmar que, no contexto do Praxis:

- A opção pelo registro apenas dos defeitos detectados em iterações seguintes às que foram injetados não é suficiente, pois a maior parte dos defeitos detectados em revisões não seria incluída.
- O critério de considerar apenas problemas detectados em artefatos já colocados sob gestão de configurações também não é adequado – o Praxis padrão não define uma política de gestão de configurações interna às iterações, e por isso este critério se torna equivalente ao anterior.

Um critério adequado, então, seria considerar para fins de medição apenas os defeitos que se enquadrem em pelo menos uma das seguintes situações:

- Problemas detectados em iterações seguintes àquelas em que foram injetados;
- Defeitos detectados em procedimentos de revisão e auditoria (independentemente de quando foram injetados);
- Defeitos registrados nos relatórios de teste.

Além dessas definições, é necessário também instanciar as classes do modelo que se referem à classificação dos defeitos em espécies, criando um mecanismo único para caracterizar todos os defeitos registrados.

Das três situações citadas, o Praxis padrão só prevê a classificação daqueles detectados em revisões. Mesmo para esses defeitos as propriedades consideradas dependem do tipo de revisão: revisões técnicas classificam defeitos por natureza e gravidade, enquanto as inspeções classificam apenas por natureza, utilizando uma taxonomia diferenciada. Para implementar a mensuração, é importante criar uma estrutura única de classificação que seja aplicável a todos os casos. A estrutura proposta é descrita a seguir.

Todos os defeitos registrados devem ser classificados quanto a duas propriedades: *gravidade* e *natureza*. Quanto à gravidade, o Praxis já prevê uma taxonomia para os defeitos detectados em revisões técnicas, descrita na TABELA 19. A utilização dessa taxonomia deve ser

estendida a todos os defeitos registrados.

Espécie	Descrição
Crítico	Defeito que reflete a ausência de um requisito essencial ou impede a utilização do produto. Não pode ser contornado, exigindo ação corretiva imediata.
Maior	Defeito que reflete a ausência de um requisito importante ou que afeta, mas não impede, a utilização do produto. Exige ação corretiva tão logo possível.
Menor	Todos os outros problemas, como erros de documentação, erros nas mensagens do produto,etc. Algumas correções podem ser deixadas para versões futuras.

TABELA 19 - Taxonomia única para classificação de defeitos quanto à gravidade [PAULA03]

Para a classificação quanto à natureza, duas taxonomias podem ser utilizadas. Para defeitos detectados em documentos, pode-se utilizar a taxonomia definida nos relatórios de revisão técnica, que define cinco espécies: *omissão*, *violação*, *imprecisão*, *estilo* e *outro*. Para aqueles detectados em modelos, pode-se utilizar a taxonomia prevista nos relatórios de inspeção, que consiste em uma adaptação dos tipos de defeitos definidos em [HUMPHREY95]. Ambas as taxonomias são descritas na TABELA 20.

Taxonomia	Espécie	Descrição
Natureza de defeitos em documentos	Omissão	Ausência de uma seção, requisito, funcionalidade ou algum outro item qualquer no documento.
	Violação	Incoerência de um item no documento com um outro documento do projeto elaborado anteriormente, ou ainda alguma incorreção percebida no documento.
	Imprecisão	Requisito, seção ou funcionalidade do documento, especificado de forma ambígua ou incompleta.
	Estilo	Erros de português, de formatação etc.
	Outro	Defeitos que não se enquadrem nos demais tipos.
Natureza de defeitos em modelos	Usabilidade	Defeitos de desenho nas interfaces e funções do produto.
	Documentação	Comentários, mensagens, padronização.
	Sintaxe	Grafia, pontuação, digitação, formatos de instruções.
	Construção	Uso de bibliotecas, uso de componentes, coesão, acoplamento.
	Atribuição	Declarações, nomes, escopo, limites.
	Interface	Chamadas de métodos e procedimentos, entrada e saída.
	Verificação	Mensagens de erro, falhas de verificação.
	Dados	Estrutura, conteúdo.
	Função	Lógica, apontadores, malhas, recursão, cálculos.
	Sistema	Configuração, temporizações, memória.
	Ambiente	Falhas nas ferramentas ou ambiente de desenolvimento.

TABELA 20 - Taxonomias para classificação de defeitos quanto à natureza, adaptado de [PAULA03]

Para os defeitos detectados em revisões técnicas, inspeções e testes, algumas das medidas a serem coletadas podem ser obtidas diretamente a partir dos relatórios produzidos nessas atividades (RRSw, RISw e RTSw). Para os demais defeitos não há artefatos do processo que já contemplem as medições propostas.

5.2.6 Revisões

O Praxis prevê os seguintes métodos de verificação, que podem ser utilizados em revisões formais de artefatos: *revisão técnica*, *inspeção*, *revisão de apresentação*, *revisão gerencial* e *auditoria de qualidade*.

Para implementar a mensuração, todas as revisões desses tipos devem ser cadastradas no formulário descrito na seção 4.1.6. Apenas as revisões informais (programação em pares, revisão individual e revisão preliminar) não devem ser registradas para fins de mensuração.

Para as revisões técnicas e inspeções, os relatórios previstos no Praxis (RRSw e RISw, segundo a nomenclatura do processo) já contêm todas as informações necessárias para o preenchimento do formulário.

5.2.7 Planejamento de projetos

O Praxis já prevê o registro das estimativas consideradas durante o planejamento dos projetos. Tal registro é feito na Memória de Planejamento de Projeto de Software (MPPSw), um dos modelos gerenciais previstos no processo.

A MPPSw contém algumas das informações sobre estimativas necessárias ao modelo de medição, mas não todas. As medidas não contempladas deverão ser incorporadas ao processo através de alguma personalização. Abaixo há uma descrição da situação do Praxis quanto à coleta de cada um dos tipos de estimativa previstos no modelo proposto:

- **Estimativa de esforço** – A MPPSw já possui uma seção para registro do esforço estimado para cada fluxo em cada iteração do processo, que já contém todas as informações previstas no formulário para registro de estimativa de esforço apresentado na seção 4.1.7.
- **Estimativa de defeitos** – O Praxis padrão não prevê a realização de uma estimativa da quantidade de defeitos inseridos ao longo dos passos do processo. Há uma seção nos relatórios e de testes (RTSw) para registro da quantidade prevista de defeitos, mas refere-se somente ao escopo das atividades de teste. Também não há um estimativa do percentual de retrabalho representado pela correção dos defeitos, como previsto no formulário para registro de estimativas de defeitos descrito na seção 4.1.7. Esse tipo de estimativa deverá ser incorporado às atividades de planejamento do fluxo de Gestão de Projetos.
- **Estimativa de revisões** – Assim como as estimativas de defeitos, as estimativas de revisões previstas na seção 4.1.7 não são contempladas pelo Praxis padrão, e deverão ser incorporadas ao processo.
- **Estimativa de tamanho, progresso e cronograma** – Os dados necessários ao

acompanhamento quantitativo do tamanho, progresso e cronograma ao longo dos marcos de projeto (que no Praxis padrão corresponde apenas ao final das iterações) já são contemplados na MPPSw. Basta que eles sejam transferidos à base de dados de medição.

- **Estimativa de estabilidade de requisitos** – O Praxis prevê o acompanhamento das alterações em requisitos, mas não contempla a estimativa da quantidade provável dessas alterações. Esse tipo de estimativa deve ser acrescentado às atividades de planejamento de projetos, como parte de uma personalização do processo.

Capítulo 6

A ferramenta Quantum

Uma das dificuldades da mensuração em software é o grande volume de informações gerado a partir das medições, principalmente no contexto dos grandes projetos de desenvolvimento. Coletar e armazenar de forma organizada esse volume de dados constituem tarefas caras, e o risco de ocorrência de erros no registro dos resultados das medições pode comprometer a confiabilidade das informações. Nesse contexto, torna-se essencial a existência de ferramentas informatizadas que forneçam suporte à coleta e ao registro das medidas pelas organizações.

Para viabilizar a aplicação do modelo de medição apresentado, este trabalho envolveu também a especificação e modelagem de uma ferramenta que possa oferecer apoio informatizado aos procedimentos propostos no Capítulo 4. A ferramenta, batizada de *Quantum*, é descrita na seção 6.1.

A seção 6.2 apresenta detalhes do processo de desenvolvimento utilizado para especificá-la e descreve algumas medições que foram realizadas durante sua elaboração, como forma de exercitar os procedimentos de mensuração propostos neste trabalho.

6.1 Descrição da ferramenta

O Quantum tem como objetivo oferecer apoio informatizado à coleta das medidas apresentadas no Capítulo 4. A idéia é prover interfaces gráficas para que os usuários possam registrar diretamente as medidas coletadas, dispensando assim a utilização de formulários em papel ou planilhas eletrônicas. Todos os dados informados são armazenados em um banco de dados centralizado, que pode ser utilizado como insumo para as atividades de análise.

Para cumprir esse objetivo, a ferramenta possui funcionalidades que contemplam o registro de todas as medidas propostas, transformando todos os formulários apresentados no Capítulo 4 em interfaces gráficas de usuário. Além disso, possui algumas funções de suporte, como a manutenção dos dados referentes aos usuários, aos projetos realizados e aos processos de

desenvolvimento utilizados. A FIGURA 37 apresenta todas as funções da ferramenta na forma de casos de uso em notação UML [RUMBAUGH99].

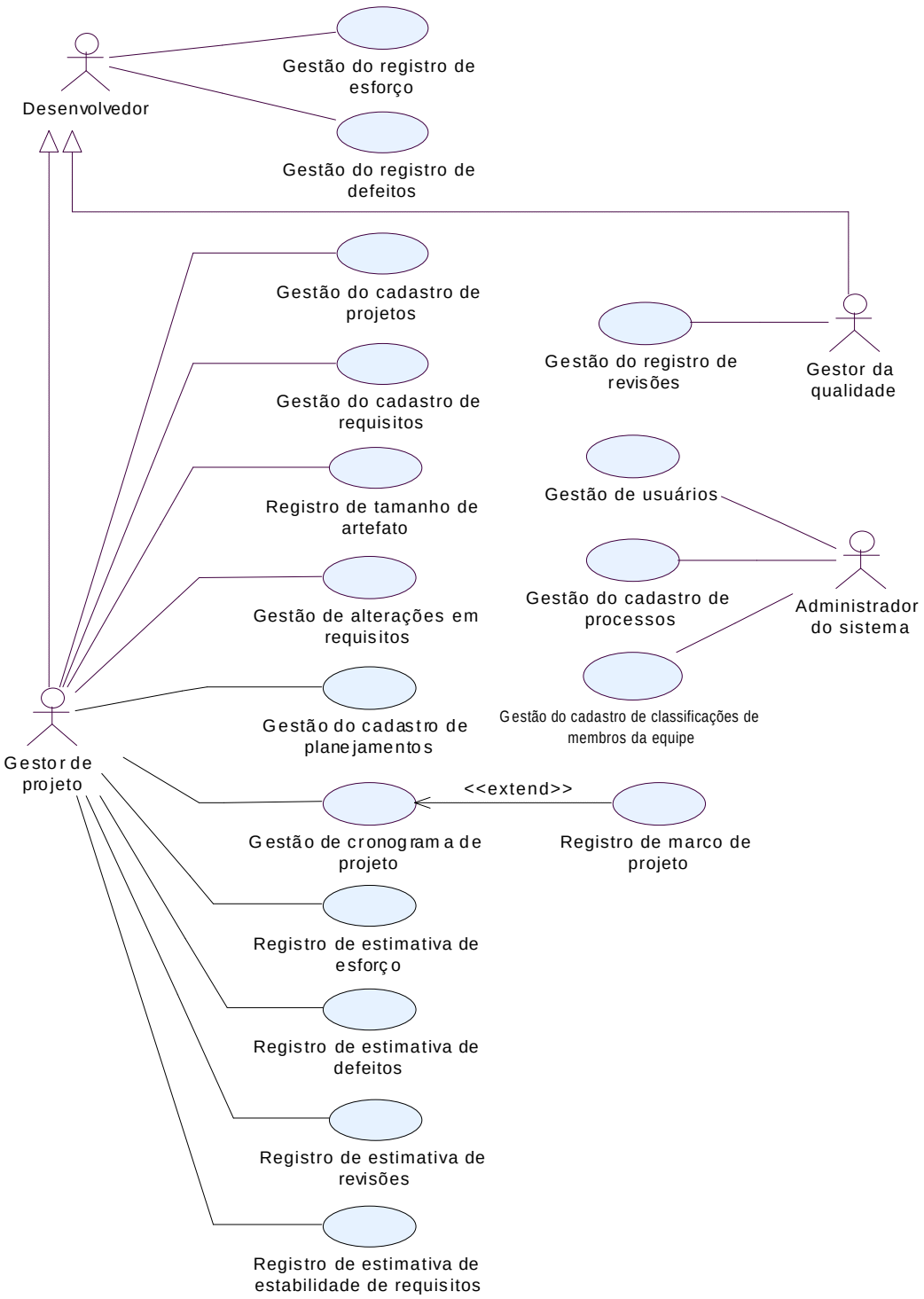


FIGURA 37 - Diagrama de contexto da ferramenta Quantum

A TABELA 21 descreve as funções apresentadas no diagrama. Como vários desses casos de uso constituem automações dos formulários de registro de medidas apresentados no Capítulo 4, a tabela apresenta também a correspondência de cada função com esses formulários.

Caso de uso	Descrição	Formulário relacionado
Gestão do registro de esforço	Controle do registro do esforço dedicado a atividades de um projeto, ao longo do desenvolvimento.	Registro de esforço (FIGURA 15).
Gestão do registro de defeitos	Controle do registro de defeitos encontrados em artefatos do projeto, com o objetivo de fundamentar a aferição quantitativa da qualidade do produto desenvolvido.	Registro de defeito (FIGURA 22).
Gestão do registro de revisões	Controle do registro das atividades de revisão (revisões técnicas, inspeções, etc.) realizadas sobre artefatos de um projeto.	Registro de revisão (FIGURA 24)
Gestão do cadastro de projetos	Inclusão, alteração, consulta e exclusão de projetos de desenvolvimento de software, com informações a respeito das equipes alocadas a cada um.	Registro de informações sobre projeto (FIGURA 17)
Gestão do cadastro de requisitos	Consulta, inclusão, exclusão e alteração dos requisitos cadastrados para um projeto.	-
Registro de métricas de artefatos	Registro do resultado de uma medição realizada sobre um ou mais artefatos, cuja mensuração está prevista no processo de desenvolvimento.	Registro de medidas de artefatos (FIGURA 11)
Gestão de alterações em requisitos	Registro de alterações em requisitos ocorridas durante o curso de um projeto, com informações sobre tamanho e esforço necessário para realização das mesmas.	Registro de alteração em requisito (FIGURA 13)
Gestão do cadastro de planejamentos	Inclusão, alteração, consulta e exclusão de planejamentos realizados para determinado projeto de desenvolvimento.	Cadastro dos planejamentos realizados em um projeto (FIGURA 26)
Gestão de cronograma de projeto	Consulta e alteração de informações a respeito do cronograma de um projeto, seja ele real ou referente a um dos planejamentos realizados.	Registro de cronograma (FIGURA 21)
Registro de marco de projeto	Consulta e alteração de informações a respeito de um marco (real ou planejado) em um projeto, caracterizando o progresso atingindo até então.	Registro de marco de projeto (FIGURA 9)
Registro de estimativa de esforço	Consulta e alteração das informações referentes à estimativa de esforço confeccionada durante um dos planejamentos cadastrados para o projeto.	Registro de estimativa de esforço (FIGURA 31)
Registro de estimativa de defeitos	Consulta e alteração das informações referentes à estimativa de defeitos confeccionada durante um dos planejamentos cadastrados para o projeto.	Registro de estimativa de defeitos (FIGURA 32)
Registro de estimativa de revisões	Consulta e alteração das informações referentes à estimativa de revisões confeccionada durante um dos planejamentos cadastrados para o projeto.	Registro de estimativa de revisões (FIGURA 33)
Registro de estimativa de estabilidade de requisitos	Consulta e alteração das informações referentes à estimativa de estabilidade de requisitos confeccionada durante um dos planejamentos cadastrados para o projeto.	Registro de estimativa de estabilidade de requisitos (FIGURA 28)
Gestão de usuários	Manutenção dos dados e das permissões de acesso de usuários do Quantum.	-
Gestão do cadastro de processos	Consulta, inclusão e exclusão de processos de desenvolvimento no acervo de processos do Quantum.	-
Gestão do cadastro de classificações de membros da equipe	Consulta, inclusão, alteração e exclusão de classificações utilizadas para os membros da equipe alocados aos projetos de desenvolvimento.	-

TABELA 21 - Descrição dos casos de uso e relacionamento com os formulários apresentados no Capítulo 4.

