

Universidade Federal de Santa Catarina

Maíra Bay de Souza

**Modelo de Processo de Software: aplicação em uma
empresa júnior**

Florianópolis, 2004

UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE CIÊNCIAS DA COMPUTAÇÃO

TÍTULO: "Modelo de Processo de Software: aplicação em uma empresa júnior"
AUTORA: Maíra Bay de Souza
ORIENTADORA: Christiane Gresse von Wangenheim
CO-ORIENTADOR: Aldo von Wangenheim
BANCA EXAMINADORA: José Eduardo De Lucca

Palavras-chave: modelo de processo, processo de software, empresa júnior

Florianópolis, 13 de Fevereiro de 2004

DEDICATÓRIA

Aos meus pais,
que me proporcionaram uma educação
incomparável e sempre acreditaram no
meu potencial. Sem vocês eu não
poderia estar aqui.

AGRADECIMENTOS

- * À minha família, pela compreensão, amor e carinho.
- * À professora Christiane Gresse von Wangenheim, pelo apoio e paciência, por todos os valiosos comentários e orientações.
- * Aos professores Aldo von Wangenheim e José Eduardo De Lucca, por fazerem parte da banca avaliadora.
- * A todos os membros e ex-membros do NPI pela cooperação na realização deste trabalho.
- * A todos aqueles que, assim como eu, acreditam no NPI e no seu sucesso.

RESUMO

Este trabalho consiste na elaboração de um modelo de processo de desenvolvimento de software para a empresa júnior dos cursos de Ciências da Computação e Sistemas de Informação da Universidade Federal de Santa Catarina. Além disso, o trabalho também engloba a atividade de análise do processo de desenvolvimento existente, bem como a atividade de produção de um plano de avaliação (utilizando a metodologia GQM), a ser executado durante a aplicação do modelo. Os próprios alunos dos cursos citados realizam as atividades técnicas e administrativas da empresa júnior. Por causa disso, o NPI (Núcleo de Projetos em Informática) possui algumas características de micro e pequenas empresas (como inexperiência em gerência e em desenvolvimento sistematizado de software) e outras bem particulares (por exemplo, altíssima rotatividade de diretores e desenvolvedores), as quais levam à ocorrência de processos e produtos de baixa qualidade. Baseando-se na literatura, acredita-se que a utilização de um modelo de processo de desenvolvimento irá contribuir para a melhoria da qualidade do software desenvolvido no NPI, colaborando para a solução dos problemas detectados. O modelo apresentado neste trabalho foi desenvolvido com base no processo existente no NPI e nos requisitos da empresa júnior para um processo ideal. Ele abrange todas as fases de um projeto típico de desenvolvimento de software no NPI, contendo tanto atividades técnicas como gerenciais. Comparando-o com alguns outros modelos desenvolvidos para outras empresas e com modelos de avaliação do processo (p.ex. CMM, SPICE), concluímos que o modelo proposto é adequado ao NPI. O modelo será aplicado na empresa júnior e então melhorado, como parte de um processo que será realizado continuamente pelos próprios membros da empresa.

Palavras-chave: modelo de processo, processo de software, empresa júnior.

ABSTRACT

This work consists of a design of a software development process model to be used by the Junior Enterprise administrated by the Computer Science and Information Systems students at the Federal University of Santa Catarina, Brazil. It also includes an analysis of the current process and a development of an evaluation plan (based on the GQM method), which will be executed while the model is in use. The students of the referred courses are in charge of both the technical and the administrative work at the Junior Enterprise. Because of that, NPI (Núcleo de Projetos em Informática, Information Technology Projects Center) has some aspects of small companies (such as lack of experience in management and in systematic software development) and others that are very particular of it's own (for instance, high turnover of directors and developers). These aspects have a great influence on the existence of low quality processes and products. Based on literature, we believe the utilization of a software development process model will help improve the quality of NPI's software and therefore, will contribute to the solution of some of the problems described. The process model presented herein is based on the existing process at NPI and the junior enterprise's requirements for an ideal development process. It comprises all the phases of a typical NPI software project, containing both technical and managerial kinds of activities. When comparing the resulting model with other process models developed for other companies or with assessment models (such as CMM, SPICE, etc), we can conclude that our model is adequate for NPI. The process model will be used by the Junior Enterprise and then improved, as part of a continuous process executed by NPI's members.

Key-words: process model, software process, junior enterprise

LISTA DE FIGURAS

Figura 1:	Organograma do NPI	23
Figura 2:	Extrato de um exemplo de esquema do banco de dados	33
Figura 3:	Exemplo de diagrama de casos de uso	33
Figura 4:	Extrato de um exemplo de modelo conceitual	34
Figura 5:	Exemplo de diagrama de seqüência, para um caso de uso	34
Figura 6:	Exemplo de diagrama de colaboração para uma operação de sistema	35
Figura 7:	Extrato de um exemplo de diagrama de classes	35
Figura 8:	Exemplo de diagrama de Atividades	36
Figura 9:	Extrato de um exemplo de documento de acompanhamento do desenvolvimento	43
Figura 10:	Modelo de ciclo de vida atual do NPI	45
Figura 11:	Modelo de ciclo de vida code-and-fix	56
Figura 12:	Modelo de ciclo de vida cascata	57
Figura 13:	Modelo de ciclo de vida em V	58
Figura 14:	Modelo de ciclo de vida desenvolvimento incremental	59
Figura 15:	Modelo de ciclo de vida em espiral	60
Figura 16:	Áreas-chave do CMM agrupadas por nível	66
Figura 17:	Estrutura do CMM	67
Figura 18:	Exemplo de um perfil definido pelo SPICE	70

Figura 19:	Áreas de processo abordadas pelo PSP e o TSP	75
Figura 20:	Relação do PSP e do TSP com o CMM	75
Figura 21:	Modelo de Ciclo de Vida proposto	94

LISTA DE QUADROS

Quadro 1	Partes de um documento de requisitos	29
Quadro 2	Fase de análise de requisitos do processo atual do NPI	31
Quadro 3	Fase de modelagem do sistema do processo atual do NPI	38
Quadro 4	Fase de implementação do processo atual do NPI	40
Quadro 5	Resumo do processo atual do NPI	46
Quadro 6	Comparação dos requisitos do NPI com outros modelos	85
Quadro 7:	Resumo das fases do modelo	96
Quadro 8:	Explicação das características de uma fase	97
Quadro 9:	Fase de Requisitos	98
Quadro 10:	Detalhamento da atividade 1 da fase de Requisitos	100
Quadro 11:	Exemplos de informações sobre o paralelismo das atividades	101
Quadro 12:	Documentos-padrão e seus códigos	102
Quadro 13:	Modelo do cabeçalho-padrão	103
Quadro 14:	Exemplo de cabeçalho preenchido	104
Quadro 15:	Um exemplo de documento padrão	105
Quadro 16:	Comparação dos requisitos do NPI com os modelos do capítulo 4 e o modelo desenvolvido	107
Quadro 17:	<i>Abstraction Sheet</i> para a meta Qualidade	114
Quadro 18:	Exemplos de perguntas do Plano GQM	115

Quadro 19:	Plano de Mensuração	117
Quadro 20:	Um dos roteiros para entrevista	118

SUMÁRIO

1 Introdução	13
1.1 <i>Objetivos</i>	15
1.2 <i>Justificativa</i>	15
1.3 <i>Metodologia</i>	18
1.4 <i>Organização do Trabalho</i>	19
2 Contextualização	21
2.1 <i>NPI - Núcleo de Projetos em Informática</i>	21
2.2 <i>Os Problemas do NPI</i>	24
2.3 <i>O Processo Existente</i>	27
2.3.1 <i>A análise de requisitos</i>	28
2.3.2 <i>A modelagem do sistema</i>	32
2.3.3 <i>A implementação</i>	38
2.3.4 <i>Aspectos de gerência e qualidade do processo</i>	40
2.3.5 <i>Discussão</i>	43
2.4 <i>Análise das Necessidades do NPI</i>	46
3 Conceitos Fundamentais	49
3.1 <i>O que é um Processo de Software?</i>	49
3.2 <i>Modelo de Processo de Software</i>	51
3.3 <i>Qualidade do Processo</i>	52
3.4 <i>Modelos de Ciclo de Vida</i>	55
3.4.1 <i>Exemplos de modelos</i>	56
3.4.2 <i>Discussão</i>	60
3.5 <i>Conceitos Complementares</i>	61
4 Estado da Arte e Prática	64
4.1 <i>Visão Geral de Modelos de Avaliação de Processo</i>	64
4.1.1 <i>CMM - Capability Maturity Model</i>	65
4.1.2 <i>SPICE - Software Process Improvement and Capability determination</i>	68
4.1.3 <i>ISO 9000-3</i>	71
4.1.4 <i>PSP e TSP</i>	74
4.2 <i>Questões Humanas na Modelagem de Processos</i>	76
4.3 <i>Relatos de Experiências</i>	80
4.4 <i>Discussão</i>	84

5 O Modelo de Processo desenvolvido para o NPI	89
5.1 <i>A elaboração do Modelo de Processo do NPI</i>	92
5.2 <i>O Conteúdo do MPDS-NPI</i>	96
5.3 <i>Discussão</i>	106
6 Avaliação do Modelo	111
6.1 <i>O Plano da Avaliação</i>	111
6.2 <i>Execução da Avaliação</i>	119
7 Conclusão	120
7.1 <i>Benefícios para o NPI</i>	121
7.2 <i>Trabalhos Futuros</i>	122
7.3 <i>Considerações Finais</i>	123
8 Bibliografia	124
Apêndice A - Artigo	128
Apêndice B - Plano da Avaliação	135

1 Introdução

Uma empresa júnior tem como um de seus objetivos principais, colocar o aluno em contato com o mercado de trabalho e com o dia-a-dia da profissão. Ela existe como uma empresa real, com produtos e clientes reais; porém, legalmente, ela deve atuar como uma associação sem fins lucrativos (semelhante a uma organização não-governamental).

O Núcleo de Projetos em Informática (NPI), a empresa júnior dos cursos de Ciências da Computação e Sistemas de Informação da Universidade Federal de Santa Catarina (UFSC), é composto apenas por alunos dos cursos citados, que atuam como diretores ou estagiários. Os diretores têm, entre outras, a responsabilidade pelo bom funcionamento da empresa júnior e pela boa qualidade do software por ela produzido.

Por ser formado por alunos de graduação, o NPI possui cargos com mandatos de curta duração, e seus diretores e estagiários têm pouca experiência (teórica e prática) de desenvolvimento sistemático de software comercial. Além disso, ao adquirirem experiência de desenvolvimento, os alunos estão chegando ao fim de seus mandatos na diretoria ou se formando - e levam com eles a experiência adquirida, o que prejudica a qualidade e a produtividade da empresa júnior.

Uma solução para esses problemas pode ser o desenvolvimento de um modelo de processo de software para o NPI.

O desenvolvimento de um software pode ser feito utilizando-se várias metodologias, inclusive a de um processo *ad-hoc*. Como será explicado mais adiante, a qualidade de um software está fortemente associada com o nível de organização do processo utilizado para desenvolvê-lo.

O processo de software constitui-se de várias tarefas, que podem ou não ser executadas simultaneamente: desenvolvimento (análise, projeto, implementação e testes), manutenção, gerência de configuração, entre outras.

Um modelo de processo de software pode ser concretizado, entre outras formas, através de um manual. Nele, consta o que deve ser feito em cada fase do processo citada anteriormente.

O trabalho aqui apresentado limita-se à criação de um modelo de processo de **desenvolvimento** de software, ou seja, o modelo servirá apenas para a tarefa de desenvolvimento - e não, por exemplo, para a de manutenção.

O modelo desenvolvido neste trabalho será aplicado ao NPI e avaliado em relação à sua contribuição para a administração dessa empresa júnior.

1.1 Objetivos

O objetivo principal deste trabalho é desenvolver e aplicar um modelo de processo de desenvolvimento de software ao NPI.

Entre os objetivos mais específicos do trabalho, destacam-se:

- * Estudar modelos de processo de software: suas características e como aplicá-los.
- * Desenvolver um modelo de processo de desenvolvimento de software para o NPI.
- * Aplicar o modelo no NPI em, pelo menos, um projeto.
- * Avaliar o modelo de processo de desenvolvimento de software no NPI comparando a situação antes e depois da aplicação do modelo.

1.2 Justificativa

Como explicado anteriormente, o NPI é formado apenas por alunos do Departamento de Informática e Estatística (INE) da UFSC, que podem participar como diretores ou como estagiários. Aos diretores cabe administrar a empresa, entrar em contato com os clientes, e outras tarefas. Os estagiários, por sua vez, desenvolvem os sistemas que a empresa comercializa. Para cada projeto, os

diretores devem selecionar alguns associados para atuarem como desenvolvedores naquele projeto. Além disso, de acordo com o estatuto do NPI (NÚCLEO DE PROJETOS EM INFORMÁTICA 1996), deve haver uma eleição para diretoria a cada um ano. Dessa maneira, pode-se perceber que há uma alta rotatividade de desenvolvedores e diretores nessa empresa júnior. Como nos dois cursos (Ciências da Computação e Sistemas de Informação), a disciplina de Engenharia de Software (ES) é ministrada somente na 4ª ou 6ª fase, a maioria dos desenvolvedores - e eventualmente os diretores - não sabe como desenvolver sistematicamente um sistema comercial e não tem experiência prática na área. Por causa dessa inexperiência, os alunos também têm dificuldades de entender a necessidade do uso de técnicas de ES durante o desenvolvimento de um sistema. Além disso, ao adquirirem experiência no processo de desenvolvimento de software, geralmente os alunos estão no final de seu mandato como diretores e estão se formando, ou seja, a experiência não "permanece" no NPI.

Segundo KEHOE e JARVIS (1996) e PAULK et al. (1993), os produtos de uma empresa que não possui um bom gerenciamento do seu processo de software certamente serão de pior qualidade do que os de uma empresa que possui um processo bem gerenciado e definido - onde qualidade é vista como a capacidade de produzir sistemas que vão ao encontro dos requisitos do cliente dentro dos prazos e custos estabelecidos no início do projeto. É esse conceito de qualidade que os certificados ISO e CMM asseguram quando são emitidos para uma empresa de software. Portanto, pode-se perceber que os fatores citados anteriormente (alta rotatividade, tanto dos desenvolvedores quanto da diretoria; inexperiência dos mesmos; dificuldade dos desenvolvedores e diretores de entender a necessidade do uso de técnicas de ES no desenvolvimento de software; experiência de

desenvolvimento permanecer com os alunos e não com a empresa júnior) prejudicam a qualidade e a produtividade do NPI.

Ainda, um dos benefícios de se ter um modelo a seguir durante o processo de produção de software é que esse processo fica independente das pessoas que estão produzindo o software, ou seja, segue-se um manual, com passos definidos, e não as idéias de uma ou outra pessoa. Assim, quando mudar o quadro de funcionários de uma empresa, ela pode continuar produzindo software de qualidade, pois o manual permanece com a empresa. Essa afirmação leva a crer que um manual de processo de software se encaixa perfeitamente na solução dos problemas do NPI.

Os *modelos de avaliação* do processo de software especificam as características que um *modelo de processo* de software deve possuir para ser considerado um modelo de qualidade. Os modelos de avaliação do processo de software são muito mais conhecidos e divulgados do que os modelos de processo de software em si, os quais geralmente não são publicados por possuírem dados confidenciais das empresas. Existem vários modelos de avaliação do processo de software propostos atualmente, mas a maioria deles (como CMM e ISO) é voltada para grandes empresas. Há na literatura, uma carência de modelos de avaliação do processo de software para micro e pequenas empresas. Essa carência é ainda maior para modelos de processo de software.

Além disso, um modelo de processo de software depende das características da empresa na qual ele será utilizado, como por exemplo, número de funcionários, o tipo de software que é desenvolvido, entre outras. Por isso, mesmo que existam

modelos de processo prontos, elaborados para outras empresas (como o proposto por WEBER, 2002) ou para servir como exemplo, um modelo de processo de software deve ser adaptado à empresa na qual ele será usado.

Assim, acredita-se que um modelo de processo de desenvolvimento de software, elaborado especificamente para o NPI pode melhorar a sua qualidade e produtividade.

1.3 Metodologia

Para a realização deste trabalho, utiliza-se a seguinte metodologia.

É feita uma pesquisa bibliográfica sobre modelos de processo de software e seu desenvolvimento e também são estudados alguns exemplos de modelos de processo já existentes. Em paralelo, é realizado um estudo do processo de desenvolvimento de software do NPI, analisando como estão sendo desenvolvidos os produtos atualmente e analisando os requisitos, necessidades e limitações existentes no NPI em relação a um modelo de processo. É então desenvolvido um modelo para o NPI, com base no processo existente na empresa e nos padrões de qualidade e *best practices*, e adaptado especificamente ao NPI. Ainda, são previstas as seguintes atividades: utilização do modelo em no mínimo um projeto de software no NPI; avaliação (durante o uso do modelo) em relação à sua aplicabilidade e aos seus custos e benefícios; e melhora do modelo com base nos resultados obtidos nessa avaliação.

1.4 Organização do Trabalho

O presente trabalho está organizado em sete capítulos.

O capítulo seguinte apresenta um estudo do processo de desenvolvimento de software do NPI, analisando como estão sendo desenvolvidos os produtos atualmente e analisando os requisitos, necessidades e limitações existentes no NPI em relação a um modelo de processo.

O capítulo 3 apresenta conceitos básicos utilizados no decorrer do trabalho. Os conceitos são principalmente das áreas de Engenharia de Software e Modelagem de Processo. Alguns conceitos são utilizados uniformemente na literatura com o mesmo nome, mas para outros, ainda não há um consenso (diferentes autores atribuem nomes diferentes a conceitos semelhantes). O objetivo do capítulo três é esclarecer com que significados serão utilizados quais termos no presente trabalho.

No capítulo 4 são apresentados alguns modelos de avaliação de processo e alguns relatos de experiências de desenvolvimento de modelos de processo de software.

O modelo de processo desenvolvido para o NPI é apresentado brevemente no quinto capítulo. São explicadas suas características principais, uma síntese do seu conteúdo e as etapas da sua elaboração. Também são apresentados vários exemplos que ilustram o seu conteúdo: uma das fases do modelo, um dos documentos-padrão, entre outros. Ao final do capítulo, é feita uma comparação do

modelo apresentado neste trabalho com os modelos discutidos no capítulo 4, em relação aos requisitos do NPI (descritos no capítulo 2).

O sexto capítulo contém uma breve descrição do plano de avaliação e de como ele deve ser executado.

Finalmente, o trabalho é concluído mostrando-se os benefícios do modelo para o NPI e o que ainda é necessário para que o NPI possua um modelo de processo de software completamente adaptado à sua realidade e às suas necessidades.

2 Contextualização

Neste capítulo serão apresentadas mais informações sobre o foco do trabalho, o Núcleo de Projetos em Informática - a empresa júnior dos cursos de Ciências da Computação e Sistemas de Informação da UFSC. Adicionalmente, será analisado o processo de desenvolvimento de software existente nessa empresa júnior e suas falhas. Finalmente, concluir-se-á sobre as necessidades atuais do NPI em relação a um processo de desenvolvimento de software.

2.1 NPI - Núcleo de Projetos em Informática

O NPI foi fundado em 5 de Dezembro de 1996 por alunos do Curso de Ciências da Computação da Universidade Federal de Santa Catarina. Seus objetivos principais são "a motivação de um espírito empreendedor no aluno de computação e a introdução do estudante no mercado de trabalho através de prestações de serviços a instituições, micro e pequenas empresas."(NÚCLEO DE PROJETOS EM INFORMÁTICA 2002).

Legalmente, uma empresa júnior funciona como uma associação sem fins lucrativos, devendo vender seus produtos e serviços a preço de custo. O estatuto e

o regimento interno da empresa governam seu funcionamento. Seus objetivos são dar oportunidade aos alunos de por em prática os conhecimentos teóricos adquiridos nas disciplinas do curso, incentivar o empreendedorismo nos estudantes, colocá-los em contato com a realidade da profissão e valorizar (tanto no mercado quanto no ambiente acadêmico) os alunos, a empresa júnior e a universidade.

Conforme o seu estatuto (NÚCLEO DE PROJETOS EM INFORMÁTICA 1996), o NPI é formado por uma Diretoria Executiva e um Conselho de Administração. O organograma da Diretoria Executiva é composto dos seguintes cargos (ilustrados na Figura 1): Diretor Presidente, Diretor Administrativo-Financeiro e um Assessor Administrativo-Financeiro, Diretor de Projetos e dois Assessores de Projetos, Diretor de Recursos Humanos e um Assessor de Recursos Humanos e Diretor de Marketing e um Assessor de Marketing. A diretoria deve ser composta por alunos associados e o conselho pode ser formado por alunos associados, professores do departamento e outras pessoas que possam contribuir para o NPI. É feita uma eleição para esses cargos, a cada um ano. Os associados que não fazem parte da diretoria nem do conselho, podem participar como desenvolvedores nos projetos do NPI. Todos os associados têm direito de votar na eleição.

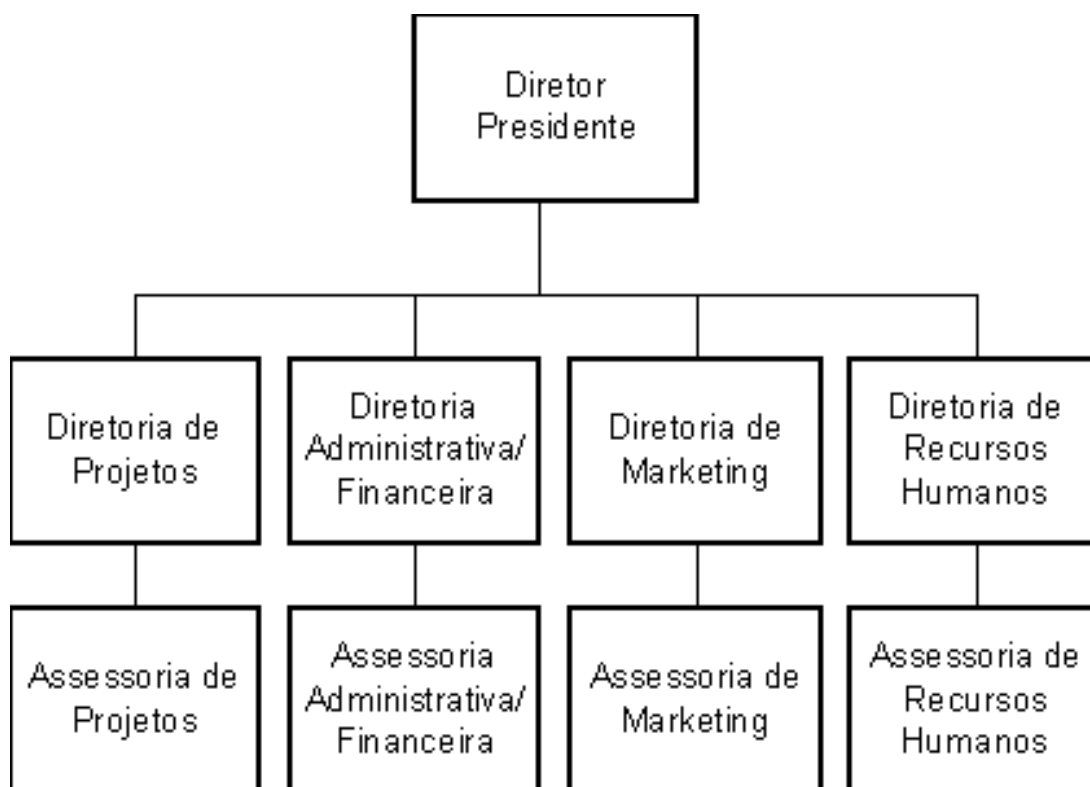


Figura 1: Organograma do NPI
(NÚCLEO DE PROJETOS EM INFORMÁTICA 2002)

A diretoria deve administrar a empresa júnior, executando tarefas que dêem suporte à realização de projetos, como marketing, gestão financeira, de recursos humanos e de projetos. O conselho deve examinar e aprovar as ações significativas da diretoria, como orçamentos, relatórios de atividades e demonstrações financeiras.

O NPI é uma empresa que vende software e serviços de informática. Seus clientes são instituições dentro da própria UFSC, empresas do mesmo e de outros ramos e até mesmo pessoas físicas.

Os projetos realizados pelo NPI foram principalmente nas áreas de Desenvolvimento de Software, Treinamento e Banco de Dados. Além desses, o NPI já realizou projetos nas áreas de Redes de Computadores e Desenvolvimento para

Web. Atualmente, o NPI realiza projetos nas áreas de Banco de Dados e Desenvolvimento e Manutenção de Software.

O funcionamento do NPI, em relação à realização de um projeto, ocorre da seguinte maneira: um cliente entra em contato com o NPI, solicitando um desenvolvimento ou outro serviço; o NPI avalia o pedido do cliente; se ele for de interesse da empresa júnior, são então selecionados associados para atuar no projeto; a partir daí, é negociado um contrato com o cliente; quando chega-se a um acordo, o contrato é fechado e dá-se início à prestação do serviço (desenvolvimento, por exemplo).

2.2 Os Problemas do NPI

Atualmente, o NPI desenvolve sistemas com sucesso. Entretanto, existem algumas deficiências graves no seu processo de desenvolvimento. Entre elas, pode-se citar:

* **Alta rotatividade de pessoal.** Como explicado anteriormente, o NPI é formado por alunos dos cursos de graduação em Ciências da Computação e Sistemas de Informação, cuja duração média é de 4 anos; assim, os alunos raramente permanecem mais do que dois anos envolvidos com o NPI.

- * **Pouco conhecimento e experiência** (principalmente prática) de desenvolvimento de software de acordo com os princípios da ES. Isso ocorre porque os alunos só entram em contato com essa área na metade do curso (a disciplina de Engenharia de Software é ministrada na 4ª fase para o curso de Sistemas de Informação e na 6ª fase para o curso de Ciências da Computação) e esse contato é essencialmente teórico.

- * **Falta de comprometimento e de conscientização** da necessidade de utilização de técnicas de ES em desenvolvimento de sistemas. Pelo mesmo motivo explicado acima, há uma certa dificuldade dos alunos em entender a necessidade de utilização de técnicas de ES quando eles passaram metade do curso desenvolvendo software com sucesso sem o uso dessas técnicas; além disso, mesmo quando os alunos se conscientizam dessa necessidade, existe a dificuldade de manter o compromisso de produzir documentação e realizar outras atividades de ES quando o desenvolvimento atrasa ou o cliente aguarda ansiosamente um protótipo.

- * **Perda das experiências adquiridas** em desenvolvimentos anteriores. Por causa da falta de um processo definido e pela baixa produção de documentação, as dificuldades, os resultados, os obstáculos superados e outras informações ficam apenas na memória dos alunos que participaram do projeto; quando esses alunos deixam o NPI, essa informação torna-se inalcançável.

- * **Falta de um processo definido** para o desenvolvimento de software. Cada desenvolvimento é realizado de uma maneira diferente e sem etapas claramente definidas.

- * **Definição precária de papéis.** Os papéis determinam atividades a serem realizadas durante o desenvolvimento (por exemplo, o testador deve testar o software); mas como as etapas e atividades são mal definidas, os papéis também ficam mal definidos.

- * **Acúmulo de funções** para o Diretor de Projetos. Pela falta de gerenciamento dos recursos humanos, há uma carência de gerentes de projeto no NPI, assim, o diretor de projetos acaba assumindo esse papel. Também, pela inexperiência dos desenvolvedores na área, o diretor de projetos acaba assumindo o papel de analista de requisitos.

- * **Má administração de recursos humanos.** Devido a um grande desconhecimento e desinteresse dos alunos de Ciências da Computação e Sistemas de Informação pelo NPI e a falta de experiência da diretoria em administração de recursos humanos, o corpo gerencial do NPI é deficiente, isto é, faltam pessoas para trabalhar como gerentes de projeto, entre outros papéis.

- * **Mal gerenciamento do desenvolvimento,** consequência do acúmulo de tarefas para o Diretor de Projetos e da sua inexperiência em gerência - um dos principais problemas de toda a diretoria do NPI.

- * **Escassez de recursos financeiros.** Leva à utilização de ferramentas gratuitas (que é uma solução bastante comum também em micro e pequenas empresas).

- * **Ambiente de desenvolvimento heterogêneo.** Em consequência do problema financeiro mencionado acima, o NPI não possui recursos computacionais disponíveis para os desenvolvedores. Por causa disso, eles são obrigados a utilizar seus próprios computadores ou os computadores dos laboratórios de informática da universidade. Assim, não se pode prever em qual ambiente será realizado o processo de desenvolvimento do software. Neste caso - quando existe um ambiente computacional heterogêneo (i.e. com diferentes hardwares, sistemas operacionais, e etc.) - é recomendado o uso de ferramentas multiplataforma.

2.3 O Processo Existente

O NPI não possui um processo claramente definido para o seu desenvolvimento de software. Assim, para entender o processo existente, foi efetuada a análise de dois projetos em andamento (sendo que um estava em conclusão e o outro estava em seu início) - foram realizadas entrevistas informais com o diretor e os desenvolvedores dos dois projetos e foram examinados os artefatos produzidos durante o desenvolvimento.

Após a investigação, pode-se inferir que o processo do NPI constitui basicamente de três fases: análise de requisitos, modelagem do sistema e implementação.

O início de um projeto no NPI ocorre quando o cliente faz um contato (normalmente através de telefone ou e-mail), solicitando um desenvolvimento.

2.3.1 A análise de requisitos

Depois do contato mencionado acima, é realizada uma reunião com o cliente e o NPI (representado pelo Diretor de Projetos e mais um ou dois diretores). Em seguida, o diretor de projetos produz um documento de requisitos - cuja aprovação faz parte da negociação do contrato com o cliente, realizada em reuniões subsequentes. Para a produção do documento de requisitos, o diretor de projetos utiliza a seguinte metodologia: descrever o sistema existente em um documento e as necessidades do cliente do em outro, juntar esses dois documentos e identificar os módulos do sistema. A partir desses três documentos, gerar o documento de requisitos. Ao final do processo, porém, esses três documentos são descartados e só é mantido o documento de requisitos. No Quadro 1, são mostradas partes do documento de requisitos do primeiro projeto.

1. Requisitos Gerais - Módulos do Sistema

ID	Descrição
...	
R.1.5	O sistema deve ter um módulo de Cadastro de Tipo de Tarefa . Esse tipo de tarefa é uma descrição resumida de cada espécie de tarefa.
R.1.6	O sistema deve ter um módulo de Cadastro de Tarefas . Nesse cadastro, serão armazenados todas as atividades realizadas por desenvolvedores do projeto.
...	

2. Requisitos Gerais - Funções Gerais por Módulo

ID	Descrição
R.2.1	Cada módulo deve gerar relatórios.
R.2.2	Cada módulo deve automaticamente importar dados do MS-Project, e deve também atualizar o MS-Project toda vez que algum dado seja criado ou alterado.
...	

