Vitor Alcântara Batista

Um programa de melhoria de estimativas em uma organização desenvolvedora 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

Junho de 2009

[Dedicatória]

Agradecimentos

[Agradecimentos]

Resumo

O crescente aumento no tamanho e complexidade dos produtos de software atualmente demandados, força as organizações desenvolvedoras a também melhorarem continuamente seus processos de gestão dos projetos. Nesse contexto, realizar boas estimativas é crucial para um bom planejamento e respectivo sucesso dos projetos em termos do cumprimento de seus compromissos. O trabalho relatado aqui apresenta um programa de melhoria de estimativas realizado em uma organização desenvolvedora de software. O programa foi executado dentro de um processo bem definido e abrange estimativas em todas as fases de seu processo de desenvolvimento. As melhorias de processo propostas utilizam métodos baseados nas melhores práticas de estimativas relatadas na literatura e foram validadas em projetos piloto para, então, serem incorporadas nos processos da organização. Os resultados desse programa servirão como base para outras organizações que desejam investir em melhoria de suas estimativas.

Abstract

The constant growth in size and complexity of today's software products forces software development organizations to continuously improve their project management processes. In this context, performing good estimation is vital to good project planning and successful schedule and cost commitments. The work reported here presents an estimation improvement program, performed inside a software development organization. This program used a well defined process, covering estimation for all phases of the software development life-cycle. The improvements proposed in this work are based on best practices found in the literature, and were validated using pilot projects, before being incorporated in the organization's software development processes. The results of this program will hopefully be useful to other organizations interested in improving their estimation processes.

Sumário

Lista de figuras

Lista de tabelas

Lista de equações

Capítulo 1

Introdução

A indústria mundial de software, a cada dia que passa, depara-se com o desafio de construir produtos cada vez maiores, mais complexos e mais confiáveis. Nesse cenário, o papel de boas estimativas torna-se crucial para o sucesso nos negócios. Uma estimativa mal feita pode levar a organização a prejuízos em seus projetos ou a perder boas oportunidades de negócio por apresentar propostas com valores muito altos. Além do efeito negativo para a organização prestadora do serviço, os clientes também acabam prejudicados por verem seus projetos entregues fora do prazo ou, em casos mais graves, cancelados.

Diversas pesquisas têm sido realizadas para acompanhar o resultado de projetos de desenvolvimento de software em relação ao cumprimento de suas metas. Entre elas, uma das mais conhecidas, é o *CHAOS Report*, elaborado pelo *Standish Group*[1] a cada dois anos. Nesse relatório, ilustrado na Figura 1-1, podemos observar que a maior parte dos projetos de desenvolvimento de software sofreu algum atraso ou foram cancelados. Além disso, a evolução ao longo do tempo não mostra melhora significativa nesses resultados. Embora seja muito conhecido, o *CHAOS Report* também é alvo de críticas, como discutido por Jørgensen e Moløkken-Østvold [Jørgensen & Moløkken-Østvold, 2006] que questionam a validade estatística da amostra pesquisada.

Um levantamento bibliográfico realizado por Yang e outros [Yang *et al.*, 2008] mostra números não menos preocupantes. Segundo eles, entre 59% a 76% dos projetos apresentam esforço superior ao estimado e de 35% a 80% apresentam estouro do prazo. Além disso, dentre os projetos com esforço superior ao estimado, o erro varia entre 18% a 41%, enquanto nos projetos com prazo superior ao estimado, o erro varia entre 22% a 25%.

---

[1] http://www.standishgroup.com/

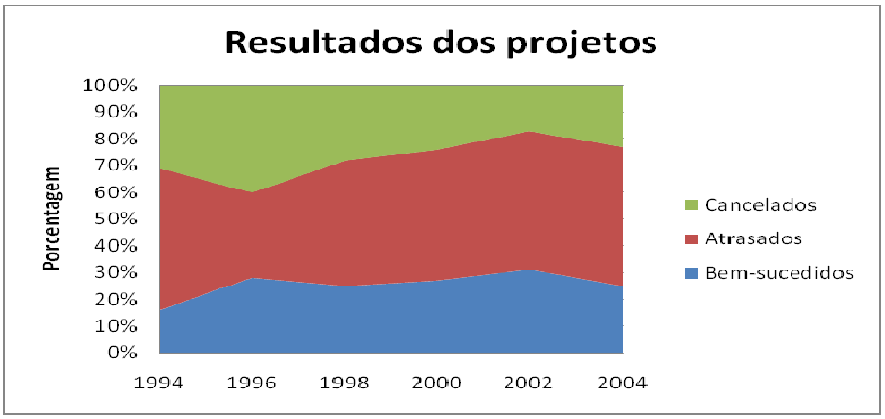

Figura 1-1 - Dados do CHAOS Report do Standish Group. Adaptado de [McConnell, 2006]

Diante desse panorama, uma organização que consegue realizar boas estimativas em seus projetos ganha um diferencial competitivo em relação ao restante do mercado, pois será capaz de oferecer serviços com o compromisso do cumprimento de prazos sem incorrer em prejuízos financeiros.

Além do benefício da competitividade no mercado, outros aspectos acabam sendo impactados positivamente dentro da organização capaz de fazer boas estimativas, como, por exemplo, o melhor planejamento e controle de seus projetos, desenvolvimento de produtos com melhor qualidade e menor pressão sobre a sua equipe. Um estudo realizado por Nan e Harter [Nan & Harter, 2009] mostra os malefícios da pressão por prazos e custos em projetos. Eles demonstraram um rápido crescimento do custo e prazo de execução do projeto quando a pressão é excessiva.

1.1 Motivação

Embora estejam claros os benefícios de uma organização realizar boas estimativas, essa é uma tarefa árdua e complexa. Vários de estudos foram realizados sobre o tema, mas ainda assim, parece estar longe de ser um problema bem resolvido. Laird especula em seu artigo "*The limitations of estimation*" [Laird, 2006] sobre possíveis causas. Entre elas, o planejamento baseado na esperança, confusão entre metas e estimativas, requisitos incompletos e instáveis e falta de treinamento dos responsáveis pelas estimativas.

McConnell, em seu livro dedicado ao tema de estimativas em projetos de software [McConnell, 2006], também sugere algumas fontes dos erros de estimativas, como falta de

informação do projeto sendo estimado, falta de informação da organização sobre sua própria capacidade produtiva, incerteza no projeto (acertar um alvo que se move) e falta de acurácia no processo de estimativa em si.

Como se pode observar acima, uma causa comum citada por ambos os autores é a falta de informação (requisitos incompletos) sobre o projeto sendo estimado. Nessa mesma linha, Boehm, durante seus trabalhos com o COCOMO [Boehm *et al.*, 2000] (um dos métodos de estimativa mais conhecidos), propôs um conceito do "Cone da incerteza", que mostra o erro potencial nas estimativas em relação à fase em que um projeto se encontra. Na Figura 1-2, que ilustra o Cone da incerteza, é possível observar que a cada fase que avançamos no projeto, a incerteza das estimativas diminui. Durante a concepção inicial, por exemplo, há um potencial de erro de 400%.

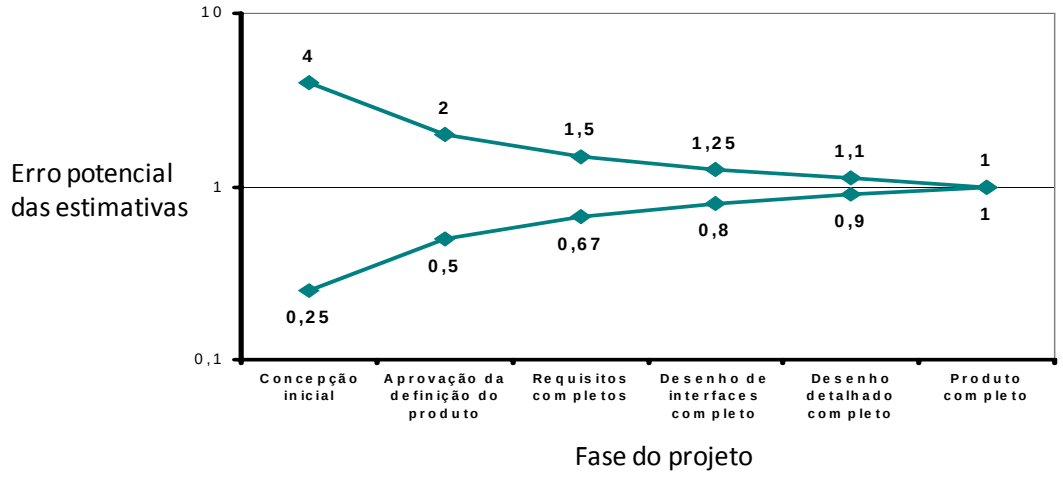

Figura 1-2 - O cone da incerteza. Adaptado de [Boehm *et al.*, 2000].

Como veremos ao longo desse trabalho, não há um consenso sobre quais são os melhores métodos de estimativa para cada situação, e cada organização deve avaliar os métodos disponíveis e escolher os que melhor se adaptam à sua realidade.

Além da dificuldade inerente na realização de boas estimativas, as organizações ainda devem lidar com a mudança de seus processos para aplicação dos métodos avaliados e escolhidos. Mudanças de processos nas organizações, por si só, já apresentam outro grande desafio e devem ser realizados dentro de um formato bem definido.

1.2 Objetivos do trabalho

O objetivo primário desse trabalho é realizar, dentro de uma organização desenvolvedora de software, um programa para melhoria de suas estimativas em projetos de desenvolvimento de novos produtos de software. Ao longo deste texto será mostrado como essas melhorias, que envolvem a mudança de processos da organização, foram realizadas dentro de um processo bem definido.

Antes de prosseguir, é necessário esclarecer os conceitos de projetos, programas e portfólios. Segundo o Project Management Institute (PMI), um projeto é “um esforço temporário empreendido para criar um produto, serviço ou resultado exclusivo" [PMI, 2004]. Tipicamente, em empresas que seguem processos, um projeto é uma instância de uma execução de um processo. Os programas são conjuntos de projetos relacionados entre si coordenados de maneira articulada, onde os objetivos de cada projeto contribuem para um objetivo maior em comum [PMI, 2008]. Já os portfólios são conjuntos de programas e/ou projetos que são agrupados de forma a facilitar o seu gerenciamento efetivo a fim de alcançar os objetivos estratégicos de uma organização [PMI, 2008].

Neste trabalho, decidiu-se chamar o conjunto de ações de melhoria nas estimativas da organização de um programa, pois já era notório que seriam necessários vários projetos distintos, mas correlatos, para alcançar esse objetivo maior. Os motivos da escolha do termo ficarão mais claros no decorrer deste texto.

O escopo do programa de melhoria envolve as estimativas realizadas em diversas fases do projeto:

- durante a concepção inicial do projeto, com o objetivo de elaborar um orçamento que servirá de base para confecção da proposta que será apresentada aos clientes;
- após a contratação do projeto, para estimar o perfil de equipe necessária e estabelecer os marcos do projeto em termos do seu escopo. Essa etapa é denominada planejamento macro;
- no início de cada iteração do projeto, para o planejamento detalhado da alocação dos colaboradores;
- nos replanejamentos do projeto, quando ocorrem alterações ou problemas em seu escopo durante a execução.

Além de propor melhorias nos processos da organização, esse trabalho tem também o objetivo de executar projetos piloto com as propostas, com a finalidade de validá-las.

Embora o programa seja realizado em uma organização específica (da qual o autor deste trabalho é colaborador), os resultados do programa poderão servir como fonte de conhecimento para outras organizações que também desejem melhorar suas estimativas, valendo-se dos métodos propostos e das dificuldades encontradas.

1.3 Limites do trabalho

Assim como é importante definir o escopo e objetivos de um trabalho, também é importante explicar suas limitações.

Primeiramente, o programa de melhoria de estimativas a ser executado tem como alvo projetos de desenvolvimento de novos produtos de software. Não serão contempladas melhorias nas estimativas de projetos de manutenção, evolução, modelagem de negócio ou consultoria, que também são serviços oferecidos na organização do autor.

Em segundo lugar, não é esperado que os resultados alcançados sejam aplicáveis a qualquer organização. Cada empresa tem suas características e particularidades, e propor processos que tenham validade para todas elas é inviável no estado atual do conhecimento.

Entretanto, isso não impede que outras organizações com características semelhantes, como por exemplo, as que possuem processo de desenvolvimento com mesma arquitetura de processo da organização, não possam se valer dos resultados apresentados aqui. Mesmo organizações com características bem diferentes poderão aproveitar alguns métodos propostos nesse trabalho ou, ainda, como realizar um programa de melhoria dentro de um processo bem estruturado.

1.4 Organização deste documento

O restante deste trabalho está organizado na mesma seqüência de passos que o próprio programa de melhoria ocorreu. Em primeiro lugar, foi realizado um levantamento bibliográfico sobre os temas relacionados a esse trabalho, que está resumido no Capítulo 2.

No Capítulo 3 são apresentados o processo adotado para realizar o programa de melhoria, a organização alvo do programa e como se deu esse processo.

O passo seguinte após a escolha do processo para execução das melhorias e sua iniciação foi a realização de um diagnóstico, que consistiu em identificar a situação atual, definir a situação desejada e apresentar propostas que levassem de uma situação a outra. Um resumo do relatório desse diagnóstico é apresentado no Capítulo 4.

De posse das propostas de melhoria, cada uma foi executada dentro de um plano de ação que originou diversas modificações nos processos da organização. No Capítulo 5 é descrito como cada uma das ações ocorreu e é apresentando um resumo das técnicas avaliadas, sua aplicação e os resultados coletados nos projetos piloto.

Finalmente, no Capítulo 6, é realizado um registro dos benefícios alcançados e dos trabalhos futuros.

Capítulo 2

Trabalhos relacionados

Esse capítulo apresenta uma revisão bibliográfica dos diversos temas relacionados a este trabalho, incluindo publicações com contribuições relevantes que serviram de base para o seu desenvolvimento. Não é o objetivo aqui explicar com detalhes o conteúdo desses trabalhos, pois isso será feito nos capítulos subseqüentes onde eles foram usados. O restante desse capítulo está organizado da seguinte forma:

- A seção 2.1 trata de processos de melhoria, utilizados para aprimorar os processos nas organizações;
- A seção 2.2 apresenta um resumo e uma classificação de métodos de estimativa em projetos de desenvolvimento de software;
- A seção 2.3 lista os indicadores de acurácia mais usados para avaliar qualidade de estimativas, discutindo as suas aplicações;
- Na seção 2.4 é apresentada uma discussão sobre medidas de tamanho de produtos de software;
- Por fim, na seção 2.5, a calibração e validação dos métodos são discutidas.

2.1 Processos de melhoria

A melhoria de processos em organizações não é uma tarefa simples e por isso deve ser planejada e executada com cuidado. A mudança dos processos envolve saber onde estamos e onde desejamos chegar, para aí definir o “como” chegar. Toda a organização deve ser envolvida nesse

processo e por isso é fundamental o apoio da alta direção, para disponibilizar os recursos necessários e motivar toda a equipe. Além disso, a própria mudança deve ser feita dentro de um processo próprio, com fases e papéis bem definidos.

Dentre as especificações de processos de melhoria existentes, uma das mais conhecidas na indústria de software é modelo IDEAL [Gremba & Myers, 1997] do *Software Engineering Institute*[2] (SEI). O modelo fornece uma arquitetura de processo e orientações para que as organizações executem suas melhorias. Ele é constituído de 5 fases que dão nome ao modelo: Iniciação, Diagnóstico, Estabelecimento, Ação e Lições. A Figura 2-1 ilustra essas fases e suas etapas.

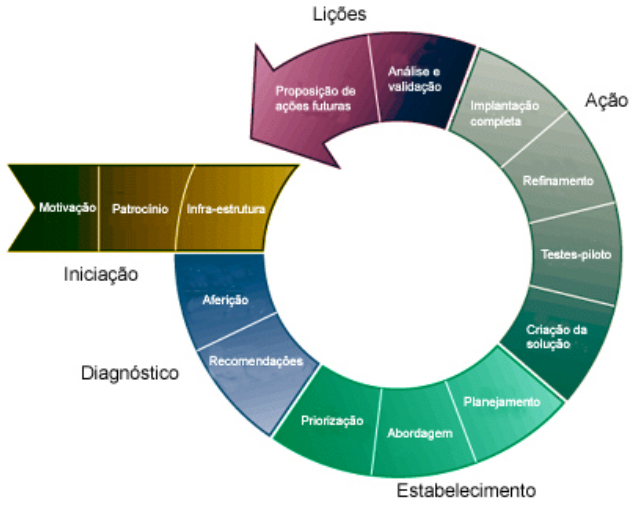

Figura 2-1 - As fases do modelo IDEAL (adaptado de [Gremba & Myers, 1997])

No entanto, o modelo IDEAL é um processo abstrato, definido apenas em alto nível. Ele não define, por exemplo, gabaritos e exemplos dos artefatos, como processos concretos o fazem. Cada organização deve adequar a arquitetura proposta dentro de suas necessidades. Os próprios autores do modelo sugerem que "atividades podem ser suprimidas ou executadas em paralelo dependendo dos recursos disponíveis na organização". O trabalho de Kautz [Kautz *et al.*, 2000] mostra a aplicação com sucesso do modelo após uma série de adaptações.

O INTRo (*IDEAL-Based New Technology Rollout*)[3] é um exemplo de processo concreto

[2] http://www.sei.cmu.edu/

[3] http://www.sei.cmu.edu/intro/

baseado no modelo IDEAL e também proposto pelo SEI, em parceria com a Computer Associates[4]. Ele apresenta um elevado nível de detalhamento, com artefatos, papéis e exemplos bem documentados. O foco desse processo é melhoria de tecnologias e ferramentas e não em processos especificamente.

Outra instância concreta do modelo IDEAL é o ProMOTe (Processo de Melhoria de Organizações Técnicas), desenvolvido pelo Synergia (ver Capítulo 3). Esse processo foi aplicado anteriormente para diagnóstico dos problemas de uma organização, como relatado no trabalho de Pádua e outros [Pádua *et al.*, 2003].

Também baseado no modelo IDEAL, o Praxis 3.0 propõe um processo subsidiário da disciplina de Engenharia de Processos que trata da Inovação Técnica em organizações [Pádua, 2008].

Um ponto importante de ressaltar é que modelos de maturidade de processos, como o CMMI [CMMI, 2006], muitas vezes são chamados de processos de melhoria. No entanto, esses modelos fornecem apenas o "o quê" deve ser feito e não o "como", como é o caso do modelo IDEAL e suas instâncias. Então, o CMMI pode ser usado como referência para as melhorias propostas na organização, mas não para o processo de execução das melhorias em si.

2.2 Métodos de estimativa

Os métodos de estimativa são alvo de pesquisas há algum tempo. Um levantamento bibliográfico concluído em 2004 e publicado em 2007, com foco em periódicos de língua inglesa cujo tema do trabalho era apenas de estimativa de esforço ou custo, identifica 304 estudos relevantes em 76 dos principais periódicos do mundo [Jorgensen & Shepperd, 2007]. Mesmo com centenas de trabalhos publicados com o objetivo de avaliar métodos de estimativa, parece não haver uma convergência quanto aos melhores métodos para cada situação específica. Como levantado por Myrtveit e outros [Myrtveit *et al.*, 2005], alguns resultados são contraditórios, onde um estudo mostra que o método A é melhor que B e outro mostra justamente o oposto. Os autores não afirmam o motivo da falta de convergência, mas especulam que poderia ser em função da falta de qualidade na execução dos experimentos, pela falta de proficiência em um método específico – o que levaria dois estudos sobre um mesmo método obterem resultados diferentes – e também pela falta de um

[4] http://www.ca.com/us/

indicador realmente confiável para comparar os métodos (esse último ponto será tratado na seção 2.3).

Mesmo não havendo uma classificação oficial para os tipos de métodos de estimativa, Myrtveit e outros [Myrtveit *et al.*, 2005] propuseram uma taxonomia para classificá-los. A divisão principal é feita entre métodos com dados esparsos (*Sparce-data methods*) e métodos com muitos dados (*Many-data methods*), ou seja, entre métodos que requerem poucos ou nenhum dado histórico e os que requerem muitos dados históricos. No primeiro grupo, encontram-se os baseados em julgamento de especialistas, como Wideband Delphi [Boehm, 1981] e RBC (Raciocínio Baseado em Casos) [Mukhopadhyay *et al.*, 1992]. No segundo grupo, onde dados históricos são necessários, encontram-se os métodos de aproximação de função arbitrária (AFA) e os baseados em funções matemáticas.

Os AFA's assumem que a relação entre as variáveis de entrada (ex: tamanho, equipe, tipo de produto) e as variáveis de resposta (ex: esforço, custo e prazo) não é conhecida e, portanto, possuem forma arbitrária. As Redes Neurais Artificiais [Aggarwal *et al.*, 2005] e Redes *Neuro-fuzzy* [Lopez-Martin *et al.*, 2006] enquadram-se nesse grupo. No grupo dos métodos baseados em funções matemáticas, estão os que assumem uma relação conhecida entre as variáveis de entrada e de resposta, tomando a forma de uma equação, como $y=Ax^B$, e são alvo de técnicas de análise de regressão. Representantes populares dessa categoria são o COCOMO [Boehm *et al.*, 2000] e o modelo e Putnam [Putnam, 1978] (atualmente comercializado como uma ferramenta denominada SLIM).

A classificação proposta acima não é precisa. Como os próprios autores alertam, há uma interseção entre alguns métodos. O RBC, por exemplo, pode ser considerado uma AFA. Além disso, há métodos que combinam com outros de mais de um grupo, como o trabalho de Ryder [Ryder, 1998], que combina a utilização do COCOMO com o uso de redes *neuro-fuzzy*, embasado pela tese de que a escolha de alguns parâmetros do modelo COCOMO é mais bem representada por conjuntos nebulosos [Zadeh *et al.*, 1977] que por faixas de valores.

Dentre os tipos de métodos citados acima, o estudo de Jørgensen e Shepperd [Jorgensen & Shepperd, 2007] mostra uma predominância do interesse em funções matemáticas e regressão (49%) dentre os 13 tipos identificados pela pesquisa. Vale notar que esse trabalho não classifica os tipos de métodos na ontologia proposta por Myrtveit e outros.

2.3 Indicadores de Acurácia

Antes de iniciar a discussão sobre indicadores de acurácia é fundamental definir o conceito de acurácia e de indicador. Acurácia, como definido em dicionário da língua portuguesa, trata da proximidade entre uma medida obtida experimentalmente e o seu valor real. No caso de estimativas, a acurácia mede a distância entre a estimativa de alguma grandeza (ex: estimativa de esforço total de um projeto) e o valor real (ex: total de esforço ao final do projeto). O termo indicador se refere ao sumário de várias medidas, nos mesmos moldes das definições de medida e de indicador do *Practical Software & System Measurement* (PSM) [McGarry *et al.*, 2001]. No caso de estimativas, um indicador é tipicamente calculado pela média ou mediana de várias medidas de acurácia.

Alguns trabalhos relatam que o *Mean Magnitude of Relative Error* (MMRE) é o indicador de acurácia mais utilizado para avaliação de métodos de estimativa, como nos trabalhos de Myrtveit e outros [Myrtveit *et al.*, 2005] e de Nguyen e outros [Nguyen *et al.*, 2008]. Mesmo sendo o indicador mais utilizado em trabalhos de comparação ou validação de métodos de estimativa, o próprio Myrtveit reconhece que o MMRE dá resultados incorretos dependendo da amostra. Outro trabalho que corrobora com o de Myrtveit é o de Foss e outros [Foss *et al.*, 2003], que utiliza simulações para mostrar que esse indicador leva a conclusões incorretas. Ambos os trabalhos sugerem o uso de outros indicadores de acurácia além do MMRE. Alguns desses indicadores e suas fórmulas de cálculo estão listados na Tabela 2-1, onde $\hat{\mathbf{y}}$ representa o valor estimado, $\mathbf{y}$ o valor real e n o número de medidas.

Dos indicadores listados na Tabela 2-1, o RSD é limitado a modelos onde há apenas uma variável de predição, o que não é o caso da maioria dos métodos de estimativa citados na seção anterior.

Outro indicador que aparece com muita frequência nos estudos de acurácia de métodos de estimativa é o PRED(x) [Chen *et al.*, 2005] [Clark *et al.*, 1998], que mostra a porcentagem de medidas cujo MRE é menor que x. Ele é apropriado para aplicação da definição de McConnell [McConnell, 2006] para avaliar se uma organização faz boas estimativas: errar em até 25% em 75% dos casos, ou seja, PRED(25%) ≥ 75%.

Tabela 2-1 - Indicadores de acurácia

Sigla do indicador	Nome indicador	Fórmula de cálculo
MMRE	Mean Magnitude of Relative Error	$MRE_i = \frac{\lvert y_i - \hat{y}_i \rvert}{y_i}$ $MMRE = \sum_{i=1}^{n} MRE_i$
MMER	Mean Magnitude of Error Relative of estimate	$MER_i = \frac{\lvert y_i - \hat{y}_i \rvert}{\hat{y}_i}$ $MMER = \sum_{i=1}^{n} MER_i$
MBRE	Mean of the Balanced Relative Error	$BRE_i = \begin{cases} \frac{(\hat{y}_i - y_i)}{y_i} \; se \; (\hat{y}_i - y_i) \geq 0 \\ \frac{(\hat{y}_i - y_i)}{\hat{y}_i} \; se \; (\hat{y}_i - y_i) < 0 \end{cases}$ $MBRE = \frac{1}{n} \sum_{i=1}^{n} BRE_i$
MIBRE	Mean of Inverted Balanced Relative Error	$IBRE_i = \begin{cases} \frac{(\hat{y}_i - y_i)}{y_i} \; se \; (\hat{y}_i - y_i) < 0 \\ \frac{(\hat{y}_i - y_i)}{\hat{y}_i} \; se \; (\hat{y}_i - y_i) \geq 0 \end{cases}$ $MIBRE = \frac{1}{n} \sum_{i=1}^{n} IBRE_i$
SD	Standard Deviation	$SD = \sqrt{\frac{\sum_{i=1}^{n} (y_i - \hat{y}_i)^2}{n-1}}$
RSD	Relative Standard Deviation	$RSD = \sqrt{\frac{\sum_{i=1}^{n} \left(\frac{y_i - \hat{y}_i}{x_i} \right)^2}{n-1}}$ onde x_i é o tamanho do projeto

Embora o MMRE e PRED sejam indicadores comumente usados para avaliar os métodos de estimativa, Kitchenham e outros [Kitchenham *et al.*, 2001] defendem que esses indicadores não

devem ser usados com o propósito de avaliar a acurácia de estimativas, mas sim para avaliar a adequação de um modelo aos pontos avaliados. A justificativa é que estimativas devem ser dadas como distribuição de probabilidade no lugar de um valor, visão essa, compartilhada por McConnell [McConnell, 2006].

Considerando o fato de no desenvolvimento de software ser importante a organização saber o sinal do erro da estimativa, ou seja, se ela foi subestimada ou superestimada, o Praxis 3.0 [Pádua, 2008] propõe uma medida de acurácia da estimativa de um projeto dada pela fórmula **(y - ŷ)/y**, semelhante ao MRE, mas sem a operação módulo.

Variantes de alguns indicadores apresentados na Tabela 2-1 utilizam a mediana em vez da média, como no trabalho de Mendes e Mosley [Mendes & Mosley, 2008]. Triola explica que a mediana é um indicador de centro preferencial para amostras muito espalhadas ou com muitos *outliers* [Triola, 2008].

2.4 Medidas de tamanho

O tamanho de um produto de software é uma informação crucial para se estimar esforço, prazo e custos do projeto de seu desenvolvimento. Todos os métodos de estimativa citados na seção 2.2 utilizam-no como entrada, mesmo que informalmente, como nos métodos baseados em julgamento de especialistas. Isso acontece, obviamente, por acreditar-se que há alguma correlação entre o tamanho do produto e o esforço para desenvolvê-lo.