O diagrama de contexto da FIGURA 37 ilustra também os quatro papéis de usuários que podem interagir com o Quantum:

- **Desenvolvedor** – Membro da equipe de desenvolvimento da organização, responsável por registrar o esforço dedicado aos projetos em andamento e os defeitos detectados ao longo do trabalho.
- **Gestor de projeto** – Membro da equipe que, além das atribuições do desenvolvedor quanto ao registro de medidas de esforço e defeitos, é responsável pela gestão de um ou mais projetos em andamento na organização.
- **Gestor da qualidade** – Membro da equipe que verifica e assegura a qualidade dos produtos e processos de software. No Quantum, é responsável por cadastrar as medidas referentes às revisões.
- **Administrador do sistema** – Usuário responsável por manter as informações de suporte necessárias ao funcionamento do sistema.

É importante ressaltar que o escopo do Quantum contempla apenas a coleta das medidas e o armazenamento destas em uma base de dados centralizada. Não inclui facilidades de análise – para esse fim a base de dados gerada deve ser utilizada como insumo a outras ferramentas, como o *PSM Insight*, mencionado no Capítulo 2 (seção 2.3.1).

Outro limite que merece ser destacado é o fato do Quantum não prever a integração com outras ferramentas para permitir a extração automática de medidas. Uma possibilidade, por exemplo, seria integrá-la a sistemas de acompanhamento de defeitos (*bug-tracking*) para que os dados sobre defeitos sejam automaticamente extraídos e armazenados na base de dados. Esse tipo de integração ficou registrado como requisito adiado para uma versão futura da ferramenta.

6.2 Descrição do processo de desenvolvimento

O Praxis 2.0, descrito no Capítulo 5, foi utilizado como processo de desenvolvimento para a especificação e modelagem do Quantum. Das quatro fases desse processo, foram concluídas as duas primeiras (Concepção e Elaboração), obtendo-se como resultados principais uma Especificação de Requisitos e um Modelo de Análise, que descrevem e modelam de forma precisa e completa todos os requisitos da ferramenta.

Na Especificação de Requisitos, os requisitos funcionais e não funcionais do Quantum foram devidamente documentados. Cada caso de uso apresentado na seção anterior teve seu fluxo de execução descrito na forma de roteiros textuais que descrevem as ações dos usuários e as respostas que o Quantum deve fornecer a essas ações. Por exemplo, para o caso de uso Gestão de Alterações em Requisitos, um dos possíveis roteiros é a inclusão de um novo registro de alteração em requisitos. Tal roteiro foi descrito como sendo um dos fluxos alternativos que podem ser executados nesse caso de uso, e a descrição feita é apresentada na FIGURA 38.

Fluxo alternativo Inclusão de alteração em requisito

Precondições	O Gestor de projeto optou por incluir uma nova alteração de requisito.
Passos	1. O Gestor de projeto informa os dados da alteração de requisito a ser incluída: projeto, identificador, data da solicitação, data de conclusão, descrição, tamanho da alteração e esforço dedicado. 2. O **Quantum** verifica se não há outra alteração de requisito cadastrada para o mesmo projeto, com o identificador informado. 3. Caso a verificação tenha obtido sucesso, o **Quantum** salva os dados da nova alteração de requisito.

FIGURA 38 - Fluxo que descreve a inclusão de um registro de alteração em requisitos, no caso de uso "Gestão de alterações em requisitos"

Além de descrever a seqüência de ações, a Especificação de Requisitos contém também informações sobre as interfaces que o produto deve ter para interagir com os agentes externos (usuários, dispositivos de hardware ou outros sistemas). Por exemplo, para o fluxo exemplificado, é necessário que o sistema possua uma tela onde o usuário possa informar os dados da alteração de requisitos que está sendo registrada. A FIGURA 39 apresenta o esboço da interface que foi especificada para esse fim. Interfaces desse tipo foram descritas para todos os casos de uso.

Registro de alteração em requisito	
Projeto	*Quantum 1.0*
Identificador da alteração	*AR_012*
Data da solicitação	*29/10/2002*
Data da conclusão	*17/11/2002*
Descrição	*Inclusão do campo "total" no relatório de defeitos.*
Tamanho (PF ajustados)	*11*
Esforço dedicado (h)	*16*
Requisitos alterados	
Identificador	Descrição
CUA3.1	*Impressão de relatório*
ITA3.2	*Relatório de defeitos*
<Incluir requisito> <Excluir requisito>	
<Novo> <Pesquisar> <Excluir> <Salvar> <Fechar>	

FIGURA 39 - Esboço da interface para registro de alterações em requisitos

No Modelo de Análise, os requisitos foram detalhados através de recursos mais formais de notação:

- **Diagramas de classes**, contendo classes, atributos e relacionamentos que representam precisamente os conceitos expressos nos requisitos;
- **Realizações de casos de uso**, que definem os fluxos dos mesmos em função das iterações entre as classes identificadas.

Os diagramas de classes foram inteiramente baseados naqueles apresentados no Capítulo 4, acrescidos de classes de fronteira (que modelam as interfaces da ferramenta) e de controle (que coordenam os fluxos dos casos de uso). Todos os roteiros de execução descritos como fluxos de casos de uso na Especificação de Requisitos foram então realizados em termos de interações entre essas classes identificadas, através de diagramas de seqüência em notação UML. Para o mesmo exemplo da inclusão de um registro de alteração em requisitos, obteve-se o diagrama apresentado na FIGURA 40.

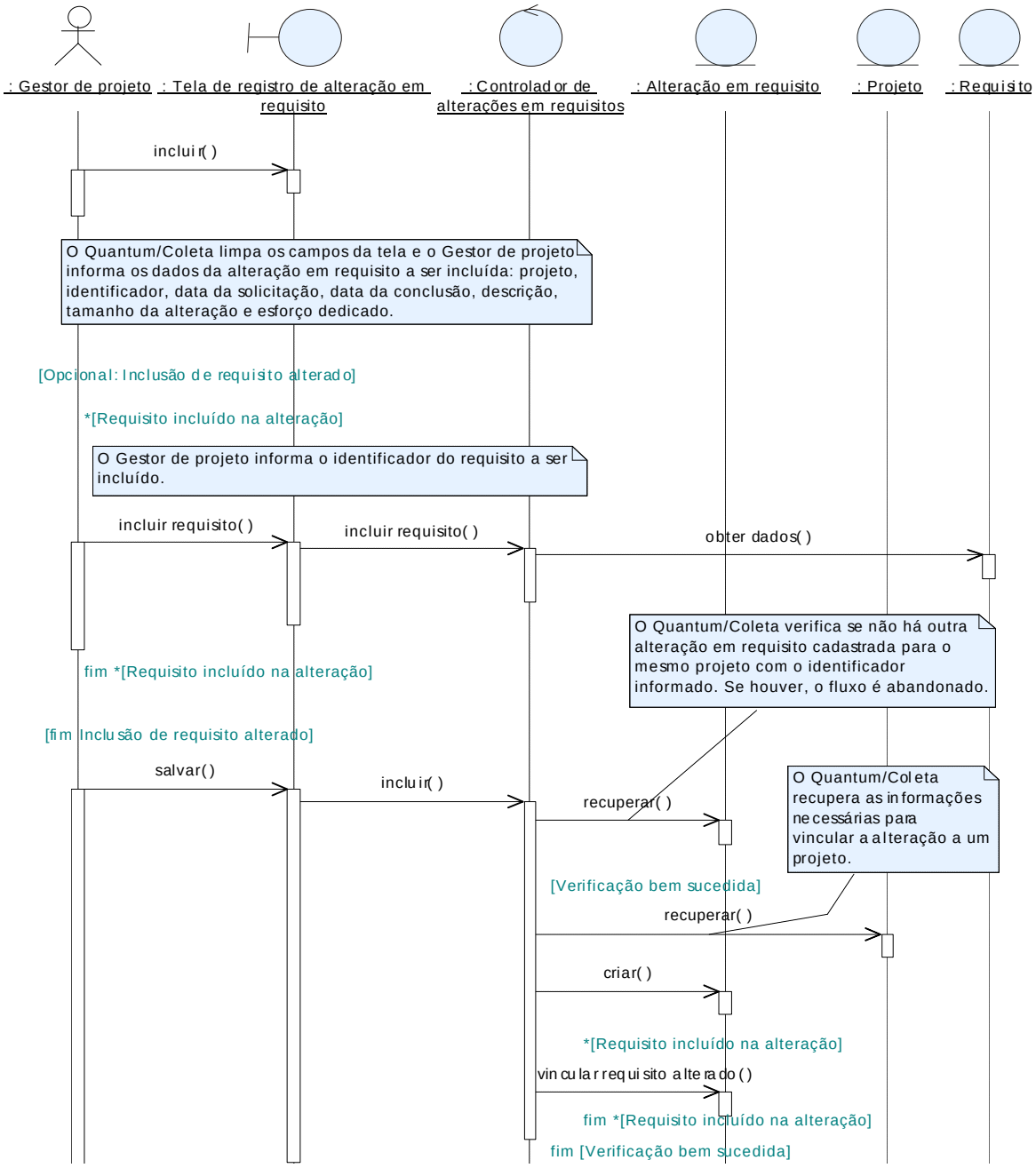


FIGURA 40 - Realização do fluxo de inclusão de um registro de alteração em requisito

Ao final da confecção da Especificação de Requisitos e do Modelo de Análise, todos os requisitos descritos foram cadastrados e classificados em um artefato específico para esse fim, o Cadastro dos Requisitos de Software. Esse artefato, por sua vez, serviu de base para a mensuração do tamanho em pontos de função – atividade cujo resultado é apresentado na seção seguinte, juntamente com outras medições que foram realizadas.

6.2.1 Medições realizadas

Durante a elaboração da ferramenta, as diversas medições sugeridas neste trabalho foram realizadas, como forma de experimentar os procedimentos de coleta descritos no Capítulo 4 e identificar eventuais ajustes necessários. Para isso, os formulários para registro das medições foram implementados em uma planilha eletrônica que foi sendo atualizada na medida em que a ferramenta era especificada.

A TABELA 22 apresenta os dados referentes à fase de Elaboração. Apesar de isoladamente não possibilitarem muitas conclusões significativas (conclusões desse tipo só são viabilizadas quando há dados de diferentes projetos para comparação), esses números ilustram os tipos de medidas que podem ser obtidas a partir do processamento dos dados coletados.

Esforço total	108,3 horas trabalhadas.				
Distribuição do esforço	**Fluxo[4] --> Iteração \/**	**Requisitos**	**Análise**	**Desenho**	**Gestão de projetos**
	LR	20,8%	3,7%	0,0%	0,0%
	AR	19,3%	30,4%	22,6%	3,2%
Tamanho do produto	384,0 Pontos de Função ajustados.				
Produtividade na Elaboração	567,3 Pontos de Função / Pessoa-Mês.				
Situação do projeto ao final da Elaboração	**Progresso**	20%			
	Valor	0%			
Quantidade de defeitos injetados	33				
Densidade de defeitos	0,09 defeito / Ponto de Função.				
Distribuição dos defeitos quanto à natureza - ERSw	**Natureza**		**%**		
	Omissão		44,4%		
	Violação		48,1%		
	Imprecisão		0,0%		
	Estilo		7,4%		
	Outro		0,0%		
Distribuição dos defeitos quanto à natureza - MASw	Usabilidade		0,0%		
	Documentação		42,9%		
	Sintaxe		0,0%		
	Construção		0,0%		
	Atribuição		0,0%		
	Interface		57,1%		
	Verificação		0,0%		
	Dados		0,0%		
	Função		0,0%		
	Sistema		0,0%		
	Ambiente		0,0%		
Total de tempo dedicado a correção de defeitos	3,85 horas.				
Percentual de retrabalho	3,6%				
Percentual de defeitos detectados em revisões	15,2%				
Esforço médio para correção de defeitos	0,12 hora / defeito.				
Medidas de artefatos	**Artefato**	**Unidade de medida**	**Valor medido**		
	ERSw	Número de páginas	103		
		Número de palavras	17.770		
		Número de figuras	35		
	MASw	Número de classes	69		
		Número de atributos	55		
		Número de operações	241		

TABELA 22 - Medidas coletadas durante a fase de Elaboração do Quantum

[4] Não houve horas registradas para os demais fluxos, além dos quatro citados.

Capítulo 7

Conclusão

A exploração de características comuns a grande parte dos processos de desenvolvimento de software possibilitou a criação de um arcabouço de medição genérico, que pode ser configurado e adaptado a diferentes contextos. As práticas de medição propostas são capazes de caracterizar quantitativamente alguns aspectos importantes no desenvolvimento de produtos de software, tais como esforço, tamanho, defeitos, revisões, progresso, cronograma e estabilidade de requisitos. Cada um dos atributos medidos não foi selecionado por acaso: a utilização de um processo específico para esse fim, o *Goal-Driven Software Measurement*, garantiu que cada uma das medidas escolhidas esteja alinhada com metas de negócio importantes para as organizações em geral, por tratarem de questões estratégicas como qualidade dos produtos, produtividade e previsibilidade.

Por sua flexibilidade, espera-se que o modelo aqui proposto possa ser utilizado como ponto de partida para a implantação da mensuração em diversas organizações, contribuindo para que aproveitem melhor o potencial que essa prática pode oferecer. O único pré-requisito é a existência de um processo de desenvolvimento organizado em uma estrutura matricial, contendo uma dimensão de fases e iterações, que divide o ciclo ao longo do tempo, e uma dimensão de fluxos, que divide as atividades em temas técnicos. Essa genericidade é importante não só para permitir a aplicação a diferentes processos: mesmo diante de um único processo, é ela quem garante a robustez para viabilizar a realização de personalizações. Já em organizações cujos processos de desenvolvimento não se enquadrem nessa estrutura matricial, ou que possuam metas de negócio diferentes daquelas levadas em consideração na seleção das medidas, a documentação do trabalho de criação do modelo ainda pode ter um importante valor educacional, servindo de exemplo para a construção de outros mais específicos.