...

4. Requisitos - Cadastros

ID	Descrição
R.4.1	Cadastro de Clientes : Cadastra o Nome do Cliente, Código, Endereço, Cidade, Estado, CEP, País, Nome do Contato na Empresa, Fone, Fax, E-mail, Site e Informações Adicionais.
R.4.2	Cadastro de Projetos : Nome do Projeto, Código, Tipo de Projeto (FK)
...	

5. Requisitos - Relatórios

ID	Descrição
R.5.1	Os tipos de Relatórios que o Sistema deve ter são: impressão da tabela atual, imprimir atividades, tarefas, projetos feitos hoje, ontem, a uma semana, a um mês, a um dia específico, a uma semana ou a um intervalo de tempo específico.
...	
R.5.3	Itens de cabeçalho e rodapé - número de páginas, data, horas, número da página. A formatação (inclusive para o título e para todo o programa) deve ser configurável (bold, itálico, centralizado, etc).
...	

Quadro 1: Partes de um documento de requisitos

É produzido também, um arquivo contendo as dúvidas do diretor em relação aos requisitos do sistema, as quais são sanadas na reunião seguinte com o cliente, e então o arquivo é atualizado (coloca-se a resposta abaixo da dúvida). Esse arquivo só foi produzido para o segundo projeto.

Para a produção desses dois artefatos, é utilizado como ferramenta o programa Word. O diretor afirmou que esses artefatos deveriam ser produzidos por um gerente do projeto ou desenvolvedor analista, mas devido à falta de recursos humanos, o Diretor de Projetos assumiu esses papéis.

O Quadro 2, mostra o processo de análise de requisitos em detalhes:

Objetivos	<p>Descrever os requisitos do sistema.</p> <p>Negociar com o cliente a proposta de projeto e o contrato.</p> <p>Selecionar desenvolvedores para atuar no projeto.</p>
Atividades técnicas	<p>2 - Gerente do projeto faz reunião com o cliente para levantar requisitos.</p> <p>3a - Gerente do projeto elabora o documento de requisitos (1.1) e a proposta de projeto (1.2) e negocia-os com o cliente, também elabora o arquivo de dúvidas (1.4) e esclarece-as com o cliente.</p>
Atividades administrativas	<p>1 - Diretor de projetos define gerente do projeto (normalmente o próprio diretor de projetos é o gerente).</p> <p>3b - Gerente do projeto e diretor de RH selecionam desenvolvedores.</p> <p>3c - Gerente do projeto elabora um plano de desenvolvimento, contendo o cronograma (1.5) do desenvolvimento.</p> <p>4 - Diretor administrativo-financeiro elabora os contratos (1.3) e providencia a nota fiscal.</p> <p>Diretor de projetos supervisiona o gerente do projeto.</p> <p>Gerente do projeto atualiza continuamente o seu documento de acompanhamento do trabalho (1.6) e o dos desenvolvedores (1.7).</p>
Papéis	<p>Gerente do projeto</p> <p>Diretor de projetos</p> <p>Diretor de RH</p> <p>Diretor administrativo-financeiro</p>
Técnicas/metodologias	<p>- Identificar requisitos:</p> <p>* descrever o sistema existente e as necessidades do cliente em documentos separados ("Descrição dos Sistemas da Empresa" e "Descrição de Necessidades do Cliente")</p> <p>* juntar tudo e a partir disso encontrar os módulos do sistema</p> <p>* formalizar esses três documentos num documento de requisitos (1.1) numerando-os</p>
Ferramentas	<p>MS-Word</p> <p>MS-Excel</p> <p>MS-Project</p>
Artefatos produzidos	<p>1.1 - documento de requisitos</p> <p>1.2 - proposta de projeto</p> <p>1.3 - contratos (NPI-Cliente e NPI-Desenvolvedores)</p> <p>1.4 - arquivo de dúvidas</p> <p>1.5 - cronograma</p>

	1.6 - documento de acompanhamento do trabalho do gerente
	1.7 - documento de acompanhamento do trabalho dos desenvolvedores
Critérios de entrada	Contato (telefone ou e-mail) de um possível cliente com o NPI.
Critérios de saída	Documento de requisitos (1.1) informalmente aprovado pelo cliente.

Quadro 2: Fase de análise de requisitos do processo atual do NPI

2.3.2 A modelagem do sistema

Durante a fase de modelagem, a análise do sistema e o projeto do sistema são feitos em paralelo, sem distinção entre essas etapas, e sem um procedimento definido. Essa fase também é realizada somente pelo diretor de projetos. A documentação é feita utilizando-se a notação da *Unified Modeling Language* (UML, ou Linguagem de Modelagem Unificada) e a modelagem Entidade-Relacionamento. O único critério para o início dessa etapa, é a especificação de requisitos informalmente aprovada pelo cliente. A partir dela, são gerados artefatos como esquema do banco de dados (Figura 2), diagrama de casos de uso (Figura 3), diagramas de atividades - um para cada caso de uso - (Figura 8), diagrama de classes (Figura 7), diagramas de seqüência para cada caso de uso (Figura 5), diagramas de colaboração para cada operação do sistema (Figura 6) e modelo conceitual (Figura 4). Esses artefatos não são gerados em uma seqüência definida, porém, são aperfeiçoados continuamente, mesmo depois do início da implementação.

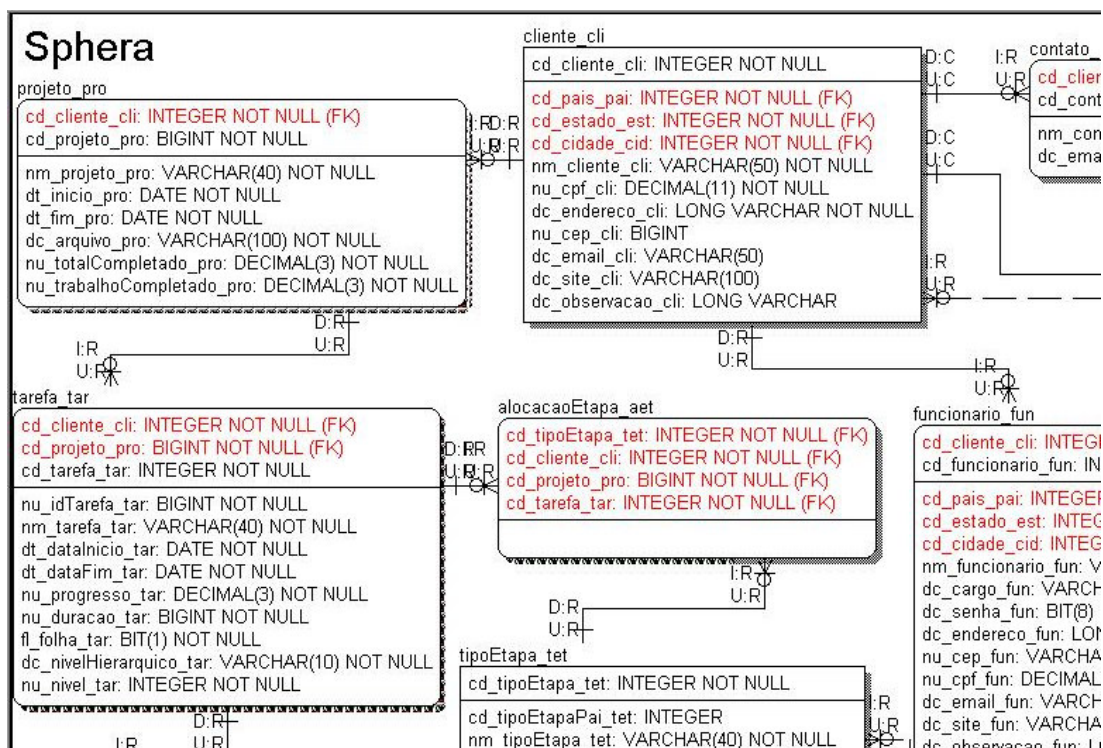


Figura 2: Extrato de um exemplo de esquema do banco de dados

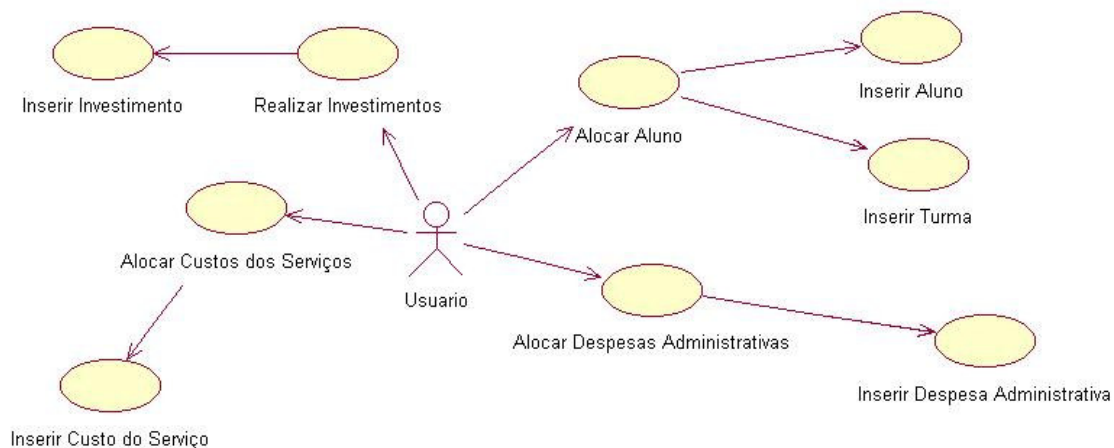


Figura 3: Exemplo de diagrama de casos de uso

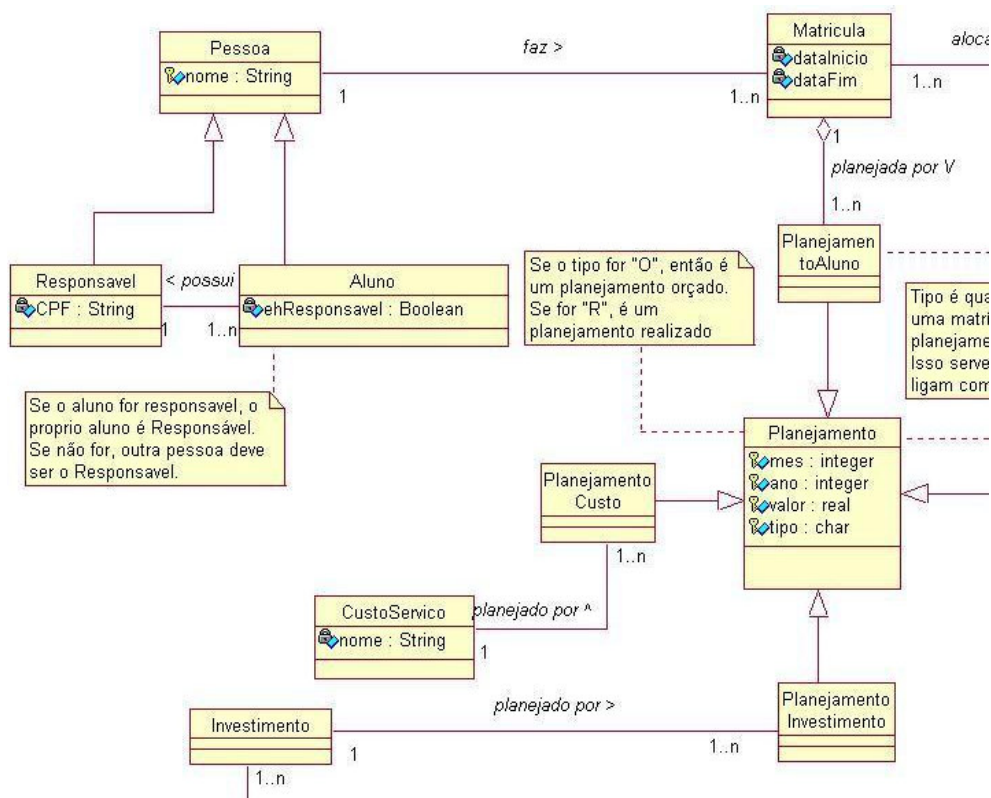


Figura 4: Extrato de um exemplo de modelo conceitual

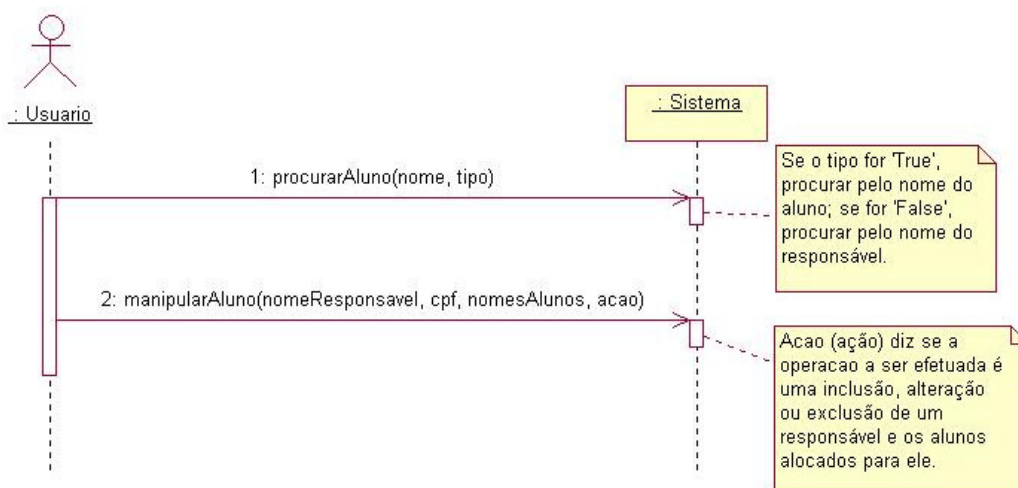


Figura 5: Exemplo de diagrama de seqüência, para um caso de uso

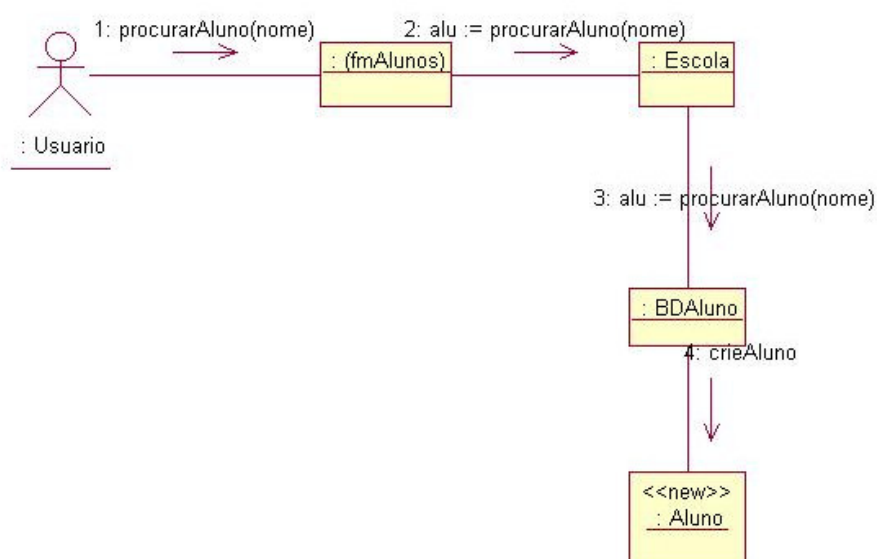


Figura 6: Exemplo de diagrama de colaboração para uma operação de sistema

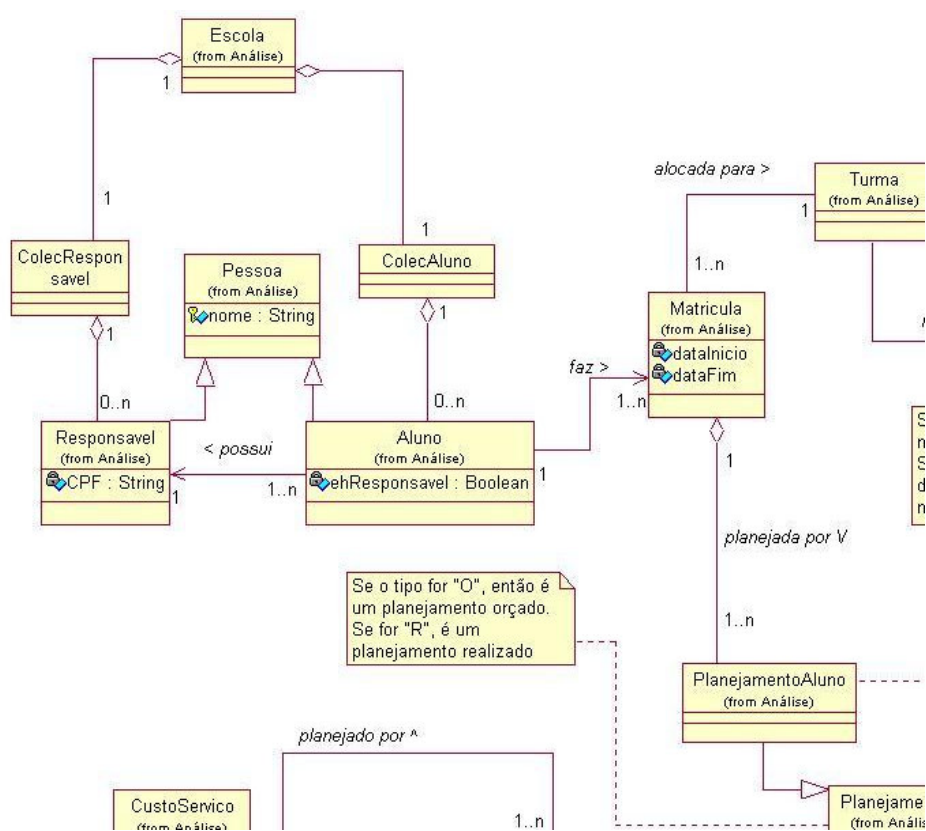


Figura 7: Extrato de um exemplo de diagrama de classes

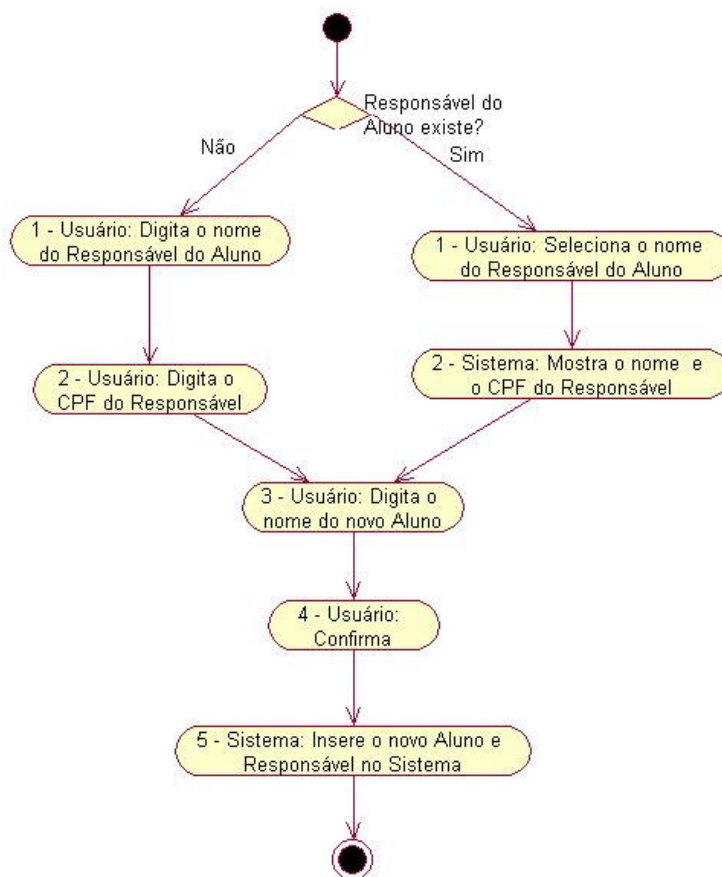


Figura 8: Exemplo de diagrama de Atividades

Para a produção do esquema do banco de dados, é utilizada a ferramenta ERWin e para os diagramas UML, a ferramenta Rational Rose.

Para o primeiro projeto, não foi utilizada nenhuma metodologia. Apenas visualizou-se um projeto de alto nível da arquitetura do sistema, que não ficou documentado; e foi produzido o esquema do banco de dados. A partir desses artefatos e do documento de requisitos foi implementado o sistema. Essa estratégia foi satisfatória porque o sistema desenvolvido era de baixa complexidade.

No segundo projeto, foi utilizada apenas a seguinte metodologia (documentada), para os casos de uso: fazer um esboço da interface com o usuário, e a partir dela, identificar os casos de uso. Nesse projeto, foram produzidos todos os artefatos citados e mostrados anteriormente nas figuras, para o primeiro caso de uso. Ao longo dessa produção, decidiu-se não utilizar a orientação a objetos pura e nem realizar a produção desses artefatos para os outros casos de uso, por falta de tempo e de recursos humanos. Assim, foram mantidos somente o diagrama dos casos de uso e o diagrama de atividades daquele caso de uso; os outros diagramas foram desconsiderados (mas não removidos, do arquivo do projeto no Rational Rose). A partir de então, foram somente produzidos diagramas de atividade para os outros casos de uso.

O Quadro 3, mostra um resumo do processo de modelagem do sistema.

Objetivos	Realizar a análise do sistema e o projeto do sistema.
Atividades técnicas	Gerente do projeto elabora o esquema do banco de dados (2.1). Gerente do projeto identifica casos de uso, faz o diagrama de casos de uso (2.2) e os diagramas de atividade de cada caso de uso (2.3).
Atividades administrativas	Gerente atualiza continuamente o documento de acompanhamento do seu trabalho (1.6). Diretor de projetos supervisiona o gerente do projeto.
Papéis	Gerente do projeto Diretor de projetos
Técnicas/metodologias	- Identificar casos de uso: fazer esboço da interface com usuário e a partir dela levantar os casos de uso. - UML
Ferramentas	Rational Rose ERWin
Artefatos produzidos	2.1 - esquema do banco de dados 2.2 - diagrama de casos de uso 2.3 - diagramas de atividade para cada caso de uso 2.4 - diagrama de classes 2.5 - diagramas de seqüência para cada caso de uso 2.6 - diagramas de colaboração para cada operação do sistema 2.7 - modelo conceitual
Critérios de entrada	Documento de requisitos (1.1) informalmente aprovado pelo cliente.
Critérios de saída	Esquema do banco de dados (2.1) pronto.

Quadro 3: Fase de modelagem do sistema do processo atual do NPI

2.3.3 A implementação

Na fase de implementação, o sistema é produzido em estágios incrementais e evolucionários. Em cada estágio é implementado um (ou parte de um) módulo. O critério de início dessa fase é o esquema do banco de dados e o diagrama de atividades do caso de uso atual (esse, utilizado somente no segundo projeto). O

sistema é baseado nos casos de uso e no modelo do banco de dados. Por exemplo, para um determinado caso de uso, há uma interface com o usuário, que também interage com o banco de dados. Os módulos são implementados em ordem crescente de dependência, ou seja, os módulos que são menos dependentes dos outros são implementados primeiro. Nessa fase, os desenvolvedores produzem o código e o diretor de projetos verifica seu trabalho. Os desenvolvedores tentaram seguir um padrão de comentários mas desistiram por falta de tempo e cobrança do diretor. Os únicos padrões existentes são de indentação e nome de variáveis. Nos dois projetos, foi utilizada a linguagem de programação Object Pascal, presente na ferramenta Borland Delphi, e para o banco de dados, foi utilizado o PostgreSQL (principalmente por ser uma ferramenta gratuita). Também foi explicado, que a utilização de um banco de dados entidade-relacionamento, apesar do sistema ser parcialmente orientado a objetos, se deve por que esse era o único tipo de banco de dados com o qual o diretor estava familiarizado.

No fim de cada estágio, são feitos testes informais, realizados pelos próprios desenvolvedores. Os desenvolvedores e o diretor afirmaram que não são planejados nem preparados testes formais, por falta de experiência deles na área, desconhecimento de metodologias de teste e pouca abordagem na literatura sobre o assunto.

Ao final dessa fase, tem-se o código executável e as tabelas do banco de dados completas.

No Quadro 4, está o resumo da fase de implementação do processo atual do NPI.

Objetivos	Escrever o código, baseado no esquema do banco de dados e dos diagramas de atividades dos casos de uso.
Atividades técnicas	1 - Desenvolvedores escrevem o código. 2 - Desenvolvedores realizam testes informais sobre o código.
Atividades administrativas	Gerente do projeto supervisiona os desenvolvedores. Diretor de projetos supervisiona o gerente do projeto. Desenvolvedores e gerente atualizam os seus documentos de acompanhamento do trabalho (1.6 e 1.7).
Papéis	Gerente do projeto Desenvolvedor
Técnicas/metodologias	- Implementar os módulos em ordem crescente de dependência. - Seguir um padrão de indentação e de nome de variáveis.
Ferramentas	PostgreSQL Borland Delphi
Artefatos produzidos	3.1 - código 3.2 - tabelas do banco de dados
Critérios de entrada	Esquema do banco de dados (2.1) pronto. Diagrama de atividades (2.3) do caso de uso atual (opcional) Terminado.
Critérios de saída	Código executável (3.1). Banco de dados com as tabelas (3.2).

Quadro 4: Fase de implementação do processo atual do NPI

2.3.4 Aspectos de gerência e qualidade do processo

Em paralelo com a análise de requisitos, são elaborados os contratos legais (do NPI com o cliente e do NPI com os desenvolvedores), um cronograma e a Proposta de Projeto. Para o primeiro projeto ainda não havia sido elaborado um contrato com os desenvolvedores e o cronograma, depois de elaborado, não foi

atualizado. No segundo projeto, o desenvolvimento havia iniciado antes do cronograma e dos contratos estarem prontos.

A Proposta de Projeto é um documento que, juntamente com o contrato, é negociado com o cliente. Há no NPI arquivos padrão de contrato e de Proposta de Projeto (entre outros documentos) que são alterados conforme as necessidades de cada projeto. A Proposta de Projeto possui os seguintes itens:

- * Breve descrição dos objetivos do software.
- * Modo como será realizado o acompanhamento do desenvolvimento pelo cliente (normalmente com relatórios quinzenais e protótipos mensais).
- * Prazo de entrega do produto.
- * Serviços incluídos no contrato (normalmente desenvolvimento, implantação, documentação para o usuário e treinamento).
- * Garantia do software (normalmente de três meses).
- * Vigência da Proposta de Projeto (normalmente de trinta dias).
- * Cronograma financeiro - o pagamento normalmente é dividido em parcelas - com um valor adicional opcional para o código fonte.
- * Anexos opcionais (documento de requisitos e glossário).

Ainda em relação à gerência do processo, para o primeiro projeto, havia sido feito um plano de desenvolvimento, contendo o cronograma. Nesse plano, o desenvolvimento era dividido em três ciclos, com atividades e prazos bem definidos para cada um. Mas, por inexperiência do diretor, as estimativas de tempo foram mal formuladas (não se sabia quanto tempo iria levar cada etapa do trabalho e elas acabaram levando mais tempo do que o previsto) e então o plano foi abandonado. A

partir daí, foram feitos apenas acompanhamentos semanais do diretor, verificando o resultado do trabalho dos desenvolvedores.

Por causa disso, os desenvolvedores estavam trabalhando horas a mais do que o cobrado para tentar terminar o software com apenas uma semana de atraso.

Apesar dos fatos explicados, os desenvolvedores e o diretor afirmaram que o cliente estava ficando satisfeito com os protótipos que vinham sendo apresentados. Essa afirmação foi aceita como verdadeira, pois o cliente estava negociando com o NPI, o desenvolvimento de mais dois softwares.

Para o segundo projeto, não foi feito um plano de desenvolvimento.

Em paralelo com as fases de análise de requisitos e modelagem do sistema é feita a seleção dos desenvolvedores que irão trabalhar no projeto. Essa etapa é realizada, normalmente, pelo diretor de projetos e o diretor de recursos humanos em conjunto. São feitas entrevistas com os associados do NPI interessados no projeto, e em seguida, os dois diretores decidem quais dos associados serão escolhidos.

Em relação ao segundo projeto, foi produzido (e estava sendo atualizado) um documento (Figura 9) com o acompanhamento do trabalho do gerente (que no segundo projeto também estava atuando como desenvolvedor), contendo o horário de início e fim de cada período de trabalho e o que foi produzido naquele período. Porém, não estava sendo produzido um documento equivalente para acompanhar o trabalho do outro desenvolvedor, por inexperiência do desenvolvedor, e falta de

cobrança do diretor. Para o primeiro projeto, foram produzidos documentos de acompanhamento semelhantes ao explicado para os dois desenvolvedores e para o gerente, mas esses documentos não foram atualizados até o fim do projeto por falta de tempo.

	A	B	C	D	E	F	G
1	Data Início	Início	Fim	Tarefa	Duração		
2	18/04/2003	14:00	23:00	Criação dos Casos de uso	09:00		
3				Esboço da Interface			
4				Desenvolvimento do Glossário			
5				Desenvolvimento do Arquivo de Dúvidas			
6				Desenvolvimento do manual de Gerenciamento e Desenvolvimento de Software (geral)			
7	19/04/2003	11:00	15:00	Criação dos Casos de uso	04:00		
8				Esboço da Interface			
...							
35	25/04/2003	01:50	02:30	Criação do Modelo Conceitual	00:40		
36				Esboço da Interface			
37				Desenvolvimento do Glossário			
38				Desenvolvimento do Arquivo de Dúvidas			
39	30/04/2003	20:00	21:00	Refinamento do Modelo Conceitual	01:00		
40				Criação das units			
41				Modelagem ER			
42	01/04/2003	00:30	04:00	Refinamento do Modelo Conceitual	03:30		
43				Criação das units			
44				Modelagem ER			
45	02/04/2003	16:00	22:00	Refinamento do Modelo Conceitual	06:00		
46				Criação das units			
47				Criação do BD			
48				Esboço da Interface			

Figura 9: Extrato de um exemplo de documento de acompanhamento do desenvolvimento

Também em aspectos de qualidade, os desenvolvedores afirmaram que o código desenvolvido possui pouca reusabilidade.

