ALEXANDRE CHAVES COELHO

ESTUDO E PROPOSTA DE UM PROCESSO DE DESENVOLVIMENTO DE SOFTWARE EM UMA COOPERATIVA DE SOFTWARE LIVRE

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

LAVRAS
MINAS GERAIS – BRASIL
2007

ALEXANDRE CHAVES COELHO

ESTUDO E PROPOSTA DE UM PROCESSO DE DESENVOLVIMENTO DE SOFTWARE EM UMA COOPERATIVA DE SOFTWARE LIVRE

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Área de Concentração:

Engenharia de Software

Orientador:

Prof. Dr. Ahmed Ali Abdalla Esmin

LAVRAS
MINAS GERAIS – BRASIL
2007

Ficha Catalográfica preparada pela Divisão de Processo Técnico da Biblioteca Central da UFLA

Coelho, Alexandre Chaves

Implantação de um Processo de Desenvolvimento de Software em uma Cooperativa de Software Livre/ Alexandre Chaves Coelho. Lavras – Minas Gerais, 2007. 63p : il.

Monografia de Graduação – Universidade Federal de Lavras. Departamento de Ciência da Computação.

1. Informática. 2. Engenharia de Software. 3. Processo de Software. I. Coelho, A. C. II. Universidade Federal de Lavras. III. Título.

ALEXANDRE CHAVES COELHO

ESTUDO E PROPOSTA DE UM PROCESSO DE DESENVOLVIMENTO DE SOFTWARE EM UMA COOPERATIVA DE SOFTWARE LIVRE

Monografia de graduação apresentada ao Departamento de Ciência da Computação da Universidade Federal de Lavras como parte das exigências do curso de Ciência da Computação para obtenção do título de Bacharel em Ciência da Computação.

Aprovada em 8 de agosto de 2007

_____

Prof. Dr. José Monserrat Neto

_____

Prof. Dr. Plínio de Sá Leitão Júnior

_____

Prof. Dr. Ahmed Ali Abdalla Esmin
(orientador)

LAVRAS
MINAS GERAIS – BRASIL
2007

Dedico aos meus pais, José Celso e Maria Angélica, por fazerem parte de minha vida e serem exemplo de pessoas para mim, e aos meus irmãos Rafael e Juliana, pelo companheirismo e amizade, e por fazerem parte dessa minha família da qual orgulho tanto. Obrigado por tornarem possível este meu sonho.

AGRADECIMENTO

Agradeço em primeiro lugar a Deus, pela força concedida para realização desse trabalho, por estar ao meu lado e por ter me iluminado a cada momento.

A toda minha família pelo apoio concedido, pelo incentivo e por ter acreditado sempre em mim.

Aos meus colegas de turma, pelos momentos bons que passamos juntos, pelo aprendizado e por todo companheirismo.

Aos integrantes da minha república, pessoas com as quais pude conviver por estes anos, e com as quais pude compartilhar momentos de alegria e de amizade.

Ao Professor Ahmed, pessoa que me orientou através de seu conhecimento e de suas experiências, e a todos os demais professores e funcionários do Departamento de Ciência da Computação da UFLA.

Aos amigos da TecnoLivre, pelo aprendizado, pelas oportunidades, pela prestatividade e pela iniciativa de cada um de vocês para tornar o sonho da TecnoLivre uma realidade.

Estudo e Proposta de um Processo de Desenvolvimento de Software em uma Cooperativa de Software Livre

Resumo. A utilização de um processo de desenvolvimento de software é de extrema importância para a padronização do desenvolvimento, permitindo assim gerar softwares dentro de cronogramas e preços estabelecidos e que venha a atender aos requisitos propostos. Este trabalho descreve os passos utilizados para a implantação de um processo em uma cooperativa de informática, desde a análise da cooperativa, de seus projetos e suas equipes. A análise de alguns processos que vêm sendo utilizados é descrita, e finalmente um processo de software é proposto.
Palavras-Chave: Engenharia de Software, Processo de Software, Cooperativa de Informática.

Study and Proposal of Software Development Process in a Cooperative of Free Software

Abstract. The use of the development software process in a software industrial is a very important step to get the standardization of software development, thus it allowing to producing software in the pre-established time and price and also take care of the requirements specification. This paper describes the implantation process in a software development cooperative, starting form its requirements, its projects and developer teams. The analysis of some processes that come being used by this cooperative is described, and finally a new software process is proposed.
Keyword: Software Engineering, Software Process, Software Cooperative.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 INTRODUÇÃO

Neste capítulo será descrita uma breve introdução do trabalho, quais foram os motivos da abordagem desse tema e os objetivos que se pretende alcançar com a realização desse trabalho.

