$PCU|_{PSP}$: Uma Estratégia para ajustar Pontos por Casos de Uso por meio do PSP em Empresas de Pequeno Porte

Arnaldo Sanchez, Renan Montebelo, Sandra Fabbri

Departamento de Computação – Universidade Federal de São Carlos (UFSCar)
São Carlos – SP – Brasil

arnaldo@linkway.com.br, {renan_montebelo, sfabbri}@dc.ufscar.br

Abstract. *This paper presents a strategy that allows the adjustment of the Use Case Points technique, so that it reflects the effective characteristics of the development team in a way in which the planning of the system to be developed can be more realistic. This adjustment is resulted from the practical application of the PSP™ (Personal Software Process) technique and from the constant evaluation of such application, together with the updating and control of the elaborated development plan. This strategy was extracted from the systematic application of these two techniques used jointly in a small business company, and the obtained results give indications of its contribution for the improvement of the software development process quality, mainly in respect to the planning and control activities.*

Resumo. *Este artigo apresenta uma estratégia que permite o ajuste da técnica de Pontos por Casos de Uso para que esta reflita efetivamente as características da equipe de desenvolvimento, de forma que o planejamento do sistema a ser desenvolvido seja mais próximo da realidade. Esse ajuste provém da aplicação de práticas da técnica PSP™ (Personal Software Process) e de uma constante avaliação das mesmas, em conjunto com a atualização e o controle do plano de desenvolvimento elaborado. Essa estratégia foi extraída da aplicação sistemática dessas duas técnicas em conjunto em empresas de pequeno porte. Os resultados obtidos dão indícios de sua contribuição para a melhoria da qualidade do processo de desenvolvimento de software, principalmente no que diz respeito às atividades de planejamento e controle.*

1. Introdução

A definição de Qualidade de Software nem sempre é clara ou consistente. Uma definição abrangente e completa diz que qualidade de software é "conformidade com requisitos funcionais e de desempenho explicitamente declarados, padrões de desenvolvimento claramente documentados e características implícitas que são esperadas de todo software desenvolvido profissionalmente" [Pressman, 2006]. Al Davis descreve qualidade de software como "nada mais, nada menos do que satisfazer as necessidades do usuário, sendo estas documentadas ou não" [Eickelmann, 2004]. Independentemente da definição de qualidade adotada, é certo que produtos com qualidade são essenciais para a sobrevivência no mercado de software, tanto para pequenas empresas quanto para grandes organizações. Tratando-se de uma atividade

essencialmente intelectual, como é o desenvolvimento de software, a qualidade do processo tem influência direta na qualidade do produto final, assim como no cronograma e no custo de desenvolvimento.

Portanto, na tentativa de garantir qualidade para o software, vários padrões de qualidade de processo foram desenvolvidos e propostos na literatura. Alguns modelos, como o SW-CMM e o CMMI, tratam de melhoria contínua de processo e, portanto, tornaram-se modelos de grande prestígio na indústria de software. O SW-CMM e o CMMI possuem diferentes níveis de maturidade que descrevem, inclusive, as atividades de engenharia de software que devem estar presentes para satisfazer uma boa prática de desenvolvimento de software. O modelo MPS-BR é baseado no CMMI e nas normas ISO/IEC 12207 e ISO/IEC 15504, mas é adaptado à realidade do mercado de software brasileiro. Sua principal vantagem é o reduzido custo de certificação em relação a normas estrangeiras. Em comum, todos os modelos citados têm atividades de planejamento e de controle como atividades iniciais à implantação de qualidade de processo. Assim, para que um processo de desenvolvimento de software seja eficiente, é importante que as atividades de Planejamento e de Acompanhamento de projetos sejam bem definidas. Essas atividades são complementares entre si e, em geral, são o cerne em torno do qual as demais atividades são exercidas. Através de um bom planejamento é possível que a empresa estime o tempo e o custo de desenvolvimento de um sistema, avaliando se o desenvolvimento é viável economicamente.

As atividades de planejamento podem ser então, consideradas estratégicas para as empresas, mas somente serão eficazes se acompanhadas por atividades de controle e acompanhamento. Pouco ou nenhum sentido há em planejar sem garantir que o planejamento definido será efetivamente obedecido. Nesse sentido, as atividades de controle e acompanhamento não só atestam a conformidade ou não ao planejamento, mas, mais importante, podem indicar os motivos da não conformidade. Por meio da análise dos dados de controle e acompanhamento é possível aperfeiçoar o planejamento atual e planejamentos futuros.

Uma das muitas técnicas que podem ser utilizadas durante o planejamento são os Pontos por Caso de Uso [Karner, 1993]. Essa técnica, derivada de Pontos por Função, foi desenvolvida por Gustav Karner e vem sendo bastante utilizada por alguns motivos: ampla utilização de diagramas UML em projetos de software (em especial o Modelo de Casos de Uso); simplicidade, se comparada às demais técnicas; e independência de linguagem e paradigma.

Para as atividades de controle e acompanhamento, como é proposto na estratégia aqui apresentada, se pode utilizar o PSP (*Personal Software Process*), em especial as atividades descritas no nível 1.1 do modelo. Embora o PSP seja um processo definido para melhoria individual, suas atividades geram métricas que podem ser úteis para uma pequena equipe ou empresa, como relatam diversos autores [Eman, 1996; Escala, 1999; Morisio, 2000]. Essas atividades fornecem avaliação imediata do processo pessoal de desenvolvimento que, através de análise, pode revelar os pontos fracos do processo. A proposta do PSP se torna ainda mais interessante uma vez que esta se mostra bastante flexível, incentivando desenvolvedores a adaptarem sua aplicação à realidade de seu trabalho.

Uma vez que a empresa dispõe de atividades de planejamento e controle

consolidadas, outras atividades estratégicas podem ser desenvolvidas com o intuito de solidificar a posição da empresa no mercado. Este artigo, neste sentido, define uma estratégia que contempla tanto atividades de planejamento quanto atividades de acompanhamento, que, de forma conjunta, possibilitam o constante ajuste das variáveis de Pontos por Caso de Uso por meio de coleta e análise de métricas do PSP. Dessa forma, é possível que pequenas empresas tenham estimativas de prazo e custo mais realísticas e, conseqüentemente, desenvolvam sistemas de software com mais qualidade.