As práticas sugeridas podem ser adotadas por organizações de diferentes níveis de maturidade. Dentro da classificação estabelecida pelo CMMI, organizações que se encontrem no nível 2 de capacitação (ou que almejem atingi-lo) podem aplicar os procedimentos de medição para compor uma base de dados que auxilie no planejamento e controle dos projetos. Organizações mais maduras, interessadas em atingir o nível 4, podem também tirar proveito do modelo aliando-o a técnicas mais formais de análise (como o controle estatístico de processos) para implementar as práticas previstas nas áreas-chaves *Desempenho dos Processos Organizacionais* e *Controle Quantitativo de Projetos*.

A descrição da aplicação do modelo a um processo concreto, o Praxis, também trouxe resultados importantes. Além de exemplificar como o arcabouço proposto pode ser configurado para atender a particularidades de cada processo, trouxe benefícios diretos ao próprio Praxis. Como em sua forma padrão o Praxis não formaliza os procedimentos de mensuração, o modelo proposto pode ser utilizado como insumo para uma personalização desse processo que contemple a medição de maneira mais completa. Além disso, constitui um primeiro passo para a obtenção de um banco centralizado para os dados de gestão, atualmente inexistente nesse processo.

Há ainda outra contribuição da mensuração para a qualidade dos processos que merece ser destacada. Um subproduto interessante de um programa de desenvolvimento de medidas é que ele obriga a fazer definições mais rigorosas dos processos nos quais será inserido, o que, por sua vez, influencia a qualidade desses processos como um todo. A necessidade de formalização de alguns conceitos, como os critérios para identificação de defeitos e a caracterização dos estados que requisitos podem assumir ao longo do desenvolvimento, contribui para uma maior precisão das definições contidas no processo.

7.1 Trabalhos futuros

Como foi ressaltado na introdução (Capítulo 1), as medidas e a estrutura propostas neste trabalho compõem um *modelo de informação*. Para que ele possa ser colocado em prática, é necessário que os procedimentos de medição sejam incorporados aos processos de desenvolvimento nos quais será aplicado. Para isso, é necessário dividir as atividades relacionadas à mensuração em um conjunto ordenado de passos e atribuir responsabilidades aos agentes envolvidos, formando um *processo* de medição. Uma extensão interessante deste trabalho seria a definição desse processo, que poderia tomar como base os conceitos do PSM e do padrão ISO/IEC 15939, mencionados no Capítulo 2.

Outra linha de pesquisa interessante é o estudo de técnicas mais formais para a análise dos dados coletados. Uma das técnicas mais tradicionais é o *Controle Estatístico de Processos*, que utiliza ferramentas estatísticas (como os gráficos de controle, ou *control charts*) para monitorar os vários atributos mensurados e verificar se suas propriedades se mantêm dentro de limites aceitáveis [FLORAC97]. Estudos mais recentes apontam também novos paradigmas que podem ser utilizados para tirar um melhor proveito dos dados coletados nas mensurações. Duas abordagens merecem ser investigadas:

- *Mineração de dados* – Tradicionalmente, a análise de medidas coletadas utiliza estratégias *top-down*. É o caso do próprio GQM, que usa um conjunto de metas previamente definidas para guiar a seleção e a análise das medidas que irão compor um programa de mensuração. [MENDONÇA98] sugere que, além dessa abordagem, técnicas de mineração de dados sejam usadas para descobrir desvios e

correlações não triviais entre os diversos valores medidos. A idéia é usar ferramentas automatizadas para percorrer a base de dados de medidas à procura de *fatos interessantes*, representados por desvios em relação a alguma distribuição estimada ou correlações inesperadas entre valores de um conjunto de atributos. Detalhes sobre essa abordagem podem ser obtidos também em [MENDONÇA99].

- *Redes Bayesianas* – O desenvolvimento recente de algoritmos para lidar com probabilidade Bayesiana possibilitam que técnicas baseadas em redes Bayesianas sejam utilizadas na análise de medições de software, assim como tem sido feito em outras áreas como diagnósticos médicos e investigações de falhas mecânicas. Essas redes são formadas por grafos associados a tabelas de probabilidade que modelam relações de causa e efeito. Através delas, é possível prever o impacto que flutuações localizadas podem ter sobre os demais atributos que compõem o modelo, sendo portanto úteis para a realização de estimativas diversas [FENTON99][FENTON00].

A base de dados formada a partir dos procedimentos propostos neste trabalho pode ser utilizada como entrada para todas essas técnicas.

Quanto à ferramenta Quantum, já especificada e modelada, vários trabalhos podem ser ainda realizados. Em primeiro lugar, obviamente, a ferramenta deve ser construída em sua totalidade. Além disso, algumas extensões podem ser feitas para que sua utilização possa trazer ainda mais benefícios para os programas de medição. Uma delas é a integração com outras ferramentas para viabilizar a coleta automática de algumas medidas. Por exemplo, a integração com um produto para acompanhamento de defeitos pode permitir que dados sobre os defeitos registrados sejam automaticamente capturados para a base de dados do Quantum, assim como interfaces com ferramentas de alocação e acompanhamento de tarefas podem possibilitar a extração automática de medidas de esforço.

Finalmente é importante ressaltar que, assim como ocorre com processos de desenvolvimento de software, procedimentos de medição devem ser experimentados, avaliados e continuamente ajustados e melhorados. Por isso, é importante aplicar o modelo proposto neste trabalho em projetos reais de desenvolvimento, com o objetivo de validar as práticas propostas e identificar possibilidades de melhoria.

Referências bibliográficas

[ALBRECHT79] ALBRECHT, A. J. Measuring application development productivity. In: IBM APPLICATIONS DEVELOPMENT SYMPOSIUM, 1979, Monterey. *Proceedings of...* Monterey: [s.n.], 1979. p. 83.

[ALBRECHT83] ALBRECHT, Allan J., GAFFNEY JR, John E. Software Function, source lines of code, and development effort prediction: a software science validation. *IEEE Transactions on Software Engineering*, [s.l.], vol. SE-9 n.6, p. 639-648, Nov 1983.

[AMBLER99] AMBLER, Scott W. *More Process Patterns.* Cambridge: Cambridge University Press, 1999. 369 p.

[BASILI92] BASILI, Victor R. *Software Modeling and Measurement: The Goal/Question/Metric Paradigm.* Technical Report UMIACS-TR-92-96. University of Maryland, 1992.

[BASILI94A] BASILI, Victor, CALDIERA, Gianluigi, ROMBACH, H. Dieter. Goal Question Metric Paradigm. In: *Encyclopedia of Software Engineering*. [s.l.]: John Wiley & Sons, Inc., 1994, p. 528-532.

[BASILI94B] BASILI, Victor, CALDIERA, Gianluigi, ROMBACH, H. Dieter. Measurement. In: *Encyclopedia of Software Engineering*. [s.l.]: John Wiley & Sons, Inc., 1994, p. 646-661.

[BOEHM76] BOEHM, B. W., BROWN, J. R., LIPOW, M. Quantitative Evaluation of Software Quality. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 2, 1976, [s.l.]. *Proceedings of the...* [s.l.]: [s.n.], 1976. p. 592 apud BASILI, Victor R. *Software Modeling and Measurement: The Goal/Question/Metric Paradigm*. Technical Report UMIACS-TR-92-96. University of Maryland, 1992.

[BOOCH94] BOOCH, Grady. *Object-Oriented Analysis and Design with Applications* 2nd ed. Redwood City: Benjamin/Cummings, 1994.

[BOOCH96] BOOCH, Grady. *Object Solutions*; Managing the Object-Oriented Project. Reading, MA: Addison-Wesley, 1996.

[CARLETON92] CARLETON, Anita D. et al. *Software Measures for DoD Systems*; Recommendations for Initial Core Measures (CMU/SEI-92-TR-019). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1992. Available from World Wide Web: <http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.019.html>.

[CMMI02] CMMI Product Team. *CMMI for Software Engineering*, Version 1.1, Staged Representation (CMMI-SW, V1.1, Staged). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2002. Available from World Wide Web: <http://www.sei.cmu.edu/publications/documents/02.reports/02tr029.html>.

[DoD00] USA. Department of Defense. *Practical Software and Systems Measurement*; A Foundation for Objective Project Management (P1045/D5.0). Washington, D.C.: Department of Defense and US Army, 2000. Available from World Wide Web: <http://www.psmsc.com>.

[DREGER89] DREGER, J. Brian. *Function Point Analysis*. Englewood Cliffs – NJ: Prentice-Hall, 1989.

[FENTON94] FENTON, Norman E. Software Measurement: A Necessary Scientific Basis. *IEEE Transactions on Software Engineering*, [s.l.], vol 20 n. 3, p. 199-206, Mar 1994.

[FENTON00] FENTON, Norman E., NEIL, Martin. Software Metrics: Roadmap. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, Limerick/Ireland. *Proceedings of the conference on The future of Software engineering*. New York: ACM Press, 2000. p.357-370.

[FENTON96] FENTON, Norman E., PFLEEGER, Shari L. *Software Metrics*; A Rigorous and Practical Approach. 2nd ed. London: International Thonson, 1996.

[FENTON99] FENTON, Norman E., NEIL, Martin. A Critique of Software Defect Prediction Models. *IEEE Transactions on Software Engineering*, [s.l.], vol. 25 n.3, p. 1-15, May/Jun 1999.

[FLORAC92] FLORAC, William A. *Software Quality Measurement*; A Framework for Counting Problems and Defects (CMU/SEI-92-TR-22). Pittsburgh, PA: Software Engineering Institute, Carnegie Melon University, 1992. Available from World Wide Web: <http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.022.html>

[FLORAC97] FLORAC, William A., PARL, Robert E., CARLETON, Anita D. *Practical Software Measurement*; Measuring for Process Management and Improvement (CMU/SEI-97-HB-003). Pittsburgh, PA: Software Engineering Institute, Carnegie Melon University, 1997. Available from World Wide Web: <http://www.sei.cmu.edu/publications/documents/97.reports/97hb003/97hb003abstract.html>.

[GARMUS01] GARMUS, David, HERRON, David. *Function Point Analysis*; Measurement Practices forSuccessful Software Projects. [s.l.]: Addison-Wesley, 2001.

[GOETHERT92] GOETHERT, Wolfhart B., BAILEY, Elizabeth K., Busby, Mary B. *Software Effort Measurement*; A Framework for Counting Staff-Hours and Reporting Schedule Information (CMU/SEI-92-TR-21). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1992. Available from World Wide Web: <http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.021.html>.

[GRADY87] GRADY, R. B, CASSWELL, D. R. *Software Metrics*; Establishing a Company-Wide Program. Englewood Cliffs, N J: Prentice-Hall, 1987.

[HUMPHREY89] HUMPHREY, Watts S. *Managing the Software Process*. Reading, MA: Addison-Wesley, 1989.

[HUMPHREY95] HUMPHREY, Watts S. *A Discipline for Software Engineering*. Reading, MA: Addison Wesley, 1995.

[IEEE90] INSTITUTE OF ELECTRICAL AND ELECTRONIC ENGINEERS, New York. *IEEE Standard P-1061/D21*; Standard for a Software Quality Metrics Methodology. New York, 1990.

[IEEE92] INSTITUTE OF ELECTRICAL AND ELECTRONIC ENGINEERS, New York. *IEEE Std 1045-1992*; Standard for Software Productivity Metrics. New York, 1992.

[IFPUG99] THE INTERNATIONAL FUNCTION POINT USERS GROUP, Princeton Junction. *Function Point Counting Practices Manual release 4.1*. [s.l.], 1999.

[ISO01] THE INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. *ISO/IEC 15939;* Software Engineering – Software Measurement Process. [s.l.], 2001.

[JACOBSON94] JACOBSON, Ivar. *Object-Oriented Software Engineering*. Reading, MA: Addison-Wesley, 1994.

[JACOBSON98] JACOBSON, Ivar, BOOCH, Grady, RUMBAUGH, James. *The Unified Software Development Process*. Reading, MA: Addison-Wesley, 1998.

[KITCHENHAM95] KITCHENHAM, Barbara, PFLEEGER, Shari L. Towards a Framework for Software Measurement Validation. *IEEE Transactions on Software Engineering*, [s.l.], vol. 21 n. 12, p. 929-944, Dec. 1995.

[KOGURE83] KOGURE, M., AKAO, Y. Quality Funcion Deployment and CWQC in Japan. *Quality Progress*, [s.l.], p. 25-29, Oct 1983 apud BASILI, Victor R. *Software Modeling and Measurement: The Goal/Question/Metric Paradigm*. Technical Report UMIACS-TR-92-96. University of Maryland, 1992.

[KRUCHTEN00] KRUCHTEN, Philippe. *Rational Unified Process: An Introductio*n. Reading, MA: Addison-Wesley, 2000.

[LANDSBAUM92] LANDSBAUM, J.B, GLASS, R. L. *Measuring and Motivating Maintenence Programmers*. Englewood Cliffs, NJ: Prentice-Hall Inc, 1992 apud AMBLER, Scott W. *More Process Patterns*. Cambridge: Cambridge University Press, 1999. 369 p.

[LONGSTREET02] LONGSTREET, David. *Function Points Analysis Training Course* [online]. [s.l.]: [s.n.], 2002. Available from World Wide Web: <http://www.softwaremetrics.com>.

[MCANDREWS93] MCANDREWS, Donald R. *Establishing a Software Measurement Process* (CMU/SEI-93-TR-16). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1993. Available from World Wide Web: <http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.016.html>.

[MCGABE76] MCGABE, T. J. A. Complexity Measure. *IEEE Transactions on Software Engineering*, [s.l], vol. SE-2, 4, p. 308-320, Dec. 1976.

[MCGARRY01] MCGARRY, John et al. *Practical Software Measurement*; Objective Information for Decision Makers. Reading, MA: Addison Wesley, 2002.

[MENDONÇA99] Manoel Mendonca, Nancy L. Sunderhaft. Mining Software Engineering Data: A Survey [online]. Rome, NY: DATA & ANALYSIS CENTER FOR SOFTWARE, 1999. Available from World Wide Web: <http://www.dacs.dtic.mil/techs/datamining>

[MENDONÇA98] MENDONÇA, M. G., BASILI, V. R. et al. An approach to improving existing measurement frameworks. *IBM Systems Journal*, [s.l.], vol. 37, number 4, 1998.

Disponível na WWW: <http://www.research.ibm.com/journal/sj/374/mendonca.html> 01/03/2003.

[MILLS88] MILLS, Everald E. *Software Metrics* (CMU/SEI-CM-12-1.1).). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1998. Available from World Wide Web: <http://www.sei.cmu.edu/publications/documents/cms/cm.012.html>.

[PARK92] PARK, Robert E. *Software Size Measurement*; A Framework for Counting Source Statements (CMU/SEI-92-TR-20). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1992. Available from World Wide Web: <http://www.sei.cmu.edu/publications/documents/92.reports/92.tr.020.html>.