2.3.5 Discussão

Com base nessa análise, conclui-se que não existe no NPI um processo padrão de desenvolvimento de software. Apesar disso, uma das conclusões a que se pode chegar é que o processo realizado na fase de análise de requisitos é

eficiente e adequado ao contexto do NPI, com exceção da transição entre as fases, que é um pouco nebulosa - pois a fase de modelagem é iniciada antes que sejam completados todos os produtos necessários para finalizar a fase de análise de requisitos. Entretanto, a fase de modelagem é bastante aleatória. Os documentos produzidos nem sempre são aproveitados, os que são aproveitados não são suficientes para traduzir o projeto em código, a produção dos documentos é desordenada, e novamente, a transição entre essa fase e a próxima, é um pouco nebulosa (pois, às vezes, as duas fases ocorrem em paralelo). Além disso, a arquitetura do sistema proíbe qualquer tentativa de reusabilidade. Porém, é importante ressaltar que toda essa aleatoriedade é consequência de um esforço por parte do diretor de projetos de melhorar o processo de desenvolvimento através de um método informal de "tentativa e erro". A implementação é bastante eficiente, porém inconsistente com a modelagem (pois os diagramas de atividades não se traduzem diretamente em código). Apesar disso, também se nota um esforço por parte dos desenvolvedores de adotar algum tipo de sistematização do processo, como por exemplo padrões de indentação e de nomes de variáveis. A fase de testes é a mais carente em termos de metodologia, pois não há um planejamento dos testes e nem o correspondente cumprimento desses planos de uma maneira sistematizada. São realizados apenas testes informais, o que é considerado ineficaz pela literatura (SILVA, 2002). Porém, também deve ser ressaltado que há um interesse dos desenvolvedores e principalmente do diretor, em seguir uma metodologia de testes.

O ciclo de vida do software desenvolvido no NPI, não é um processo com a separação clara das atividades de cada fase. Porém, em geral, pode ser identificada a seguinte seqüência de fases: análise de requisitos; modelagem do sistema;

implementação e testes informais do primeiro módulo; implementação e testes informais do segundo módulo; e assim por diante, até o último módulo; entrega para o cliente. Assim, pode-se dizer que o modelo de ciclo de vida atual do NPI é uma mistura de cascata com iteratividade na implementação e teste de módulos. A Figura 10, ilustra essa conclusão:

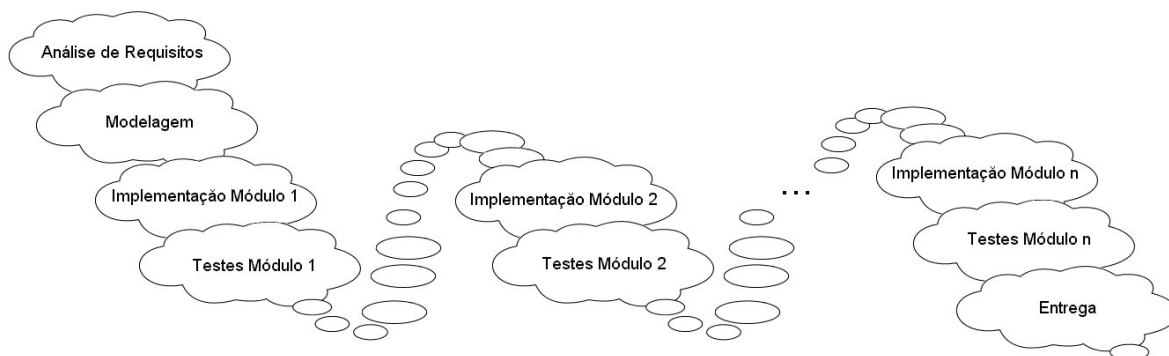


Figura 10: Modelo de ciclo de vida atual do NPI

Em relação à documentação produzida durante o processo de desenvolvimento, também se pôde perceber que não há nenhuma sistematização quanto ao uso de ferramentas independentes de plataforma. Em termos de administração, constata-se que há uma falha geral em todas as fases nos processos de gerência de configuração, controle de documentos e supervisão administrativa. Isso é conseqüência de uma grande falta de experiência em exercício de cargos administrativos, o que é constatado tanto na diretoria de projetos como nas outras diretorias do NPI.

O Quadro 5, resume o processo existente no NPI, em termos da qualificação dos documentos produzidos.

Fase	Documentos Produzidos	P1	P2
Análise de Requisitos	* Documento Formal de Requisitos	A, C	A, C
	* Arquivo de Dúvidas	N	A, C
Modelagem	* Esquema do Banco de Dados	A, C	A, C
	* Diagrama de Casos de Uso	N	A, C
	* Diagramas de Atividades	N	A
	* Diagrama de Classes	N	U
	* Diagramas de Seqüência	N	U
	* Diagramas de Colaboração	N	U
Implementação	* Modelo Conceitual	N	U
	* Código	C	A
Gerência do Processo	* Contratos Legais (NPI-Cliente e NPI-Desenvolvedores)	I	I
	* Proposta de Projeto	A, C	A, C
	* Plano do Desenvolvimento	U	N
	* Cronograma	U	U
	* Acompanhamento do trabalho dos desenvolvedores e do gerente	D, C	A, I

Legenda:	
P1	Produzido no Primeiro Projeto
P2	Produzido no Segundo Projeto
C	Completo
I	Incompleto
A	Atualizado
D	Desatualizado
N	Não foi produzido nesse projeto
U	Produzido, porém não utilizado em fases posteriores

Quadro 5: Resumo do processo atual do NPI

2.4 Análise das Necessidades do NPI

A partir dos dados coletados nas entrevistas e da inspeção dos documentos dos projetos enviado-nos pelo diretor, pôde-se perceber a realidade do processo de desenvolvimento do NPI. Através da análise dessa realidade, conclui-se sobre as necessidades do NPI em relação a um processo de desenvolvimento de software.

Os requisitos do NPI para um modelo de processo de desenvolvimento do software são:

- * Possuir um modelo de processo para cada fase citada anteriormente (análise de requisitos, modelagem, implementação e testes), integrado a atividades de gerência administrativa e a processos de gerência de configuração e controle de documentos.
- * Definir um roteiro de como realizar essas fases e oferecer documentos padrão (com partes que possam ser alteradas ou preenchidas, diferentemente em cada projeto) para os documentos a serem produzidos.
- * Ser voltado a pessoas sem grande experiência e conhecimento na área de Engenharia de Software, e necessitar que elas passem por pouco tempo de treinamento (por exemplo, 10 horas) para entender como realizar o processo.
- * Ser baseado no processo existente, melhorando-o nos seus pontos mais fracos.
- * Ser adequado ao contexto do NPI (escassez de pessoal e de tempo, orçamentos pequenos, e outras características específicas de uma empresa júnior). Deve incluir apenas a produção de documentos que serão aproveitados em outras fases, e não a produção de documentos que depois não serão utilizados.
- * Facilitar e auxiliar o trabalho colaborativo, por intermédio ou não de computador.

- * Possuir um meio, sistematizado e baseado em indicadores, para continuamente melhorar o próprio modelo. Ou seja, o modelo deve estabelecer uma metodologia para a sua alteração; e essa alteração deve ser baseada em dados reais e mensuráveis, observados no processo de desenvolvimento de software do NPI.

3 Conceitos Fundamentais

Neste capítulo, serão definidos os termos, no contexto de modelagem de processo, que serão utilizados a partir de agora neste trabalho. Alguns desses termos são conceitos da área da Engenharia de Software. Outros, entretanto, são substantivos com significados genéricos e semelhantes (tais como tarefa, atividade e fase) - comumente encontrados na literatura, mas com significados diferentes para cada autor - aos quais serão atribuídos um significado único e específico, o qual será utilizado na continuidade deste trabalho. Ressalta-se, mais uma vez, que são do interesse deste trabalho apenas os conceitos relacionados com a área de Modelagem de Processo de Software, uma sub-área da Engenharia de Software.

3.1 O que é um Processo de Software?

Um **processo** é uma seqüência de atividades ou acontecimentos, ordenados ou não, que levam a um resultado. Mas o que é um **processo de software**? Partindo da definição anterior, um processo de software é o conjunto de atividades realizadas para atingir o objetivo principal de desenvolver um software (e, também partindo da definição anterior, essas atividades podem ser realizadas de uma maneira aleatória ou de uma maneira sistematizada).

Segundo LARMAN (2000, p. 40), "um processo [sistematizado] de desenvolvimento de software é um método para organizar as atividades relacionadas com a criação, entrega e manutenção de sistemas de software." O SEI fornece uma definição ainda mais completa:

Um *processo de software* pode ser definido como um conjunto de atividades, métodos, práticas e transformações que as pessoas utilizam para desenvolver e manter software e seus produtos associados (por exemplo, planos de projeto, documentos de projeto do sistema, código, casos de teste e manuais do usuário). (PAULK et al., 1993, p.3, tradução livre)

Essas duas citações dão uma amostra da falta de consenso que existe na literatura em relação à definição do que é um "processo de software". Pois, o que foi definido por "processo de software" é chamado por alguns autores de "processo de engenharia de software" (KEHOE e JARVIS, 1996) ou "processo de desenvolvimento de software" (LARMAN, 2000).

Neste trabalho, diferenciar-se-á o conceito de **processo de software** do conceito de **processo de desenvolvimento de software**. Assim, para o este estudo, um processo de desenvolvimento de software aborda apenas os aspectos relativos ao desenvolvimento (análise de requisitos e de sistema, projeto de sistema, implementação e testes) e um processo de software aborda, além dos aspectos relativos ao desenvolvimento, também os aspectos de manutenção, de subcontratação, entre outros.

3.2 Modelo de Processo de Software

Um **modelo de processo de software** define o que deve ser realizado em cada fase do desenvolvimento e dá as instruções de como realizar essas atividades. Um modelo serve como um guia, um roteiro para a execução de um processo de desenvolvimento. KELLNER et al. (1998) chamam o modelo de “guia do processo”. "Um guia do processo é um documento de consulta para um certo processo, orientando os participantes do processo na sua condução" (Idem, op.cit., p.1, tradução livre).

Esse guia pode tomar a forma de um software, utilizado para acompanhar o desenvolvimento, ou seja, uma ferramenta CASE (*Computer-Aided Software Engineering*, ou Engenharia de Software Auxiliada por Computador), ou então de um manual impresso, a ser lido e consultado durante o desenvolvimento de um software. KEHOE e JARVIS (1996) chamam esse manual impresso de *Software Process Handbook* (ou SPH). KELLNER et al. (1998) também mencionam essas duas opções chamando o manual impresso de *paper-based process guide* e o manual em meio eletrônico de EPG (*Electronic Process Guide*). Pode-se perceber mais uma falta de consenso na literatura, desta vez quanto à definição de um nome padrão para o que foi definido como modelo de processo de software.

Como explicado inicialmente, o presente trabalho propõe-se a desenvolver um guia impresso.

Um modelo de processo pode ser muito ou pouco flexível, por exemplo, ele pode permitir a escolha de um modelo de ciclo de vida diferente em cada desenvolvimento ou determinar que todos os desenvolvimentos irão usar o mesmo modelo de ciclo de vida. Além disso, ele pode ser bastante detalhista ou mais alto-nível: pode definir sub-atividades dentro de cada atividade ou pode apenas dar uma definição geral do que deve ser realizado em cada fase do desenvolvimento. Com tantas possibilidades, é fácil perceber que um modelo deve ser adequado à realidade de uma empresa ou de um tipo de empresas, por exemplo, um modelo para grandes empresas deve ser diferente de um modelo para pequenas empresas. Apesar disso, são muito recentes os estudos de modelos de processo de software para micro e pequenas empresas (os quais serão abordados no capítulo 4).

3.3 Qualidade do Processo

Segundo ROCHA (2002), um **software de qualidade** é aquele que atende às necessidades de seu usuário. Tecnicamente falando, essas necessidades se traduzem em "requisitos do sistema". Assim, um software de qualidade é aquele que atende aos requisitos especificados inicialmente no seu planejamento.

Como já explicado anteriormente, "a qualidade de um produto de software é fortemente dependente da **qualidade do processo** pelo qual ele é construído e mantido" (Idem, op. cit., p.9, grifo nosso). Ou seja, a utilização de um processo metódico e sistematizado tem muito mais chances de produzir um software de qualidade do que a utilização de um processo caótico e desordenado. Por "processo metódico e sistematizado", entende-se por "um processo cujas etapas são

detalhadas de tal forma que engenheiros podem consistente e repetidamente utilizá-lo" (WEBER, 2002, p.25). Ou seja, para cada fase do ciclo de vida são descritas as atividades a serem realizadas e os responsáveis por cada uma delas, os documentos a serem produzidos e seus conteúdos, as condições de entrada e de saída, as ferramentas e metodologias a serem utilizadas, entre outros.

"A implantação de um Programa de Qualidade começa pela definição e implantação de um processo de software [de qualidade]" (ROCHA, 2002, p.12). De acordo com ULRIKE, HAMANN e VERLAGE (1997), a tarefa de implantação (ou melhora da qualidade) de um processo de software engloba duas atividades principais: a elaboração de um **modelo de processo descritivo** e depois, de um **modelo de processo prescritivo**. "Um modelo descritivo retrata como um processo é executado em um ambiente em particular; um modelo prescritivo retrata como um processo *deveria* ser executado" (Idem, op. cit., p.2, tradução livre). Essa abordagem é utilizada no presente trabalho: o capítulo 2 apresenta o modelo descritivo do processo do NPI e o capítulo 5 apresenta o modelo prescritivo.

Existem diversos modelos para a avaliação e melhora do processo de software. A parte 9 do relatório Software Process Assessment (SPICE PROJECT ORGANISATION, 1995) apresenta várias definições relacionadas à avaliação (ou certificação) e melhora de processo. "[...] **avaliação do processo**: Uma avaliação disciplinada dos processos de software de uma organização comparando-os a um modelo de processo [...]" (Idem, op. cit., p.9, tradução livre). Outra definição interessante presente no relatório é a definição de **determinação da capacidade do processo**:

[...] avaliação e análise sistemáticas de processos de software selecionados em uma organização comparando-os a uma capacidade alvo, realizada com o objetivo de identificar os pontos fortes e fracos e os riscos associados na mudança dos processos para que eles atendam um certo requisito especificado (SPICE PROJECT ORGANISATION, 1995, p.9, tradução livre).

O conceito de melhora do processo também é definido de maneira bastante precisa. "[...] **melhora do processo**: Ação tomada para mudar os processos de uma organização para que eles satisfaçam as necessidades da organização e atinjam os seus objetivos de negócio mais eficientemente" (Idem, op. cit., loc.cit., tradução livre).

De acordo com ROCHA (2002), a avaliação - ou certificação - do processo pode ser feita tanto pela própria empresa desenvolvedora quanto por terceiros (i.e. auditores independentes). "Um guia de processo pode também abranger as atividades de planejamento da execução do processo e de certificação" (KELLNER et al.,1998, p.5, tradução livre).

Os principais modelos de avaliação do processo são CMM, SPICE (ou ISO/IEC 15504) e ISO 9000-3, sendo que o SPICE também é um modelo de melhora do processo. Além deles, também existem outros dois modelos (que facilitam a implementação do CMM) os quais considera-se importante mencionar, são eles: PSP (um modelo de métricas individuais de trabalho) e TSP (um modelo de gerência do processo de software). Todos esses modelos serão discutidos mais detalhadamente no capítulo 4.

Como é explicado no capítulo 4, a literatura da área de Processo de Software se concentra principalmente nos modelos de avaliação do processo descritos acima. Muito se diz sobre "o que é um processo de qualidade" e "como avaliar um processo" mas raramente se comenta sobre "como elaborar um modelo de processo". Entende-se que a explicação para esse fato, é que geralmente a modelagem de processos é feita por empresas de consultoria, que não iriam revelar sua "metodologia de desenvolvimento de modelos de processo" pois ela é a base do seu negócio (seria como o fabricante da Coca-Cola revelar a sua fórmula). Outra explicação pode ser simplesmente por que a área de modelagem de processo de software é ainda muito nova, e existe pouca pesquisa sobre o assunto.

3.4 Modelos de Ciclo de Vida

Outro conceito relacionado com processo de software é o conceito de **modelo de ciclo de vida**. O modelo de ciclo de vida de um software determina as fases pelas quais o desenvolvimento de um software deve passar e, principalmente, a ordem e o número de repetições dessas fases. Enquanto o ciclo de vida trata apenas da ordenação das fases técnicas do desenvolvimento, um processo abrange aspectos gerenciais e mais detalhados, como, definição de papéis, ferramentas e métricas. Apesar disso, o processo de desenvolvimento de um software está intimamente ligado com o modelo de ciclo de vida adotado para desenvolvê-lo. Não pode haver um modelo de processo definido sem um ciclo de vida definido.

A seguir, serão explicados brevemente alguns modelos de ciclo de vida. Os exemplos escolhidos são os mais citados na literatura consultada, e além disso, percebeu-se que a partir deles são derivados muitos outros - o que leva a concluir que esses modelos são os fundamentais.

3.4.1 Exemplos de modelos

Code-and-Fix: Esse tipo de ciclo de vida se caracteriza por duas fases: produção de código (*code*) e conserto do código quando são detectados erros (*fix*). Essa seqüência é repetida até que se esteja satisfeito com o código resultante. Apesar de ser considerada uma técnica um tanto primitiva, ainda é muito utilizada. A Figura 11, mostra como a produção do software por meio desse modelo é uma atividade "nebulosa".

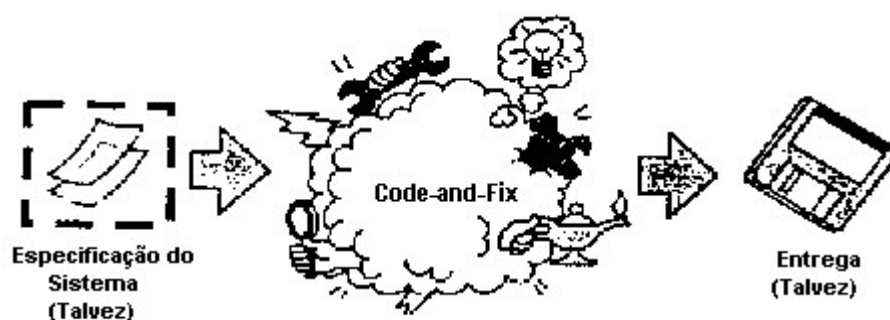


Figura 11: Modelo de ciclo de vida code-and-fix
(McCONNELL apud WEBER, 2002, p.30)

Cascata: As fases (normalmente: análise, projeto, implementação e testes) são executadas em seqüência, sendo que uma só inicia após a outra haver sido

completamente terminada. Além disso, uma vez terminada a fase, os resultados dela não são mais alterados (por exemplo, durante a fase de projeto não há a possibilidade de alteração dos requisitos, documentados na fase da análise) . Por esse motivo, o método cascata não é muito utilizado. Na prática, ele é combinado com um ou mais modelos para resolver o problema. A Figura 12 ilustra a seqüência de fases como explicado:

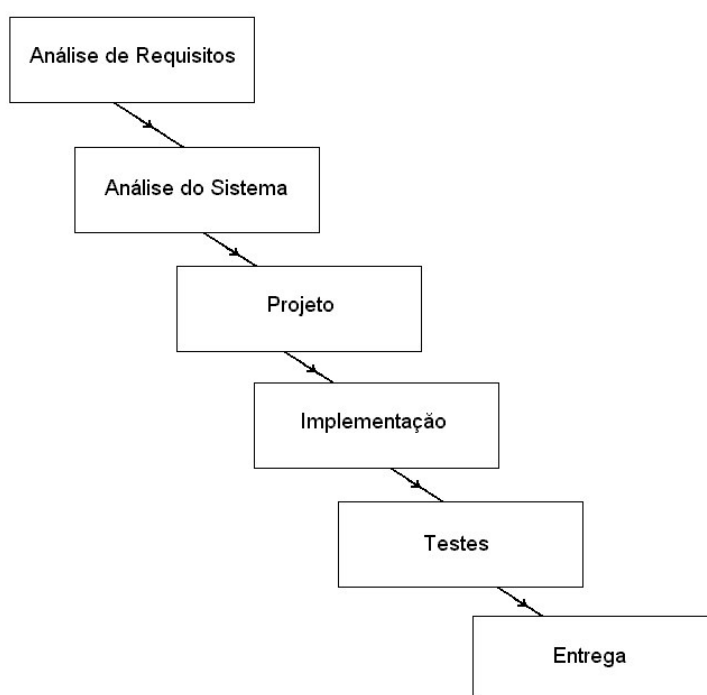


Figura 12: Modelo de ciclo de vida cascata

Modelo em V: É semelhante ao cascata, diferenciando-se apenas em relação aos testes. Durante as fases de análise e projeto são planejados os testes e durante a fase de implementação eles são preparados. Na fase de testes, eles são apenas executados. Esse modelo é melhor do que o cascata porque leva a procedimentos de teste mais criteriosos e tende a diminuir a quantidade de erros não identificados (SILVA, 2002). Na Figura 13, pode-se observar a seqüência das fases e como os

testes são projetados, produzidos e executados. Ela também permite entender o motivo do nome do modelo.

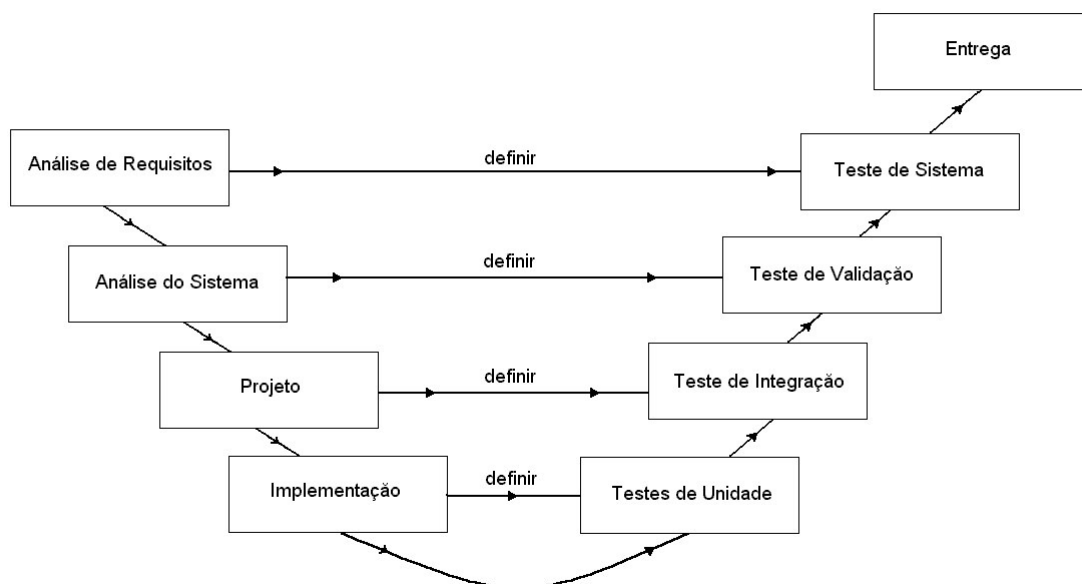


Figura 13: Modelo de ciclo de vida em V
(baseada em SILVA, 2002)

Desenvolvimento Incremental: Esse modelo é considerado um dos melhores, pois resolve o principal defeito do cascata: permite que os documentos de uma fase sejam melhorados (ou *incrementados*) mesmo após essa fase haver sido completada. Isso ocorre porque o desenvolvimento é baseado em ciclos. Um ciclo constitui uma seqüência das fases de análise, projeto, implementação e testes. Os requisitos podem ser divididos grupos, e cada grupo é implementado em um ciclo. Assim, muda-se de uma fase para a próxima, mesmo sem ela haver sido completamente terminada. Além disso, o desenvolvimento incremental permite que, por exemplo, ao encontrar um requisito do sistema durante a fase de projeto (o que é **bastante** comum), pode-se incluir esse requisito no documento de requisitos, na fase de análise do próximo ciclo.

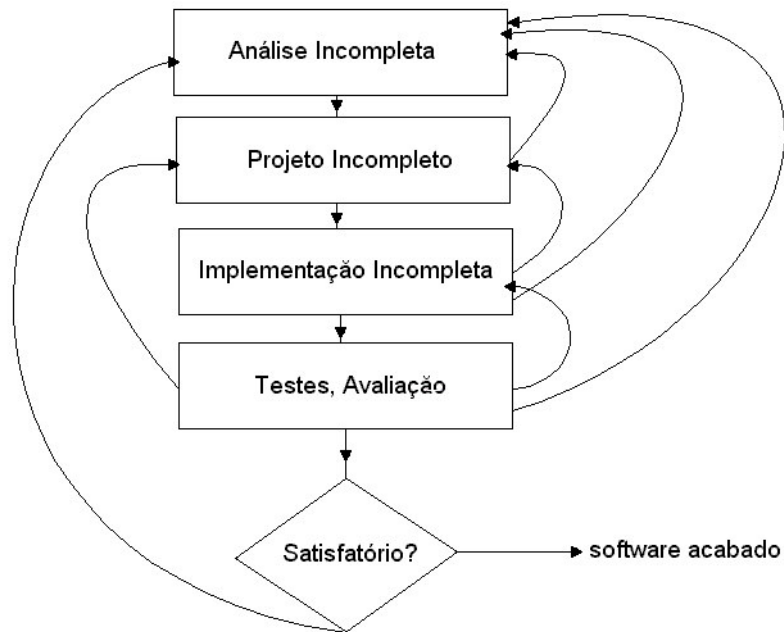


Figura 14: Modelo de ciclo de vida desenvolvimento incremental (baseada em SILVA 2002)

Espiral: Segundo WEBER (2002, p.37), o modelo espiral, "é um modelo orientado a riscos que reparte o projeto em mini-projetos. Cada mini-projeto resolve um ou mais riscos maiores até que os riscos maiores sejam resolvidos." Ao final da espiral, os grandes riscos estão eliminados, e pode-se continuar o desenvolvimento com outro modelo de ciclo de vida. Entretanto, esse modelo necessita de um gerenciamento de riscos um tanto complexo, sendo recomendado apenas para equipes mais experientes. A Figura 15, ilustra mais detalhadamente esse modelo.

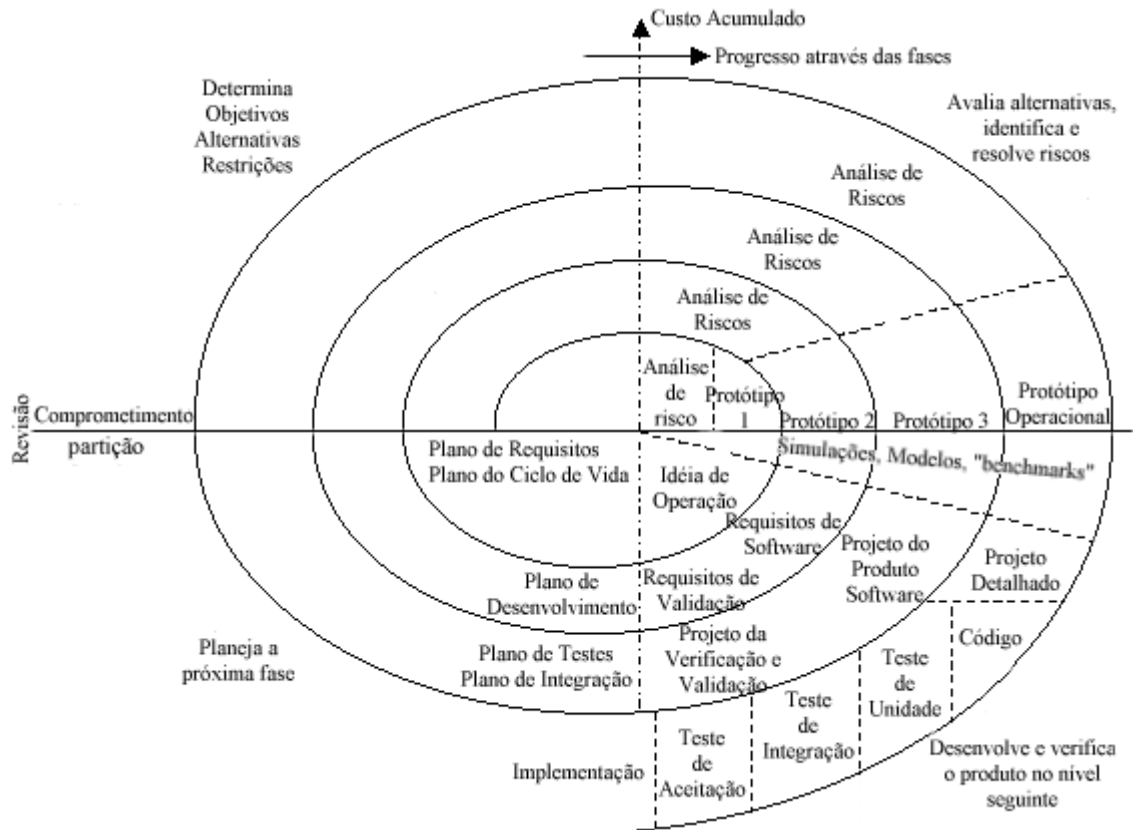


Figura 15: Modelo de ciclo de vida em espiral (BOEHM apud WEBER, 2002, p.38)

3.4.2 Discussão

Entende-se que um novo modelo de ciclo de vida para o NPI não pode ser baseado no code-and-fix, pois, além desse modelo ser considerado ineficiente, isso seria um retrocesso no seu desenvolvimento de software. O modelo espiral também é inapropriado por que a análise de riscos exige equipes experientes nessa área, o que certamente não é o caso do NPI. O principal defeito, anteriormente comentado, dos modelos cascata e em V é o fato deles não possibilitarem a revisão dos artefatos já criados. Pode-se dizer que devido aos projetos que o NPI realiza

(sistemas pequenos e médios sob encomenda) e à sua equipe (pequena e com pouca experiência), a mudança - principalmente dos requisitos - durante o desenvolvimento é bastante comum. Assim, a utilização do modelo cascata ou em V, sozinho (i.e. sem combinações) não é adequada para o NPI. Essa necessidade de revisar e reelaborar os artefatos do desenvolvimento não é exclusiva do NPI, muitas empresas também possuem essa necessidade, e é por isso que o modelo de desenvolvimento incremental (e suas variações) é hoje o mais aceito pela comunidade. Assim, acredita-se que um modelo de ciclo de vida para o NPI deve ser baseado principalmente no modelo de desenvolvimento incremental combinado com outro(s) modelo(s). A discussão do modelo de ciclo de vida escolhido para o NPI será abordada no capítulo 5.

3.5 Conceitos Complementares

Outro conceito, já bastante utilizado neste trabalho, e que também não possui uma definição padrão na literatura, é o conceito de fase. Neste trabalho, quando se fala em **fase**, está se referindo a uma das principais partes do desenvolvimento de um software, como por exemplo: Análise de Requisitos, Análise do Sistema, Projeto do Sistema, Implementação, Testes.

Quando é mencionado o termo **atividade** - outra palavra que também não possui uma definição padrão na literatura, e muitas vezes se confunde com “fase” - está se referindo a uma tarefa a ser executada em uma determinada fase. Por

exemplo, na fase de análise de requisitos, deve-se realizar a atividade "elaborar o documento de requisitos".

O termo **papel** tem uma definição bastante precisa na literatura, a qual será adotada também para este trabalho. Papéis são "descrições de um conjunto de obrigações e permissões relacionadas à execução de atividades" (KELLNER et al., 1998, p.9, tradução livre). Simplificando, um papel é uma responsabilidade assumida por uma ou mais pessoas. Alguns exemplos de papéis são: gerente de projeto, desenvolvedor, testador, analista de requisitos e diretor de projetos. A definição de papéis durante o planejamento do desenvolvimento é importante para que se tenha explicitamente determinado "quem vai fazer o quê".