Este artigo se encontra organizado em oito seções: nesta primeira seção foi apresentada a introdução e a contextualização da proposta; na Seção 2 é apresentada de forma sucinta a técnica de Pontos por Caso de Uso; a técnica PSP é apresentada na Seção 3; na Seção 4 são caracterizadas as duas empresas de pequeno porte abordadas neste artigo; na Seção 5 é apresentada a estratégia proposta, enquanto na Seção 6 são apresentados dois estudos de caso de sua aplicação; na Seção 7 são apresentadas as lições aprendidas durante o estabelecimento e implantação da proposta; e na Seção 8 são apresentadas as conclusões e os trabalhos futuros.

2. Pontos por Caso de Uso

Estimativas de custo e de tempo de produção de um sistema de software sempre representaram um desafio para os desenvolvedores, uma vez que não é uma tarefa trivial calcular o porte e, conseqüentemente, o custo do sistema. Diversas técnicas foram formuladas para facilitar a estimativa de esforço na construção de um sistema, entre as quais se destaca, neste artigo, a técnica de Pontos por Caso de Uso. Essa técnica, proposta por Gustav Karner (1993), foi baseada na técnica de Pontos por Função [Albrecht, 1979]. As estimativas são calculadas tomando como base a complexidade dos casos de uso que compõem o sistema e dos atores que utilizarão o mesmo. Essa estimativa também leva em consideração os fatores de complexidade técnica relativos aos requisitos não funcionais, de forma semelhante à técnica de Pontos por Função. No entanto, acrescenta-se uma inovação proposta por Karner: a aplicação de Fatores Ambientais, que consideram o nível de competência da equipe que desenvolverá o sistema [Karner,1993].

O peso dos atores é determinado de acordo com a interface disponível para utilização do sistema. Se o ator acessa o sistema através de um outro sistema, por uso de uma API, é considerado simples e tem peso 1; se o sistema é acessado por uma interface texto ou interface gráfica, a complexidade do ator é, respectivamente, média ou complexa, cujos pesos são 2 e 3.

Os Casos de Uso são classificados de acordo com o número de transações e de objetos de análise envolvidos. Analogamente aos atores, são classificados como simples, médios ou complexos. A Tabela 1 apresenta a classificação dos Casos de Uso, segundo o trabalho de Karner. A soma dos pesos de todos os Casos de Uso com os pesos de todos os atores constitui os Pontos por Caso de Uso Não Ajustados.

Tabela 1 - Classificação de Caso de Uso [Karner, 1993]

Tipo do Caso de Uso	Número de Transações	Número de Objetos de Análise	Peso
Simples	Até 3	Até 5	5
Médio	De 4 até 7	De 6 até 10	10
Complexo	Acima de 8	Acima de 11	15

O Fator de Complexidade Técnica (FCT) e os Fatores Ambientais (FA) são calculados para que se considerem os requisitos não funcionais e ambientais, respectivamente. FCT foi adaptado da técnica de Pontos por Função, tendo alguns fatores adicionados, removidos ou alterados, de acordo com a experiência de Karner na empresa Objectory (atualmente Rational Software). FA é uma criação de Karner para a técnica de Pontos por Caso de Uso.

Cada FCT e FA possuem diferentes pesos no desenvolvimento do sistema, como mostrado na Tabela 2. Atribui-se para cada FCT e FA um nível de influência que varia em uma escala de zero a cinco, sendo que zero significa que é irrelevante, enquanto cinco significa que é essencial. Se o fator não é essencial e nem irrelevante, Karner sugere que seja atribuído valor três, pois assim, se todos os fatores tiverem esse valor, o FCT e/ou FA será aproximadamente de valor um e, portanto, não influencia no ajuste dos Pontos por Casos de Uso.

Tabela 2 - Fatores de Complexidade Técnica e Fatores Ambientais [Karner,1993]

FCT	Descrição	Peso	FA	Descrição	Peso
F1	Sistema distribuído	2	E1	Familiaridade com o processo de desenvolvimento.	1,5
F2	Tempo de Resposta	1	E2	Desenvolvedores em meio expediente.	-1
F3	Eficiência	1	E3	Presença de analistas experientes	0,5
F4	Processamento complexo	1	E4	Experiência com a aplicação em desenvolvimento.	0,5
F5	Código reusável	1	E5	Experiência em Orientação a Objetos.	1
F6	Facilidade de instalação	0,5	E6	Motivação	1
F7	Facilidade de uso	0,5	E7	Dificuldade com a linguagem de programação	-1
F8	Portabilidade	2	E8	Requisitos estáveis	2
F9	Facilidade de mudança	1			
F10	Concorrência	1			
F11	Recursos de segurança	1			
F12	Acessível por terceiros	1			
F13	Requer treinamento especial	1			

Os Fatores de Complexidade Técnica e os Fatores Ambientais são calculados de acordo com as seguintes fórmulas [Karner, 1993]:

$$FCT = 0,6 + 0,01 \sum_{i-1}^{13} F_i * Influência_i \qquad FA = 1,4 - 0,03 \sum_{i-1}^{8} E_i * Influência_i$$

sendo que F_i e E_i são, respectivamente, os pesos tabelados de cada FCT e FA, e a $Influência_i$ é o valor entre zero e cinco atribuído.

O Cálculo do total de Pontos por Caso de Uso de um sistema é realizado através da seguinte fórmula [Karner, 1993]:

$$PCU = Pontos\ por\ Caso\ de\ Uso\ não\ ajustados * FCT * FA$$

O valor do nível de esforço em horas/homens proposto por Karner é de 20 horas para cada Ponto por Caso de Uso. Convencionou-se, neste artigo, chamar esse valor de NDE - Nível de Esforço. Segundo Probasco [Probasco, 2002], a principal dificuldade encontrada para aplicar a técnica proposta por Karner está na compreensão da complexidade dos casos de uso. Probasco faz algumas recomendações importantes para facilitar a aplicação da técnica:

- Os Casos de Uso não podem ser nem muito decompostos e nem muito alto nível;
- Pode-se considerar que um cenário de Caso de Uso seja o que equivale a uma transação atômica da proposta de Karner (1993);
- Deve-se estabelecer um procedimento para determinar a granularidade dos casos de uso e este deve ser seguido coerentemente durante todo o desenvolvimento.