1.1 Contextualização e Motivação

Vivemos em um cenário extremamente globalizado e capitalista, onde cada vez mais a busca por um espaço e a competição entre as empresas são mais acirradas. Em se tratando de empresas ou instituições desenvolvedoras de software, este cenário não é diferente, e por esse motivo a busca pelo desenvolvimento de um produto de qualidade e que atenda à expectativa do cliente dentro do prazo e custo estipulado é fundamental. Entre os maiores problemas enfrentados por empresas desse tipo nos últimos tempos estão a dificuldade de se estabelecer uma metodologia de desenvolvimento de software, dificuldade de levantar com clareza os requisitos do software, dificuldade no controle de versão e manutenção do produto, falta de uma documentação, falta de definição de um processo de desenvolvimento de software, entre outros problemas que por conseqüência geram produtos não condizentes com seu propósito, além extrapolarem os custos e prazos previstos.

Segundo Pressman (2002), a falta de adoções de métodos, ferramentas e procedimentos no desenvolvimento de software, resulta em números expressivos de projetos não concluídos, e projetos concluídos que não atendem às necessidades do cliente. Para solucionar este problema, existe uma área do conhecimento da computação denominada engenharia de software que surgiu em meados dos anos 70, propondo a criação e a utilização de sólidos princípios de engenharia, a fim de obter software de maneira econômica e confiável. Os fundamentos científicos para a engenharia de software envolvem o uso de modelos abstratos e precisos que permitem ao engenheiro especificar, projetar, implementar e manter sistemas de software, avaliando e garantindo suas qualidades. Além disto, a engenharia de software deve oferecer mecanismos para se planejar e gerenciar o processo de desenvolvimento. A engenharia de software compreende

diversas áreas específicas, onde podemos enumerar algumas delas:

- Requisitos de software;
- Projeto (Design) do software;
- Construção de software;
- Teste de software;
- Manutenção de software;
- Gerência de configuração de software;
- Gerência de software;
- Processos de software;
- Ferramentas e métodos de software;
- Qualidade de software.

Dentre estas diversas áreas, neste trabalho será focada a área de processo de engenharia de software, ou simplesmente processo de software, que por sua vez compreende diversas atividades, sendo estas relacionadas às áreas da engenharia de software. De acordo com Filho (2003), um processo é um conjunto de passos parcialmente ordenados, constituídos por atividades, métodos, práticas e transformações, usadas para atingir uma meta. Ainda em relação à definição de processo, Reis (2003) afirma que processo de software é um conjunto de atividades realizadas para construir software, levando em consideração os produtos sendo construídos, as pessoas envolvidas, e as ferramentas com as quais trabalham.

Tendo como base as definições de processo e o seu propósito especificado, o estudo em questão refere-se ao estudo e proposta de um processo de desenvolvimento de software em uma cooperativa de software livre denominada TecnoLivre. Como membro atuante da cooperativa e tendo vivenciado algumas dificuldades em relação a falta de padrões nos projetos até então desenvolvidos, tem-se como motivação para este trabalho a falta de um processo de desenvolvimento de software e os reflexos negativos de sua ausência na TecnoLivre.

A TecnoLivre está presente na cidade de Lavras – MG. De acordo com a TecnoLivre (2006), a idéia da criação do grupo nasceu de estudos do Professor José Monserrat Neto (Departamento de Ciência da Computação / Universidade Federal de Lavras) sobre Software Livre e Economia Solidária. Entre os anos de 2005 e 2006, seu

interesse pelas áreas o entusiasmou para conhecer e discutir sobre o assunto em fóruns virtuais e presenciais, como o FISL 7.0 (Fórum Internacional do Software Livre). Ao voltar deste evento, a idéia de criação de um grupo foi apresentada em um colóquio para estudantes de Ciência da Computação da Universidade Federal de Lavras, a qual obteve interesse de alguns dos primeiros membros. Hoje a cooperativa TecnoLivre é composta por dezoito cooperados, sendo todos estudantes do curso de Ciência da Computação da Universidade Federal de Lavras. Além da área de desenvolvimento de software a TecnoLivre atua nas seguintes áreas:

- Desenvolvimento de web sites utilizando CMS's;
- Estudo de viabilidade e implantação de rede de computadores com terminais leves;
- Desenvolvimento de sistemas em geral;
- Treinamentos: Java, PHP, C/C++, Allegro, Linux;
- Migração de software proprietário para software livre.

1.2 Objetivos e Justificativas

Este trabalho objetiva a definição de um processo de desenvolvimento de software que se adeque às características e necessidades da TecnoLivre, de forma a garantir uma maior qualidade do produto desenvolvido e uma maior organização interna. Dessa forma propõe-se o estudo de alguns modelos de processos existentes, afim de avaliá-los e então, dentre estes, escolher o que melhor se enquadre na realidade da TecnoLivre, ou até mesmo formalizar um modelo customizado dentre as várias características dos modelos a serem estudados.

Espera-se que com a definição de um processo consiga-se futuramente aplicá-lo e, assim possibilitar estimar de forma mais precisa os prazos e preços dos produtos desenvolvidos e que se passe a ter um maior controle dos projetos em andamento e de seus envolvidos, justificando assim os objetivos propostos.