Medidas de tamanho podem ser classificadas em dois tipos: funcional e físico [Pádua, 2008]. A primeira está relacionada com os requisitos funcionais do produto e a segunda com seu tamanho real após a sua construção.

Um bom retrospecto das medidas de tamanho funcional pode ser encontrado no trabalho de Gencel e Demirors [Gencel & Demirors, 2008], onde se destacam os pontos de função e suas variações e derivações. Outra medida encontrada na literatura são os Pontos de caso de uso [Arnold & Pedross, 1998], mas trata-se de um método ainda sem um padrão bem definido e adotado amplamente [Pádua, 2008].

Quanto às medidas de tamanho físico, a mais comumente utilizada é a linha de código. Ela pode ser expressa tanto em linhas físicas quanto lógicas. A primeira, como o próprio nome diz, conta as linhas físicas de um arquivo de código-fonte de um produto e a segunda a quantidade de

instruções executáveis desse mesmo arquivo. Não há um padrão único que defina a forma de contagem dessas linhas, mas um relatório técnico apresentado pelo SEI [Park *et al.*, 1992] detalha um *framework* para cada organização definir seu processo de contagem, que faz extenso uso de listas de conferência.

Embora pareça um contra-senso, no caso específico de linhas de código físicas, Rosenberg [Rosenberg, 1997] mostra que a formatação (ou padrão de codificação) é irrelevante para a contagem, contando que sejam feitas comparações que utilizem a mesma formatação.

2.5 Calibração e validação de métodos de estimativa

Os métodos de estimativa que utilizam muitos dados históricos, como os AFA's e os baseados em equações matemáticas, devem utilizar preferencialmente dados da organização onde um projeto está sendo estimado, aumentando assim a acurácia das estimativas [McConnell, 2006]. Uma possível justificativa para isso é a enorme diferença de produtividade entre organizações, mesmo quando lidamos com projetos similares. Analisando o banco de dados do *International Software Benchmarking Standards Group* (ISBSG), versão 10, pode-se ver uma variação muito grande da produtividade – expressa em pontos de função(PF)/Pessoa-mês(PM) – entre os projetos. A Figura 2-2, mostra o resumo estatístico da produtividade para projetos cujo tipo do produto é sistema financeiro, a linguagem é o Visual Basic e a qualidade da coleta dos dados foi declarada como Muito boa ou Boa. A escolha desses filtros foi feita de forma a obter um número razoável de projetos com dados confiáveis que tivessem características semelhantes, para que a análise tivesse alguma validade. Pode-se ver que mesmo em projetos semelhantes observamos uma variação grande da produtividade, onde a média da produtividade é 23,8 PF/PM e no primeiro e terceiro quartis temos produtividade de 4,3 e 30,7 PF/PM respectivamente.

Anda e outros [Anda *et al.*, 2009] realizaram um estudo com quatro empresas desenvolvedoras de software que executaram um projeto para construir o mesmo produto. Nos resultados apresentados pelas organizações pode-se notar não só uma grande variação nos resultados de esforço e prazo como também na qualidade. Curiosamente, a organização que ofereceu o projeto pelo menor valor foi a que incorreu nos maiores estouros de custo e prazo, além de apresentar a pior qualidade.

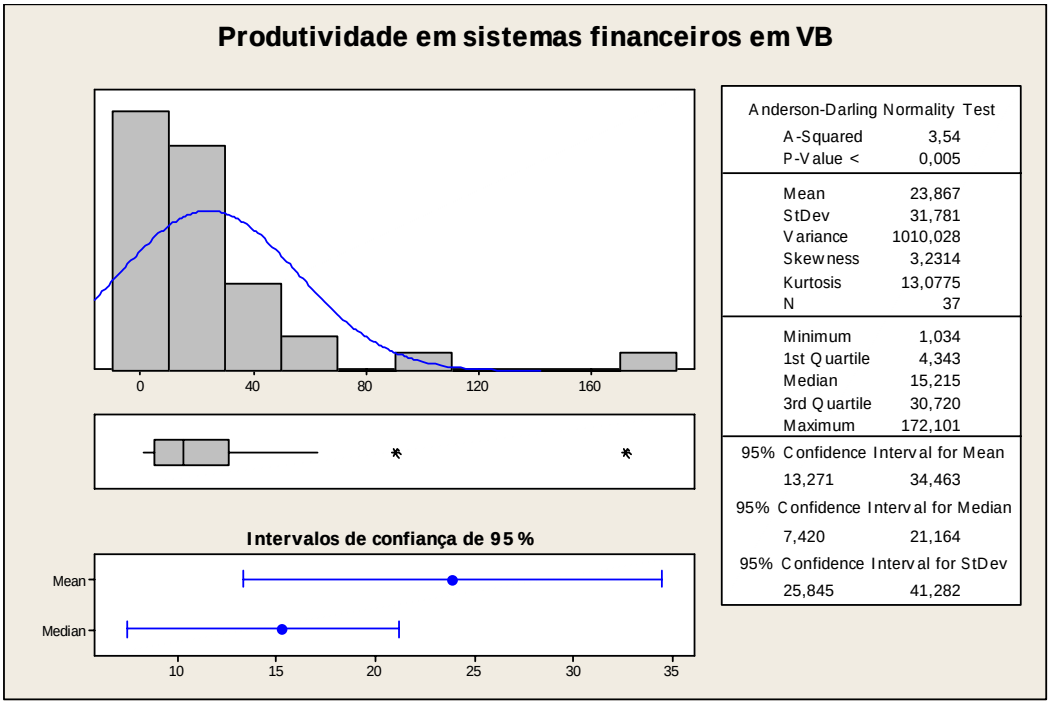

Figura 2-2 – Produtividade de projetos de sistemas financeiros escritos em Visual Basic (ISBSG)

Outros relatos de grande variação de produtividade podem ser encontrados nos trabalhos de Pádua [Pádua, 2007] e [Pádua, 2009], que mostra resultados em projetos executados em cursos de Especialização que utilizam mesma tecnologia e processo.

A utilização de dados históricos nos métodos permite calibrá-los, ajustando seus parâmetros para melhor representar os projetos da organização. Relatos da calibração de modelos matemáticos podem ser encontrados em [Clark *et al.*, 1998] e [Barry *et al.*, 2002].

A calibração tipicamente envolve a utilização de técnicas de regressão para ajustar o modelo. No entanto, alguns autores vão além, propondo formas diferentes de escolher os dados históricos a serem utilizados, como Chen e outros [Chen *et al.*, 2005] que combinam a escolha dos projetos (estratificação) e de apenas alguns parâmetros do COCOMO mostrando resultados melhores do que a calibração convencional.

Além da calibração com dados da própria organização, deve-se validar o modelo calibrado para ver seu grau de acurácia ao realizar estimativas. Entre os métodos de validação, podemos citar:

- ***Holdout***: separa-se o conjunto de dados históricos em 2 conjuntos disjuntos. O primeiro é usado para calibrar o modelo e o segundo é usado para avaliar os erros do modelo calibrado;

- ***K-fold cross-validaton***: também conhecido como *v-fold cross-validation*, os dados históricos são divididos em k conjuntos de tamanho n. Um dos conjuntos é retido para validação e os demais k-1 são utilizados na calibração. O procedimento é repetido k vezes, escolhendo um conjunto diferente a cada iteração, para se computar a média dos erros, utilizando um dos indicadores da seção 2.3;
- ***Leave-one-out cross validation***: também conhecido como *n-fold*, é uma variante do *k-fold* onde os k conjuntos são de tamanho 1, ou seja a cada iteração, apenas 1 dado da amostra fica de fora do conjunto de calibração.

Um estudo detalhado desses métodos pode ser obtido em [Kohavi, 1995].

Pelo fato de que o método *Leave-one-out cross validation* se parece mais com a situação real nas organizações, quando temos um conjunto de dados históricos e queremos estimar o próximo projeto, Myrtveit e outros [Myrtveit *et al.*, 2005] defendem seu uso na avaliação da calibração. Acreditamos também que esse é o melhor método pelo fato dele exercitar todas as combinações de conjuntos de calibração e de validação possíveis, ao passo que nos outros métodos, dois ou mais projetos podem nunca ficar separados entre esses os conjuntos de calibração e validação. Para exemplificar, suponha a utilização do *k-fold* em um conjunto de 6 projetos (nomeados de A a F), com conjuntos de tamanho 2 (k=3, n=2). Os conjuntos seriam (AB) (CD) (EF). Assim, os projetos A e B estariam sempre juntos, ou no conjunto de calibração ou no conjunto de validação, mas nunca separados entre os dois, o que poderia causar alguma distorção na avaliação da calibração.

Capítulo 3

O Programa de melhoria de estimativas

Esse trabalho optou pela utilização do ProMOTe para melhoria dos processos de estimativa da Organização alvo desse programa. O motivo é a familiaridade da equipe com ele e a sua aplicação anterior com sucesso em outros projetos. Esse capítulo apresenta sua estrutura, a organização na qual ele foi utilizado e as personalizações na sua aplicação.

- A seção 3.1 descreve o ProMOTe, suas fases, papéis e artefatos;
- A seção 3.2 apresenta o Synergia, organização alvo desse programa de melhoria de estimativas;
- A seção 3.3 detalha como ocorreu a execução do ProMOTe dentro do Synergia, mostrando os papéis (incluindo o do autor deste trabalho) e responsabilidades, a duração, entre outros detalhes.

3.1 O ProMOTe

O ProMOTe [Pádua *et al.*, 2003], Processo para Melhoria de Organizações Técnicas, é, como o próprio nome sugere, um processo de melhoria de processos. O ProMOTe foi elaborado no Synergia (ver seção 3.2) e é baseado no modelo IDEAL, sendo constituído nas mesmas 5 fases e define alguns papéis a serem desempenhados dentro do programa de melhoria. Na Figura 3-1 há um organograma desses papéis.

O Comitê Executivo é o grupo formado pelas pessoas de maior poder de decisão, representa o nível hierárquico e estratégico mais alto da organização. É responsável pelas decisões

estratégicas da organização.

O Grupo Diretor é o responsável pela coordenação das atividades do programa de melhoria, pela gestão da infra-estrutura que irá facilitar e permitir a sua execução, pela manutenção da motivação e entusiasmo de todos os níveis da organização e pela definição das informações que devem ser divulgadas sobre o programa. Além disso, detalha as metas e objetivos do programa; direciona e prioriza as atividades; fornece diretrizes e ações corretivas, quando necessário, para manter o programa em sintonia com a missão e visão da organização. Os artefatos produzidos durante o programa de melhoria devem ser aprovados pelo Grupo Diretor.

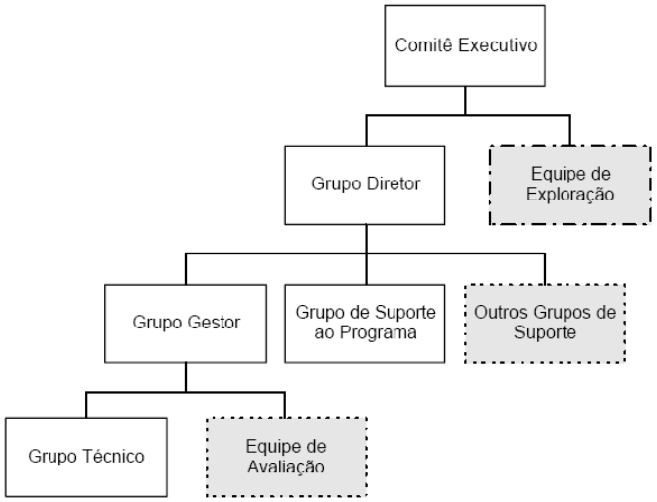

Figura 3-1 - Organograma dos papéis definidos no ProMOTe, extraído de seu manual.

A responsabilidade pelo planejamento e controle do programa é do Grupo Gestor. Também é responsável pela criação dos Grupos Técnicos, além de dar suporte e aprovar o trabalho destes grupos. Outra responsabilidade desse grupo é a criação, caso necessário, de equipes de avaliação para caracterizar o estado atual de determinados aspectos da organização.

O Grupo de Suporte ao Programa mantém o apoio ao programa de melhoria em um ambiente de mudanças, norteando e sustentando as atividades de melhoria individuais. Esse grupo não é o responsável pela implantação da melhoria e sim um facilitador das atividades, ajudando em qualquer dificuldade encontrada no projeto.

Os Grupos Técnicos são os responsáveis pelo desenvolvimento das soluções para o programa. Para isso, seus membros devem ser pessoas que integram as equipes de trabalho que utilizam, rotineiramente, o objeto alvo da melhoria, tendo bastante conhecimento sobre os pontos

fortes a serem mantidos e os pontos fracos a serem melhorados.

A Equipe de Exploração é um grupo transitório - só existe durante as primeiras atividades do programa. Uma vez que a execução da fase de Início tenha sido aprovada, os demais grupos do programa (Diretor, Gestor e Suporte) devem ser formados e a Equipe de Exploração é imediatamente desfeita.

Os Outros Grupos de Suporte e a Equipe de Avaliação são grupos temporários e opcionais durante o programa. A Equipe de Avaliação é criada durante o Diagnóstico, caso o Grupo Gestor detecte a necessidade de formar grupos específicos para diagnosticar a situação atual de determinados aspectos.

A Tabela 3-1 abaixo descreve de forma resumida as atividades executadas em cada uma das 5 fases do processo, a responsabilidade de cada papel definido acima e os artefatos produzidos.

Tabela 3-1 - Descrição das fases do ProMOTe, extraído de seu manual.

Fase	Descrição
Início	A fase de Início deixa claras as necessidades da organização, os benefícios e o escopo do programa, além de estabelecer a infra-estrutura necessária para continuidade do programa. Nele são recrutadas as pessoas que irão compor o Grupo Gestor e o Grupo Diretor, definindo seus papéis e responsabilidades. Nessa fase, devem ser definidas as metas da organização, em termos de prazos e custos para o programa de melhoria, sendo feito um planejamento em alto nível das próximas fases. As atividades da fase de Início são críticas e, se forem bem executadas, podem garantir o prosseguimento das atividades subseqüentes com menos dificuldades. Os principais artefatos confeccionados na fase de Início são a Descrição do Programa (DP) e o Plano do Programa (PP). No fim dessa fase, a continuidade do programa de melhoria deve ser aprovada pelo Comitê Executivo. Caso contrário, o programa será cancelado.
Diagnóstico	O objetivo da fase de Diagnóstico é obter um entendimento mais detalhado do programa de melhoria. Para isso, deve ser realizada uma caracterização do estado atual e desejado da organização, através de atividades de avaliação. Com base nessas avaliações, deve ser proposto um conjunto de recomendações para que o estado desejado seja atingido. Os resultados das avaliações, bem como as recomendações derivadas, são compatibilizados com os esforços de melhoria existentes ou planejados. A definição do estado atual, do estado desejado e a lista de recomendações compõem o Relatório do Diagnóstico (RD).

Estabelecimento	Durante o Estabelecimento, as questões que a organização decidiu focar no programa de melhoria são priorizadas e as estratégias para alcançar as soluções são formuladas. Nessa fase, o Plano Estratégico de Ação (PEA) será produzido. Esse plano irá guiar os trabalhos durante a fase de Ação. Os projetos necessários para o desenvolvimento das soluções são identificados e priorizados. Essa priorização deve considerar as prioridades das recomendações propostas na fase de Diagnóstico que são atendidas pelos projetos e também as prioridades gerenciais da organização. Além disso, os Grupos Técnicos responsáveis por cada um dos projetos devem ser selecionados e treinados. Para cada projeto identificado cria-se um documento chamado Plano Tático de Ação (PTA) que, em um primeiro momento, mantém a descrição do projeto e a composição de seu Grupo Técnico. Além disso, nessa fase deve ser criado um plano de execução dos projetos, demonstrando a possibilidade de paralelismo existente e as dependências identificadas. Uma agenda para o programa, com detalhes dos períodos dedicados a cada projeto é criada. Sempre que o Grupo Gestor considerar necessário, o programa de melhoria pode ser replanejado.
Ação	Durante a fase de Ação as soluções necessárias são propostas, produzidas e colocadas em produção. As atividades dessa fase tipicamente consomem mais tempo e recursos que as atividades de todas as outras fases. Projetos-piloto são definidos e executados com o objetivo de testar e avaliar as soluções que foram propostas. Após a sua execução, as soluções devem ser aperfeiçoadas para refletir o que foi aprendido. Após a validação da solução, ela será implantada na organização.
Lições	O principal objetivo dessa fase é fazer com que uma próxima execução do programa de melhoria, caso ocorra, seja mais efetiva. Nesse momento, as soluções já foram desenvolvidas e lições foram aprendidas. Essas informações são coletadas e registradas de forma a garantir que elas sejam fonte de informação futura. De posse das informações coletadas, serão feitas avaliações da estratégia, dos métodos e da infra-estrutura utilizados no programa de melhoria. Se ações futuras forem propostas, correções e ajustes na estratégia, métodos ou infra-estrutura devem ser sugeridos e o próximo ciclo deve ser definido e planejado.

3.2 O Synergia

O Synergia, organização alvo desse programa de melhoria, é um laboratório de pesquisas em Engenharia de Software e Sistemas do Departamento de Ciência da Computação (DCC) da UFMG, que oferece serviços de desenvolvimento de software para empresas públicas e privadas. O surgimento do laboratório tem sua origem em um convênio entre o DCC e uma das maiores empresas de telefonia do Brasil assinado na década de 90 para desenvolvimento de um sistema de grande porte para essa empresa. Diante da complexidade desse desafio, surgiu a necessidade de criação de um

processo bem definido para execução do projeto, e com essa necessidade surgiu o processo PROSE. Mais tarde, com a experiência do PROSE e pesquisas realizadas pelo professor do DCC e um dos coordenadores do laboratório, Wilson de Pádua, foi desenvolvido o Praxis (PRocesso para Aplicativos eXtensíveis InterativoS), baseado em processos como MBase, desenvolvido pela equipe de Barry Boehm na USC, e o UP (*Unified Process*). O Praxis atualmente se encontra em sua terceira versão [Pádua, 2008].

Atualmente, o laboratório conta com cerca de 80 colaboradores entre doutores, mestres e vários especialistas que em sua maioria têm formação em Ciência da Computação.

O principal serviço prestado pelo Synergia é o desenvolvimento de produtos de software. O laboratório já desenvolveu produtos de 300 até 5000 pontos de função. Além do desenvolvimento de produtos de software, também são oferecidos os serviços:

- Modelagem de negócio: fornecendo uma documentação abrangente dos processos de negócio executados em um cliente;
- Especificação de sistema: levantando os requisitos e elaborando o projeto conceitual de uma família de produtos de software que visam automatizar os processos do cliente;
- Treinamento e consultoria: em processos, melhoria de processos e técnicas de desenvolvimento de software.

O Synergia adota uma versão personalizada do Praxis, denominada Praxis-Synergia, adequada às necessidades encontradas durante a execução de projetos reais. A proximidade do autor do processo com o laboratório cria um ambiente de troca de experiências entre as pesquisas realizadas no ambiente acadêmico e a execução de projetos reais, fazendo com que ambos os processos evoluam em paralelo. Mais detalhes do Praxis-Synergia serão apresentados na próxima seção.

Até o fim de 2008, a versão do Praxis-Synergia era baseada na versão 2.1 do Praxis padrão, que possuía ciclo de vida de entrega evolutiva. Todos os projetos da Organização utilizados para coleta e análise de dados nesse trabalho, utilizaram esse tipo de ciclo de vida da versão 2.1 do Praxis. Entretanto, por imposição do mercado, esses projetos foram todos divididos em dois: **Projeto de Especificação**, que inclui o levantamento e análise dos requisitos e **Projeto de Construção**, que inclui as atividades de desenho em diante. O **Projeto de construção** não deve ser confundido com a **Fase de construção** do Praxis. A Figura 3-2 ilustra as atividades contempladas em cada um dos projetos.

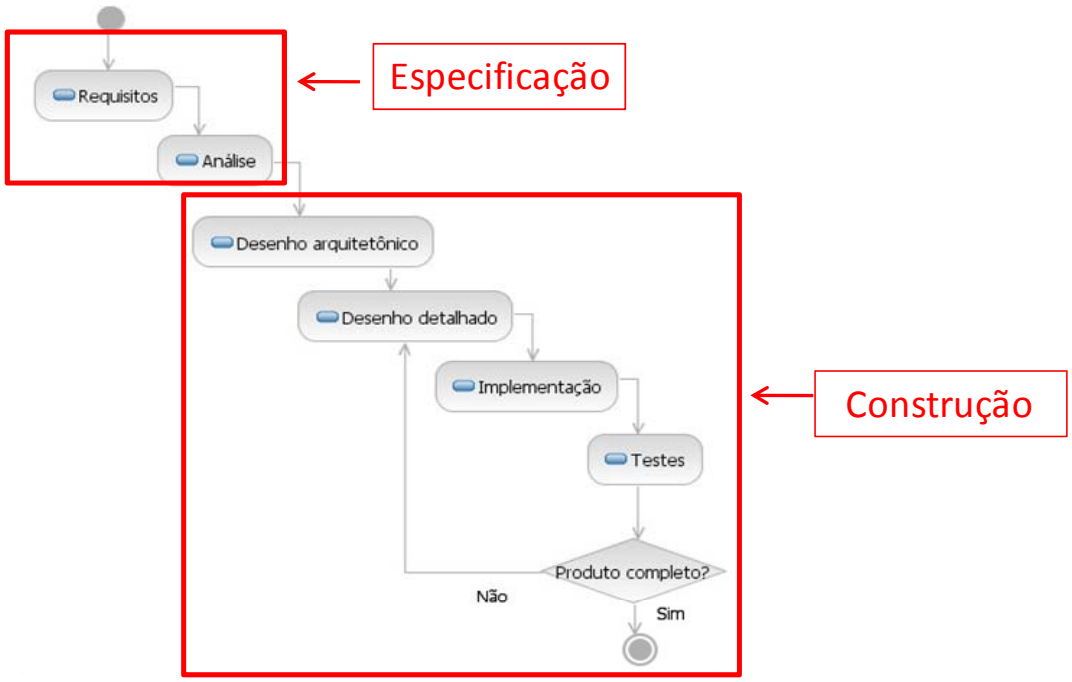

Figura 3-2 - Ciclo de vida de entrega evolutiva do Praxis 2.1.

No fim de 2008, houve o lançamento da terceira versão do Praxis [Pádua, 2008] e a Organização prontamente iniciou o processo de atualização de sua versão personalizada para se adequar aos novos conceitos do Praxis 3.0. Essa nova versão, que traz mudanças significativas em relação à anterior, possui ciclo de vida quase-espiral, onde, a cada iteração das fases de Elaboração e Construção, uma versão parcial do produto é completada. Atualmente (Maio de 2009), um grande projeto está sendo desenvolvido utilizando esta nova versão e diversas ações propostas neste programa de melhoria encontram-se em utilização nesse projeto. Voltaremos a essas questões de tipos de ciclo de vida na seção 5.1.

Outras informações sobre o laboratório podem ser encontradas no trabalho de Pimentel e outros [Pimentel *et al.*, 2006].

3.3 A aplicação do ProMOTe no Synergia

Nessa seção serão apresentados os detalhes de como ocorreu a aplicação do ProMOTe no Synergia. Como já foi mencionado anteriormente, o autor deste trabalho é membro da equipe da Organização, mais especificamente, membro da equipe de Engenharia de Processos. Boa parte do

esforço do autor nesse programa fez parte do escopo de suas atividades como funcionário da Organização, principalmente as atividades relacionadas com a execução das melhorias propostas. A parcela de dedicação como aluno de pós-graduação foi concentrado na confecção do Relatório de Diagnóstico e na elaboração deste texto.

O primeiro passo seguido foi a definição dos papéis. Por se tratar de um programa de porte relativamente pequeno, em uma Organização também de pequeno porte, alguns dos papéis do ProMOTe foram agrupados. O Comitê Executivo foi composto pela Diretoria da Organização e se responsabilizou pela definição dos objetivos do programa e da aprovação e priorização de todas as ações propostas. O Grupo Diretor era representado pelo gerente da equipe de Engenharia de Processos e agiu como um facilitador, dentro da Organização para a execução do programa. O Grupo de Suporte ao Programa consistia de representantes da equipe de Administração de Recursos Computacionais, fornecendo equipamentos e ferramentas que seriam utilizados. Os demais grupos permanentes, Grupo Técnico e Grupo Gestor, foram assumidos pelo autor do trabalho. Por último, o Grupo de Exploração, responsável pela confecção do Relatório de Diagnóstico foi composto pelos membros da equipe de Engenharia de Processos e pelos Gerentes de Projetos da Organização.

A seguir, dados os objetivos gerais do programa, estabelecidos pelo Comitê Executivo, o Grupo Diretor e Grupo de Exploração elaboraram os documentos de Definição do Programa e Plano do Programa. Essa atividade durou 2 semanas. Na seqüência os documentos foram apresentados ao Comitê Executivo para aprovação e revisão.

Aprovados esses documentos, a Equipe de Exploração iniciou o diagnóstico da situação atual e da situação desejada. Essa atividade foi realizada, principalmente, através de reuniões internas na Organização. O resultado desse trabalho, que durou aproximadamente 2 meses, é o Relatório de Diagnóstico, apresentado em detalhes, no Capítulo 4.

Definidas as ações para levar à situação desejada, cada uma delas tornou-se um pequeno projeto que foi executado, essencialmente, pela equipe de Engenharia de Processos. Para cada projeto foram definidos os objetivos, tecnologias que seriam avaliadas, além de orçamento (esforço dos colaboradores da Organização) e metas de prazo. Em aproximadamente 16 meses esses projetos foram concluídos e seus detalhes e resultados alcançados são apresentados no Capítulo 5.

A fase de Lições, última fase do ciclo de melhorias, não foi executada na Organização por limites de prazo imposto para confecção deste texto, mas já foi possível coletar informações valiosas durante a execução das ações que permitirão tornar mais eficientes novas instâncias de programas como este.

Capítulo 4

Relatório de diagnóstico

Nesse capítulo será apresentado um resumo do Relatório de Diagnóstico (RD) confeccionado durante o Programa de melhoria de estimativas. O RD do ProMOTe tem o objetivo de levantar o estado atual e desejado da organização em relação ao objetivo do programa de melhoria, no caso, melhoria de estimativas. O relatório mostra também quais ações devem ser realizadas para se alcançar o estado desejado a partir do atual, levando em consideração os pontos fortes e fracos da organização.

As seções a seguir detalham o RD e estão organizadas da seguinte forma:

- A seção 4.1 enumera os aspectos contemplados no diagnóstico;
- Na seção 4.2, a situação atual da organização em relação a esses aspectos é apresentada;
- Na seção 4.3, a situação desejada em relação aos mesmos aspectos é descrita;
- Por fim, na seção 4.4, as ações propostas, para levar da situação atual para a desejada, são enumeradas.

4.1 Aspectos contemplados no diagnóstico

O Comitê Executivo do programa definiu duas necessidades importantes para o diagnóstico do programa de melhoria de estimativas da Organização:

- Ter estimativas de esforço, prazo e custo confiáveis para elaboração de propostas

comerciais de projetos de desenvolvimento de software.

- Conseguir elaborar planejamento mais confiável dos projetos com menor variação entre planejado e realizado, evitando replanejamentos do projeto. Esse planejamento envolve tanto o planejamento macro (esforço, custo e prazo totais das iterações) quanto o micro-gerenciamento (atribuição de tarefas aos colaboradores).

Diante dessas necessidades, foram derivados os aspectos a serem considerados no diagnóstico. A Tabela 4-1 apresenta esses aspectos.