3. Personal Software Process

O PSP é uma técnica que tem por objetivo melhorar a previsibilidade, a produtividade e, principalmente, a qualidade do trabalho de um desenvolvedor de software [Humphrey, 1995]. Embora o PSP use vários métodos convencionais de engenharia de software, seu principal objetivo é mostrar aos desenvolvedores que um processo bem definido e controlado pode ajudá-los a melhorar seu desempenho pessoal. O PSP consiste em sete níveis que, gradualmente, introduzem novos dados e análises, que por sua vez, podem ser utilizados para determinar o desempenho e medir a eficácia dos métodos utilizados.

As práticas do PSP são derivadas do CMM^{TM} (*Capability Maturity Model*), também desenvolvido pelo SEI (*Software Engineering Institute*). O CMM captura as melhores práticas de desenvolvimento de software utilizadas por grandes corporações e é organizado em cinco níveis de maturidade. O PSP é o resultado de uma adaptação do CMM, mas com foco no indivíduo. O PSP reconhece que cada engenheiro tem características únicas e, assim, se deve adaptar o PSP à sua maneira de trabalho. Assim, ao usar os dados imediatamente disponibilizados pelo PSP, os engenheiros podem determinar quais métodos são mais eficazes em seu modo de trabalho.

O PSP sugere que as atividades de desenvolvimento de software sejam divididas em Planejamento, Desenvolvimento (projeto, codificação e testes) e Análise Final, embora variações sejam aceitas. Anterior ao planejamento, assume-se que o levantamento de requisitos tenha sido realizado e que o artefato gerado desse levantamento sirva como entrada à primeira etapa mencionada.

Durante o Planejamento, todas as estimativas são realizadas e o desenvolvedor se concentra em produzir valores que se aproximem ao máximo dos futuros dados reais. Durante o Desenvolvimento, o desenvolvedor despende a maior parte do tempo efetivamente construindo o software, ou seja, projetar, codificar e testar o software. Por último, como já diz o próprio nome, a Análise Final é utilizada para levantar informações a respeito de todo o trabalho realizado nas fases anteriores e estabelecer um comparativo entre o que foi planejado e estimado com o que foi executado realmente.

O nível 0, chamado de *Baseline*, representa apenas uma pequena introdução do PSP nas práticas de desenvolvimento pessoal. Nessa etapa, as práticas do PSP são abrangentes o suficiente para que o desenvolvedor sinta pouca diferença entre o processo de desenvolvimento que ele já usa e o que está sendo introduzido. Algumas métricas começam a ser coletadas para que os alicerces do processo de melhoria sejam estabelecidos como o tempo gasto no desenvolvimento e o número de defeitos identificados em cada fase de desenvolvimento. No nível 0.1 são acrescentados os conceitos de padronização de código e de contagem de linhas de código. Basicamente, o desenvolvedor define um mínimo de estimativas na fase de planejamento, indicando quanto tempo ele pretende dedicar para realizar cada fase do processo, quantas linhas de

código serão criadas, alteradas e reusadas, para então iniciar a fase de desenvolvimento. Cada defeito identificado durante o processo é registrado e armazenado em um *Log* de Defeitos, para que se consigam identificar, exatamente, as maiores dificuldades.

O nível 1 acrescenta atividades de planejamento ao PSP. O relatório de testes, juntamente com estimativas de tempo e recursos, são os principais acréscimos desse nível. Assim, possibilita-se que desenvolvedores comecem a entender a relação entre linhas de código produzidas e o tempo despendido no desenvolvimento, ao mesmo tempo em que entendem como planejar e avaliar seu trabalho com os dados coletados. O objetivo do nível 1.1 é acrescentar o planejamento de recursos (principalmente tempo), à comparação do desempenho do desenvolvedor frente suas estimativas, e os formulários de planejamento de tempo e cronograma. Isso significa que, a partir desse nível, o desenvolvedor realiza, na fase de Planejamento, as estimativas de quantidade de linhas de código e de tempo a ser despendido em cada fase de desenvolvimento. Dessa forma, é possível organizar o trabalho e medi-lo freqüentemente ao longo de todo ciclo de desenvolvimento. Isso mostra ao desenvolvedor como é o desempenho de seu trabalho em cada uma das fases do processo, possibilitando que se notem os pontos fracos e fortes, e que assim o desenvolvedor procure aprimorar os aspectos que representam maiores problemas para o seu desempenho pessoal.

A qualidade passa a ser o principal critério do nível 2. O ponto mais importante é a introdução de *checklists* de revisão usados para a identificação precoce de defeitos. Inicialmente, esses *checklists* são próprios do PSP, mas o desenvolvedor é encorajado e responsável por analisar seus pontos fracos no desenvolvimento e, assim, personalizar os *checklists*. No nível 2.1, a preocupação não é só com a qualidade da implementação do software, mas também com a qualidade com que o projeto do sistema é conduzido. Dessa forma, o PSP não diz exatamente como deve ser feita a modelagem ou a especificação do software, mas sugere como fazer especificações completas e consistentes. O objetivo é ajudar a reduzir os defeitos inseridos, principalmente, na fase de projeto, oferecendo critérios para a avaliação dos mesmos.

Até os níveis anteriormente descritos, o PSP é praticável para programas com até algumas poucas mil linhas de código. Seria muito provável perder a lógica do sistema ou despender muito tempo em testes considerando o software como uma única unidade. O nível 3 possui o objetivo de ajudar o desenvolvedor na divisão de seu projeto de muitas mil linhas de código em pequenos projetos adequados ao PSP nível 2.1. Para isso, o PSP propõe,que uma porção principal do software ("*kernel*") seja desenvolvida com a aplicação dos níveis 2 e 2.1, e que as diversas iterações necessárias sejam executadas sobre esse "*kernel*". A cada ciclo de iteração, os critérios de qualidade são aplicados garantindo-se que outra iteração possa ser realizada sem que ocorra regressão no desenvolvimento. Assim, o PSP mostra-se como um processo bem definido de melhoria contínua pessoal, útil tanto para pequenos sistemas quanto para sistemas maiores.

4. Caracterização das Empresas de Pequeno Porte Utilizadas no Estudo

As empresas escolhidas para o estudo de caso foram a Linkway, um pequeno provedor de Internet que desenvolve aplicativos Web, e a NBS, uma pequena empresa de software focada em aplicativos de gestão e administração pública. Ambas são situadas

na cidade de São Carlos, interior do Estado de São Paulo.

