UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO
2006.2

ABORDAGEM COMPARATIVA ENTRE MODELOS DE MELHORIA PARA PROCESSO DE SOFTWARE

Autor

Leonardo Monteiro Reinaldo (lmr@cin.ufpe.br)

Orientador

Prof. Ph.D. Alexandre Marcos Lins de Vasconcelos

Co-Orientador

Prof. M. Sc. Sandro Ronaldo Bezerra Oliveira

Recife, Março 2007.

Universidade Federal de Pernambuco
Centro de Informática

LEONARDO MONTEIRO REINALDO

ABORDAGEM COMPARATIVA ENTRE MODELOS DE MELHORIA PARA PROCESSO DE SOFTWARE

Monografia apresentada ao Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação sob orientação do Prof. Ph.D. Alexandre Vasconcelos e co-orientação do aluno de doutorado M. Sc. Sandro Ronaldo Bezerra Oliveira.

Recife, Março 2007.

ASSINATURAS

Este Trabalho de Graduação é resultado dos esforços do aluno Leonardo Monteiro Reinaldo, sob a orientação do professor Alexandre Marcos Lins de Vasconcelos, na participação do projeto “Abordagem Comparativa entre Modelos de Melhoria para Processo de Software”, conduzido pelo Centro de Informática da Universidade Federal de Pernambuco. Todos abaixo estão de acordo com o conteúdo deste documento e os resultados deste Trabalho de Graduação.

Leonardo Monteiro Reinaldo

Alexandre Marcos Lins de Vasconcelos

À minha família.

Agradecimentos

Em primeiro lugar, aos meus pais, Francisco Reinaldo e Sineyde Monteiro, pelo apoio e incentivo que sempre me deram, pois é graças a sua dedicação e esforço que hoje eu posso estar concluindo o curso de graduação em ciência da computação.

A minha namorada, Rita de Cássia, pelo amor e carinho que tem me dedicado, e pela paciência e apoio que teve comigo durante o período de desenvolvimento desse trabalho.

Ao meu orientador, Professor Alexandre Vasconcelos, pela oportunidade e orientação concedidas nesta monografia;

Ao aluno de doutorado e Professor, Sandro Oliveira, pela sua amizade, apoio e co-orientação no decorrer deste trabalho, fornecendo material para pesquisa e estando sempre apto a ajudar.

Aos meus irmãos Alexandre Monteiro, Maxwell Monteiro e Cibele Monteiro e aos meus amigos, pela torcida e auxílio durante a elaboração deste trabalho.

Ao Centro de Informática, pelo acolhimento e excelente formação acadêmica a mim concedidos.

“Um passo a frente e você não está mais no mesmo lugar.”
(Chico Science)

Abordagem Comparativa entre Modelos de Melhoria para Processo de Software

Resumo

O objetivo desse trabalho é realizar uma análise comparativa entre duas abordagens do programa de melhoria de processo de software, focando, principalmente, nos procedimentos propostos por cada uma delas. Uma das abordagens de estudo trata da metodologia do *ProImprove* [Oliveira 06b], um fluxo de trabalho adaptado para atender melhoria de processo de software que possui como base o modelo IDEAL e outros elementos de medição que transcendem os propósitos desse modelo. A outra abordagem é o PRO2PI, definida em [Salviano 06], que propõe uma melhoria de processo de software dirigida por perfis de capacidade de processo. O foco é analisar as duas iniciativas a fim de identificar semelhanças, diferenças e melhorias entre as duas diferentes abordagens.

Palavras-Chave: Análise Comparativa, Processo de Software, Abordagem para Melhoria de Processos de Software, Qualidade de Software.

Índice

LISTA DE TABELAS

LISTA DE FIGURAS

1 INTRODUÇÃO

Neste capítulo serão abordados o contexto de atuação do trabalho, a motivação para o seu desenvolvimento, os objetivos que caracterizam o trabalho realizado, a metodologia usada como linha de execução do mesmo e também sua estrutura.

1.1 CONTEXTO

Atualmente, as organizações necessitam de uma melhoria contínua do seu processo, e essa melhoria tem se mostrado na prática ser uma abordagem viável, eficaz e eficiente para a necessária melhoria das organizações intensivas em software [Salviano 06], visto que o aperfeiçoamento de suas atividades para dispor de serviços e produtos com qualidade torna-se cada vez mais indispensável e propício para os seus clientes.

Portanto, o objetivo desse trabalho é realizar uma análise comparativa entre duas abordagens do programa de melhoria de processo de software, focando, principalmente, nos procedimentos propostos por cada uma delas. Uma das abordagens de estudo trata da metodologia do *ProImprove* [Oliveira 06b], um fluxo de trabalho adaptado para atender melhoria de processo de software que possui como base o modelo IDEAL e outros elementos de medição que transcendem os propósitos desse modelo. A outra abordagem é o PRO2PI, definida em [Salviano 06], que propõe uma melhoria de processo de software dirigida por perfis de capacidade de processo. O foco é analisar

as duas iniciativas a fim de identificar semelhanças, diferenças e melhorias entre as duas diferentes abordagens.

O *ProImprove* é parte integrante do *ImPProS* (Ambiente de Implementação Progressiva de Processos de Software) [Oliveira 05], um ambiente proposto em um plano de trabalho do programa de doutorado submetido e aprovado pelo Centro de Informática da Universidade Federal de Pernambuco.

1.2 MOTIVAÇÃO

Ver como as abordagens realizam os conceitos teóricos de melhoria contínua para promover o aperfeiçoamento das práticas organizacionais a partir de procedimentos direcionados à análise do contexto organizacional.

Para que isso seja possível, faz-se necessário uma análise de como o *ProImprove* funciona, desde a fase inicial do seu ciclo de melhoria até a coleta de lições aprendidas acerca da melhoria realizada. O mesmo esforço faz-se necessário para analisar as fases de melhoria que realiza a abordagem PRO2PI. Isso é necessário uma vez que as organizações, com foco em desenvolvimento de software, requerem estruturas que orientem o seu trabalho de aperfeiçoamento.

1.3 OBJETIVOS

O objetivo deste trabalho é analisar cada uma das fases que promovem o aperfeiçoamento das práticas organizacionais sugeridas nas abordagens antes citadas, bem como suas atividades e princípios, fazer um paralelo

entre elas e, a partir disso, saber o que cada abordagem tem a oferecer no contexto de melhoria contínua de processo de software.

Para alcançar o objetivo principal, o trabalho foi subdividido nos seguintes objetivos:

- Estudo sobre processo software em uma visão geral;
- Estudo do modelo de ciclo de vida do *ProImprove*;
- Estudo do modelo de ciclo de vida do PRO2PI;
- Comparação entre os modelos.

1.4 METODOLOGIA

Como metodologia de pesquisa a ser usada para reunir informações será a pesquisa Bibliográfica, a fim de conhecer o funcionamento e os conceitos inseridos na metodologia do *ProImprove*, principalmente, buscando entender seu fluxo de atividades e também as etapas melhoria definidas pela abordagem PRO2PI, além da análise e comparação dos mesmos.

1.5 ESTRUTURA

Além deste capitulo introdutório, o trabalho está organizado da seguinte forma:

- O capítulo 2 apresenta Visão Geral no que se refere ao Processo de Software.
- O capítulo 3 discute sobre Abordagens de Melhoria Contínua de Processos de Software.

- O capítulo 4 apresenta uma Análise Comparativa entre as Abordagens.
- Finalmente, o capítulo 5 apresenta as conclusões desse trabalho, bem como propostas de trabalhos futuros.

2 PROCESSO DE SOFTWARE: UMA VISÃO GERAL

Inicialmente, esse capítulo introduzirá os conceitos sobre tecnologia de processo de software, os ambientes de desenvolvimento de software e implementação de software centrados em processo, além da definição de processo de software.

2.1 TECNOLOGIA DE PROCESSO DE SOFTWARE

Um processo de software é formado por um conjunto de passos de processo parcialmente ordenados, relacionados com conjuntos de artefatos, pessoas, recursos, estruturas organizacionais e restrições e tem como objetivo produzir e manter os produtos de software finais requeridos [Reis 98].

Os passos que executam um processo podem ser classificados como atividades ou tarefas. As do início são as mais bem gerenciadas, no entanto, as do final são as elementares, que acertadas levam à execução de uma atividade.

O objetivo das atividades é gerar ou modificar um certo conjunto de artefatos. As atividades podem também possuir relacionamentos entre si, estando associadas com papéis, ferramentas e artefatos. Elas tanto podem ser executadas por pessoas, com o suporte de ferramentas, quanto por aplicações de maneira totalmente automatizada, sem a intervenção humana.

Cada artefato é um produto que pode ser criado ou alterado durante um processo. Ele resulta de uma atividade e pode ser usado posteriormente

como base para a mesma ou para outra atividade com o intuito de gerar novos produtos.

O modelo de processo de software é a descrição abstrata do processo de software. As informações de quem, quando, onde e por que os passos são realizados, devem também ser integrado ao modelo de processo de software [Reis 03].

Um modelo de processo pronto para a execução é um modelo de processo instanciado ou processo executável. Portanto, segundo [Pressman 02], um projeto é a instância de um processo, com objetivos e restrições específicos.

2.2 AMBIENTES DE DESENVOLVIMENTO DE SOFTWARE CENTRADOS EM PROCESSO

A definição de Ambiente de Desenvolvimento de Software (ADS) decorreu do fato em que o reconhecimento da comunicação e da coordenação entre todas as ferramentas CASE (*Computer-Aided Software Engineering* – Engenharia de Software Auxiliada por Computador), utilizadas no processo de desenvolvimento e manutenção de software são essenciais para a aquisição de produtos de qualidade. Saber como as ferramentas são definidas, desenvolvidas, adaptadas e integradas tem sido o ponto-chave desta área de conhecimento.

Segundo [Reis 00a], um ADS tem por principal objetivo promover um ambiente onde produtos de software que são de grande porte possam ser desenvolvidos através da integração de um conjunto de ferramentas que

suportam métodos de desenvolvimento, apoiados por uma estrutura que permite a comunicação e cooperação entre as ferramentas.

E assim, a evolução dos ADS prosseguiu com a utilização de várias tecnologias, tais como: processo de software, sistemas distribuídos, inteligência artificial, banco de dados não convencionais, suporte ao trabalho coorporativo, padrões de integração, técnicas de gerência de projeto, entre outras.

Os ADS tiveram uma evolução significativa com a tecnologia de processo de software. Os ADS mais recentes incorporaram a automação do processo de software o que os tornaram ADS centrados em processos (ou orientados a processo). Conhecido na literatura como PSEE – Process-Centered Software Engineering Environment. Tais ambientes formam uma nova geração de ADS, os quais suportam além da função de desenvolvimento de software, também as funções associadas de gerência e garantias da qualidade durante o ciclo de vida do software. Um ADS centrado em processo baseia-se em uma definição explícita do processo de desenvolvimento de software. Por isso o processo de software utilizado na organização deve estar formalizado e ser obedecido.

As ações dos ADS centrados em processos ocorrem de forma mais abrangente no desenvolvimento de software. Segundo [Reis 00a], as funções gerais que podem ser suportadas por um ADS centrado em processo são:

Engenharia de Processos: definição e manutenção de modelos de processo de software. O ADS deve prover facilidades de definição, análise e simulação de processos;

Engenharia de Software: desenvolvimento e manutenção de um produto de software através do seguimento de um processo de software;

Gerência de Projetos: coordenação e monitoramento das atividades da engenharia de software a fim de garantir que o processo está sendo seguido.

2.3 IMPLEMENTAÇÃO DE PROCESSO DE SOFTWARE

Não é uma tarefa trivial mudar o processo de desenvolvimento de software de uma empresa e, quase sempre, leva-se muito tempo para se observar os resultados. [Balduino 02] diz que adotar uma nova ferramenta de desenvolvimento é diferente, pois basta instalá-la, entender o manual do usuário e seguir as instruções de um tutorial. Ou quem sabe ainda fazer um curso sobre ela, levando horas ou talvez dias para realizar isso. Porém, modificar o processo de desenvolvimento de software de uma organização afeta a maneira como os indivíduos trabalham, vêem, e dão valor ao resultado de seu trabalho. No entanto, ter uma mudança no processo de uma organização provoca um impacto muito maior sobre os indivíduos que a compõe do que apenas uma mudança na tecnologia.