Tabela 4-1 - Aspectos considerados no diagnóstico

Necessidade	Aspectos Derivados
1	Processo para as estimativas e para o registro de informações de propostas e orçamentos
2	Registro e análise de informações de execução dos projetos. Processo de planejamento macro (iterações) dos projetos. Processo de estimativa para tarefas individuais nos projetos.

Esses quatro aspectos foram avaliados pela Equipe de Exploração do programa, composta por membros da Equipe de Engenharia de Processos da Organização, que, por meio de reuniões e entrevistas com colaboradores internos, fez o levantamento da situação atual (apresentada na próxima seção). De posse do relatório da situação atual, essa equipe se reuniu novamente com o Comitê Executivo e chegou-se a uma proposta de situação desejada, também para cada um dos 5 aspectos identificados na Tabela 4-1.

4.2 Situação anterior

Nessa seção faremos um resumo do diagnóstico da situação da Organização com relação aos aspectos discutidos na seção anterior na época em que foi realizado o diagnóstico.

4.2.1 Aspecto Processo e registro de informações de propostas e orçamentos

Na Organização, o orçamento era feito sem processo definido. Usualmente utilizava-se o

Microsoft Project para elaborar a estrutura analítica do projeto, dividindo-o em iterações. Para se estimar o esforço das iterações era utilizado um cálculo simples com base na produtividade dos outros projetos já executados e no tamanho estimado (ou medido) do produto que seria executado em cada iteração. O prazo era derivado da equipe que se imaginava estar disponível. A elaboração desse orçamento dependia exclusivamente da experiência do profissional que o estava fazendo.

As propostas são separadas em técnicas e comerciais. A primeira possui a descrição do escopo com detalhes e um resumo do orçamento de esforço e prazo. A segunda, que é baseada na proposta técnica, contém o preço de venda do projeto, formado sobre o perfil da equipe imaginada para a sua execução. Ambas eram armazenadas apenas como documentos, o que dificultava a coleta de dados.

Um levantamento realizado com dados de projetos passados mostrou um erro médio (MMRE) na estimativa do esforço de 45,9% total dos projetos.

4.2.2 Aspecto Registro e análise de informações de execução de projetos

O registro de informações de tamanho, esforço, prazo, qualidade e custo de execução dos projetos são essenciais para embasar estimativas para novos projetos. Um ponto forte observado na Organização, é que os esforços são armazenados em um sistema de ponto eletrônico, chamado Apropriação de Horas (AH) o que dá uma confiabilidade muito alta dessa informação. Ao iniciar suas atividades o colaborador dispara um comando que começa a marcar o tempo de trabalho e a cada vez que está saindo do local de trabalho, informa em quais tarefas de quais projetos trabalhou desde que iniciou o sistema. Essas tarefas armazenam informações importantes, como o requisito associado, a disciplina (ou fluxo) a que se refere e a iteração em que foi executada.

Não havia registro formal de prazo dos projetos. No entanto, ele podia ser inferido a partir dos registros do AH, utilizando a menor e maior data registradas, mas era preciso alguma análise manual para excluir atividades que ocorreram antes e depois do prazo real de execução (exemplo: algumas tarefas de pequeno esforço realizadas bem após o término do projeto, para organizar documentação para envio ao cliente são apropriadas no projeto, distorcendo o prazo).

O custo podia ser obtido também a partir do AH, mas é um trabalho manual e moroso, já que envolve buscar o valor-hora de um colaborador na época da apropriação e multiplicar pelo esforço deste colaborador no período desejado. Todos os valores do custo por hora dos colaboradores eram armazenados em planilhas eletrônicas e de acesso restrito.

O tamanho dos produtos é medido em pontos de função não ajustados utilizando o padrão do *International Function Point User Group* (IFPUG). Ao final da especificação dos requisitos dos projetos, era feita uma contagem dos pontos de função do produto e registrado em planilhas eletrônicas. Durante a construção do produto, esses registros não eram mantidos. Os pontos de função de alteração não eram integrados à última contagem de tamanho. Isto impossibilitou determinar de forma direta o tamanho do produto final, necessitando nova contagem, caso a Organização ache necessário recuperar essa informação.

As informações de qualidade (defeitos) ficam armazenadas em um sistema informatizado. A coleta dos dados é parcialmente automatizada, o que permite evitar os erros decorrentes de consolidação manual das informações.

4.2.3 Aspecto Processo de planejamento macro (iterações) dos projetos

Assim que o projeto iniciava, o seu gerente elaborava um planejamento macro de suas iterações. Para cada iteração era elaborada a estrutura analítica do projeto, com todas as tarefas e perfis de recurso necessário e uma estimava do esforço para executá-las. Com isso era possível saber o esforço e duração total de cada iteração. Para se estimar o esforço, não havia um procedimento definido. O gerente usualmente utilizava planilhas auxiliares que fornecem um esforço padrão para cada tipo de tarefa baseado numa classificação da complexidade do escopo. Essa classificação era feita por um colaborador experiente. Mais detalhes desse procedimento serão apresentados na seção 5.3.

Ao fim do planejamento macro, o gerente confrontava o esforço total com o orçamento e caso o ultrapasse, ele procurava a diretoria da Organização e o cliente para negociações.

4.2.4 Aspecto Processo de estimativa para tarefas individuais nos projetos

No início de cada iteração, o planejamento detalhado era realizado pelo gerente do projeto, alocando o recurso (colaborador) que executaria cada uma das tarefas. Nesse momento o esforço planejado durante o planejamento macro era revisado. Em alguns casos o gerente arbitrava o esforço que concederia ao colaborador. Esse esforço era definido sem um procedimento definido. Em alguns casos consultavam-se dados de execução de outros projetos, em outros apenas o sentimento e experiência do gerente eram usados. As tarefas não planejadas, que surgiam no decorrer do projeto,

também eram estimadas das duas formas sem um critério definido.

Em entrevista com os gerentes de projeto foi possível perceber que havia um sentimento de que os procedimentos usados não levavam a boas estimativas, principalmente para tarefas relacionadas à codificação de casos de uso. Essas tarefas são de esforço grande: é comum terem esforço maior que 200h, como relatado por eles. Erros grandes nessas tarefas ocasionam também em grande impacto no projeto. Um levantamento realizado em 2 módulos (num total de 39 casos de uso) de um dos projetos executado na Organização, mostra um esforço médio de 100,2h para codificação de cada um dos casos de uso. Analisando as estimativas feitas para as 39 tarefas nos projetos, observou-se um erro médio (MMRE) relativamente alto: 52,8% ± 20,1%, confirmando as impressões dos gerentes. Esse levantamento foi feito apenas no projeto cujas informações foram armazenadas de forma a ser possível a coleta automatizada.

4.3 Situação desejada

Nessa seção será apresentado o resumo da situação desejada pela Organização com relação aos aspectos discutidos na seção 4.1.

4.3.1 Aspecto Processo e registro de informações de propostas e orçamentos

Em relação ao aspecto Processo e registro de informações de propostas e orçamentos, a Organização deseja ter definido um processo que faça uso de seus dados históricos e que se baseie em análise dos riscos envolvidos. Esse processo deve utilizar de métodos consagrados como o COCOMO II.

Também deseja-se criar um repositório centralizado de orçamento e propostas, armazenando não só os arquivos enviados para clientes (e suas várias versões), mas as informações de custo, tamanho, esforços, prazos e riscos de forma estruturada. Espera-se com isso ser mais fácil a comparação dos orçamentos com os valores reais de execução dos projetos.

4.3.2 Aspecto Registro e análise de informações de execução de

projetos

Nesse aspecto, a Organização quer ter um processo de fechamento do projeto que inclua o registro de esforço, prazo, custo, tamanho, qualidade e personalizações de processo utilizadas, armazenando em um repositório unificado e estruturado, permitindo análises futuras e comparações entre os projetos. Algumas medidas/informações a serem coletadas:

- Tamanho em PF e a evolução dessa contagem.
- Tamanho em LOC (separando aplicação, testes de unidade, scripts de testes, camada de reuso).
- Riscos identificados e concretizados.

Para a coleta da contagem dos pontos de função é necessário ter um processo de atualização da contagem para todas as alterações de escopo do produto, deixando registrados os PF incluídos, excluídos e alterados em cada solicitação de alteração do cliente ou da equipe interna.

Também deve-se possuir uma classificação dos projetos quanto às suas características: tamanho, complexidade, tecnologias, requisitos não-funcionais, equipe e outros, permitindo a comparação entre projetos e possibilitando um orçamento com critérios mais consistentes. Para essa caracterização pode-se utilizar como base os parâmetros de classificação do COCOMO II ou parâmetros de outros métodos que venham a ser adotados.

Outro ponto importante desse aspecto é melhorar o acompanhamento atual dos projetos, fornecendo à diretoria e aos gerentes informações de melhor qualidade para tomada de decisão. Algumas informações solicitadas que não são produzidas e acompanhadas atualmente e que devem integrar a gestão dos projetos:

- Escopo planejado *versus* adquirido ao longo do tempo.
- Evolução do tamanho físico do código fonte, do código dos testes de desenvolvimento e do código dos *scripts* de teste automatizados, comparado ao progresso em PF. Essas medidas darão uma indicação do grau de reutilização de código.
- Evolução do custo por PF.
- Comparação do orçamento de custo do projeto com o custo real.
- Evolução do número de defeitos por PF ao longo do tempo.

4.3.3 Aspecto Processo de planejamento macro (iterações) dos projetos

Atualmente o planejamento macro dos projetos é feito, principalmente, com base em tabelas de esforços padrão para cada atividade por complexidade do requisito. Essa complexidade não é definida de forma objetiva. Outro ponto levantado é que as atividades que não estão ligadas diretamente aos casos de uso são estimadas por pura intuição.

A Organização deseja definir um processo de planejamento macro que seja mais racional e que produza o básico: qual o escopo, prazo e perfil de equipe necessária para cada iteração do projeto.

4.3.4 Aspecto Processo de estimativa para tarefas individuais nos projetos

Deseja-se formalizar um processo para estimativas de tarefas individuais do projeto, preferencialmente feitas pelos próprios colaboradores que as executarão no projeto. Essa estimativa ocorre no início de cada iteração, quando o gerente faz o planejamento detalhado de todas as suas tarefas.

Para dar suporte a essas estimativas é necessário que o cadastro de requisitos permita rastrear o impacto de alterações nos artefatos produzidos, facilitando a estimativa de esforços para alterações.

O Gerente deve ter à sua disposição esforços padrões de cada atividade do processo normalizados por pontos de função ou outra medida, dependendo do tipo da atividade. Esse repositório facilitará as estimativas dos colaboradores inexperientes e servirá como balizador quando houver divergências entre a opinião do Gerente e do colaborador que fizer a estimativa para a tarefa.

4.4 Ações propostas

De posse do levantamento da situação atual e desejada em relação aos aspectos levantados nas seções anteriores desse capítulo, a Equipe de Exploração, juntamente com o Comitê Executivo, propuseram uma série de ações e definiram suas prioridades. Essas ações estão listadas na

Tabela 4-2 com suas respectivas prioridades, onde prioridade 1 é maior.

Cada uma dessas ações se transformou em um Projeto Tático de Ação (PTA) e seus resultados serão apresentados nas seções do capítulo seguinte.

Tabela 4-2 – Ações propostas no Relatório de Diagnóstico

No. de ordem	Descrição	Prioridade
1	Definir um processo de estimativa para orçamentos	1
2	Aperfeiçoar o processo de estimativa para o planejamento das iterações	2
3	Definir o processo de contagem de pontos de função de alterações e integrar ao processo de controle de mudanças existente	3
4	Definir um processo de estimativa para o planejamento de tarefas, revisões e alterações nos projetos	3
5	Criar repositório centralizado de registro de dados de propostas, projetos e dados sumarizados de casos de uso desenvolvidos no projeto	3
6	Definir uma categorização dos recursos e criar um repositório centralizado de seus custos	3
7	Aperfeiçoar o Cadastro de Requisitos	3

Capítulo 5

Desenvolvimento das ações

Nesse capítulo os resultados das ações propostas pelo Programa de Melhoria de Estimativas serão apresentados. Cada seção desse capítulo trata de uma ou mais ações da Tabela 4-2.

O restante desse capítulo está organizado da seguinte forma:

- A seção 5.1 apresenta o método escolhido para estimativas realizadas durante a elaboração das propostas, juntamente com a sua validação e adaptações para adoção na Organização;
- Na seção 5.2, é apresentada uma proposta para contagem de Pontos de função diretamente nos modelos UML, garantindo consistência da contagem e de sua atualização. Também é endereçada a questão de contagem de Pontos de função de alteração;
- Na seção 5.3, o planejamento macro dos projetos é tratado, onde apresentamos uma simplificação do processo atualmente adotado na Organização;
- Já na seção 5.4, é abordado o planejamento detalhado dos projetos, apresentando a técnica escolhida e os resultados obtidos;
- Por fim, na seção 5.5, é apresentado o repositório centralizado de medidas proposto para a Organização.

5.1 O processo para estimativa de orçamentos

Como relatado na seção 4.2.1, a Organização apresenta um erro relativamente alto – MMRE de 45,9% – quando se compara o orçamento de esforço utilizado para suas propostas e os resultados dos projetos quando concluídos. Embora seja um valor alto para desvios nos projetos, há de se levar em consideração que boa parte desses projetos sofreu alteração de escopo durante sua execução, o que certamente impactou o esforço e prazo total desses projetos. Infelizmente, as alterações e as renegociações decorrentes delas não foram registradas de forma estruturada, inviabilizando uma avaliação precisa desses erros. A ação proposta na seção 5.2 visa sanar esses problemas em projetos futuros.

O primeiro passo para definir o processo para estimativa dos orçamentos foi a escolha do método de estimativa. A técnica de estimativa *bottom-up*, definida pelo PMBoK [PMI, 2004] já era usada e não apresentou bons resultados. Nessa técnica, uma estrutura analítica do projeto era elaborada até um nível de detalhamento possível com a informação que se possuía após entrevistas com o cliente. Os pacotes de trabalho eram estimados individualmente baseados na experiência dos profissionais e os totais de esforços, prazo e custos eram agrupados até se chegar ao orçamento.

Resolveu-se então, avaliar o COCOMO (*Construtive COst MOdel*) II [Boehm *et al.*, 2000] como método para estimativas dos orçamentos, já que essa é a técnica indicada pelo Praxis 3.0, processo no qual a Organização baseia o seu, além de vários relatos indicando seu uso com sucesso: [Boehm & Valerdi, 2008], [Helm, 1992]. Os resultados da utilização do COCOMO II foram promissores. Nas seções seguintes esse método será detalhado, e a forma de adoção e calibração explicada, além da apresentação dos seus resultados nos projetos da base histórica. O processo de estimativa para orçamento de projetos proposto é apresentado na seqüência.

5.1.1 O COCOMO II

O COCOMO é um método enquadrado na categoria de funções matemáticas, como descrito na seção 2.2. Ele se baseia em um conjunto de parâmetros de entrada utilizados em equações matemáticas que produzem as estimativas de esforço e prazo para um projeto. O método suporta estimativas tanto para projetos executados em ciclo de vida em espiral – como MBASE e processos similares (ex: RUP e Praxis 3.0) – quanto para ciclos de vida em cascata. No primeiro caso, o esforço medido em Pessoas-mês (PM) corresponde às fases de elaboração e construção. Já no caso de projetos executados no modelo em cascata, o esforço corresponde às fases de desenho em diante

(inclusive). Além disso, dois modelos são suportados: o *Early design* e o *Post-architecture*. O primeiro deve ser usado quando ainda se sabe pouco sobre o projeto e o segundo quando ao menos o desenho arquitetônico do produto já foi definido.

Os parâmetros do modelo consistem de seu tamanho estimado, em milhares de linhas de código (KLOC), os multiplicadores de esforços (EM) e os fatores de escala (SF). As equações abaixo são usadas no cálculo do esforço do projeto em PM, que correspondem a 152 h.

$$PM = A * KLOC^{E} * \prod_{i=1}^{n} EM_i$$

$$onde\ E = B + 0{,}01 * \sum_{i=1}^{5} SF_i$$

$$sendo\ A = 2{,}94\ e\ B = 0{,}91\ constantes$$

Equação 5-1 - Fórmula para estimativa de esforço do COCOMO II

A diferença do modelo *Early design* para o *Post-architecture* nas equações acima está relacionada com os multiplicadores de esforços EM. No primeiro modelo são usados apenas 7 multiplicadores e no segundo 17. Cada um desses multiplicadores de esforços é definido em níveis que variam de **Muito Baixo** até **Extra Alto**, sendo que cada nível possui um valor associado que aumentará, não impactará ou diminuirá o esforço estimado do projeto. A Tabela 5-1 mostra os multiplicadores utilizados no modelo *Post-Architecture*.

Pela tabela, pode-se perceber que parâmetros definidos no nível Nominal não aumentam e nem diminuem a estimativa de esforço, já que multiplicam por 1 a equação. O detalhamento de cada um dos parâmetros pode ser obtido no manual do COCOMO II, que também fornece diretrizes para a escolha dos níveis. Note que nem todos os parâmetros possuem todos os níveis.

Os fatores de escala possuem tabela semelhante, mas como eles estão localizados no expoente da equação, o impacto é não linear. A justificativa é a deseconomia de escala existente no desenvolvimento de software, onde usualmente percebemos um crescimento não linear do esforço em relação ao tamanho, como explicado no manual do método.

Tabela 5-1 - Valores dos níveis dos multiplicadores de esforços do modelo *Post-Architecture*

Símbolo	Nome	Nível					
		Muito baixo	Baixo	Nominal	Alto	Muito Alto	Extra alto
RELY	Confiabilidade requerida	0,82	0,92	1,00	1,10	1,26	
DATA	Volume de dados		0,90	1,00	1,14	1,28	
DOCU	Documentação requerida	0,81	0,91	1,00	1,10	1,23	
CPLX	Complexidade do produto	0,73	0,87	1,00	1,17	1,34	1,74
RUSE	Reusabilidade requerida		0,95	1,00	1,07	1,15	1,24
TIME	Restrições de tempo de execução			1,00	1,11	1,29	1,63
STOR	Restrições de memória			1,00	1,05	1,17	1,46
PVOL	Volatilidade da plataforma		0,87	1,00	1,15	1,30	
ACAP	Capacitação dos analistas	1,42	1,19	1,00	0,85	0,71	
APEX	Experiência com a aplicação	1,22	1,10	1,00	0,88	0,81	
PLEX	Experiência com a plataforma	1,19	1,09	1,00	0,91	0,85	
PCAP	Capacitação dos programadores	1,34	1,15	1,00	0,88	0,76	
LTEX	Experiência com a linguagem e ferramentas	1,20	1,09	1,00	0,91	0,84	
PCON	Continuidade de pessoal	1,29	1,12	1,00	0,90	0,81	
TOOL	Adequação das ferramentas	1,17	1,09	1,00	0,90	0,78	
SCED	Folga em relação ao prazo	1,43	1,14	1,00	1,00	1,00	
SITE	Co-localização da equipe	1,22	1,09	1,00	0,93	0,86	0,80

Para o cálculo do tempo de desenvolvimento (TDEV) em meses, são utilizados o esforço dado pela Equação 5-1 (PM) e o somatório dos fatores de escala (SF) como apresentado na equação abaixo.

$$TDEV = C * PM^F$$

$$onde\ F = D + 0{,}002 * \sum_{i=1}^{5} SF_i$$

$$sendo\ C = 3{,}67\ e\ D = 0{,}28$$

Equação 5-2 - Fórmula para estimativa de prazo do COCOMO II

5.1.2 Tamanho do produto

Como é mostrado na Equação 5-1, O COCOMO II utiliza linhas de código para medir o tamanho do produto que está sendo estimado. Ele faz uso de uma lista de conferência, como recomendado pelo relatório do SEI [Park *et al.*, 1992] para formalizar essa contagem. Na lista de conferência utilizada para formalização da contagem há dois tipos principais de definições:

- o que é e o que não é considerado como linha de código na contagem sob os diversos aspectos: tipo de instrução, como foi produzida, propósito de uso, estado do desenvolvimento, origem, entre outros;
- vetores de dados, que consiste em classificar cada linha considerada na contagem conforme opções dentro de cada vetor. Para melhorar o entendimento, a Figura 5-1 exemplifica a definição de um vetor de dados. Por esse exemplo, cada linha de código deve ser enquadrada em exatamente uma opção de Tipo de arquivo. Uma lista de conferência pode ter vários desses vetores e fica a critério de cada organização definí-los.

Tipo de arquivo	Definição		Vetor de dados	X	Inclui	Exclui
1 Java (arquivos com extensão .java, exceto com o padrão de nome *Helper.java)					X	
2 Web Estática (.html, .htm, .css, .xsl)					X	
3 Web Dinâmica (.jsp, .js)					X	
4 Configuração (.xml, .config, .properties)					X	
5 Robot (.rec, .sbl, .sbh)					X	

Figura 5-1 - Exemplo de vetor de dados utilizado na lista de conferência de contagem de linhas de código

Nesse trabalho, utilizam-se as linhas de código físicas como medida de tamanho para utilização no modelo, embora o manual do COCOMO II defina a utilização das linhas lógicas (ver seção 2.4 para detalhes sobre tipos de linhas de código). A escolha de linhas físicas sobre as lógicas vem da sua facilidade de obtenção e da crença que ambas representam igualmente o tamanho físico do produto, desde que haja padronização de codificação entre os produtos medidos e a linguagem de programação seja a mesma. Esse é justamente o caso da Organização alvo desse programa, cujo processo executado possui inspeções de código para garantir a padronização. O estudo de Rosenberg [Rosenberg, 1997] corrobora essa hipótese.

Para definição formal da contagem de linhas de código físicas, utilizaram-se as

recomendações do mesmo relatório do SEI, definindo-se uma lista de conferência, que pode ser vista em detalhes no Apêndice A.1.

Além da formalização da definição da contagem, foi desenvolvida uma ferramenta de contagem de linhas de código que é aderente ao padrão definido pelo SEI. A ferramenta é totalmente personalizável, através de arquivo XML, com qualquer definição que utilize o formato de lista de conferência proposto. A Figura 5-2 mostra um exemplo de configuração do vetor de dados mostrado na Figura 5-1, que avalia expressões regulares [Friedl, 2006] comparando-as ao caminho completo de cada arquivo analisado para enquadrar cada linha contada em uma das opções do vetor. A ferramenta não garante que dada uma configuração feita pelo usuário, uma linha contada receba exatamente um rótulo do vetor, mas ao final o usuário poderá verificar se alguma linha foi rotulada com mais de uma ou nenhuma opção do vetor e fazer as devidas correções na configuração.

```
<tag nome="Tipo de arquivo">
	<item valor="Java">
		<padrao valor="\.java" tipo="inclui" />
	</item>
	<item valor="Web">
		<padrao valor="\.css" tipo="inclui" />
		<padrao valor="\.htm?" tipo="inclui" />
		<padrao valor="\.xsl" tipo="inclui" />
	</item>
	<item valor="Web dinâmica">
		<padrao valor="\.js" tipo="inclui" />
		<padrao valor="\.jsp" tipo="inclui" />
		<padrao valor="\.tdl" tipo="inclui" />
	</item>
	<item valor="Config">
		<padrao valor="\.xml|\.config|\.properties" tipo="inclui" />
	</item>
	<item valor="Robot">
		<padrao valor="\.rec|\.sbl|\.sbh" tipo="inclui" />
	</item>
</tag>
```

Figura 5-2 - Configuração de vetor de dados da ferramenta de contagem de linhas de código

O relatório de contagem emitido pela ferramenta está no formato CSV (*comma separated values*), que permite a importação por diversas ferramentas para análise dos dados. O relatório apresentado na Tabela 5-2 foi elaborado com o recurso de tabela dinâmica do *Microsoft Excel*, mostrando a contagem de um projeto incluindo os 3 vetores de dados definidos no apêndice A.1. Vale ressaltar que esse relatório apresentado na tabela inclui linhas de código que não deveriam ter sido contadas conforme a lista de conferência da Organização, que, por exemplo, exclui a contagem das linhas de código de testes de desenvolvimento e de sistema. Essas linhas não foram consideradas para

formar o tamanho do produto, mas a ferramenta de contagem foi configurada para coletar esses dados extras que fornecem informações importantes do projeto, como a relação entre linhas de código de testes e as do produto. No exemplo apresentado na Tabela 5-2, basta remover os valores das células sombreadas na tabela da contagem final.

Tabela 5-2 - Exemplo de relatório de contagem de linhas de código

Origem do Código	Camada	Config	Java	Web	Web dinâmica	Total geral
Código Novo	Controle	61	34.966			35.027
	Entidade		147.217		153	147.370
	Fronteira	7.845	114.135	10.591	102.234	234.805
	Outros	5.600	31.044		124.553	161.197
	Povoadores		156.757			156.757
	Testes de desenvolvimento	896	245.253	181	11	246.341
	Testes de sistema		81.232			81.232
Total Código Novo		14.402	810.604	10.772	226.951	1.062.729
Reuso	Fronteira	60	7.214	2.932	3.318	13.524
	Outros	1.018	1.533	113	4.620	7.284
	Povoadores		28			28
	Testes de desenvolvimento	690	2.108	1.215	8.051	12.064
Total Reuso		1.768	10.883	4.260	15.989	32.900
Total geral		16.170	821.487	15.032	242.940	1.095.629

5.1.3 Calibração e validação do COCOMO II

Como apresentado na seção 2.3, não há um consenso sobre um indicador de acurácia que avalie de forma conclusiva a qualidade das estimativas de um método. Portanto, nesse trabalho, foram utilizados vários deles para chegarmos às nossas conclusões. Já para a escolha do método de avaliação, foi escolhida a técnica *Leave-one-out cross validation*, que como explicado na seção 2.5, reflete melhor a situação real nas organizações, quando se possui dados históricos de um conjunto de projetos já executados e deseja-se estimar o próximo.

Para validar a escolha do COCOMO II como método para estimar os orçamentos de novos projetos, aplicou-se as fórmulas do modelo *Post-Architecture* aos 8 projetos já concluídos pela Organização, cujos dados coletados, principalmente os de esforço e de tamanho são confiáveis. São todos projetos de produtos para web desenvolvidos em linguagem Java, com tamanho em Pontos de função, variando entre 200 e 4500 e esforços reais variando entre 7,5 e 955,2 Pessoas-mês.

Conforme explicado na seção 3.2, que fala da Organização alvo desse programa, esses projetos foram executados no ciclo de vida de entrega evolutiva (versão 2.1 do processo) e os dados analisados nessa seção dizem respeito ao Projeto de Construção dos produtos (a Especificação deles foi feita em projeto anterior). Comparando-se as etapas do Projeto de Construção executado em ciclo de vida de Entrega evolutiva com as fases do modelo Cascata suportado pelo COCOMO II, observamos uma correspondência do esforço estimado pelo modelo com as etapas do Projeto de Construção, como mostrado na Figura 5-3.