A Linkway tem uma equipe de desenvolvimento formada por um Gerente, dois consultores de vendas, dois *Web Designers* e três programadores Java. A empresa está há dez anos no mercado desenvolvendo aplicativos *Web* sob encomenda. A estratégia de mercado da empresa de não desenvolver uma linha de produtos de uso genérico, mas sim, de construir aplicativos baseados nas necessidades de cada cliente, criou um ambiente de desenvolvimento de intensa produção de software com as mais variadas necessidades. O controle de prazos e custo é, portanto, vital, uma vez que a receita está diretamente vinculada à entrega de cada sistema dentro do custo e do prazo estipulados. Assim, a cada novo sistema, um novo ciclo de desenvolvimento é criado. A necessidade de resultados rápidos - uma característica de sistemas *Web* [Capilla, 2003] - criou um ambiente favorável de melhoria contínua nos processos de desenvolvimento dos projetos. Os ciclos rápidos de desenvolvimento, associados à melhoria contínua, foram fundamentais para a formulação da estratégia proposta neste artigo.

A NBS é uma empresa que participa de um mercado com relativa estabilidade de requisitos, pois as regras de negócio de administração pública não mudam na mesma intensidade de outros setores. O desenvolvimento está concentrado em construir novos módulos aos aplicativos já existentes, com a finalidade de acrescentar novas funcionalidades e, assim, tornar o sistema mais abrangente. A equipe de desenvolvimento da NBS é formada por um Gerente, dois programadores, dois analistas de suporte e um consultor de vendas.

As duas empresas utilizam o paradigma de Orientação a Objetos e, para a modelagem dos sistemas, a *Unified Modeling Language*. Ainda, ambas as empresas utilizam protótipos de tela parcialmente funcionais, a fim de definir bem os requisitos.

5. PCU|$_{PSP}$: Estratégia de Ajuste de Pontos por Casos de Uso Baseada no PSP

Nesta seção apresenta-se a estratégia aqui proposta, a qual foi estabelecida a partir do uso sistemático e conjunto das técnicas PCU e PSP, estratégia esta que dá suporte à elaboração do planejamento e ao acompanhamento do plano de desenvolvimento, melhorando com isso, a qualidade do processo de software. Na Figura 1 apresenta-se um esquema da estratégia.

Nota-se que atividades de planejamento e de controle são constantemente executadas, complementando uma à outra. O objetivo da primeira é permitir que se determinem estimativas adequadas de tamanho, prazos e custos antes do desenvolvimento do sistema ou mesmo de uma rotina. O objetivo das atividades de controle é avaliar se o planejamento determinado está sendo obedecido e, em caso negativo, determinar o motivo associado a esse fato. É através do *feedback* das atividades de controle que as atividades de planejamento são constantemente ajustadas.

A estratégia se inicia quando o Gerente tem disponíveis os Casos de Uso que compõem todo o sistema, ou parte dele (Etapa 1). Imediatamente, o Gerente convoca uma reunião com cada um dos desenvolvedores, na qual serão discutidos todos os cenários de cada Caso de Uso. Essa é a primeira interação bem definida entre Gerente e Desenvolvedor, em que o objetivo é atingir um consenso sobre o entendimento de cada

cenário de cada Caso de Uso. Assim, nessa reunião o Gerente determina, com a ajuda do desenvolvedor, a complexidade dos Casos de Uso, assim como os níveis de influência de cada Fator Técnico e Fator Ambiental [Karner, 1993].

Uma vez que os Casos de Uso foram discutidos, o Gerente pode realizar o planejamento de alto nível de todo o projeto (Etapa 2). Nesse momento, o Gerente, através da aplicação da técnica de Pontos por Caso de Uso, tem uma estimativa do tamanho do sistema, o qual é a soma dos Pontos por Caso de Uso atribuídos a cada desenvolvedor. Aplicando a variável de nível de esforço, se pode determinar também o custo e o tempo de desenvolvimento. Ressalta-se que, em uma primeira vez, enquanto não se tem uma caracterização de cada desenvolvedor, se pode adotar o valor proposto por Karner de 20 horas/homens.

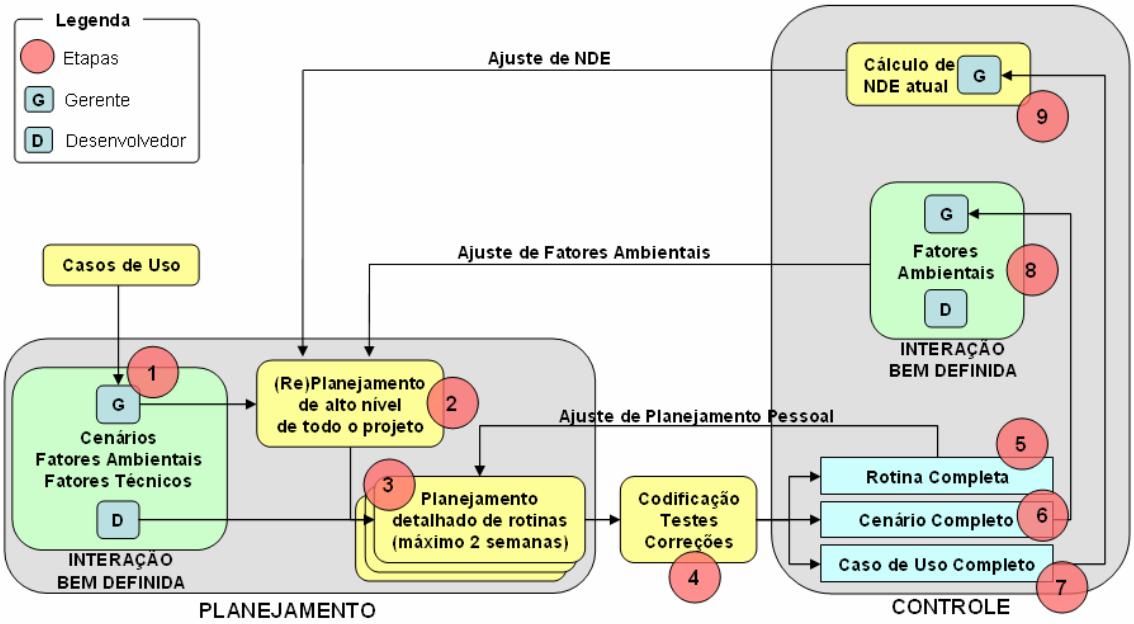

Figura 1 – Esquema da estratégia PCU|PSP

Na Etapa 3, cada desenvolvedor utilizará o Planejamento de alto nível, da Etapa 2, como base para montar o planejamento detalhado de cada cenário. O desenvolvedor é responsável por dividir os cenários dos Casos de Uso em rotinas menores, e deve fazê-lo da maneira que julgar mais apropriada. Tais rotinas são as atividades essenciais para o correto funcionamento do cenário. Assim, os desenvolvedores devem realizar o planejamento de cada uma das rotinas de acordo com o PSP Nível 1.1, elaborando um cronograma detalhado de desenvolvimento que inclui também o tamanho das rotinas (em número de linhas de código). O cronograma deve estar limitado a, no máximo, o planejamento de duas semanas, mesmo que vários cronogramas devam ser elaborados ao longo do desenvolvimento dos cenários que lhe foram atribuídos. Isso evita erros e distorções comuns em longos planejamentos.