1.3 Estrutura do Trabalho

O presente trabalho apresenta primeiramente conceitos e exemplos de processos de software no Capítulo 2, onde serão abordados os processos RUP, PDS2, Praxis e Extreme Programming. Em seguida, no Capítulo 3, será apresentada a metodologia adotada no desenvolvimento deste trabalho e a justificativa pela escolha dos processos de software adotados neste trabalho. No Capítulo 4 têm-se os resultados obtidos, a formalização do processo de software a ser adotado e as justificativas de sua escolha. Por fim encerra-se este trabalho apresentando uma conclusão do trabalho e algumas propostas de trabalhos futuros no Capítulo 5, e no Capítulo 6 têm-se as referências da bibliografia estudada.

2 REVISÃO DE LITERATURA

Neste capítulo serão apresentados conceitos de processo de desenvolvimento de software e em seguida serão exemplificados alguns processos de desenvolvimento de software existentes, sendo estes enquadrados em duas categorias de processos de software: os processos tradicionais e os processos ágeis.

2.1 Processo de Software

Um processo nada mais é que um conjunto ordenado de atividades a fim de atingir um propósito comum, ou seja, podemos definir processo de software como um roteiro que devemos seguir para alcançarmos de forma satisfatória o resultado esperado. Um processo isoladamente não nos diz muita coisa, um processo só se torna aplicável e efetivo quando se aplicam métodos sobre ele, isso quer dizer, quando estabelecemos como elaborar determinada atividade, e os métodos por sua vez se apóiam em ferramentas que irão automatizar ou semi-automatizar o processo e também os métodos.

O processo de software é extremamente importante no cotidiano de uma empresa que desenvolve software, segundo Pressman (2002), o processo de software fornece estabilidade, controle e organização para uma atividade, que se deixada sem controle, pode-se tornar bastante caótica. Um processo de software bem definido apresenta uma série de vantagens. Segundo Humphrey (1990) e Kruchten (2002) citados por Vasconcelos (2005), um bom processo de software garante facilidades nas medições de processo e qualidade de software, propicia uma aquisição de experiências com projetos passados, estabelece uma padronização dos artefatos produzidos, além de facilitar a comunicação entre os membros da equipe.

Para Pressman (2002) um processo, independentemente de ser um processo de software e independentemente do projeto em questão, se resume em três fases:

- A fase de definição é onde se define *o que* deve ser feito, deve-se focar nos objetivos a serem alcançados, quais informações são relevantes para o desenvolvimento, quais as requisições e restrições do projeto e qual a validação

para o seu sucesso.

- Na fase de desenvolvimento focaliza *o como*, define como deve ser implementado o software, como devem ser estruturados os dados, como serão realizados os testes do software.
- A fase de manutenção focaliza as modificações associadas com a correção de erros, as adaptações necessárias à medida que o ambiente de software evolui.

Com o crescente aumento do desenvolvimento de software, surgiram vários modelos de processos, dentre eles podemos definir dois grupos de processo de software, que são os processos tradicionais e os processos ágeis.

O primeiro grupo compreende um tipo de processo mais formal, muitas vezes usado em projetos maiores e que envolve um maior número de pessoas. Este tipo de processo estabelece uma padronização no desenvolvimento dos projetos da empresa. Neste trabalho serão apresentados três tipos de processos tradicionais, sendo eles o RUP, o PDS2 e o PRAXIS.

Os processos ágeis apresentam uma abordagem mais flexível. Este tipo de processo estabelece uma relação mais próxima entre os clientes e os desenvolvedores, sendo esta relação um dos seus principais focos, uma maior comunicação entre os envolvidos no projeto, ao invés de focarem apenas em ferramentas, documentações e contratos como pregam os processos tradicionais. Como exemplo de processos ágeis será feito uma abordagem em torno do processo eXtreme Programming (XP).

A probabilidade de sucesso de um projeto está estritamente ligado ao processo utilizado. Segundo Cockburn (2000) citado por Vasconcelos (2005), um bom processo não deve ser muito pesado, nem muito leve, ou seja, um processo não deve ser muito burocrático, nem muito flexível. Cockburn (2000) mede a probabilidade de sucesso de um projeto de acordo como peso do processo, na Figura 1, pode ser notado que um bom processo se enquadra no meio termo entre processos pesados e processos leves.