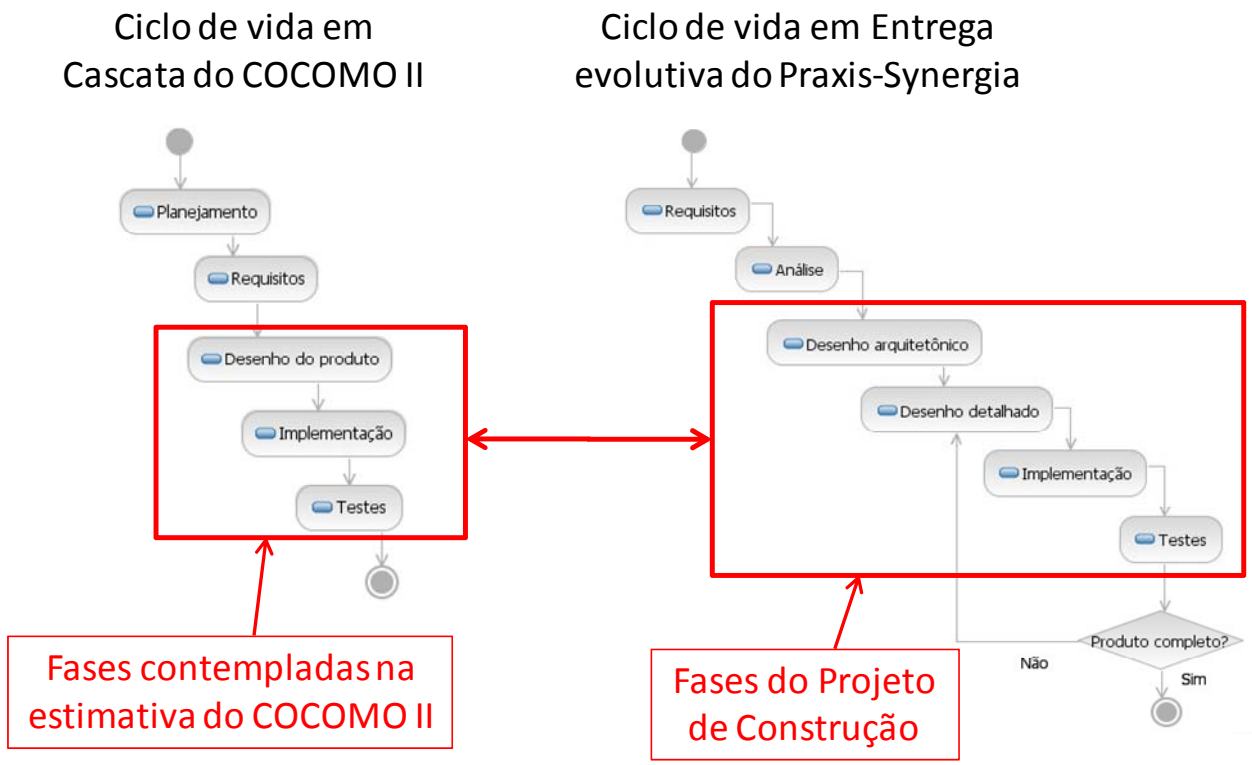

Figura 5-3 - Comparação do modelo cascata do COCOMO II com o Praxis-Synergia.

Como se pode ver na Tabela 5-3, os resultados da utilização do método foram muito ruins, mesmo em indicadores que não utilizam o operador módulo, como no MBRE e MIBRE, quando erros em projetos subestimados tendem a se cancelar com erros nos superestimados[5]. A principal justificativa encontrada, é que as particularidades da Organização tornam seus projetos muito diferentes dos projetos que constituem a base do modelo padrão. Um exemplo é a relação de

[5] Os resultados do MMER e MMER ficaram iguais aos do MBRE e MIBRE porque os projetos foram todos superestimados pelo modelo padrão.

linhas de código por ponto de função. Na Organização, a média dessa relação entre os 8 projetos analisados é de 170 LOC/PF para a linguagem Java, enquanto o modelo padrão sugere uma relação de 53 LOC/PF.

Tabela 5-3 - Resultados da avaliação de estimativa de esforço do COCOMO II com o modelo padrão

MMRE	MMER	MBRE	MIBRE	PRED(0,25)
319,50%	71,97%	319,50%	71,97%	0,00%

Com essa magnitude de erro na estimativa, torna-se necessário calibrar o COCOMO II, como seus próprios autores sugerem fazer. Essa calibração envolve utilizar regressão linear múltipla pelo método *Ordinary Least Squares* (OLS) (ver [Jain, 1991] para maiores detalhes). Entretanto, como o próprio nome sugere, o OLS pode ser utilizado apenas em modelos lineares, o que não é o caso, já que temos variáveis no expoente das fórmulas. A saída, nesse caso, é linearizar a fórmula por manipulação algébrica, utilizando Logaritmo.

$$Log(PM) = Log(A) + (B + 0{,}01 * \sum_{i=1}^{5} SF_i) * Log(KLOC) + Log(\prod_{i=1}^{n} EM_i)$$

Equação 5-3 - Fórmula de estimativa de esforço linearizada

A partir da Equação 5-3, o passo seguinte é isolar as constantes A e B que devem ser calibradas, o que resulta na Equação 5-4. Em seguida, aplica-se a regressão linear múltipla para ajustar as constantes A e B.

$$Log(PM) - \left(0{,}01 * \sum_{i=1}^{5} SF_i\right) * Log(KLOC) - Log(\prod_{i=1}^{n} EM_i) = Log(A) + B * Log(KLOC)$$

Equação 5-4 - Manipulação da Equação 5-3 para isolar as constantes A e B

Após a calibração das constantes, reavaliamos os mesmos indicadores para as estimativas feitas nos 8 projetos e os resultados são apresentados na Tabela 5-4. Port e Korte [Port & Korte, 2008] sugerem que um MMRE menor que 25% e PRED(0,3) maior que 75% são considerados valores aceitáveis como boas estimativas, o que nos leva a crer que os resultados da calibração foram muito bons. Quando se considera o sinal dos erros (superestimativa compensando subestimativa), como no caso dos indicadores MBRE e MIBRE, observamos desvios médios muito pequenos, próximos de zero.

Tabela 5-4 - Resultados da avaliação de estimativa de esforço do COCOMO II com modelo calibrado

MMRE	MMER	MBRE	MIBRE	PRED(0,25)
15,63%	14,44%	-0,31%	-1,51%	87,50%

O mesmo procedimento de calibração foi realizado para as equações de estimativa de prazo para obter novas constantes C e D e os resultados da avaliação encontram-se na Tabela 5-5.

Tabela 5-5 - Resultados da avaliação de estimativa de prazo do COCOMO II com modelo calibrado

MMRE	MMER	MBRE	MIBRE	PRED(0,25)
43,34%	44,68%	0,17%	1,51%	37,50%

Esses resultados não foram tão bons quanto os de esforços. Uma possível justificativa é a presença de alguns projetos que tiveram sua execução paralisada e não foi possível recuperar o tempo de paralisação para os mesmos, tornando a calibração menos precisa. Outro motivo é o provável aumento do escopo durante a construção do produto, mas como não há registro do tamanho das alterações, fica inviável tirar conclusões definitivas a respeito.

De posse dos indicadores acima, pode-se dizer que o COCOMO II apresentou-se como um método promissor para ser introduzido no processo de estimativa dos orçamentos de novos projetos da Organização. Entretanto, essa análise deve ser refeita periodicamente a medida que novos projetos compuserem os dados históricos, já que, 8 projetos são uma massa de dados pequena para se obter conclusões realmente precisas.

5.1.4 A adoção do COCOMO II na Organização

Embora os resultados apresentados na seção anterior sejam promissores, um ponto muito importante de se salientar, é que a validação foi feita com o tamanho real final dos produtos. Quando se está estimando um novo projeto com o COCOMO II, é necessário estimar também o seu tamanho em linhas de código, que é uma tarefa difícil. Uma saída proposta pelos autores do método é a utilização de Pontos de função não-ajustados (UFP) do IFPUG [IFPUG, 2005] como medida alternativa de tamanho do produto. A explicação é a facilidade de obtenção dessa informação logo nas etapas iniciais do projeto.

A Organização já utiliza a estimativa de tamanho em Pontos de função não-ajustados em seus projetos. Para isso, identifica-se junto aos usuários a lista das funções de transação e dados do

produto. No melhor caso, quando há modelagem de sistema que inclui o produto sendo estimado, essa lista carrega um grau de incerteza baixo. Quando não há essa modelagem, a lista de funções normalmente sofre um grau maior de variação entre a estimativa e o que virá a ser o produto, já que ela será obtida por meio de poucas reuniões durante a iteração de Ativação do projeto.

A partir da lista de funções, os analistas de requisitos mais experientes estimam a complexidade de cada função, obtendo assim, o total de Pontos de função não-ajustado do produto (ver resumo do processo de contagem na seção 5.2.2).

Outra forma de estimativa de Pontos de função não-ajustados utilizada pela Organização é descrita por McConnell [McConnell, 2006] como o "Método Holandês" (pois é definido pelo *Netherlands Software Metrics Association* – NESMA). Nesse método, a contagem indicativa de Pontos de função não-ajustados é dada por **(35 x número de Arquivos Lógicos Internos) + (15 x número de Arquivos de Interface Externa)**. O resultado dessa equação é confrontado com a estimativa obtida a partir da lista de funções como "verificação de sanidade".

De posse da estimativa em Pontos de função não-ajustados, deve-se obter a média de linhas de código por Ponto de função (LOC/PF) da base histórica dos projetos, para obter o número de linhas de código que servem como medida de tamanho do COCOMO II. Entretanto, utilizar a média só faz sentido se a variação das medidas não for grande. Não foi possível realizar essa análise nos dados históricos da Organização pela ausência da contagem final de Pontos de função dos produtos e essa questão deve ser observada no futuro para validar a utilização do COCOMO II como método de estimativa.

Como se pode inferir a partir da Tabela 5-1, na seção 5.1.1, a escolha de um nível para cada multiplicador de esforço é uma tarefa delicada, já que uma escolha errada pode levar a uma grande variação no esforço estimado. Para ilustrar, a escolha do nível **Muito baixo** para a **Capacitação dos analistas** (ACAP), multiplicaria o esforço total estimado por 1,42, ou seja, aumentaria o esforço em 42% considerando os demais multiplicadores como constantes. Se a escolha que melhor refletisse a realidade do projeto fosse o nível **Baixo**, o esforço seria multiplicado por 1,19. Com isso, a diferença entre a escolha feita e a melhor escolha influenciaria a estimativa final de esforço em 23%. No trabalho de Clark e outros [Clark *et al.*, 1998] são apresentados possíveis efeitos de má interpretação desses parâmetros, que podem levar a comportamento inverso ao previsto no método. Com o intuito de minimizar esse problema, foi elaborado um documento para auxiliar a interpretação de cada parâmetro, que inclui diretrizes que complementam a documentação do método [Boehm *et al.*, 2000] sob o ponto de vista das especificidades da Organização.

O documento proposto acima diminui o risco de má interpretação na escolha dos

parâmetros, mas não elimina os riscos inerentes de todo projeto de desenvolvimento. Foi elaborada uma planilha para avaliar qualitativamente os diversos riscos dos cenários possíveis para o projeto e quais parâmetros seriam impactados em cada um deles. Dessa forma, os gestores da Organização poderão tomar melhores decisões na formação de seus preços a partir do custo estimado pelo COCOMO II. A Figura 5-4 exemplifica a utilização dessa planilha.

Cenários			Estimativas	
Cenário	**Parâmetros/Fatores variado (informar novo valor)**	**Riscos envolvidos**	**Esforço (PM)**	**Prazo (mês)**
Esperado	-	-	323,4	8,7
Pessimista	ACAP: de Muito Alto para Alto PCAP: de Alto para Nominal	Saída dos colaboradores mais experientes da equipe do projeto	439,9	9,4
Otimista	CPLX: de Alto para Nominal	Complexidade do produto pode ser menor que o levantado inicialmente	276,4	8,4

Figura 5-4 - Planilha para análise qualitativa de riscos das estimativas.

5.1.5 O processo de estimativa para orçamento de projetos

Com base na avaliação positiva do COCOMO II e nas propostas da seção anterior, foi definido um processo para orçar os novos projetos da Organização. Esse processo, que será apresentado a seguir, foi documentado utilizando o *Eclipse Process Framework* (EPF)[6], ferramenta de código livre aderente ao *Software & Systems Process Engineering Metamodel* 2.0 (SPEM) [OMG, 2008], que é o padrão definido pelo OMG (*Object Modeling Group*)[7] para documentação de processos de desenvolvimento de software e sistemas.

O processo proposto com base nas técnicas e melhorias apresentadas nas seções anteriores,, cujas atividades são ilustradas na Figura 5-5, foi incorporado à fase de Iniciação do Praxis-Synergia.

[6] www.eclipse.org/epf

[7] www.omg.org

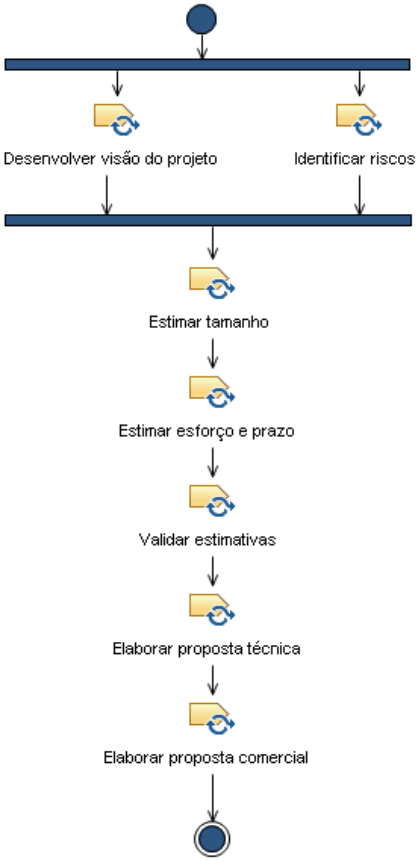

Figura 5-5 - Processo para orçamento de projetos.

Na Tabela 5-6 cada uma das atividades é detalhada, bem como os seus insumos e resultados. Esse processo foi incorporado à iteração de ativação na personalização do Praxis adotado pela Organização.

Tabela 5-6 - Detalhamento das atividades do processo de orçamento proposto.

Atividade	Descrição
Desenvolver visão do projeto	A visão geral do projeto consiste na produção de dois artefatos: **Documento de visão** e **Diagrama de contexto**, ambos levantados junto a reuniões com o cliente potencial. No **Documento de visão** constam os objetivos do projeto, a lista de requisitos funcionais e não-funcionais, produtos que devem ser integrados ou ter seus dados migrados, metas gerenciais, limitações e premissas do projeto. No **Diagrama de contexto**, está o modelo UML com os casos de uso e atores identificados durante as reuniões.
Identificar riscos	Nessa atividade os riscos e oportunidades do projeto são identificados de forma preliminar, já que um novo levantamento detalhado é realizado no início do projeto quando ele for contratado. Essa lista serve como insumo para as atividades de estimativa de tamanho, esforço e prazo.

Estimar tamanho	A estimativa de tamanho consiste em estimar o tamanho do produto em Pontos de função não-ajustados, como descrito na seção 5.1.4. As funções de dados e transações são obtidas em reuniões de levantamento de requisitos junto ao cliente e a complexidade delas é estimada para se chegar ao total de Pontos de função.
Estimar esforço e prazo	Nessa atividade utiliza-se o COCOMO II para estimar o esforço e prazo do projeto. O primeiro passo é calibrar o COCOMO II com os dados históricos dos projetos. Então, utiliza-se a estimativa de tamanho e a planilha de interpretação dos parâmetros (ver seção 5.1.4) para estimar o esforço e o prazo. Essas estimativas são realizadas com um perfil de equipe disponível em mente, já que esse perfil impacta diversos fatores do método. A lista de riscos identificados também é utilizada aqui para avaliar o impacto de cada risco e oportunidade nos parâmetros do método e criar possíveis cenários, com as estimativas correspondentes, como ilustrado na Figura 5-4 da seção anterior. O resultado dessa atividade produz um relatório com os parâmetros informados, o perfil de equipe assumido e a análise qualitativa dos riscos.
Validar estimativas	A validação das estimativas consiste numa revisão gerencial dos artefatos produzidos, verificando a validade das premissas assumidas e a consistência entre os artefatos. Essa revisão é feita por uma pessoa externa à equipe de produção, que não possui compromisso com a venda do projeto. Isso diminui a tendência ao otimismo na imputação dos parâmetros do COCOMO II.
Elaborar proposta comercial	A partir dos resultados produzidos pelas estimativas, a direção da Organização toma decisões sobre o preço de venda, incluindo (ou não) a margem de lucro desejada.

5.2 O processo de contagem de pontos de função

Nessa seção as melhorias relacionadas à contagem de Pontos de função dos projetos serão tratadas. A Organização já utiliza o padrão de contagem do *International Function Point User Group* (IFPUG) [IFPUG, 2005] formalmente para medir o tamanho de seus produtos, mas o registro é realizado em planilhas eletrônicas. Os principais problemas associados a essa forma de registro são:

- Não há uma amarração com a modelagem do produto. Com isso, alterações feitas no modelo não são refletidas automaticamente na contagem, tornando-se um risco da contagem ficar inconsistente;
- Esforço grande para o registro da contagem, pois a descrição dos itens considerados é feito de forma textual e manual.

Para solucionar esses problemas foi proposta uma nova maneira de registro da contagem, realizada diretamente sobre os modelos UML do produto sendo modelado e uma integração com o **Cadastro de Requisitos** do projeto. Na versão padrão (ou educacional) do Praxis, já há uma

proposta de mapeamento de elementos do **Modelo do Problema** para funções da contagem de Pontos de função. Essa proposta inova apenas na forma de identificar, mapear e registrar a contagem em modelos UML, mas continua totalmente aderente ao manual de práticas de contagem do IFPUG e ao Praxis padrão.

Além da forma de registro de contagem proposta, um processo de gestão das alterações e respectiva atualização da contagem também foram desenvolvidos. Nas seções seguintes apresentam-se os detalhes dessas propostas.

Vale aqui destacar que a nossa proposta é restrita a projetos de desenvolvimento de software que adotem a UML como notação de modelagem para expressar os requisitos.

5.2.1 O modelo do problema

Nessa seção apresentaremos de forma resumida o artefato denominado **Modelo do Problema**, que é utilizado na Organização, que adota o mesmo padrão proposto no Praxis 3.0. A ferramenta utilizada nessa modelagem é o *Rational Software Architect*[8] e as figuras de diagramas UML [OMG, 2005b] apresentadas aqui são extraídos dela. Será dado enfoque nos pontos importantes para a contagem de pontos de função que será apresentada na seção seguinte. Detalhes do artefato podem ser obtidos no Capítulo 16 do livro do processo [Pádua, 2008].

A modelagem do problema consiste em desenvolver um modelo UML que represente os requisitos do cliente em relação ao produto sendo desenvolvido e o entendimento dos desenvolvedores de como esse sistema irá funcionar. Essa modelagem é independente de tecnologia e da solução que será proposta para o problema, o que a torna mais próxima da Visão do Usuário descrita no manual de contagem do IFPUG.

O modelo segue as recomendações do padrão IEEE-830 [IEEE, 1998] e é dividido em duas visões principais: Visão de requisitos e Visão de análise. Na primeira estão as funções do produto do ponto de vista do usuário, mostrando os casos de uso (com seus fluxos), os atores e a interação entre eles documentadas com diagramas de atividade, como ilustra a Figura 5-6. Em cada fluxo dos casos de uso é aplicado um estereótipo para representar seu tipo: principal, subfluxo ou fluxo alternativo («mainFlow», «subFlow» e «altFlow» respectivamente).

[8] http://www-01.ibm.com/software/awdtools/swarchitect/websphere/

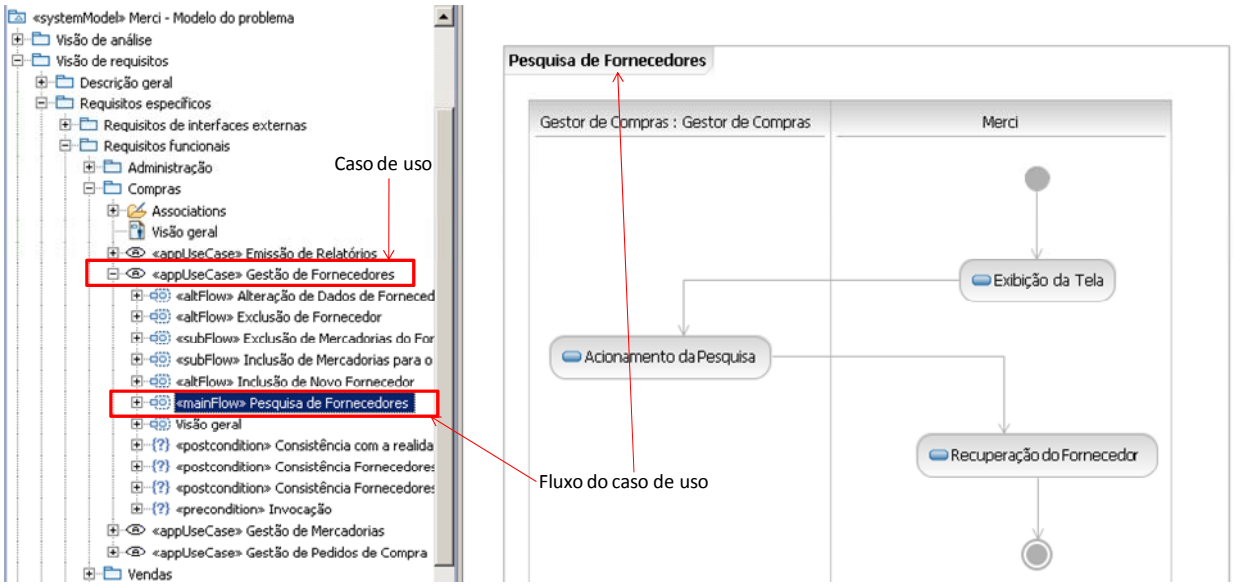

Figura 5-6 - Modelo do problema: visão de requisitos. Extraído do exemplo do Praxis 3.0 [Pádua, 2008]

Na Visão de análise, encontram-se as colaborações que realizam os fluxos dos casos de uso. Cada caso de uso é realizado por uma colaboração das classes modeladas pelo analista. A Figura 5-7 mostra a associação de realização entre uma colaboração e o caso de uso.

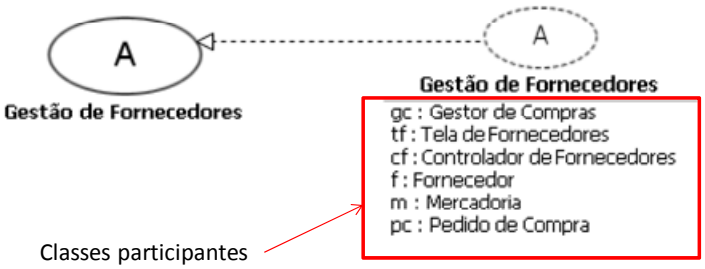

Figura 5-7 - Realização de um caso de uso por uma colaboração. Extraído do exemplo do Praxis 3.0.

Cada um dos fluxos dos casos de uso descritos na Visão de requisitos é realizado por uma interação entre as classes identificadas, modeladas em diagramas de seqüência. Essas classes são estereotipadas como entidade, entidade persistente, controle e fronteira («entity», «persistentEntity», «control» e «boundary» respectivamente). A Figura 5-8 mostra a realização do fluxo ilustrado na Figura 5-6.

Figura 5-8 - Realização de um fluxo de caso de uso. Extraído do exemplo do Praxis 3.0

5.2.2 Introdução à contagem de pontos de função do IFPUG

Nessa seção apresentaremos um breve resumo dos conceitos e procedimentos de contagem de pontos de função conforme especificação do IFPUG [IFPUG, 2005]. A Figura 5-9 mostra as atividades que compõem o processo de contagem do manual.

Na primeira atividade, devemos definir o tipo de contagem que está sendo feito. Podem ser de 3 tipos: contagem de Pontos de função de **projeto de desenvolvimento**, que mede o tamanho de um novo produto sendo desenvolvido; contagem de Pontos de função **de aplicação**, que engloba a contagem anterior mais os pontos de função para migração e conversão de dados para que o produto entre em operação; e a contagem de Pontos de função de **projeto de melhoria**, que mede o tamanho de uma evolução/manutenção em um produto já desenvolvido.

A seguir deve-se identificar o escopo da contagem e a fronteira da aplicação. Esse é um passo importante, que cria os limites entre o produto e o(s) usuário(s) dele.

Definido e o escopo e a fronteira, deve-se então contar os pontos de função de dados e os pontos de função de transação.

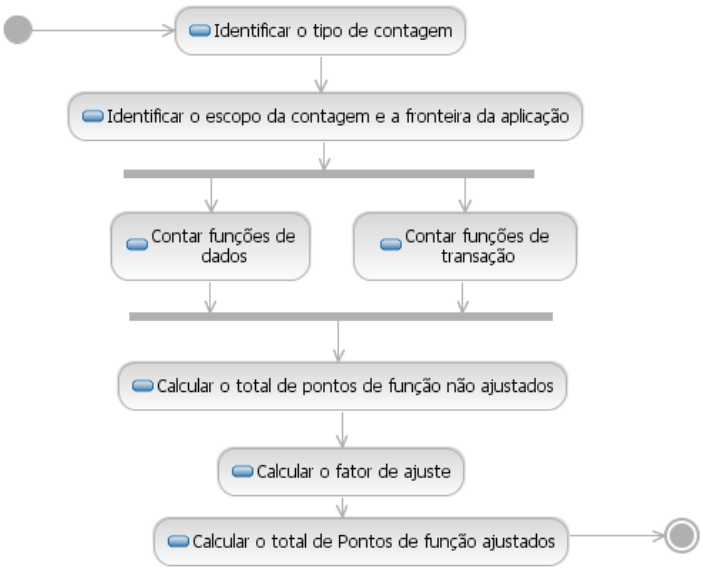

Figura 5-9 - Procedimento de contagem de pontos de função. Elaborado com base no manual do IFPUG.

As funções de dados podem ser de dois tipos: **Arquivo Lógico Interno** (ALI) e **Arquivo de Interface Externa** (AIE). O primeiro trata dos requisitos de dados que serão mantidos e consultados pela aplicação sendo contada e o segundo representa dados apenas consultados pela aplicação, mas mantidos por outro produto.

As funções de transação representam funcionalidades que o produto fornece ao usuário para processar os dados. Elas podem ser de três tipos:

- **Consulta Externa** (CE): são funções que consultam dados da aplicação e os envia para o usuário. Nessas funções nenhuma função de dados é alterada e não há nenhum cálculo ou aplicação de fórmulas sobre elas;
- **Saída Externa** (SE): similares às Consultas Externas, mas possuem fórmulas de cálculo ou dados derivados das funções de dados. Podem atualizar uma ou mais ALIs e alterar o comportamento do sistema;
- **Entrada Externa** (EE): são funções que partem de fora da fronteira da aplicação (normalmente disparadas pelo usuário) e que alteram uma ou mais ALIs.

Dada a contagem de funções de dados e transações, utilizam-se tabelas que, a partir de características dessas funções, identificam a complexidade delas em Baixa, Média e Alta. A partir da complexidade, outra tabela fornece o tamanho em pontos de função não ajustados para cada função dada sua complexidade (ver Tabela 5-7).

Tabela 5-7 - Pontos de função de cada função dada sua complexidade. Extraída do manual de contagem.

Função	Complexidade		
	Baixa	Média	Alta
ALI	7	10	15
AIE	5	7	10
CE	3	4	6
SE	4	5	7
EE	3	4	6

Em seguida, o padrão do IFPUG determina o cálculo de um fator de ajuste, que visa ajustar o tamanho do produto conforme características não funcionais que impactam a sua produtividade. No entanto, como a Organização utiliza Pontos de função não-ajustados, essa etapa não será detalhada aqui.

5.2.3 A contagem de pontos de função a partir do modelo do problema

De posse dos conceitos apresentados nas seções anteriores, descreve-se aqui a proposta para contagem dos pontos de função diretamente no **Modelo do Problema**. A base dessa proposta é a extensa utilização de mecanismos de extensão da UML [OMG, 2005b], com utilização de estereótipos e restrições formais escritas em OCL [OMG, 2005a]. Foram criados diversos estereótipos, com propriedades para guardar informações de contagem de Pontos de função, e restrições OCL associadas a eles. Os estereótipos e restrições foram encapsulados em um Perfil UML que é implantado na ferramenta de modelagem (*Rational Software Architect*) através de um plugin, que é o mecanismo de extensão dessa ferramenta.