No entanto, ter uma mudança desse tipo não se faz da noite para o dia. Deve-se ter o maior cuidado para sempre se fazer um planejamento e um gerenciamento com relação às mudanças. Pode-se obter uma implementação mais adequada fazendo-se valer de uma abordagem de adoção gradual do processo de desenvolvimento e ferramentas de apoio, no qual cada passo seja planejado, executado e avaliado com critério.

Os quatro passos distintos mostrados a seguir descrevem, sob o aspecto da engenharia de software, como se dá a implementação de um novo processo em uma empresa de desenvolvimento de software.

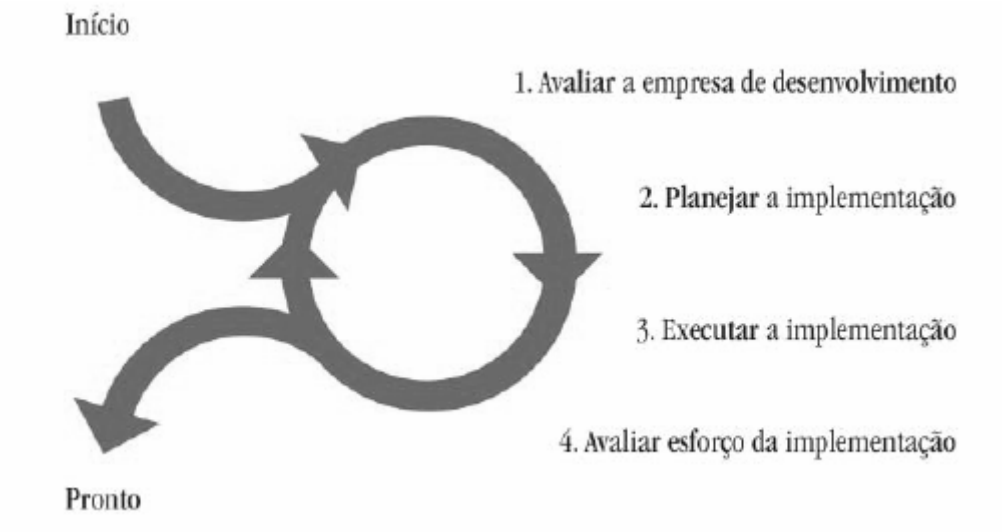

Figura 2-1: Passos para implementar processo em uma empresa [Oliveira 05]

2.4 Definição de Processo de Software

Um processo de software segundo [Humphrey 89] é o conjunto de tarefas de engenharia de software necessárias para transformar os requisitos dos usuários em software. Na definição de um processo de software devem ser consideradas as seguintes informações: atividades a serem realizadas, recursos utilizados, artefatos consumidos e gerados, procedimentos adotados, paradigma e tecnologia adotados, e o modelo de ciclo de vida utilizado [Falbo 98].

Entender melhor as atividades executadas por todos os membros da mesma equipe é permitido pelo trabalho organizado de profissionais de

engenharia de software que possuem em sua organização um processo de software definido [Humphrey 89].

Definir processos que sejam genericamente aplicados ao vasto leque de aplicações esbarra na diversidade de políticas e procedimentos organizacionais, na complexidade de projeto, nos métodos e estratégias de aquisição, no tamanho, nos métodos de desenvolvimento do sistema, entre outros. Fazendo com que eles sejam bem diferentes uns dos outros.

Assim, na definição de um processo é preciso levar em consideração desde características da equipe trabalho, da própria organização até as tecnologias utilizadas, o tipo de software envolvido, o domínio da aplicação e o grau de maturidade da equipe em engenharia de software ([Humphrey 89], [Rocha 01]).

2.5 Considerações Finais

Após a apresentação de uma visão geral dos conceitos e definições relativos ao Processo de Software, o capítulo seguinte mostrará os conceitos de melhoria contínua e medição de processo de software, uma explanação do modelo IDEAL que adota as características da melhoria contínua dos processos organizacionais e, por fim, as iniciativas metodológicas para melhoria de processos de software que serviram de base para o desenvolvimento deste trabalho.

3 ABORDAGENS DE MELHORIA CONTÍNUA DE PROCESSOS DE SOFTWARE

Esse capítulo começa descrevendo os conceitos de melhoria contínua e medição de processo de software, bem como os conceitos e o ciclo de melhoria do modelo IDEAL [Mcfeeley 96] que serviram de base para a compreensão das fases da metodologia *ProImprove*, um dos principais objetos de estudo deste trabalho. E, por fim, apresenta-se as iniciativas metodológicas para melhoria de processos de software que expõem as principais informações para o desenvolvimento desse trabalho.

3.1 MELHORIA CONTÍNUA DE PROCESSOS

Cada vez mais as tecnologias geram novos recursos para tornar nosso dia-a-dia mais fácil e ágil. Muitos deles, logo após sua criação, passam a se tornar indispensáveis, principalmente no tocante ao desenvolvimento de produtos de software e sistemas que, a cada dia, impressionam pela complexidade com que vêm sendo criados.

Todavia, sabe-se que a complexidade gera um aumento das dificuldades no decorrer do desenvolvimento. [Humphrey 89] diz que à medida que aumenta a complexidade dos sistemas menor é a produtividade das equipes, e isso ocorre por causa da baixa qualidade dos sub-produtos que formam o produto final. Além de todos os problemas que surgem quando o produto é liberado para uso, na grande maioria dos projetos constata-se problemas de cronograma e orçamento. Assim, sobreviver num mercado tão competitivo vem sendo um desafio para as organizações

desenvolvedoras de software. Observa-se que a qualidade dos produtos torna-se cada vez mais inerente à sobrevivência das empresas.

Nos últimos anos, observamos que garantir maior qualidade nos produtos e aperfeiçoar o desenvolvimento de softwares provocaram mudanças no enfoque em relação ao processo de software [Rocha 01]. Surge, então, uma nova abordagem onde a prioridade está em garantir a qualidade do processo de produção, mostrando-se como fator decisivo para se obter qualidade do produto final.

A partir dessa mudança de enfoque, intensificou-se as pesquisas sobre o processo de desenvolvimento e várias normas e modelos de qualidade foram definidos para auxiliar na definição e melhoria de processos de software. Dentre os quais vale destacar a Norma ISO/IEC 12207 [ISO 97], o CMM (Capability Maturity Model) [Paulk 95], o SPICE (Software Process Improvement and Capability dEtermination) hoje a norma ISO/IEC 15504 [ISO 98], o CMMI (Capability Maturity Model Integrarion) [CMMI 02], e o MPS.Br (Melhoria do Processo de Softwre Brasileiro) [Softex 05].

Chegou-se a conclusão que para se chegar a patamares cada vez mais altos de qualidade, era preciso melhorar cada etapa do ciclo de desenvolvimento. Contudo, tornar isso realidade era necessário buscar e analisar dados quantitativos que expressassem de forma clara quão melhor o processo está sendo realizado. Assim, mensurar os produtos de software tornou-se pré-requisito indispensável na melhoria de processo [Basili 85].

Os seguintes objetivos listados abaixo foram alcançados a partir de muitas propostas e aplicações práticas [Park 96]:

- Melhorar o entendimento sobre o processo, produto, recursos e ambiente de desenvolvimento e, assim, estabelecer bases para a comparação entre as medições;
- Avaliar o andamento do projeto comparando com dados planejados;
- Fazer previsões sobre o futuro andamento do projeto com base em comportamentos passados;
- Promover melhorias identificando falhas, ineficiências e outras oportunidades para melhorar a qualidade do produto e o desenvolvimento do processo.

Contudo, não é fácil definir, coletar e analisar um conjunto de métricas. A realização destas atividades requer cuidados especiais, pois podem gerar um aumento dos problemas enfrentados durante o desenvolvimento de software. De acordo com [Basili 94], deve-se:

- Concentrar-se em objetivos específicos;
- Ser realizadas sobre todos os produtos, processos e recursos do ciclo de vida;
- Ter seus resultados interpretados com base em características do contexto organizacional e do ambiente.

Finalmente, encontrar um conjunto de modificações que melhore os resultados obtidos seja do processo, seja da organização é bastante desafiador. Na seção a seguir, será descrita uma abordagem para medição e melhoria de processos de software que inclui definição, coleta e análise de métricas aplicadas aos processos de software [Oliveira 06a].

3.2 Abordagem Para a Medição e Melhoria de Processo de Software

Desenvolver aplicativos e sistemas de software não é uma tarefa trivial, requer empenho de equipes de trabalho que em conjunto põem em prática toda sua criatividade e capacidade de desenhar e implementar soluções cada vez mais complexas. Para tanto, seguir uma linha de execução para que um objetivo comum seja alcançado está sendo a alternativa que as organizações estão aplicando para identificar suas diferentes dimensões e encontrar problemas que precisam ser analisados a fim de estabelecer práticas efetivas. Sendo assim, supõe-se que a qualidade adquirida do produto de software quando se realiza um processo no seu desenvolvimento está diretamente ligada à qualidade do processo utilizado para o sua construção e manutenção [Gomes 01]. Seguindo o raciocínio de cultivar a utilização de processo quando acontece algum problema, é preciso que não apenas o defeito encontrado seja corrigido, mas também o processo que permitiu que tal erro acontecesse. Logo, trabalhos futuros não sofrerão os mesmos tipos de correção.

Á propósito, não se conhece um processo que seja genérico, ou seja, aplicável a qualquer tipo de projeto, pois cada um tem sua particularidade. Sabe-se que há diferenças na estratégia de aquisição, tamanho e complexidade do projeto, bem como nas políticas e procedimentos organizacionais que influenciam o produto de software, desde a aquisição até a manutenção.

Além de todas as características citadas acima, existem outras que determinam na definição de um processo, dentre os quais podemos destacar

as tecnologias de desenvolvimento, a experiência e o conhecimento das equipes de trabalho, o porte da empresa, bem como sua cultura e os objetivos do projeto específico [Machado 00].

No entanto, para melhor entender como sistemas complexos de software são produzidos surgiu a necessidade de uma pesquisa mais a fundo sobre a definição e a modelagem de processo de software. Com isso, permite-se gerenciar a execução, melhorar a compreensão, estudar o aperfeiçoamento e automatizar parte da sua execução [Araújo 98].

De início, várias ferramentas se propuseram a resolver o problema de automatização de parte do processo. Daí, reparou-se que a preocupação maior para o sucesso de um projeto está no pessoal e nos processos que o compõe. A tecnologia não se demonstra como um fator tão de risco, talvez seja o menos problemático [Gomes 01]. No entanto, o desenvolvedor deve, sim, se valer das ferramentas, mas elas por si só não são uma garantia de êxito.

As pesquisas mostram que a melhoria da qualidade dos softwares e, por conseguinte, dos processos de software é crescente quando se têm empresas e projetos cada vez maiores. O que vem se observando também é que a definição dos processos deve ser de forma bem dinâmica para poder acompanhar as expectativas de um mercado exigente e competitivo. Para tanto, necessita-se melhorar cada etapa do processo de desenvolvimento, mas para que isso ocorra é preciso obter dados quantitativos que possam expressar o andamento atual do projeto. Várias métricas foram propostas com o objetivo de suprir essa necessidade [Oliveira 06a].

[Rocha 01] baseia-se nas seguintes etapas para a abordagem de medição e melhoria de processos de software:

- Seleção / definição de métricas adequadas, para realizar as medições, com base em objetivos previamente identificados;
- Realização de medições como parte integrante do processo de desenvolvimento de software;
- Análise dos resultados do uso do processo em projetos, apoiada por um sistema com base em conhecimento;
- Realização de estudos empíricos envolvendo medição de processos de software.

Procurou-se, ainda, separar as atividades de medição, análise e melhoria de processo das atividades de desenvolvimento de software. Percebeu-se que uma empresa de desenvolvimento de produtos de software tem que manter o foco no seu objetivo, respeitando os prazos e os custos para tal. A análise do processo deve ser realizada por uma equipe de fora da organização desenvolvedora. Esta fornecerá os dados necessários para se obter o *feedback* dos seus projetos com relação aos processos, incluindo as diretrizes de melhoras dos mesmos [Basili 94].

3.2.1 Definição de Métricas

A tarefa de escolher as métricas que vão compor o conjunto de referências para a medição do processo de software é bastante complicada. Olhar a literatura e decidir qual métrica é melhor para determinada situação não é trivial. A seguir serão mostradas as métricas escolhidas para a

metodologia do *ProImprove*, onde, segundo [Rocha 01] não é um conjunto perfeito de métricas, mas é suficiente.