[PARK96] PARK, Robert E., GOETHERT, Wolfhart B., FLORAC, William A. *Goal-Driven Software Measurement: A Guidebook* (CMU;SEI-96-HB-002). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1996. Available from World Wide Web: <http://www.sei.cmu.edu/publications/documents/96.reports/96.hb.002.html>.

[PAULA03] PAULA FILHO, Wilson de Pádua. *Engenharia de software*; fundamentos, métodos e padrões. 2.ed. Rio de Janeiro: Editora LTC, 2003. 602 p.

[PAULK93] PAULK, Mark C. et al. *Key Practices of the Capability Maturity Model SM , Version 1.1*. (CMU/SEI-93-TR-025). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1993. Available from World Wide Web: <http://www.sei.cmu.edu/publications/documents/93.reports/93.tr.025.html>.

[PAULK95] PAULK, Mark C. et al. *The Capability Maturity Model*; Guidelines for Improving the Software Process. Reading, MA: Addison-Wesley, 1995. 441p.

[PFLEEGER97] PFLEEGER, Shari Lawrence et al. Status Report on Software Measurement. *IEEE Software*, [s.l.], p. 33-43, March/April 1997.

[PSMSC03] PRACTICAL SOFTWARE MEASUREMENT SUPPORT CENTER. *PSM Insight Tool* [online]. [s.l.][s.d.]. Available from World Wide Web: <http://www.psmsc.com/insight.htm>. (Fev. 2003).

[ROMBACH89] ROMBACH, H. Dieter, ULERY, Bradford T. Improving software maintenence through measurement. *Proceedings of the IEEE*, [s.l.], vol. 77, No.4, p. 581-595, April 1989.

[RUMBAUGH91] RUMBAUGH, James et al. *Object-Oriented Modeling and Design*. Englewood Cliffs, NJ: Prentice Hall, 1991.

[RUMBAUGH99] RUMBAUGH, James, JACOBSON, Ivar, BOOCH, Grady. *Unified Modeling Language Reference Manual*. Reading, MA: Addison-Wesley, 1999.

[SEL95] NASA SOFTWARE ENGINEERING LABORATORY, Greenbelt. *Software Measurement Guidebook, revision 1* (SEL-94-102). Greenbelt: NASA Goddard Space Flight Center, 1995. Available from World Wide Web: <http://sel.gsfc.nasa.gov/website/documents/online-doc.htm>.

[SOLINGEN99] SOLINGEN, Rini, BERGHOUT, Egon. *The Goal/Question/Metric Method*; a practical guide for quality improvement of software development. London: McGraw-Hill, 1999.

Apêndices

A.1 Entidades e atributos identificados durante a seleção das medidas

Durante a utilização do *Goal-Driven Software Measurement* (descrita no Capítulo 3), que guiou a seleção das medidas que compõem o modelo de mensuração proposto, uma das tarefas consistiu em identificar entidades e atributos que serão submetidos à mensuração. Essa tarefa corresponde ao quarto passo do processo, executado imediatamente após a definição de um conjunto de submetas, conforme encontra-se descrito na seção 3.1. Nessa seção, para identificar detalhar as metas de negócio em submetas, foram enumeradas perguntas cujo entendimento contribuiria para que as submetas pudessem ser atingidas de maneira controlada.

No quarto passo, para cada pergunta, devem ser primeiramente identificadas as entidades nela implícitas. Em seguida, são listados os atributos associados a cada entidade. Devem ser listados apenas atributos que, se quantificados, contribuam para a obtenção de respostas às perguntas formuladas ou estabeleçam o contexto para a interpretação de tais respostas.

É importante destacar que, na listagem de atributos, deve ser dada uma atenção especial para que sejam identificados *atributos* (características de uma entidade) e não *medidas* (escalas e regras usadas para atribuir valores aos atributos).A definição precoce de medidas específicas tende a restringir excessivamente a visão dos atributos e prejudicar o reconhecimento de novas possibilidades de medição [PARK96].

As tabelas a seguir apresentam as entidades e atributos identificados para cada uma das perguntas associadas às três metas de negócio. A TABELA 23 diz respeito à meta de qualidade dos produtos, a TABELA 24 refere-se à meta de produtividade, e a TABELA 25 trata da meta de cumprimento dos compromissos assumidos.

Meta 1: Produzir software de qualidade		
Pergunta	**Entidade(s)**	**Atributo(s)**
Quais os principais tipos de defeitos injetados e detectados ao longo do ciclo de desenvolvimento?	Conjunto de defeitos detectados ao longo do ciclo de desenvolvimento	Distribuição dos defeitos, por tipo.
Em que iteração e em que fluxo do processo os defeitos estão sendo injetados nos artefatos?	Conjunto de defeitos detectados ao longo do ciclo de desenvolvimento	Quantidade de defeitos injetados em cada fluxo e cada iteração.
Em que iteração e em que fluxo do processo os defeitos dos artefatos estão sendo detectados e corrigidos?	Conjunto de defeitos detectados ao longo do ciclo de desenvolvimento	Quantidade de defeitos detectados e corrigidos em cada fluxo e cada iteração.
Os requisitos estão sendo especificados de forma correta?	Defeitos detectados em requisitos	Taxa de defeitos detectados nos requisitos especificados.
Os requisitos especificados refletem as necessidades dos usuários?	Solicitações de alterações em requisitos	Taxa de ocorrência de alterações em requisitos.
As atividades de revisão se mostram eficazes na detecção dos defeitos?	Atividades de revisão	Rendimento das revisões na detecção de defeitos. Esforço empregado nas atividades de revisão. Número de defeitos detectados em atividades de revisão.
Está sendo disponibilizado esforço suficiente para as atividades de revisão?	Atividades de revisão	Esforço empregado nas atividades de revisão. Fração dos defeitos que são detectados em atividades de revisão.
Quais os tipos de defeitos detectados em atividades de revisão?	Conjunto dos defeitos detectados em atividades de revisão	Distribuição dos defeitos, por tipo.
Está sendo disponibilizado esforço suficiente para as atividades de teste?	Atividades de teste	Esforço empregado nas atividades de teste. Fração dos defeitos que são detectados em atividades de teste.
Os testes se mostram eficazes na detecção de defeitos?	Atividades de teste	Rendimento dos testes na detecção de defeitos. Esforço empregado nas atividades de teste. Número de defeitos detectados em testes.
Quais os tipos de defeitos detectados em atividades de testes?	Conjunto dos defeitos detectados em atividades de teste	Distribuição dos defeitos, por tipo.
O produto final possui taxa de defeitos em nível aceitável?	Conjunto de defeitos detectados no produto final	Tamanho do produto. Quantidade de defeitos detectados após a entrega do produto.
Quais os principais tipos de defeitos encontrados no produto final?	Conjunto de defeitos detectados no produto final	Distribuição dos defeitos, por tipo.

TABELA 23 - Entidades e atributos identificados para a meta de qualidade

Meta 2: Obter boa produtividade		
Pergunta	**Entidade(s)**	**Atributo(s)**
Qual o impacto dos defeitos injetados em cada fluxo e cada iteração, no que diz respeito à produtividade?	Defeitos corrigidos ao longo do ciclo de desenvolvimento	Quantidade de defeitos injetados em cada fluxo e iteração. Esforço médio gasto na correção dos defeitos, de acordo com o fluxo e iteração em que foram injetados.
Qual o esforço necessário para corrigir cada tipo de defeito?	Defeitos corrigidos ao longo do ciclo de desenvolvimento	Esforço médio gasto na correção dos defeitos, para cada tipo de defeito.
Qual a produtividade da equipe em cada um dos fluxos do processo?	Fluxos do processo	Esforço dedicado às atividades de cada fluxo; Tamanho do produto desenvolvido.
Qual a produtividade da equipe em cada iteração do processo?	Iterações do processo	Esforço dedicado às atividades de cada iteração; Tamanho do produto desenvolvido.
Que passos do processo estão apresentando melhor produtividade?	Fluxos e iterações do processo	Produtividade da equipe em cada fluxo, para cada iteração.
Que passos do processo estão apresentando pior produtividade?	Fluxos e iterações do processo	Produtividade da equipe em cada fluxo, para cada iteração.
Qual a taxa de ocorrência de mudanças em requisitos?	Solicitações de alteração em requisitos	Fração de requisitos alterados ao longo do projeto.
Qual o impacto das mudanças de requisitos na produtividade?	Solicitações de alteração em requisitos	Fração de requisitos alterados ao longo do projeto. Tempo gasto na adequação de artefatos e produtos às mudanças de requisitos.
Qual o tamanho do produto de software construído?	Produto de software	Tamanho do produto desenvolvido.
Qual foi o esforço total gasto para construí-lo?	Esforço da equipe de desenvolvimento	Esforço total dedicado ao projeto.
Qual foi o tamanho de cada artefato produzido?	Conjunto de artefatos construídos	Tamanho de cada artefato.
Qual o esforço gasto para construir cada artefato?	Conjunto de artefatos construídos	Esforço empregado na construção de cada artefato.
Qual o esforço investido em atividades de revisão (revisões técnicas, inspeções de código, etc)?	Atividades de revisão	Fração do esforço total que foi dedicado a atividades de revisão (revisões técnicas, inspeções, etc).
Qual o impacto dessas atividades na produtividade da equipe?	Atividades de revisão	Rendimento das revisões na detecção de defeitos. Esforço médio gasto na correção dos defeitos, de acordo com as iterações em que foram injetados e corrigidos.

TABELA 24 - Entidades e atributos identificados para a meta de produtividade

Meta 3: Cumprir os compromissos de prazo e custo assumidos		
Pergunta	**Entidade(s)**	**Atributo(s)**
O esforço estimado para cada fluxo ao longo do ciclo de desenvolvimento é adequado?	Estimativa de esforço	Esforço estimado para cada fluxo. Esforço obtido para cada fluxo.
A estimativa da distribuição do esforço entre as iterações do processo de desenvolvimento está adequada?	Estimativa de esforço	Esforço estimado para cada iteração. Esforço obtido para cada iteração.
Qual o grau de confiabilidade das estimativas produzidas?	Conjunto de estimativas produzidas	Erro apresentado por cada estimativa produzida no planejamento dos projetos.
O esforço total consumido no desenvolvimento dos produtos está dentro dos limites estimados?	Estimativa de esforço	Esforço total estimado para os projetos. Esforço total obtido para os projetos.
O planejamento dos prazos e do esforço está adequado, tendo em vista as características da organização e dos produtos a serem desenvolvidos?	Planejamento dos projetos	Cronograma planejado para os projetos. Esforço total planejado para os projetos. Valores estimados para esforço e prazos. Erro apresentado pelas estimativas de esforço. Erro apresentado pelas estimativas de cronograma.
O número de pessoas alocadas ao projeto é adequado tendo em vista os compromissos assumidos?	Planejamento dos projetos	Estimativa do esforço necessário para cada iteração, em cada projeto. Esforço obtido em cada iteração, em cada projeto.
O perfil dessas pessoas é adequado?	Planejamento dos projetos	Estimativa do esforço necessário para cada fluxo, em cada projeto. Esforço obtido em cada fluxo, em cada projeto.
A quantidade de esforço disponibilizado para os projetos é suficiente para concluir o desenvolvimento dos produtos dentro dos prazos assumidos?	Planejamento dos projetos	Esforço total estimado para cada projeto. Esforço total obtido para cada projeto.
O andamento de cada projeto tende ao cumprimento dos compromissos de prazo e custo?	Planejamento e acompanhamento dos projetos	Cronograma planejado para cada projeto. Esforço total estimado para cada projeto. Fração já concluída dos projetos. Esforço já empregado até o momento. Cronograma obtido até o momento.
Os produtos estão sendo entregues nos prazos acordados com os clientes?	Planejamento e acompanhamento dos projetos	Data planejada para conclusão de cada projeto. Data real de conclusão de cada projeto.
Todos os artefatos produzidos foram concluídos nos prazos previstos?	Planejamento e acompanhamento dos projetos	Cronograma planejado para os projetos. Cronograma obtido para os projetos.

TABELA 25 - Entidades e atributos identificados para a meta de cumprimento dos compromissos assumidos

A.2 Relação de indicadores

Este apêndice apresenta uma descrição dos indicadores criados para transformar as medidas coletadas em informações úteis, que possam contribuir para o alcance das metas de negócio da organização. O conceito de indicadores encontra-se descrito na seção 2.2.2.

Um primeiro esboço desses indicadores foi feito durante a execução do *Goal-Driven Software Measurement*. No sexto passo desse processo (descrito na seção 3.3), é elaborado um conjunto de perguntas e são identificados vários indicadores (gráficos e tabelas) que possam contribuir para a resposta a essas perguntas.

Posteriormente, após a formalização das medidas que compõem o modelo (seção 4.1), os indicadores foram refinados e documentados. Nessa documentação, além da definição de um código e um nome para cada indicador, foram registradas as seguintes informações:

- Descrição – breve descrição da informação contida no indicador;
- Ilustração – gráfico ou tabela que ilustra a forma de apresentação do indicador;
- Aplicabilidade – descrição da contribuição trazida pelas informações contidas no indicador, servindo também como um guia para a interpretação dos dados;
- Variações possíveis – descrição das diferentes formas de construção do indicador;
- Referências – seções deste documento que descrevem as medidas tratadas pelo indicador.

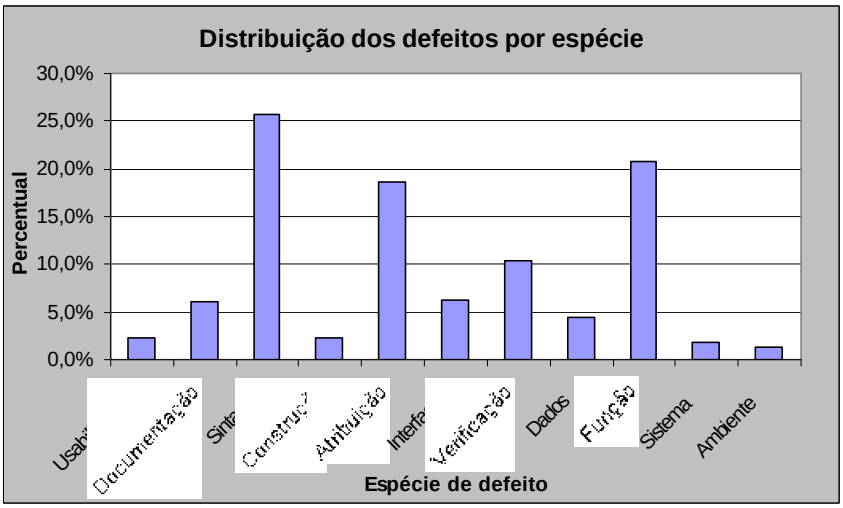
Ao todo foram criados 17 indicadores. As seções a seguir apresentam a documentação de cada um deles.

I-1 Distribuição dos defeitos por espécie

Descrição

Distribuição percentual dos defeitos detectados em um produto ou conjunto de produtos de acordo com as espécies em que foram classificados, para que seja possível detectar os tipos de defeitos mais comuns.

Ilustração



Aplicabilidade

O conhecimento dos tipos de defeitos mais comuns pode ser útil para guiar atividades de melhoria de processo que visam à redução dos defeitos e à detecção precoce dos mesmos. Testes e revisões, por exemplo, podem ser orientados para dar enfoque maior aos tipos de defeitos mais recorrentes.