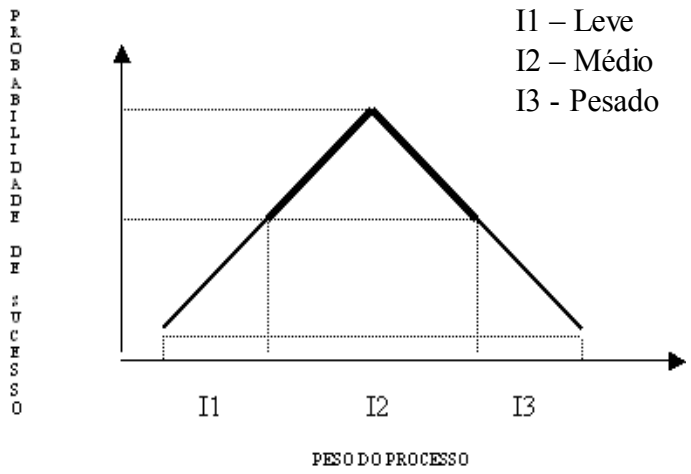

Figura 1: Probabilidade de sucesso de acordo com o peso do processo utilizado no projeto – Cockburn (2000)

2.2 Processos Tradicionais

2.2.1 Processo RUP – Rational Unified Process

O RUP é um processo de desenvolvimento de software desenvolvido pela Rational Software Corporation. Segundo R. Dennis Gibbs (2006), sua origem se deu no ano de 1980, e seu sucesso foi comprovado na aplicação de grandes e complexos projetos. Seu uso principal ocorreu em áreas governamentais, principalmente no Departamento de Defesa dos Estados Unidos. Para dar suporte a essa metodologia muitas ferramentas foram desenvolvidas, como ferramentas para testes, análise e design, requisitos, entre outras. O RUP foi focado em algumas práticas de gerência de projetos, conhecidas como seis melhores práticas, que mais tarde sofreram algumas atualizações feitas pela IBM, que havia adquirido a Rational Software no ano de 2003. As seis e atuais melhores práticas que o RUP prega são as seguintes:

- Adaptar o processo – o processo deve ser adaptado às características de cada projeto específico.

- Balancear a competição entre as prioridades dos *stakeholders* (pessoas envolvidas no projeto).
- Colaboração entre a equipe.
- Demonstrar o valor da iteração.
- Elevado nível de abstração.
- Foco contínuo na qualidade.

A meta do RUP é garantir uma produção de software de alta qualidade que atenda às necessidades dos usuários, dentro de uma programação e um orçamento previsível. O RUP é focado em quatro partes:

- Responsável – define o papel de um indivíduo ou de uma equipe dentro de um contexto do projeto;
- Atividade – representa uma porção de trabalho específica contida no projeto como um todo, é desempenhada por um responsável;
- Artefato – resultado produzido em uma atividade;
- Fluxo – representa a seqüência de atividades a serem executadas, para que se chegue em um propósito comum.

O RUP é dividido em quatro fases, onde cada fase possui o seu ciclo de iteração, suas atividades e geram seus artefatos. Na Figura 2 abaixo serão apresentadas as fases de Iniciação, Elaboração, Construção e Transição presentes no RUP, além de mostrar o grau de presença de cada disciplina em cada fase. Uma disciplina compreende um conjunto atividades em comum.

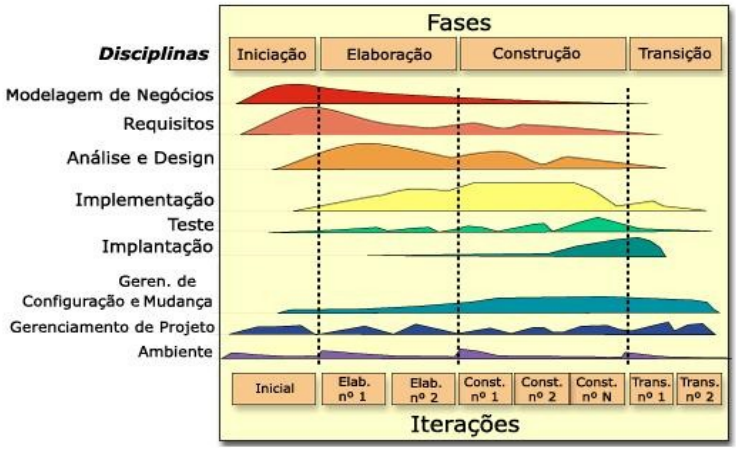

Figura 2: Arquitetura do RUP

Fonte: http://www.wthreex.com/rup/

2.2.1.1 Fase de Concepção

A fase de concepção tem por objetivo principal estabelecer um consenso entre os envolvidos no projeto. Durante esta fase será formalizado o escopo do projeto, bem como o planejamento das atividades e seus envolvidos. É também na fase de concepção que se faz o levantamento dos requisitos de software, determina-se os casos de uso e prevê os possíveis riscos do projeto. Na Tabela 1 abaixo, serão apresentados os artefatos essenciais produzidos nesta fase e seus responsáveis.