Seguem três objetivos os quais as métricas foram selecionadas. São eles: caracterizar o projeto e seu contexto; medir, avaliar e sugerir melhorias em um processo de software específico; e realizar estudos empíricos envolvendo medição de processos de software. Veja como ficou a organização do conjunto selecionado [Oliveira 06a]:

- **Tempo**: tempo total do projeto; tempo nas fases de análise, projeto, codificação, teste de unidades feitos por analistas, testes do sistema e testes de homologação; tempo em reuniões de revisão; e tempo em retrabalho;
- **Precisão de estimativa de cronograma**: estimativa de cronograma de todo o projeto, estimativas de cronograma para as fases de análise, projeto, codificação, teste de unidades feitos por analistas, teste do sistema e teste para homologação;
- **Esforço**: esforço total do projeto; esforço nas fases de análise, projeto, codificação, teste de unidades feitos por analistas, testes do sistema e testes de homologação, esforço em reuniões de revisão; esforço em revisões; e esforço em retrabalho;
- **Precisão da estimativa de esforço**: estimativa de esforço para todo o projeto; estimativa de esforço para as fases de análise, projeto, codificação, teste de unidades feitos por analistas, teste do sistema, teste de homologação e revisões;
- **Tamanho do sistema**: número de linhas de código; pontos por função; pontos por casos de uso;

- **Número de erros**: número de erros na especificação de requisitos e no projeto do sistema encontrados em reuniões de revisão, erros no código encontrados nos testes de unidade feitos por analistas;
- **Número de modificações**: número de modificações na especificação de requisitos, projeto ou código após a sua aprovação;
- **Densidade de defeitos**: número de erros somado ao número de modificações em relação ao tamanho do sistema;
- **Rotatividade do pessoal**: percentual de pessoas que saíram, entraram ou mudaram de função durante o desenvolvimento do projeto;
- **Produtividade**: número de linhas de código produzidas por unidade de esforço; pontos por função ou casos de uso por esforço;
- **Deterioração do software**: relação entre o esforço gasto para corrigir os problemas encontrados após a liberação do sistema para o usuário comparado ao esforço gasto da liberação do software para o usuário.
- **Experiência da equipe**: experiência na linguagem de programação, no domínio da aplicação, nas ferramentas, no método e no processo de desenvolvimento, tipo de treinamento em engenharia de software e tempo total de experiência profissional.

O objetivo de caracterizar o projeto, bem como seu contexto de desenvolvimento se enquadra nas métricas que procuram coletar e comparar resultados de projetos similares. Já o objetivo de medir, avaliar e sugerir melhorias em um processo de software recai sobre as métricas que tem por meta avaliar e melhorar a precisão das estimativas, avaliar e melhorar a qualidade dos produtos, e medir e reduzir o retrabalho. Por fim, para [Oliveira 06a] o objetivo de formar uma base de dados para futuros estudos

empíricos rege as métricas que coletam informações para o aumento do entendimento sobre os processos do ciclo de vida de software.

Para garantir que as métricas selecionadas tenham uma utilização correta. É preciso definir sua estrutura operacional, para garantir que pessoas diferentes as implementem de forma coerente e consistente e que os dados sejam interpretados corretamente [Rocha 01].

3.2.2 Coleta de Métricas

Vimos que a melhor maneira de se trabalhar com essas atividades de medição de processo de software é descentralizando as responsabilidades, ou seja, deixando elas fora do comando da organização desenvolvedora que tem como atividade principal o desenvolvimento do produto de software. Logo, para fins de análise posterior, os dados devem ser resgatados de documentos simples e geralmente produzidos pela organização durante todo o processo. Seja ele, um relatório de histórico e/ou uma planilha de atividades [Rocha 01].

Discriminando o que cada um dos documentos acima citados poderiam oferecer, estão: o registro de todas as atividades do processo de desenvolvimento com as datas de início de término, as reuniões de avaliação realizadas e seus resultados (relatório de histórico); os registro de todas as atividades realizadas por cada membro da equipe de desenvolvimento e a gerência do projeto (planilha de atividades) [Oliveira 06a].

3.2.3 Análise das Métricas Coletadas

Aqui, estão definidos dois objetivos que a análise das métricas seguem, são eles: analisar os resultados buscando melhorias para o processo de software e realizar estudos empíricos de comparação das métricas colhidas em medições efetuadas nos diversos projetos [Rocha 01].

A análise aqui descrita será sustentada por um sistema que possui uma base de conhecimento construído (algum ambiente de desenvolvimento de software descrito na literatura especializada) que foi gerada pelo conhecimento de especialistas, dado os resultados coletados por meio de medições. As informações oriundas dessa atividade formam um conglomerado de recomendações para a melhoria de processo. Especialistas podem definir esse conglomerado de recomendações e verificar a existência de resultados não aceitáveis dentro do processo de software ou com relação ao contexto em que é realizado o desenvolvimento. Caso essa situação ocorra, deverá haver alguma inferência no sistema com base no conhecimento sobre as características do processo [Oliveira 06a].

3.3 Modelo IDEAL Para a Melhoria Contínua de Processos

Cada vez mais, organizações reconhecem que é preciso orientação especial na execução das suas atividades no instante em que métodos, processos e ferramentas de engenharia de software são escolhidos. Muitos esforços de melhoria, incluindo melhoria do processo de software, contínuo gerenciamento de riscos, ou a introdução de um novo ambiente de desenvolvimento, são tão complexos, e seus efeitos tão de longo alcance,

que eles requerem uma abordagem especializada e sistemática para gerenciar o ciclo de vida de adoção da tecnologia. O Software Engineering Institute (SEI) desenvolveu e refinou o Modelo IDEAL - Initiating, Diagnosing, Establishing, Acting & Learning - para ajudar a satisfazer essa necessidade.

O modelo IDEAL, como concebido originalmente, era um modelo de ciclo de vida para a melhoria do processo de software baseada no Capability Maturity Model (CMM) [Paulk 93], e por esta razão o modelo usou termos da melhoria do processo.

O Modelo IDEAL fornece uma abordagem usável e fácil de entender para o aprimoramento contínuo por mostrar os passos necessários para se estabelecer um programa de melhoria de processos bem sucedido. Seguir as fases, atividades e princípios do Modelo IDEAL tem se provado benéfico em muitos esforços de melhoramentos. O modelo fornece uma abordagem de engenharia disciplinada para o aprimoramento, foca no gerenciamento do programa de aprimoramento, e estabelece a fundação para uma estratégia de melhoramento de longo prazo. O modelo consiste em cinco fases e são apresentadas na Figura 3-1, são elas:

- *Initiating*: esta é a fase onde a infra-estrutura inicial da melhoria é estabelecida, os papéis e as responsabilidades para esta infra-estrutura são inicialmente definidos, e os recursos inicias são atribuídos. Este fase foca no estímulo para a melhoria do processo de software, na definição do contexto e do patrocinador e no estabelecimento da infra-estrutura inicial para suporte da melhoria [Oliveira 06a];

- *Diagnosing*: o foco nesta fase é desenvolver um entendimento mais completo do trabalho de melhoria. Durante esta fase duas estratégias para a melhoria do processo da organização são definidas: o estado atual e o estado futuro desejado. Estes estados organizacionais são usados para desenvolver uma abordagem para a prática de melhoria do negócio. Assim, o objetivo é caracterizar o estado atual e o estado futuro desejado da organização, e desenvolver recomendações de como proceder nas fases subseqüentes [Oliveira 06a];
- *Establishing*: o objetivo desta fase é desenvolver um plano de trabalho detalhado. As prioridades de quais práticas organizacionais serão melhoradas são ajustadas para refletir as recomendações feitas durante a fase *Diagnosing*. Este passo tem a finalidade de colocar prioridades para as alterações, desenvolver uma estratégia para realização do trabalho, identificar recursos disponíveis, e desenvolver um plano de implementação detalhado (onde as ações, marcos de referência e as responsabilidades são incorporados em um plano de ação) [Oliveira 06a];
- *Acting*: as atividades desta fase servem para ajudar uma organização a implementar o trabalho realizado nas três fases anteriores (*Initiating*, *Diagnosing* e *Establishing*). Assim, o foco é criar a melhor solução que atenda as necessidades organizacionais identificadas, testar a solução criada através de um projeto piloto, modificar a solução para refletir o

conhecimento, experiências e lições obtidas no projeto piloto, e implementar a solução em toda a organização [Oliveira 06a];

- *Learning*: nesta fase a experiência obtida com execução do modelo IDEAL é revista para determinar se os esforços conseguiram atingir os objetivos pretendidos, e como a organização pode executar a mudança mais eficazmente e / ou eficientemente no futuro. Para isso, as lições são coletadas, analisadas e documentadas, as necessidades identificadas na fase Initiating são reexaminadas para ver se foram atendidas, e propostas de alterações para melhoria futura devem ser fornecidas [Oliveira 06a].

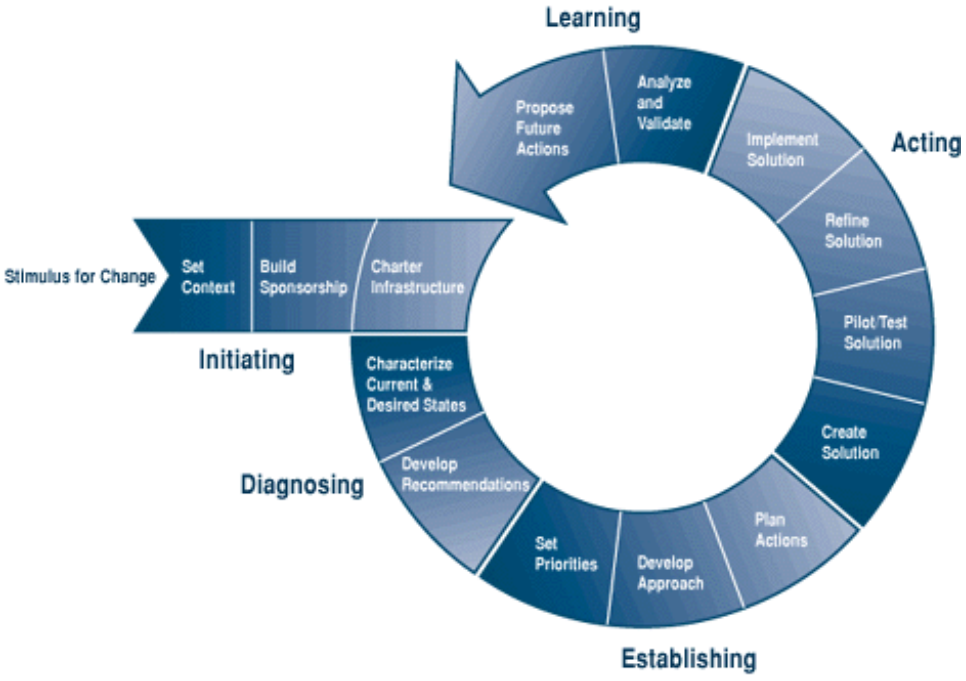

Figura 3-1: Modelo IDEAL [Mcfeeley 96]

3.5 INICIATIVAS METODOLÓGICAS PARA MELHORIA DE PROCESSOS DE SOFTWARE

A seguir, serão mostradas as abordagens que serviram de base para a análise comparativa proposta neste trabalho. Revelando os conceitos inerentes a cada uma delas, bem como suas peculiaridades com relação às suas atividades.

3.4.1 PROIMPROVE

O *ProImprove* faz parte de um ambiente cooperativo formado por nove ferramentas principais. Sua função é possibilitar a execução sistemática das atividades de melhoria contínua de processo de software. O *ImPProS*, ambiente em que o *ProImprove* está inserido é um projeto de iniciativa do Centro de Informática da UFPE – Universidade Federal de Pernambuco com a parceria da UNAMA – Universidade da Amazônia, financiado pelo CNPq – Conselho Nacional de Desenvolvimento Científico e Tecnológico, que visa a criação de um ambiente de apoio à implementação de um processo de software em uma organização de forma progressiva. O termo “progressiva” decorre do fato de que a implementação do processo é aperfeiçoada com as experiências aprendidas na sua definição, simulação, execução e avaliação.

Em um trabalho de graduação [Correia 05], desenvolvido por Rafael Correia, o modelo IDEAL acima citado foi adaptado para ser utilizado no ambiente de implementação de processo de software. Como parte desse trabalho foi desenvolvida toda a metodologia do *ProImprove*, além de um protótipo dessa adaptação. Num outro trabalho de graduação [Matos 05],

desenvolvido por Pedro Matos, o *ProImprove* foi estendido para oferecer suporte a métricas para validar se uma dada melhoria atingiu seus objetivos e avaliar também o processo de melhoria em si.