Além disso, a distribuição de defeitos verificada em um determinado projeto pode ser comparada com a distribuição típica da organização, e a existência de divergências significativas entre ambas pode indicar sintomas de problemas de qualidade no projeto em questão.

Variações possíveis

Este é um indicador que pode ser utilizado de várias formas, cada uma considerando um conjunto distinto de defeitos:

1. Todos os defeitos registrados;
2. Todos os defeitos injetados em determinado fluxo;
3. Todos os defeitos injetados em determinada iteração;
4. Todos os defeitos detectados em determinado fluxo;
5. Todos os defeitos detectados em determinada iteração;
6. Todos os defeitos detectados em determinado tipo de revisão.

Para cada uma das situações acima, há ainda duas opções: considerar defeitos pertencentes a um projeto individual, ou considerar um conjunto de projetos.

Medidas necessárias

- Classificação de cada defeito quanto à espécie;
- Projeto ao qual pertence cada defeito;
- Fluxo em que cada defeito foi injetado (para a variação 1) e detectado (para a variação 4);
- Iteração em que cada defeito foi injetado (para a variação 3) e detectado (para a variação 5);
- Tipo de revisão em que cada defeito foi detectado (para a variação 6).

Referências

O conceito de *espécie de defeito* é apresentado na seção 4.2.2.

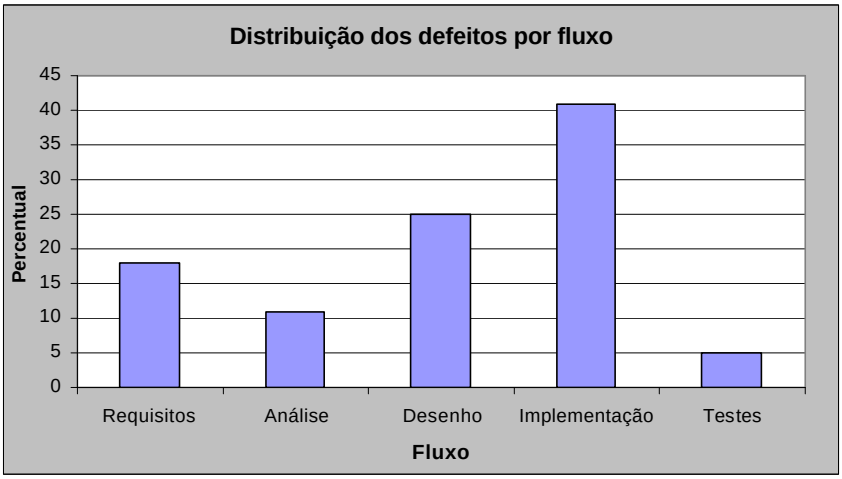
O registro de defeitos é detalhado na seção 4.1.5.

I-2 Distribuição dos defeitos por fluxo

Descrição

Distribuição percentual dos defeitos registrados em um produto ou conjunto de produtos de acordo com o fluxo em que foram detectados ou injetados.

Ilustração



Aplicabilidade

No contexto da organização, a distribuição dos defeitos em relação aos fluxos em que são injetados permite a identificação dos fluxos que estão gerando a maior parte deles, o que pode servir de base para um programa de melhoria de processos que tenha como meta a redução dos defeitos.

No contexto de um projeto, a distribuição de defeitos em relação aos fluxos em que foram injetados pode ser comparada com a distribuição típica da organização, e a existência de divergências significativas entre ambas pode indicar sintomas de problemas de qualidade em algumas atividades do projeto em questão.

A distribuição em relação aos fluxos em que os defeitos estão sendo detectados, por sua vez, permite verificar se os fluxos que visam à detecção de defeitos (como é o caso do fluxo de testes) realmente estão sendo eficazes.

Variações possíveis

O indicador pode ser usado para avaliar os fluxos em que os defeitos foram injetados ou os fluxos em que foram detectados.

Em ambos os casos, algumas variações são possíveis:

1. Considerar todos os defeitos registrados;
2. Considerar apenas os defeitos injetados em determinada iteração;
3. Considerar apenas os defeitos detectados em determinada iteração.
4. Considerar apenas defeitos de determinada espécie.

Para cada uma das situações acima, há ainda duas opções: considerar defeitos pertencentes a um projeto individual, ou considerar um conjunto de projetos.

Medidas necessárias

Para cada defeito detectado:

- Projeto ao qual pertence;
- Fluxo em que foi injetado e detectado;
- Iteração em que foi injetado (para a variação 2) e detectado (para a variação 3);
- Classificação quanto à espécie (para a variação 4).

Referências

O registro de defeitos é detalhado na seção 4.1.5.

I-3 Distribuição dos defeitos por iteração

Descrição

Distribuição percentual dos defeitos registrados em um produto ou conjunto de produtos de acordo com a iteração em que foram injetados e corrigidos.

Ilustração

Distribuição de defeitos por iteração										
Injetados na iteração	**Corrigidos na iteração**									
	AT	LR	AR	DI	L1	L2	TA	TB	OP	Total
AT	0,3%	2,7%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	0,0%	3,0%
LR		1,5%	3,4%	0,5%	0,3%	0,2%	0,0%	0,1%	0,0%	6,0%
AR			2,1%	1,8%	0,5%	0,6%	0,0%	0,0%	0,0%	5,0%
DI				7,0%	3,6%	2,3%	0,1%	0,0%	0,0%	13,0%
L1					16,2%	4,5%	3,3%	1,6%	0,4%	26,0%
L2						27,0%	4,8%	2,5%	0,7%	35,0%
TA							4,2%	0,8%	0,0%	5,0%
TB								3,9%	0,6%	4,5%
OP									2,5%	2,5%
Total	0,3%	4,2%	5,5%	9,3%	20,6%	34,6%	12,4%	8,9%	4,2%	100,0%

Aplicabilidade

Quanto mais cedo um defeito é detectado e corrigido, menor o custo de sua correção [HUMPHREY95]. É desejável, portanto, que um defeito injetado em uma iteração seja detectado o quanto antes possível. O indicador apresentado permite um acompanhamento do momento em que os defeitos injetados em cada iteração estão sendo corrigidos.

O indicador permite também verificar quais iterações estão inserindo o maior número de defeitos, fornecendo subsídios para as atividades de melhoria de processo.

No contexto de um projeto, a distribuição de defeitos em relação às iterações em que foram injetados e corrigidos pode ser comparada com a distribuição típica da organização, e a existência de divergências significativas entre ambas pode indicar sintomas de problemas de qualidade em algumas atividades do projeto em questão.

Variações possíveis

Há duas abordagens possíveis:

1. Considerar todos os defeitos registrados;
2. Considerar apenas defeitos de determinada espécie.

Para cada uma das situações acima, há ainda duas alternativas: considerar defeitos pertencentes a um projeto individual, ou considerar um conjunto de projetos.

Medidas necessárias

Para cada defeito detectado:

- Projeto ao qual pertence;
- Iteração em que foi injetado;
- Iteração em que foi corrigido;
- Classificação quanto à espécie (para a variação 2).

Referências

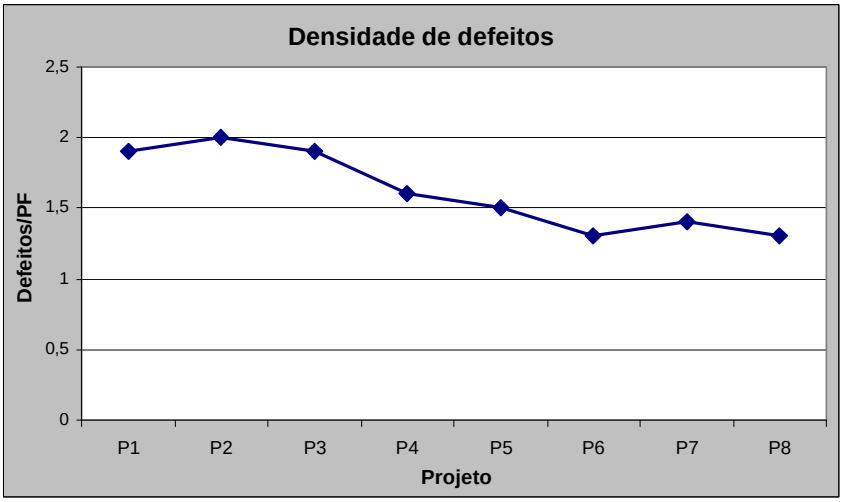
O registro de defeitos é detalhado na seção 4.1.5.

I-4 Densidade de defeitos

Descrição

Exibe a evolução da densidade de defeitos (quantidade de defeitos normalizada pelo tamanho dos produtos) ao longo dos projetos de desenvolvimento executados pela organização.

Ilustração



Aplicabilidade

A densidade de defeitos permite o acompanhamento da qualidade dos produtos desenvolvidos.

Densidades muito superiores à média da organização podem significar problemas de qualidade, enquanto densidades muito inferiores podem indicar que as atividades de verificação (testes e revisões) não estão sendo devidamente executadas.

Além disso, o acompanhamento da evolução da densidade de defeitos ao longo dos projetos pode ser usado para avaliar o impacto de melhorias feitas no processo para reduzir a quantidade de defeitos nos produtos.

Variações possíveis

Há quatro possibilidades de construção desse indicador:

1. Considerar todos os defeitos registrados em cada um dos projetos;
2. Considerar apenas os defeitos de determinada espécie;
3. Considerar apenas os defeitos injetados em determinada iteração;
4. Considerar apenas os defeitos injetados em determinado fluxo;

Medidas necessárias

Para cada defeito registrado:

- Projeto ao qual pertence;
- Classificação quanto à espécie (para a variação 2);
- Iteração em que foi injetado (para a variação 3;
- Fluxo em que foi injetado (para a variação 4).

Para cada projeto:

- Tamanho do produto desenvolvido, em Pontos de Função.

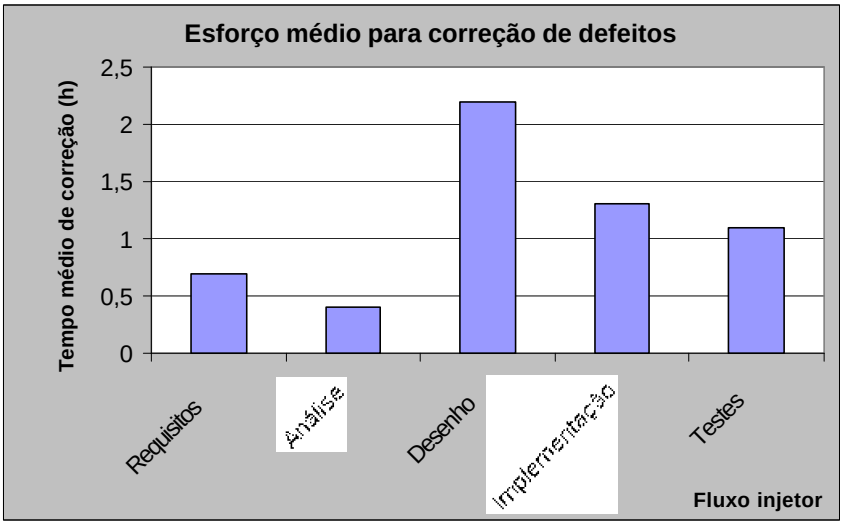
Referências

O registro de defeitos é detalhado na seção 4.1.5.

I-5 Esforço médio para correção de defeitos

Descrição Exibe o esforço médio para a correção de defeitos, separado por espécie de defeito ou por fluxo injetor.

Ilustração



Aplicabilidade No contexto da organização, o conhecimento do esforço típico para a correção de cada tipo de defeito permite que as ações de melhoria dos processos possam se concentrar na redução daqueles que representam maior custo.

No contexto de um projeto específico, conhecendo-se o esforço médio para a remoção de cada tipo de defeito é possível estimar o esforço que será necessário para corrigir defeitos ainda abertos no produto em desenvolvimento.

Variações possíveis O indicador pode exibir o tempo médio para correção dos defeitos de duas formas:

1. Em função dos fluxos em que foram injetados (como na ilustração);
2. Em função das iterações em que foram injetados e/ou corrigidos;
3. Em função da espécie dos defeitos.

Para cada uma das situações acima, há ainda duas alternativas: considerar defeitos pertencentes a um projeto individual, ou considerar um conjunto de projetos.

Medidas necessárias Para cada defeito registrado e corrigido:

- Projeto ao qual pertence;
- Esforço empregado em sua correção;
- Fluxo em que foi injetado (para a variação 1);
- Iteração em que foi injetado (para a variação 2);
- Iteração em que foi corrigido (para a variação 2);
- Classificação quanto à espécie (para a variação 3).

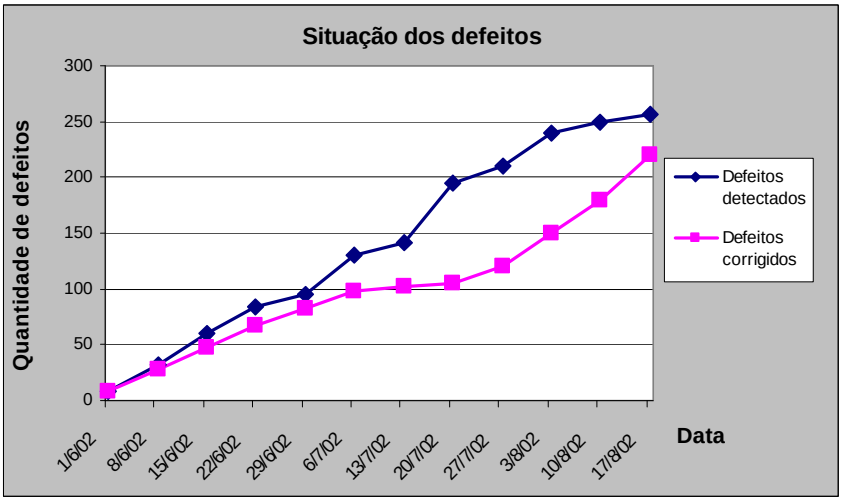
Referências O registro de defeitos é detalhado na seção 4.1.5.

I-6 Situação dos defeitos

Descrição

Exibe a evolução da quantidade acumulada de defeitos detectados e corrigidos ao longo do desenvolvimento de um produto. A diferença entre as duas corresponde à quantidade de defeitos abertos (defeitos detectados mas ainda não corrigidos), a cada momento.

Ilustração



Aplicabilidade

A comparação entre a quantidade de defeitos detectados e corrigidos ao longo do desenvolvimento permite a detecção de eventuais problemas de qualidade ou riscos de atraso.

A taxa de correção deve permanecer próxima à taxa de detecção. Grandes diferenças entre as duas linhas podem indicar que as correções dos defeitos estão sendo adiadas, o que pode resultar em problemas de prazo e recursos em etapas posteriores do projeto [DoD00].

Quedas na taxa de detecção de defeitos podem significar tanto uma maior maturidade do produto quanto problemas nas atividades de teste e revisão.

Variações possíveis

O indicador pode ser criado considerando-se todos os registros de defeitos de um projeto, ou apenas defeitos classificados como sendo de determinada espécie.

Medidas necessárias

Para cada defeito detectado:

- Projeto ao qual pertence;
- Data em que foi detectado;
- Data em que foi corrigido;
- Classificação quanto à espécie (caso o indicador seja referente apenas a defeitos de determinada espécie).