Propostas semelhantes de contagem de pontos de função sobre modelos UML já foram apresentadas anteriormente: [Caldiera *et al.*, 1998], [Cantone *et al.*, 2004], [Harput *et al.*, 2005], [Uemura *et al.*, 1999]. No entanto, essas propostas tentam automatizar parte ou toda a contagem de Pontos de função a partir do modelo. A diferença para o que será proposto aqui é que a contagem não é automatizada em nenhum momento, mas sim integrada ao **Modelo do Problema**. O analista deve tomar todas as decisões de modelagem e mapeamento do modelo para as funções de dados e transação, como mostraremos na próxima seção.

A proposta apresentada aqui se restringe à contagem de Pontos de função não ajustados. A utilização do fator de ajuste vem sendo questionada em alguns trabalhos. Lokan [Lokan, 2000], por exemplo, mostra que a utilização do fator não melhora em nada a correlação do tamanho com o

esforço. Outro motivo é a adoção do COCOMO II (como descrito na seção 5.1), que utiliza Pontos de função não-ajustados como entrada. Por último, o fator de ajuste também não é utilizado em normas como a IEEE *Std 1045-1992* [IEEE, 1992] que define métricas de produtividade em software.

5.2.3.1 Contagem das funções de dados

No **Modelo do Problema**, os dados mantidos pela aplicação são modelados como classes da UML, com estereótipo «persistentEntity». Para contar os pontos de função de dados diretamente no modelo, foram acrescentados atributos ao estereótipo «persistentEntity», que tornou-se abstrato. Criou-se também 2 novos estereótipos: «internalPersistentEntity» e «externalPersistentEntity», onde o primeiro representa as classes do produto sendo modelado e o segundo representa as classes de outros sistemas. A Figura 5-10 mostra o diagrama desses estereótipos.

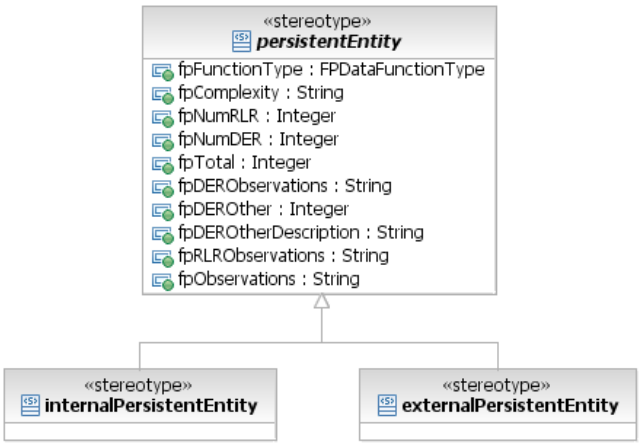

Figura 5-10 - Estereótipos para contagem de funções de dados.

Na Tabela 5-8 é feita uma descrição de cada atributo para contagem dos pontos de função de dados. O mapeamento de classes de entidade para as funções de dados não é direto e nem é passível de automação, cabendo ao analista especializado em contagem fazer os mapeamentos.

Além dos atributos nos estereótipos, o *plugin* para o *Rational Software Architect* facilita o preenchimento das informações e o cálculo da complexidade e do total de pontos de função conforme o padrão do IFPUG. A seguir, apresentaremos um exemplo da utilização desse *plugin* num problema simples.

Tabela 5-8 - Atributos para contagem de funções de dados.

Atributo	Descrição
fpFunctionType	Tipo de função de dados: ALI, AIE ou nenhum.
fpComplexity	Complexidade da função de dados: Alta, Média, Baixa. É calculada por comando do usuário, a partir das demais informações.
fpNumRLR	Contagem de RLR[9] (Registros Lógicos Referenciados).
fpNumDER	Contagem de DER (Dados Elementares Referenciados).
fpTotal	Total de pontos de função, calculado a partir da complexidade e do tipo.
fpDER	Lista de DERs considerados na contagem. Contém uma lista de atributos de classes (Property da UML). Não aparece na Figura 5-10 por limitações da ferramenta.
fpDERObservations	Texto com observações para contagem de DER.
fpDEROther	Utilizado para informar DERs que eventualmente não estejam modelados como atributos das classes, não estando, portanto incluídos no atributo fpDER. É utilizado, por exemplo, para contar DERs referentes a relacionamentos.
fpDERDescription	Documentação descritiva dos itens informados no campo fpDEROther.
fpRLR	Lista de RLR considerados na contagem. Contém uma lista de classes. Não aparece na Figura 5-10 por limitações da ferramenta.
fpRLRObservations	Observações para a contagem de RLR.

Um dos aspectos importantes da contagem das funções de dados, segundo as regras do IFPUG, é que a identificação das funções deve ser sempre feita sob a perspectiva do usuário, independentemente de como isso se reflete na modelagem ou na solução dada para o requisito. Como conseqüência, seria incorreto assumir que cada classe persistente corresponde a uma função de dado (ALI ou AIE), já que a modelagem das classes reflete as decisões de modelagem do analista de requisitos. Diversas classes podem compor uma única função de dado, e pode haver classes que sequer sejam consideradas para a contagem (é o caso dos "code data", tratados no manual do IFPUG). Por isso, na abordagem aqui proposta, o analista é quem deve avaliar quais classes devem ser agrupadas, e quais devem ser desconsideradas.

Dado o diagrama de classes da Figura 5-11, extraído do exemplo do Praxis 3.0, o analista deve decidir quais classes serão agrupadas em cada função de dados. Podemos ver na figura a identificação de 2 funções ALI. Nas funções que agrupam mais de uma classe, como é o caso do grupo **Pedido de compra** e **Item de compra**, deve-se escolher uma delas como sendo a principal, cujo nome representa melhor o agrupamento. Nesse caso, foi escolhida a classe **Pedido de compra**.

[9] Notar que a terminologia usada aqui é a mesma do manual de contagem do IFPUG em português, mas diferente da adotada no Praxis padrão, que utiliza TAR e TER no lugar de RLR e DER.

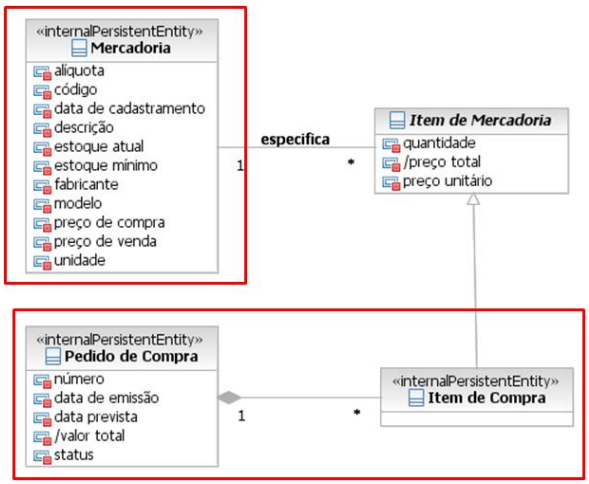

Figura 5-11 - Exemplo de identificação de funções de dados em diagrama de classes.

Decididos os agrupamentos das funções de dados, seleciona-se a classe principal do grupo e são preenchidas as informações de contagem. A lista de classes agrupadas (RLR – Registros Lógicos Referenciados) é montada selecionando-se quaisquer classes do modelo que possuem estereótipo de entidade («internalPersistentEntity» e «externalPersistentEntity»). A lista de DER (Dados Elementares Referenciados) é montada a partir dos atributos das classes selecionadas como RLR. Após o preenchimento dos dados, o analista dispara o comando que calcula a complexidade e o total de pontos de função daquela função. Cabe explicitar que o total contado de DER e RLR não é simplesmente a lista de classes e atributos selecionados, já que o analista pode alterar a contagem desses itens acrescentando, por exemplo, DERs adicionais – como os que representam associação entre classes – ou desconsiderar algum item que não atenda aos requisitos de contagem.

A Figura 5-12 mostra o preenchimento das informações da função de dados **Pedido de Compra** no *plugin* desenvolvido.

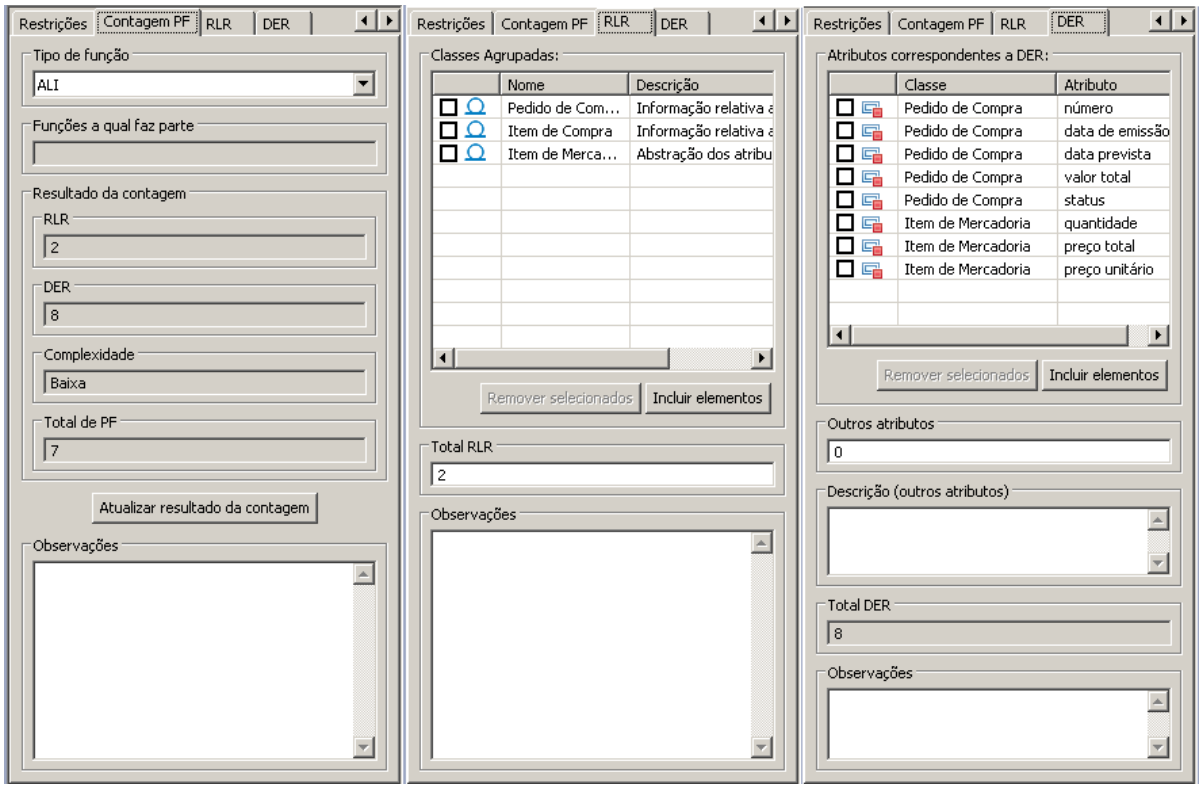

Figura 5-12 - Preenchimento das informações de contagem de função de dados no *plugin*.

Para complementar o *plugin*, utilizou-se o arcabouço da UML 2.0 para prevenir alguns tipos de erros no processo de contagem. Foram criadas restrições escritas em OCL (*Object Constraint Laguange*) [OMG, 2005a] associadas aos estereótipos das classes. Essas restrições são checadas durante a validação disparada na ferramenta (*Rational Software Architect*) e automatizam várias verificações que antes eram feitas manualmente, durante os procedimentos de inspeção do modelo do problema. A Figura 5-13 ilustra uma dessas restrições aplicadas à contagem de função de dados. Nessa restrição, quando uma classe com estereótipo «internalPersistentEntity» ou «externalPersistentEntity» tem o atributo **fpFunctionType** com valor diferente de "nenhum", ou seja, é a classe principal de um agrupamento que representa uma função de dados da contagem, então o atributo **fpNumRLR** deve ser diferente de zero. Outro exemplo é a verificação da consistência da contagem. Se porventura um atributo incluído na lista de DER de alguma função de dados é excluído, isso causará um erro na validação do modelo.

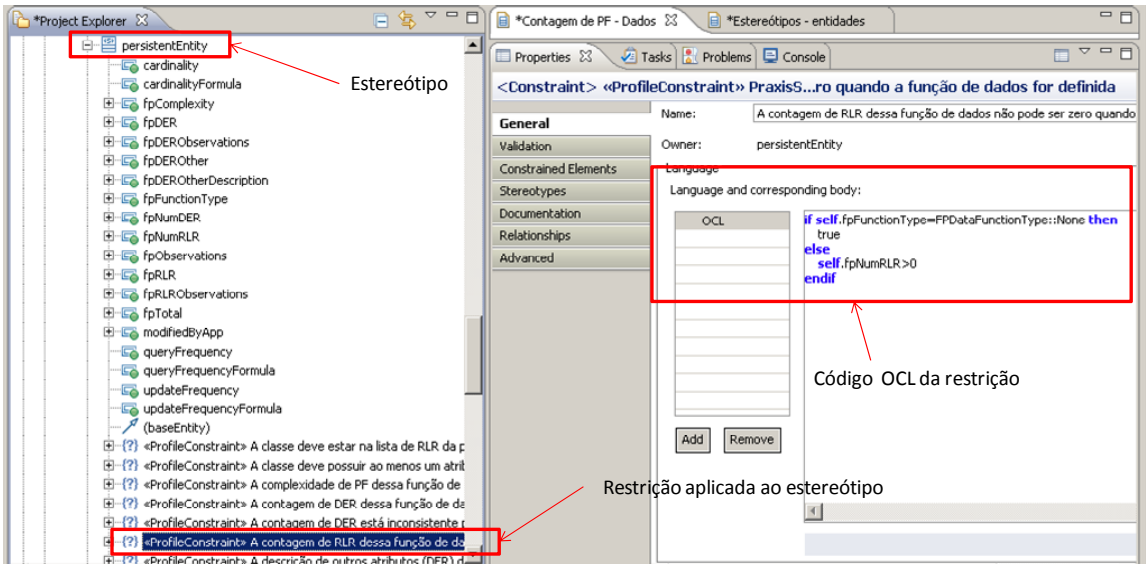

Figura 5-13 - Restrição OCL aplicada ao estereótipo utilizado na contagem de PF.

Os benefícios das restrições em termos de redução do retrabalho ainda não puderam ser medidos, já que sua primeira aplicação ainda está em progresso em um projeto executado pela Organização.

5.2.3.2 Contagem das funções de transação

Para a contagem das funções de transação, o mesmo arcabouço proposto na seção anterior foi desenvolvido. As funções de transação são mapeadas nos fluxos dos casos de uso do produto. Aqui, assim como na contagem de funções de dados, o mapeamento é manual e depende da interpretação de analista especializado. Nem todos os fluxos são mapeados em funções de transação.

Foi criado um estereótipo abstrato chamado «eventFlow» do qual os estereótipos para fluxos do Praxis 3.0 passam a ser filhos, como ilustrado na Figura 5-14.

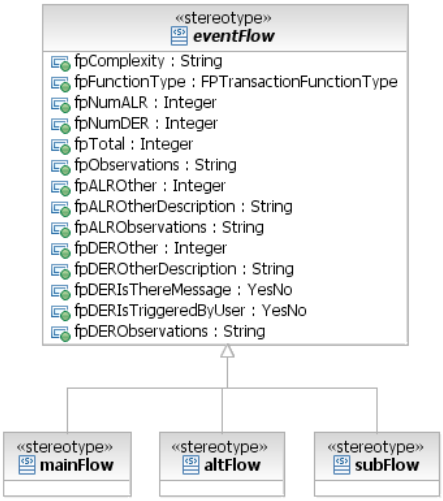

Figura 5-14 - Estereótipos para contagem de funções de transação.

Não apresentaremos aqui o detalhamento completo como feito para as funções de dados por ser extremamente semelhante, mas ressaltaremos as diferenças. A primeira delas é com relação à seleção dos itens que comporão a lista de ALR (Arquivos Lógicos Referenciados). Enquanto para funções de dados pode-se escolher quaisquer classes de entidade do modelo para compor a lista de RLR, nas funções de transação, quando o usuário abre a tela de seleção da lista de ALR, o *plugin* exibe apenas as funções de dados que participam da colaboração que realiza o caso de uso (ver seção 5.2.1 que detalha o modelo do problema). Isso garante que a lista de classes da colaboração se mantenha íntegra.

A segunda diferença é na seleção dos DER (Dados Elementares Referenciados). Nas funções de dados, escolhem-se atributos das classes agrupadas na lista de RLR. Aqui, a ferramenta permite ao usuário selecionar apenas atributos de classes de fronteira (que possuem estereótipos próprios) que participam da colaboração, que representam os campos das telas que participarão da transação. O analista também dispõe dos recursos necessários para contar ALRs e DERs que não sejam mapeados diretamente para classes e atributos, como por exemplo, a emissão de mensagens pela aplicação, que conta como 1 DER.

Da mesma forma como foi feito para a contagem de funções de dados, várias restrições OCL foram criadas para garantir a integridade da contagem e prevenir erros no preenchimento das informações.

5.2.4 A integração com o cadastro de requisitos

Na seção anterior foi apresentada a proposta para contagem de pontos de função diretamente no **Modelo do Problema**. A partir das informações preenchidas no modelo, a contagem de pontos de função é atualizada automaticamente no **Cadastro de Requisitos** do projeto, que é o artefato oficial da contagem. Para isso, o analista faz um mapeamento dos requisitos aos elementos de modelagem, como ilustrado na Figura 5-15.

O mapeamento de funções de dados e de transação para os elementos do **Modelo do Problema** não cobre todos os itens que são considerados em uma contagem. Para tratar esse problema, foi criado no **Cadastro de Requisitos** um tipo especial de requisito denominado **Requisito não-mapeado**, onde o analista poderá complementar a contagem de Pontos de função diretamente no Cadastro de requisitos para os itens que não puderam ser mapeados.

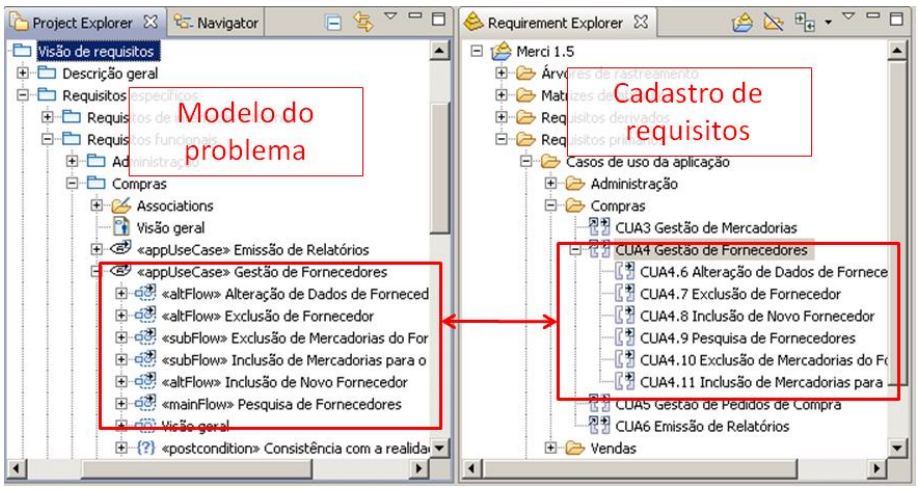

Figura 5-15 - Mapeamento entre elementos do Modelo do Problema e Cadastro de Requisitos.

Finalizado o procedimento de contagem, um relatório final com o seu detalhamento é emitido automaticamente a partir do **Cadastro de Requisitos** para ser consumido por clientes em um formato mais amigável e sem necessidade de instalação das ferramentas utilizadas pela Organização. Um trecho do relatório desenvolvido pode ser visto na Figura 5-16.

Além da integração entre o **Modelo do Problema** e **Cadastro de Requisitos** para fins de contagem de Pontos de função, o *plug-in* mencionado nas seções anteriores, cria, automaticamente, rastreabilidade entre os requisitos do **Cadastro de Requisitos** a partir de informações disponíveis no **Modelo do Problema.** Por exemplo, se uma determinada entidade persistente participa da colaboração que realiza um caso de uso, o *plug-in* cria a rastreabilidade desse caso de uso para a

entidade. Diversas regras como essa foram incluídas no *plug-in*. Com isso, a análise de impacto de alterações de requisitos torna-se mais fácil para os colaboradores.

Funções de dados

Nome	Tipo	RLR		DER		Complexidade	PF
		Cont.	Descrição	Cont.	Descrição		
Pedido de compra	ALI	2	Pedido de compra, Item do pedido de compra	7	Item de mercadoria (quantidade, valor total, preço unitário); Pedido de compra (número, data da compra, valor total, status)	Baixa	7
Mercadoria	ALI	1	Mercadoria	10	Mercadoria (código, data de cadastramento, descrição, estoque corrente, estoque mínimo, fabricante, modelo, preço de compra, preço de venda, unidade)	Baixa	7

Funções de transação

Nome	Tipo	ALR		DER		Complexidade	PF
		Cont.	Descrição	Cont.	Descrição		
Inclusão de novo fornecedor	EE	2	Fornecedor, Mercadoria	9	Tela de fornecedor (CPF/CNPJ, nome, endereço, telefone); Descritor de mercadoria (código, descrição); command; message	Média	4
Alteração de dados do fornecedor	EE	2	Fornecedor, Mercadoria	9	Tela de fornecedor (CPF/CNPJ, nome, endereço, telefone); Descritor de mercadoria (código, descrição); command; message	Média	4
Exclusão de fornecedor	EE	3	Fornecedor, Pedido de compra, Mercadoria	3	Tela de fornecedor (CPF/CNPJ); comando; mensagem	Média	4
Pesquisa de fornecedor	CE	1	Fornecedor	4	Tela de fornecedor (CPF/CNPJ, Nome); comando; mensagem	Baixa	3

Figura 5-16 - Relatório de contagem de PF gerado automaticamente.

5.2.5 O processo de contagem de Pontos de função de alteração

Como relatado na seção 4.2.2, um dos problemas diagnosticados na Organização é a manutenção da contagem de pontos de função dos produtos. Quando ocorrem alterações de requisitos durante o projeto, a respectiva contagem da alteração não é registrada e nem o tamanho atual do produto é mantido. Essa falha na coleta ocasionou algumas limitações nesse trabalho, como, por exemplo, o problema para obter a média de PF/LOC dos projetos da Organização, como descrito na seção 5.1.4.

A solução apresentada nas seções anteriores, para contagem de Pontos de função diretamente no **Modelo do problema** resolve parte do problema, que é manter a contagem final do produto atualizada. Essa contagem atualizada equivale à contagem de Pontos de função de aplicação do manual do IFPUG. No entanto, persiste o problema de contar os Pontos de função de alterações (procedimento de contagem para **projeto de melhoria**) que normalmente têm seu custo repassado ao cliente. Para ilustrar e melhorar a compreensão, um produto cujo **Modelo do problema** tenha registrado 100 PF sofreu alterações em seus requisitos e o levaram a uma nova contagem final de 110

PF. Entretanto, contando-se os Pontos de função de alteração, calcula-se 20 PF de alterações. Essa última contagem é a que deve ser cobrada do cliente e não os 10 PF da diferença entre as contagens finais.

A Organização já conta com um sistema de gestão de alterações, integrado com as demais ferramentas utilizadas na gestão de seus projetos. Ele foi implementado utilizando a ferramenta *IBM Rational ClearQuest*. O fluxo de trabalho para a gestão das alterações de requisitos segue uma máquina de estados, como mostrado no diagrama de estados da UML na Figura 5-17, onde os eventos que causam as transições de estado são comandos disparados no aplicativo. Resumidamente, quando uma alteração é registrada, ela é recebida pelo gerente do projeto que pode rejeitá-la ou designar alguém para análise. A pessoa designada inicia a análise do impacto, associando os requisitos que serão afetados e realizando estimativa de esforço para as alterações. Finalizada a análise, o Gerente aprova ou rejeita a alteração. Caso aprovada, ela segue para execução das alterações, que quando finalizadas, são verificadas por outro colaborador, que pode aceitar ou reenviar para correção de problemas.

Esse aplicativo funciona bem e traz informações importantes para a Organização (por exemplo, o esforço total gasto em alterações), mas tem a deficiência de não lidar com a contagem dos Pontos de função de alteração.

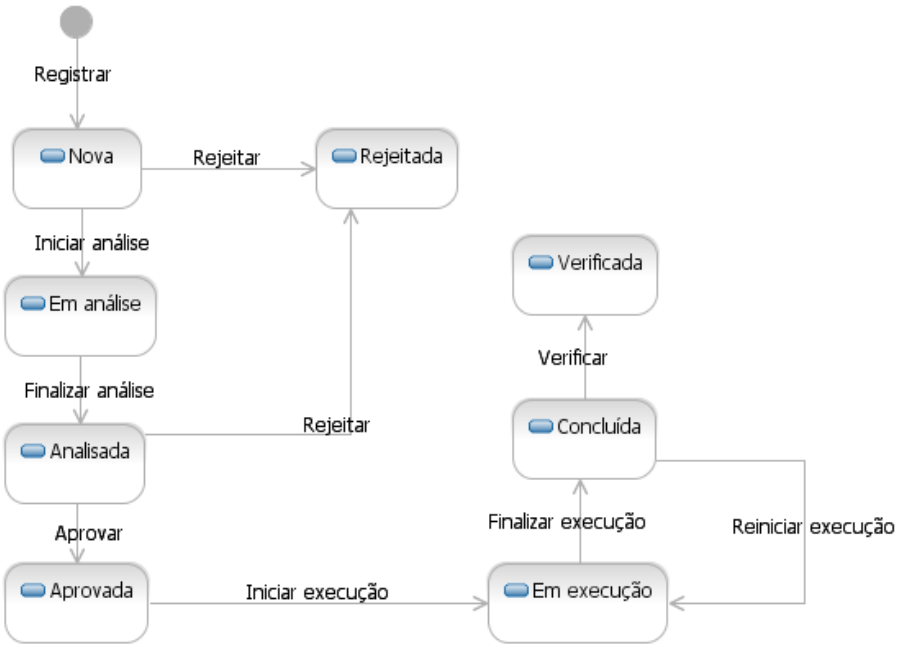

Figura 5-17 - Máquina de estados do sistema de gestão de alterações existente.

A solução proposta para contar os Pontos de função de alteração implicou em duas mudanças no aplicativo. A primeira é, no ato de finalização da análise, uma cópia dos requisitos associados e de todas as suas informações de contagem de Pontos de função, é armazenada junto ao registro da alteração. Essa cópia das informações é realizada antes da execução da alteração, quando ocorre a transição para o estado *Analisada*.

A segunda mudança foi a criação de um novo estado na máquina: *Contagem de PF realizada*. Nesse ponto do fluxo de trabalho, o **Modelo do problema** e o **Cadastro de requisitos** já tiveram a contagem final de Pontos de função atualizadas, além dos demais artefatos impactados já terem sido alterados, como por exemplo, **Modelo da solução**, **Especificação de testes** e **Código fonte**. Na transição para esse novo estado final, o especialista em contagem monta uma nova lista de requisitos associados à alteração, baseada na lista inicial, removendo os requisitos que foram excluídos, mantendo os que foram alterados e incluindo os novos. Comparando as duas listas, o aplicativo calcula automaticamente, conforme as regras do IFPUG, o total de Pontos de função de alteração e faz o registro dessa contagem no aplicativo. A Figura 5-18 exemplifica, de forma simplificada, como a contagem de Pontos de função de alteração pode ser obtida através da comparação das duas listas. Note que a contagem final não foi alterada, mantendo-se em 21 PF, mas a contagem da alteração registrou 28 PF.

Função de dado	Mudança	Situação atual					Situação anterior				
		Tipo	RLR	DER	Compl.	PF	Tipo	RLR	DER	Compl.	PF
Cotação eletrônica	Alteração funcional	ALI	3	11	Baixa	7	ALI	3	10	Baixa	7
Participação de fornecedor em cotação eletrônica	Inclusão	ALI	1	4	Baixa	7					
Lance de lote	Alteração funcional	ALI	2	11	Baixa	7	ALI	2	17	Baixa	7
Pregão presencial	Exclusão						ALI	2	17	Baixa	7

Pontos de função de alteração

Figura 5-18 - Exemplo de contagem de Pontos de função de alteração.