Tabela 1: RUP - Fase de Concepção – Artefatos, descrição e responsáveis

Artefatos básicos	**Descrição**	**Responsável**
Visão	Este documento representa a visão dos envolvidos no projeto, representa a base contratual para um maior detalhamento dos requisitos.	Analista de Sistemas
Caso de Negócio	Representa uma avaliação do retorno do produto a ser desenvolvido, sobre a validade de se investir no projeto.	Gerente de Projeto
Lista de Riscos	É elaborada uma lista em ordem decrescente dos riscos do projeto e suas conseqüências no desenvolvimento.	Gerente de Projeto
Plano de Desenvolvimento de Software	Reúne todas as informações necessárias para o gerenciamento do projeto, é representado por um conjunto de artefatos.	Gerente de Projeto
Plano de Iteração	Representa a divisão de um conjunto de atividades e tarefas ao longo do projeto, e os recursos envolvidos.	Gerente de Projeto

Caso de Desenvolvimento	Descreve o processo de desenvolvimento a ser seguido no projeto.	Engenheiro de Processo
Ferramentas	Definição das ferramentas que darão suporte ao desenvolvimento.	Especialista em Ferramentas
Glossário	Definição dos termos importantes mencionados no projeto.	Analista de Sistemas
Modelo de Casos de Uso (Atores, Casos de Uso)	Representa as funcionalidades do sistema e seus atores.	Analista de Sistemas
Repositório do Projeto	O repositório armazena todas as versões de documentos do projeto	Gerente de Configuração
Solicitação de Mudanças	Fornece uma documentação das mudanças efetuadas ao longo do projeto.	Gerente de Controle de Mudanças

Fonte: Adaptação - http://www.wthreex.com/rup/

2.2.1.2 Fase de Elaboração

Na fase de elaboração, será focada a arquitetura do sistema, os requisitos anteriormente levantados serão revistos e aprimorados e o levantamento de riscos será revisado. Esta fase servirá de sustentação à construção do software por meio da criação de protótipos, e da modelagem do sistema e de seus dados. Os artefatos gerados nesta fase serão representados na Tabela 2, nota-se que muitos artefatos produzidos na fase anterior serão revisados e detalhados durante esta fase.

Tabela 2: RUP - Fase de Elaboração - Artefatos, descrição e responsáveis

Artefatos básicos	**Descrição**	**Responsável**
Protótipos	Os protótipos ajudam a prever riscos no projeto, e dão uma visão do produto ao cliente.	
Lista de Riscos	É elaborada uma lista em ordem decrescente dos riscos do projeto e suas conseqüências no desenvolvimento.	Gerente de Projetos
Caso de Desenvolvimento	Descreve o processo de desenvolvimento a ser seguido no projeto.	Engenheiro de Processo
Ferramentas	Definição das ferramentas que darão suporte ao desenvolvimento.	Especialista em Ferramentas
Documento de Arquitetura de Software	Este documento fornece uma visão abrangente da arquitetura do sistema.	Arquiteto de Software
Modelo de *Design*	O modelo de design é uma abstração da implementação do sistema.	Arquiteto de Software
Modelo de Dados	Representação lógica e física dos dados persistentes no sistema, complementa o modelo de *design*.	*Designer* de Banco de Dados
Modelo de Implementação	Representa uma hierarquia de subsistemas de implementação.	Arquiteto de Software
Visão	Este documento representa a visão dos envolvidos no projeto, representa a base contratual para um maior detalhamento dos requisitos.	Analista de Sistemas
Plano de Desenvolvimento de Software	Reúne todas as informações necessárias para o gerenciamento do projeto, é representado por um conjunto de artefatos.	Gerente de Projetos
Guia de *Design* e Guia	Descreve as diretrizes a serem seguidas	Arquiteto de

de Programação	durante o *design* e a programação do sistema.	Software
Plano de Iteração	Representa a divisão de um conjunto de atividades e tarefas ao longo do projeto, e os recursos envolvidos.o	Gerente de Projetos
Modelo de Casos de Uso (Atores, Casos de Uso)	Representa as funcionalidades do sistema e seus atores.	Analista de Sistemas
Especificações Suplementares	Compreende a especificação de requisitos não levantados anteriormente, como requisitos legais, usabilidade, etc.	Analista de Sistemas
Conjunto de Testes	Artefato que agrupa os *scripts* de testes em seqüência de execução.	*Designer* de Teste
Arquitetura para Automatização de Testes	É uma composição de diversos elementos de automatização de testes e suas especificações que representam as características fundamentais do sistema de software de automatização de testes.	*Designer* de Teste

Fonte: Adaptação - http://www.wthreex.com/rup/

2.2.1.3 Fase de Construção

Baseado nos requisitos levantados e na arquitetura a ser utilizada, inicia-se a fase de construção. Durante esta fase, o primordial é o gerenciamento dos recursos e atividades, a fim de minimizar os custos da produção e o tempo de desenvolvimento. Os artefatos resultantes desta fase estão presentes na Tabela 3 abaixo, e assim como na fase anterior alguns artefatos são mencionados novamente.