O desenvolvedor inicia, então, a atividade de desenvolvimento de software (Etapa 4), incluindo codificação, testes e correção de defeitos. Ao final do ciclo de desenvolvimento de cada rotina, segue-se uma fase de *post-mortem* (Etapa 5) em que o desenvolvedor analisará seu desempenho real em comparação ao desempenho estimado na fase de planejamento (Etapa 3), correspondente ao nível 1.1 do PSP [Humphrey,

1995]. São analisados, portanto, o tempo despendido em relação ao tempo estimado para a rotina desenvolvida, e o tamanho final da rotina em relação ao tamanho estimado. Por meio dessa análise torna-se possível que o desenvolvedor ajuste o planejamento de outras rotinas, caracterizando-se assim um processo pessoal de melhoria contínua, que o leva a ajustar o seu planejamento para futuro desenvolvimento. Apesar da estratégia aqui proposta incluir somente atividades do nível 1.1 do PSP, o desenvolvedor continua com total liberdade para exercer as demais atividades dos outros níveis; no entanto, essas atividades, por ora, ainda não participam da estratégia aqui proposta, mas serão analisadas em trabalhos futuros para um eventual aproveitamento em outras versões da estratégia.

Quando o desenvolvedor conclui um conjunto de rotinas que caracterizam completamente um cenário (Etapa 6), o Gerente deve ser informado do fato. Esta, por sua vez, compara o desempenho real do desenvolvedor em relação ao planejamento de alto nível desenvolvido na Etapa 2. Se houver discrepância relevante o suficiente para que o planejamento de alto nível do sistema esteja ameaçado, é necessária uma reunião entre Gerente e Desenvolvedor para identificar as causas de tal discrepância e buscar, em conjunto, meios de transpor tais dificuldades (Etapa 8). É muito importante que o Gerente tenha uma postura pró-ativa em relação ao incentivo de melhoria mútua, uma vez que, em reunião anterior, buscou-se o consenso sobre a complexidade do Caso de Uso. O erro, nesse caso, pode estar no planejamento de ambos os lados. Assim, determina-se a segunda interação bem definida entre Gerente e Desenvolvedor, cujo objetivo é encontrar, em consenso, um meio para contornar os problemas encontrados. Exemplificando, o Desenvolvedor pode informar ao Gerente que, ao contrário de sua opinião inicial, ele não possui domínio suficiente de uma linguagem de programação e, portanto, solicita um tempo maior de treinamento na linguagem. O Gerente, por sua vez, atualiza o nível de influência do Fator Ambiental "Dificuldade com a Linguagem de Programação", aumentando o seu valor. Tal aumento tem impacto na técnica de Pontos por Caso de Uso, que, por sua vez, tem impacto no Planejamento de Alto Nível do projeto.

Pelo fato do Gerente ser informado do desempenho de cada desenvolvedor ao final do desenvolvimento dos cenários, é possível, portanto, ajustar continuamente o Planejamento de Alto Nível. Neste ponto, é apropriado realizar tal ajuste através da revisão da intensidade dos Fatores Ambientais dos Pontos por Casos de Uso (Etapa 8). Ressalta-se então, que no acompanhamento realizado entre as etapas 6, 8 e 2, apenas o nível de influência dos Fatores Ambientais necessita ser alterado, uma vez que os Fatores Técnicos se mantêm inalterados quando os requisitos do sistema também assim se mantêm. Essa atividade de ajuste do planejamento de alto nível se caracteriza também como um processo de melhoria contínua do processo de desenvolvimento como um todo. A cada cenário concluído, repete-se esse ciclo e o planejamento de alto nível é novamente ajustado.

Quando todos os cenários de um Caso de Uso foram desenvolvidos (Etapa 7), o Gerente deve comparar o planejamento estimado para o desenvolvimento desse Caso de Uso em relação ao tempo real gasto para tal atividade. Assim, é possível ajustar o NDE (Etapa 9) de cada desenvolvedor logo no primeiro caso de uso concluído, para que este reflita a verdadeira relação entre Pontos por Casos de Uso e o esforço em horas/homens.

O mecanismo de ajuste é bastante simples, bastando dividir o tempo real gasto (fornecido pelo PSP após o Caso de Uso ser concluído) pela quantidade de Pontos do Caso de Uso em questão. Tal mecanismo deve ser repetido sempre que um Caso de Uso é finalizado (Etapa 7), ou seja, acumulam-se o tempo gasto e os Pontos por Caso de Uso já realizados e divide-se um pelo outro. Dessa forma, ao ajustar o NDE de cada Desenvolvedor está sendo também ajustado o planejamento de alto nível de todo o projeto. Esse constante ajuste do NDE (Etapa 9) se caracteriza, mais uma vez, como uma melhoria contínua do processo.

Em suma, a estratégia aqui proposta considera que os diferentes ajustes no planejamento sejam determinados de acordo com a "grandeza" e impacto da atividade que está sendo realizada. Assim, erros nas estimativas de rotinas afetam somente o planejamento do desenvolvedor (Etapa 5); erros nas estimativas de cenários afetam somente Fatores Ambientais (Etapas 6 e 8); e, finalmente, erros nas estimativas de Caso de Uso afetam o NDE (Etapas 7 e 9). Nota-se que, quanto maior é a unidade de desenvolvimento (rotina / cenário / Caso de Uso) na qual o erro de estimativa se encontra, maior o impacto na estratégia (planejamento pessoal do desenvolvedor / Fatores Ambientais / NDE). Tal mecanismo pode ser facilmente entendido observando-se a Figura 1.