3.4.1.1 Adaptação do Modelo IDEAL

O trabalho realizado por [Correia 05] consistiu de uma análise de todas as fases do modelo visando a contextualização dessas para o ambiente de implementação de processos de software. Sua adaptação foi alcançada e baseada em dois objetivos principais: a utilização dos conceitos de melhoria contínua no contexto do ambiente e estudar como o modelo IDEAL se comportaria na implantação de melhorias em processos de software dentro desse contexto.

A adaptação proposta definiu três visões distintas:

Gerenciamento: responsável por planejar como as informações seriam manipuladas no ambiente pelos usuários.

Uso (Execução): responsável pela por manter informações relativas à execução das melhorias.

Avaliação: responsável por todas as atividades onde é necessário avaliar alguma informação a fim de inferir um resultado.

A adaptação definiu ainda dois perfis de pessoas responsáveis pela melhoria de um processo de software:

Projetista de Processo: responsável por descrever a melhoria, identificar a motivação para que esta seja executada, definir o contexto organizacional, alocar a infra-estrutura necessária, definir os procedimentos necessários e fornecer uma análise e validação da solução implantada.

Gerente de Processo: responsável por definir um plano de ações, criar a solução, testá-la, refiná-la e por fim implantá-la.

A próxima seção fornece uma rápida visão do fluxo das atividades envolvidas na adaptação proposta.

3.4.1.2 Fluxo de Atividades do *ProImprove*

A adaptação pode ser visualizada como um fluxo de atividades distribuídas entre o projetista do processo, o gerente do processo e o patrocinador que fornecerá o apoio necessário para que melhoria ocorra. A Figura 3-2 ilustra esse fluxo de atividades.

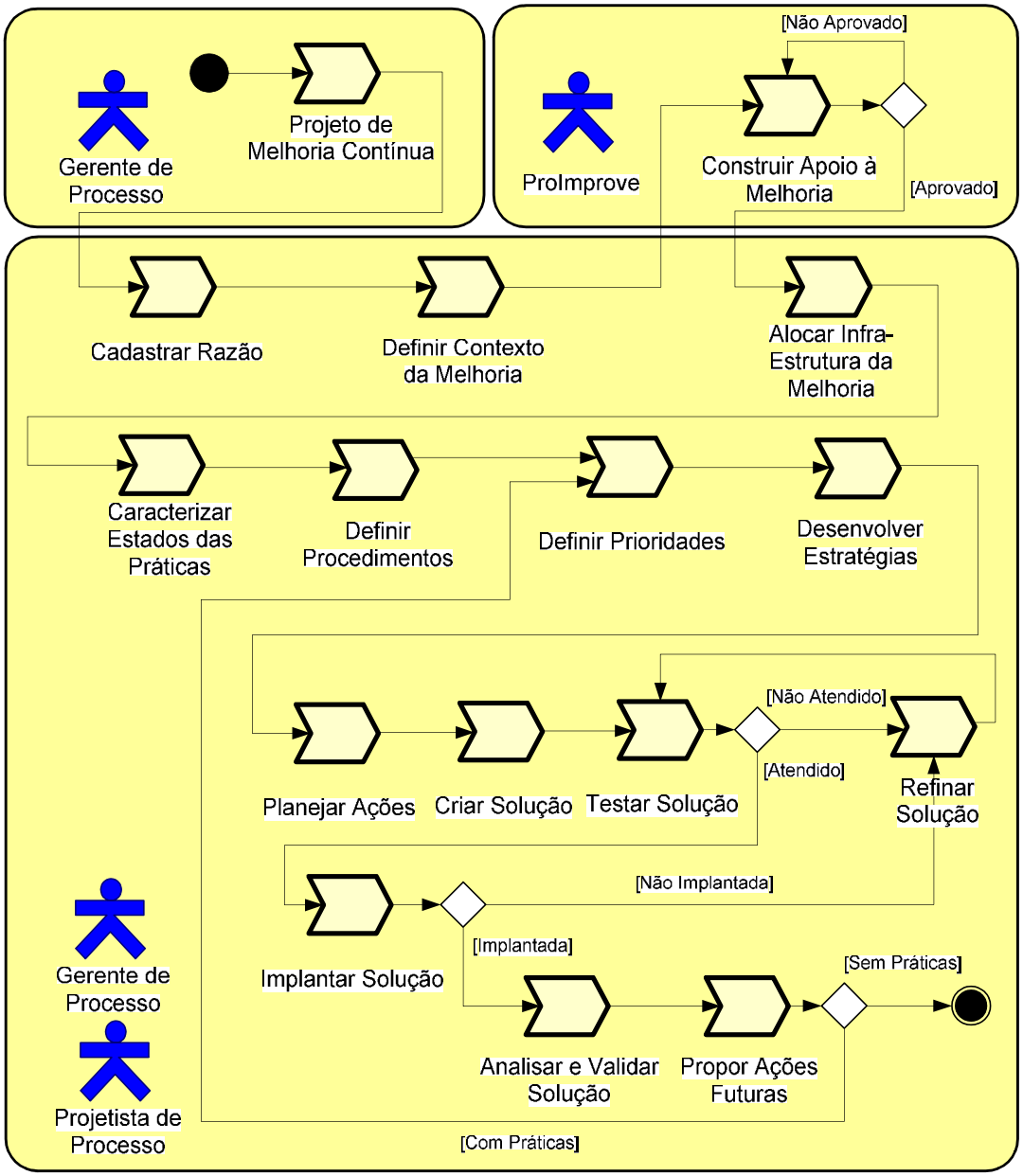

Figura 3-2: Fluxo de Atividades do *ProImprove* [Oliveira 06a]

A atividade *Cadastrar Razão* corresponde à fase de definição do estímulo da mudança no modelo IDEAL. É nessa fase onde o projetista definirá quais razões farão a empresa investir na melhoria de seus processos, bem como quais as práticas que farão a melhoria acontecer. Já na atividade *Definir Contexto*, o projetista descreverá os objetivos que a melhoria pretende buscar, quais os benefícios que a melhoria trará para a organização

e como ela afetará os trabalhos existentes que estão relacionados com a melhoria.

Para que o projeto siga a diante, na atividade *Construir Apoio* algum membro da alta administração da organização deve assumir o papel de patrocinador, formalizando o seu interesse pela melhoria e garantindo o apoio necessário para que a mesma seja executada.

Após receber o apoio necessário, o projetista do processo deverá executar a atividade *Alocar Infra-Estrutura*, nesse momento precisa-se saber se há alguma restrição antes que a melhoria seja executada. Há que se definir com o que cada pessoa alocada para essa empreitada se comprometerá e quais esforços serão necessários para a realização da melhoria. Uma vez montada a infra-estrutura, o projetista terá de concluir a atividade *Caracterizar Estados*. O foco dessa atividade é definir os estados atual e desejado para cada prática do processo software que sofrerá modificação para servir de base para a validação da melhoria. Além disso, nesta atividade, foi adicionada a habilidade do projetista especificar um conjunto de métricas.

A partir daí, definido claramente o estado em que a prática está e o estado que ela pretende alcançar, o projetista caracterizará toda configuração das métricas que avaliará as práticas decorrentes da melhoria, primeiramente realizando a ação de *Configurar Questões para Validação das Práticas*, nela o projetista informará as questões cujas respostas influenciarão na validação da prática e ajudará a identificar se o estado desejado para a prática foi alcançado. Seguindo essa configuração temos, *Configurar Análise das Questões*, onde o projetista cadastrará todos os

valores possíveis para todas as questões inseridas na etapa anterior. Esses valores podem ser qualidades, no caso de métricas qualitativas, podem ser números ou intervalos numéricos, no caso de métricas quantitativas. Para cada valor inserido, deve ser definida também uma interpretação para a métrica: Prática Validada ou Prática Não Validada. Concluindo essa especificação de métricas, o projetista irá *Configurar Validação da Prática.* Nela, deverá se escolher o percentual mínimo de questões cuja inferência deva ser Prática Valida para que o ambiente possa efetivamente inferir que a prática foi validada.

Definida todas as métricas para o projeto de melhoria, o passo seguinte é executar a atividade *Definir Procedimentos*, na qual especificará as recomendações que irão servir de apoio para as atividades posteriores.

A próxima etapa a ser executada pelo projetista é *Definir Prioridades*. Nessa etapa, subgrupos são formados para se ter uma maior flexibilidade quanto à execução da melhoria. São definidas quais práticas serão focadas inicialmente e na seqüência da melhoria. Após definir prioridades, o projetista deverá *Desenvolver Estratégias*, ou seja, criar um plano estratégico de execução da melhoria.

Uma vez definido o plano estratégico, o gerente do processo poderá dar início à atividade *Planejar Ações*. O foco dessa atividade é a definição de um plano de ações para implantação da melhoria, contendo um cronograma de tarefas, milestones e pontos de decisão. O plano de ações inclui ainda um gerenciamento de recursos, atribuição de responsabilidades, definição de métricas, mecanismos de tracking, planejamento de estratégias e gerenciamento de riscos.

Após a definição do plano de ações, o gerente poderá iniciar a atividade *Criar Solução*. A solução é composta pelas ferramentas e processos que serão utilizados; conhecimentos e habilidades necessárias; ajuda externa que será requerida e quaisquer outras informações adicionais. Após a criação da solução esta deverá passar pelas atividades de *Testar Solução* e *Refinar Solução*. Quando a solução for finalmente validada, esta será implantada na atividade *Implantar Solução*.

Uma vez implantada a solução, através da atividade *Analisar e Validar Solução* o projetista do processo poderá saber se para cada prática escolhida para a melhoria foi validada ou não. Essa atividade foi estendida para que o projetista pudesse executar a ação de *Analisar Questões de Validação da Prática*, onde se deve fornecer um valor válido para cada uma das questões de validação da prática, valores que servirão como resposta correta para o parâmetro de avaliação. E também, observar as informações em *Sugerir Resultado de Validação da Prática,* onde os valores mostrados serão inferidos a partir dos dados inseridos na fase *Caracterizar Estados*. Informa-se a resposta e a interpretação para cada questão de validação da prática, bem como, a sugestão de validação da prática e o seu percentual de inferência. Além disso, há ainda a ação *Analisar Características de Execução da Sub-melhoria*, onde se irá avaliar a melhoria como um todo. Deve-se fornecer um valor válido para cada uma das suas características, para a partir daí o *ProImprove Inferir Questões de Execução da Sub-melhoria*. O valor de inferência será informado podendo o projetista aceitar ou não.

Finalizando o fluxo, o projetista executará a atividade *Propor Ações Futuras*. Após registrar as lições aprendidas durante a sua execução elas

formarão uma base de conhecimento do método de implantação de melhorias, permitindo que boas práticas sejam reconhecidas e erros sejam evitados no futuro. Se julgar necessário, ele poderá ainda propor as ações futuras que irão trazer mais eficiência e confiabilidade à execução de melhorias que vierem adiante.

3.4.2PRO2PI

A abordagem PRO2PI definida na Tese de Doutorado de Clênio Figueiredo Salviano pela Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas propõe uma engenharia de processo dirigida por perfis de capacidade de processo. A seguir, descreveremos como são estruturadas suas fases e atividades. Uma visão geral da utilização da abordagem PRO2PI é pode ser vista no Apêndice A.

3.4.2.1 Fases e Atividades do PRO2PI

O objetivo principal das atividades de definição e utilização de PRO2PI é definir e utilizar PRO2PI. O produto de entrada e de saída é o perfil de capacidade de processo. As atividades podem ser consideradas como uma implementação das práticas base definidas para o processo de estabelecimento de processo, que estão descritas no Apêndice B.

A seguir, as fases e as atividades que compões a abordagem PRO2PI serão expostas para que sejam melhor compreendidas.

De acordo com PRO2PI a primeira fase é *Iniciar Ciclo de Melhoria*, onde se faz o levantamento de informações gerais sobre a empresa, incluindo o contexto e objetivos estratégicos, metodologias existentes,

tecnologias e projetos existentes, a definição das Metas de Melhoria, a definição do Perfil de Capacidade de Processo, a revisão e detalhamento do Programa de Melhoria de Processo e a definição do Plano Preliminar da Avaliação das Práticas Correntes.

Seguindo esse fluxo, temos a fase *Avaliar Práticas Correntes,* onde se executa as atividades de revisão do Plano da Avaliação das Práticas Correntes, o treinamento da equipe de Avaliação sobre o Processo da Avaliação, a adaptação das Práticas de Referência para a Avaliação, o levantamento e validação de dados para a avaliação dos Projetos e Processos selecionados, a derivação e elaboração dos Resultados da Avaliação, a apresentação dos Resultados da Avaliação e a revisão do Perfil de Capacidade de Processo.