A contagem de Pontos de função de alteração definida pelo IFPUG é sujeita a questionamento, já que qualquer alteração em uma função de dados ou transação implica em contar a funcionalidade inteira novamente. A inclusão de um simples campo em uma tela faz com que a contagem dos Pontos de função de alteração equivalha ao desenvolvimento da funcionalidade do zero. Como exemplo, a função de dado *Cotação Eletrônica*, mostrada na Figura 5-18, teve um novo atributo incluído, mas o procedimento de contagem determina que sejam contados como se fosse uma

nova função de dados. Por esse motivo, a Organização adota o procedimento do *The Netherlands Software Metrics Users Association* (NESMA) [NESMA, 2001] para contar os Pontos de função de alteração.

O método do NESMA tenta corrigir essa distorção aplicando um fator de impacto na contagem final. Esse fator de impacto, que varia entre 0,25 a 1 para funções de dados e de 0,25 a 1,5 para funções de transação, é multiplicado pela contagem final de Pontos de função após as alterações. A contagem final é obtida pelo mesmo procedimento do IFPUG. Na Tabela 5-9 é apresentado um resumo do cálculo do fator de impacto para cada tipo de alteração nos requisitos. Note que no método do NESMA contam-se Pontos de função para alterações cosméticas, como a troca de um rótulo de um campo, ao contrário do procedimento do IFPUG que as ignora. Na Figura 5-19 o exemplo de alteração apresentado na Figura 5-18 é refeito utilizando o método do NESMA.

O procedimento de contagem pelo método do NESMA foi também incorporado no aplicativo de Gestão das alterações. A diferença é que o especialista em contagem deve informar dados adicionais para cada função alterada para aí sim, o aplicativo fazer o cálculo do número de Pontos de função de alteração.

Tabela 5-9 - Cálculo do fator de impacto para contagem de PF de alteração do método NESMA.

Tipo de função	Tipo de alteração	Fator de impacto
Dados	Inclusão	1
	Exclusão	0,4
	Alteração	Calcula-se a porcentagem de DERs alterados (incluídos + excluídos + modificados). Com base na porcentagem, consulta-se uma tabela que define o fator entre 0,25 até 1.
Transação	Inclusão	1
	Exclusão	0,4
	Alteração	Calcula-se a porcentagem de DERs e ALRs alterados (incluídos + excluídos + modificados). Com base nas duas porcentagens, consulta-se uma tabela que define o fator entre 0,25 até 1,5.
	Alteração cosmética	0,25

Função de dado	Mudança	% DER alterados	Fator de impacto	PF de alteração	Situação atual					Situação anterior				
					Tipo	RLR	DER	Compl.	PF	Tipo	RLR	DER	Compl.	PF
Cotação eletrônica	Alteração funcional	10%	0,25	1,75	ALI	3	11	Baixa	7	ALI	3	10	Baixa	7
Participação de fornecedor em cotação eletrônica	Inclusão	-	1	7	ALI	1	4	Baixa	7					
Lance de lote	Alteração funcional	35%	0,5	3,5	ALI	2	11	Baixa	7	ALI	2	17	Baixa	7
Pregão presencial	Exclusão	-	0,4	2,8						ALI	2	17	Baixa	7

Pontos de função de alteração

Figura 5-19 - Exemplo de contagem de PF de alteração pelo método do NESMA.

Com a proposta acima, a Organização resolveu as questões de registro da contagem de Pontos de função de alteração em seus projetos. Esse aplicativo está sendo testado já em um projeto atualmente em execução. A utilização do método do IFPUG ou do NESMA para contagem é definida junto ao cliente na contratação do projeto, mas o registro é realizado das duas formas para estudos futuros de correlação do tamanho da manutenção e o seu esforço, que atualmente é estimado pelo responsável na análise da alteração com base apenas em sua opinião.

5.3 O processo de planejamento macro dos projetos

Nessa seção serão endereçadas as questões relacionadas ao planejamento macro dos projetos, que consiste em estimar o perfil de equipe, esforço e duração total de cada iteração e fase do projeto, conforme o escopo previsto em cada uma. O Planejamento macro é realizado no início de cada fase do projeto. Após o planejamento macro, ao iniciar uma iteração, ocorre o planejamento detalhado, que será discutido na seção 5.4.

Durante a fase de diagnóstico deste programa de melhoria, foi identificado que o Planejamento macro dos projetos não segue um procedimento bem definido. Cada gerente planejava da forma que lhe era mais conveniente, mas foi identificado que a prática mais comum era a utilização de dados históricos para extrair a média de esforço para cada atividade do processo dada a complexidade do escopo, que era definido de forma qualitativa em baixa, média e alta. Para ilustrar os dados consultados, a Tabela 5-10 mostra um exemplo fictício.

A partir dos dados históricos, o gerente elaborava uma Estrutura Analítica do Projeto (EAP) baseada no processo documentado, já com o último nível de detalhamento das atividades, e

preenchia o esforço estimado para cada uma e o perfil de profissional que a executaria. As atividades que não estavam diretamente ligadas ao escopo do projeto, como a própria gestão do projeto, eram estimadas separadamente apenas na opinião do gerente. Esse procedimento era realizado utilizando a ferramenta *Microsoft Project*. Assim, eram obtidos o esforço total, equipe necessária e prazo de cada iteração e fase.

Tabela 5-10 - Exemplo de tabela utilizada no planejamento macro.

Tarefa	Complexidade		
	Alta	**Média**	**Baixa**
Elaborar desenho externo	40 h	32 h	24 h
Apoiar elaboração do desenho externo	8 h	6 h	6 h
Inspecionar usabilidade	6 h	4 h	4 h
Corrigir e verificar (usabilidade)	8 h	6 h	6 h
Inspecionar modelagem	8 h	6 h	4 h
Corrigir e verificar (modelagem)	10 h	8 h	6 h
Inspecionar mapeamento de análise	8 h	6 h	4 h
Corrigir e verificar (mapeamento)	10 h	6 h	6 h
Inspecionar testabilidade	6 h	6 h	4 h
Corrigir e verificar (testabilidade)	8 h	6 h	6 h
Especificar testes	32 h	32 h	16 h
Inspecionar especificação de testes	16 h	16 h	8 h
Corrigir e verificar	12 h	12 h	8 h

Esse procedimento foi executado em vários projetos não só para o planejamento macro das iterações, mas também para o planejamento detalhado, como relatado na seção 5.4. A diferença é que no planejamento macro são usados recursos genéricos, como: Programador Júnior, Analista de requisitos Sênior, etc., enquanto no planejamento detalhado, os colaboradores reais são utilizados. Outra diferença é que no planejamento detalhado os recursos devem ser nivelados para evitar super-alocação, ou seja, planejar mais horas do que a quantidade que o colaborador trabalha.

Para avaliar a acurácia desse procedimento, foi coletado de um grande projeto, o esforço estimado e o realizado das 4 atividades do processo de desenvolvimento que mais consomem esforço (dentre as planejadas com o procedimento descrito acima). Nessa coleta foram consideradas somente as atividades executadas sem alterações de escopo durante sua execução. Além disso, foram excluídas tarefas com esforço real pequeno: menor que 20h. O motivo é que em tarefas de pequeno esforço, as variações implicam em erros muito grandes. Por exemplo, se um esforço planejado de 3h é realizado em 2h, o MRE calculado é 50%. O resultado dessa análise é apresentado na Tabela 5-11.

Tabela 5-11 - Erros medidos no planejamento macro.

Tarefa	PRED(0,25)	MMRE	Int. Conf (95%)	Amostra
Codificar unidade	36,8%	52,82%	± 20,1%	39
Elaborar desenho externo	44,6%	34,79%	± 3,8%	92
Especificar testes de integração	48,8%	29,16%	± 4,5%	84
Executar testes de integração	66,7%	21,10%	± 5,3%	48

Pelos dados coletados podemos observar que o procedimento adotado para o planejamento macro apresenta uma acurácia baixa (embora não seja desastrosa). Um dos motivos prováveis é a classificação subjetiva da complexidade dos casos de uso.

Outro problema identificado é que o procedimento cobre apenas as atividades ligadas diretamente à produção do escopo do produto, ou seja, não considera atividades como reuniões, gestão do projeto e treinamentos. Talvez essa seja uma explicação para a grande variação no esforço total dos projetos identificada na seção 4.2.1, que indicou um MMRE de 45,9% na base histórica da Organização, já que esse procedimento adotado para o planejamento macro é similar ao procedimento usado para o orçamento de esforço dos projetos para elaboração das propostas.

Não foi possível, no escopo desse trabalho, realizar coletas para os erros da estimativa total das iterações. A Organização mantém um repositório detalhado no nível de Tarefas, mas informações consolidadas de iterações ficam armazenadas apenas nos cronogramas dos projetos. Isso dificulta a coleta automática via consultas ao banco de dados e implica em recuperar as várias versões do artefato para identificar qual foi realmente a primeira linha de base do planejamento macro das iterações. Esse problema será endereçado na seção 5.5.

Na tentativa de diminuir os problemas encontrados no procedimento adotado para o planejamento macro, serão utilizados os bons resultados colhidos na adoção do COCOMO (ver seção 5.1.3). Também será proposta uma simplificação no procedimento inacurado adotado atualmente, baseada no processo de planejamento macro do Praxis 3 padrão[10]. Por último, a proposta visa o planejamento macro para a utilização em projetos executados em ciclo de vida espiral, que passou a ser adotado pela Organização na atualização de seu processo de desenvolvimento para a versão 3 do Praxis.

A nova forma de realizar o planejamento macro consiste de quatro passos:

---

[10] O planejamento macro da Organização corresponde ao Planejamento Geral do Projeto no Praxis 3 padrão.

1. Obter o esforço total estimado para o projeto, utilizando o COCOMO.
2. A partir da lista de requisitos (casos de uso) identificados do produto, distribuí-los entre as iterações do projeto. O percentual de escopo em cada iteração, medido em Pontos de função não ajustados, indicará o percentual do esforço naquela iteração.
3. Coletar da base histórica o perfil de equipe necessário para o projeto. Essa informação é obtida pela distribuição do esforço do projeto para cada tipo de atividade. Cada atividade pertence a uma disciplina (ou fluxo) e é executada por um papel específico.
4. De acordo com a disponibilidade da equipe e pelas metas de prazo do cliente, estimar o prazo para cada iteração. É importante o gerente ter em mente os efeitos da deseconomia de escala existente em software [Boehm *et al.*, 2000].

Para melhorar o entendimento da proposta será apresentado um exemplo, com algumas informações extraídas do próprio exemplo do Praxis 3 padrão. Supondo um esforço total estimado pelo COCOMO de 11,9 Pessoas-mês (1.808,8 h), distribuímos esse esforço conforme a proporção do escopo (em Pontos de função não-ajustados) em cada iteração, como mostra a Tabela 5-12.

Tabela 5-12 - Exemplo de planejamento macro. Distribuição do esforço por iteração.

Iteração	Função (Caso de uso)	Tamanho (PF não-ajustados)	Fração do escopo	Esforço (h)
E1	Gestão de Usuários	26	13,6%	246,2
E2	Gestão de Mercadorias	30	15,7%	284,1
C1	Gestão Manual de Estoque, Gestão de Fornecedores	8 + 30	19,9%	359,9
C2	Operação de Venda, Abertura do Caixa, Fechamento do Caixa	14 + 14 + 4	16,8%	303,0
C3	Gestão de Pedidos de Compra	35	18,3%	331,5
C4	Emissão de Relatórios, Emissão de Nota Fiscal	20 + 10	15,7%	284,1
Total			100,0%	1808,8

A seguir recupera-se dos dados históricos a distribuição do esforço dos seus projetos nas disciplinas, que estão intimamente ligadas às equipes da Organização. A Tabela 5-13 ilustra como os dados devem ser recuperados. A partir desse ponto, essa proposta difere do procedimento apresentado no Praxis 3 padrão.

Tabela 5-13 - Exemplo de planejamento macro. Distribuição de esforço por disciplina.

Disciplina	Distribuição do esforço
Desenho & Impl.	56,1%
Eng. Processos	0,1%
Gestão de Alterações	0,3%
Gestão de projeto	11,4%
Qualidade	0,2%
Req. & Análise	2,9%
Testes	23,7%
Usabilidade	5,4%
Total geral	100,0%

A partir da Tabela 5-12 e da Tabela 5-13 o gerente será capaz de estimar os recursos necessários. Por exemplo, na iteração C1, será necessário um esforço de 85,3h (23,7% x 359,9h) de um analista de testes (papel responsável pelas atividades da disciplina de Testes). A partir daí, o prazo da iteração pode ser calculado conforme a disponibilidade/necessidade dos recursos. No caso de o prazo ser uma restrição do problema, a lógica inversa deve ser usada, estimando-se a quantidade de recursos necessários a partir do prazo fixado pelo cliente e do esforço estimado.

A proposta apresentada acima para o planejamento macro, baseada no Praxis 3 padrão, é relativamente mais simples que o procedimento atual adotado pelos gerentes. Acreditamos que ela é mais racional e sujeita a menos subjetividade (como a classificação da complexidade do escopo realizada sem critérios objetivos), já que é baseada apenas em dados históricos coletados durante a execução dos projetos da Organização. No entanto, é necessário validá-la durante a execução de um projeto real. Como não foi possível avaliar o erro total nas estimativas por iteração do procedimento atual, será realizado um piloto no próximo projeto executado pela organização utilizando os dois métodos e posteriormente comparando a acurácia de cada um para decidir pela mudança ou não no procedimento.

A proposta apresentada acima leva em consideração que todos os casos de uso de uma iteração serão levados ao estado 100 (Completo). Entretanto, para casos em que seja necessário realizar apenas parte do escopo de alguns casos de uso dentro de uma iteração, como levá-los do estado 20 (Detalhado) para o 40 (Desenhado), o mesmo raciocínio pode ser seguido. A informação extra necessária é a distribuição do esforço por estado do caso de uso e por disciplina, como ilustrado com dados fictícios na Tabela 5-14.

Tabela 5-14 - Exemplo de distribuição de esforço por estado e por disciplina.

Disciplina (Fluxo)	Estado										Total %
	10	20	30	40	50	60	70	80	90	100	
Requisitos	0,9%	5,5%	2,6%	0,1%	0,1%	0,1%	0,1%	0,1%	0,1%	0,0%	9,4%
Análise	0,0%	1,4%	6,9%	0,2%	0,1%	0,1%	0,1%	0,1%	0,0%	0,0%	8,8%
Desenho	0,0%	0,5%	0,7%	4,7%	4,4%	4,4%	4,4%	4,4%	0,5%	0,2%	24,3%
Testes	0,0%	0,0%	0,3%	1,9%	2,4%	2,4%	2,4%	2,4%	2,3%	2,9%	17,1%
Implementação	0,0%	0,0%	0,0%	5,9%	5,4%	5,4%	5,4%	5,4%	1,0%	0,3%	28,7%
Gestão do Projeto	0,8%	0,8%	0,8%	0,8%	0,8%	0,8%	0,8%	0,8%	0,8%	0,8%	8,0%
Usabilidade	0,1%	0,2%	0,3%	0,4%	0,1%	0,1%	0,1%	0,1%	0,2%	0,1%	1,6%
Gestão da Qualidade	0,0%	0,0%	0,0%	0,0%	0,1%	0,1%	0,1%	0,1%	0,3%	0,8%	1,6%
Eng. de Processos	0,0%	0,0%	0,1%	0,3%	0,0%	0,0%	0,0%	0,0%	0,1%	0,0%	0,5%
Total %	1,8%	8,4%	11,6%	14,3%	13,4%	13,4%	13,4%	13,4%	5,2%	5,0%	100,0%

5.4 Estimativas para o planejamento detalhado das iterações de um projeto

Nessa seção trataremos dos problemas apresentados na seção 4.2.4, que levanta os problemas relacionados com o planejamento detalhado das iterações. Conforme já dito anteriormente, a atividade de codificação dos casos de uso é a que consome mais esforço do volume total do projeto (aproximadamente 35%). O estudo apresentado na seção 5.3 mostra um MMRE de 52,8% quando se compara o esforço planejado e realizado.

Essas estimativas eram realizadas predominantemente pelos gerentes do projeto, com base no histórico de outros projetos. O gerente se valia do procedimento descrito na seção 5.3, utilizando uma tabela similar à ilustrada pela Tabela 5-10. A partir daí o gerente cria uma única tarefa de codificação para um determinado implementador.

Para tentar diminuir os erros das estimativas foi proposto um novo processo para o planejamento dessas tarefas. A primeira alteração foi com relação a ter uma única tarefa de codificação para cada caso de uso. Resolveu-se utilizar a técnica de decomposição e recomposição, como proposto pelo PMBoK [PMI, 2004]. Segundo McConnell [McConnell, 2006], essa técnica tira

proveito da “Lei dos grandes números”, onde os erros das estimativas das partes pequenas tendem a se anular (superestimativas cancelam subestimativas), produzindo erros totais menores. Dessa forma, o implementador designado para a codificação passou a propor uma divisão dessa tarefa em tarefas menores, com no máximo 40h. Para cada uma das tarefas deve ser informado uma série de itens que serão implementados: componentes de interface, regras de negócio e classes de entidade.

A segunda alteração foi o responsável pela estimativa, que passou a ser feita pelo próprio implementador que executará a codificação. McConnell [McConnell, 2006] afirma que essa deve ser a técnica preferida para estimativas no nível de tarefas de um projeto.

Para facilitar o lançamento dos dados foi criada uma planilha, ilustrada na Figura 5-20, que é preenchida pelo implementador. Após o preenchimento, a planilha é encaminhada ao gerente que criará as tarefas no cronograma do projeto.

Nome do Caso de us Gestão de especificações de anulação de despesa
Implementador: Wagner Moro Aioffi
Data: 11/3/2008

Nro	Tarefa	Componentes de interface	Principais Entidades	Pred.	Esforço
1	Implementar abas de consolidação	Aba consolidação por item de nota anulado; Aba consolidação por item de pedido	* Esssa tarefa, envolverá a implementação de métodos de consolidação no controle do caso de uso		24
2	Implementar outras abas	Tela cadastro de especificação de anulação de despesa; Aba dados; Aba Histórico; Aba itens de anulação de despesa	Especificação de anulação de despesa		24
3	Implementar cadastro de item de anulação	Tela de seleção de especificação de conformidade; Tela de seleção de item de conformidade	Especificação de item de anulação de despesa	2	24
4	Implementar gestão de rateio	Tela de gestão de rateios; Aba gestão de reteio por item de nota; Aba gestão de rateio por item de pedido;		2	16
5	Implementar cadastro de rateio por item de pedido	Tela de cadastro de rateio por item de pedido; Tela de seleção de item de nota do item de pedido	Rateio de anulação de despesa por item de pedido	4	40
6	Implementar cadastro de rateio por item de nota	Tela de cadastro de rateio por item de nota; Tela de seleção de item de nota do item de conformidade	Rateio de anulação de despesa por item de nota	4	40

Figura 5-20 - Planilha para apoio ao planejamento de tarefas de codificação de caso de uso

Para validar essa proposta, o novo processo foi aplicado como projeto piloto em um módulo de um dos projetos que estava em execução na Organização. Os resultados foram coletados para a codificação de 23 casos de uso. Para cada um deles foi montada uma planilha, ilustrada pela Tabela 5-15, que analisa os erros entre os esforços estimados e realizados. Os erros foram calculados utilizando-se o indicador MRE (ver seção 2.3, que trata dos indicadores de acurácia).

Tabela 5-15 - Coleta de dados para análise de erros de estimativas realizadas pelos implementadores

Caso de uso	Tarefas	Estimado	Realizado	MRE
CUD108 - Gestão de supressões de valor-quantidade em item de processo	Revisar entidades e seus relacionamentos	16	4,92	225,2%
	Criar regras de validação e testes unitários	30	7,98	275,9%
	Codificar controle e gestores	20	23	13,0%
	Codificar povoador	30	20,29	47,9%
	Codificar camada de fronteira	28	83,98	66,7%
	Integrar internamente ao CUD Gestão de alterações contratuais	8	8,69	7,9%
	Executar testes de implementação	8	12,11	33,9%
	Média dos erros (MMRE)			**95,8%**
	Total	**140**	**160,97**	**13,0%**

No gráfico da Figura 5-21 mostramos os resultados para os 23 casos de uso que utilizaram a nova proposta. Pode-se observar que em quase todos os casos, o erro total foi menor do que a média da magnitude dos erros relativos (MMRE) das tarefas individuais. Apenas para melhorar o entendimento, o ponto 6 do gráfico corresponde aos dados coletados na Tabela 5-15. Note também que a escala do eixo y do gráfico foi ajustada para mostrar no máximo até 150%, pois 3 casos de uso tiveram valores da média muito acima disso, o que distorceria a exibição.

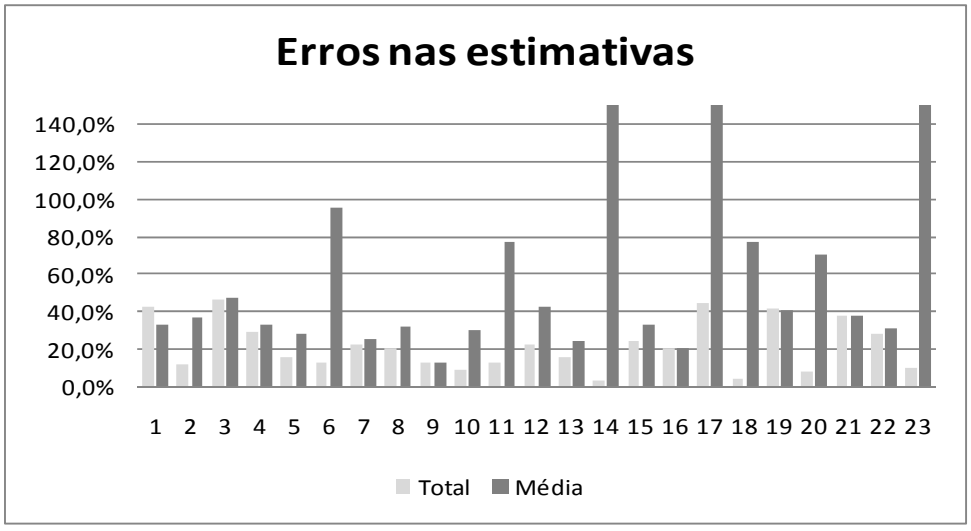

Figura 5-21 - Comparação do erro total por caso de uso e da média dos erros.

Calculando-se a média dos erros totais (barra da esquerda, mais clara, em cada ponto), chega-se a um MMRE de 21,8% ± 5,7% (com 95% de confiança). Esse resultado é significativamente

melhor do que o observado nos outros módulos do projeto, cujo MMRE foi de 52,8% para as tarefas de codificação dos casos de uso. Utilizando o indicador PRED(0,25) para avaliar esses dados, chega-se a 69,6%, em comparação com 36,8% medido nos outros módulos.

Outro fenômeno observado nesse projeto piloto foi a melhora das estimativas de implementadores que passaram por esse procedimento por mais de uma vez. Na Tabela 5-16 pode-se observar a redução do erro total das estimativas de dois implementadores que codificaram 3 casos de uso cada (dentre os 23 da amostra). Esse fato indica um potencial de melhora nos números apresentados no parágrafo anterior.

Tabela 5-16 - Melhora das estimativas dos implementadores.

Implementador	Erro total (MRE)		
	1ª estimativa	2ª estimativa	3ª estimativa
1	46,30%	15,50%	13,50%
2	44,80%	37,50%	28,80%

Diante dos resultados obtidos, esse procedimento de estimativa foi oficializado e incorporado ao processo de planejamento detalhado das iterações utilizado pela Organização. Ele passará a ser utilizado não só para as atividades de codificação, mas para todas as outras atividades que serão executadas em uma iteração.

5.5 O repositório centralizado de dados dos projetos

No relatório de diagnóstico apresentado no Capítulo 4, um dos problemas relatado foi a dificuldade de obtenção de informações dos projetos executados anteriormente. A Organização já utiliza o PSM [McGarry *et al.*, 2001] como método para definir, coletar e armazenar suas métricas, atendendo assim, à área de processo *Measurement and Analysis* (MA) do nível 2 do CMMI [CMMI, 2006]. Entretanto, os diversos dados coletados encontram-se em diversas fontes e formatos. Em alguns casos são mantidos em planilhas ou documentos texto, o que dificultou em muito a sua coleta, inclusive dos dados que serviram como base para as demais ações propostas neste trabalho.

Uma das ações propostas nesse mesmo relatório foi a construção de um repositório centralizado dos dados dos projetos, que servirá de base para as estimativas em projetos futuros. O grande desafio aqui é definir um formato no qual esses dados serão armazenados e automatizar, na

medida do possível, a coleta das informações das diversas fontes, já que as informações continuarão a estar disponíveis em aplicativos, bancos de dados, planilhas e documentos diversos que estão em constante evolução.

Existem no mercado, várias ferramentas que se propõem a solucionar esse problema, mas acreditar que um único produto será a panacéia para esse desafio não nos parece razoável, já que cada organização tem suas próprias características e especificidades. Um estudo realizado por Auer e outros [Auer *et al.*, 2003], que parece corroborar nossa hipótese, procurou avaliar diversas ferramentas de coleta e armazenamento de métricas e chegaram à conclusão de que a utilização de um aplicativo dessa natureza envolve um alto custo de personalização, intervenção manual do usuário e necessidade de conversão de dados sujeita a erro humano.

Olsina e outros [Olsina *et al.*, 2002] propuseram um repositório universal para métricas de projetos de desenvolvimento de software. A idéia apresentada consiste num banco de dados flexível e aderente a qualquer processo de desenvolvimento, pois ele se baseia no próprio conceito de processo, onde uma atividade produz e consome artefatos, gasta recursos e as métricas podem estar associadas a qualquer um desses elementos.

Outra proposta de repositório centralizado de métricas foi apresentada por Harrison [Harrison, 2004]. Segundo ele, a sua proposta, que consiste num modelo de classes UML, é flexível bastante para armazenar métricas de diversas naturezas.

Embora as propostas de Olsina e Harrison sejam realmente flexíveis para armazenamento dos dados, elas introduzem outro obstáculo: a recuperação da informação armazenada. Em ambas as propostas, recuperar a informação requer um alto grau de entendimento de como os dados estão armazenados e na confecção de consultas extremamente complexas ao banco de dados. O esforço para obtenção de uma informação nova, que não havia sido pensada previamente, implica na intervenção de um especialista para elaborar uma nova consulta que atenda a essa nova necessidade.

Em vista desses desafios, uma proposta que parece mais razoável, é a construção de um *Data Warehouse* utilizando modelagem dimensional, cujos principais defensores, e usualmente chamados de pais do DW, são Kimball [Kimball & Ross, 2002] e Inmon [Inmon, 2005]. O grande diferencial da modelagem dimensional, segundo os autores, é a facilidade da obtenção de informações, mesmo para perguntas que sequer tenham sido pensadas durante a modelagem do repositório. Por essa razão, a Organização decidiu adotar essa estratégia.

Os dois autores divergem sobre alguns pontos relacionados ao projeto do DW, embora o conceito principal de um repositório centralizado de fácil consumo seja a mesmo. A principal diferença entre as duas abordagens, como o próprio Inmon descreve em seu livro [Inmon, 2005] está

relacionada com o formato em que os dados ficam armazenados no DW. Inmon defende um modelo relacional normalizado enquanto Kimball defende um modelo multidimensional desnormalizado. A proposta de Inmon é mais flexível a mudanças dos requisitos de informação e torna o processo de extração dos dados das diversas fontes mais fácil, enquanto a proposta de Kimball é de mais fácil consumo pelos usuários e apresenta desempenho melhor para as consultas.