O conceito de **artefato** também não possui uma definição padrão, mas ele é usado de maneira relativamente consistente por todos os autores. Artefatos são "descrições dos produtos criados ou modificados durante a execução do processo [...]" (Idem, op. cit., loc. cit., tradução livre). Para este trabalho, o termo artefato representa o resultado de uma das fases do desenvolvimento. A palavra **documento** é algumas vezes utilizada como sinônimo de artefato, o qual será também considerado no presente trabalho. Classifica-se o artefato, ou documento, em dois tipos: **técnico** (por exemplo: diagramas de atividades, documento de requisitos e código-fonte) e **gerencial** (produtos administrativos, como: documento de aceitação do software pelo cliente, contrato com o cliente, planilha de acompanhamento do trabalho dos desenvolvedores e plano do desenvolvimento). Além disso, os artefatos podem ser compostos de outros artefatos (inclusive de tipos diferentes). Por exemplo, o plano do desenvolvimento - que é um artefato gerencial - pode conter o cronograma - que também é um artefato gerencial - e mais o documento de requisitos e o diagrama de casos de uso - ambos artefatos técnicos.

A palavra **ferramenta** também é utilizada de maneira uniforme na literatura. Uma ferramenta é um recurso utilizado para a produção de um artefato. Pode ser um recurso computacional (como MS Word, Borland Delphi e Rational Rose) ou não (como lápis e papel).

4 Estado da Arte e Prática

Neste capítulo, serão apresentadas as idéias e teorias mais aceitas atualmente na área de modelagem de processos. E também, serão analisados relatos de experiências que de alguma forma estão ligadas ao presente trabalho.

4.1 Visão Geral de Modelos de Avaliação de Processo

Como explicado anteriormente, serão agora apresentados os principais modelos de avaliação de processo aceitos pela comunidade de engenharia de software (CMM, SPICE e ISO 9000-3) e outros dois modelos (um de métricas individuais de trabalho e outro de gerência do processo de software) não tão citados na literatura mas que considera-se importante mencionar (PSP e TSP respectivamente).

4.1.1 CMM - Capability Maturity Model

É um Modelo para analisar a Maturidade de uma empresa em termos da Capacidade do seu processo de software (PAULK et al., 1993). Esse modelo foi desenvolvido no início dos anos 90 pelo SEI (Instituto de Engenharia de Software da Universidade Carnegie Mellon, Estados Unidos), como um projeto para o Departamento de Defesa (DoD) americano, que era um grande comprador de software sob encomenda, e freqüentemente recebia produtos de má qualidade. Em pouco tempo, esse modelo passou a ser utilizado como referência no mundo todo para medir a qualidade do processo de software em uma empresa.

Como explicado anteriormente, um dos princípios do CMM é o de que um processo de qualidade tende a gerar um resultado (software) de qualidade. O CMM classifica o processo em cinco níveis de maturidade: (1) inicial, (2) repetível, (3) definido, (4) gerenciado e (5) otimizado. Esses níveis são como "fases ou estágios através dos quais as empresas desenvolvedoras de software evoluem quando elas definem, implementam, medem, controlam e melhoram seus processos de software" (REZENDE, 1999, p.146). Cada nível de maturidade, exceto o primeiro, possui uma série de áreas-chave do processo, ou *Key Process Areas* (KPAs). "Cada área-chave do processo identifica uma série de atividades relacionadas que, quando realizadas em conjunto, atingem um grupo de objetivos considerados importantes para melhorar a capacidade do processo" (PAULK et al., 1993, p.30). Essas atividades são chamadas de *Key Practices*, ou práticas-chave. Assim, para uma empresa

atingir determinado nível de maturidade, ela deve realizar todas as práticas-chave de todas as áreas-chave daquele nível. A Figura 16 apresenta as áreas-chave de cada nível e a Figura 17 resume a estrutura de KPAs e práticas-chave explicadas acima.

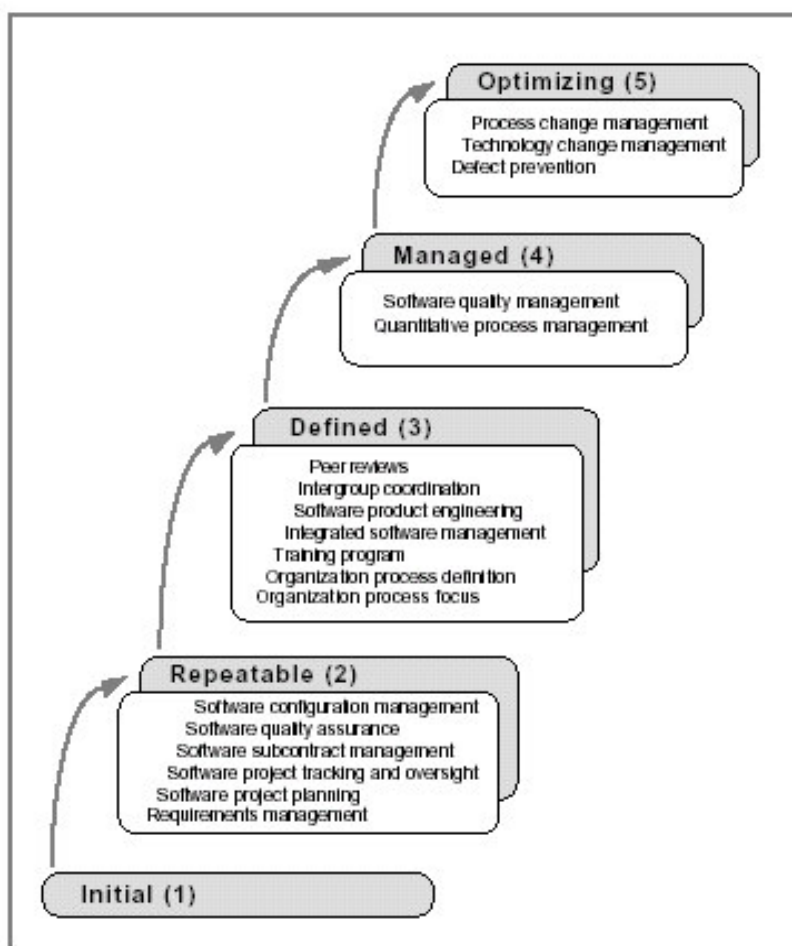


Figura 16: Áreas-chave do CMM agrupadas por nível (PAULK et al., 1993, p.31)

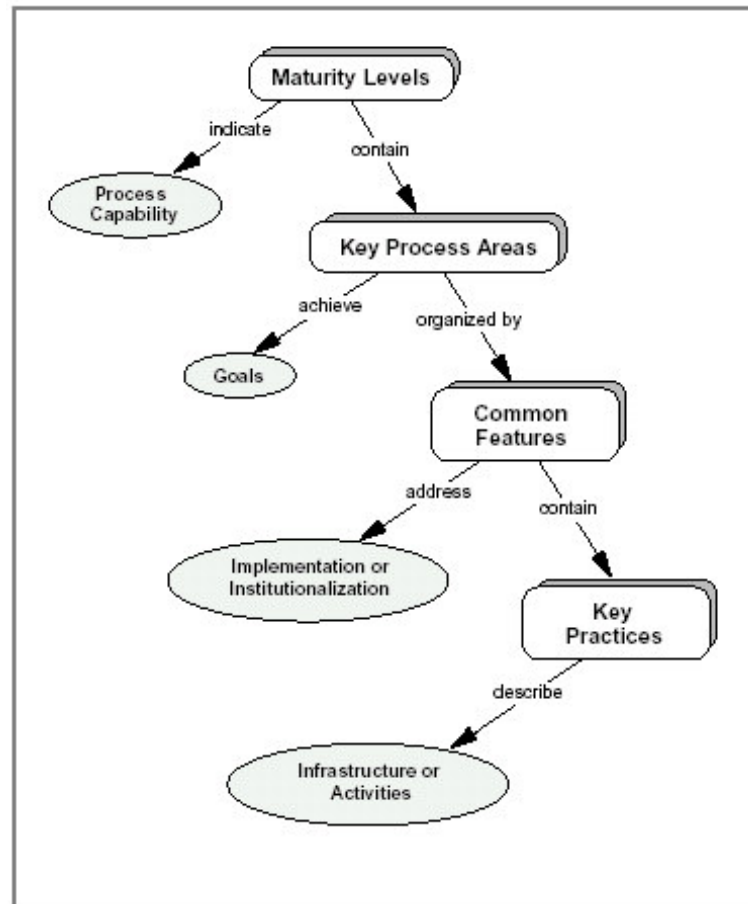


Figura 17: Estrutura do CMM
(PAULK et al., 1993, p. 29)

Desde 1993, a área de processo de software evoluiu bastante (inclusive graças à aplicação do CMM em uma grande quantidade de empresas) e foram surgindo muitas críticas a respeito desse modelo (como, ser ideal para grandes empresas mas não para empresas pequenas, ser muito rígido em relação à transição entre os níveis, e outras). Por causa disso, o SEI foi melhorando e aprimorando o CMM inicial, chegando ao atual CMMI (CMMI PRODUCT TEAM, 2002), que além de haver flexibilizado essa transição entre os níveis através da criação de um *modelo contínuo* (o CMM antigo - que continua sendo aceito - foi rebatizado de *modelo em estágios*), também expandiu a abrangência do CMM para outras áreas relacionadas direta ou indiretamente com o desenvolvimento de software (como engenharia de

sistemas, gerenciamento de recursos humanos e outros), criando toda uma família de CMMs. O *modelo contínuo* permite que uma empresa possua **níveis de capacidade** (muito semelhantes aos níveis de maturidade do *modelo em estágios*) diferentes em cada área-chave do processo. Uma consequência muito importante, e estrategicamente planejada pelo SEI, da existência do *modelo contínuo* é que ele permite ao CMMI ser compatível com o SPICE.

4.1.2. SPICE - Software Process Improvement and Capability dEtermination

Essa norma foi publicada pela ISO (*International Organization for Standardization*) e a IEC (*International Engineering Consortium*) em 1998, e foi revisada recentemente (em Junho de 2002) (INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2003). Também é conhecida pelo seu número: ISO 15504. O SPICE é um modelo para melhoria (*Improvement*) do Processo de Software e dEterminação da Capacidade desse processo de produzir software de qualidade (perceba que o conceito de capacidade é o mesmo do CMM).

A parte 2 do relatório Software Process Assessment (SPICE PROJECT ORGANISATION, 1995) apresenta um modelo de referência que "define, em alto nível, as atividades que são essenciais para uma boa engenharia do software [...]" (Idem, op. cit., p.10, tradução livre).

O SPICE inclui um **modelo de referência**, que serve como base para o processo de avaliação. Este modelo é um conjunto padronizado de processos fundamentais, que orientam para uma boa engenharia de software (REZENDE, 1999, p. 162).

O modelo de referência do SPICE especifica “categorias de processo”, que são sub-processos realizados dentro do processo de software.

Este modelo é dividido em **cinco grandes categorias de processo**: *Cliente-Fornecedor, Engenharia, Suporte, Gerência e Empresa*. Cada uma destas categorias é detalhada em processos mais específicos e descrito em detalhes pela norma (Idem, op.cit., loc.cit.).

Além das categorias de processos, o SPICE define também **seis níveis de capacitação** para cada processo, que podem ser: *não realizado, realizado informalmente, planejado e acompanhado, bem definido, quantitativamente controlado e continuamente melhorado*. "O resultado de uma avaliação, portanto, retrata um perfil da instituição em forma de matriz, onde os processos estão nas linhas e os níveis nas colunas" (Idem, op. cit., loc. cit.). A Figura 18 ilustra essa matriz.

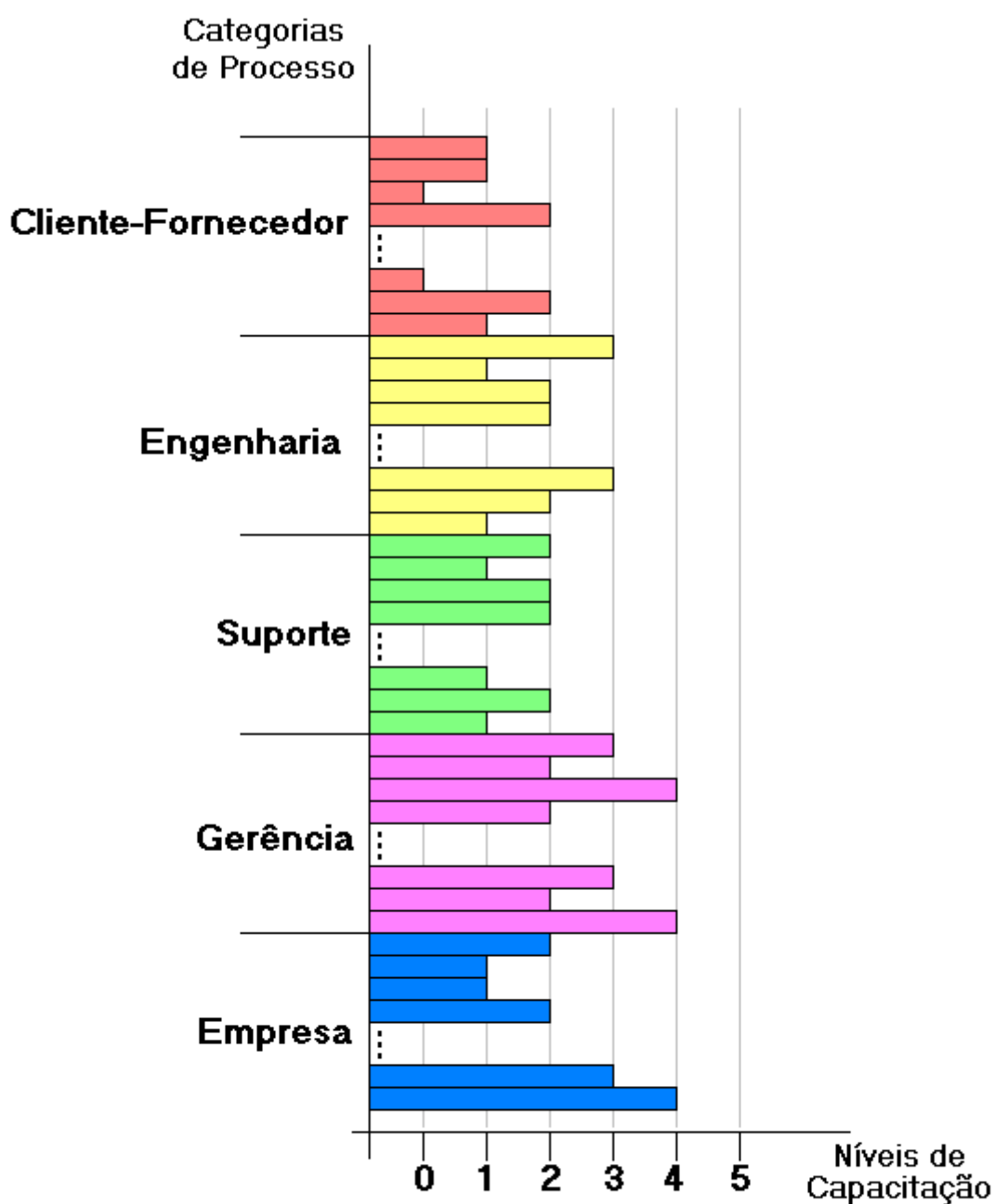


Figura 18: Exemplo de um perfil definido pelo SPICE

Para a melhora do processo, o SPICE utiliza a seguinte abordagem: com a capacidade do processo existente determinada através de uma matriz, são analisados os seus pontos fortes e fracos; na mesma matriz, são traçadas as capacidades desejadas para o processo (como se elas representassem um outro

processo existente); comparando os dois processos, pode-se ter uma idéia do que deve ser melhorado.

Fazendo uma analogia, os níveis de capacidade do CMM correspondem aos níveis de maturidade do SPICE e as KPAs do CMM podem ser agrupadas em conjuntos correspondentes às categorias de processos do SPICE.

Essa norma é uma tentativa da ISO de estabelecer um padrão para medição da qualidade do processo de software de uma empresa. Para facilitar a padronização, a parte 2 do relatório Software Process Assessment (SPICE PROJECT ORGANISATION, 1995) contém tabelas de mapeamento das normas ISO12207 e ISO9001 (nos anexos E e F, respectivamente) para os processos e categorias de processos do SPICE.

4.1.3 ISO 9000-3

É uma norma anterior ao SPICE, pois foi publicada em 1997. Faz parte do padrão para gerenciamento da qualidade e garantia de qualidade, o ISO 9000. A parte 3 do ISO 9000 (por isso o 9000-3) é um guia para a aplicação da ISO 9001 para o desenvolvimento, fornecimento, instalação e manutenção de software para computador (a ISO 9001, publicada em 1994, é uma norma genérica para garantia de qualidade no projeto, desenvolvimento, produção, instalação e fornecimento de serviços) (INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, 2003).

Por ser um guia, o ISO 9000-3 apenas explica o que deve ser feito para haver um processo de software de qualidade na empresa. No livro "ISO 9000-3: a tool for

software product and process improvement" (KEHOE e JARVIS, 1996), os autores dão algumas indicações do que fazer para concretizar as orientações do ISO 9000-

3. Segundo eles, a norma se concentra em três temas principais:

(1) Responsabilidades da gerência tanto do lado do fornecedor quanto do comprador.

"A gerência do fornecedor é responsável pela [...] institucionalização e o uso de um processo de engenharia" (Idem, op.cit., p.20, tradução livre). A gerência deve definir uma Política de Qualidade, um documento de alto nível que garanta o foco da organização no desenvolvimento de um produto de qualidade. As responsabilidades do comprador consistem em garantir que os requisitos foram claramente definidos, autorizar as mudanças desses requisitos e preparar e executar os testes de aceitação.

(2) O processo de desenvolvimento.

um processo de engenharia é constituído de várias fases, com entradas e saídas definidas para essas fases [...], existem papéis e responsabilidades definidas para os engenheiros que implementarão o processo e existem meios para testar os produtos e subprodutos desse processo de engenharia (KEHOE e JARVIS, 1996, p.20, tradução livre).

A necessidade de haver um processo de engenharia é uma consequência da Política de Qualidade. Além disso, o processo de engenharia deve ser gerenciado por planos que identificam cronograma, recursos e mecanismos de revisão e auditoria.

(3) As atividades de apoio.

existem atividades que apóiam os processos de desenvolvimento, testes e documentação da engenharia. Elas são: gerência de configuração [...], controle de documentos, medição da qualidade do produto e do processo e treinamento. Planos para a implementação dessas atividades devem ser desenvolvidos e recursos devem ser alocados para que esses planos possam ser executados (Idem, op. cit., loc. cit., tradução livre).

O documento que implementa a Política de Qualidade é chamado pela norma de Manual de Qualidade ou Processo de Qualidade, porém, como uma das críticas feitas pelos autores (Idem, op.cit., p. 21) à norma é o uso excessivo da palavra qualidade, eles recomendam que esse "manual de qualidade" seja chamado de Manual do Processo de Software (ou *Software Process Handbook*, SPH).

Os autores ainda recomendam, que para cada projeto de desenvolvimento e manutenção deve haver um SDP (*Software Development Plan*, ou Plano do Desenvolvimento de Software), identificando recursos, cronogramas e uma abordagem específica para implementar o processo de engenharia descrito no SPH. Além disso, o SDP deve "incluir ou referenciar planos subordinados (como o plano de gerência de configuração, os planos de testes, o plano de controle de documentos, entre outros)" (KEHOE e JARVIS, 1996, p.22, tradução livre).

Um dos pontos mais importantes da norma, é o de que a gerência é a principal responsável pela adoção de um processo de qualidade, é dela que deve partir o maior comprometimento com essa política de qualidade.

4.1.4 PSP e TSP

PSP é a abreviação de *Personal Software Process*, enquanto TSP é a abreviação para *Team Software Process* (SOFTWARE ENGINEERING INSTITUTE, 2003). Esses dois modelos foram também desenvolvidos pelo SEI. Após o desenvolvimento do CMM, muitos especialistas do SEI perceberam que as pessoas que iam adotar o CMM em uma empresa não sabiam como fazê-lo, pois o modelo não explica em detalhes o que fazer para mudar o processo existente. Para resolver esse problema, foram criados o PSP e o TSP.

O PSP é um modelo que detalha as tarefas de cada desenvolvedor em termos de medição e melhoria do seu *Processo Pessoal* de produção de *Software*, ou seja, ele enfoca tarefas como contagem de horas trabalhadas, estimativa de tempo para produção de um módulo ou componente, entre outras. Já o TSP, um modelo voltado ao *Processo de Software* da equipe (*Team*) de desenvolvimento, explica o que os líderes de uma equipe de desenvolvimento ou manutenção devem fazer para conseguir um bom processo de software. Ele enfoca mais a parte de coordenação e motivação da equipe de desenvolvimento e explica como a equipe pode ser auto-dirigida.

Os dois modelos auxiliam na introdução de uma disciplina de desenvolvimento de software, ou seja, na criação de bons hábitos de engenharia de software no dia-a-dia do trabalho dos desenvolvedores e seus chefes. Cabe também ressaltar que os dois modelos são complementares, e que o TSP só pode ser utilizado após a

implantação do PSP. A implantação desses dois modelos requer um treinamento grande de todos os envolvidos no processo de software em uma empresa, e por isso, ela é normalmente conduzida por um especialista do SEI ou um especialista autorizado.

A Figura 19 ilustra as principais áreas de processo que o PSP e o TSP abordam, e a Figura 20 mostra a relação dos dois modelos com o CMM.

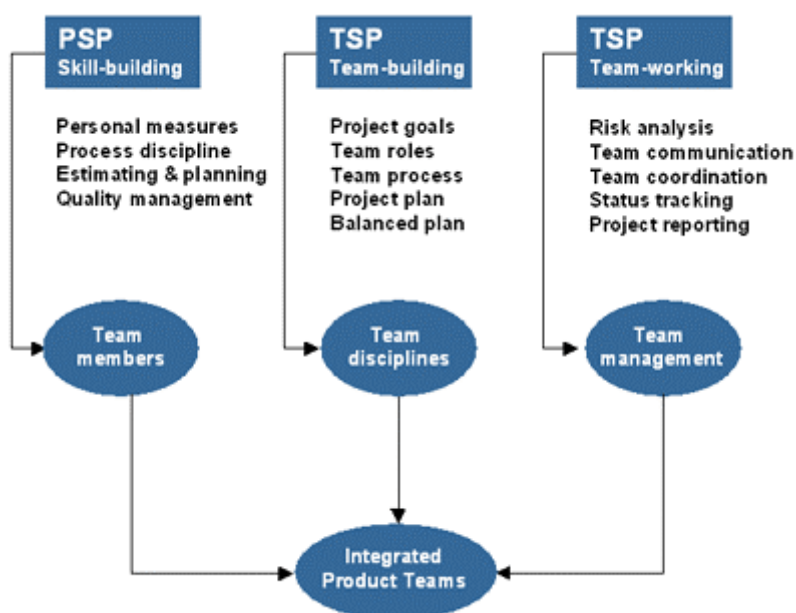


Figura 19: Áreas de processo abordadas pelo PSP e o TSP (SOFTWARE ENGINEERING INSTITUTE, 2003)

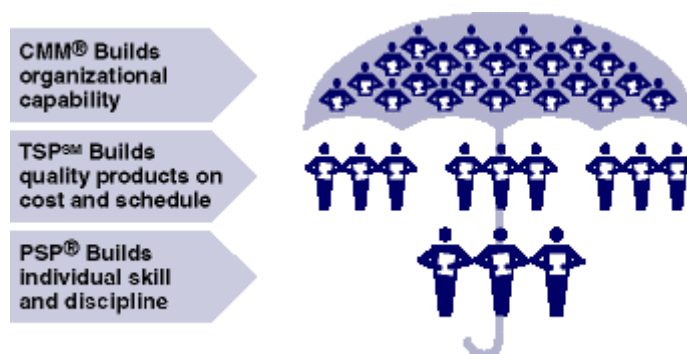


Figura 20: Relação do PSP e do TSP com o CMM (SOFTWARE ENGINEERING INSTITUTE, 2003)

O *site* referente ao TSP e PSP (SOFTWARE ENGINEERING INSTITUTE, 2003) também mostra que os resultados da aplicação dos dois modelos reduzem significativamente a ocorrência de defeitos, o tempo de duração dos testes, os desvios das estimativas do cronograma e do esforço empregado no desenvolvimento, entre outros. Sendo os dois modelos, preparatórios para a aplicação do CMM, pode-se perceber que (assim como o CMM) eles são voltados para grandes empresas, pois exigem dedicação de grande parte do tempo de desenvolvimento à medição e controle do processo, o que é proibitivo em empresas muito pequenas, como é o caso do NPI.

4.2 Questões Humanas na Modelagem de Processos

A seguir, são destacadas algumas idéias sobre modelagem de processo, que são recorrentes na literatura. Essas idéias estão relacionadas principalmente a fatores humanos que influem, ou que devem ser levados em conta, na modelagem de processos. É essencial considerar esses fatores, pois o sucesso de um modelo de processo depende tanto das suas características técnicas como das pessoas que irão segui-lo - afinal, para que serve um bom modelo se ele não é utilizado? Existem muitos motivos para que as pessoas sigam ou não um modelo, por exemplo, elas podem não gostar da aparência do modelo, achar que ele é muito complicado e não querer mudar o processo existente. Assim, são destacadas algumas questões

humanas na modelagem de processo, as quais percebeu-se que são consideradas importantes pela literatura.

Os modelos citados anteriormente apenas dão diretrizes do que fazer para se ter um processo de qualidade. Nenhum deles contém uma "fórmula mágica" que irá transformar uma empresa desordenada em uma empresa metódica. Por isso, os modelos devem ser sempre adaptados à realidade e às necessidades da empresa. "Focalize nos requisitos essenciais e personalize o seu processo para o tamanho da sua organização e a urgência da aplicação" (KEHOE e JARVIS, 1996, p. 25, tradução livre). "Um processo de engenharia de software deve ser personalizado de acordo com os cronogramas, orçamentos, metas de qualidade, e os pontos fortes e fracos inerentes em uma organização" (Idem, op. cit., p.13, tradução livre). Além disso, o modelo deve ser aplicado na empresa aos poucos, ou seja, não se deve tentar mudar o processo de uma empresa "da noite para o dia". Uma maneira de fazer isso é aplicar o modelo em um ou dois projetos, e ir progressivamente adotando-o nos outros projetos.

No espírito da personalização, deve-se tomar cuidado para não criar um modelo muito difícil de ser seguido, pois o objetivo da adoção de um modelo de processo de desenvolvimento deve ser sempre o de melhorar o processo da empresa, e não de piorá-lo. Por exemplo, ao adotar, em uma micro-empresa, um modelo que demanda um extenso e complexo sistema de qualidade, isso certamente irá atrapalhar o processo mais do que ajudar, pois os poucos funcionários da empresa gastarão muito tempo preenchendo formulários e escrevendo relatórios comparado com o tempo em que estarão efetivamente produzindo o software. Depois de alguns projetos, eles irão ficar desmotivados e o

modelo será progressivamente abandonado, prejudicando assim, a qualidade do software da empresa - o que era o objetivo inicial do modelo!

A mudança do processo de desenvolvimento requer principalmente uma mudança na maneira de pensar de todos os funcionários (desde a alta gerência até os programadores), isso é bastante enfatizado por vários autores (como, REZENDE, 1999 e KEHOE e JARVIS, 1996). WEINBERG (1993), explica que essa mudança implica em uma alteração no padrão cultural de pensamento das pessoas. Esse é um dos principais desafios da área de modelagem de processos.

Ainda em aspectos humanos, ROCHA (2002, p.12) afirma que "o processo de software deve estar documentado, ser compreendido e seguido". Ou seja, de nada adianta um bom manual de processo de software, se ele não é seguido. Nesse sentido, KEHOE e JARVIS (1996) apontam para a importância da gerência em comprometer-se com a qualidade do processo e ULRIKE, HAMANN e VERLAGE, (1997) explicam que um dos fatores que contribuem para o seguimento do modelo é a sua compreensibilidade: "Devido ao fato dos modelos serem utilizados por humanos e não por máquinas, eles devem ser elaborados levando-se em conta fatores como compreensibilidade e legibilidade, e devem apresentar a informação de uma maneira atraente" (Idem, op. cit., p.13, tradução livre).

Os autores também enfatizam a questão da aceitação e confiança (dos envolvidos no processo) nas pessoas que irão elaborar o modelo (tanto o descritivo quanto o prescritivo). Uma das maneiras de ganhar confiança é permitir e estimular a participação de todos os envolvidos no processo (desde a alta gerência até os desenvolvedores) nas atividades de elaboração do modelo.

É muito importante envolver todos os desenvolvedores desde o início [...] quando se faz a modelagem do processo da empresa. Se todas as pessoas envolvidas no processo participam da modelagem desde o princípio, eles sentem que as suas necessidades estão sendo levadas em conta, o que aumenta a sua aceitação. (ULRIKE, HAMANN e VERLAGE, 1997, p.7, tradução livre)

Além disso, é explicado que a elaboração do modelo deve ser abordada como uma tarefa construtiva e não uma tarefa prescritiva. Pois dessa maneira, os desenvolvedores sentirão que verdadeiramente fizeram parte do processo de elaboração e que o modelo pertence a eles.

É muito importante que os desenvolvedores se vejam como os donos dos modelos, apesar do *grupo de processo* [i.e., as pessoas que estão elaborando o processo] ser quem gerencia os modelos. [...] As pessoas sabem reconhecer quando um modelo apresentado em uma certa situação é apropriado ou não. (Idem, op.cit., p.12, tradução livre, grifo nosso)

Ainda, os autores comentam que um modelo de processo não deve ser tomado como algo rígido e imutável - devido à sua natureza não-determinística - e que as pessoas ficam mais satisfeitas com um processo que lhes dê liberdade para flexibilizá-lo em certos pontos.

Os modelos de processo, mesmo quando elaborados através de métodos rigorosos, não devem ser usados rigorosamente. Os processos de desenvolvimento de software são estocásticos e não determinísticos e os

desenvolvedores são pessoas inteligentes e criativas, que precisam de liberdade para interpretações pessoais. (ULRIKE, HAMANN e VERLAGE, 1997, p.13, tradução livre)

Portanto, após essa análise, pode-se perceber que é importante considerar as influências humanas na realização da modelagem de um processo. Alguns requisitos do NPI, apresentados no capítulo 2, lidam com questões dessa área (por exemplo, o terceiro requisito, apresentado na página 47) e o restante deste trabalho, como poderá ser visto nos capítulos a seguir, também considera alguns fatores humanos; assim, entende-se que esses fatores estão sendo levados em conta.