Tabela 3: RUP - Fase de Construção - Artefatos, descrição e responsáveis

Artefatos básicos	**Descrição**	**Responsável**
"O sistema"	O sistema executável	
Plano de Implantação	Descrição do conjunto de tarefas necessárias para a implantação do sistema	Gerente de Implantação
Modelo de Implementação	Representa uma hierarquia de subsistemas de implementação.	Arquiteto de Software
Conjunto de Testes	Artefato que agrupa os *scripts* de testes em seqüência de execução.	*Designer* de Testes
Materiais de Treinamento	Refere-se aos materiais utilizados no apoio ao aprendizado do cliente final, a fim de manusear de forma adequada o produto desenvolvido.	Desenvolvedor do Curso
Plano de Iteração	Representa as funcionalidades do sistema e seus atores.	Gerente de Projetos.
Modelo de *Design*	O modelo de design é uma abstração da implementação do sistema.	Arquiteto de Software
Caso de Desenvolvimento	Descreve o processo de desenvolvimento a ser seguido no projeto.	Engenheiro de Processos
Ferramentas	Definição das ferramentas que darão suporte ao desenvolvimento.	Especialista em Ferramentas
Modelo de Dados	Representação lógica e física dos dados persistentes no sistema, complementa o modelo de *design*.	*Designer* de Banco de Dados

Fonte: Adaptação - http://www.wthreex.com/rup/

2.2.1.4 Fase de Transição

A fase transição compreende o fechamento do processo de desenvolvimento. Durante esta fase será validada a versão final do produto, a versão beta será revista, baseada no *feedback* do usuário. Além disso, critérios de usabilidade e acessibilidade serão analisados, e o produto estará pronto para ser repassado a terceiros, ou seja, chegar ao consumidor final. Estratégias de marketing serão lançadas, assim como também será traçado um planejamento para suporte ao usuário final.

Tabela 4: RUP - Fase de Transição - Artefatos, descrição e responsáveis

Artefatos básicos	**Descrição**	**Responsável**
O *Build* do Produto	Comercialização do produto, ou seja, fabricação e empacotamento em massa do produto.	Gerente de Implantação
Notas de *Realease*	Identifica mudanças e erros em uma versão do produto.	Gerente de Implantação
Artefatos de Instalação	Documentação necessária para a instalação do produto.	Implementador
Material de Treinamento	Refere-se aos materiais utilizados no apoio ao aprendizado do cliente final, a fim de manusear de forma adequada o produto desenvolvido.	Desenvolvedor do Curso
Material de Suporte para o Usuário Final	Material para auxílio à aprendizagem, utilização, operação e manutenção do produto.	

Fonte: Adaptação - http://www.wthreex.com/rup/

2.2.2 PDS2 - Processo de Desenvolvimento de Software

O PDS2 é um processo de desenvolvimento de software adotado pela DATASUS (2007), que é um órgão da Secretaria Executiva do Ministério da Saúde do Brasil, responsável por coletar, processar e disseminar informações sobre saúde. Este processo é seguido em todos os projetos dirigidos pela DATASUS, assim como os projetos terceirizados pela mesma. O PDS2 é uma versão aprimorada do PDS, que foi o primeiro processo de desenvolvimento de software criado pela DATASUS no ano de 2002. O PDS é um processo baseado no RUP, e ele surgiu devido à necessidade da DATASUS para padronizar os sistemas desenvolvidos por terceiros, e os sistemas desenvolvidos pela própria DATASUS. A sua adoção se deve também ao fato de ser um processo mais aplicável em projetos menores e que envolvem um menor número de participantes. Com o passar do tempo alterações foram feitas no PDS, de modo que o processo se encaixasse melhor na realidade da DATASUS, tornando o PDS2 a versão atual adotada pela DATASUS.

Assim como o RUP, o PDS2 é baseado em quatro elementos básicos, diferindo apenas na sua menor complexidade e detalhamento das atividades, assim como dos artefatos produzidos. Sendo estes elementos:

- Papéis – quem faz;
- Artefatos – o que;
- Atividades – como;
- Fluxo de Atividades – quando.

Segundo a DATASUS (2007), algumas práticas de desenvolvimento são seguidas ao longo do projeto, como o desenvolvimento iterativo e incremental, onde pequenos módulos são planejados, executados e testados/corrigidos de forma isolada ao invés de executar cada um desses passos em apenas um momento do desenvolvimento. Tais práticas garantem avaliar os riscos do projeto previamente, evitando que o projeto seja comprometido ao seu final. Além destas, outras práticas são seguidas, como:

- Gerência dos requisitos – levantamento dos requisitos acordados com o cliente;
- Gerência de configuração – controle das versões do sistema;

- Gerência de mudanças – avaliação e formalização das mudanças a serrem ocorridas;
- Participação ativa dos usuários – usuário representa o cliente, e o seu envolvimento no projeto é de extreme importância;
- Melhoria contínua do processo de desenvolvimento – o processo deve ser sempre condizente com realidade da empresa, acompanhando suas evoluções e mudanças.

A Figura 3, sintetiza a organização do processo PDS2. O PDS2 também é estruturado em quatro fases (Concepção, Elaboração, Construção e Transição), onde o final de cada fase representa um marco no processo. A descrição de cada fase compreende os mesmos conceitos apresentados pelo RUP. As figuras a seguir irão demonstrar as atividades de cada fase, e as tabelas irão especificar os artefatos de cada fase, suas descrições e responsáveis.