6. Estudo de Caso da Aplicação da Estratégia

Nesta seção são relatados dois estudos de caso de aplicação da estratégia $PCU|_{PSP}$. Um deles na empresa Linkway e outro na empresa NBS, descritas na Seção 4. O primeiro é um sistema Web, enquanto o segundo é um sistema tradicional. Em ambos os projetos, a ferramenta livre *Process Dashboard* [PSPD, 2007], de apoio ao PSP, foi utilizada.

- **Estudo de Caso da Linkway – Sistema Web**

Esse estudo de caso foi baseado no desenvolvimento de um portal Web para uma indústria de tapetes com as seguintes funções básicas: um catálogo dos produtos fabricados, um demonstrativo dos representantes, notícias pertinentes aos produtos e outras informações institucionais sobre a empresa. As informações contidas nesse portal são armazenadas em um banco de dados e são totalmente gerenciadas pela Web através do próprio sistema. O desenvolvimento do código em linguagem Java consumiu aproximadamente 216 horas/homens, distribuídas pelos nove Casos de Uso que compuseram o sistema. Essa aplicação foi inteiramente desenvolvida por apenas um desenvolvedor.

É importante, neste ponto, fazer algumas considerações sobre a implantação da estratégia descrita neste artigo para a realidade de uma pequena empresa. Um projeto de aproximadamente 200 horas, na realidade de uma pequena empresa, é considerado um grande projeto, pois fazendo um cálculo básico em que um programador é capaz de trabalhar em média 5 horas por dia, esse projeto teria a duração de quase dois meses. Por outro lado, a técnica proposta por Karner propõe um NDE de 20 horas/homens para cada Ponto por Caso de Uso. Probasco afirma que realmente esse valor pode variar entre 20 e 30 em grandes empresas como a IBM e a SUN MicroSystem [Probasco, 2002]. Entretanto, com o NDE nesse patamar, um Caso de Uso pode variar entre 240 a 360 horas/homens, o que faria com que a previsão de tempo de desenvolvimento do sistema em questão alcançasse valores entre 2.160 a 3.240 horas/homens.

Percebe-se, na prática das pequenas empresas analisadas neste artigo, que o NDE deve ser ajustado para a realidade de cada empresa. Tal diferença se deve, principalmente, à granularidade assumida para cada Caso de Uso. Uma pequena empresa que trabalha com projetos menores deve assumir uma granularidade menor do Caso de Uso para, assim, ter um detalhamento maior do sistema; por outro lado, uma grande empresa que trabalha em grandes projetos deve assumir uma granularidade maior de cada Caso de Uso, pois quanto maior o sistema, menor deve ser o detalhamento da análise.

Na Linkway, através da interação entre Gerente e Desenvolvedor, foram definidos os Fatores de Complexidade Técnica, Ambientais, o primeiro valor do NDE, e a estimativa inicial de horas/homens para o desenvolvimento, de acordo com a técnica de Pontos por Casos de Uso. Esses valores e o total de horas/homens efetivamente consumidas para o desenvolvimento do sistema estão apresentados na Tabela 3.

Tabela 3 – Valores Iniciais de Planejamento do Sistema Web desenvolvido na Linkway

Descrição	Valor
Fator de Complexidade Técnica	1,10
Fator Ambiental	0,85
Pontos por Caso de Uso Ajustados	93,45
NDE Inicial (Dado Histórico da Empresa)	3 horas/homens
Tempo Total Inicialmente Estimado	*280,40 horas/homens*
Tempo Total Efetivamente Consumido	*216 horas/homens*
Erro na Estimativa	*29,69%*

Na Tabela 4 apresentam-se os nove Casos de Uso do sistema e o resultado da aplicação da estratégia $PCU|_{PSP}$. Cada linha dessa tabela apresenta, em relação a cada Caso de Uso que compõe o sistema, sua complexidade, os Pontos por Caso de Uso (individual e acumulado), as horas que foram gastas para o desenvolvimento desse Caso de Uso (individual e acumulada), o NDE, que corresponde ao nível de esforço real para seu desenvolvimento, o qual é calculado dividindo-se as horas acumuladas pelos Pontos por Casos de Uso acumulados, o total de horas para desenvolvimento do sistema re-estimado agora, com o novo valor do NDE, e a diferença de horas de desenvolvimento, que caracteriza o erro nas horas planejadas inicialmente (280,40), em relação a esse novo valor re-estimado.

Assim, para cada Caso de Uso que é concluído, calculam-se os valores acumulados dos Pontos por Casos de Uso que já foram desenvolvidos e de horas que foram gastas para desenvolvê-los. Tomando-se como exemplo o Caso de Uso 2, depois de calculados os valores acumulados, calcula-se o novo NDE dividindo-se as horas acumuladas (32,96) pelos pontos por casos de uso acumulados (19,74) obtendo-se o valor 1,67. Esse valor significa que o nível de esforço até o momento não é de 3, como foi atribuído inicialmente, mas de 1,67. Com esse novo nível de esforço calcula-se novamente a quantidade de horas para desenvolvimento do sistema todo (156,07) multiplicando-se o novo NDE pelo total de Pontos por Casos de Uso do sistema (93,45). Isso significa que se o desenvolvimento transcorrer dessa forma desse ponto em diante, o planejamento inicial possui um erro em relação ao que efetivamente está acontecendo, de 124,29 horas. Observando-se o Caso de Uso 9, percebe-se que o NDE subiu para 2,31 e que o total efetivo de horas gastas para desenvolver o sistema todo foi de 216,21.

Ressalta-se aqui que essa constante atualização e re-planejamento é baseado na aplicação do PSP, que possibilita uma estabilidade do programador no seu planejamento pessoal, o que impacta em uma maior constância do NDE no projeto como um todo.