Referências

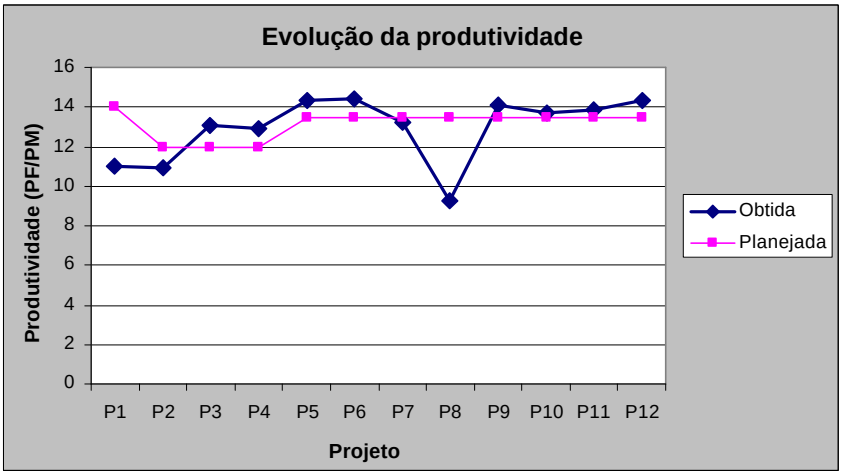
O registro de defeitos é detalhado na seção 4.1.5.

I-7 Evolução da produtividade

Descrição

Exibe a evolução da produtividade da organização ao longo dos projetos realizados.

Ilustração



Aplicabilidade

A produtividade está relacionada à quantidade de trabalho produzida por cada pessoa-mês (uma pessoa-mês equivale a 160 horas trabalhadas).

O acompanhamento da evolução da produtividade permite:

- Comparar o desempenho da organização em diferentes projetos;
- Observar tendências de queda ou crescimento;
- Verificar a estabilidade do processo – grandes variações de produtividade ou diferenças significativas entre a produtividade prevista e a realizada significam que o processo é pouco previsível;
- Acompanhar o efeito de mudanças em pessoas, processos ou tecnologia sobre a produtividade.

Variações possíveis

O indicador pode ser construído para representar a produtividade da organização sob diferentes aspectos:

1. Produtividade total (considerando todo o esforço dedicado aos projetos);
2. Produtividade de um fluxo específico (considera-se apenas o esforço dedicado àquele fluxo);
3. Produtividade de uma iteração específica (considera-se apenas o esforço dedicado àquela iteração).

Além disso, a produtividade pode ser calculada em função do tamanho do produto (em Pontos de Função) ou do tamanho de algum de seus artefatos (em alguma unidade de medida previamente definida). Um exemplo de produtividade em função de um artefato seria a quantidade de linhas de código produzida por pessoa-mês (nesse caso, o artefato seria o código-fonte e a unidade de medida seria o número de linhas de código).

Medidas necessárias

Tamanho do produto (em Pontos de Função ou em termos do tamanho de algum de seus artefatos);

Para cada atividade realizada por um desenvolvedor:

1. Projeto à qual pertence;
2. Duração;
3. Fluxo ao qual pertence (para a variação 2);
4. Iteração à qual pertence (para a variação 3).

Para cada planejamento de projeto:

5. Tamanho estimado;
6. Esforço estimado.

Referências

O registro de esforço, que constitui a base para a construção deste indicador, é detalhado na seção 4.1.3.

I-8 Distribuição do esforço

Descrição

Distribuição percentual do esforço dedicado a um projeto (ou conjunto de projetos) pelas iterações e fluxos do processo.

Ilustração

Distribuição do esforço										
Fluxo > Iteração \/	RQ	AN	DS	TS	IM	GP	GQ	ES	EP	Total %
LR	5,5%	3,0%	0,7%	0,1%	0,3%	0,3%	0,1%	0,0%	0,0%	10,0%
AR	2,0%	7,5%	3,5%	0,4%	1,0%	0,4%	0,2%	0,0%	0,0%	15,0%
DI	0,3%	1,5%	8,5%	4,0%	5,0%	0,4%	0,3%	0,0%	0,0%	20,0%
LI	0,2%	0,2%	2,0%	4,0%	8,0%	0,4%	0,2%	0,0%	0,0%	15,0%
L2	0,2%	0,2%	2,0%	4,0%	8,0%	0,4%	0,2%	0,0%	0,0%	15,0%
L3	0,1%	0,1%	1,0%	3,2%	5,0%	0,4%	0,2%	0,0%	0,0%	10,0%
TA	0,0%	0,0%	0,0%	3,0%	1,6%	0,2%	0,2%	0,0%	0,0%	5,0%
TB	0,0%	0,0%	0,0%	3,0%	1,0%	0,2%	0,2%	0,6%	0,0%	5,0%
OP	0,0%	0,0%	0,0%	3,0%	0,6%	0,2%	0,2%	1,0%	0,0%	5,0%
Total %	8,3%	12,5%	17,7%	24,7%	30,5%	2,9%	1,8%	1,6%	0,0%	100,0%

Adaptado de [PAULA03].

Aplicabilidade

No contexto de um projeto, o conhecimento da distribuição típica do esforço pelos fluxos e iterações do processo é importante para as atividades de planejamento, possibilitando uma distribuição mais adequada dos recursos a serem disponibilizados em cada etapa.

No contexto da organização, esse indicador é útil para as atividades de melhoria de processo, que podem se concentrar sobre os fluxos e iterações que consomem a maior parte dos recursos. Além disso, a comparação da distribuição obtida em diferentes projetos pode fornecer informações sobre a estabilidade e a previsibilidade do processo.

Variações possíveis

Esse indicador pode ser construído de duas formas:

1. Refletindo os valores reais obtidos ao longo da execução de um projeto;
2. Refletindo os valores planejados para um projeto.

Caso o indicador seja referente aos dados reais, a distribuição do esforço pode ser referente a um projeto ou a um conjunto de projetos.

Medidas necessárias

Para cada atividade realizada por um desenvolvedor (variação 1):

- Projeto à qual pertence;
- Duração;
- Fluxo ao qual pertence;
- Iteração à qual pertence.

Para cada planejamento de projeto (variação 2):

- Esforço planejado para cada fluxo, em cada iteração.

Referências

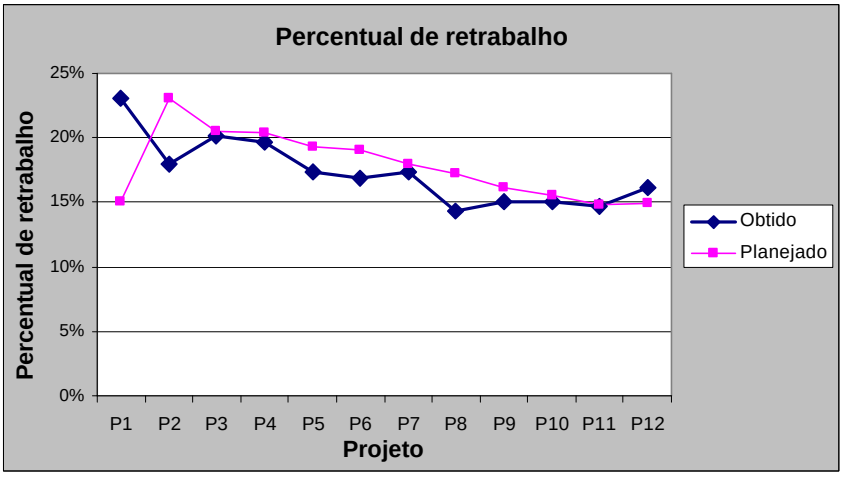
O registro de esforço é detalhado na seção 4.1.2.

I-9 Percentual de retrabalho

Descrição

Exibe a fração do esforço total de cada projeto que foi dedicada a atividades que representam retrabalho (correção de defeitos e alterações em requisitos).

Ilustração



Aplicabilidade

Um dos fatores que afeta diretamente a produtividade é a quantidade de esforço que é empregada para refazer parte do trabalho, seja para corrigir um defeito, seja para implementar uma alteração de requisito.

Este indicador permite que o esforço de retrabalho seja acompanhado quantitativamente, fornecendo informações úteis para avaliar o impacto desses eventos na produtividade.

A comparação entre o valor planejado e o valor obtido para cada projeto permite também verificar a estabilidade do processo e a acurácia das estimativas.

Variações possíveis

Há três formas de considerar o percentual de retrabalho:

- Considerar o percentual total de retrabalho em cada projeto;
- Considerar apenas o retrabalho representado pelas correções dos defeitos;
- Considerar apenas o retrabalho representado pelas alterações em requisitos.

Medidas necessárias

Para cada defeito corrigido:

- Projeto ao qual pertence;
- Esforço empregado em sua correção.

Para cada alteração em requisito registrada:

- Projeto ao qual pertence;
- Esforço empregado em sua correção.

Para cada projeto

- Percentual estimado de retrabalho em decorrência de defeitos;
- Percentual estimado de retrabalho em decorrência de alterações em requisitos.

Referências

O registro de defeitos é descrito na seção 4.1.5.

O registro das alterações em requisitos é apresentado na seção 4.1.2.

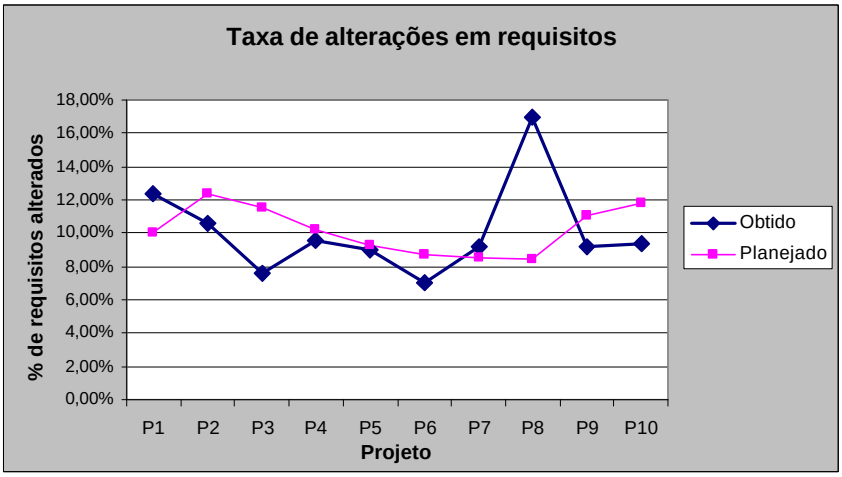
A coleta dos valores planejados para as medidas é descrita na seção 4.1.7.

I-10 Taxa de alterações em requisitos

Descrição

Exibe a evolução da taxa de alterações em requisitos ocorridas ao longo de cada projeto.

Ilustração



Aplicabilidade

Alterações em requisitos têm impacto direto sobre a produtividade, já que exigem a reconstrução de parte dos artefatos produzidos. Por isso, é importante controlar o volume de alterações desse tipo ocorridas nos projetos. Taxas muito altas de alterações em requisitos são sintomas de problemas nas atividades de levantamento de requisitos ou na comunicação com o cliente.

Como em outros indicadores, o acompanhamento da taxa planejada em relação à obtida permite verificar a estabilidade e previsibilidade do processo.

Variações possíveis

A taxa de alterações em requisitos pode ser expressa de duas formas diferentes:

1. O percentual representado pelo número de requisitos alterados em relação ao número total de requisitos, em cada projeto.
2. O percentual representado pelo tamanho de todas as alterações em requisitos registradas para um projeto (em pontos de função ajustados) em relação ao tamanho total do projeto.

Medidas necessárias

Para a variação 1:

- Quantidade de requisitos que sofreram alguma alteração, para cada projeto;
- Quantidade total de requisitos em cada projeto.

Para a variação 2:

- Tamanho de cada alteração de requisito registrada, em pontos de função ajustados;
- Tamanho total de cada projeto, em pontos de função ajustados.

Referências

O registro das alterações em requisitos é descrito na seção 4.1.2.

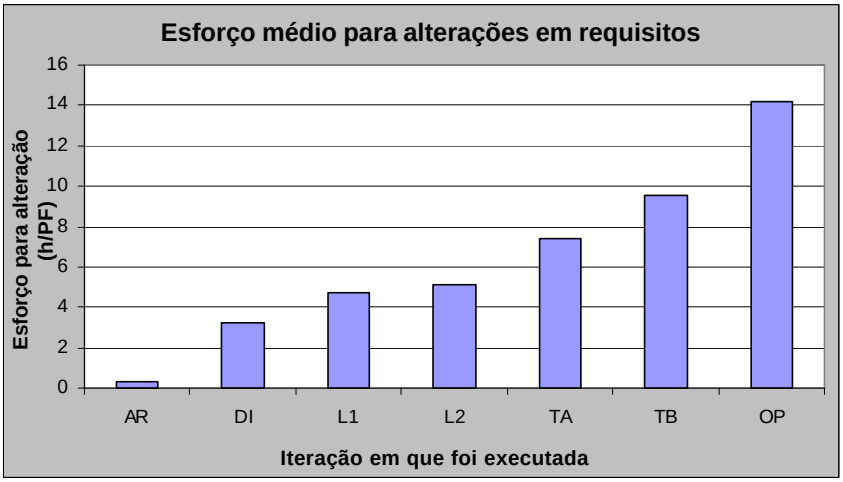
O registro da estimativa de alterações em requisitos é apresentado na seção 4.1.7.2.

I-11 Esforço médio para alterações em requisitos

Descrição

Exibe o esforço médio para a implementação de alterações em requisitos, por unidade de tamanho, em função da iteração em que a alteração foi realizada.

Ilustração



Aplicabilidade

O custo para a realização de uma alteração em requisito depende do momento em que a alteração é realizada. Quanto mais tarde um requisito é alterado, maior será o esforço necessário para implementar essa alteração, já que alterações desse tipo implicam na perda de parte do trabalho já realizado.

Quando uma alteração de requisito é solicitada pelo cliente, é necessário fazer uma estimativa do impacto que essa alteração pode trazer para o andamento do projeto. O indicador ilustrado permite a realização desse tipo de estimativa, exibindo o esforço típico necessário para a realização de alterações em requisito, para cada unidade de tamanho. Uma vez conhecendo-se o tamanho da alteração e a iteração em que ela está ocorrendo, é possível utilizar este indicador para estimar o seu impacto no esforço total do projeto.

Variações possíveis

O indicador pode ser construído para um projeto específico ou para um conjunto de projetos.

Medidas necessárias

Para cada alteração de requisito realizada:

- Projeto ao qual pertence;
- Iteração em que ocorreu;
- Tamanho da alteração (em pontos de função ajustados)
- Esforço empregado para alterar e validar todos os artefatos impactados.