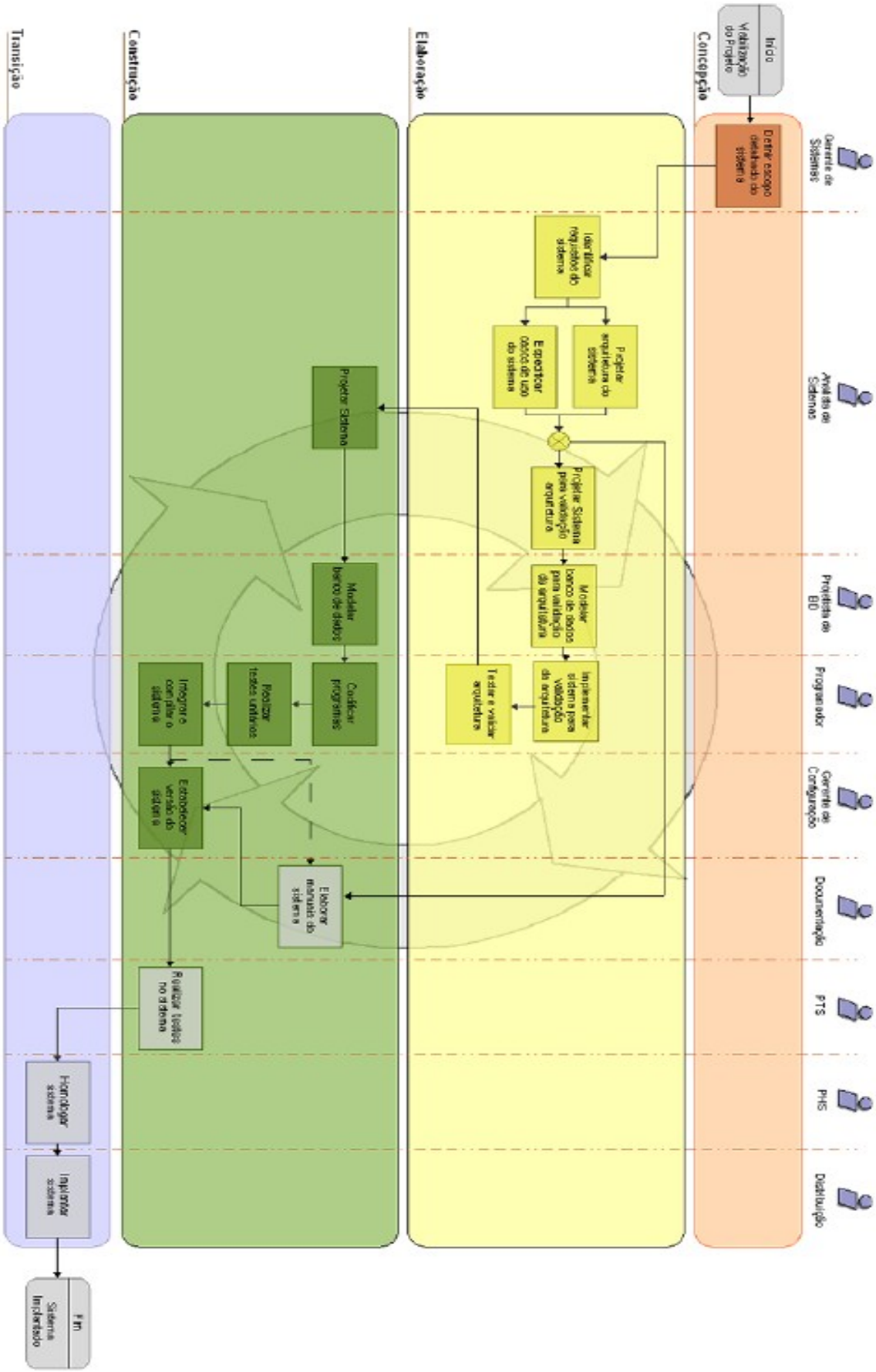

Figura 3: Arquitetura do PDS2

Fonte: http://pds.datasus.gov.br

2.2.2.1 Fase de Concepção

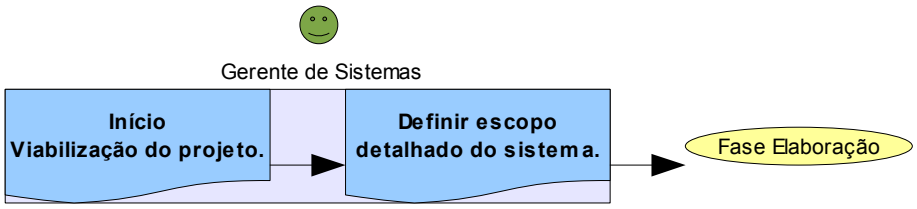

Figura 4: Fase de Concepção

Fonte: Adaptação – http://pds.datasus.gov.br

- Viabilização do projeto: representa toda parte de negociação entre a empresa prestadora do serviço e o cliente; esta área está mais voltada para a parte de gerência de projetos, do que à parte de processo de software propriamente dita.
- Definir escopo detalhado do sistema: esta atividade descreve todo o objetivo e funcionalidades do sistema, assim como suas fronteiras, deixando de forma clara o que deve e o que não deve ser feito. O escopo também estabelece o cronograma de atividades previstas, seus prazos e seus respectivos responsáveis.