4.3 Relatos de Experiências

Nesta seção, são apresentados alguns relatos de experiências de modelagem de processos de desenvolvimento de software. Conforme explicado em capítulos anteriores, o NPI tem características parecidas com as de Micro e Pequenas Empresas (MPEs), assim, foi dada ênfase na pesquisa por relatos de experiências nesse tipo de empresa. Entretanto, como também já foi citado anteriormente, há uma grande carência na literatura em relação a MPEs.

O trabalho apresentado por WEBER (2002) relata a experiência do desenvolvimento de um modelo de processo de software para uma micro-empresa incubada no Centro GeNESS (Centro de Geração de Novos Empreendimentos em

Software e Serviços), localizado na Universidade Federal de Santa Catarina, em Florianópolis.

O modelo foi desenvolvido levando-se em conta as características da empresa (composta por três sócios-desenvolvedores) e do tipo de software que ela produzia (sistemas sob encomenda do tipo cliente-servidor para executar no ambiente WEB). Ele possibilita a escolha entre três tipos de modelos de ciclo de vida (ou adaptações deles) para cada projeto. O escopo total do modelo é apenas o processo técnico do desenvolvimento de software: análise de requisitos, projeto, implementação e testes.

O trabalho apresenta extratos do modelo, referentes a duas fases do processo: levantamento de requisitos e análise de requisitos. Cada fase possui um texto introdutório citando os objetivos, critérios de entrada e saída, descrição das atividades, produtos de saída, papéis envolvidos, métodos e técnicas utilizadas, ferramentas utilizadas e métricas importantes. Após o texto, é apresentado um modelo dos documentos daquela fase.

Na publicação do trabalho, o modelo ainda não havia sido utilizado em sua totalidade, mas a conclusão do autor é que o modelo o contribuiu significativamente para melhorar o processo da empresa.

Em seu artigo, LIMA (2002) relata a experiência de utilização de algumas técnicas e metodologias, consideradas atualmente como sendo estado da arte, por uma empresa de sistemas e automação localizada em Belo Horizonte. A empresa não é pequena, mas o artigo foi escolhido por relatar a utilização de alguns dos modelos de avaliação citados na primeira seção deste capítulo.

Utilizou-se "[...] o RUP [(*Rational Unified Process*)] como modelo de processo de desenvolvimento de software [e] o CMM como modelo para melhoria do processo

[...]", ao mesmo tempo em que se tentava atender às diretrizes de qualidade da ISO 9001 (a empresa já utilizava o RUP em alguns projetos e já havia obtido certificação ISO 9001 anteriormente). Para isso, foram implantadas algumas KPAs do nível 2 do CMM, foi definido e implantado um Processo Corporativo para Desenvolvimento de Software (contendo diretrizes para sua adaptação a projetos com características específicas), foi implantado um banco de métricas de projetos, e foi realizado um re-treinamento da equipe técnica e gerencial. Além disso, criou-se um Grupo de Garantia da Qualidade e um Grupo de Processos Engenharia de Software.

Para a implantação do novo processo, foi feita uma avaliação da situação atual da empresa, um planejamento da implantação do processo (em níveis organizacional e de projeto), a implantação em si, e finalmente, o acompanhamento e a avaliação dos resultados. A implantação teve as seguintes etapas: desenvolvimento de guias (ou manuais), treinamento e utilização em um projeto piloto.

Com isso, a empresa obteve um novo certificado ISO 9001; melhorou o seu controle de requisitos, de riscos, de mudanças e de qualidade; e o planejamento dos projetos pôde ser mais detalhado e realista.

O artigo apresentado por TAVARES, PAIM e CARVALHO (2002), apesar de relatar a experiência de modelagem de processo em uma grande empresa, é interessante pois algumas de suas lições aprendidas são válidas para organizações de todos os tamanhos. Os autores descrevem a estratégia da empresa para atingir o nível 2 do CMM, e as dificuldades e soluções encontradas durante esse processo. A estratégia descrita consiste em primeiro, conscientizar os membros da empresa de que um processo disciplinado é importante, depois treinar e implantar o processo, e

por último, validar esse processo através da garantia da qualidade. Algumas das conclusões dos autores que podem ser aproveitadas para organizações de todos os tamanhos são:

- * considerar e aproveitar a infra-estrutura já existente de hardware e software;
- * não se deve implantar cada área-chave separadamente, por que há uma forte dependência entre elas;
- * combater a tendência dos desenvolvedores de ir diretamente para a fase de projeto através do fortalecimento dos processos de requisitos;
- * a participação dos líderes de equipe e o apoio dos altos gerentes na implantação do processo garantem o seu rumo na direção certa;
- * meios de documentação dos resultados, como *sites de publicação*, melhoram a comunicação entre as equipes;
- * o envolvimento e aceitação do processo por parte do cliente melhora a aceitação do processo por parte das equipes da empresa;
- * os profissionais da empresa capacitados em instituições acadêmicas por meio de especializações ou pós-graduações voltadas à área de qualidade de software têm muito a contribuir para a empresa.

Outras conclusões, porém, não são aplicáveis a micro e pequenas empresas, por exemplo: "É imprescindível a alocação de recursos para formação de grupos de engenharia de software (SEPG) e de garantia de qualidade (GQS) [...]".

Uma conclusão interessante dos autores, resumida uma frase, merece ser destacada: "*quando o processo trabalha para as pessoas, as pessoas trabalham para o processo*" (TAVARES, PAIM e CARVALHO, 2002).

Infelizmente, como já foi explicado na introdução deste trabalho, nenhum dos artigos publicou o modelo de processo em si. Assim, não foi possível uma avaliação mais profunda dos modelos citados. Foi necessário basear-se apenas nos dados apresentados pelos autores dos artigos.

4.4 Discussão

Nesta seção são analisados os modelos de avaliação de processo e os relatos de experiências apresentados, em relação à expectativa de modelo para o NPI.

O Quadro 6 resume os modelos e experiências citados na seção anterior, comparando-os com os requisitos do NPI para um modelo de processo, os quais foram listados no capítulo 2.

Requisitos do NPI para um processo:	E1	E2	E3	CMM	SPICE	ISO 9000-3	TSP	PSP
1 - Possuir um modelo de processo para cada fase, integrado a atividades de gerência administrativa e a processos de gerência de configuração e controle de documentos.	±	↑	±	±	±	±	±	↓
2- Definir um roteiro de como realizar essas fases e oferecer documentos padrão para os documentos a serem produzidos.	↑	↑	?	↓	↓	↓	±	±
3 - Ser voltado a pessoas sem grande experiência e conhecimento na área de Engenharia de Software, e necessitar que elas passem por pouco tempo de treinamento para entender como realizar o processo.	↑	±	↓	↓	↓	↓	±	↑
4 - Ser baseado no processo existente, melhorando-o nos seus pontos mais fracos.	↓	↓	↓	↓	↓	↓	↓	↓
5 - Ser adequado ao contexto do NPI (escassez de pessoal e de tempo, orçamentos pequenos, e outras características específicas de uma empresa júnior).	↓	↓	↓	↓	↓	↓	↓	↓
6 - Facilitar e auxiliar o trabalho colaborativo, por intermédio ou não de computador.	?	?	↑	↓	↓	↓	↑	↓
7 - Possuir um meio, sistematizado e baseado em indicadores, para continuamente melhorar o próprio modelo.	±	↑	?	±	±	±	?	↑

Legenda:	
E1	Primeira experiência relatada (WEBER 2002)
E2	Segunda experiência relatada (LIMA, 2002)
E3	Terceira experiência relatada (TAVARES, PAIM e CARVALHO, 2002)
↑	Atende ao requisito
±	Atende parcialmente ao requisito
↓	Não atende ao requisito
?	Dado não disponível

Quadro 6: Comparação dos requisitos do NPI com outros modelos

Como pode ser visto no Quadro 6, nenhum dos modelos de avaliação nem os modelos apresentados nos relatos de experiência atendem a todos os requisitos do NPI para um modelo de processo de desenvolvimento de software.

Comparando os modelos, percebe-se que os relatados por WEBER (2002) e LIMA (2002) são os que atendem o maior número de requisitos do NPI. Porém, os dois requisitos principais do NPI (4 e 5) não são cumpridos por nenhum desses dois modelos - nem pelos outros.

Também é possível perceber, que não é fácil encontrar um modelo que seja voltado a pessoas com pouca experiência e com pouco tempo para receber treinamento.

Além disso, a maioria dos modelos separa as atividades administrativas e gerenciais das atividades de desenvolvimento, isso é possível em empresas com recursos humanos suficientes para constituir equipes separadas de desenvolvimento e de gerência; mas em micro e pequenas empresas (que são as que mais se assemelham com o NPI), devido ao número reduzido de funcionários, as pessoas que trabalham na gerência são as mesmas que realizam as atividades de desenvolvimento; assim, mais uma vez percebe-se por que os modelos apresentados no Quadro 6 não são completamente adequados ao NPI.

Os modelos de avaliação não oferecem documentos padrão nem definem um roteiro de como realizar as fases do desenvolvimento, justamente por que eles são modelos de avaliação e não de "realização". Já os modelos apresentados nas três experiências citadas, são modelos que serão utilizados para a *realização* de um processo e não somente para a sua avaliação - por isso, eles precisam definir documentos padrão e roteiros de como realizar as fases do desenvolvimento. Portanto, é esperado que os modelos de avaliação não satisfaçam o requisito 2. O TSP e o PSP definem um roteiro de como realizar as fases do desenvolvimento, porém, o PSP não define documentos padrões e para o TSP essa informação não estava disponível. Além disso, o TSP e o PSP são modelos inviáveis por que requerem a presença de um especialista da SEI e muito tempo de treinamento.

Da mesma maneira, os modelos de avaliação, devido à sua natureza, não contêm nenhuma metodologia para auxiliar o trabalho em grupo. Os relatos de experiências não apresentam os modelos de uma maneira suficientemente aprofundada para que se possa concluir sobre a sua influência no trabalho em grupo. O TSP, como o próprio nome já diz, foi criado para melhorar o trabalho em equipe, portanto, é natural que ele atenda completamente o requisito 6 do NPI. E o PSP, também já explicado no seu nome, não se propõe a facilitar o trabalho em equipe mas sim o trabalho individual de um desenvolvedor.

A maioria dos modelos de avaliação considera importante que o processo de desenvolvimento possua algum mecanismo (de preferência baseado em dados estatísticos) para que ele possa ser continuamente melhorado. Um exemplo é o CMM, onde as KPAs do nível 5 são dedicadas à mudança do processo. Porém, os modelos de avaliação não dizem exatamente como fazer essas mudanças - e é nesse ponto que eles não atendem ao requisito 7 do NPI.

Assim, conclui-se que não há nenhum modelo que satisfaça completamente todos os requisitos do NPI, e sim alguns modelos que satisfazem alguns requisitos e outros modelos que satisfazem outros requisitos. Por esse motivo é que se deve elaborar um modelo de processo unicamente para o NPI e não simplesmente adotar um modelo já existente.

5 O Modelo de Processo desenvolvido para o NPI

Neste capítulo é apresentado o modelo de processo desenvolvido para o NPI, o qual é o principal objetivo, e ao mesmo tempo resultado, deste trabalho. O modelo foi desenvolvido especificamente para o NPI, a partir das análises do processo existente (presentes no capítulo 2). Ele abrange todas as fases de um projeto típico de desenvolvimento de software do NPI (desde o primeiro contato com o cliente até a entrega da versão final do software). Além disso, ele contém documentos-padrão, que auxiliam e agilizam a realização do processo. A sua produção foi baseada inicialmente no modelo de ciclo de vida existente; a partir dele, foi concebido um novo modelo de ciclo de vida, e então, foi definido o conteúdo das fases. Também foram realizadas algumas avaliações informais de uma versão incompleta do modelo, pelos membros do NPI, durante a sua produção. Terminada a elaboração de todas as fases, o modelo foi apresentado informalmente aos diretores da empresa júnior. Ele agora faz parte dos documentos do NPI (na forma de um manual impresso) e é referenciado pelo nome de "Manual do Processo de Desenvolvimento de Software do NPI" (ou MPDS-NPI). O MPDS-NPI satisfaz a maioria dos requisitos do NPI para um processo, e comparando-o com os modelos estudados no capítulo 4, ele satisfaz um número maior de requisitos do que os aqueles modelos.

O MPDS-NPI foi elaborado a partir do processo de desenvolvimento existente, deduzido através da análise de documentos produzidos em dois projetos e da realização de entrevistas com os membros da empresa júnior. A primeira versão do

modelo de ciclo de vida do MPDS-NPI foi baseada no modelo de ciclo de vida existente (vide Figura 10, página 45). A partir desse novo modelo de ciclo de vida, foram detalhadas as fases e suas atividades. Durante esse refinamento, o modelo de ciclo de vida foi sendo alterado para se adequar ao paralelismo das fases. Também durante esse processo, foram realizadas avaliações informais (de uma versão preliminar do modelo de processo) pelos membros do NPI, igualmente contribuindo para a produção do modelo. Algumas fases não puderam ser baseadas no processo existente, pois elas eram raramente executadas. Nesse caso, foram buscadas metodologias na literatura, as quais foram adaptadas ao contexto do NPI. A parte de gerência e organização da informação (um dos pontos fracos do processo existente) é bastante sistematizada no MPDS-NPI. Além disso, o modelo de processo desenvolvido também considera questões humanas (algumas delas, mencionadas na seção 4.2 do capítulo 4), como conscientização da utilidade do modelo e imagem passada ao cliente. As características do modelo e a sua elaboração, são temas que serão aprofundados na seção 5.1 deste capítulo.

Em relação ao conteúdo, o modelo de processo elaborado abrange todas as fases de um projeto típico de desenvolvimento de software do NPI - desde o contato inicial do cliente até a entrega do software. Inicialmente, o MPDS-NPI apresenta um texto introdutório, seguido do modelo de ciclo de vida, e de um quadro-resumo de cada uma das fases: Proposta, Contrato, Requisitos, Projeto, Implementação, Testes e Entrega. Em seguida, cada fase é detalhada em dois níveis de refinamento: no primeiro, é apresentado um quadro com todas as características da fase (Objetivos, Atividades, Papéis e Responsabilidades, Critérios de Entrada e de Saída, entre outros); no segundo, essas Atividades - que são a parte mais importante da fase - são explicadas em detalhe (por exemplo, com instruções de como preencher os

documentos, de quais ferramentas utilizar e de como aplicar as metodologias sugeridas). Em ambos os níveis de refinamento, as atividades são sempre seguidas ou precedidas de informações sobre o seu paralelismo com atividades de outras fases (por exemplo "em paralelo a esta atividade, os Analistas também realizam a atividade 1 da fase de Testes"). Também fazem parte do MPDS-NPI, os documentos-padrão, que são documentos a serem preenchidos durante o desenvolvimento. É nesses documentos que são registradas as informações tanto técnicas como gerenciais do processo de desenvolvimento que está sendo executado (como por exemplo, os diagramas de UML, o esquema do banco de dados, o cronograma do projeto e o contrato com o cliente). Cada um desses documentos tem um nome e um código únicos, que os identificam, e pelos quais eles são referenciados em todo o texto do modelo. O conteúdo do MPDS-NPI é descrito com mais detalhes na seção 5.2 deste capítulo.

O modelo desenvolvido atende à maioria dos requisitos do NPI para um processo (descritos na seção 2.4 do capítulo 2, página 46): abrange todas as fases de desenvolvimento de software no NPI, possui um roteiro, é voltado a pessoas com pouca experiência e tempo disponível para treinamento, é baseado no processo existente e melhora os seus pontos mais fracos, é relativamente adequado ao contexto do NPI, auxilia um pouco o trabalho colaborativo e possui um meio para ser modificado. E também, em relação aos requisitos do NPI para um modelo de processo, o MPDS-NPI atende a um número maior de requisitos do que os modelos apresentados no capítulo 4. Na seção 5.3 deste capítulo, essa comparação é mais bem explicada: é apresentado o Quadro 9 (que a resume) seguido de uma explicação detalhada de como o MPDS-NPI atende a cada requisito e de como ele satisfaz certos requisitos que os outros modelos não satisfazem.

5.1 A elaboração do Modelo de Processo do NPI

Nesta seção, é dada uma breve explicação das características e da elaboração do modelo de processo desenvolvido para o NPI, o MPDS-NPI. Resumidamente, o modelo foi elaborado de acordo com os seguintes passos: analisar o processo existente; conceber um modelo de ciclo de vida a partir do modelo de ciclo de vida existente; para cada fase, definir suas atividades (e outras características explicadas mais adiante, como critérios de entrada e de saída, etc); e então, descrever detalhadamente as instruções de como realizar essas atividades. Uma das principais características do MPDS-NPI é a definição de políticas para a organização de documentos e para a gerência de pessoas. Essas e outras características serão apresentadas mais adiante. No próximo parágrafo, a explicação dos passos seguidos na produção do modelo é aprofundada.

Para a elaboração do modelo, foi inicialmente feito um estudo das características da empresa júnior (formada somente por alunos) e de seus principais problemas (alta rotatividade de pessoal e inexperiência deles em gerência de projetos e administração de empresas). Em seguida, foi realizada uma análise do processo existente no NPI. Para isso, foram feitas entrevistas com os membros do NPI envolvidos no processo de desenvolvimento da empresa júnior, e também foram analisados os documentos produzidos durante dois projetos, que estavam em finalização no período da elaboração deste trabalho. A partir dessa análise, chegou-se a um modelo de ciclo de vida, apresentado na Figura 10 (página 45) e também se

concluiu sobre as necessidades do NPI para um processo de desenvolvimento de software. Esse estudo é apresentado em detalhes no capítulo 2.

O modelo (MPDS-NPI) foi elaborado procurando-se atender aos requisitos do NPI para um processo (levantados no capítulo 2), principalmente o de "ser baseado no processo existente, melhorando-o nos pontos mais fracos" (vide página 47).

Então, inicialmente, foi concebido um modelo de ciclo de vida baseado no modelo de ciclo de vida existente. A partir desse novo modelo foram sendo detalhadas as fases (e suas atividades), uma de cada vez. Mas durante essa elaboração, percebeu-se um certo paralelismo entre algumas atividades de determinadas fases (por exemplo, enquanto estava sendo desenvolvido o projeto das classes da aplicação, poderia também ser realizado o projeto dos casos de teste). Isso implicava em uma mudança do modelo de ciclo de vida para que ele fosse ajustado a esse paralelismo (de acordo com o exemplo anterior, havia um paralelismo entre uma atividade de projeto e uma atividade de teste). Assim, a cada nova fase elaborada, o modelo de ciclo de vida era alterado. Portanto, o modelo de ciclo de vida foi evoluindo ao longo dessa elaboração. Na Figura 21, é apresentado o modelo de ciclo de vida em sua concepção final.

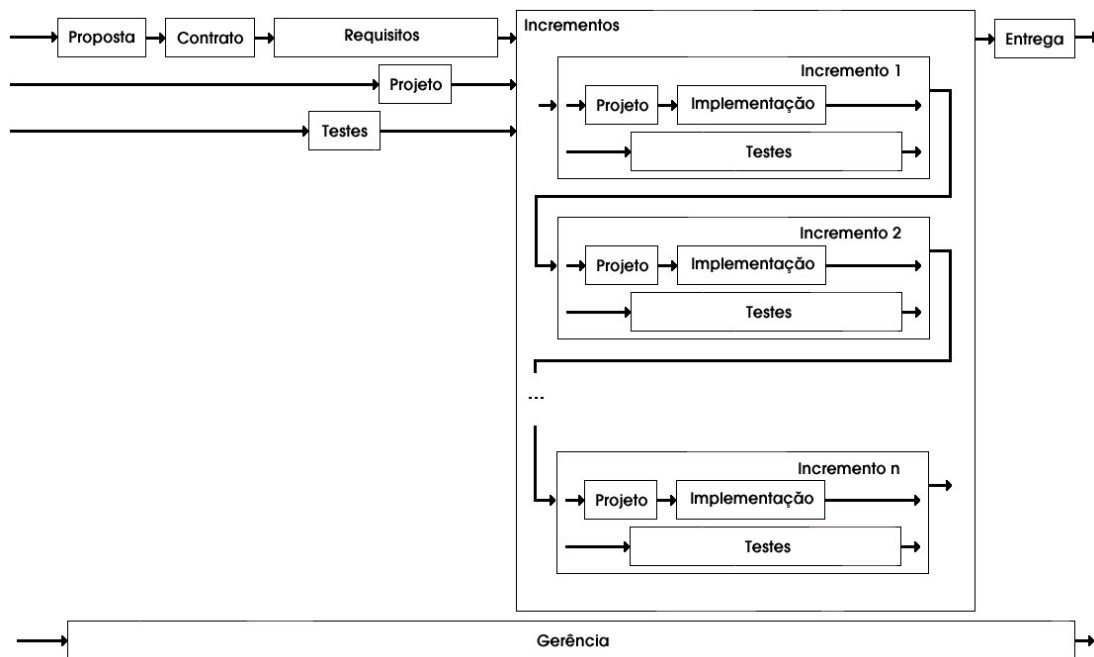


Figura 21: Modelo de Ciclo de Vida proposto

Além disso, durante a modelagem, foram feitas avaliações informais de uma versão preliminar do modelo de processo, pelos próprios membros do NPI. Durante essa avaliação surgiram conceitos como, por exemplo, as três opções de tipos de arquitetura em camadas - esse conceito está presente na fase de Projeto.

Ainda em relação ao processo existente, a fase de Entrega foi baseada no contrato padrão do NPI, que especifica um período de testes de aceitação.

Porém, certas fases não puderam ser baseadas no processo existente. Elas eram muito raramente realizadas, ou então realizadas de uma maneira muito informal. Ou seja, o processo existente apresentava falhas nessas fases ou atividades. Isso foi detectado principalmente nas fases de Testes e de Gerência. Para alguns desses casos, foi necessário buscar na literatura metodologias que se adequassem ao contexto do NPI (ou então adaptá-las), para serem utilizadas no processo. Entre as metodologias pesquisadas, destacam-se: o padrão de projeto de testes de casos de uso definido por BINDER (1999); o *framework* de testes sugerido

por BECK (1999); e a metodologia de desenvolvimento explicada em LARMAN (2000).

Durante a elaboração do MPDS-NPI, também foi dada bastante atenção à gerência e organização da informação: foi definido o Documento de Padrões, um documento especial contendo os padrões de codificação a serem utilizados na implementação e as ferramentas a serem utilizadas para cada função (ferramenta CASE, processador de texto, etc); foi definido um cabeçalho padrão para todos os documentos do processo; foram definidas políticas para o controle de versões, de alterações e de armazenamento dos documentos. Ainda para a fase de Gerência, também foram definidas políticas de acompanhamento e supervisão do trabalho e de acompanhamento do cronograma. Estas atividades eram bastante ausentes no processo existente, o que foi claramente constatado durante a análise dos documentos, dos dois projetos em andamento no período da elaboração deste trabalho.

Finalmente, o modelo de processo desenvolvido contém ainda, em algumas partes, considerações sobre fatores humanos em projetos de software, como por exemplo: sugestões do que dizer e de como se comportar diante do cliente; ênfase na responsabilidade do Diretor de Projetos, de conscientizar os Estagiários de que o MPDS-NPI não é uma imposição burocrática e sim um guia para a realização de um projeto de software bem-sucedido; conselhos sobre como selecionar o Gerente do projeto; entre outros. Alguns conceitos apresentados na seção 4.2 do capítulo 4 também foram incorporados ao modelo, de uma maneira mais sutil. Portanto, afirma-se com convicção que o modelo também possui uma ênfase - mesmo que pequena - em relações humanas.

5.2 O Conteúdo do MPDS-NPI

O modelo de processo desenvolvido para o NPI abrange todas as fases de um projeto típico de desenvolvimento de software na empresa júnior (desde o primeiro contato do cliente até a entrega da versão final do software), incluindo não só as atividades técnicas, como também as gerenciais. As atividades técnicas estão concentradas nas seguintes fases: Requisitos, Projeto, Implementação e Testes; e as atividades gerenciais são abordadas nas fases de Proposta, Contrato, Entrega e - principalmente - Gerência. O Quadro 7 descreve de maneira sucinta os objetivos de cada uma dessas fases.

Fase	Objetivo
Proposta	Realizar uma proposta de projeto para o cliente, contendo estimativas de tempo e custo. Negociar a proposta com o cliente.
Contrato	Assinar o contrato com a Empresa cliente. Assinar o contrato com os Estagiários.
Requisitos	Fazer a análise mais detalhada dos requisitos.
Projeto	Fazer o projeto do sistema de modo que ele possa ser mapeado diretamente em código.
Implementação	A partir do projeto, escrever o código.
Testes	Preparar os requisitos para o projeto dos casos de teste. Projetar e executar os casos testes
Entrega	Entregar o software ao cliente.
Gerência	Gerenciar o desenvolvimento do software.

Quadro 7: Resumo das fases do modelo

Esse quadro é apresentado na introdução do MPDS-NPI, juntamente com a Figura 21 (o modelo de ciclo de vida) e algumas outras explicações. Após a introdução, para cada fase, são apresentados dois níveis de refinamento: o primeiro é um quadro com as características da fase e o segundo é uma descrição detalhada de cada atividade daquela fase.

O quadro das características da fase possui os seguintes itens: Nome, Objetivos, Atividades, Entradas, Saídas, Papéis e Responsabilidades, Critérios de Entrada, Critérios de Saída. Cada uma dessas características é explicada no Quadro 8, cuja estrutura é o modelo do quadro das características.

Nome	<i>Nome da fase.</i>
Objetivos	<i>Objetivos gerais (e resumidos) da fase.</i>
Atividades	<i>Enumeração das atividades da fase e o seu paralelismo com outras atividades de outras fases.</i>
Entradas	<i>Documentos (produzidos em outras fases) que são necessários para que se possam iniciar as atividades desta fase.</i>
Saídas	<i>Documentos produzidos na fase.</i>
Papéis e Responsabilidades	<i>Listagem dos papéis que participam da fase e suas responsabilidades (i.e. as atividades que eles irão realizar).</i>
Critérios de Entrada	<i>Condições que devem ser atendidas para que se possa iniciar a fase (por exemplo, documentos que devem estar completos, ou informações que devem estar presentes em determinados documentos, etc).</i>

Critérios de Saída	<i>Condições necessárias para que se possa considerar a fase como terminada (muitas vezes, são simplesmente os critérios de entrada da fase seguinte).</i>
--------------------	--

Quadro 8: Explicação das características de uma fase

Para ilustrar ainda melhor essa forma de modelagem das fases, é apresentado o quadro das características da fase de Requisitos, no Quadro 9.

Nome	Requisitos
Objetivos	Fazer a análise mais detalhada dos requisitos.
Atividades	<p><i>(em paralelo a esta fase, o Gerente do Projeto realiza as atividades 6, 7, 8, 9, 10 e 11 da fase de Gerência; e o Diretor de Projetos realiza as atividades 12 e 13 dessa mesma fase)</i></p> <p>1 - Os Analistas fazem a reunião de análise dos requisitos com o cliente.</p> <p><i>(em paralelo a esta atividade, os Analistas também realizam a atividade 1 da fase de Testes)</i></p> <p>2 - Os Analistas levantam os Casos de Uso (DT-002) a partir das anotações da reunião da atividade 1 e do DT-001. E ao mesmo tempo, levantam as operações do sistema e (opcionalmente) elaboram o modelo conceitual (parte do DT-003).</p> <p>2.1 - Levantar os Casos de Uso.</p> <p>2.2 - Levantar as Operações de Sistema.</p> <p>2.3 - Elaborar o Modelo Conceitual. (opcional)</p> <p><i>(em paralelo a esta atividade, os Projetistas realizam a atividade 1 da fase de Projeto)</i></p> <p>3 - Os Analistas priorizam os casos de uso e dividem o sistema em incrementos.</p> <p>4 - Os Analistas fazem a revisão dos requisitos levantados, com o cliente. Nesta ocasião eles também</p>

	resolvem as dúvidas do Documento de Dúvidas (DG-006).
Entradas	- Documento de Requisitos (DT-001). - Documento de Pré-Planejamento do Projeto (DG-002).
Saídas	- Casos de Uso, DT-002 (contendo: casos de uso essenciais, sua priorização e alocação para os incrementos e operações do sistema). <i>(opcional)</i> - Diagramas da Análise, DT-003 (contendo o diagrama dos casos de uso, o modelo conceitual e os diagramas de seqüência).
Papéis e Responsabilidades	- Analista: * analisar os requisitos com o cliente; * documentar os requisitos (através de casos de uso essenciais, modelo conceitual (opcional) e operações do sistema); * apresentar e explicar os requisitos ao cliente para a sua aprovação.
CrITÉRIOS de Entrada	- Contrato NPI-Empresa (DG-004) e Contrato NPI-Estagiaros (DG-005) assinados.
CrITÉRIOS de Saída	- DT-002 e DT-003 (caso se opte por produzir algum(ns) de seus diagramas) completos e aprovados pelo cliente.

Quadro 9: Fase de Requisitos

Além do quadro das características (exemplificado no Quadro 9), a modelagem de uma fase inclui também a descrição detalhada de todas as suas atividades. Nessa descrição constam informações como: o que deve ser realizado, qual ferramenta deve ser utilizada, quais técnicas ou metodologias devem ser adotadas, quais documentos serão produzidos, se esses documentos estarão prontos nesta

atividade ou só serão completados em outra atividade posterior, exemplos de como preencher os documentos, entre outras.

A título de exemplo, é apresentado no Quadro 10, o detalhamento da atividade 1 da fase de Requisitos.

1 - Os Analistas fazem a reunião de análise dos requisitos com o cliente.

É importante explicar ao cliente que essa reunião tem como objetivo o NPI entender o que o cliente quer que o sistema faça, e que esse entendimento depende de quão bem os requisitos são analisados.

Essa reunião deve ser realizada com os futuros usuários do sistema, ou seja, os atuais usuários do sistema existente.

Nesta atividade deve-se estudar o sistema (informatizado ou não) existente, **tendo como objetivo descobrir os casos de uso**, utilizando as seguintes técnicas (SOUZA e SILVA, 2003):

- ler os documentos utilizados (caso sejam relevantes) e coletá-los ou tirar cópias (caso isso seja possível);
- observar o ambiente de trabalho (i.e. as atividades realizadas pelos usuários);
- realizar algumas entrevistas informais.

Durante as entrevistas e as observações, devem ser realizadas anotações por melhor que seja a memória.

Para sistemas pequenos, uma reunião de três horas deve ser suficiente. Para sistemas médios, podem ser necessárias até uma ou duas reuniões a mais.

Deve ser informado ao cliente, que após a formalização dos requisitos, será feita uma nova reunião para que ele aprove os requisitos, em forma de Casos de Uso e Interfaces Gráficas com o Usuário, ou GUIs (*Graphical User Interfaces*).