Tabela 4 – Resultado da Aplicação da Estratégia PCU|$_{PSP}$ na Linkway

	Etapa 1	Etapas 1 e 2		Etapa 4		Etapa 9	Etapa 2	Etapa 2
Caso de Uso	Complexidade	Pontos por Caso de Uso ajustado		Horas Efetivamente Consumidas para		NDE	Estimado (horas)	Diferença em horas em relação ao
		Individu	Acumulado	Individual	Acumulad			
1	Complexo	14,50	14,50	24,60	24,60	1,70	158,87	121,49
2	Simples	5,24	19,74	8,36	32,96	1,67	156,07	124,29
3	Médio	9,87	29,61	38,20	71,16	2,40	224,29	56,07
4	Complexo	14,50	44,10	53,50	124,66	2,83	264,47	15,89
5	Médio	9,87	53,97	10,20	134,86	2,50	233,63	46,73
6	Simples	5,24	59,22	6,50	141,36	2,39	223,35	57,01
7	Médio	9,87	69,09	29,50	170,86	2,47	230,83	49,53
8	Médio	9,87	78,96	28,50	199,36	2,52	235,50	44,86
9	Complexo	14,50	93,45	16,85	216,21	2,31	215,88	64,48

Outro ponto a ser observado está relacionado à complexidade dos Casos de Uso. Os Casos de Uso de complexidade simples (2 e 6) consumiram uma quantidade de horas compatível com a esperada, menor que os Casos de Uso de complexidade média. Já os Casos de Uso complexos não apresentaram essa consistência, pois apenas o 4 consumiu mais horas que os médios. Assim, o fato do Caso de Uso 1, embora sendo complexo, consumir menos horas do que os de complexidade média gerou um erro grande na primeira estimativa. Esse é outro ponto que está sendo investigado no contexto desta pesquisa. Além de outras variáveis que interferem nessa distorção, o que se pode observar é que à medida que o desenvolvedor se familiariza com o domínio da aplicação, com as técnicas utilizadas, ganhando experiência de uma maneira geral, seu nível de esforço decresce.

- **Estudo de Caso da NBS – Sistema Tradicional**

No Estudo de Caso da NBS, foi analisado o desenvolvimento de um módulo novo para um sistema já existente de informatização de Câmaras Municipais. O resultado da primeira interação entre Gerente e Desenvolvedor está apresentado na Tabela 5. Nesse caso, em decorrência de dados históricos considerou-se o NDE inicial com valor 5. Novamente, apenas um desenvolvedor participou desse projeto.

Tabela 5 – Valores Iniciais de Planejamento do Sistema Tradicional desenvolvido na NBS

Descrição	Valor
Fator de Complexidade Técnica	1,06
Fator Ambiental	0,83
Pontos por Caso de Uso Ajustados	71,26
NDE Inicial (Dado Histórico da Empresa)	5 horas/homens
Tempo Total Inicialmente Estimado	*356,32 horas/homens*
Tempo Total Efetivamente Necessário	*323,45 horas/homens*
Erro na Estimativa	*10,15%*

A interpretação dos dados apresentados na Tabela 6 é análoga àquela feita para a Tabela 4. O que se observou nesse estudo foi que as complexidades atribuídas para os Casos de Uso, de acordo com a técnica proposta por Karner, não mostrou nenhuma distorção, ou seja, os Casos de Uso complexos consumiram mais horas que os médios, que consumiram mais horas que os simples. Assim, o NDE decresceu à medida que o desenvolvedor se familiarizou com a aplicação e que ganhou mais experiência. Outro ponto foi que, como atribuiu-se um NDE inicial de 5 e na prática ele foi igual a 6, houve um erro no planejamento inicial negativo, ou seja, errou-se para menos em 115,30 horas.

Tabela 6 – Resultado da Aplicação da Estratégia PCU|$_{PSP}$ na NBS

Caso de Uso	Etapa 1	Etapas 1 e 2		Etapa 4		Etapa 9	Etapa 2	Etapa 2
	Complexidade	Pontos por Caso de Uso ajustado		Horas Efetivamente Consumidas para		NDE	Estimado (horas)	Diferença em horas em relação ao planejamento
		Individu	Acumulado	Individual	Acumulad			
1	Complexo	14,08	14,08	93,16	93,16	6,62	471,62	-115,30
2	Médio	9,68	23,75	29,83	122,99	5,18	368,97	-12,65
3	Complexo	14,08	37,83	60,58	183,57	4,85	345,79	10,52
4	Simples	5,28	43,11	13,41	196,98	4,57	325,62	30,70
5	Complexo	14,08	57,19	74,29	271,27	4,74	338,04	18,27
6	Complexo	14,08	71,26	52,18	323,45	4,54	323,45	32,87

Em ambos os estudos, o nível de influência dos Fatores Ambientais e de Complexidade Técnica iniciaram com o valor três. Após a Etapa 1 (Figura 1) obteve-se uma solução técnica mais apropriada para o sistema em questão, com base na qual os Fatores de Complexidade Técnica foram redefinidos. Em nenhum dos dois sistemas houve alteração dessa solução. Nessa mesma etapa, os Fatores Ambientais foram acordados entre o Gerente e o Desenvolvedor e, apesar de serem revisados na Etapa 8 (Figura 1), não houve alteração até o término do desenvolvimento. Isso ocorreu devido ao fato dos Desenvolvedores já terem trabalhado em outros sistemas nas respectivas empresas. Porém, novos desenvolvedores normalmente exigem um período maior de ajuste dos Fatores Ambientais. Assim, em decorrência da aplicação desses dois estudos, para o próximo sistema a ser desenvolvido nas empresas, já se tem um novo valor do NDE inicial, a saber, 2,5 para a Linkway e 4,5 para a NBS.

7. Lições Aprendidas

O desenvolvimento deste trabalho proporcionou um importante aprendizado, não só referente à extração da estratégia PCU|$_{PSP}$ com o uso sistemático e conjunto dessas duas técnicas, mas, principalmente, pela aplicação da estratégia nas duas pequenas empresas. Um aprendizado importante é que, através da aplicação da estratégia, é possível identificar as características de cada desenvolvedor, determinando seus pontos fortes e fracos. Uma vez que as características do desenvolvedor ficam mais expostas, o Gerente tem que lidar de maneira pró-ativa para ajudar os desenvolvedores a superarem suas dificuldades. Os erros no planejamento são rapidamente detectados, permitindo que ações corretivas sejam tomadas em tempo hábil para evitar problemas maiores. Outro aspecto interessante decorrente da aplicação da estratégia é que, mesmo sem o Gerente observar de perto o desenvolvimento das rotinas, os desenvolvedores cuidam, sozinhos, de manter o seu desempenho no mais alto patamar possível.