Referências

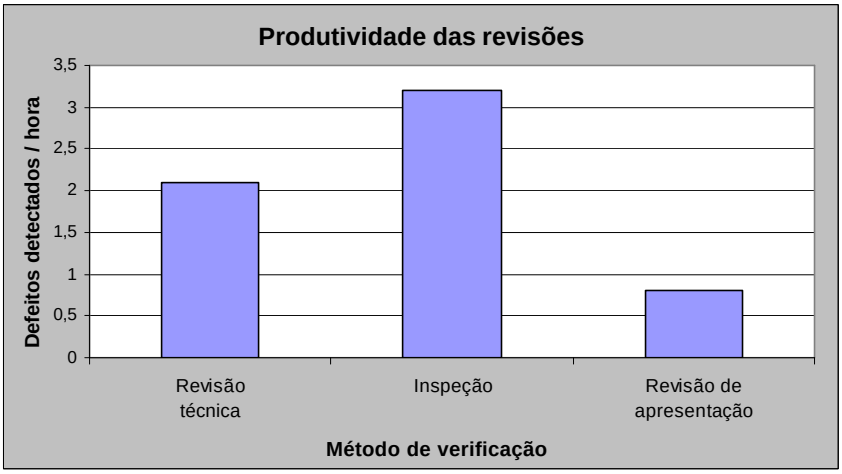
O registro das alterações em requisitos é descrito na seção 4.1.2.

I-12 Produtividade das revisões

Descrição

Exibe a produtividade de cada tipo de revisão, em termos do número de defeitos detectados por hora de revisão.

Ilustração



Aplicabilidade

No contexto da organização, o acompanhamento da produtividade das revisões em função do método de verificação utilizado fornece suporte às atividades de melhoria do processo, permitindo identificar os métodos mais eficazes (que podem então ser incentivados) e os menos eficazes (que podem ser alvo de melhorias).

No contexto de um projeto, a produtividade de cada revisão pode ser comparada aos valores exibidos neste indicador. Uma taxa de detecção de defeitos muito alta pode significar problemas de qualidade no produto revisado, enquanto uma taxa muito baixa pode indicar que o processo de revisão não está sendo conduzido corretamente.

Variações possíveis

O indicador pode se referir a um projeto específico ou a um conjunto de projetos. Além disso, duas abordagens podem ser adotadas:

1. Considerar todas as revisões realizadas;
2. Considerar apenas as revisões realizadas sobre o material de um fluxo específico.

Além disso, em alguns casos pode ser interessante considerar apenas defeitos de determinada espécie.

Medidas necessárias

Para cada revisão realizada:

- Projeto a qual pertence;
- Método de verificação utilizado;
- Quantidade de defeitos detectados;
- Esforço total dedicado à revisão;
- Fluxo ao qual pertence o material revisado (para a variação 2).
- Classificação de cada defeito detectado, quanto à espécie (para os casos em que forem considerados apenas defeitos detectados de determinada espécie).

Referências

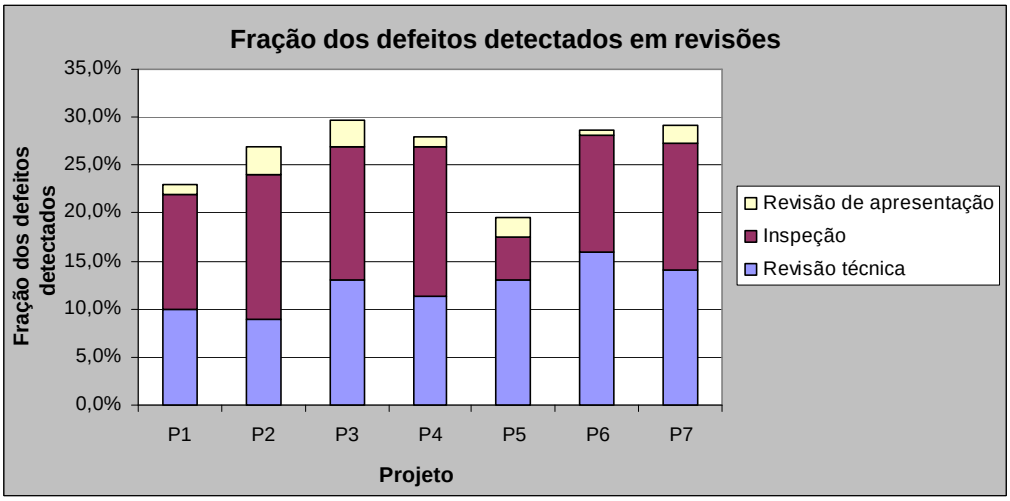
A coleta de informações sobre as revisões é descrita na seção 4.1.6.

I-13 Fração dos defeitos detectados em revisões

Descrição

Exibe a fração dos defeitos que é detectada em cada tipo de revisão.

Ilustração



Aplicabilidade

A maior vantagem das revisões é a possibilidade de identificação precoce de defeitos, reduzindo os custos necessários para correção. Por isso, é interessante que a maior fração possível de defeitos seja detectada nas revisões.

Projetos cuja fração de defeitos detectados em revisões seja muito baixa podem indicar problemas com a condução das atividades de revisão. Além disso, o gráfico permite comparar a fração representada por cada método de verificação, facilitando a investigação da causa de eventuais flutuações em alguns projetos.

Além disso, a observação desses dados ao longo dos projetos permite o acompanhamento da estabilidade do processo.

Variações possíveis

Há três abordagens possíveis:

1. Considerar todos os defeitos registrados;
2. Considerar apenas os defeitos de determinada espécie;
3. Considerar apenas os defeitos injetados em determinado fluxo.

Para cada uma das situações acima, há ainda duas alternativas: considerar defeitos pertencentes a um projeto individual, ou considerar um conjunto de projetos.

Medidas necessárias

Para cada defeito detectado:

- Projeto ao qual pertence;
- Revisão em que foi detectado (caso tenha sido detectado em alguma revisão);
- Classificação quanto à espécie (para a variação 2);
- Fluxo em que foi injetado (para a variação 3);

Para cada revisão realizada:

1. Projeto ao qual pertence;
2. Método de verificação utilizado.

Referências

A coleta de informações sobre as revisões é descrita na seção 4.1.6.

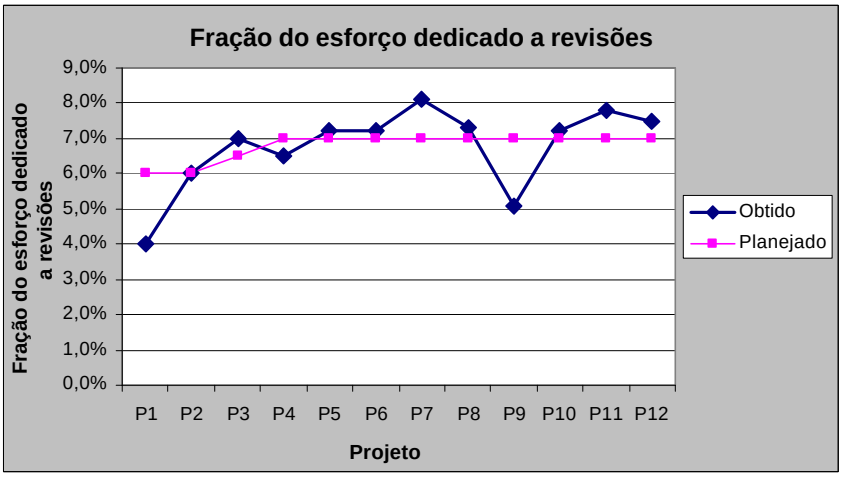
O registro de defeitos é apresentado na seção 4.1.5.

I-14 Fração do esforço dedicado a revisões

Descrição

Exibe a fração do esforço de cada projeto que é dedicada a atividades de revisão.

Ilustração



Aplicabilidade

Além do acompanhamento da produtividade das revisões (indicador I-12) e do papel destas na detecção dos defeitos (indicador I-13), é importante acompanhar a fração do esforço de cada projeto que é dedicada a atividades de revisão.

Através desse indicador, é possível acompanhar o custo relativo das revisões em relação ao custo total dos projetos, e verificar se realmente todo o esforço planejado para elas está sendo devidamente aplicado para esse fim.

Frações muito elevadas podem indicar baixa produtividade no processo de revisão ou excesso de defeitos no produto (o que torna as revisões mais trabalhosas). Por outro lado, percentuais muito baixos podem significar que as atividades de revisão não estão sendo realizadas conforme o planejado.

Variações possíveis

Há basicamente três variações:

1. Considerar todas as revisões em relação ao esforço total dos projetos;
2. Considerar apenas as revisões e o esforço referentes a um fluxo específico;
3. Considerar apenas as revisões e o esforço referentes a uma iteração específica.

Para cada uma delas, ainda é possível considerar todos os tipos de revisões ou apenas aquelas que sigam determinado método de verificação.

Medidas necessárias

Para cada revisão realizada:

- Projeto ao qual pertence;
- Esforço total dedicado à revisão;
- Fluxo ao qual pertence o material revisado (para a variação 2);
- Iteração em que a revisão foi executada (para a variação 3);
- Método de verificação utilizado (para os casos em que for considerado apenas um método de verificação específico).

Para cada projeto:

- Esforço total dedicado ao projeto (para a variação 1);
- Esforço total dedicado a cada fluxo do processo (para a variação 2);
- Esforço total dedicado a cada iteração do processo (para a variação 3).

Referências

A coleta de informações sobre as revisões é descrita na seção 4.1.6.

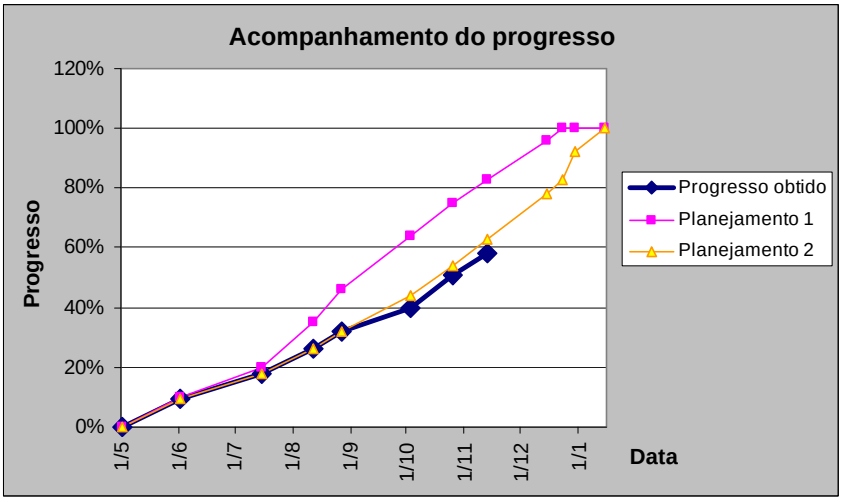
O registro de esforço é tratado na seção 4.1.2.

I-15 Acompanhamento do progresso

Descrição

Exibe a evolução do progresso obtido ao longo da execução de um projeto e a sua comparação com os valores planejados.

Ilustração



Aplicabilidade

O acompanhamento acurado do progresso obtido ao longo da realização de um projeto permite que riscos de atraso em relação aos prazos assumidos sejam identificados o mais cedo possível, permitindo que providências possam ser tomadas em tempo hábil. Qualquer desvio significativo do progresso obtido em relação ao planejamento mais recente deve ser examinado, e pode indicar a necessidade de um replanejamento ([DOD00] recomenda que desvios superiores a 20% em qualquer um dos pontos, ou desvios acumulados superiores a 10% sejam considerados alarmantes).

O progresso é calculado a partir do estado dos requisitos em cada marco de projeto. Por estarem baseados nos requisitos do produto – que por sua vez estão diretamente relacionados à visão que o usuário tem do produto –, a evolução do progresso pode ser utilizada também para informar aos clientes e usuários o estado atual de um projeto.

É possível ocorrer uma redução no progresso entre marcos de projeto consecutivos. Isso pode ocorrer quando houver um acréscimo no tamanho do produto (em função de uma alteração de requisito, por exemplo) ou quando o estado de algum requisito regredir (o que pode ser por alteração em requisito ou por detecção tardia de defeitos).

Variações possíveis

Há duas formas de construir este indicador, dependendo do conceito que é utilizado para representar o andamento do projeto [PAULA03]:

1. Progresso: calculado a partir do estado em que se encontra cada requisito. Cada estado representa um percentual de progresso, e o progresso total é calculado como a média do percentual de progresso de cada requisito ponderada pelo respectivo tamanho.
2. Valor: calculado de forma semelhante ao item anterior, mas para o valor só são considerados os requisitos que atingiram um progresso de 100%. Estados intermediários, nesse caso, são considerados como sendo um progresso de 0%.

Medidas necessárias

Para cada marco de projeto (real ou planejado):

- Projeto ao qual pertence;
- Planejamento ao qual pertence (no caso de marcos planejados);
- Data do marco;
- Tamanho de cada requisito;
- Estado em que de encontra cada requisito.

Referências

O conceito de *marco de projeto* e o registro do progresso estão detalhados na seção 4.1.4.

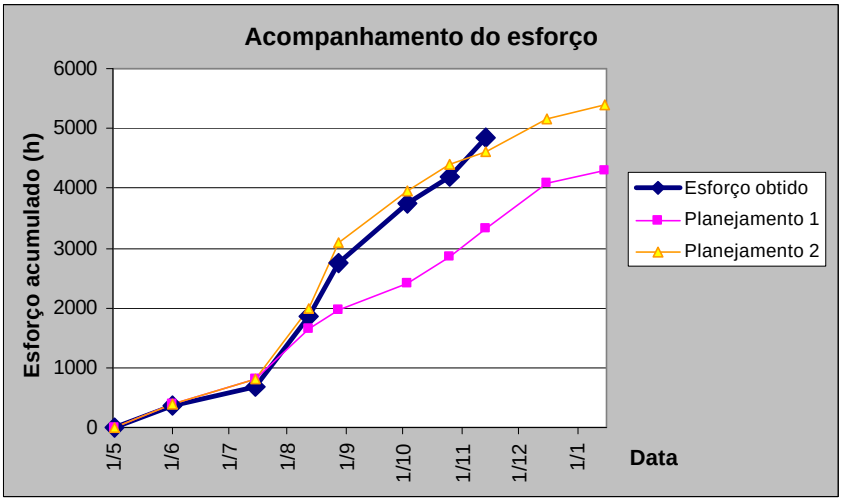
Detalhes sobre o conceito de *estado* dos requisitos podem ser vistos na seção 4.2.2

I-16 Acompanhamento do esforço

Descrição

Exibe a evolução do esforço consumido ao longo da execução de um projeto e a sua comparação com os valores planejados.

Ilustração



Aplicabilidade

Assim como o acompanhamento do progresso permite a redução de riscos de descumprimento de prazos, o acompanhamento do esforço permite o controle do custo do projeto, monitorando o volume de recursos humanos consumidos no decorrer do projeto e comparando-o com os valores planejados.

De forma análoga ao indicador I-15, o objetivo deste indicador é explicitar o quanto antes possível a ocorrência de desvios significativos em relação aos planejamentos. Na verdade, esses indicadores devem ser sempre analisados em conjunto: o cumprimento das metas de esforço ao longo do tempo só indica um bom andamento do projeto se o progresso também estiver sendo cumprido conforme o planejado.

Desvios podem indicar a necessidade de replanejamento.

Variações possíveis

-

Medidas necessárias

Para cada planejamento de projeto:

- Projeto ao qual pertence;
- Esforço planejado para cada iteração;
- Data do marco final de cada iteração;

Para cada esforço registrado ao longo da execução do projeto:

- Projeto ao qual se refere;
- Data;
- Duração.

Referências

O planejamento do esforço é descrito na seção 4.1.7.1.