Quadro 10: Detalhamento da atividade 1 da fase de Requisitos

Ainda sobre as atividades de cada fase, em ambos os níveis de refinamento (tanto no quadro de características como na explicação detalhada), constam informações sobre o paralelismo ou seqüência das atividades: antes de uma atividade, é explicado se existe outra atividade com a qual ela deve ser paralelamente executada ou se existe uma outra atividade que deve ser executada

antes dela; após uma atividade, é explicado se deve ser realizada alguma atividade de outra fase depois dela. O Quadro 11 mostra essas três situações.

<p>.... <i>(em paralelo a esta atividade, os Projetistas realizam a atividade 1 da fase de Projeto)</i> 3 - Os Analistas priorizam os casos de uso e dividem o sistema em incrementos. 1 - Os Programadores preparam o instalador do software e elaboram o Manual de Instalação. <i>(após essa atividade, o Gerente do Projeto realiza a atividade 15 da fase de Gerência)</i> <i>(realizada após a atividade 6 da fase de Entrega)</i> 17 - O Gerente do Projeto e os outros Estagiários elaboram o Documento de Fim de Projeto (DG-013). ...</p>
--

Quadro 11: Exemplos de informações sobre o paralelismo das atividades

Após a apresentação das fases nos dois níveis de detalhamento, o MPDS-NPI apresenta ainda, uma lista de sugestões de melhorias a serem feitas no modelo e também algumas referências bibliográficas, tanto das metodologias utilizadas no modelo como de leituras que podem auxiliar a sua compreensão.

Além do texto do modelo - cuja parte principal é a descrição das fases (caracterizada pelo quadro das características e pelo detalhamento das atividades) - o MPDS-NPI também inclui os documentos-padrão, que são documentos a serem preenchidos durante as atividades. Eles são listados como entradas e saídas de cada fase, por exemplo, na fase de Requisitos, as saídas são os Casos de Uso (DT-002) e os Diagramas da Análise (DT-003). Nos documentos-padrão, constam as informações (tanto técnicas como gerenciais) resultantes do processo, por exemplo, a listagem das seqüências dos casos de uso, os diagramas de projeto, o esquema do banco de dados, o cronograma do projeto, a linguagem de programação

escolhida para aquele projeto, entre outras. Todos os documentos-padrão possuem um nome e um código únicos, pelos quais eles são referenciados em todo o texto do modelo (o código do documento Diagramas da Análise, por exemplo, é DT-003). DT significa que o documento é um Documento Técnico e DG um Documento Gerencial, o número que segue essa sigla foi dado na ordem em que os documentos foram sendo criados. Essas siglas foram utilizadas de acordo com a definição do conceito de documento estabelecida no capítulo 3 (página 62). O Quadro 12 mostra todos os nomes dos documentos-padrão e seus respectivos códigos.

Código	Nome
DG-001	Ficha de Pré-Projeto
DG-002	Documento de Pré-Planejamento do Projeto
DG-003	Proposta de Projeto
DG-004	Contrato NPI-Empresa
DG-005	Contrato NPI-Estagiários
DG-006	Documento de Dúvidas
DG-007	Documento de Aceitação
DG-008	Formulário de Erros
DG-009	Documento de Padrões
DG-010	Documento de Acompanhamento do Trabalho
DG-011	Relatório de Progresso do Trabalho
DG-012	Documento de Alteração da Data de Entrega
DG-013	Documento de Fim do Projeto
DG-014	Documento de Mudanças do Modelo
DG-015	Plano do Projeto
DT-001	Documento de Requisitos
DT-002	Casos de Uso
DT-003	Diagramas da Análise
DT-004	Diagramas de Projeto
DT-005	Esquema do Banco de Dados
DT-006	Script do Banco de Dados
DT-007	Decisões de Projeto
DT-008	Documento de Testes
-	Código-fonte do programa
-	Código-fonte dos casos de testes
-	Sistema de Ajuda
-	Manual de Instalação
-	Software executável

Quadro 12: Documentos-padrão e seus códigos

Todos esses documentos padrão possuem instruções no modelo de como eles devem ser preenchidos. Além disso, o modelo também define em qual atividade a primeira versão do documento está pronta, para cada documento. Desse modo, a numeração das versões pode ser feita de maneira precisa e sistematizada.

Os documentos também possuem um cabeçalho padrão, para facilitar a sua identificação. Esse cabeçalho contém número e nome do projeto, nome do cliente, versão do documento, fase e número do incremento onde ele foi produzido e a data em que ele foi produzido. Algumas partes desse cabeçalho são as mesmas para todo o projeto, mas outras são diferentes para cada documento. O Quadro 13 mostra o modelo do cabeçalho-padrão, com as explicações de como preenchê-lo. Esse modelo de cabeçalho (seguido de algumas explicações) está presente no MPDS-NPI na sua parte introdutória.

Código do Documento - Nome do Documento	
Número do Projeto:	No formato XXX/AAAA. Para cada projeto que se iniciar na fase Proposta - mesmo que depois ele não seja realizado - dar um número em seqüência (isso permitirá uma estimativa de "projetos realizados" / "projetos propostos"). A cada ano deve-se reiniciar a contagem. Projetos diferentes para um mesmo cliente devem ter numeração diferente.
Nome do Projeto:	Escolher um nome para o projeto. Caso se decida mudar o nome do projeto, deve-se lembrar de mudar esse campo em todos os artefatos.
Cliente:	Colocar aqui o nome do cliente.
Versão:	Numerar as versões em seqüência: 1, 2, 3, etc. (mais informações na atividade 4 da fase de Gerência)
Fase e número do incremento:	A fase, e se for aplicável, também o número do incremento ao qual o documento pertence.
Data:	Data em que foi terminada a produção ou modificação do documento. Ao mudar a versão, deve-se mudar a data também.

Quadro 13: Modelo do cabeçalho-padrão

Para exemplificar o preenchimento de um cabeçalho, o Quadro 14 mostra um exemplo de cabeçalho para a Proposta de Projeto. Esse exemplo também está presente no texto introdutório do MPDS-NPI.

DG-003 - Proposta de Projeto	
Número do Projeto:	002/2003 (2º projeto de 2003)
Nome do Projeto:	Projeto Nortia II
Cliente:	Nortia Consultores
Versão:	4
Fase e número do incremento:	Proposta
Data:	30 / 05 / 2003

Quadro 14: Exemplo de cabeçalho preenchido

Cada documento possui um conteúdo diferente, de acordo com o seu propósito. A maioria dos documentos possui quadros a serem preenchidos ou perguntas a serem respondidas. Isso permite um preenchimento rápido do documento e um melhor esclarecimento das informações que nele devem constar. Para tornar um pouco mais claro o conteúdo e a forma dos documentos, é exibido como exemplo o Documento de Requisitos no Quadro 15.



NÚCLEO DE PROJETOS EM INFORMÁTICA

Documento de Requisitos

DT-001 - Documento de Requisitos	
Número do Projeto:	___/20__
Nome do Projeto:	
Cliente:	
Versão:	
Fase e número do incremento:	Proposta
Data:	___ / ___ / 20__

1. Requisitos

ID	Descrição
R.1	O sistema deve ...
R.2	O sistema deve ...
...	
R.n	O sistema deve ...

2. Glossário (opcional)

Palavra	Significado
ononono	ononononono
...	...

Quadro 15: Um exemplo de documento padrão

Alguns documentos, possuem instruções de como eles devem ser preenchidos e exemplos de como preenchê-los (para outros documentos, todas as instruções de como preenche-los estão presentes no próprio texto do MPDS-NPI). No exemplo do Quadro 15, o "(opcional)" em cinza é uma instrução que indica que o Documento de Requisitos não precisa necessariamente possuir um glossário.

Pelo fato de que o modelo (MPDS-NPI) na íntegra é muito extenso, ele não será apresentado aqui. Porém, ele está disponível como um anexo deste trabalho em meio digital (disquete) e também no NPI (NÚCLEO DE PROJETOS EM INFORMÁTICA, 2003).

5.3 Discussão

Após a apresentação do modelo (suas características e seu conteúdo), bem como os métodos utilizados para a sua elaboração, pode-se inferir sobre os seus benefícios para o NPI.

O modelo possui muitas qualidades que foram caracterizadas como importantes para o NPI (por exemplo, ter sido produzido especialmente para essa empresa júnior, ser voltado a pessoas com pouca experiência, entre outras). Por possuir todas essas características (que os modelos avaliados no capítulo 4 não possuem), pode-se concluir que o modelo é adequado ao NPI.

Entre as principais vantagens do MPDS-NPI em relação aos outros modelos estudados no capítulo 4, pode-se citar:

* abrange tanto atividades técnicas como gerenciais;

* possui um roteiro de como realizar cada uma das fases do processo de desenvolvimento;

* é voltado a pessoas com pouca experiência e tempo disponível para treinamento;

* é baseado no processo existente e melhora os seus pontos mais fracos;

* é mais adequado ao contexto do NPI.

Comparando o modelo desenvolvido com as necessidades do NPI (os requisitos listados no final do capítulo 2) e os modelos de avaliação e modelos apresentados nos relatos de experiências (ambos citados no capítulo 4), pode-se concluir que o MPDS-NPI atende à maioria dos requisitos e ainda, que ele atende a um número maior de requisitos do que os modelos estudados no capítulo 4.

O Quadro 16 apresenta essa comparação com um pouco mais de detalhes (as características do modelo desenvolvido estão destacadas em vermelho). E, em seguida, ele é discutido minuciosamente.

Requisitos do NPI para um processo:	E1	E2	E3	CMM	SPICE	ISO 9000-3	TSP	PSP	MPDS-NPI
1 - Atividades técnicas e gerenciais	±	↑	±	±	±	±	±	↓	↑
2- Existência de um roteiro e documentos-padrão	↑	↑	?	↓	↓	↓	±	±	↑
3 - Pouca necessidade de experiência	↑	±	↓	↓	↓	↓	±	↑	↑
4 - Baseado no processo existente	↓	↓	↓	↓	↓	↓	↓	↓	↑
5 - Adequado ao contexto do NPI	↓	↓	↓	↓	↓	↓	↓	↓	↑
6 - Facilitar o trabalho colaborativo	?	?	↑	↓	↓	↓	↑	↓	±
7 - Sistemáticamente modificável	±	↑	?	±	±	±	?	↑	±

Legenda:	
E1	Primeira experiência relatada (WEBER 2002)
E2	Segunda experiência relatada (LIMA, 2002)
E3	Terceira experiência relatada (TAVARES, PAIM e CARVALHO, 2002)
MPDS-NPI	O modelo desenvolvido neste trabalho
↑	Atende ao requisito
±	Atende parcialmente ao requisito
↓	Não atende ao requisito
?	Dado não disponível

Quadro 16: Comparação dos requisitos do NPI com os modelos do capítulo 4 e o modelo desenvolvido

Observando o Quadro 16, pode-se perceber que a maioria dos requisitos planejada inicialmente para o modelo foi cumprida.

A fase de Gerência possui atividades de gerência administrativa, gerência de configuração e processos de controle de documentos. Essas atividades não são muito complexas, elas são o mínimo necessário para que se possa considerar que existe uma gerência administrativa, de configuração e de documentos.

Como já foi explicado, o modelo define uma série de documentos-padrão que devem ser preenchidos ao longo do desenvolvimento.

O modelo é voltado para os alunos dos cursos de Ciências da Computação e Sistemas de Informação da UFSC, a única exigência do modelo é que os alunos já tenham cursado a disciplina de Engenharia de Software e/ou Análise e Projeto de Sistemas. Portanto, o modelo é claramente voltado para pessoas com pouco conhecimento e experiência em desenvolvimento sistematizado de software.

Também, como já foi explicado anteriormente, o modelo foi inteiramente baseado no processo existente e melhorou-o nos seus pontos mais fracos.

O modelo é adequado ao contexto do NPI, dentro do possível. Acredita-se que esse requisito poderá ser ainda melhor atendido em versões posteriores do modelo.

Quanto ao trabalho colaborativo, o modelo é um facilitador no sentido de que ele define uma política de alteração de documentos já existentes e armazenamento dos documentos recém produzidos (o Gerente do Projeto centraliza essas atividades). Porém, ao mesmo tempo, o modelo não é tão facilitador quanto desejado pelos membros do NPI, pois tudo depende do Gerente e da sua disponibilidade para enviar e receber os documentos.

Para a modificação do modelo, não há uma metodologia ou política baseada em indicadores como foi idealizada. Há apenas uma política um tanto simples, mas que já é bem melhor do que não haver política nenhuma: as modificações no modelo devem ser justificadas por escrito e devem ser armazenadas juntamente com a versão anterior do modelo.

Assim, pode-se concluir que o modelo desenvolvido é bastante satisfatório em relação aos requisitos que haviam sido propostos inicialmente.

Em comparação aos modelos citados no capítulo 4, os requisitos 4 e 5, que não foram atendidos por nenhum dos modelos, são bem atendidos pelo modelo desenvolvido.

Outra característica exclusiva do modelo produzido é que ele é voltado a pessoas com pouca experiência em desenvolvimento sistematizado de software. Como explicado no capítulo anterior, nenhum dos modelos analisados satisfazia a esse requisito.

O modelo apresentado neste trabalho, apesar de também separar as atividades administrativas das atividades de desenvolvimento, é adequado ao contexto do NPI porque essas duas atividades são realizadas por pessoas da mesma equipe, pessoas que estão trabalhando juntas diariamente (e não pessoas

de departamentos diferentes, que se encontram somente em reuniões mensais, ou no máximo, semanais).

Em relação aos requisitos 6 e 7, um dos objetivos futuros do modelo de processo desenvolvido é adaptar ao contexto do NPI algumas das características presentes nos modelos de avaliação (em termos do requisito 7) e no TSP (em termos do requisito 6), de uma maneira semelhante à explicada no parágrafo acima.

Assim, após apresentação e discussão de todos os benefícios que o modelo produzido neste trabalho traz para a empresa júnior, conclui-se que ele é adequado ao NPI.

6 Avaliação do Modelo

Com o objetivo de avaliar a aplicabilidade do modelo e os seus pontos fortes e fracos no NPI, é apresentado o plano para a avaliação do modelo.

Esse plano será executado em paralelo à utilização do modelo pelo NPI em um projeto piloto. Como no decorrer deste trabalho, não se iniciou nenhum novo projeto na empresa júnior, a execução do plano de avaliação não foi possível.

Por esse motivo, o presente capítulo apresentará principalmente o plano de avaliação, e abordará brevemente a sua execução.

6.1 O Plano da Avaliação

O plano da avaliação foi realizado com base na metodologia GQM (*Goal Question Metric* - Meta Pergunta Medida). Essa metodologia, como o próprio nome já diz, é orientada a metas, e serve para mensuração tanto de produtos como de processos de software. "Esta abordagem segue a definição de medidas relevantes top-down via metas, perguntas e modelos e a análise e interpretação bottom-up dos dados coletados [...]"(ANACLETO, 2001, p.18)

Ela define um modelo de processo, isto é, uma série de passos a serem seguidos, para a realização das medições e da avaliação dos resultados, são eles (WANGENHEIM, 1999):

- * estudo prévio da organização (incluindo a elaboração de um modelo de processo descritivo, caso seja necessário);
- * identificação das Metas GQM (ou seja, os objetivos do estudo);
- * desenvolvimento do Plano GQM (descrevendo a meta GQM em detalhes);
- * desenvolvimento do Plano de Mensuração (contendo informações como: quem irá realizar as medições, quando elas serão realizadas, etc);
- * coleta de dados;
- * análise e interpretação dos dados (incluindo reuniões de *feedback*);
- * e captura de experiências (por exemplo, definição de padrões para a organização).

O objetivo geral da avaliação é medir a contribuição do MPDS-NPI para um processo de desenvolvimento de software mais produtivo e de melhor qualidade. A partir desse objetivo e da descrição do processo existente no NPI (apresentado no capítulo 2), foram definidas as metas GQM: qualidade, duração e gerência dos projetos.

Essas metas foram então refinadas num documento que o GQM chama de *Abstraction Sheet* (folha de abstração), que consiste em um quadro com as seguintes informações: fatores de qualidade (*o quê* será medido), fatores de variação (elementos que influenciam nos fatores de qualidade), hipótese de linha base antes e depois (como o processo estava antes da aplicação do MPDS-NPI e como se supõe que ele estará após a sua aplicação) e impacto na hipótese de linha-base (*como* os fatores de variação exercem a sua influência). Além disso, na parte

superior do quadro, deve estar presente a identificação da meta GQM, a qual inclui: o objeto (sendo avaliado), o objetivo (da avaliação), o enfoque da qualidade (i.e. as metas), o ponto de vista (ou seja, *quem* utilizará os dados coletados) e o contexto (em qual empresa e/ou projeto está sendo feita a avaliação).

Foram elaboradas duas *abstraction sheets*, uma para as metas Qualidade e Duração (pois ambas estão relacionadas) e outra para a meta Gerência. Para cada item da primeira *abstraction sheet*, foram separados os dados que pertencem à meta Qualidade e os que pertencem à meta Duração. O Quadro 17 apresenta um extrato dessa primeira *abstraction sheet* (para tornar o exemplo mais claro, foram mostrados somente os dados da meta Qualidade). As duas *abstraction sheets* produzidas encontram-se completas no Apêndice B.

Objeto	Objetivo	Enfoque da Qualidade	Ponto de Vista	Contexto
Modelo de Processo	Avaliar	Qualidade e Duração	Pesquisador	NPI
Fatores de Qualidade			Fatores de Variação	
<i>Qualidade:</i> - Consistência entre requisitos e projeto - Consistência entre projeto e implementação - Sistematização da produção dos documentos ...			<i>Qualidade:</i> - Comprometimento e conscientização da equipe de desenvolvimento da necessidade de utilização de técnicas de engenharia de software - Experiência da equipe em desenvolvimento de software ...	
Hipótese de Linha-Base			Impacto na Hipótese de Linha-Base	
Antes		Depois (Hipótese)		
<i>Qualidade:</i> - O nível de consistência entre os requisitos e o projeto é Baixo. - O nível de consistência entre o projeto e a implementação é Baixo. - Não há uma ordem para a produção dos documentos. ...		<i>Qualidade:</i> - O nível de consistência entre os requisitos e o projeto é Alto. - O nível de consistência entre o projeto e a implementação é Alto. - Há um modelo de processo que determina a ordem de produção dos documentos e essa ordem é seguida. ...	<i>Qualidade:</i> - ↑ experiência em desenvolvimento de software: ↑ consistência entre requisitos e projeto ↑ consistência entre projeto e implementação - ↑ comprometimento e conscientização da necessidade de utilizar técnicas de engenharia de software ↑ consistência entre requisitos e projeto ↑ consistência entre projeto e implementação ↑ produção sistematizada de documentos ...	

Quadro 17: *Abstraction Sheet* para a meta Qualidade

O plano GQM, resumido nas *abstraction sheets*, foi então detalhado em forma de perguntas. Essas perguntas também são definidas no processo GQM. A apresentação do plano GQM em forma de perguntas auxilia a sua compreensão. As perguntas mostram diretamente como serão medidos: os fatores de qualidade, os fatores de variação, e a influência dos fatores de variação nos fatores de qualidade.

Para ilustrar essa explicação, no Quadro 18 são apresentadas algumas das perguntas GQM para as metas Qualidade e Duração.

Definição das Variações

Q1) A experiência da equipe em desenvolvimento de software tem um impacto na consistência entre os requisitos e o projeto?

Hipótese: quanto mais experiência a equipe possui em desenvolvimento de software, maior será a consistência entre os requisitos e o projeto.

Q1.1) Qual a experiência da equipe em desenvolvimento de software?

Modelo: a experiência da equipe deve ser a média aritmética das experiências de todos os membros da equipe.

M1.1 experiência da equipe [ordinal:

"Grande (trabalhou na área mais de 4 anos)";

"Média (trabalhou na área entre 2 e 4 anos)";

"Pequena (trabalhou na área menos de 2 anos)"].

Hipóteses Antes e Depois: a equipe possui uma experiência Média.

Q2) A experiência da equipe em desenvolvimento de software tem um impacto na consistência entre o projeto e a implementação?

Hipótese: quanto mais experiência a equipe possui em desenvolvimento de software, maior será a consistência entre o projeto e a implementação.

...

Definição da Qualidade

...

Q10) Qual o grau de sistematização da produção dos documentos?

Modelo: esse grau pode ser medido pela existência ou não de um processo sistematizado para a produção e por sua utilização ou não.

M10.1 existência de um processo sistematizado [ordinal:

"Existe um modelo de processo";

"Existe uma ordem documentada informalmente";

"Existe uma ordem não-documentada";

"Não existe uma ordem"].

Hipótese Antes: não existe uma ordem para a produção dos documentos.

Hipótese Depois: há um modelo de processo que determina a ordem de produção dos documentos.

M10.2 utilização da ordem [ordinal: é seguida; não é seguida].

Hipótese Antes: como não existe ordem, ela não é seguida.

Hipótese Depois: a ordem é seguida.

...

Quadro 18: Exemplos de perguntas do Plano GQM

As perguntas dividem-se em dois tipos: definição das variações (que são as perguntas sobre os fatores de variação) e definição da qualidade (que são as perguntas sobre os fatores de qualidade).

Para a definição das variações, a primeira pergunta (Qn) busca confirmar (ou negar) o impacto na hipótese de linha-base definido na *abstraction sheet* - esse impacto é repetido abaixo da pergunta (em "Hipótese:"). Assim, a pergunta Q1 busca confirmar a hipótese de que a experiência da equipe influencia na consistência entre os requisitos e o projeto. Para tornar a medição mais precisa, as sub-perguntas (Qn.m) quantificam o fator de variação (segundo o exemplo, a sub-pergunta Q1.1 quantifica a experiência da equipe). Na definição da qualidade, a primeira pergunta (Qn) já quantifica o fator de qualidade.

O "Modelo:" define como o fator de variação (ou de qualidade) deve ser calculado (ainda para o exemplo de Q1.1, o "Modelo:" define que a experiência da equipe deve ser calculada realizando-se a média aritmética da experiência de cada membro da equipe). Finalmente, tem-se a medida (Mn.m) do fator de variação (ou de qualidade), que pode ser do tipo ordinal, racional, inteiro, intervalo, entre outros (mais informações em WANGENHEIM, 1999, p.16), e cujos valores possíveis são colocados após o ":". Como o plano GQM desenvolvido avaliará um processo, o tipo da maioria das medidas é ordinal. Para cada medida, também são repetidas as hipóteses antes e depois - definidas na parte "Hipótese de Linha-Base" da *abstraction sheet*. Para a definição da qualidade, todas as perguntas derivam uma ou mais medidas, enquanto que, para a definição das variações, algumas perguntas podem não derivar medida nenhuma.

A partir das medidas definidas nesse plano GQM, foi elaborado o Plano de Mensuração. Ele contém informações sobre quando coletar as medidas, quem irá coletá-las e como elas serão coletadas. Em relação à informação de "como coletar", o plano de mensuração também inclui roteiros para as entrevistas, nas quais os

dados serão coletados. O plano de mensuração desenvolvido é apresentado no Quadro 19.

Número	Medidas Associadas	Descrição	Tempo	Papel	Instrumento
1	M1.1, M3.1, M3.2	relação da equipe com a ES	após a atividade 5 da fase de Contrato	Pesquisador	Entrevista com a Equipe de Desenvolvimento
2	M6.1, M7.1	relação da equipe com estimativas	após a atividade 4 da fase de Proposta	Pesquisador	Entrevista com os Diretores
3	M8.1, M9.1, M9.2, M9.3, M9.4	consistência entre requisitos, projeto e implementação	após o fim do projeto	Pesquisador	Inspeção dos Documentos
4	M10.1, M10.2	sistematização da produção dos documentos	após o fim do projeto	Pesquisador	Entrevista com o Gerente
5	M11.1, M11.2	atraso do projeto	após o fim do projeto	Pesquisador	Inspeção dos Documentos
6	M12.1	dedicação do gerente ao projeto	após o fim do projeto	Pesquisador	Entrevista com o Gerente
7	M15.1	duração estimada do projeto	após a atividade 4 da fase de Proposta	Pesquisador	Entrevista com os Diretores
8	M17.1, M17.2, M17.3	nível de controle das versões dos documentos	após o fim do projeto	Pesquisador	Entrevista com o Gerente
9	M18.1, M18.2, M18.3	nível de planejamento e controle do cronograma	após o fim do projeto	Pesquisador	Entrevista com o Gerente
10	M19.1, M19.2	nível do acompanhamento do trabalho	após o fim do projeto	Pesquisador	Entrevista com o Gerente

Quadro 19: Plano de Mensuração

Para cada grupo de medidas associadas, foi planejado um instrumento de coleta de dados. A maioria dos instrumentos planejados, são entrevistas, mas há também uma inspeção de documentos, que, sendo realizada pelo pesquisador, é o modo mais viável de coletar esses dados. Para cada tipo de entrevista, foi definido

um roteiro de como realizá-la. Para exemplificar, um desses roteiros é apresentado no Quadro 20.

Roteiro para a entrevista com a Equipe de Desenvolvimento (Gerente e os outros Estagiários)

Essas perguntas devem ser realizadas a todos os Estagiários ao mesmo tempo. Eles devem entrar em consenso (dentro do possível) e escolher uma das respostas.

1 - (M1.1) Qual a experiência de vocês em desenvolvimento de software? (fazer uma média aritmética das experiências de cada Estagiário)

- a) Grande (já trabalharam na área por mais de 4 anos)
- b) Média (já trabalharam na área entre 2 e 4 anos)
- c) Pequena (já trabalharam na área por menos de 2 anos)

2 - (M3.1) Qual a importância que vocês vêem na utilização de técnicas de ES para o desenvolvimento de software no NPI?

- a) é importante
- b) não é importante

3 - (M3.2) Vocês continuariam utilizando a técnica mesmo diante de acontecimentos críticos (como atraso no cronograma) ?

- a) sim
- b) não

Quadro 20: Um dos roteiros para entrevista

Facilitando ainda mais a coleta, o número de cada medida associada com a pergunta da entrevista, está entre parênteses antes da pergunta.

Também foi definido um roteiro para a inspeção dos documentos, com questões a serem respondidas durante a inspeção.

Devido à extensão do plano de avaliação (desde as *abstraction sheets* até os roteiros das entrevistas), ele não é apresentado integralmente nesta seção. Mas, para melhorar a sua compreensão, ele encontra-se completo no Apêndice B.

6.2 Execução da Avaliação

Durante o período de elaboração deste trabalho, não foi possível a aplicação do modelo no NPI, pois não surgiram novos projetos (e se um projeto, não há como utilizar o modelo). Porém, com o plano de avaliação completamente definido, esse estudo da avaliação do modelo é previsto para ser realizado no futuro (quando se iniciar um novo projeto na empresa júnior).

7 Conclusão

O principal resultado deste trabalho, é um modelo de processo de desenvolvimento de software para o NPI. Também são importantes o plano de avaliação do modelo e o levantamento do processo existente.

Para obtenção desses resultados, inicialmente foi realizada uma pesquisa bibliográfica, voltada para a área de modelagem de processo, e também, foi analisado o processo de desenvolvimento de software existente no NPI - essa análise foi o primeiro resultado deste trabalho. A partir disso, foi elaborado o modelo de processo de desenvolvimento do NPI - que foi o segundo, e mais importante, resultado obtido neste trabalho. Em seguida, foi também elaborado um plano de avaliação desse modelo - o terceiro resultado -, que só não foi executado por que não se iniciaram projetos no NPI nesse período.

Entretanto, mesmo não tendo a oportunidade de avaliar o modelo desenvolvido num projeto piloto no NPI - no decorrer deste trabalho - analisando (em teoria) os requisitos do NPI em relação ao modelo desenvolvido, pode-se observar que:

- * o modelo desenvolvido possui descrições detalhadas de como realizar um projeto - característica declarada nos primeiros dois requisitos (página 47 do capítulo 2).

- * ele define políticas e documentos que auxiliam a comunicação, coordenação e organização das atividades e dos resultados das fases - exigência dos dois primeiros e dos dois últimos requisitos (páginas 47 e 48);

* ele é voltado para as necessidades (por exemplo, utilização de software grátis) e características (como desenvolvedores e gerentes pouco experientes) específicas do NPI - solicitações do terceiro, quarto e quinto requisitos (páginas 47 e 48).

O capítulo 5 apresenta com mais detalhes essa comparação dos requisitos do NPI para o processo com o modelo desenvolvido.

Assim, pode-se concluir que, dentro do possível, os resultados do trabalho satisfazem os seus objetivos iniciais.

7.1 Benefícios para o NPI

A maior vantagem deste trabalho é que o NPI agora conta com um modelo de processo, ou seja, um guia de como proceder nos projetos de desenvolvimento de software. Isso pode ser de grande valia, pois (como já foi explicado anteriormente) a empresa júnior possui uma rotatividade altíssima de funcionários, e agora, os novos diretores, gerentes e desenvolvedores não estarão sem rumo quando forem realizar o seu primeiro projeto de desenvolvimento de software no NPI.

Portanto, se o modelo for seguido corretamente, espera-se que o NPI:

- * tenha um processo (e como consequência, um software) de maior qualidade;
- * tenha uma visão clara e definida do seu processo de desenvolvimento;
- * tenha um maior controle e planejamento gerencial de seus projetos;
- * desperte em seus membros a conscientização e o comprometimento com as técnicas de Engenharia de Software;

* consiga preservar as experiências adquiridas nos projetos para serem utilizadas por futuras gestões;

Assim, pode-se perceber que o modelo resultante deste trabalho tem grande potencial para contribuir para a melhoria das condições do desenvolvimento do software no NPI.

7.2 Trabalhos Futuros

Como trabalhos futuros, inicialmente, aponta-se a utilização do modelo em paralelo com a execução do plano de avaliação. Com base nos resultados dessa avaliação, recomenda-se a melhora do modelo, principalmente as fases de Testes e de Gerência, que contêm atividades que até agora raramente foram realizadas no NPI (e que também são muito pouco enfatizadas atualmente nas disciplinas da área nos cursos de Ciências da Computação e Sistemas de Informação da UFSC) - e por isso, foi necessário buscar na literatura metodologias de como efetuar-las -, as quais precisam ser utilizadas, avaliadas e melhor adaptadas ao seu contexto.

Outras possibilidades de futuros trabalhos incluem o desenvolvimento de uma ferramenta de software para tornar mais prática a utilização do modelo e para facilitar a sua alteração. Adicionalmente, também é importante a elaboração de um modelo de processo de manutenção de software para o NPI.

7.3 Considerações Finais

Este trabalho foi a primeira iniciativa para sistematizar o processo de desenvolvimento de software no NPI. Assim, o modelo que foi elaborado está apenas em sua primeira versão. Dessa forma, não se espera que ele seja totalmente perfeito e adaptado ao NPI. Ele deve evoluir ao longo da sua utilização cotidiana. O modelo deve ser aplicado em um ou mais projetos para que se possa avaliar os seus pontos fortes e fracos, e então ele deve ser melhorado; em seguida, esse novo modelo deve ser aplicado em mais alguns projetos, e então, novamente ele deve ser avaliado e melhorado. Considerando uma melhoria contínua, esse ciclo nunca terminará (e o modelo nunca estará totalmente pronto), pois uma organização - e portanto, o NPI - está sempre mudando. O presente trabalho, é assim, o primeiro passo em uma longa caminhada.