Como o foco do resultado da ação que apresentaremos nessa seção é o usuário final e pelo fato da Organização possuir uma Equipe de Engenharia de Processos dedicada, que entre suas atribuições está a gestão das métricas da Organização, resolveu-se adotar a metodologia proposta por Kimball.

Nas duas seções seguintes apresentaremos, respectivamente, uma introdução à modelagem dimensional de Kimball e o modelo que a Organização projetou para armazenar os dados de seus projetos, que futuramente fomentarão as estimativas. Vale ressaltar que a definição desse repositório não é o foco desse trabalho. O objetivo dessa ação é fornecer um repositório centralizado que atenda as principais demandas de informação para realizar as estimativas. Em um próximo ciclo de melhoria do ProMOTe a Organização poderá investir em melhores práticas para construção desse repositório.

5.5.1 *Data Warehouses* e a modelagem dimensional

Um *Data Warehouse* (DW), como o próprio nome diz, consiste no armazém de dados de uma organização, de onde a informação deve ser extraída. Ele se contrapõe aos sistemas transacionais, que são onde as informações são imputadas.

De acordo com Kimball e Ross [Kimball & Ross, 2002], alguns dos requisitos para um bom DW são:

- tornar os dados da organização de fácil acesso;
- apresentar a informação de forma consistente, ou seja, a informação recuperada deve ser confiável;
- deve ser adaptável e resiliente às mudanças dos processos da organização, sem ter que ser refeito a cada nova demanda;
- deve servir como base para melhores decisões dos gestores;

Como pode-se observar, é necessário não só um bom especialista em banco de dados, que conheça bem os repositórios dos aplicativos da organização, mas também uma boa estratégia para apresentação dos dados consolidados para os usuários, que não são, necessariamente, proficientes em informática. Com isso em mente, um DW foi dividido em componentes, ilustrados na Figura 5-22.

O primeiro componente consiste nas fontes de dados da organização, armazenados em diversos repositórios e formatos diferentes, como bancos de dados relacionais, documentos texto e planilhas eletrônicas. Muitas vezes, algumas informações estão duplicadas nos diversos aplicativos. Um exemplo típico é o cadastro dos funcionários, onde cada aplicativo mantém sua própria base de dados. Dessa forma, um especialista deve extrair as informações das diversas fontes e levá-la para o segundo componente, onde as informações são deduplicadas, padronizadas e organizadas. Essa área de tratamento da informação é de acesso restrito aos especialistas em banco de dados, ou seja, os usuários finais não acessam esse componente. Em seguida os dados são carregados no terceiro componente, que são os *Data Marts*, que armazenam a informação em formato dimensional. Esse processo de extração, transformação e carga dos dados é comumente chamado de ETL (*Extration Transformation Load*). A partir dos *Data Marts*, os usuários consultam as informações desejadas utilizando o aplicativo de sua preferência.

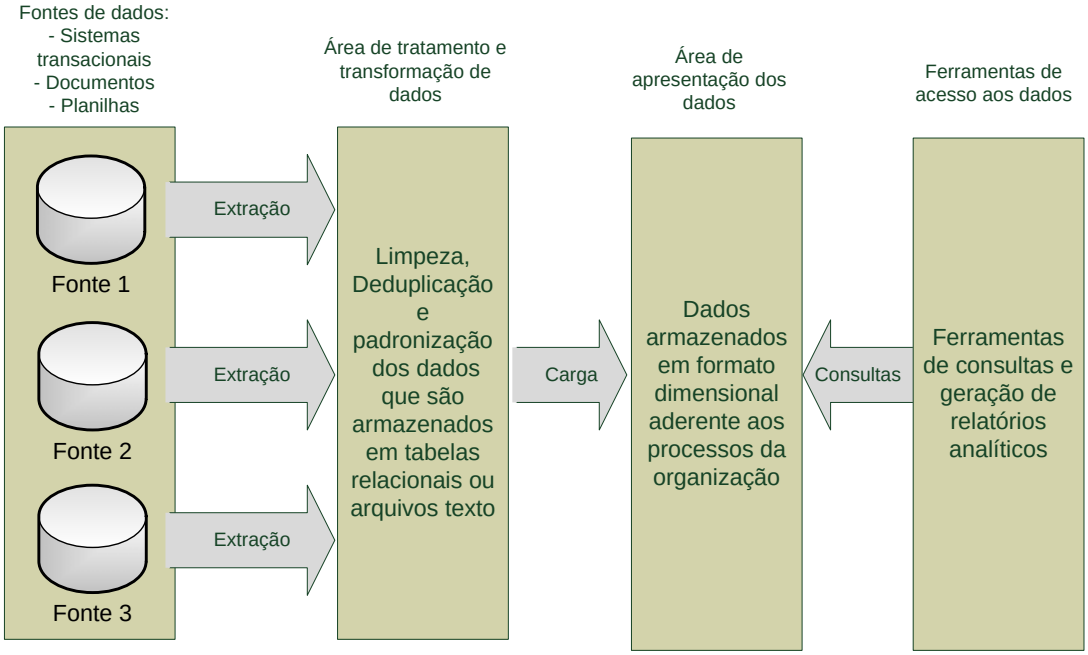

Figura 5-22 - Componentes de um DW. Adaptado de [Kimball & Ross, 2002].

A modelagem dimensional difere significativamente da modelagem relacional na

Terceira Forma Normal (3NF), embora ambas sejam armazenadas em bancos de dados relacionais. O objetivo da modelagem 3NF é eliminar a redundância dos dados, criando diversas entidades que relacionam entre si. A modelagem 3NF é extremamente útil para aplicativos transacionais, onde as informações são imputadas, pois a inclusão, atualização ou remoção de uma informação ocorre apenas em uma porção pequena dos dados, melhorando o desempenho dos aplicativos. Entretanto, modelos normalizados são tipicamente complicados demais para os usuários extraírem as informações que necessitam. Aí entra a modelagem dimensional, que propõem um armazenamento diferente para os dados, em poucas entidades, consistindo em um modelo extremamente desnormalizado, com a vantagem de apresentar os dados de forma mais compreensível aos usuários finais.

De forma simplificada, a modelagem dimensional divide a informação em dois tipos de tabela: Fato e Dimensão. As tabelas de fatos armazenam as medidas numéricas de desempenho dos processos da organização. Cada registro em uma tabela de Fato representa uma medida de algum fato que ocorreu durante a execução de um processo. O fato ocorre na interseção de uma ou mais dimensões, que representam a descrição textual do processo. Os atributos das tabelas de dimensão são utilizados para definir os relatórios e consultas que se deseja realizar no DW.

Para exemplificar, o processo de vendas de uma rede de lojas que vendem produtos quaisquer, poderia ser representado por um modelo dimensional como o da Figura 5-23. Note que o modelo é desnormalizado. A **Dimensão Data**, por exemplo, que contém um registro para cada dia do ano, duplica informações como Ano e Mês, que poderiam ser obtidas diretamente da data. A justificativa de Kimball é que dessa forma os usuários conseguem realizar consultas mais facilmente e com melhor desempenho.

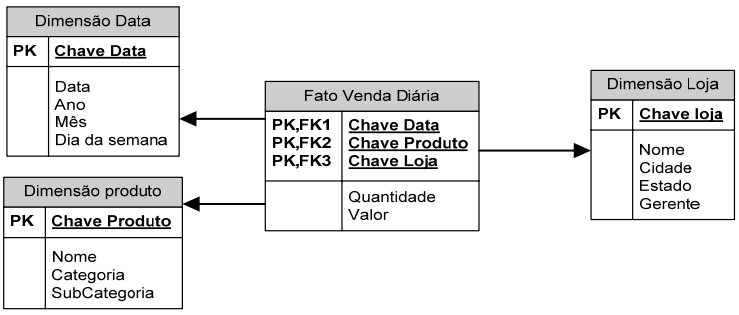

Figura 5-23 - Exemplo de modelo dimensional para uma rede de lojas.

5.5.2 O modelo dimensional da Organização

Nessa seção será apresentada a proposta de modelagem dimensional que suportará o armazenamento dos dados de projetos da organização. Como dito anteriormente, a Organização já havia construído um Plano de Medição utilizando o PSM. O foco desse plano foi a coleta de informações relacionadas ao retrabalho e à qualidade da produção dos artefatos.

Analisando o Relatório de Diagnóstico, pode-se observar que várias necessidades de informação para fomentar as estimativas não estão cobertas no Plano de Medição anterior. Então, foi realizada uma nova rodada do ciclo do PSM para identificar as medidas e indicadores que deveriam estar presentes no repositório centralizado. Os resultados, consolidando os dois planos de medição, são apresentados na Tabela 5-17. Foram omitidas as construções mensuráveis e medidas identificadas por limitação de espaço.

Tabela 5-17 - Necessidades de informação da Organização e indicadores propostos.

Necessidade de informação	Indicadores
Retrabalho no projeto	Distribuição do Total de Horas de Trabalho e Retrabalho por Pacote de trabalho
	Distribuição do Retrabalho
	Evolução do Percentual das Inspeções Aceitas por Módulo
	Evolução Mensal do Retrabalho
	Retrabalho Total Acumulado
Eficiência dos procedimentos de qualidade	Número de defeitos encontrados por esforço de revisão
	Número de Defeitos não detectados por Inspeção
Qualidade dos artefatos produzidos pela equipe	Número de Defeitos Detectados por Inspeção e por Gravidade
	Evolução do número de defeitos detectados por artefato
	Evolução do número de defeitos abertos e corrigidos
	Número de defeitos detectados pelo Cliente
Distribuição do esforço	Distribuição do esforço por estado do caso de uso
	Distribuição do esforço por artefato
	Distribuição do esforço por atividade do processo
Acompanhamento de custo do projeto	Valores adquirido, planejado e real do custo atual do projeto.
	Valores adquirido, planejado e real do custo atual do projeto por iteração.
Progresso do escopo	Evolução do tamanho do produto em Pontos de função
	Evolução do número de Pontos de função de alteração
	Número de casos de uso por estado alcançado
	Evolução do número de LOC por categoria (conforme padrão de contagem).

Cumprimento do prazo	Prazo planejado e realizado por iteração
	Prazo planejado e realizado por atividade
Estabilidade do produto	Evolução do número de alterações por Pacote de trabalho
	Número de requisitos incluídos, alterados e excluídos
Acurácia das estimativas	MMRE por atividade do processo
	PRED(0,25) por atividade do processo
Produtividade da equipe	Número de horas por Ponto de Função
	Número de horas por Ponto de Função de alteração
	Custo por Ponto de Função

De posse dos indicadores e medidas identificados, o desafio agora é propor um modelo dimensional que comporte esses dados de forma a tornar sua recuperação simples e ágil pelos interessados. Um trabalho similar ao que está sendo proposto aqui foi apresentado por Becker e outros [Becker *et al.*, 2006]. Eles também partiram de uma lista de medidas básicas e derivadas e propuseram um modelo dimensional para armazená-las. A diferença é que os autores propuseram uma estrutura intermediária que representa um metamodelo de um processo de desenvolvimento de software. Porém, não é detalhado como é feita a transformação do metamodelo para o modelo dimensional que propuseram.

A proposta que será apresentada abaixo, se valeu da proposta de Becker e outros [Becker *et al.*, 2006], mas foi necessário realizar vários ajustes para atender as necessidades de informação da Organização, que são diferentes das necessidades apresentadas no trabalho deles.

Kimball e Ross [Kimball & Ross, 2002] sugerem uma seqüência de quatro passos para auxiliar na modelagem dimensional:

- Escolher um processo da organização para ser modelado. A partir do processo podemos derivar os fatos que ocorrem durante a execução do processo.
- Declarar a granularidade dos fatos identificados. Isso significa explicitar o que cada um dos registros da Tabela Fato representa.
- Enumerar quais dimensões se aplicam a cada registro de cada fato. Corresponde a identificar os atributos de caracterização daquele fato.
- Identificar as medidas numéricas que podem ser coletadas em cada registro do fato.

Embora os passos acima auxiliem na racionalização da modelagem, o resultado está sempre sujeito à criatividade do modelador frente aos requisitos dos usuários. Outra pessoa com os mesmos requisitos de informação em mente poderia produzir um modelo diferente.

O processo escolhido para modelagem foi o próprio processo de desenvolvimento da Organização. Dividir esse processo em processos menores poderia levar a modelos difíceis de integrar no futuro.

Os fatos identificados durante a modelagem, já com a declaração de granularidade, são apresentados na Tabela 5-18.

Tabela 5-18 - Fatos identificados durante a modelagem dimensional.

Fato	Descrição/Granularidade
Trabalho	Consiste na execução de alguma atividade por um colaborador em uma determinada data.
Defeito	Representa uma transação realizada em relação a um defeito, como detecção, correção, verificação, etc.
Tamanho em PF	É a medida de tamanho em Pontos de função não-ajustados de um determinado requisito em um determinado dia.
Tamanho em LOC	É a medida de tamanho em Linhas de código de um determinado pacote de trabalho de uma determinada classificação em um determinado dia.
Alteração	É o registro de uma alteração de escopo do produto.
Progresso	Contém o registro diário do progresso do projeto, analisando os indicadores BCWS, BCWP, ACWP sugeridos no padrão do PMBoK [PMI, 2004] para acompanhamento do projeto.

A partir da Tabela 5-18, extraiu-se o modelo dimensional apresentado nas Figuras abaixo, que mostram o modelo lógico do banco de dados. Os atributos das dimensões são apresentados apenas no primeiro modelo em que ela aparece, para facilitar a leitura. Nos modelos, já constam as medidas numéricas relacionadas a cada fato. As tabelas cujo nome inicia com **Dim** representam as dimensões e as iniciadas por **Fact** representam os Fatos. Um detalhamento completo dos atributos de cada uma das dimensões pode ser visto no Apêndice A.2.

Uma das questões endereçadas na modelagem dimensional proposta foi o fornecimento da informação de custo dos projetos. Como relatado no Capítulo 4, os gerentes de projeto da Organização não tinham acesso ao custo individual dos colaboradores por questões de confidencialidade. A solução proposta foi criar uma classificação dos cargos da Organização de acordo com a equipe em que trabalha e com a sua experiência (Implementador I, Implementador II, Testador I, Testador II, etc.). Para cada categoria foi calculada uma média do valor hora dos colaboradores dessa categoria. Essa média é utilizada para calcular o custo de execução das tarefas, que fica armazenado na tabela Fato Trabalho (FactTrabalho) como mostra a Figura 5-24. Dessa forma os gerentes passam a planejar e acompanhar o custo em seus projetos. Essas categorias e respectivo histórico de custo médio são mantidos no banco de dados do aplicativo Apropriação de horas (ver

seção 4.2.2), com acesso restrito aos gerentes da Organização.

Outra ação proposta que também foi tratada durante a elaboração do modelo dimensional foi o armazenamento dos dados das propostas. Na Dimensão Projeto (DimProjeto) ficam armazenados o custo, esforço e prazo que foram considerados para a elaboração da proposta apresentada ao cliente. Entretanto, vale ressaltar que esses valores consistem no orçamento interno, sem a margem de lucro aplicada na proposta comercial, já que essa é uma informação confidencial e de acesso restrito à diretoria da Organização.

O processo de recuperação, transformação e carga dos dados, como era esperado, foi uma tarefa complexa, devido às várias origens, duplicidade e falta de padronização dos dados. Várias tabelas auxiliares tiveram que ser confeccionadas para mapear entidades entre sistemas diferentes. Apenas para ilustrar, a Organização possui 2 sistemas para registro de defeitos, sendo que um deles é usado apenas para defeitos encontrados no código executável. Cada um dos sistemas possui o registro de requisitos separado e pequenas variações nos nomes digitados forçou a criação de uma tabela que mapeasse os requisitos entre os dois sistemas. A quantidade de particularidades tais como essa foi enorme e por serem específicas da Organização e seus sistemas, não será relatado aqui todo o processo de carga do modelo dimensional proposto.

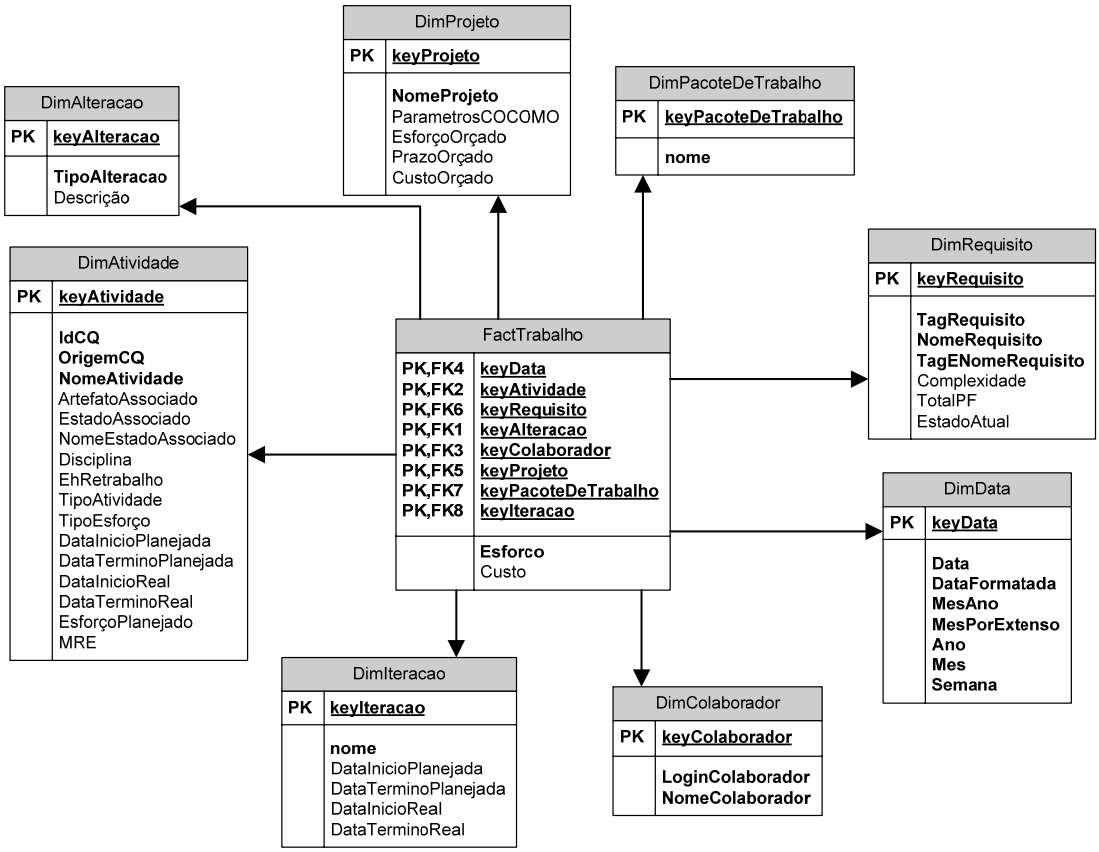

Figura 5-24 - Fato Trabalho e Dimensões relacionadas.

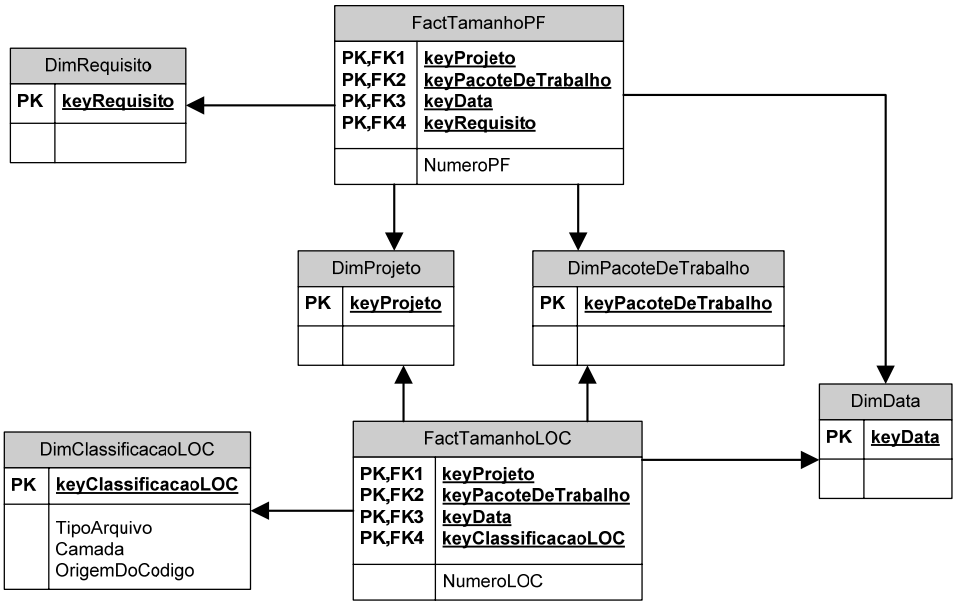

Figura 5-25 – Fatos de Tamanho (PF e LOC) e Dimensões relacionadas.[11]

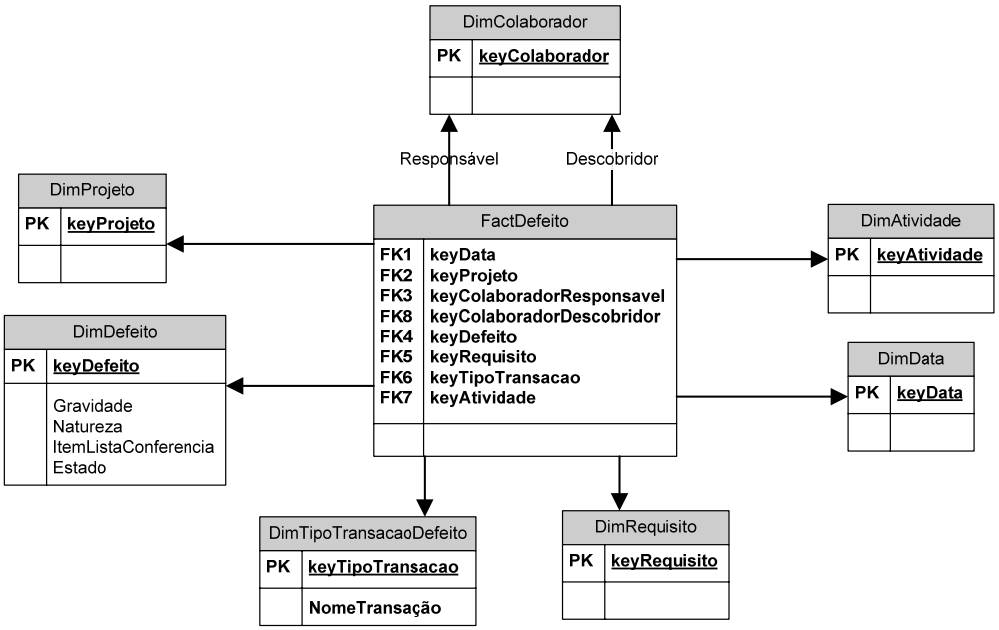

Figura 5-26 - Fato Defeito e Dimensões relacionadas.

[11] A Dimensão Classificação LOC, na verdade, consiste de uma dimensão para cada vetor de dados definido pela Organização. Ver o apêndice A.1.

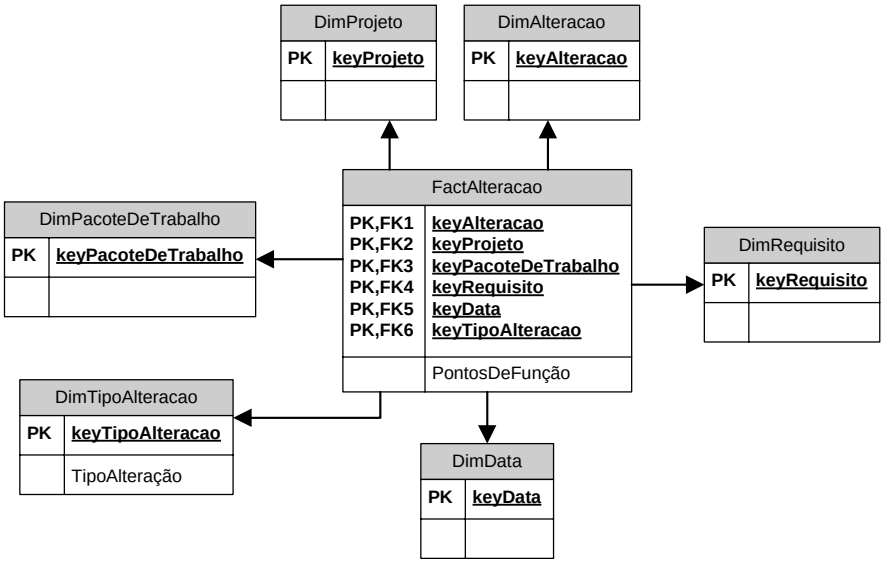

Figura 5-27 - Fato Alteração e Dimensões relacionadas.

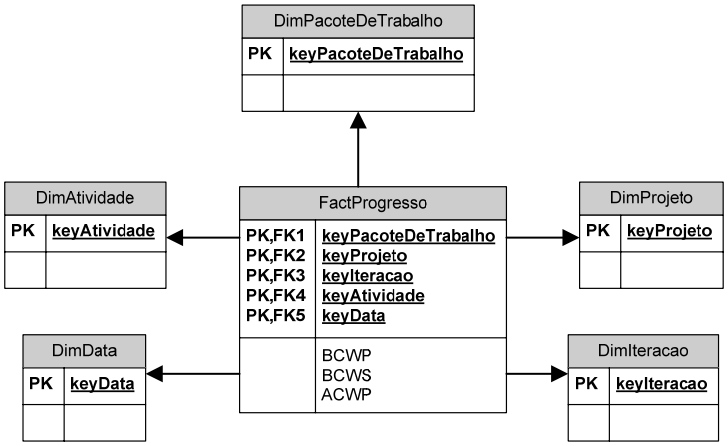

Figura 5-28 - Fato Progresso e Dimensões relacionadas.

Para validar o modelo apresentado acima foi realizada uma carga com dados de um grande projeto da Organização que já havia sido encerrado. Então, tentou-se obter os indicadores enumerados na Tabela 5-17 e foi possível emitir relatórios de quase todos eles, com exceção dos indicadores baseados na contagem de Pontos de função de alteração, já que ainda não foram coletados em nenhum projeto. Alguns desses relatórios são apresentados nas figuras abaixo. Eles foram

elaborados através do recurso de tabela dinâmica do *Microsoft Excel*, conectando diretamente no servidor do DW.

	Retrabalho	
Disciplina	Não	Sim
Desenho & Impl.	63,540%	36,460%
Eng. Processos	100,000%	0,000%
Gestão de Alterações	92,848%	7,152%
Gestão de projeto	98,309%	1,691%
Qualidade	58,605%	41,395%
Req. & Análise	82,741%	17,259%
Testes	79,054%	20,946%
Treinamento	100,000%	0,000%
Usabilidade	82,465%	17,535%
Total geral	**75,893%**	**24,107%**

Figura 5-29 - Distribuição de esforço de trabalho e retrabalho entre as Disciplinas

	Estado											
Disciplina	Geral	10	20	30	40	50	60	70	80	90	100	Total geral
Desenho & Impl.	3,780%	0,000%	0,000%	0,000%	5,327%	0,000%	26,469%	0,000%	13,247%	0,000%	0,000%	48,822%
Eng. Processos	0,014%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,014%
Gestão de Alterações	0,294%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,294%
Gestão de projeto	10,936%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	10,936%
Qualidade	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,070%	0,070%
Req. & Análise	0,526%	0,759%	1,963%	0,024%	1,012%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	4,284%
Testes	5,584%	0,000%	0,000%	0,000%	0,742%	5,353%	0,024%	1,915%	3,598%	3,800%	0,000%	21,016%
Treinamento	9,243%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	0,000%	9,243%
Usabilidade	2,025%	0,273%	0,236%	0,000%	0,968%	0,000%	0,476%	1,343%	0,000%	0,000%	0,000%	5,321%
Total geral	**32,401%**	**1,032%**	**2,199%**	**0,024%**	**8,048%**	**5,353%**	**26,969%**	**3,259%**	**16,845%**	**3,800%**	**0,070%**	**100,000%**

Figura 5-30 - Distribuição de esforço entre Disciplinas e Estados

A proposta acima foi apresentada e aprovada pelos os diretores e gerentes da Organização. Durante as apresentações surgiram algumas novas demandas e melhorias. O modelo proposto mostrou-se altamente resiliente às solicitações: com pouco esforço foi possível atendê-las.