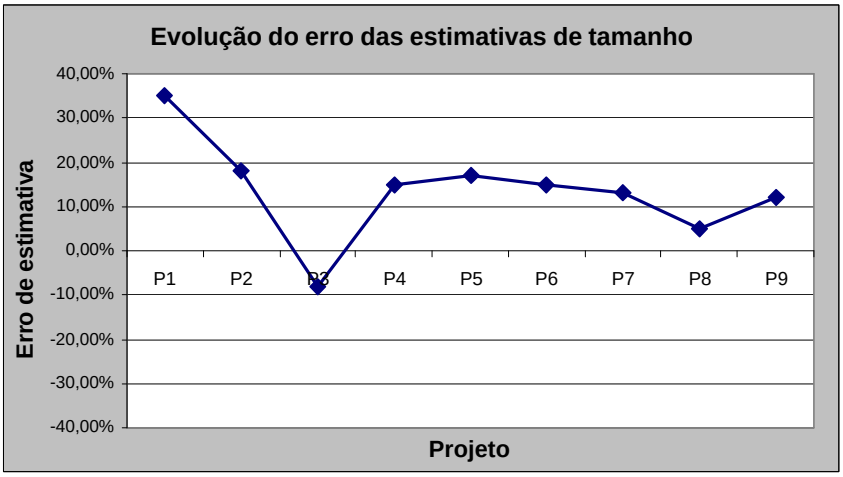
O registro de esforço ao longo dos projetos é apresentado na seção 4.1.2.

I-17 Evolução do erro de estimativas

Descrição

Apresenta o erro das estimativas contidas nos planejamentos dos projetos em relação aos valores obtidos.

Ilustração



Aplicabilidade

Previsibilidade é uma característica desejável a qualquer processo de desenvolvimento, pois permite a execução de planejamentos mais acurados e a conseqüente redução dos riscos dos projetos. Quanto menor o erro das estimativas produzidas, maior a previsibilidade do processo.

No contexto de um projeto, o conhecimento do erro típico das estimativas feitas pela organização é importante para o cálculo dos fatores de contingência a serem considerados no planejamento.

No contexto da organização, esse acompanhamento permite um melhor conhecimento da estabilidade do processo e fornece subsídios para as atividades de melhoria de processo.

Variações possíveis

Este indicador pode ser utilizado para monitorar o erro de qualquer estimativa prevista no modelo de medição. Há, portanto, uma série de variações:

1. Erro das estimativas de tamanho (em pontos de função ou em função de medidas realizadas sobre os artefatos);
2. Erro das estimativas de esforço (que pode considerar o esforço total dos projetos ou apenas o esforço de determinado fluxo ou iteração);
3. Estimativas da densidade de defeitos (que podem considerar todos os defeitos ou apenas os defeitos injetados em determinado fluxo ou iteração);
4. Erro das estimativas de produtividade;
5. Erro das estimativas da fração estimada de defeitos detectados em revisões;
6. Erro do percentual estimado de esforço dedicado a correção de defeitos;
7. Erro do percentual estimado de requisitos alterados (em número de requisitos ou em tamanho das alterações em relação ao tamanho total do projeto);
8. Erro do percentual estimado de esforço dedicado a alterações em requisitos.
9. Erro da fração estimada de esforço dedicada às atividades de revisão.

Medidas necessárias

Para cada planejamento de projeto:

- Projeto ao qual se refere;
- Tamanho estimado para o produto em pontos de função em termos das medidas de seus artefatos (para as variações 1, 3 e 4);
- Esforço estimado para cada fluxo em cada iteração (para as variações 2 e 4);
- Quantidade estimada de defeitos injetados em cada fluxo e cada iteração (para a variação 3);
- Fração estimada de defeitos detectados em revisões (para a variação 5);
- Percentual estimado de esforço dedicado a correção de defeitos (para a variação 6);
- Percentual estimado de requisitos alterados, e em número de requisitos e em tamanho das alterações (para a variação 7);
- Percentual estimado de esforço dedicado a alterações em requisitos (para a variação 8).

- Fração estimada de esforço a ser dedicada às atividades de revisão (para a variação 9).

Além disso, os valores reais obtidos para todas essas medidas ao longo do projeto devem ser registrados.

Referências A seção 4.1.7 descreve o registro dos valores estimados para todas as medidas mencionadas.
O registro dos valores reais é descrito nas seções 4.1.1 a 4.1.6.

A.3 Relação de medidas derivadas

Este apêndice descreve um conjunto de medidas derivadas que podem ser obtidas a partir do modelo proposto, conforme descrito na seção 4.3. O objetivo aqui é definir medidas que possam ser utilizadas para construir os indicadores descritos no Apêndice A.2 e exemplificar como diferentes medidas básicas podem ser combinadas entre si para fornecer informações úteis. Além das relacionadas aqui, várias outras medidas derivadas podem ser calculadas durante a análise dos dados coletados.

Para cada medida proposta são apresentados o nome, uma breve descrição e algumas informações adicionais:

- Unidade – Unidade de medida através da qual o valor obtido é expresso.
- Parâmetros – Várias das medidas propostas são parametrizáveis, podendo ser utilizada para mensurar elementos diferentes. Por exemplo, a produtividade pode ser calculada para um projeto completo, ou para um fluxo ou iteração específico do processo.
- Forma de cálculo – Descreve como o valor da medida descrita deve ser calculado a partir de medidas simples contidas no modelo de medição.
- Indicadores – Relação dos indicadores, entre aqueles descritos no Apêndice A.2, que utilizam a medida em sua construção.

A relação completa é apresentada na tabela a seguir. De todas as medidas derivadas apresentadas, as quatro primeiras se aplicam somente no contexto de um projeto individual. As demais podem ser calculadas sobre os dados de um único projeto ou de um conjunto de projetos que utilizem o mesmo processo.

Medida	Descrição	Unidade	Parâmetros	Forma de cálculo	Indicadores
Tam (MP)	Tamanho do produto em um marco de projeto	Pontos de Função Ajustados	MP – Marco de projeto no qual o tamanho deve ser calculado.	*Tam = TamReq(MP) * FatorAjuste(MP)* Onde: *TamReq(MP)* = Somatório do tamanho de todos os requisitos, em Pontos de Função Brutos, conforme registrado no formulário referente ao marco de projeto em questão. *FatorAjuste(MP)* = Fator de ajuste para a contagem de Pontos de Função, conforme registrado no formulário referente ao marco de projeto em questão.	- (Utilizada para gerar outras medidas derivadas)
EA (MP)	Esforço acumulado até um marco de projeto	Horas Trabalhadas	MP – Marco de projeto para o qual deverá ser calculado o esforço acumulado.	*EA (MP)* = Somatório de todas as horas trabalhadas registradas para datas anteriores ou iguais à data do marco de projeto em questão. Considera-se apenas as horas registradas para o projeto ao qual pertence o marco.	I-16
Prog (MP)	Progresso obtido em um marco de projeto	%	MP – Marco de projeto para o qual deverá ser calculado o progresso.	$Prog\ (MP) = \frac{E(R1)*T(R1) + E(R2)*T(R2) + ... + E(Rn)*T(Rn)}{T(R1) + T(R2) + ... + R(Rn)} * 100$ Onde: *R1, R2, ..., Rn* são os requisitos do produto que está sendo desenvolvido; *E(R*i*)* = Percentual de progresso representado pelo estado em que se encontra o requisito *i* (de 0 a 1); *T(R*i*)* = Tamanho do requisito *i*, em pontos de função brutos.	I-15
Valor (MP)	Valor obtido em um marco de projeto (difere do progresso por só considerar os requisitos que obtiveram 100% de progresso).	%	MP – Marco de projeto para o qual deverá ser calculado o valor.	$Valor\ (MP) = \frac{T(RC1) + T(RC2) + ... + T(RCn)}{TotalPFBrutos} * 100$ Onde: *RC1, RC2, ..., RCn* são os requisitos do produto que já atingiram o estado que representa 100% de progresso. *T(RC*i*)* = Tamanho do requisito *i*, em pontos de função brutos. *TotalPFBrutos* = Total de pontos de função brutos do produto.	I-15
ESP (SP)	Esforço total dedicado a uma subdivisão do processo	Horas Trabalhadas	SP – Subdivisão do processo (fase, fluxo, iteração ou o processo como um todo) para a qual se deseja conhecer o esforço dedicado.	*ESP (SP)* = Somatório das horas trabalhadas informadas em todos os registros de esforço referentes à subdivisão do processo (fase, fluxo ou iteração) em questão.	I-8
DDef (SP)	Densidade de defeitos injetados em uma subdivisão do processo.	Defeitos / Ponto de Função	SP – Subdivisão do processo (fase, fluxo, iteração ou o processo como um todo) a ser levada em consideração.	*DDef(SP) = TotalDefeitosInjetados(SP) / TamanhoProduto* Onde: *TotalDefeitosInjetados(SP)* = Número de defeitos registrados que foram injetados na subdivisão do processo em questão. Não considerar defeitos duplicados. *TamanhoProduto* = Tamanho total do produto, em Pontos de Função Ajustados. Pode ser calculado como sendo a medida derivada *Tam(MP)* calculada para o marco de projeto mais recente.	I-4

PDefInj (ED, SP)	Percentual de defeitos de determinada espécie injetados em uma subdivisão do processo.	%	SP – Subdivisão do processo (fase, fluxo, iteração ou o processo como um todo) a ser levada em consideração.	*PDefInj (ED, SP) = NumDefInjetados(ED, SP) * 100 / TotalDefInjetados(SP)* Onde: *NumDefInjetados(ED, SP)* = Número de defeitos da espécie ED injetados na subdivisão do processo em questão. Não considerar defeitos duplicados. *TotalDefInjetados(SP)* = Número total de defeitos injetados na subdivisão do processo em questão. Não considerar defeitos duplicados.	I-1, I-3
PDefCor (ED, SP)	Percentual de defeitos de determinada espécie detectados em uma subdivisão do processo.	%	SP – Subdivisão do processo (fase, fluxo, iteração ou o processo como um todo) a ser levada em consideração.	*PDefCor (ED, SP) = NumDefCor(ED, SP) * 100 / TotalDefCor(SP)* Onde: *NumDefCor (ED, SP)* = Número de defeitos da espécie ED corrigidos na subdivisão do processo em questão. Não considerar defeitos duplicados. *TotalDefCor(SP)* = Número total de defeitos corrigidos na subdivisão do processo em questão. Não considerar defeitos duplicados.	I-1, I-3
Prod (SP)	Produtividade obtida em uma subdivisão do processo.	Ponto de Função / Pessoa-Mês	SP – Subdivisão do processo (fase, fluxo, iteração ou o processo como um todo) para a qual se deseja obter a produtividade.	*Prod(SP) = ESP (SP) / TamanhoProduto* Onde: *ESP (SP)* = Esforço total dedicado à subdivisão em questão (ver medida derivada de mesmo nome). *TamanhoProduto* = Tamanho total do produto, em Pontos de Função Ajustados. Pode ser calculado como sendo a medida derivada *Tam(MP)* calculada para o marco de projeto mais recente.	I-7
PRetD	Percentual de retrabalho representado pelas correções de defeitos	%	-	*PRetD = EsforçoCorreçãoDefeitos * 100 / EsforçoTotal* Onde: *EsforçoCorreçãoDefeitos* = Somatório do esforço dedicado à correção de cada defeito. Não considerar defeitos duplicados. *EsforçoTotal* = Esforço total, calculado como a medida *ESP(SP)*, considerando-se o processo como um todo.	I-9
PRetAR	Percentual de retrabalho representado pelas alterações em requisitos.	%	-	*PRetAR = EsforçoAlterações * 100 / EsforçoTotal* Onde: *EsforçoAlterações* = Somatório do esforço dedicado a cada alteração em requisitos. *EsforçoTotal* = Esforço total, calculado como a medida *ESP(SP)*, considerando-se o processo como um todo.	I-9
PRet	Percentual total de retrabalho	%	-	*PRet = PRetD + PRetAR*. Onde: *PRetD* e *PRetAR* são as medidas derivadas de mesmo nome.	I-9

PDefRev (MV, SP)	Percentual de defeitos detectados em revisões que utilizam determinado método de verificação, em uma determinada subdivisão do processo.	%	MV – Método de verificação a ser medido; SP – Subdivisão do processo (fase, fluxo, iteração ou o processo como um todo) a ser levada em consideração.	PDefRev (MV, SP) = *TotalDefRevisoes(MV, SP) * 100 / TotalDefDetectados(SP)* Onde: *TotalDefRevisoes(MV, SP)*: Número total de defeitos detectados em revisões realizadas na subdivisão do processo em questão. Considerar apenas revisões que utilizem o método de verificação especificado como MV. Não considerar defeitos duplicados. *TotalDefDetectados(SP)*: Total de defeitos detectados na subdivisão do processo em questão. Não considerar defeitos duplicados.	I-13
PReqAlt	Percentual de requisitos alterados	%	-	*PReqAlt = NumRequisitosAlterados * 100 / NumTotalRequisitos* Onde: *NumRequisitosAlterados* = Número total de requisitos alterados ao longo da execução de um projeto. Devem ser contados todos os requisitos mencionados no registro de alterações em requisitos apresentado na seção 4.1.2. *NumTotalRequisitos* = Número total de requisitos do produto em desenvolvimento.	I-10
PTamAlt	Percentual representado pelo tamanho total de todas as alterações, comparado ao tamanho total do produto	%	-	*PTamAlt = TamAlterações * 100 / TamanhoProduto* Onde: *TamAlterações* = Somatório do tamanho de cada alteração em requisito registrada, em Pontos de Função ajustados. O registro do tamanho das alterações é descrito na seção 4.1.2. *TamanhoProduto* = Tamanho total do produto, em Pontos de Função ajustados. Pode ser calculado como sendo a medida derivada *Tam(MP)* calculada para o marco de projeto mais recente.	I-10
EMCD (ED)	Esforço médio para correção de defeitos de determinada espécie.	Horas trabalhadas / Defeito	ED – Espécie de defeito a ser considerada	*EMCD (ED) = EsforçoCorrecaoDefeitos(ED) / NúmeroDeDefeitos(ED)* Onde: *EsforçoCorrecaoDefeitos(ED)* = Somatório do esforço dedicado à correção defeitos da espécie ED. Não considerar defeitos duplicados. *NúmeroDeDefeitos(ED)* = Número de defeitos registrados para a espécie ED. Não considerar defeitos duplicados.	I-5
EMAR (IT)	Esforço médio para a implantação de alterações em requisitos injetados em determinada iteração do processo.	Horas trabalhadas / Ponto de Função	IT – Iteração do processo a ser considerada.	*EMAR (IT) =EsforçoAlterações(IT) / TamanhoAlterações(IT)* Onde: *EsforçoAlterações(IT)* = Somatório do esforço dedicado a alterações de requisitos concluídas na iteração IT. *TamanhoAlterações(IT)* = Somatório do tamanho de cada alteração em requisito concluída na a iteração IT, em Pontos de Função ajustados.	I-11
Erro (VE, VO)	Erro de determinada estimativa, calculado a partir do valor estimado e do valor obtido na prática.	%	VE – Valor estimado; VO – Valor obtido.	*Erro (VE, VO) = 100 * (VO – VE) / VE* Definição adaptada de [HUMPHREY95].	I-17