8 Bibliografia

ANACLETO, Alessandra. **Modelo de Mensuração para Gerência de Projetos em Microempresas de Software**. 2001. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Universidade Federal de Santa Catarina, Florianópolis, 2001.

BECK, Kent. **Simple Smalltalk Testing: With Patterns**. 1999. Disponível em: <<http://xprogramming.com/testfram.htm>>. Acesso em: 16 out. 2003.

BINDER, Robert V. **Extended Use Case Test Design Patterns**. 1999. Disponível em: <www.rbsc.com/docs/TestPatternXUC.pdf>. Acesso em: 14 out. 2003.

DENGER, Christian; MORA, Maricel Medina. **Test case derived from Requirement Specifications**. abr. 2003. IESE-Report 033.03/E. Disponível em: <http://www.iese.fhg.de/Publications/iese_reports_1/>. Acesso em: 14 out. 2003.

CMMI PRODUCT TEAM. **Capability Maturity Model Integration for Software Engineering, Version 1.1, Continuous Representation**. ago 2002. CMU/SEI-2002-TR-028. Disponível em: <<http://www.sei.cmu.edu/pub/documents/02.reports/pdf/02tr028.pdf>>. Acesso em: 15 jun 2003.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION. **ISO - International Organization for Standardization**. Disponível em: <<http://www.iso.org>>. Acesso em: 13 jun. 2003.

KEHOE, Raymond; JARVIS Alka. **ISO 9000-3: a tool for software product and process improvement**. Nova York: Springer, 1996.

KELLNER, Marc I. et al. **Process Guides: effective guidance for process participantes**. ago. 1998. IESE-Report 012.98/E. Disponível em: <http://www.iese.fhg.de/Publications/iese_reports/>. Acesso em: 28 jun. 2003.

LARMAN, Craig. **Utilizando UML e Padrões**: uma introdução à análise e ao projeto orientados a objetos. Tradução de Luiz Augusto Meirelles Salgado. Porto Alegre: Bookman, 2000.

LIMA, Rodrigo Duran. Gerência de Projetos de Software com RUP, CMM e ISO 9001. In: CONGRESSO INTERNACIONAL DE TECNOLOGIA DE SOFTWARE, 13., 2002, Curitiba. **Anais...** Curitiba: CITS, 2002. p.107-115.

NÚCLEO DE PROJETOS EM INFORMÁTICA. **Estatuto**. Florianópolis, 1996. Universidade Federal de Santa Catarina.

NÚCLEO DE PROJETOS EM INFORMÁTICA. **NPI - Núcleo de Projetos em Informática**. 2002. Universidade Federal de Santa Catarina. Disponível em: <<http://www.npi.inf.ufsc.br>>. Acesso em: 21 mar. 2003.

NÚCLEO DE PROJETOS EM INFORMÁTICA. **Manual do Processo de Desenvolvimento de Software do NPI**. Florianópolis, 2003. Universidade Federal de Santa Catarina.

PAULK, Mark C. et al. **Capability Maturity Model for Software, Version 1.1**. fev. 1993. CMU/SEI-93-TR-024. Disponível em: <<http://www.sei.cmu.edu/pub/documents/93.reports/pdf/tr24.93.pdf>>. Acesso em: 10 dez. 2002.

PRESSMAN, Roger S. **Engenharia de Software**. 5.ed. Rio de Janeiro: McGraw-Hill, 2002.

REZENDE, Denis Alcides. **Engenharia de Software e Sistemas de Informações**. Rio de Janeiro: Brasport, 1999.

ROCHA, Ana Regina. **Qualidade de Software**: processo e produto. In: ENCONTRO DA QUALIDADE E PRODUTIVIDADE EM SOFTWARE, 2002, Petrópolis. Disponível em <[http://www.mct.gov.br/Temas/info/Dsi/PBQP/Reuniao Petropolis/Palestra COPPE.pdf](http://www.mct.gov.br/Temas/info/Dsi/PBQP/Reuniao_Petropolis/Palestra_COPPE.pdf)>. Acesso em: 29 jun. 2003.

SAVI, Rafael. **Desenvolvimento de um Sistema para Planejamento de Projetos de Software em Micro e Pequenas Empresas**. 2002. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Universidade Federal de Santa Catarina, Florianópolis, 2002.

SILBERSCHATZ, Abraham; KORTH, Henry F.; SUDARSHAN S. **Sistema de Banco de Dados**. 3. ed. São Paulo: Makron Books, 1999.

SILVA, Ricardo Pereira e. **Apostila de Engenharia de Software** - Semestre 2002.2. 598 transparências. Florianópolis, 2002.

SOFTWARE ENGINEERING INSTITUTE. **Building High Performance Teams Using TSP and PSP**. 2003. Carnegie Mellon University. Disponível em: <<http://www.sei.cmu.edu/tsp/>>. Acesso em: 16 jun 2003.

SOUZA, Maíra Bay de; SILVA, Rodrigo Grumiche. **Engenharia de Requisitos**. 2003. Trabalho escrito como requisito parcial para aprovação na disciplina Engenharia de Software, Universidade Federal de Santa Catarina, Florianópolis, 2003.

SPIICE PROJECT ORGANISATION. **Software Process Assessment**. jun. 1995. Technical Report Type 2. Disponível em: <<http://www.sqi.gu.edu.au/spice/suite/download.html>>. Acesso em: 29 jun. 2003.

SUN MICROSYSTEMS. **Code Conventions for the Java Programming Language**. 1999. Disponível em: <<http://java.sun.com/docs/codeconv/>>. Acesso em: 22 set. 2003.

SUTTON JR., Stanley M. The role of process in a software start-up. **IEEE Software**, p. 33-39, jul./ago. 2002.

TAVARES, Helena Cristina; PAIM, Fábio Rilston Silva; CARVALHO, Ana Elizabete. Implantando CMM Nível 2: A Estratégia SERPRO. In: SIMPÓSIO BRASILEIRO DE QUALIDADE DE SOFTWARE, 1., 2002, Gramado. **Anais...** 1 CD-ROM.

ULRIKE, Becker; HAMANN, Dirk; VERLAGE, Martin. **Descriptive Modeling of Software Process**. dez. 1997. IESE-Resport 045.97/E. Disponível em: <http://www.iese.fhg.de/Publications/lese_reports/>. Acesso em: 28 jun. 2003.

WANGENHEIM, Christiane Gresse won. **Melhoramento de Software Baseado em Mensuração: Como Aplicar GQM na Prática?** Florianópolis: GeNESS, 1999. Relatório Técnico GeNESS 001.99P.

WEBER, Sérgio. **Um estudo de caso em uma micro empresa de software para o desenvolvimento de um modelo de processo de software**. 2002. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Universidade Federal de Santa Catarina, Florianópolis, 2002.

WEINBERG, Gerald M. **Software com Qualidade**: pensando e idealizando sistemas. Tradução de Flávio Deney Steffen. Revisão Técnica: Silvio Carmo Palmeri. São Paulo: Makron Books, 1993.

WQS WORKSHOP DE QUALIDADE DE SOFTWARE, 1994, Curitiba. **Anais do VIII Simpósio Brasileiro de Engenharia de Software**. Curitiba, 1994.

Apêndice A - Artigo

Modelo de Processo de Software: aplicação em uma empresa júnior

Maíra Bay de Souza
maira@inf.ufsc.br, Universidade Federal de Santa Catarina

Resumo

O NPI (Núcleo de Projetos em Informática - a empresa júnior de Ciências da Computação e Sistemas de Informação da UFSC) possui algumas características de micro e pequenas empresas e outras bem particulares, as quais levam à ocorrência de processos e produtos de baixa qualidade. Baseando-se na literatura, acredita-se que a utilização de um modelo de processo de desenvolvimento de software irá contribuir para a melhoria da qualidade do software desenvolvido no NPI, colaborando para a solução dos problemas detectados. O modelo apresentado neste trabalho foi produzido com base no processo existente no NPI e nos requisitos dessa empresa júnior para um processo ideal. Ele abrange todas as fases de um projeto típico de desenvolvimento de software no NPI, contendo tanto atividades técnicas como gerenciais. O modelo será aplicado na empresa júnior e então melhorado, como parte de um processo que será realizado continuamente pelos próprios membros da empresa júnior.

1 Introdução

Este artigo relata o estudo realizado em 2003 como trabalho de conclusão de curso da aluna Maíra Bay de Souza do curso de Ciências da Computação da Universidade Federal de Santa Catarina. O trabalho completo encontra-se em SOUZA (2004).

1.1 Empresa Júnior

Uma empresa júnior tem como um dos seus objetivos principais realizar a integração entre os estudantes e o mercado de trabalho. Ela funciona como empresa normal, mas legalmente é uma organização sem fins lucrativos, devendo vender seus produtos a preço de custo. Ela deve ser totalmente comandada por estudantes, tanto na parte administrativa quanto na parte operacional. O NPI (Núcleo de Projetos em Informática) é a empresa júnior composta pelos alunos dos cursos de Ciências da Computação e Sistemas de Informação da Universidade Federal de Santa Catarina. Os alunos podem atuar no NPI como Estagiários ou como Diretores. Os Diretores realizam todas as tarefas administrativas da empresa, como contabilidade, marketing, gerência de recursos humanos, e coordenação de projetos. Os Estagiários realizam as atividades operacionais da empresa júnior, como desenvolvimento de software.

2 Os Problemas do NPI

O NPI realiza projetos de todos os tipos na área da informática, mas a sua principal atividade é o desenvolvimento de software comercial sob encomenda. Assim, o início de um projeto típico do NPI consiste nas seguintes etapas: um cliente entra em contato com a empresa júnior solicitando um produto; os Diretores selecionam Estagiários para realizar o desenvolvimento; assina-se um contrato entre o NPI e o cliente, e um contrato entre o NPI e os Estagiários; inicia-se o desenvolvimento. Portanto, a cada projeto são selecionados Estagiários diferentes, eles não fazem parte do quadro permanente da empresa júnior. Os Diretores, por sua vez, devem permanecer nos seus cargos pelo período de um ano. Essas regras (definidas no estatuto da empresa júnior, NÚCLEO DE PROJETOS EM INFORMÁTICA, 1996), implicam em uma altíssima rotatividade de desenvolvedores e de Diretores. Essa é uma característica bastante peculiar do NPI (ou melhor, das empresas juniores de um modo geral).

Além dessa dificuldade, o NPI também apresenta mais dois problemas que são relevantes. Por ser uma empresa júnior formada por alunos, os Diretores e desenvolvedores possuem muito pouca experiência em desenvolvimento sistematizado de software (principalmente por que a disciplina de Engenharia de Software só é ministrada na metade do curso); além disso, os Diretores (por serem alunos de informática) também têm pouquíssima experiência em administração de empresas. Essas características também estão bastante presentes em micro e pequenas empresas.

Um outro fator agravante é que, quando os alunos adquirem experiência nessas áreas (administração de empresas e desenvolvimento sistematizado de software), eles geralmente estão se formando e portanto, saindo do NPI. Assim, a experiência adquirida durante os projetos de desenvolvimento não permanece na empresa júnior.

3 O Processo Existente

O NPI não possui um processo **claramente definido** para o seu desenvolvimento de software. Assim, para entender o processo existente, foi efetuada a análise de dois projetos em andamento. Foram realizadas entrevistas informais com o Diretor de Projetos e os desenvolvedores dos dois projetos, e foram examinados os artefatos produzidos durante o desenvolvimento.

Após a investigação, concluiu-se que o processo do NPI constitui basicamente de três fases: análise de requisitos, modelagem do sistema e implementação.

Na análise de requisitos, o Diretor de Projetos se reúne com o cliente, levanta os requisitos, formaliza-os e divide-os em módulos. Na fase de modelagem do sistema, são feitas a análise e o projeto do sistema, em paralelo, sem uma distinção clara entre essas duas atividades. Após essas duas fases, é realizado um "incremento" para cada módulo, contendo a fase de implementação e a realização de testes informais. Após essas três fases o software é entregue ao cliente.

O processo é facilitado por alguns documentos-padrão que já existiam na empresa, como a Proposta de Projeto e o Contrato de Prestação de Serviços de Desenvolvimento de Software. Esses documentos possuem um conteúdo genérico, que pode ser adaptado a cada projeto.

Em todas as fases houve um acúmulo de papéis, principalmente para o Diretor de Projetos, que atuou como gerente dos dois projetos. Além disso, em ambos os projetos, foram iniciados muitos documentos de gerência do projeto (como cronograma, documento de acompanhamento do trabalho, etc) - e também alguns de análise e projeto de sistemas - mas eles foram abandonados no meio do projeto por falta de tempo para terminá-los. Um problema muito grave detectado é que o desenvolvimento do segundo projeto já havia iniciado e ainda não havia sido assinado um contrato com o cliente.

Assim, o processo de desenvolvimento do NPI é bastante aleatório e nebuloso. Porém, deve ser destacado que há um esforço (tanto do Diretor de Projetos como dos desenvolvedores) de sistematizá-lo de alguma forma.

4 A Abordagem Proposta para a Solução do Problema

Para auxiliar na resolução dos problemas do NPI explicados anteriormente, foi escolhida a abordagem da área de modelagem de processo (de desenvolvimento de software).

4.1 O Processo e a sua Qualidade

O desenvolvimento de um software pode ser feito utilizando-se várias metodologias, inclusive a de um processo *ad-hoc*. É uma idéia bastante aceita na literatura de que a qualidade de um software está fortemente associada com o nível de organização do processo utilizado para desenvolvê-lo. "A qualidade de um produto de software é fortemente dependente da **qualidade do processo** pelo qual ele é construído e mantido" (ROCHA, 2002, p.9, grifo nosso). Ou seja, a utilização de um processo metódico e sistematizado tem muito mais chances de produzir um software de qualidade do que a utilização de um processo caótico e desordenado.

O conceito de qualidade do processo também é consenso na literatura, sendo definido como a capacidade de produzir sistemas que vão ao encontro dos requisitos do cliente dentro dos prazos e custos estabelecidos no início do projeto (é esse conceito de qualidade que os certificados ISO e CMM - apresentados mais adiante - asseguram quando são emitidos para uma empresa de software).

Portanto, segundo a abordagem de modelagem de processo, pode-se concluir que os fatores citados anteriormente (alta rotatividade, tanto dos desenvolvedores quanto da diretoria; inexperiência dos mesmos; experiência de desenvolvimento permanecer com os alunos e não com a empresa júnior) prejudicam a qualidade e a produtividade do NPI. E por isso, também segundo essa mesma abordagem, a solução para os problemas do NPI pode ser a melhora da qualidade do seu processo de desenvolvimento de software, através da elaboração de um *modelo de processo de software* prescritivo.

4.2 Modelo de Processo de Software, sua Avaliação e Melhoria

Um **modelo de processo de software** define o que deve ser realizado em cada fase do desenvolvimento e dá as instruções de como realizar essas atividades. Ele serve como um guia, um roteiro para a execução de um processo de desenvolvimento.

De acordo com ULRIKE, HAMANN e VERLAGE (1997), a tarefa de implantação (ou melhora da qualidade) de um processo de software engloba duas atividades principais: a elaboração de um **modelo de processo descritivo** e depois, de um **modelo de processo prescritivo**. "Um modelo descritivo retrata como um processo é executado em um ambiente em particular; um modelo prescritivo retrata como um processo *deveria* ser executado" (Idem, op. cit., p.2, tradução livre). Essa abordagem é utilizada neste artigo: a seção 2 apresenta o modelo descritivo do processo do NPI e a seção 5 apresenta o modelo prescritivo.

Existem diversos modelos para avaliação e melhoria do processo de software. Os principais e mais conhecidos são o CMM e o SPICE (ISO15504). Uma diferença fundamental entre modelos de processo e modelos de avaliação, é que os modelos de avaliação apenas definem **o que** deve ser realizado durante o processo de desenvolvimento para ele possa ser considerado um processo de qualidade, enquanto os modelos de processo contêm instruções de **como** realizar as tarefas que o tornarão um processo de desenvolvimento de qualidade. Tanto o CMM (CMMI PRODUCT TEAM, 2002) quanto o SPICE (REZENDE, 1999, p. 162-165 e SPICE PROJECT ORGANISATION, 1995) definem *níveis de capacidade* (ou seja, de qualidade), que podem ter valores diferentes para cada uma das partes do processo. Por exemplo, a empresa pode ser nível 5 em gerência de configuração, e nível 3 em suporte pós-venda. Assim, continuando com o exemplo, eles apenas definem *o que* caracteriza um processo de gerência de configuração nível 5, mas não explicam *como* realizar esse processo.

4.3 A Literatura da Área

Os modelos de avaliação são bastante divulgados e conhecidos, ao contrário de modelos de processo, que são raramente divulgados. Entende-se que a explicação para esse fato, é que geralmente a modelagem de processos é feita por empresas de consultoria, que não iriam revelar sua "metodologia de desenvolvimento de modelos de processo" pois ela é a base do seu negócio (seria como o fabricante da Coca-Cola revelar a sua fórmula).

Para a realização do trabalho descrito neste artigo, foi realizada uma pesquisa bibliográfica de relatos de experiências de modelagem de processo, mas novamente, nenhum deles apresentava o modelo desenvolvido em detalhes. Além disso, os relatos de experiências em micro e pequenas empresas também são muito escassos, de modo que, nos três estudos cuja pesquisa foi aprofundada, apenas WEBER (2002) tratava desse tipo de empresa. Os outros dois estudos (LIMA, 2002 e TAVARES, PAIM e CARVALHO, 2002) tratam de uma empresa média e de uma empresa grande. Algumas conclusões dos autores, entretanto, se aplicam a organizações de qualquer tamanho, como por exemplo, combater a tendência dos desenvolvedores de ir diretamente para a fase de projeto através do fortalecimento dos processos de requisitos.

5 O Modelo de Processo Proposto

A partir da análise do processo de desenvolvimento existente no NPI, foram definidos os seus requisitos para um processo de desenvolvimento de software ideal.

O modelo de processo proposto foi elaborado a partir do processo existente e desses requisitos. Inicialmente foi elaborado o modelo de ciclo de vida (baseado no existente) e então foram sendo definidas as fases e suas atividades. Para cada fase foram definidas suas características (como atividades, critérios de entrada, critérios de saída, papéis e responsabilidades, objetivos, etc), as quais foram colocadas em um quadro (chamado de quadro das características). Após a elaboração desse quadro, foram listadas as atividades seguidas sempre de uma explicação detalhada de como realizá-las. Em adição a esses dados, o modelo inclui também uma série de documentos-padrão, que devem ser preenchidos ou alterados de acordo com as informações de cada projeto.

Além da explicação do modelo de ciclo de vida e da definição das fases, o modelo de processo desenvolvido também apresenta explicações introdutórias de como utilizá-lo, como mudá-lo, como preencher os documentos, entre outras.

O Quadro 1 apresenta as fases do modelo de processo desenvolvido e os seus objetivos.

Fase	Objetivo
Proposta	Realizar uma proposta de projeto para o cliente, contendo estimativas de tempo e custo. Negociar a proposta com o cliente.
Contrato	Assinar o contrato com a Empresa cliente. Assinar o contrato com os Estagiários.
Requisitos	Fazer a análise mais detalhada dos requisitos.
Projeto	Fazer o projeto do sistema de modo que ele possa ser mapeado diretamente em código.
Implementação	A partir do projeto, escrever o código.
Testes	Preparar os requisitos para o projeto dos casos de teste. Projetar e executar os casos testes
Entrega	Entregar o software ao cliente.
Gerência	Gerenciar o desenvolvimento do software.

Quadro 1: Resumo das fases do modelo

Como pode ser visto, o modelo de processo produzido inclui tanto atividades técnicas como gerenciais, abrangendo todas as fases de um processo típico de desenvolvimento de software do NPI (desde o primeiro contato do cliente até a entrega da versão final do software).

A fase de Gerência (assim como a fase de Testes) recebeu uma atenção especial, pois suas atividades eram bastante ausentes no NPI. Foram definidas várias políticas para as atividades gerenciais, como controle do cronograma, controle de versões, acompanhamento do trabalho, entre outras. Para a fase de Testes, foi adotada uma adaptação da metodologia proposta por BINDER (1999).

Além disso, o modelo também aborda algumas questões humanas que são importantes de se considerar na sua utilização (por exemplo, ênfase no fato de que o modelo deve ser visto como um guia e não como uma imposição burocrática da diretoria).

6 A avaliação

Com o objetivo de avaliar a aplicabilidade do modelo e os seus pontos fortes e fracos no NPI, é apresentado (em linhas gerais) o plano para a avaliação do modelo. Esse plano será executado em paralelo à utilização do modelo pelo NPI em um projeto piloto. Como no decorrer do trabalho descrito neste artigo, não se iniciou nenhum novo projeto na empresa júnior, a execução do plano de avaliação não foi possível.

O plano de avaliação foi elaborado segundo a metodologia GQM (*Goal Question Metric*), que possui uma série de passos e documentos bem definidos que devem ser elaborados durante a produção do plano e a sua execução.

Inicialmente, deve-se definir as metas da avaliação, ou seja, quais aspectos do processo serão avaliados. Para o plano de avaliação apresentado, definiram-se as seguintes metas: qualidade, duração e gerência dos projetos. Então, foram elaboradas as *abstraction sheets*, que são tabelas onde constam resumidamente os elementos de cada meta a serem avaliados, os fatores que influenciam a sua variação, as hipóteses dos valores desses elementos, entre outros. Para a meta qualidade foram definidos os seguintes elementos: consistência entre requisitos e projeto, consistência entre projeto e implementação e grau de sistematização da produção dos documentos. Em seguida, foram elaboradas perguntas que derivam das *abstraction sheets* e que apresentam claramente *o quê* será avaliado e quais serão os critérios dessa avaliação (segundo com o exemplo, foi definida a pergunta "Qual o nível de consistência entre requisitos e projeto?" e as opções de respostas "Alto, Médio, Baixo" seguidas de uma explicação precisa do que é Alto, Médio e Baixo). A partir dessas definições desenvolveu-se o plano de mensuração, que apresenta instruções de quando coletar os dados, qual instrumento utilizar para a coleta, quem deve coletar os dados, e como eles se relacionam com as metas e as perguntas. Os instrumentos de coleta são essencialmente roteiros para entrevistas com os membros do NPI e também um roteiro para inspeção de documentos. Além disso, foram criadas tabelas e gráficos com lacunas para serem preenchidas após a execução da avaliação. O plano de avaliação completo encontra-se como anexo no trabalho de SOUZA (2004).

7 Conclusão

Os três principais resultados do trabalho aqui relatado são: a análise do processo existente no NPI, o modelo de processo de desenvolvimento de software do NPI - que é o resultado mais importante - e o plano de avaliação.

Os principais benefícios do modelo de processo elaborado para o NPI, se ele for seguido corretamente, poderão ser: melhoria da qualidade do processo e do software desenvolvido, fruição de uma visão clara e definida do seu processo de desenvolvimento, aumento do controle e do planejamento gerencial dos seus projetos, preservação das experiências adquiridas nos projetos para serem utilizadas por futuras gestões. Além disso, comparando o modelo de processo proposto com os outros três modelos pesquisados e com os modelos de avaliação, percebemos que o modelo proposto atende a um número maior de requisitos e que ele atende à maioria dos requisitos do NPI (enquanto os outros satisfazem apenas três ou quatro requisitos). Essa comparação é apresentada em mais detalhes por SOUZA (2004). Assim, pode-se concluir que o modelo é adequado ao NPI.

No decorrer da realização do trabalho apresentado neste artigo, não foram iniciados novos projetos de desenvolvimento de software no NPI, de modo que não foi possível executar o plano de avaliação elaborado (entretanto, o plano está totalmente disponível para ser utilizado assim que se iniciar um novo projeto). A utilização do modelo em paralelo com a execução da avaliação, seguida da melhoria do modelo com base na avaliação, é um processo que será realizado continuamente pelos próprios membros do NPI. Portanto, o modelo proposto é apenas a primeira versão do modelo de processo de desenvolvimento de software do NPI, que ainda será melhorado muitas vezes até que esteja completamente adequado a essa empresa júnior.

8 Bibliografia

BINDER, Robert V. **Extended Use Case Test Design Patterns**. 1999. Disponível em: <www.rbsc.com/docs/TestPatternXUC.pdf>. Acesso em: 14 out. 2003.

CMMI PRODUCT TEAM. **Capability Maturity Model Integration for Software Engineering, Version 1.1, Continuous Representation**. ago 2002. CMU/SEI-2002-TR-028. Disponível em: <<http://www.sei.cmu.edu/pub/documents/02.reports/pdf/02tr028.pdf>>. Acesso em: 15 jun 2003.

LIMA, Rodrigo Duran. Gerência de Projetos de Software com RUP, CMM e ISO 9001. In: CONGRESSO INTERNACIONAL DE TECNOLOGIA DE SOFTWARE, 13., 2002, Curitiba. **Anais...** Curitiba: CITS, 2002. p.107-115.

NÚCLEO DE PROJETOS EM INFORMÁTICA. **Estatuto**. Florianópolis, 1996. Universidade Federal de Santa Catarina.

NÚCLEO DE PROJETOS EM INFORMÁTICA. **NPI - Núcleo de Projetos em Informática**. 2002. Universidade Federal de Santa Catarina. Disponível em: <<http://www.npi.inf.ufsc.br>>. Acesso em: 21 mar. 2003.

NÚCLEO DE PROJETOS EM INFORMÁTICA. **Manual do Processo de Desenvolvimento de Software do NPI**. Florianópolis, 2003. Universidade Federal de Santa Catarina.

REZENDE, Denis Alcides. **Engenharia de Software e Sistemas de Informações**. Rio de Janeiro: Brasport, 1999.

ROCHA, Ana Regina. **Qualidade de Software: processo e produto**. In: ENCONTRO DA QUALIDADE E PRODUTIVIDADE EM SOFTWARE, 2002, Petrópolis. Disponível em <http://www.mct.gov.br/Temas/info/Dsi/PBQP/Reuniao_Petropolis/Palestra_COPPE.pdf>. Acesso em: 29 jun. 2003.

SOUZA, Máira Bay de. Modelo de Processo de Software: aplicação em uma empresa júnior. 2004. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Universidade Federal de Santa Catarina, Florianópolis, 2004.

SPICE PROJECT ORGANISATION. **Software Process Assessment**. jun. 1995. Technical Report Type 2. Disponível em: <<http://www.sqi.gu.edu.au/spice/suite/download.html>>. Acesso em: 29 jun. 2003.

TAVARES, Helena Cristina; PAIM, Fábio Rilston Silva; CARVALHO, Ana Elizabete. Implantando CMM Nível 2: A Estratégia SERPRO. In: SIMPÓSIO BRASILEIRO DE QUALIDADE DE SOFTWARE, 1., 2002, Gramado. **Anais...** 1 CD-ROM.

ULRIKE, Becker; HAMANN, Dirk; VERLAGE, Martin. **Descriptive Modeling of Software Process**. dez. 1997. IESE-Resport 045.97/E. Disponível em: <http://www.iese.fhg.de/Publications/lese_reports/>. Acesso em: 28 jun. 2003.

WANGENHEIM, Christiane Gresse won. **Melhoramento de Software Baseado em Mensuração: Como Aplicar GQM na Prática?** Florianópolis: GeNESS, 1999. Relatório Técnico GeNESS 001.99P.

WEBER, Sérgio. **Um estudo de caso em uma micro empresa de software para o desenvolvimento de um modelo de processo de software**. 2002. Trabalho de Conclusão de Curso (Bacharelado em Ciências da Computação) - Universidade Federal de Santa Catarina, Florianópolis, 2002.

Apêndice B - Plano da Avaliação



NÚCLEO DE PROJETOS EM INFORMÁTICA

PLANO DE AVALIAÇÃO DO MPDS-NPI (VERSÃO 1.0)

SUMÁRIO

1 - Plano GQM	138
1.1 - Abstraction Sheets	138
1.1.1 - Qualidade e Duração	138
1.1.2 - Gerência	139
1.2 - Perguntas GQM Resumidas	140
1.2.1 - Qualidade e Duração	140
1.2.1.1 - Definição das Variações	140
1.2.1.2 - Definição da Qualidade	140
1.2.2 - Gerência	140
1.2.2.1 - Definição das Variações	140
1.2.2.2 - Definição da Qualidade	141
1.3 - Perguntas GQM em Detalhes	142
1.3.1 - Qualidade e Duração	142
1.3.1.1 - Definição das Variações	142
1.3.1.2 - Definição da Qualidade	143
1.3.2 - Gerência	145
1.3.2.1 - Definição das Variações	145
1.3.2.2 - Definição da Qualidade	145
2 - Plano de Mensuração	147
2.1 - Quadro do Plano	147
2.2 - Instrumentos para Coleta de Dados	148
2.2.1 - Roteiro para Entrevista com a Equipe de Desenvolvimento	148
2.2.2 - Roteiro para Entrevista com os Diretores	149
2.2.3 - Roteiro para Entrevista com o Gerente	150
2.2.4 - Roteiro para Inspeção dos Documentos	151
3 - Preparação para a Análise e a Interpretação	152
3.1 - Representação em Tabela	152
3.1.1 - Qualidade e Duração	152
3.1.1.1 - Qualidade	152
3.1.1.2 - Duração	153
3.1.2 - Gerência	154
3.2 - Representação em Gráficos	155

1 - Plano GQM

O plano GQM é composto de duas partes: as Abstraction Sheets e as Perguntas GQM. As perguntas são apresentadas em detalhes e também de forma resumida, para facilitar uma leitura rápida do plano.

1.1 - Abstraction Sheets

As Abstraction Sheets resumem as metas do plano de avaliação.

A primeira parte da Abstraction Sheet (Objeto, Objetivos, etc) está bem explicada em WANGENHEIM (1999, p.11).

Fatores de Qualidade são os fatores que serão avaliados pra cada meta. Os Fatores de Variação são fatores que influenciam nos fatores de qualidade. Em Impacto na Hipótese de Linha-Base são colocadas as hipóteses de como os fatores da variação irão influenciar os fatores de qualidade. A Hipótese de Linha-Base apresenta a medida do fator de qualidade que se supõe que existia Antes da aplicação do MPDS-NPI e que se supõe que irá existir Depois que o MPDS-NPI for aplicado.