A fase *Planejar Ações de Melhoria*, é realizada com as atividades de treinamento de pessoas selecionadas sobre o Processo de Planejamento da Melhoria e elementos para sua implantação, definição da Estratégia para Implantação da Melhoria, definição das Ações de Melhoria, elaboração do Plano de Melhoria de Processo, definição das responsabilidades e necessidades de compromissos, pela elaboração do Plano de Preparação da Organização, revisão do Plano de Melhoria de Processo por pessoas selecionadas, estabelecimento das responsabilidades para execução do Plano de Melhoria de processo, incluindo as funções de gerente, grupo de processo e grupos de implementação de processo, estabelecimento dos compromissos para execução do Plano, atividades para consolidação de planejamento, revisão das atividades e resultados da subfase anterior, atualização do Plano,

treinamento de pessoas selecionadas sobre elementos para implantação do Plano e apresentação para início da implantação das ações de melhoria.

A próxima, é a fase *Realizar Ações de Melhoria*, que realiza as atividades descritas no Plano, monitora as atividades conforme o Plano, revisa o Plano e realiza ações corretivas, quando necessário.

Para *Preparar institucionalização da melhoria* tem-se que revisar o resultado da Melhoria e elaborar o Plano de Institucionalização.

Por fim, temos a fase *Institucionalizar a melhoria*, onde se promove a realização das atividades descritas no Plano de Institucionalização, o monitoramento das atividades conforme o Plano de Institucionalização, a Revisão do plano de Institucionalização e a realização das ações corretivas, caso seja necessário.

3.5 CONSIDERAÇÕES FINAIS

Neste capítulo foram apresentados os conceitos de melhoria contínua e medição de processo de software, uma descrição das fases do modelo IDEAL e as atividades das duas iniciativas metodológicas para melhoria de processos de software que são as bases para a análise comparativa que será mostrada no capítulo seguinte.

4 ANÁLISE COMPARATIVA ENTRE AS ABORDAGENS

Este capítulo tem por objetivo traçar uma análise comparativa com diversos aspectos das abordagens mencionadas no capítulo anterior, na tentativa de ilustrar semelhanças e diferenças que possam justificar uma abordagem mais ampla. Talvez esta análise venha auxiliar em futuras pesquisas, com o surgimento de novos elementos que estendam o trabalho aqui realizado.

4.1 COMPARATIVO

Através do comparativo realizado entre os fluxos de atividades, ver Apêndice C, contemplados pelos dois ciclos de vida de melhoria de processo de software, o primeiro mostrado na Figura 3-1, que sofreu adaptações quando aplicado ao *ProImprove* e o segundo descrito na seção 3.4.2.1, descobriu-se quais atividades pertenciam a um determinado ciclo e não pertenciam ao outro e vice-versa. As Figuras 4-1 e 4-2 abaixo sintetizam esses dois ciclos.

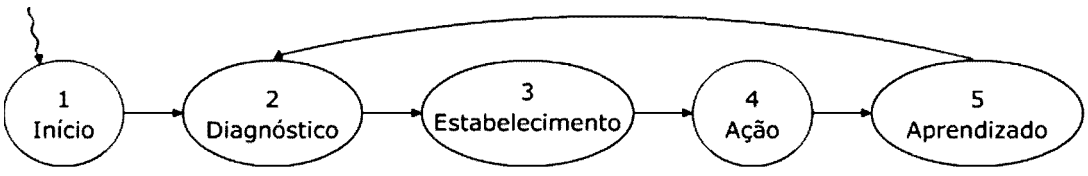

Figura 4-1: Ciclo de vida do modelo IDEAL [Salviano 06]

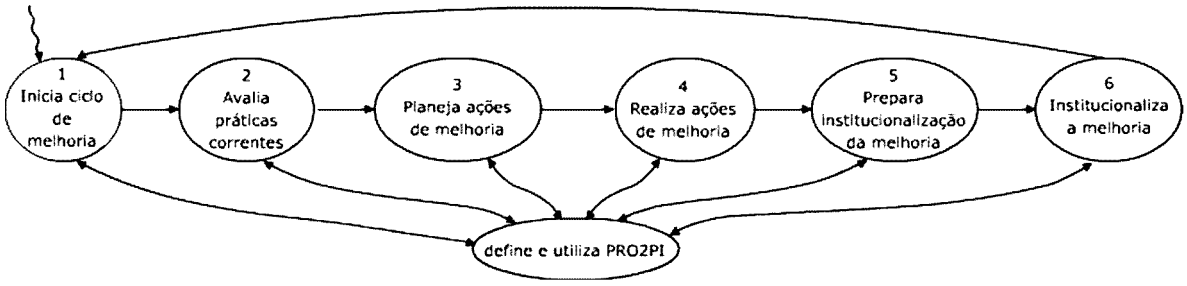

Figura 4-2: Ciclo de vida da abordagem PRO2PI [Salviano 06]

Atividades do PRO2PI não contempladas pelo *ProImprove*:

- Revisão e detalhamento do Programa de Melhoria de Processo:
    - Esta atividade de revisão não é contemplada pelo *ProImprove* porque as revisões são feitas continuamente e automaticamente pelo sistema. O que não acontece com o PRO2PI, visto que a dinâmica de atividades é realizada com relatórios escritos, ou seja, manualmente.
- Treinamento da equipe de avaliação sobre o processo da avaliação, Treinamento de pessoas selecionadas sobre o processo de planejamento da melhoria e elementos para sua implantação e Treinamento de pessoas selecionadas sobre elementos para implantação do Plano:
    - Não há atividade de treinamento, pois o treinamento no *ImPProS* é realizado antes de entrar no ciclo de melhoria de processo. Antes de qualquer coisa, há um treinamento para a execução da metodologia do *ProImprove* que permitirá aos usuários realizar as atividades de melhoria de processo que nela estão contempladas.

- Apresentação para início da implantação das ações de melhoria:
    - Esta atividade só existe no PRO2PI porque as atividades não são automatizadas nem integradas em um ambiente (*ImPProS*) que permita a uma visualização de tudo que está sendo executado, ou seja, um ciclo de vida voltado para automatizar as fases de implementação do processo de software (definição, simulação, execução e avaliação).

Atividades do *ProImprove* não contempladas no PRO2PI:

- Configurando Validação da Prática:
    - Como o *ProImprove* realiza todas suas atividades automatizadas, bem como seu processo de métricas para medição das melhorias, há uma atividade onde o usuário deverá escolher o percentual mínimo de questões cuja inferência deva ser Prática Valida para que o ambiente possa efetivamente inferir que a prática foi validada. Esta atividade é própria do *ProImprove*.
- Definir prioridades:
    - Atividade própria do *ProImprove* onde são informadas quais as práticas que serão focadas inicialmente na seqüência da melhoria, ou seja, a melhoria é tratada de forma particionada, garantindo, assim, o seu objetivo de ser contínuo. No PRO2PI não há essa flexibilidade quanto à execução da melhoria, ou seja, um único grupo de processos escolhidos é melhorado.

4.2 RESULTADOS

Com base na análise das duas abordagens observa-se a adequação do modelo de ciclo de vida para o *ProImprove* como mostra na Figura 4-3, onde as atividades de definição do Perfil de Capacidade de Processo, definição do Plano Preliminar da Avaliação das Práticas Correntes, revisão do Plano da Avaliação das Práticas Correntes e revisão do Perfil de Capacidade de Processo que em comparação ao PRO2PI estariam no fluxo de atividades do *ProImprove*, seriam executadas na base do ambiente *ImPProS*, ou seja, fora do seu fluxo normal.

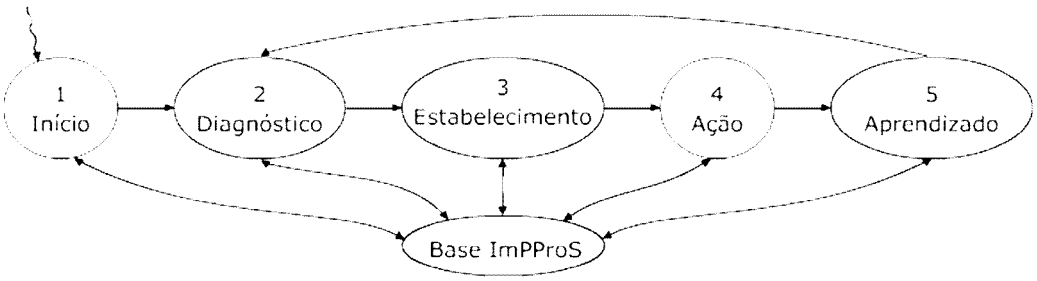

Figura 4-3: Modelo de ciclo de vida para o *ProImprove*

A partir da observação da tabela comparativa disponível no Apêndice C, observou-se que os dois modelos são adequados para executar a tarefa de melhoria contínua de processo de software. O *ProImprove*, usando como filosofia e base o Modelo IDEAL, bem como seu modelo de métricas e, o PRO2PI, focando na Capacidade de Processo, onde apesar das diferenças conceituais ambos possibilitam agregar valor ao trabalho que cada um se propõe.

4.3 CONSIDERAÇÕES FINAIS

Neste capítulo, foram apresentados o comparativo e os resultados obtidos a partir análise feita entre os fluxos de melhoria dos modelos apresentados. O próximo capítulo mostrará a conclusão desse trabalho.

5 CONCLUSÃO

Neste capítulo serão apresentadas às considerações finais desse trabalho, descrevendo em linhas gerais o sumário do trabalho, e os possíveis trabalhos de extensão propostos a este.

5.1 SUMÁRIO DO TRABALHO

Uma abordagem para melhoria de processo de software é uma orientação para um conjunto de ações para a melhoria de processo em uma organização intensiva em software. A vantagem disso é que o processo de melhoria de processo garante a melhoria contínua da eficiência e eficácia da organização por meio dos processos que estão sendo utilizados e mantidos de forma alinhada às necessidades de negócio.

Neste trabalho foi apresentada a análise de duas abordagens de melhoria de processo de software, focando, principalmente, as atividades de melhoria de cada uma delas. O ciclo de vida do *ProImprove* mostrou-se bastante completo, sobretudo, porque oferece a automatização de todo o processo além de oferecer, no seu fluxo, mecanismos onde se definem métricas para avaliar o esforço da melhoria. Já nas fases da abordagem PRO2PI, observa-se que, diferentemente do *ProImprove*, as revisões, os treinamentos e as apresentações são feitas no decorrer do fluxo de atividades e mostram-se desfavorecidos por não prover um processo automatizado, nem uma configuração para validação das práticas. Apesar disso, através dessa análise, percebeu-se que funcionalmente as duas

propostas estão compatíveis dentro do que eles focam em melhoria contínua.

Espera-se então, que o resultado deste trabalho seja utilizado como base para outras pesquisas que compartilhem de objetivos semelhantes e sirva de inspiração para o surgimento de novos elementos na análise comparativa.

5.2 TRABALHOS FUTUROS

Como proposição para a extensão deste trabalho, pode-se relacionar atividades cujo interesse siga a linha de melhoria contínua do processo de software, a saber: experimentação das duas propostas de melhorias contínuas usando como base um cenário único para análise dos resultados obtidos e assim uma verificação comparativa mais direcionada a aplicação das duas abordagens em um contexto real.

REFERÊNCIAS

[Ahern 01] Ahern D. M., Clouse A., e Turner R., CMMI Distilled: A Practical Introduction to Integrated Process Improvement, SEI Series in Software Engineering, Addison- Wesley, 2001.

[Araújo 98] Araújo, M. A. P., Automatização do Processo de Desenvolvimento de Software nos Ambientes Instanciados pela Estação TABA, Orientadora: Ana Regina Rocha. Dissertação de Mestrado, COPPE/UFRJ, 1998.

[Basili 85] Basili, V. R., Quantitative Evaluation of Software Methodology, Keynote Address, First Pan Pacific Computer Conference, Melbourne, Australia, 1985.

[Basili 94] Basili, V. R., Kan, S.H. e Shapiro, L.N., Software Quality: An Overview from the Total Quality Management Perspective, IBM Systems Journal, 1994.

[Chrissis 03] Chrissis M. B., Konrad M. e Shrum S., CMMI: Guidelines for Process Integration and Product Improvement, Addison-Wesley Pub Co, 2003.

[CMMI 02] CMMI Product Team, Capability Maturity Model Integration (CMMI) for Software Engineering Version 1.1, Staged Representation, Carnegie Mellon, USA, 2002.