Capítulo 6

Conclusão

O trabalho apresentado aqui relatou a execução de um programa de melhoria nas estimativas dentro de uma organização desenvolvedora de software. O programa foi realizado seguindo um processo bem definido, o ProMOTe, e abordou as estimativas realizadas em todas as fases do ciclo de vida dos projetos de desenvolvimento de novos produtos. A partir de um diagnóstico e de um levantamento bibliográfico sobre os temas relacionados aos problemas encontrados foi possível definir várias ações de melhoria dos processos da Organização.

Dentre as melhorias propostas, algumas foram validadas em projetos piloto e já foram incorporadas aos processos da Organização, como a técnica de estimativa para o planejamento detalhado das iterações e a utilização do COCOMO como método de estimativa para elaboração das propostas. Outras melhorias foram colocadas em prática, mas ainda não foi possível medir seu benefício quantitativamente, como a proposta para o planejamento macro e a contagem de Pontos de função diretamente no Modelo do Problema. Entretanto, ambas as propostas tiveram *feedback* positivo dos colaboradores da Organização.

A contagem de Pontos de função não-ajustados realizada diretamente no Modelo do problema mostrou três benefícios: 1) a consistência entre o modelo e a contagem, já que não há mais artefatos separados; 2) e informação do tamanho final do produto, informação muito importante para as propostas apresentadas ao longo desse trabalho; 3) a diminuição de erros no procedimento de contagem garantida pelas diversas verificações realizadas pelas restrições OCL criadas e pela prevenção de erros na interface gráfica do plugin desenvolvido.

A criação de um repositório centralizado de medidas dos projetos vai proporcionar, não só o embasamento dos processos de estimativas propostos nesse trabalho, mas também vai fornecer informações gerenciais importantes para o acompanhamento dos projetos e para as decisões de melhoria de processos da Organização.

Além dos benefícios medidos e esperados para a Organização alvo desse programa, o trabalho apresentado aqui pode servir de base para outras organizações que desejem melhorar suas estimativas. Elas podem se valer tanto das propostas e técnicas apresentadas, quanto do formato do programa de melhoria descrito, que ajuda a racionalizar e melhorar as chances de sucesso da alteração de seus processos.

6.1 Trabalhos futuros

A última fase do ProMOTe, do registro de lições aprendidas, não foi executada no escopo deste trabalho e fica como trabalho futuro para a Organização, para que tenha melhor proveito de novos ciclos de melhoria.

As diversas ações de melhoria propostas nesse trabalho cobrem uma quantidade muito grande de aspectos relacionados às estimativas em projetos de desenvolvimento de software. Por limitação de tempo, não foi possível explorar de forma detalhada cada um desses aspectos, já que cada um deles poderia por si só, render um trabalho do porte deste.

Com relação às estimativas para elaboração das propostas, quando é definido um esforço, prazo e custo globais para um projeto, diversas outras técnicas podem ser avaliadas futuramente pela Organização. Para cada tipo de método definido pela ontologia proposta por Myrtveit e outros [Myrtveit *et al.*, 2005] para classificar os métodos de estimativa, há uma grande quantidade de trabalhos relacionados. No entanto, alguns desses trabalhos carecem de uma quantidade de dados maior do que a base de dados histórica da Organização, principalmente os métodos de Aproximação de Função Arbitrária, onde se encaixam as técnicas de inteligência artificial, como redes neurais e redes *neuro-fuzzy*.

Embora não haja trabalhos que mostrem a viabilidade de automatizar completamente a contagem de Pontos de função a partir de um modelo UML, pode-se investir em alguma automação parcial. No entanto, deve-se avaliar se uma automação parcial acarretará em mais trabalho de conferência dos resultados produzidos do que na própria contagem manual.

A proposta para o planejamento macro das iterações do projeto apresentada não foi validada. É preciso comparar a sua eficácia com o procedimento atual executado pela organização para, a partir dessa análise, tomar a decisão de qual método deve ser utilizado. Em um novo ciclo do ProMOTe essa questão deve ser revisitada. Também devem ser investigadas outras formas de abordar

o problema do planejamento das iterações dos projetos e, além disso, um trabalho interessante que pode ser realizado é como integrá-lo à gestão do portfólio de projetos da organização e a alocação dos recursos nos diversos projetos.

Já na proposta de repositório centralizado de medidas, pode-se investir em meios de facilitar a extração e transformação dos dados dos diversos sistemas para carga no DW, através da utilização de metamodelos e modelos mais genéricos. Uma pesquisa bibliográfica sobre o tema revelou poucos trabalhos nesse sentido.

Por último, a Organização poderá avaliar a aplicabilidade dos métodos utilizados para estimativa em projetos de desenvolvimento de novos produtos para outros tipos de projetos, como modelagem de negócio, evolução e manutenção de produtos.

Referências bibliográficas

[Aggarwal *et al.*, 2005] AGGARWAL, K. K., SINGH, YOGESH, CHANDRA, PRAVIN, & PURI, MANIMALA. 2005. Evaluation of various training algorithms in a neural network model for software engineering applications. *SIGSOFT Softw. Eng. Notes*, **30**(4), 1–4.

[Anda *et al.*, 2009] ANDA, BENTE C.D., SJØBERG, DAG I.K., & MOCKUS, AUDRIS. 2009. Variability and Reproducibility in Software Engineering: A Study of Four Companies that Developed the Same System. *Software Engineering, IEEE Transactions on*, **35**.

[Arnold & Pedross, 1998] ARNOLD, MARTIN, & PEDROSS, PETER. 1998. Software size measurement and productivity rating in a large-scale software development department. *Pages 490–493 of: ICSE '98: Proceedings of the 20th international conference on Software engineering*. Washington, DC, USA: IEEE Computer Society.

[Auer *et al.*, 2003] AUER, M., GRASER, B., & BIFFL, S. 2003 (Sept.). A survey on the fitness of commercial software metric tools for service in heterogeneous environments: common pitfalls.

[Barry *et al.*, 2002] BARRY, E. J., MUKHOPADHYAY, T., & SLAUGHTER, S. A. 2002. Software Project Duration and Effort: An Empirical Study. *Information Technology and Managemen, 2002.*

[Becker *et al.*, 2006] BECKER, KARIN, RUIZ, DUNCAN D., CUNHA, VIRGINIA S., NOVELLO, TAISA C., & E SOUZA, FRANCO VIEIRA. 2006 (April). SPDW: A Software Development Process Performance Data Warehousing Environment.

[Boehm, 1981] BOEHM, BARRY W. 1981. *Software Engineering Economics*. Prentice Hall PTR.

[Boehm *et al.*, 2000] BOEHM, BARRY W., ABTS, CHRIS, BROWN, A. WINSOR, CHULANI, SUNITA, CLARK, BRADFORD K., HOROWITZ, ELLIS, MADACHY, RAY, REIFER, DONALD J., & STEECE, BERT. 2000. *Software Cost Estimation with COCOMO II*. Prentice Hall PTR.

[Boehm & Valerdi, 2008] BOEHM, B.W., & VALERDI, R. 2008. Achievements and Challenges in Cocomo-Based Software Resource Estimation. *Software, IEEE*, **25**(5), 74–83.

[Caldiera *et al.*, 1998] CALDIERA, G., ANTONIOL, G., FIUTEM, R., & LOKAN, C. 1998 (Nov).

Definition and experimental evaluation of function points for object-oriented systems. *Pages 167–178 of: Proceedings. Fifth International Software Metrics Symposium, 1998. Metrics 1998.*

[Cantone *et al.*, 2004] CANTONE, G., PACE, D., & CALAVARO, G. 2004 (Sept.). Applying function point to Unified Modeling Language: conversion model and pilot study. *Pages 280–291 of: 10th International Symposium on Software Metrics, 2004. Proceedings.*

[Chen *et al.*, 2005] CHEN, Z., MENZIES, T., PORT, D., & BOEHM, D. 2005. Finding the right data for software cost modeling. *Software, IEEE*, **22**(6), 38–46.

[Clark *et al.*, 1998] CLARK, B., DEVNANI-CHULANI, S., & BOEHM, B. 1998 (19-25 April). Calibrating the COCOMO II Post-Architecture model. *Pages 477–480 of: Software Engineering, 1998. Proceedings of the 1998 (20th) International Conference on.*

[CMMI, 2006] CMMI. 2006. *CMMI for Development, Version 1.2*. Tech. rept. CMU/SEI-2006-TR-008. Software Engineering Institute / Carnegie Mellon University.

[Foss *et al.*, 2003] FOSS, T., STENSRUD, E., KITCHENHAM, B., & MYRTVEIT, I. 2003. A simulation study of the model evaluation criterion MMRE. *Software Engineering, IEEE Transactions on*, **29**(11), 985–995.

[Friedl, 2006] FRIEDL, JEFFREY E. F. 2006. *Mastering Regular Expressions*. 3rd edition. O'Reilly.

[Gencel & Demirors, 2008] GENCEL, CIGDEM, & DEMIRORS, ONUR. 2008. Functional size measurement revisited. *ACM Trans. Softw. Eng. Methodol.*, **17**(3), 1–36.

[Gremba & Myers, 1997] GREMBA, JENNIFER, & MYERS, CHUCK. 1997. The IDEAL(SM) Model: A Practical Guide for Improvement. *Software Engineering Institute (SEI)*. Tech Report SEL-95-102.

[Harput *et al.*, 2005] HARPUT, V., KAINDL, H., & KRAMER, S. 2005 (Sept.). Extending function point analysis to object-oriented requirements specifications. *Pages 10 pp.–39 of: Software Metrics, 2005. 11th IEEE International Symposium.*

[Harrison, 2004] HARRISON, WARREN. 2004. A flexible method for maintaining software metrics data: a universal metrics repository. *Journal of Systems and Software*, **72**(2), 225 – 234.

[Helm, 1992] HELM, J.E. 1992. The viability of using COCOMO in the special application software bidding and estimating process. *Engineering Management, IEEE Transactions on*, **39**(1), 42–58.

[IEEE, 1992] IEEE. 1992. *IEEE Standard for Software Productivity Metrics*. Tech. rept. IEEE Std 1045-1992.

[IEEE, 1998] IEEE. 1998. *IEEE Recommended Practice for Software Requirements Specifications*. Tech. rept. IEEE Std 830-1998.

[IFPUG, 2005] IFPUG. 2005. *IFPUG Counting Practices Manual - Release. 4.2.1*. International Function Point Users Group, Westerville, OH.

[Inmon, 2005] INMON, WILLIAN H. 2005. *Building the Data Warehouse*. Fourth edn. Wiley Publishing, Inc.

[Jain, 1991] JAIN, RAJ. 1991. *Art of Computer Systems Performance Analysis Techniques For Experimental Design Measurements Simulation And Modeling*. Wiley Computer Publishing.

[Jorgensen & Shepperd, 2007] JORGENSEN, M., & SHEPPERD, M. 2007. A Systematic Review of Software Development Cost Estimation Studies. *Software Engineering, IEEE Transactions on*, **33**(1), 33–53.

[Jørgensen & Moløkken-Østvold, 2006] JØRGENSEN, MAGNE, & MOLØKKEN-ØSTVOLD, KJETIL. 2006. How large are software cost overruns? A review of the 1994 CHAOS report. *Information and Software Technology*, **48**(4), 297 – 301.

[Kautz *et al.*, 2000] KAUTZ, KARLHEINZ, HANSEN, HENRIK WESTERGAARD, & THAYSEN, KIM. 2000. Applying and adjusting a software process improvement model in practice: the use of the IDEAL model in a small software enterprise. *Pages 626–633 of: ICSE '00: Proceedings of the 22nd International Conference on Software Engineering*. New York, NY, USA: ACM Press.

[Kimball & Ross, 2002] KIMBALL, RALPH, & ROSS, MARGY. 2002. *The Data Warehouse Toolkit*. Second edition. Wiley Computer Publishing.

[Kitchenham *et al.*, 2001] KITCHENHAM, B.A., PICKARD, L.M., MACDONELL, S.G., & SHEPPERD, M.J. 2001. What accuracy statistics really measure [software estimation]. *Software, IEE Proceedings* -, **148**(3), 81–85.

[Kohavi, 1995] KOHAVI, RON. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. Morgan Kaufmann.

[Laird, 2006] LAIRD, L.M. 2006. The Limitations of Estimation. *IT Professional*, **8**(6), 40–45.

[Lokan, 2000] LOKAN, C. J. 2000. An empirical analysis of function point adjustment factors. *Information and Software Technology*, **42**(9), 649 – 659.

[Lopez-Martin *et al.*, 2006] LOPEZ-MARTIN, CUAUHTEMOC, YANEZ-MARQUEZ, CORNELIO, & GUTIERREZ-TORNES, AGUSTIN. 2006. A Fuzzy Logic Model Based upon Reused and New & Changed Code for Software Development Effort Estimation at Personal Level. *Computing, 2006. CIC '06. 15th International Conference on*, Nov., 298–303.

[McConnell, 2006] MCCONNELL, STEVE. 2006. Software Estimation, Demystifying the Black Art. Microsoft Press.

[McGarry *et al.*, 2001] MCGARRY, JOHN, CARD, DAVID, JONES, CHERYL, LAYMAN, BETH, CLARK, ELIZABETH, DEAN, JOSEPH, & HALL, FRED. 2001. Practical Software Measurement: Objective Information for Decision Makers. Addison-Wesley Professional.

[Mendes & Mosley, 2008] MENDES, E., & MOSLEY, N. 2008. Bayesian Network Models for Web Effort Prediction: A Comparative Study. *Software Engineering, IEEE Transactions on*, **34**(6), 723–737.

[Mukhopadhyay *et al.*, 1992] MUKHOPADHYAY, TRIDAS, VICINANZA, STEVEN S., & PRIETULA, MICHAEL J. 1992. Examining the feasibility of a case-based reasoning model for software effort estimation. *MIS Q.*, **16**(2), 155–171.

[Myrtveit *et al.*, 2005] MYRTVEIT, I., STENSRUD, E., & SHEPPERD, M. 2005. Reliability and validity in comparative studies of software prediction models. *Software Engineering, IEEE Transactions on*, **31**(5), 380–391.

[Nan & Harter, 2009] NAN, NING, & HARTER, DONALD E. 2009. Impact of Budget and Schedule Pressure on Software Development Cycle Time and Effort. *Software Engineering, IEEE Transactions on*, **35**.

[NESMA, 2001] NESMA. 2001 (August). *Function Point Analysis for Software Enhancement*. The Netherlands Software Metrics Users Association.

[Nguyen *et al.*, 2008] NGUYEN, VU, STEECE, BERT, & BOEHM, BARRY. 2008. A constrained regression technique for COCOMO calibration. *Pages 213–222 of: ESEM '08: Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*. New York, NY, USA: ACM.

[Olsina *et al.*, 2002] OLSINA, LUIS, LAFUENTE, GUILLERMO, & PASTOR, OSCAR. 2002. Towards a reusable repository for Web metrics. *Journal of Web Engineering*, **1**(1), 061–073.

[OMG, 2005a] OMG. 2005a. *Object Constraint Language 2.0 Specification*. Object Management Group (OMG).

[OMG, 2005b] OMG. 2005b. *Unified Modeling Language Specification, version 2.0*. Object Management Group (OMG).

[OMG, 2008] OMG. 2008. *Software & Systems Process Engineering Meta-Model Specification (SPEM) 2.0*. Object Modeling Group.

[Park *et al.*, 1992] PARK, ROBERT E., PARK, ROBERT E., MILLER, THOMAS R., & COL, LT. 1992. *Software size measurement: A framework for counting source statements*. Tech. rept. CMU/SEI-92-TR-020. CMU/SEI.

[Pádua, 2007] PÁDUA, WILSON. 2007 (July). Using Model-Driven Development in Time-Constrained Course Projects. *CSEET '07. 20th Conference on Software Engineering Education and Training*, 2007.

[Pádua, 2008] PÁDUA, WILSON. 2008. *Engenharia de Software: Fundamentos, Métodos e Padrões*. 3a. edição. LTC.

[Pádua, 2009] PÁDUA, WILSON. 2009. Using Quality Audits to Assess Software Course Projects. CSEET '09. *22th Conference on Software Engineering Education and Training, 2009.*

[Pádua *et al.*, 2003] PÁDUA, WILSON, MACHADO, FABIANA TRINDADE, DRUMOND, FERNANDA PAIVA, NOGUEIRA, MÁRCIA MÔNICA, & DE MESQUITA FERREIRA, GISELE RODRIGUES. 2003 (Out.). Aplicação da fase de Diagnóstico de um processo para melhoria de organizações técnicas. *In: V Simpósio Internacional de Melhoria de Processos de Software (SIMPROS)*.

[Pimentel *et al.*, 2006] PIMENTEL, BRUNO, FILHO, WILSON P. PAULA, PÁDUA, CLARINDO, & MACHADO, FABIANA T. 2006. Synergia: a software engineering laboratory to bridge the gap between university and industry. *Pages 21–24 of: SSEE '06: Proceedings of the 2006 International Workshop on Summit on Software Engineering Education*.

[PMI, 2004] PMI. 2004. *Project Management Body of Knowledge*. 3rd edition. Project Management Institute.

[PMI, 2008] PMI. 2008. *The Standard for Portfolio Management*. 2nd edn. Project Management Institute.

[Port & Korte, 2008] PORT, DAN, & KORTE, MARCEL. 2008. Comparative studies of the model evaluation criterions mmre and pred in software cost estimation research. *Pages 51–60 of: ESEM '08: Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*. New York, NY, USA: ACM.

[Putnam, 1978] PUTNAM, L.H. 1978. A General Empirical Solution to the Macro Software Sizing and Estimating Problem. *Software Engineering, IEEE Transactions on*, **SE-4**(4), 345–361.

[Rosenberg, 1997] ROSENBERG, J. 1997. Some misconceptions about lines of code. *Software Metrics Symposium, 1997. Proceedings., Fourth International*, Nov, 137–142.

[Ryder, 1998] RYDER, J. 1998. Fuzzy modeling of software effort prediction. *Information Technology Conference, 1998. IEEE*, Sep, 53–56.

[Triola, 2008] TRIOLA, MARIO F. 2008. *Introdução à Estatística*. 10a. edição. LTC Editora.

[Uemura *et al.*, 1999] UEMURA, T., KUSUMOTO, S., & INOUE, K. 1999. Function point measurement tool for UML design specification. *Proceedings., Sixth International Software Metrics Symposium, 1999*.

[Yang *et al.*, 2008] YANG, DA, WANG, QING, LI, MINGSHU, YANG, YE, YE, KAI, & DU, JING. 2008. A survey on software cost estimation in the chinese software industry. *Pages 253–262 of: ESEM '08: Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, 2008*.

[Zadeh *et al.*, 1977] ZADEH, L. A., FU, K. S., TANAKA, K., SHIMURA, M., & NEGOITA, C. V. 1977. Fuzzy Sets and Their Applications to Cognition and Decision Processes. *Systems, Man and Cybernetics, IEEE Transactions on*, **7**(2), 122–123.

Apêndices

A.1 Lista de conferência de definição de contagem de linhas de código

Nome da definição: **Linhas de código físicas** Data: 1/11/2008

Unidade de medida		
	Linhas de código físicas	X
	Instruções lógicas de código	

Tipo de instrução		Definição	Vetor de dados	Ordem de precedência -->	Inclui	Exclui
		X				
1	Executável			1	X	
2	Não-executável					
3	Declarações			2	X	
4	Diretivas de compilação			3	X	
5	Comentários					
6	Em linha própria			4		X
7	Em linha que contém código-fonte			5		X
8	Banners (ex: faixas de comentário em cabeçalhos)			6		X
9	Comentários em branco			7		X
10	Linhas em branco			8		X

Obs: Quando uma linha se encaixa em mais de um tipo, classificar como o tipo de maior prioridade.

Como foi produzida		Definição	Vetor de dados	Inclui	Exclui
		X			
1	Programada			X	
2	Gerada com geradores de código				X
3	Convertida com tradutores de código			X	
4	Copiada ou reusada sem modificação			X	
5	Modificada			X	
6	Removida				X

Origem	Definição	X	Vetor de dados		Inclui	Exclui
1 Projeto novo: não existia anteriormente					X	
2 Código já existente (fonte ou objeto), utilizado ou adaptado de:						
3 Versão anterior de projetos do Synergia.					X	
4 Bibliotecas de reuso próprias, desenvolvidas fora do projeto.					X	
5 Bibliotecas ou produtos de terceiros, utilizados no projeto sem alteração						X
6 Bibliotecas ou produtos de terceiros, utilizados com alguma alteração					X	

Propósito de uso	Definição	X	Vetor de dados		Inclui	Exclui
1 Dentro do produto primário (códigos de testes de desenvolvimento estão inclusos)					X	
2 Externo ou de suporte ao produto primário (simuladores, cotos, etc.)						X

Entrega	Definição	X	Vetor de dados		Inclui	Exclui
1 Entregue						
2 Entregue como código-fonte					X	
3 Entregue como código-objeto, mas não como código-fonte					X	
4 Não entregue						
5 Está sob controle de versão						X
6 Não está sob controle de versão						X

Funcionalidade do código	Definição	X	Vetor de dados		Inclui	Exclui
1 Operante					X	
2 Inoperante						
3 Funcional (código intencionalmente morto, contornado, que pode ser reativado)					X	
4 Não funcional (presente, mas sem intenção)						X

Estado do desenvolvimento	Definição	X	Vetor de dados		Inclui	Exclui
1 Antes de iniciar a codificação						X
2 Codificado						X
3 Testes de desenvolvimento codificados						X
4 Aprovado nas inspeções de testes de desenvolvimento e implementação						X
5 Aprovado na revisão de implementação						X
6 Testes de integração executados						X
7 Todos os defeitos de testes de integração corrigidos					X	

Tipo de arquivo	Definição		Vetor de dados	X	**Inclui**	**Exclui**
1 Java (arquivos com extensão .java, exceto com o padrão de nome *Helper.java)					X	
2 Web Estática (.html, .htm, .css, .xsl)					X	
3 Web Dinâmica (.jsp, .js)					X	
4 Configuração (.xml, .config, .properties)					X	
5 Robot (.rec, .sbl, .sbh)					X	

Camada	Definição		Vetor de dados	X	**Inclui**	**Exclui**
1 Fronteira					X	
2 Controle					X	
3 Entidade					X	
4 Testes de desenvolvimento (unidade)					X	
5 Povoadores de banco de dados					X	
6 Testes de sistema (scripts do Rational Robot e Functional Tester)					X	

Origem do código	Definição		Vetor de dados	X	**Inclui**	**Exclui**
1 Código novo					X	
2 Código reusado sem modificação					X	
3 Código reusado com modificação					X	

A.2 Descrição dos atributos do repositório centralizado de medidas da Organização

Dimensão	Atributo	Descrição
Alteração	Tipo de alteração	Defeito detectado em artefato aprovado nas revisões, Alteração de requisito, Alteração em solução
	Descrição	Texto descrevendo a alteração a ser executada
Projeto	Nome do Projeto	Nome do projeto
	Parâmetros do COCOMO	Consiste nos diversos fatores de escala e multiplicadores de esforço que caracterizam o projeto
	Esforço Orçado	Esforço total orçado para o projeto
	Custo Orçado	Custo total orçado para o projeto. Utiliza as categorias dos colaboradores e portanto é baseado na média do custo dos colaboradores dentro das categorias.
	Prazo Orçado	Prazo total orçado para o projeto em meses.
Atividade	ID CQ	Identificador do registro na ferramenta Rational ClearQuest. Pode ser usado para recuperar detalhes do registro quando necessário.
	Origem CQ	Tipo de registro no Rational ClearQuest: Tarefa, Revisão, Alteração, etc.
	Nome Atividade	Nome da atividade executada. É consistente com a definição do processo Praxis-Synergia.
	Artefato Associado	Artefato sendo modificado pela atividade ou 'Nenhum' quando for uma tarefa geral, como uma reunião.
	Estado Associado	Estado associado do caso de uso associado à atividade. Varia de 10 a 100 conforme definição do Praxis-Synergia.
	Nome do estado associado	Descrição textual do estado. Ex: Identificado, Detalhado, etc.
	Disciplina	Refere-se à disciplina associada à atividade executada.
	É retrabalho	Define se uma atividade é considerada ou não retrabalho no projeto.
	Tipo de atividade	
	Tipo de esforço	Definido em: Tarefa, revisão, correção, verificação, alteração
	Data de início planejada	Data de início planejada pelo gerente. Armazena apenas o primeiro planejamento.
	Data de término planejada	Data de término planejada pelo gerente. Armazena apenas o primeiro planejamento.
	Data de início real	Data de início real da atividade.
	Data de término real	Data de término real da atividade.
	Esforço planejado	Esforço planejado pelo gerente. Armazena apenas o primeiro planejamento.
	MRE	Medida do erro de estimativa do esforço.

Dimensão	Atributo	Descrição
Pacote de trabalho	Nome	Nome do pacote de trabalho. São subdivisões do escopo do produto.
Requisito	Tag	Identificador do requisito na ferramenta Rational RequisitePro, utilizada para manter o cadastro de requisitos do projeto.
	Nome	Nome do requisito
	Tag e nome	Texto concatenando a Tag e Nome do requisito
	Complexidade	Complexidade definida para o requisito. Pode ser Baixa, Média ou Alta.
	Total PF	Total de pontos de função não ajustados do requisito.
	Estado Atual	Estado em que se encontra o requisito. Varia de 10 a 100 conforme definição do processo Praxis-Synergia.
Data	Data	Data no calendário.
	Data formatada	Texto representando a data formatada para exibição em relatórios.
	MesAno	Texto representando o mês e ano da data.
	MesPorExtenso	Texto representando o mês associado à data.
	Ano	Texto representando o ano associado à data.
	Semana	Data do Domingo anterior à data. Representa a data do início da semana.
Colaborador	Login	Login de rede do colaborador. É considerado a chave primária dessa entidade.
	Nome	Nome do colaborador.
Iteração	Nome	Nome da iteração. Ex: E1, E2, C1, C2, etc.
	Data de início planejada	Data de início planejada pelo gerente. Armazena apenas o primeiro planejamento.
	Data de término planejada	Data de término planejada pelo gerente. Armazena apenas o primeiro planejamento.
	Data de início real	Data de início real da atividade.
	Data de término real	Data de término real da atividade.
Classificação LOC	Tipo de arquivo	Opções definidas na lista de conferência de contagem de Linhas de Código. Ver Apêndice A.1.
	Camada	Opções definidas na lista de conferência de contagem de Linhas de Código. Ver Apêndice A.1.
	Origem do Código	Opções definidas na lista de conferência de contagem de Linhas de Código. Ver Apêndice A.1.

Dimensão	Atributo	Descrição
Defeito	Gravidade	Gravidade do defeito: Blocante, Crítico, Maior, Menor, Sugestão.
	Natureza	Natureza do defeito: modelagem, imprecisão, codificação, estilo, omissão, redundância.
	Item Lista de Conferência	Código do item da lista de conferência relacionado ao defeito.
	Estado	Situação do defeito: Novo, Corrigido, Verificado, Inválido, Fechado.
Transação do defeito	Nome	Nome da transação realizada com o defeito. Está associado à máquina de estados. Pode ser: Registrado, Invalidado, Reaberto, Corrigido, Verificado, Fechado.