1.1.1 - Qualidade e Duração

Objeto	Objetivo	Enfoque da Qualidade	Ponto de Vista	Contexto
Modelo de Processo	Avaliar	Qualidade e Duração	Pesquisador	NPI
Fatores de Qualidade		Fatores de Variação		
<p><i>Qualidade:</i></p> <ul style="list-style-type: none"> - Consistência entre requisitos e projeto - Consistência entre projeto e implementação - Sistematização da produção dos documentos <p><i>Duração:</i></p> <ul style="list-style-type: none"> - Atraso no Projeto 		<p><i>Qualidade:</i></p> <ul style="list-style-type: none"> - Comprometimento e conscientização da equipe de desenvolvimento da necessidade de utilização de técnicas de engenharia de software - Experiência da equipe em desenvolvimento de software <p><i>Duração:</i></p> <ul style="list-style-type: none"> - Experiência da equipe em fazer estimativas - Quantidade de informação sobre a duração de projetos anteriores semelhantes 		
Hipótese de Linha-Base		Impacto na Hipótese de Linha-Base		
Antes	Depois (Hipótese)	<p><i>Qualidade:</i></p> <ul style="list-style-type: none"> - ↑ experiência em desenvolvimento de software: ↑ consistência entre requisitos e projeto ↑ consistência entre projeto e implementação - ↑ comprometimento e conscientização da necessidade de utilizar técnicas de engenharia de software ↑ consistência entre requisitos e projeto ↑ consistência entre projeto e implementação ↑ produção sistematizada de documentos <p><i>Duração:</i></p> <ul style="list-style-type: none"> - ↑ experiência da equipe em fazer estimativas ↓ erro nas estimativas (e portanto ↓ o atraso) - ↑ informação sobre a duração de projetos anteriores semelhantes ↓ erro nas estimativas (e portanto ↓ o atraso) 		
<p><i>Qualidade:</i></p> <ul style="list-style-type: none"> - O nível de consistência entre os requisitos e o projeto é Baixo. - O nível de consistência entre o projeto e a implementação é Baixo. - Não há uma ordem para a produção dos documentos. <p><i>Duração:</i></p> <ul style="list-style-type: none"> - O atraso no projeto é de 5 meses. 	<p><i>Qualidade:</i></p> <ul style="list-style-type: none"> - O nível de consistência entre os requisitos e o projeto é Alto. - O nível de consistência entre o projeto e a implementação é Alto. - Há um modelo de processo que determina a ordem de produção dos documentos e essa ordem é seguida. <p><i>Duração:</i></p> <ul style="list-style-type: none"> - O atraso no projeto é de 1 mês ou menos. 			

1.1.2 - Gerência

Objeto	Objetivo	Enfoque da Qualidade	Ponto de Vista	Contexto
Modelo de Processo	Avaliar	Gerência	Pesquisador	NPI
Fatores de Qualidade			Fatores de Variação	
<ul style="list-style-type: none"> - Controle de versões - Planejamento e controle do cronograma - Acompanhamento e supervisão do trabalho 			<ul style="list-style-type: none"> - Tempo disponível do gerente para o projeto - Tamanho do Projeto 	
Hipótese de Linha-Base			Impacto na Hipótese de Linha-Base	
Antes		Depois (Hipótese)		
<ul style="list-style-type: none"> - Não há nenhuma política para o controle de versões dos documentos. - Existe um cronograma mas ele não é seguido. - O acompanhamento do trabalho é Insatisfatório, e é abandonado no meio do projeto. 		<ul style="list-style-type: none"> - Há uma política documentada para o controle de versões dos documentos e ela é seguida. - Existe um cronograma, ele é re-elaborado quando ocorrem atrasos, e é seguido durante todo o projeto. - O acompanhamento do trabalho é Satisfatório, e é seguido durante todo o projeto. 		
		<ul style="list-style-type: none"> - ↑ tempo disponível do gerente para o projeto ↑ controle de versões ↑ planejamento e controle do cronograma ↑ acompanhamento e supervisão do trabalho - ↑ tamanho do projeto ↑ necessidade de controle das versões ↑ necessidade de planejamento e controle do cronograma 		

1.2 - Perguntas GQM Resumidas

As perguntas GQM apresentam-se primeiro em forma resumida. Essa forma apresenta apenas as perguntas e as sub-perguntas derivadas delas.

1.2.1 - Qualidade e Duração

As perguntas sobre Qualidade e Duração foram agrupadas em um item só, pois elas estão na mesma Abstraction Sheet.

1.2.1.1 - Definição das Variações

Q1) A experiência da equipe em desenvolvimento de software tem um impacto na consistência entre os requisitos e o projeto?

Q1.1) Qual a experiência da equipe em desenvolvimento de software?

Q2) A experiência da equipe em desenvolvimento de software tem um impacto na consistência entre o projeto e a implementação?

Q3) O nível de comprometimento e conscientização da equipe em relação à necessidade da utilização de técnicas de engenharia de software tem um impacto na sistematização da produção dos documentos?

Q3.1) Qual o nível de comprometimento e conscientização da equipe em relação à necessidade da utilização de técnicas de engenharia de software?

Q4) O nível de comprometimento e conscientização da equipe em relação à necessidade da utilização de técnicas de engenharia de software tem um impacto na consistência entre os requisitos e o projeto?

Q5) O nível de comprometimento e conscientização da equipe em relação à necessidade da utilização de técnicas de engenharia de software tem um impacto na consistência entre o projeto e a implementação?

Q6) A experiência da equipe em fazer estimativas influencia a precisão das estimativas?

Q6.1) Qual a experiência da equipe em fazer estimativas?

Q7) A quantidade de informação sobre a duração de projetos anteriores semelhantes influencia a precisão das estimativas (e portanto, no atraso do projeto)?

Q7.1) Qual a quantidade de informação sobre a duração de projetos anteriores semelhantes?

1.2.1.2 - Definição da Qualidade

Q8) Qual o nível de consistência entre os requisitos e o projeto?

Q9) Qual o nível de consistência entre o projeto e a implementação?

Q10) Qual o grau de sistematização da produção dos documentos?

Q11) Qual o atraso do projeto?

1.2.2 - Gerência

1.2.2.1 - Definição das Variações

Q12) O tempo disponível do Gerente para o projeto tem um impacto na qualidade do controle de versões?

Q12.1) Qual o tempo disponível do gerente para o projeto?

Q13) O tempo disponível do Gerente para o projeto tem um impacto na qualidade do planejamento e controle do cronograma?

Q14) O tempo disponível do Gerente para o projeto tem um impacto na qualidade do acompanhamento e supervisão do trabalho?

Q.15) O tamanho do projeto altera a necessidade de controle de versões?

Q15.1) Qual o tamanho do projeto?

Q16) O tamanho do projeto altera a necessidade de planejamento e controle de cronograma?

1.2.2.2 - Definição da Qualidade

Q17) Qual o nível de qualidade do controle das versões dos documentos?

Q18) Qual o nível de qualidade do controle do cronograma?

Q19) Qual o nível de qualidade do acompanhamento do trabalho?

1.3 - Perguntas GQM em Detalhes

Nesta parte, as perguntas GQM estão detalhadas, contendo:

- * *Hipótese* (o conteúdo de Impacto na Hipótese de Linha-Base da Abstraction Sheet),
- * Modelo (como calcular a medida),
- * MX.X (número da medida, seguido do seu nome e dos valores que ela pode assumir),
- * *Hipótese Antes* (conteúdo da Hipótese de Linha-Base Antes da Abstraction Sheet) e
- * *Hipótese Depois* (conteúdo da Hipótese de Linha-Base Depois da Abstraction Sheet).

1.3.1 - Qualidade e Duração

Novamente, as perguntas sobre Qualidade e Duração foram agrupadas em um item só, pois elas estão na mesma Abstraction Sheet.

1.3.1.1 - Definição das Variações

Q1) A experiência da equipe em desenvolvimento de software tem um impacto na consistência entre os requisitos e o projeto?

Hipótese: quanto mais experiência a equipe possui em desenvolvimento de software, maior será a consistência entre os requisitos e o projeto.

Q1.1) Qual a experiência da equipe em desenvolvimento de software?

Modelo: a experiência da equipe deve ser a média aritmética das experiências de todos os membros da equipe.

M1.1 experiência da equipe [ordinal:

"Grande (trabalhou na área mais de 4 anos)";

"Média (trabalhou na área entre 2 e 4 anos)";

"Pequena (trabalhou na área menos de 2 anos)"].

Hipóteses Antes e Depois: a equipe possui uma experiência Média.

Q2) A experiência da equipe em desenvolvimento de software tem um impacto na consistência entre o projeto e a implementação?

Hipótese: quanto mais experiência a equipe possui em desenvolvimento de software, maior será a consistência entre o projeto e a implementação.

Q3) O nível de comprometimento e conscientização da equipe em relação à necessidade da utilização de técnicas de engenharia de software tem um impacto na sistematização da produção dos documentos?

Hipótese: quanto mais comprometida e conscientizada em relação à necessidade de utilização de técnicas de engenharia de software a equipe está, mais sistematizada será a produção dos documentos.

Q3.1) Qual o nível de comprometimento e conscientização da equipe em relação à necessidade da utilização de técnicas de engenharia de software?

Modelo: o nível de comprometimento e conscientização da equipe em relação à necessidade da utilização de técnicas de ES, pode ser medido através da importância que a equipe atribui à utilização de técnicas de ES e do questionamento se a equipe continuaria utilizando essas técnicas mesmo diante de acontecimentos críticos como atraso no cronograma.

M3.1 valor de importância atribuído à ES [ordinal: é importante; não é importante].

Hipóteses Antes e Depois: a equipe acha que é importante utilizar técnicas de ES.

M3.2 continuaria utilizando a técnica mesmo diante de acontecimentos críticos (como atraso no cronograma) [ordinal: sim; não].

Hipótese Antes: a equipe abandonaria a ES diante de acontecimentos críticos.

Hipótese Depois: a equipe continuaria utilizando a ES mesmo diante de acontecimentos críticos.

Q4) O nível de comprometimento e conscientização da equipe em relação à necessidade da utilização de técnicas de engenharia de software tem um impacto na consistência entre os requisitos e o projeto?

Hipótese: quanto mais comprometida e conscientizada em relação à necessidade de utilização de técnicas de engenharia de software a equipe está, maior será a consistência entre os requisitos e o projeto.

Q5) O nível de comprometimento e conscientização da equipe em relação à necessidade da utilização de técnicas de engenharia de software tem um impacto na consistência entre o projeto e a implementação?

Hipótese: quanto mais comprometida e conscientizada em relação à necessidade de utilização de técnicas de engenharia de software a equipe está, maior será a consistência entre o projeto e a implementação.

Q6) A experiência da equipe em fazer estimativas influencia a precisão das estimativas?

Hipótese: quanto mais experiência a equipe tem em fazer estimativas, mais precisas são as suas estimativas.

Q6.1) Qual a experiência da equipe em fazer estimativas?

Modelo: a experiência da equipe em fazer estimativas pode ser medida pelo número de projetos para os quais a equipe já fez estimativas. Deve ser feita uma média aritmética do número de projetos feitos por cada membro da equipe.

M6.1 experiência da equipe em fazer estimativas [ordinal:
"Grande (já realizou estimativas para mais de 6 projetos)";
"Média (já realizou estimativas para 3, 4 ou 5 projetos)";
"Pequena (já realizou estimativas para 2 ou menos projetos)"].

Hipótese Antes: a equipe possui experiência Pequena.

Hipótese Depois: a equipe possui experiência Média.

Q7) A quantidade de informação sobre a duração de projetos anteriores semelhantes influencia a precisão das estimativas (e portanto, no atraso do projeto)?

Hipótese: quanto mais informação sobre a duração de projetos anteriores semelhantes, melhor será a estimativa realizada pela equipe.

Q7.1) Qual a quantidade de informação sobre a duração de projetos anteriores semelhantes?

Modelo: pode ser medida pela existência ou não de informações sobre a duração real e a duração planejada dos projetos anteriores semelhantes.

M7.1 informação sobre projetos anteriores semelhantes [ordinal:
"Existe informação sobre a duração real e planejada";
"Existe informação apenas sobre a duração planejada";
"Não existe informação sobre a duração"].

Hipótese Antes: não existe informação sobre a duração de projetos anteriores semelhantes.

Hipótese Depois: existe informação sobre a duração real e planejada de projetos anteriores semelhantes.

1.3.1.2 - Definição da Qualidade

Q8) Qual o nível de consistência entre os requisitos e o projeto?

Modelo: o nível de consistência entre os requisitos e o projeto pode ser medido comparando-se as operações de sistema e os diagramas de colaboração.

M8.1 nível de consistência entre as operações de sistema e os diagramas de colaboração [ordinal:
"Alto (para cada operação de sistema de cada caso de uso, existe um diagrama de colaboração)"
"Médio (para 70% a 99% das operações de sistema, existe um diagrama de colaboração correspondente a cada operação)"

"Baixo (para menos de 70% das operações de sistema, existe um diagrama de colaboração correspondente a cada operação)"].

Hipótese Antes: o nível de consistência entre as operações de sistema e os diagramas de colaboração é Baixo.

Hipótese Depois: o nível de consistência entre as operações de sistema e os diagramas de colaboração é Alto.

Q9) Qual o nível de consistência entre o projeto e a implementação?

Modelo: esse nível pode ser medido comparando-se o diagrama de classes com as classes implementadas.

M9.1 nível de consistência entre as classes do projeto e do código [ordinal:

"Alto (todas as classes projetadas foram implementadas)";

"Médio (de 70% a 99% das classes que aparecem no diagrama de classes foram implementadas)";

"Baixo (menos de 70% das classes projetadas foram implementadas)"].

Hipótese Antes: o nível de consistência entre as classes do projeto e do código é Baixo.

Hipótese Depois: o nível de consistência entre as classes do projeto e do código é Alto.

M9.2 nível de consistência entre os nomes das classes do projeto e do código [ordinal:

"Alto (os nomes das classes implementadas e projetadas são iguais)";

"Médio (70% a 99% dos nomes das classes implementadas são iguais aos nomes das classes projetadas)";

"Baixo (menos de 70% dos nomes das classes implementadas são iguais aos nomes projetados para elas no diagrama de classes)"].

Hipótese Antes: o nível de consistência entre os nomes das classes do projeto e do código é Baixo.

Hipótese Depois: o nível de consistência entre os nomes das classes do projeto e do código é Alto.

M9.3 nível de consistência entre os atributos e métodos das classes do projeto e do código [ordinal:

"Alto (todos os atributos e métodos de todas as classes que aparecem no diagrama de classes foram implementados em suas devidas classes)";

"Médio" (70% a 99% dos atributos e métodos que aparecem no diagrama de classes foram implementados)";

"Baixo (menos de 70% dos atributos e métodos que foram projetados estão implementados)"].

Hipótese Antes: a consistência entre os atributos e métodos das classes do projeto e do código é Baixa.

Hipótese Depois: a consistência entre os atributos e métodos das classes do projeto e do código é Alta.

M9.4 nível de consistência entre os nomes dos atributos e métodos das classes do projeto e do código [ordinal:

"Alto (todos os atributos e métodos possuem o mesmo nome na implementação e no projeto)";

"Médio (70% a 90% dos atributos e métodos implementados possuem o mesmo nome projetado)";

"Baixo (menos de 70% dos atributos e métodos possuem o mesmo nome na implementação e no projeto)"].

Hipótese Antes: o nível consistência entre os nomes dos atributos e métodos das classes do projeto e do código é Baixo.

Hipótese Depois: o nível de consistência entre os nomes dos atributos e métodos das classes do projeto e do código é Alto.

Q10) Qual o grau de sistematização da produção dos documentos?

Modelo: esse grau pode ser medido pela existência ou não de um processo sistematizado para a produção e por sua utilização ou não.

M10.1 existência de um processo sistematizado [ordinal:

"Existe um modelo de processo";

"Existe uma ordem documentada informalmente";

"Existe uma ordem não-documentada";

"Não existe uma ordem"].

Hipótese Antes: não existe uma ordem para a produção dos documentos.

Hipótese Depois: há um modelo de processo que determina a ordem de produção dos documentos.

M10.2 utilização da ordem [ordinal: é seguida; não é seguida].

Hipótese Antes: como não existe ordem, ela não é seguida.

Hipótese Depois: a ordem é seguida.

Q11) Qual o atraso do projeto?

Modelo: atraso = data prevista para o fim do projeto - data real do fim do projeto

M11.1 data prevista [intervalo: data]

M11.2 data real [intervalo: data]

Hipótese Antes: o atraso é de 5 meses.

Hipótese Depois: o atraso é de 1 mês ou menos.

1.3.2 - Gerência

1.3.2.1 - Definição das Variações

Q12) O tempo disponível do Gerente para o projeto tem um impacto na qualidade do controle de versões?

Hipótese: quanto mais tempo disponível para o projeto o Gerente tem, melhor é o controle de versões.

Q12.1) Qual o tempo disponível do gerente para o projeto?

Modelo: esse tempo pode ser medido pelo número de horas que o gerente se dedica ao projeto por semana.

M12.1 dedicação do gerente [racional: inteiro (horas/semana)].

Hipótese Antes: 5 horas/semana.

Hipótese Depois: 10 horas por semana.

Q13) O tempo disponível do Gerente para o projeto tem um impacto na qualidade do planejamento e controle do cronograma?

Hipótese: quanto mais tempo disponível para o projeto o Gerente tem, melhor é a qualidade do planejamento e do controle do cronograma, realizados por ele.

Q14) O tempo disponível do Gerente para o projeto tem um impacto na qualidade do acompanhamento e supervisão do trabalho?

Hipótese: quanto mais tempo disponível para o projeto o Gerente tem, melhor é a qualidade do acompanhamento e da supervisão do trabalho, realizados por ele.

Q.15) O tamanho do projeto altera a necessidade de controle de versões?

Hipótese: quanto maior o projeto, mais crítico se torna o controle de versões.

Q15.1) Qual o tamanho do projeto?

Modelo: o tamanho de um projeto pode ser medido pelo tempo que foi estimado que ele irá durar.

M15.1 tamanho do projeto [ordinal:

"Grande (duração planejada maior do que 8 meses)";

"Médio (duração planejada entre 4 e 8 meses)";

"Pequeno (duração planejada menor ou igual a 4 meses)"].

Hipótese Antes: Pequeno.

Hipótese Depois: Médio.

Q16) O tamanho do projeto altera a necessidade de planejamento e controle de cronograma?

Hipótese: quanto maior o projeto, mais críticos são o planejamento e o controle do cronograma.

1.3.2.2 - Definição da Qualidade

Q17) Qual o nível de qualidade do controle das versões dos documentos?

Modelo: esse nível pode ser medido através da constatação da existência ou não de uma política para o controle das versões, da verificação se essa política é documentada ou não e da verificação se a política é seguida ou não.

M17.1 existência de uma política para o controle de versões [ordinal: existe; não existe].

Hipótese Antes: não existe uma política para o controle de versões.

Hipótese Depois: existe uma política para o controle de versões.

M17.2 documentação ou não da política [ordinal:

"A política está documentada";

"A política existe como uma cultura ou modelo mental não-documentado"].

Hipótese Antes: como não existe política, consideramos a segunda opção.

Hipótese Depois: a política está documentada.

M17.3 utilização da política [ordinal:

"A política é seguida sempre";

"A política não é seguida"].

Hipótese Antes: a política não é seguida, pois não há política.

Hipótese Depois: a política é seguida.

Q18) Qual o nível de qualidade do controle do cronograma?

Modelo: esse nível pode ser medido pela constatação da existência ou não de um cronograma; pela verificação se ele é seguido ou não durante todo o projeto e pela verificação se o cronograma é re-elaborado quando ocorrem atrasos.

M18.1 existência do cronograma [ordinal: existe; não existe].

Hipóteses Antes e Depois: existe um cronograma.

M18.2 utilização do cronograma [ordinal:

"O cronograma é seguido em todo o projeto";

"O cronograma é abandonado após a re-elaboração";

"O cronograma nunca é seguido"].

Hipótese Antes: o cronograma nunca é seguido.

Hipótese Depois: o cronograma é seguido em todo o projeto.

M18.3 O cronograma é re-elaborado quando ocorrem atrasos [ordinal: sim; não].

Hipótese Antes: o cronograma não é re-elaborado quando ocorrem atrasos.

Hipótese Depois: o cronograma é re-elaborado quando ocorrem atrasos.

Q19) Qual o nível de qualidade do acompanhamento do trabalho?

Modelo: esse nível pode ser medido pela maneira como ele é feito e também pelo seguimento ou não dessa maneira de acompanhamento até o fim do projeto.

M19.1 maneira de acompanhar o trabalho [ordinal:

"Satisfatória (são registradas as horas trabalhadas e as atividades realizadas nessas horas)";

"Média (são realizadas comunicações informais do estado do desenvolvimento)";

"Insatisfatória (nenhum acompanhamento é realizado)"].

Hipótese Antes: a maneira de acompanhar o trabalho é Insatisfatória.

Hipótese Depois: a maneira de acompanhar o trabalho é Satisfatória.

M19.2 seguimento do acompanhamento [ordinal:

"O acompanhamento do trabalho é realizado durante todo o projeto";

"O acompanhamento do trabalho é abandonado no meio do projeto"].

Hipótese Antes: o acompanhamento do trabalho é abandonado no meio do projeto.

Hipótese Depois: o acompanhamento do trabalho é realizado durante todo o projeto.

2 - Plano de Mensuração

O Plano de Mensuração inclui um quadro que resume o plano e os instrumentos para coleta de dados.

2.1 - Quadro do Plano

O quadro é composto dos seguintes itens:

- * Número (o número da coleta),
- * Medidas Associadas (as medidas que serão coletadas),
- * Descrição (do assunto que tratam as medidas),
- * Tempo (em que parte do processo será realizada a coleta),
- * Papel (quem irá realizar a coleta) e
- * Instrumento (nome do instrumento que irá guiar a coleta).

Número	Medidas Associadas	Descrição	Tempo	Papel	Instrumento
1	M1.1, M3.1, M3.2	relação da equipe com a ES	após a atividade 5 da fase de Contrato	Pesquisador	Entrevista com a Equipe de Desenvolvimento
2	M6.1, M7.1	relação da equipe com estimativas	após a atividade 4 da fase de Proposta	Pesquisador	Entrevista com os Diretores
3	M8.1, M9.1, M9.2, M9.3, M9.4	consistência entre requisitos, projeto e implementação	após o fim do projeto	Pesquisador	Inspeção dos Documentos
4	M10.1, M10.2	sistematização da produção dos documentos	após o fim do projeto	Pesquisador	Entrevista com o Gerente
5	M11.1, M11.2	atraso do projeto	após o fim do projeto	Pesquisador	Inspeção dos Documentos
6	M12.1	dedicação do gerente ao projeto	após o fim do projeto	Pesquisador	Entrevista com o Gerente
7	M15.1	duração estimada do projeto	após a atividade 4 da fase de Proposta	Pesquisador	Entrevista com os Diretores
8	M17.1, M17.2, M17.3	nível de controle das versões dos documentos	após o fim do projeto	Pesquisador	Entrevista com o Gerente
9	M18.1, M18.2, M18.3	nível de planejamento e controle do cronograma	após o fim do projeto	Pesquisador	Entrevista com o Gerente
10	M19.1, M19.2	nível do acompanhamento do trabalho	após o fim do projeto	Pesquisador	Entrevista com o Gerente

2.2 - Instrumentos para Coleta de Dados

Os instrumentos para coleta de dados incluem três entrevistas e uma inspeção de documentos. A seguir são apresentados os roteiros para realização de cada uma dessas coletas.

2.2.1 - Roteiro para Entrevista com a Equipe de Desenvolvimento

Por Equipe de Desenvolvimento entende-se o Gerente e os outros Estagiários.

Essas perguntas devem ser realizadas a toda Equipe ao mesmo tempo. Eles devem entrar em consenso (dentro do possível) e escolher uma das respostas.

1 - (M1.1) Qual a experiência de vocês em desenvolvimento de software? (fazer uma média aritmética das experiências de cada Estagiário)

- a) Grande (já trabalharam na área por mais de 4 anos)
- b) Média (já trabalharam na área entre 2 e 4 anos)
- c) Pequena (já trabalharam na área por menos de 2 anos)

2 - (M3.1) Qual a importância que vocês vêem na utilização de técnicas de ES para o desenvolvimento de software no NPI?

- a) é importante
- b) não é importante

3 - (M3.2) Vocês continuariam utilizando a técnica mesmo diante de acontecimentos críticos (como atraso no cronograma) ?

- a) sim
- b) não

2.2.2 - Roteiro para Entrevista com os Diretores

Essas perguntas devem ser realizadas a todos os Diretores ao mesmo tempo. Eles devem entrar em consenso (dentro do possível) e escolher uma das respostas.

1 (M6.1) - Qual a experiência de vocês em fazer estimativas de duração de projetos? (fazer uma média aritmética das experiências de cada Diretor)

- a) Grande (já realizaram estimativas para mais de 6 projetos)
- b) Média (já realizaram estimativas para 3, 4 ou 5 projetos)
- c) Pequena (já realizaram estimativas para 2 ou menos projetos)

2 - (M7.1) Qual a quantidade de informação que vocês possuem sobre a duração de projetos anteriores semelhantes?

- a) Existe informação sobre a duração real e planejada de projetos anteriores semelhantes.
- b) Existe informação apenas sobre a duração planejada de projetos anteriores semelhantes.
- b) Não existe informação sobre a duração de projetos anteriores.

3 - (M15.1) Qual foi o tamanho que vocês estimaram para o projeto?

- a) Grande (duração planejada maior do que 8 meses)
- b) Médio (duração planejada entre 4 e 8 meses)
- c) Pequeno (duração planejada menor ou igual a 4 meses)

2.2.3 - Roteiro para Entrevista com o Gerente

1 - (M10.1) Existe um processo sistematizado para a produção dos documentos?

sim, e ele faz parte de um modelo de processo.

b) sim, existe uma ordem para a produção dos documentos mas ela não está documentada.

c) não. (pular a questão a seguir)

2 - (M10.2) Esse processo foi seguido durante todo o projeto?

a) sim.

b) não.

3 - (M12.1) Qual foi a sua dedicação para o projeto? ____ horas/semana aproximadamente.

4 - (M17.1) Existe uma política para o controle das versões dos documentos?

sim.

b) não. (pular as próximas 2 questões)

5 - (M17.2) Essa política está documentada de alguma maneira?

sim.

b) não.

6 - (M17.3) Essa política foi seguida durante todo o projeto?

a) sim.

b) não.

7 - (M18.1) Foi feito um cronograma para o projeto?

a) sim.

b) não. (pular as próximas 2 questões)

8 - (M18.2) Esse cronograma foi seguido durante todo o projeto?

a) sim, foi seguido durante todo o projeto.

b) não, ele foi abandonado após a re-elaboração.

c) não, o cronograma nunca foi seguido.

9 - Ocorreram atrasos no cronograma?

a) sim.

b) não. (pular a questão a seguir)

9 - (M18.3) Quando ocorreram atrasos, o cronograma foi re-elaborado?

a) sim.

b) não.

10 - (M19.1) Qual foi a qualidade do acompanhamento do trabalho?

a) Satisfatória (foram registradas as horas trabalhadas e as atividades realizadas nessas horas)

b) Média (foram realizadas comunicações informais do estado do desenvolvimento)

c) Insatisfatória (nenhum acompanhamento é realizado)

11 - (M19.2) A maneira de realizar o acompanhamento do trabalho foi seguida durante todo o projeto?

a) sim, o acompanhamento do trabalho é realizado durante todo o projeto.

b) não, o acompanhamento do trabalho é abandonado no meio do projeto.

2.2.4 - Roteiro para Inspeção dos Documentos

Peça ao Gerente do Projeto os documentos do projeto (de preferência, o último backup), e procure os seguintes documentos: "Documento de Pré-Planejamento do Projeto (DG-002)", "Documento de Fim do Projeto (DG-013)", "Casos de Uso (DT-002)", "Diagramas de Projeto (DT-004)" e o Código-fonte.

1 - (M8.1) Qual é o nível de consistência entre as operações de sistema e os diagramas de colaboração? (observar e comparar as operações de sistema registradas no DT-002 com os diagramas de colaboração presentes no DT-004)

- a) Alto (para cada operação de sistema de cada caso de uso, existe um diagrama de colaboração)
- b) Médio (para 70% a 99% das operações de sistema, existe um diagrama de colaboração correspondente a cada operação)
- c) Baixo (para menos de 70% das operações de sistema, existe um diagrama de colaboração correspondente a cada operação)

2 - (M9.1) Qual o nível de consistência entre as classes do projeto e do código? (observar e comparar as classes do diagrama de classes, presentes no DT-004, com as classes implementadas no código-fonte)

- a) Alto (todas as classes projetadas foram implementadas)
- b) Médio (de 70% a 99% das classes que aparecem no diagrama de classes foram implementadas)
- c) Baixo (menos de 70% das classes projetadas foram implementadas)

3 - (M9.2) Qual o nível de consistência entre os nomes das classes do projeto e do código? (idem 2)

- a) Alto (os nomes das classes implementadas e projetadas são iguais)
- b) Médio (70% a 99% dos nomes das classes implementadas são iguais aos nomes das classes projetadas)
- c) Baixo (menos de 70% dos nomes das classes implementadas são iguais aos nomes projetados para elas no diagrama de classes)

4 - (M9.3) Qual o nível de consistência entre os atributos e métodos das classes do projeto e do código? (idem 2)

- a) Alto (todos os atributos e métodos de todas as classes que aparecem no diagrama de classes foram implementados em suas devidas classes)
- b) Médio (70% a 99% dos atributos e métodos que aparecem no diagrama de classes foram implementados)
- c) Baixo (menos de 70% dos atributos e métodos que foram projetados estão implementados)

5 - (M9.4) Qual o nível de consistência entre os nomes dos atributos e métodos das classes do projeto e do código? (idem 2)

- a) Alto (todos os atributos e métodos possuem o mesmo nome na implementação e no projeto)
- b) Médio (70% a 90% dos atributos e métodos implementados possuem o mesmo nome projetado)
- c) Baixo (menos de 70% dos atributos e métodos possuem o mesmo nome na implementação e no projeto)

6 - (M11.1) Qual era a data prevista para o final do projeto? (copiar do DG-002) __/__/20__

7 - (M11.2) Qual foi a data real do final do projeto? (copiar do DG-013) __/__/20__

3.1.1.2 - Duração

	Qual a experiência da equipe em fazer estimativas de duração de projetos?	Qual a quantidade de informação que a equipe possui sobre a duração de projetos anteriores semelhantes?	Qual foi o atraso do Projeto?
Medida	M6.1	M7.1	M.11.1 e M11.2
Hipótese Antes	Pequena	Não há nenhuma informação sobre a duração de projetos anteriores.	5 meses
Hipótese Depois	Média	Existe informação sobre a duração real e planejada de projetos anteriores.	1 mês
Projeto Piloto 1			
Projeto Piloto 2			

3.2 - Representação em Gráficos

A representação em gráficos encontra-se no arquivo anexo "graficos-avaliacao.xls".

Para cada meta há uma planilha, contendo os gráficos e as tabelas a partir do qual eles foram construídos.

Cada medida foi separada em um gráfico e sua correspondente tabela.

Desse modo, após a coleta deve-se apenas preencher as tabelas do arquivo que o gráfico será atualizado automaticamente.