[Correia 05] Correia, R., Avaliação (adaptação) do modelo IDEAL para um ambiente de implementação de processo de software. Trabalho de conclusão da graduação em Ciência da Computação, Universidade Federal de Pernambuco, 2005.

[Falbo 98] Falbo, R. A., Integração de Conhecimento em um Ambiente de Desenvolvimento de Software, Orientadora: Ana Regina Cavalcanti da Rocha. Tese de Doutorado, COPPE/UFRJ, 1998.

[Florac 99] Florac e Carleton, Measuring the Software Process – Statistical Process Control for Software Process Improvement, 1999.

[Gomes 01] Gomes, Augusto G., Avaliação de Processos de Software baseada em Medições, Orientadora: Ana Regina Rocha e Káthia Marçal de Oliveira. Dissertação de Mestrado, COPPE/UFRJ, 2001.

[Humphrey 89] Humphrey, Watts S., Managing the Software Process, The SEI Series in Software Engineering. Addison-Wesley, 1989.

[ISO 97] NBR ISO/IEC 12207:1995, Tecnologia de Informação – processos de ciclo de vida de software. Associação Brasileira de Normas Técnicas, 1997.

[ISO 98] ISO/IEC TR 15504, Information Technology – software process assessment. International Organization for Standardization, 1998.

[ISO 04] ISO/IEC 15504 - Information Technology - process assessment – Part 1: Concepts and Vocabulary, 2004.

[Machado 00] Machado, L. F. C., Santos, G., Oliveira, K. M. e Rocha, A. R., Def-Pro: Uma ferramenta para apoiar a definição do processos padrão. XI CITS – Conferência Internacional de Tecnologia de Software. Curtiba, PR, 2000.

[Mcfeeley 96] Mcfeeley, B., IDEALSM: A User's Guide for Software Process Improvement. Software Engineering Institute Handbook. Carnegie Mellon University. CMU/SEI-96-HB-001. 1996.

[Oliveira 05] Oliveira S. R. B. e Vasconcelos A. M. L., Proposta de um ambiente de implementação de processo de software, Proposta de doutorado submetida ao Centro de Informática da Universidade Federal de Pernambuco, Recife, PE, 2005.

[Oliveira 06a] Oliveira, S. R. B., Processo de software: princípios, ambientes e mecanismos de execução, Exame de qualificação do programa de doutorado do CIn/UFPE, orientado pelo prof. Alexandre Vasconcelos, Recife, PE, 2006.

[Oliveira 06b] Oliveira, S. R. B., Vasconcelos, A. M. L. "A Continuous Improvement Model in *ImPProS*", Proceedings on COMPSAC Fast Abstract

Session - 30th Annual International Computer Software and Applications Conference, Chicago, USA, 2006.

[Park 96] Park, R. E. & Goethert, W. B. & Florac, W. A., Goal-driven software measurement – A guidebook, Pittsburgh, PA, CMU/SEI-96-HB-002, Software Engineering Institute, Carnegie Mellon University, 1996.

[Paulk 93] Paulk, M. C., Chrissis, M. B., Curtis, B. e Weber, C. V., Capability Maturity Model for Software, Version 1.1. Technical Report CMU/SEI-93-TR-024. Software Engineering Institute - Carnegie Mellon University, 1993.

[Paulk 95] Paulk, M. C., Chrissis, M. B., Curtis, B. e Weber, C. V., The Capability Maturity Model: Guidelines for Improving Software Process, Addison – Wesley, USA, 1995.

[Pressman 02] Pressman, R., Engenharia de Software, São Paulo: Makron Books, 2002.

[Reis 98] Reis, C. A. L., Um Gerenciador de Processos de Software para o Ambiente PROSOFT, Orientador Daltro Nunes. Dissertação de Mestrado. PPGC–UFRGS, 1998.

[Reis 03] Reis, C. A. L., Uma Abordagem Flexível para Execução de Processos de Software Evolutivos, Orientador Daltro Nunes. Tese de Doutorado. PPGC–UFRGS, 2003.

[Rocha 01] Rocha, A. R. C. da, Maldonado, J. C. e Weber, K. C., Qualidade de Software: Teoria e Prática, Prentice Hall, São Paulo, 2001.

[Salviano 06] Salviano, C. F., “Uma Proposta Orientada a Perfis de Capacidade de Processo para Evolução da Melhoria de Processo de Software”, Exame de qualificação do programa de doutorado da Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, orientado pelo prof. Dr. Mario Jino, Campinas, SP, 2006.

[Softex 05] Softex - Sociedade para Promoção da Excelência do Software Brasileiro, MPS.BR - Melhoria de Processo do Software Brasileiro, Guia Geral, versão 1.0, 2005.

[Sommerville 00] Sommerville, Ian, Software Engineering, 6th edition, Addison-Wesley, 2000.

GLOSSÁRIO

Ciclo de vida. Conjunto das atividades que ocorrem durante o desenvolvimento de um software.

CMM (*Capability Maturity Model*). Um modelo de maturidade dos processos organizacionais. É uma iniciativa do SEI (*Software Engineering Institute*); é um modelo para avaliar e certificar a maturidade dos processos de uma organização que desenvolve software. Em general, se aplica principalmente em grandes empresas de software.

CMMI (*Capability Maturity Model Integration*). É uma evolução do CMM e procura estabelecer um único modelo para o processo de melhoria corporativo, integrando diferentes modelos e disciplinas.

ISO (*International Standard Organization*). Organização de Padrões Internacionais. Normaliza diversas características de processos e produtos no domínio do software ou de outros domínios. Por exemplo, normas sobre as características de qualidade do produto de software se encontram especificadas no padrão 9126; para o processo de avaliação de qualidade de produtos, no padrão 14598; para a avaliação dos processos de ciclo de vida do software, no padrão 12207; etc.

Requisito. Uma necessidade ou desejo do usuário, explícito ou implícito, que se traduz em atributos, características ou funções necessárias de um software.

SEI (*Software Engineering Institute*). Centro de pesquisa e desenvolvimento federal patrocinado pelo Departamento de Defesa dos Estados Unidos e operado pela Carnegie Mellon University.

APÊNDICE A

Uma visão geral da utilização da abordagem PRO2PI é apresentada a seguir por meio de quatro funções. O diagrama com a função de definição, ou atualização, de PRO2PI é apresentado na Figura A-1.

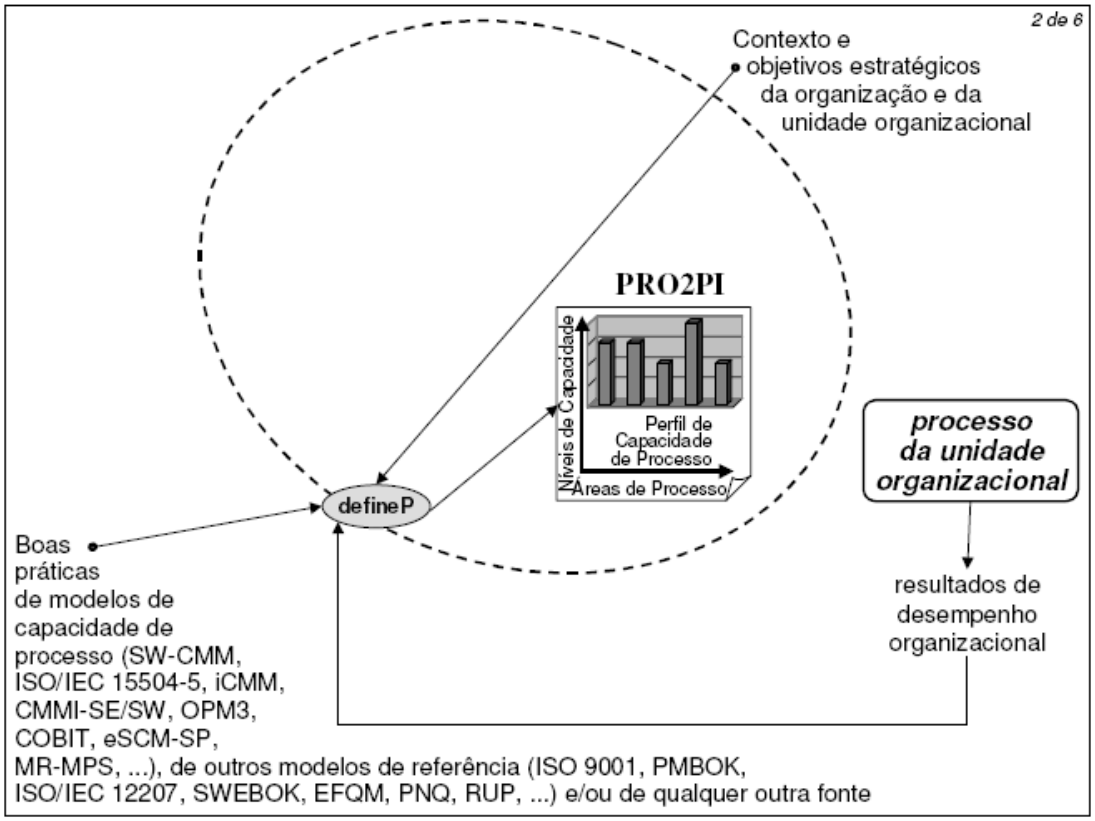

Figura A-1: Diagrama da definição de PRO2PI [Salviano 06]

De início, uma organização seleciona elementos de um ou mais modelos de referências para definir um perfil de capacidade de processo que represente os elementos selecionados desses modelos e de qualquer outra fonte. Buscando evoluir os processos para atender a esse perfil um ciclo de melhoria de processo é realizado. Esse perfil é ajustado com o contexto e objetivos estratégicos da organização e da unidade organizacional. Segundo [Salviano 06] o perfil pode conter boas práticas de referência selecionadas

dos modelos mais genéricos existentes e de outras fontes, de forma a representar orientações relevantes à organização.

A qualquer momento em função de novas percepções, alterações do contexto ou dos objetivos estratégicos e dos resultados da utilização da versão corrente o perfil pode ser alterado. Esse uso é representado pela função “defineP” (define, ou atualiza, perfil de capacidade de processo) da Figura A-1.

O diagrama com a função de utilização de PRO2PI é apresentado na Figura A-2 como um acréscimo em relação à Figura A-1.

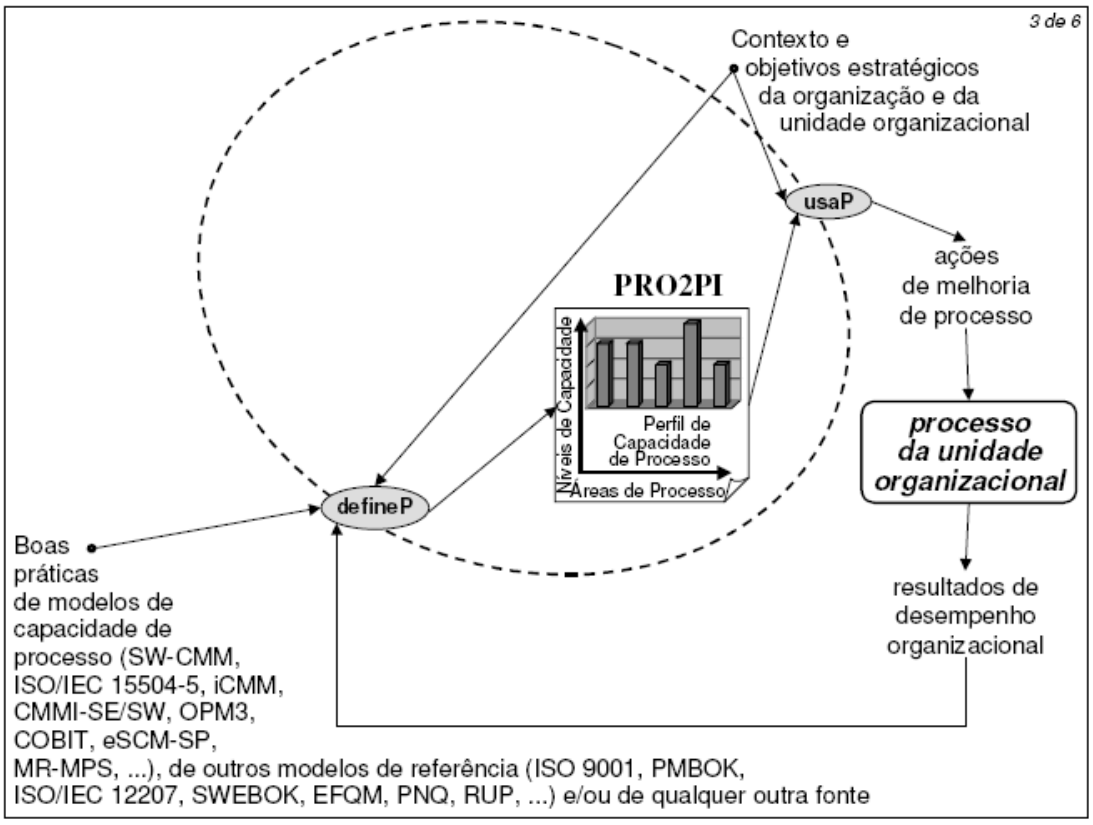

Figura A-2: Diagrama da utilização (e definição) de PRO2PI [Salviano 06]

[Salviano 06] define a função “usaP” (usa perfil de capacidade de processo) da Figura A-2 como responsável pelo uso desse perfil para orientar as ações de melhoria. Neste caso as ações de melhoria devem ser suficientes para que o processo resultante atenda a todos os requisitos

representados no perfil de capacidade de processo e apenas a estes requisitos.