8. Conclusões e Trabalhos Futuros

Apresentou-se neste artigo a estratégia $PCU|_{PSP}$ para ajuste dos Pontos por Caso de Uso com dados provenientes da aplicação do PSP. Essa estratégia aborda tanto a etapa de planejamento, no que diz respeito ao tempo gasto para o desenvolvimento, como também o controle desse planejamento. Essa estratégia confirma o fato relatado na literatura, sobre o Nível de Esforço da técnica de Pontos por Caso de Uso, o qual pode variar bastante da proposta original de Karner (1993). Nos estudos de caso aqui apresentados, o NDE variou próximo do dobro de uma empresa para outra, o que sugere que esse valor deva mesmo ser ajustado para cada empresa, baseando-se também em ajustes para cada desenvolvedor. Os dados históricos das empresas devem servir como valores iniciais, enquanto a contínua aplicação da estratégia é que fornecerá os valores corretos do NDE ajustado para a realidade do projeto em questão. De fato, com o passar do tempo, é natural que o desenvolvedor se torne mais experiente e que isso interfira constantemente na técnica de Pontos por Casos de Uso.

Tanto no estudo de caso da NBS quanto no da Linkway, houve uma relativa estabilidade do NDE dentro do mesmo projeto, o que permitiu o constante acompanhamento do desenvolvimento através da análise dessa variável. Esse acompanhamento foi facilitado porque em ambos os projetos participou apenas um desenvolvedor, embora não seja provável que muitas dificuldades surjam se o número de desenvolvedores aumentar, uma vez que a diferença na aplicação da estratégia se resume na repetição do processo aqui descrito para cada um dos desenvolvedores.

Uma vez ajustados os Fatores Ambientais e de Complexidade Técnica, a variação do NDE não se mostrou significativa. Caso aconteçam grandes variações do NDE, o Gerente tem condições de avaliar objetivamente a razão do desvio ocorrido em relação ao Planejamento e assim, tomar ações corretivas. Tanto a análise da variação do NDE e dos Fatores Ambientais, quanto à análise dos dados coletados pelo PSP fornecem ao Gerente condições para um bom julgamento do desempenho dos desenvolvedores. Esse é um dos fatores diferenciais desta estratégia, pois a detecção de problemas no planejamento e no desenvolvimento é extremamente rápida e eficiente. Assim, o Gerente pode assumir uma postura de "Treinador" (*coach*), indicando diversas atividades para o aperfeiçoamento dos desenvolvedores. Assim, quando participam vários desenvolvedores, cabe ao gerente tratar os dados individuais e, se necessário, reavaliar e realizar o replanejamento em alto nível do sistema como um todo. De maneira geral, a data final corresponde ao planejamento mais demorado dentre os planejamentos individuais.

Portanto, de acordo com os estudos de casos apresentados, foram obtidos indícios de que a estratégia proposta neste artigo dá suporte ao planejamento e ao controle do projeto, permitindo um auto-aperfeiçoamento tanto do Gerente quanto do Desenvolvedor, tornando o desenvolvimento de sistemas mais preciso em relação a prazos e custos. Ressalta-se que no contexto de empresas maiores, talvez a estratégia tenha que ser adaptada, pois, dependendo do tamanho da equipe de desenvolvimento um controle individual não seja factível. Além disso, nos dois estudos apresentados, houve a participação de um único desenvolvedor. Essa é uma limitação desses estudos, embora no caso de equipes pequenas, já que a estratégia visa pequenas empresas, o fato de ter

mais desenvolvedores participando do mesmo projeto não causaria sobrecarga relevante na atividade de controle.

Como trabalhos futuros, pretende-se incluir nessa estratégia outros níveis do PSP, aplicá-la em projetos com mais do que um desenvolvedor e explorar também outros tipos de métricas que poderiam melhorar ainda mais o planejamento. Por fim, pretende-se realizar uma análise da aplicação da estratégia $PCU|_{PSP}$ como uma estratégia facilitadora à implantação de alguns processos do CMMI (*Capability Maturity Model Integration*) [CMMI, 2006], juntamente com abordagens ágeis de desenvolvimento.

Agradecimentos

Agradecemos às empresas Linkway Internet & Telecom e NBS Sistemas pelos estudos de caso presentes neste artigo e ao CNPq pelo apoio financeiro.

Referências

Albrecht A. J. Measuring application development productivity. Proc. of IBM Applic. Dev. Joint SHARE/GUIDE Symposium, Monterey, CA, 1979, pp. 83-92.

Anda, B.; Benestad, H.C.; Hove, S.E. A multiple-case study of software effort estimation based on use case points, Empirical Software Engineering, International Symposium,17-18 Nov/2005.

Capilla, R. & Due, J. C. Light-Weight Product-Lines for Evolution and Maintenance of Web Sites CSMR '03: Proceedings of the Seventh European Conference on Software Maintenance and Reengineering, IEEE Computer Society, 2003, 53

CMMI,, Capability Maturity Model Integration Version 1.2. (CMMI-SE/SW, V1.2 – Continuos Representation), SEI Techinical Report CMU/SEI-2006-TR-008. Disponível em: http://www.sei.cmu.edu/cmmi/, último acesso em: 12/04/2007.

Eickelmann, N. & Hayes, J. Quality time - New year's resolutions for software quality Software, IEEE, 2004, 21, 12-13

El Emam, K.; Shostak, B. & Madhavji, N. Implementing concepts from the Personal Software Process in an industrial setting, Software Process, 1996. Proceedings., Fourth International Conference, 1996, 117-130

Escala, D. & Morisio, M. A metric suite for a team PSP, Software Metrics Symposium, 1998. Metrics 1998. Proceedings. Fifth International, 1998, 89-92

Glass, R.L., Defining Quality Intuitively, IEEE Software,v.15,no.3,Mai/Jun,1998 pp.103–104

Humphrey, W., A Discipline for Software Engineering, Addison-Wesley Professional, 1ª Ed.,Dezembro 1994.

Karner, G. Resource Estimation for Objectory Projects, Objective Systems SF AB, Set., 1993.

Larman, C., "Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development", Addison Wesley Professional, 3ª Ed., 2004.

Morisio, M. Applying the PSP in industry Software, IEEE, 2000, 17, 90-95

Pressman, R. S., Software Engineering: A Practitioner´s Approach, McGraw-Hill, 6ª ed, 2006.

Probasco, L. Function Points and Use Cases, 2002, Disponível em http://www-128.ibm.com/developerworks/rational/library/2870.html, último acesso 02/04/2007.

PSPD, 2007, último acesso em 12/04/2007 em http://processdash.sourceforge.net/

Tsukumo, A.N., et al, Qualidade de Software: Visões de Produto e Processo de Software, II ERI da SBC, 1997, 173-189