O diagrama com a função de avaliação de capacidade de processo em relação a um PRO2PI é apresentado na Figura A-3 como um acréscimo em relação à Figura A-2.

O processo da unidade organizacional pode ser examinado com uma avaliação de capacidade de processo em relação ao perfil de capacidade de processo. Esse exame é representado pela função “avaliaPr” (avalia capacidade de processo em relação a um PRO2PI). Observe a Figura A-3. Os resultados de capacidade de processo gerados por essa avaliação podem ser utilizados como referências adicionais para a definição, ou alteração, do perfil de capacidade de processo [Salviano 06].

O diagrama com a função de definição de modelos de capacidade de processo mais específicos com a abordagem PRO2PI é apresentado na Figura A-5.

Um modelo mais específico pode ser definido a partir do contexto de negócio de um segmento qualquer, como, por exemplo, o segmento bancário, ou de um domínio, como, por exemplo, engenharia de requisitos. Esse modelo pode ser composto por áreas de processo ou perfis de capacidade de processo mais específicos para um segmento ou domínio.

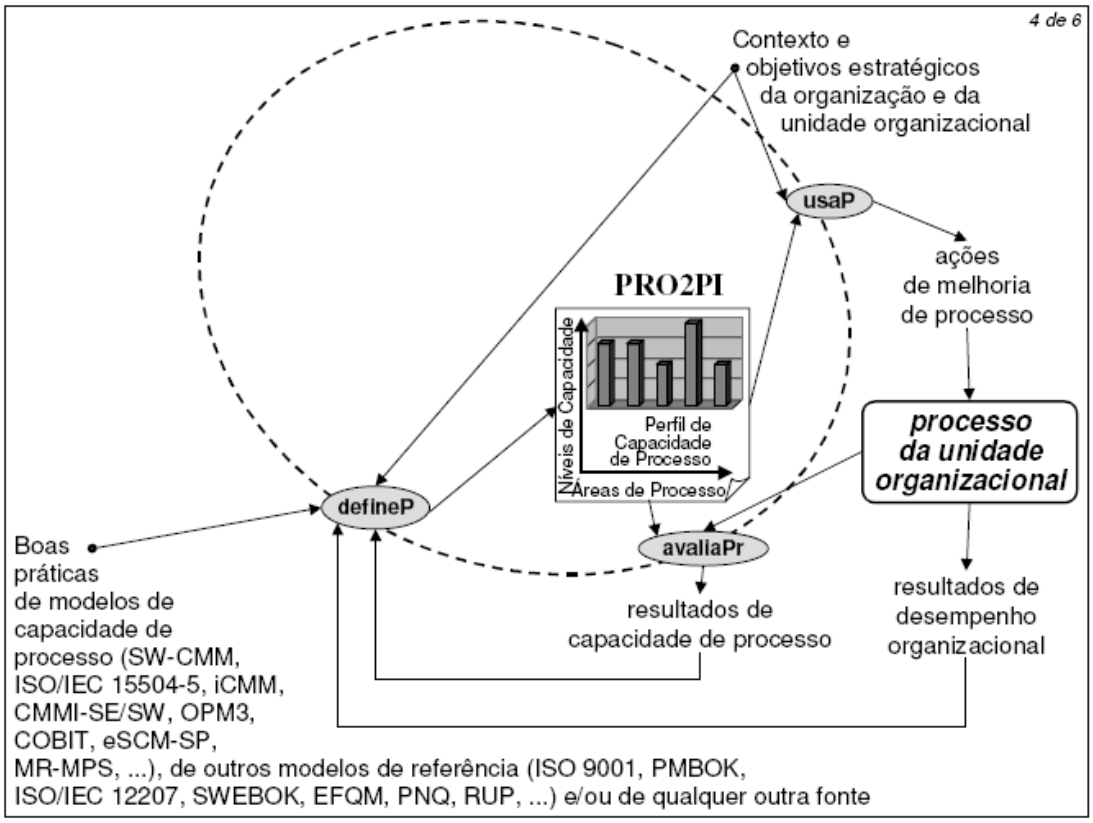

Figura A-3: Diagrama da avaliação de processo (e definição e utilização de PRO2PI) [Salviano 06]

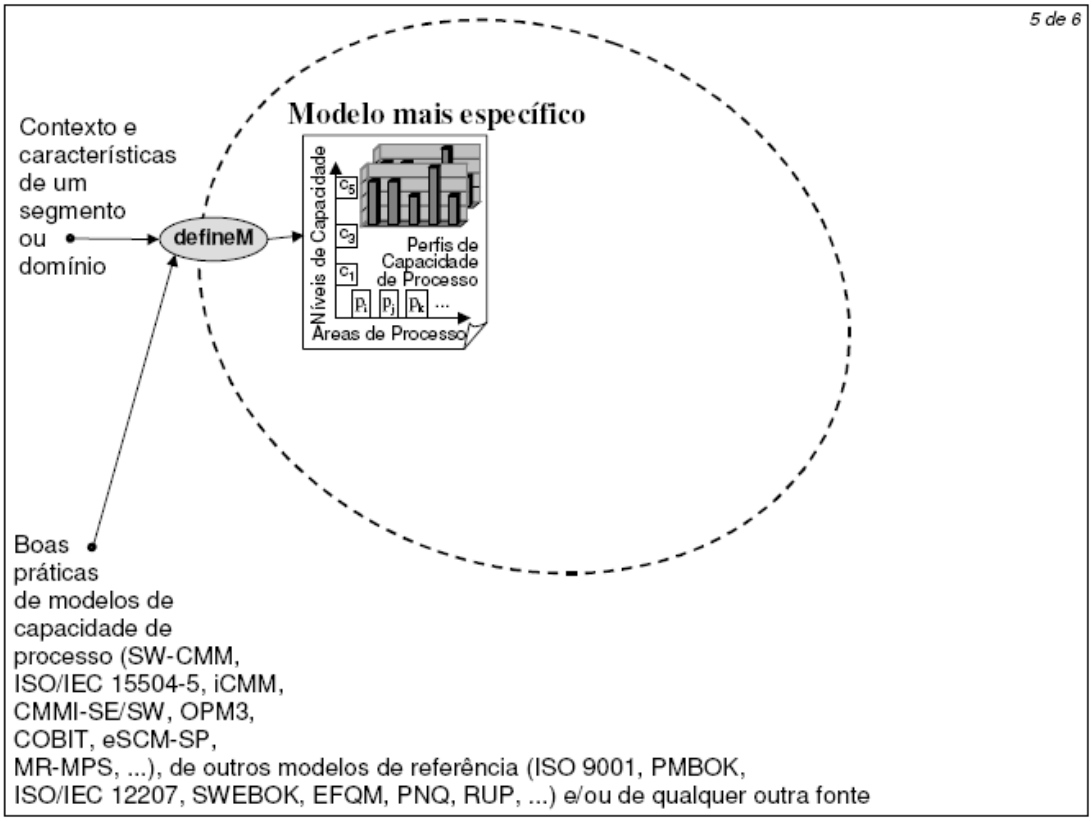

Figura A-4: Diagrama da definição de modelos mais específicos [Salviano 06]

Esse modelo mais específico, diz [Salviano 06], pode conter boas práticas selecionadas dos modelos de capacidade de processo, modelos de outros tipos ou de qualquer outra fonte de forma a representar orientações relevantes no segmento ou domínio. A função “defineM” (define, ou atualiza, modelo mais especifico de capacidade de processo) da Figura A-5 é que representa esse uso.

Um modelo mais específico pode ser que qualquer uma das três gerações de arquiteturas de modelo de capacidade de processo. Para [Salviano 06] caso seja um modelo estagiado fixo, ele será composto por uma hierarquia de perfis de capacidade de processo, como, por exemplo, a hierarquia do modelo PMMM (*Project Management Maturity Model*, Modelo de Maturidade da Gerência de Projeto) para o domínio de gerência de projeto e do modelo KMMM (*Knowledge Management Maturity Model,* Modelo de Maturidade de Gerência de Conhecimento) para o domínio de gerência de conhecimento. Caso seja um modelo contínuo fechado, ele será composto por um conjunto de áreas de processo, como, por exemplo, as áreas de processo do modelo Automotive SPICE para o segmento de fornecedores de software para automóveis e as do modelo OOSPICE para o domínio de desenvolvimento de software orientado a componentes.

Esses modelos podem ter sugestões de hierarquia de perfis de capacidade de processo. Caso seja um modelo contínuo aberto, ele será composto de especializações dos níveis de capacidade de processo.

A Figura A-5 de [Salviano 06] representa um diagrama esquemático completo da utilização da abordagem PRO2PI para desenvolvimento de modelos mais específicos e para realização de ações de melhoria dirigida por

perfil de capacidade de processo. Essa utilização é realizada por meio das quatro funções: defineM, defineP, usaP e avaliaPr. Não existe uma ordem pré-definida de utilização destas funções. Elas podem ser utilizadas várias vezes cada uma independente da execução das outras funções.

Na abordagem PRO2PI existe uma busca contínua em direção ao alinhamento do perfil de capacidade de processo como uma representação, segundo o aspecto de capacidade de processo, do processo da unidade organizacional.

Conforme representado na Figura A-5 uma melhoria de processo com a abordagem PRO2PI é realizada com a definição e utilização de um PRO2PI que pode conter boas práticas selecionadas dos modelos de capacidade de processo, modelos de outros tipos, modelos mais específicos de um segmento ou domínio ou de qualquer outra fonte.

Essa utilização da abordagem PRO2PI pode ser considerada como análoga á utilização da engenharia de requisitos no desenvolvimento de software, que podemos denominar de desenvolvimento de software dirigido por requisitos. A Figura A-6 apresenta um diagrama da utilização da engenharia de requisitos para desenvolvimento de software de forma análoga ao diagrama de utilização de PRO2PI descrito na Figura A-5.

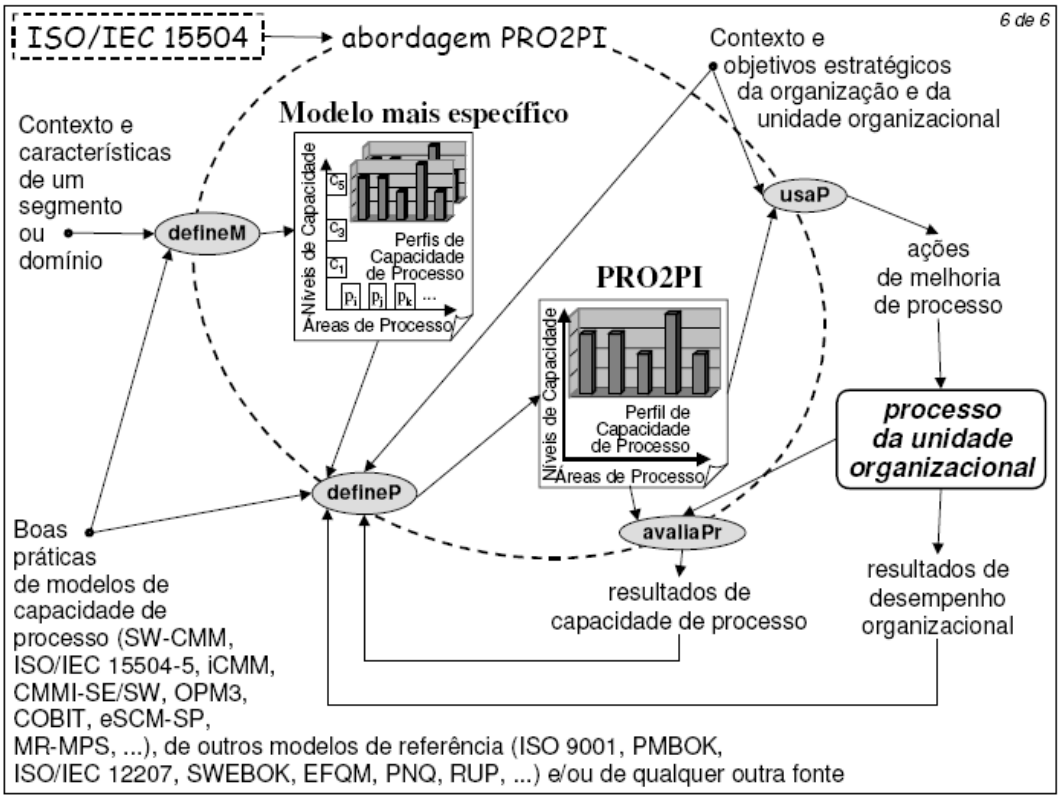

Figura A-5: Diagrama da abordagem PRO2PI para modelos e melhoria de processo [Salviano 06]

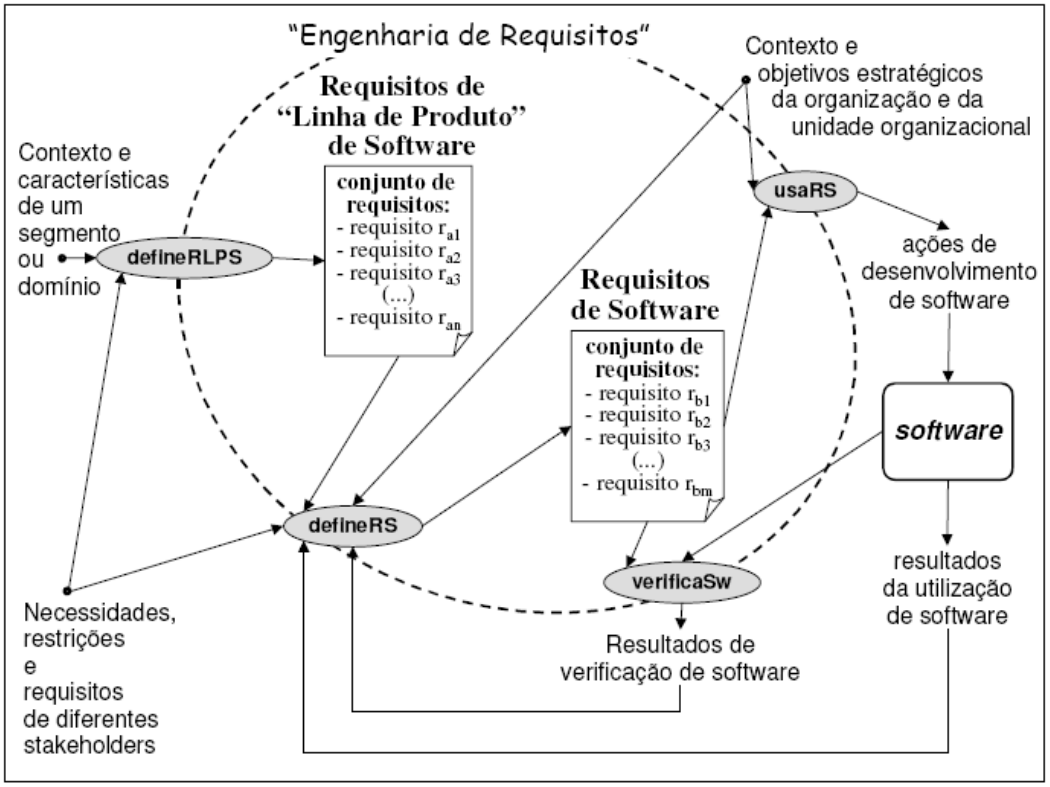

Figura A-6: Diagrama, análogo ao de PRO2PI, do desenvolvimento de software [Salviano 06]

Uma forma hipotética de desenvolvimento de software seria fazer esse desenvolvimento baseado em conjuntos de necessidades, restrições e requisitos de diferentes stakeholders, sem sistematizar esses conjuntos em um único conjunto de requisitos. Essa forma é hipotética porque não é utilizada, ou não deveria ser utilizada, em praticamente nenhum desenvolvimento de software.

O diagrama da Figura A-6 representa uma forma de desenvolvimento de software que é mais utilizada atualmente, para a qual os conjuntos de necessidades, restrições e requisitos dos diferentes stakeholders são analisados e representados como um sistema em um único conjunto de requisitos para o software [Salviano 06]. Desta forma o software tem tudo que os requisitos requisitam e somente o que os requisitos requisitam. Nesta forma de desenvolvimento, a função “defineR” (define requisitos) seria utilizada para definir, ou atualizar, o conjunto de requisitos, a função “usaRS” (usa requisitos de software) seria utilizada para orientar as ações de implementação do software de forma que isto seja dirigida pelos requisitos e a função “testaSw” (testa software) seria utilizada para buscar problemas no software e com isto sugerir o quanto o software desenvolvido atende aos requisitos [Salviano 06].

O desenvolvimento de requisitos para uma “linha de produto” de software, de tal forma que esses requisitos sejam um sistema é representado pela função “defineRLPS” (define requisitos de “linha de produto” de software), sendo que os requisitos foram baseados em diferentes conjuntos de requisitos, de diferentes stakeholders. Para [Salviano 06] esta forma de

desenvolvimento de requisitos é análoga à forma de desenvolvimento de modelos mais específicos representada na Figura A-5.

Segundo [Slaviano 06] o diagrama da Figura A-6 representa a integração da forma de desenvolvimento de requisitos para um framework de produtos de software e da forma de desenvolvimento de software dirigida a requisitos de software, de tal forma que esses requisitos sejam um sistema, sendo que os requisitos foram baseados em diferentes conjuntos de requisitos, de diferentes stakeholders, e em requisitos de um framework de software. Esta forma de desenvolvimento de requisitos é análoga à forma de desenvolvimento de modelos e realização de melhoria de processo representada na Figura A-5.

APÊNDICE B

A Tabela B-1 abaixo descreve as práticas base para o estabelecimento de Perfil de Capacidade de Processo que tem como propósito definir, utilizar, manter e melhorar um perfil de capacidade de processo como abstração do processo atual ou idealizado para uma unidade organizacional.

Práticas Base	Definição
1 Analisar objetivos (estratégicos) de negócio	Identificar e analisar os objetivos, estratégia, contexto e/ou qualquer outro aspecto relevante de negócio da unidade organizacional e da organização, para subsidiar e orientar a definição dos objetivos de melhoria.
2 Identificar objetivos da melhoria	Identificar os objetivos da melhoria, incluindo objetivos mais específicos para o próximo ciclo de melhoria e objetivos mais gerais para o programa de melhoria, sempre alinhado aos objetivos, estratégia, contexto e/ou qualquer outro aspecto relevante de negócio identificados.
3 Estabelecer critérios de qualidade para perfil	Estabelecer critérios de qualidade para avaliar e melhorar um perfil de capacidade de processo.
4 Identificar modelos de capacidade e áreas de processo	Identificar modelos de capacidade de processo que sejam relevantes para a organização, e selecionar áreas de processo desses modelos e, caso necessário, definir novas áreas de processo, que sejam relevantes para o processo atual e sua melhoria.
5 Definir ou selecionar um perfil	Definir, selecionar e/ou adaptar um perfil de capacidade de processo com áreas de processo selecionadas e/ou definidas, de forma que esse perfil esteja alinhado aos objetivos de melhoria.
6 Utilizar perfil para entendimento dos processos	Utilizar o perfil de capacidade de processo definido para orientar o entendimento dos processos correntes da unidade organizacional.
7 Utilizar perfil para a melhoria	Utilizar o perfil de capacidade de processo definido e os resultados do entendimento dos processos correntes, para orientar as ações de melhoria dos processos da unidade organizacional.
8 Analisar perfil definido	Analisar o perfil de capacidade de processo definido segundo os critérios de qualidade estabelecidos, quando necessário, como, por exemplo, em marcos definidos do ciclo de melhoria, em eventos ou periodicamente.
9 Ajustar o perfil definido	Ajustar o perfil de capacidade de processo definido segundo resultados de análise do perfil.
10 Monitorar o perfil definido	Monitorar o perfil de capacidade de processo definido em relação aos objetivos estratégicos para identificar possíveis desvios no alinhamento entre o perfil e os objetivos estratégicos.
11 Atualizar perfil definido	Atualizar o perfil de capacidade de processo definido sempre que o monitoramento identificar desvios significativos no alinhamento entre o perfil e os objetivos estratégicos, de forma que este alinhamento seja mantido.
12 Buscar correspondência entre perfil e processo	Buscar, por meio de ajustes no perfil e/ou ações de melhoria, a correspondência entre o perfil de capacidade de processo definido e o processo da unidade organizacional.

Tabela B-1: Práticas base para o estabelecimento de Perfil de Capacidade de Processo

APÊNDICE C

Fases do PRO2PI	Atividades do PRO2PI	Atividades do *ProImprove*
Inicia ciclo de melhoria	Levantamento de informações gerais sobre a empresa, incluindo o contexto e objetivos estratégicos, metodologias existentes, tecnologias e projetos existentes	Cadastrar Razão Definir Contexto Construir Apoio Alocar infra-estrutura
	Definição das Metas de Melhoria	Caracterizar estados
	Definição do Perfil de Capacidade de Processo	Base do *ImPProS* (Definição)
	Revisão e detalhamento do Programa de Melhoria de Processo	Não se Aplica ao *ProImprove*
	Definição do Plano Preliminar da Avaliação das Práticas Correntes	Base do *ImPProS* (Avaliação)
Avalia práticas correntes	Revisão do Plano da Avaliação das Práticas Correntes	Base do *ImPProS* (Avaliação)
	Treinamento da equipe de avaliação sobre o processo da avaliação	Não se Aplica ao *ProImprove*
	Adaptação das práticas de referência para a avaliação	Configurando Questões para Validação das Práticas
	Levantamento e validação de dados para a avaliação dos projetos e processos selecionados	Configurando análise das Questões
	Derivação e elaboração dos resultados da avaliação	Analisando e Validando a Execução da Sub-melhoria Analisando Questões de Validação da Prática
	Apresentação dos resultados da avaliação	Inferindo Questões de Execução da Sub-melhoria Sugerindo Resultado de Validação da Prática
	Revisão do perfil de capacidade de processo	Base do *ImPProS* (Definição)
Planeja ações de melhoria	Treinamento de pessoas selecionadas sobre o processo de planejamento da melhoria e elementos para sua implantação	Não se Aplica ao *ProImprove*
	Definição da estratégia para implantação da melhoria	Definir Procedimentos Desenvolver estratégias

	Definição das ações de melhoria	Planejar ações
	Elaboração do Plano de Melhoria de Processo	
	Definição das responsabilidades e necessidades de compromissos	
	Elaboração do Plano de Preparação da Organização	Criar solução
	Revisão do Plano de Melhoria de Processo por pessoas selecionadas	Planejar ações
	Estabelecimento das responsabilidades para execução do Plano de Melhoria de processo, incluindo as funções de gerente, grupo de processo e grupos de implementação de processo	Alocar Infra-estrutura
	Estabelecimento dos compromissos para execução do Plano	
	Atividades para consolidação de planejamento	Planejar ações
	Revisão das atividades e resultados da sub-fase anterior	
	Atualização do Plano	
	Treinamento de pessoas selecionadas sobre elementos para implantação do Plano	Não se Aplica ao *ProImprove*
	Apresentação para início da implantação das ações de melhoria	Não se Aplica ao *ProImprove*
Realiza ações de melhoria	Realizar atividades descritas no Plano	Testar solução
	Monitorar as atividades conforme o Plano	Refinar solução
	Revisar o plano e realizar ações corretivas	
Prepara institucionalização da melhoria	Revisão do resultado da melhoria	
	Elaboração do Plano de institucionalização	Implantar solução
Institucionaliza a melhoria	Realizar atividades descritas no Plano de Institucionalização	
	Monitorar as atividades conforme o Plano de Institucionalização	Analisar e validar
	Revisar o plano de Institucionalização e realizar ações corretivas	Propor Ações Futuras

Tabela C-1: Tabela comparativa entre as atividades do PRO2PI e do *ProImprove*