Tabela 5: PDS2 - Fase de Concepção - Artefatos, descrição e responsáveis

Artefatos básicos	Descrição	Responsável
Documento de Consenso do Produto	Este documento representa um acordo formal entre as partes do projeto, contendo as funcionalidades do produto e suas delimitações.	Gerente de Sistemas
Glossário	Este documento define os termos do projeto, tornando assim claro a sua compreensão pelas partes do projeto.	Analista de Sistemas

Fonte: Adaptação – http://pds.datasus.gov.br

2.2.2.2 Fase de Elaboração

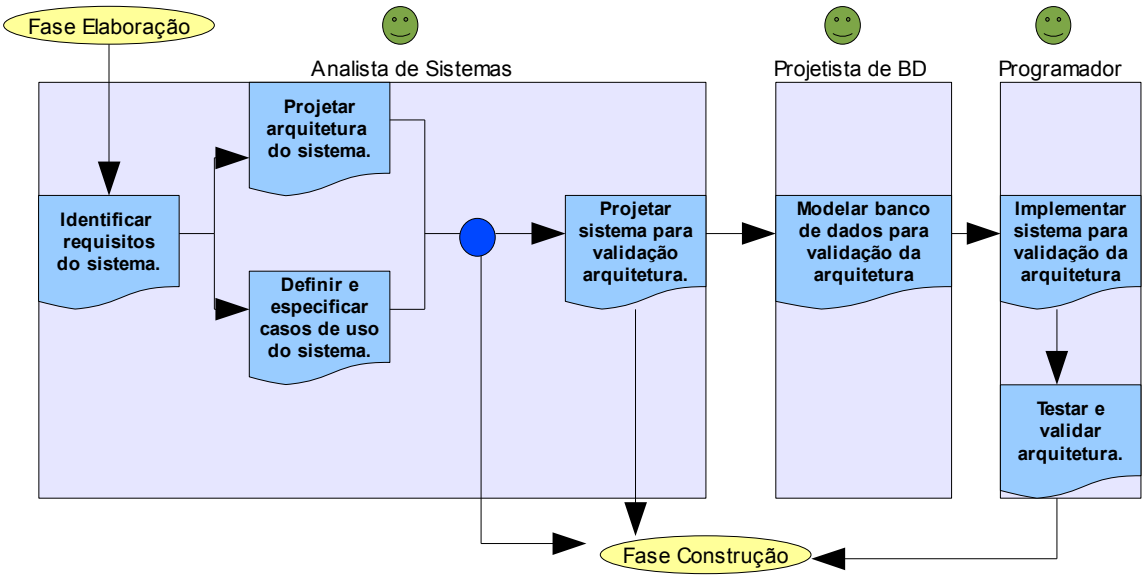

Figura 5: Fase de Elaboração

Fonte: Adaptação – http://pds.datasus.gov.br

- Identificar requisitos do sistema: um requisito representa uma condição ou capacidade que o sistema precisa atender. Os requisitos são levantados baseados no escopo do projeto e em entrevistas com o cliente, sendo estes esboçados numa matriz de requisitos e em alguns diagramas. Estes requisitos serão classificados quanto ao grau de prioridade e quanto suas funcionalidades, se são requisitos funcionais, não-funcionais, informativos ou de regras de negócio.
- Projetar a arquitetura do sistema: projetar a arquitetura do sistema, define como o sistema será implementado, decisões referente à linguagem de programação, sistema gerenciador de banco de dados, ferramentas a serem utilizadas, necessidades de hardware e software, entre outras. Além disso, consiste em uma representação do sistema por meio de diagramas, definindo o sistema de forma estática ou dinâmica (em funcionamento).
- Especificar casos de uso do sistema: os casos de usos especificam a interação do

usuário com o sistema, definindo assim os atores do sistema, ou seja, qual funcionalidade será atribuída para cada ator. Além disso, são especificadas as descrições de cada funcionalidade, suas pré e pós-condições.

- Projetar sistema para a validação da arquitetura: nesta atividade os analistas de sistemas irão verificar se os requisitos identificados estão de acordo com o cenário previsto, a fim de validar esta parte do projeto.
- Modelar banco de dados para validação da arquitetura: nesta atividade o projetista de banco de dados irá modelar o sistema do banco de dados, baseado no cenário definido o no diagrama de classes criado.
- Implementar sistema para a validação da arquitetura: com base nos documentos e diagramas anteriores, os desenvolvedores irão criar um versão executável do sistema, esta é uma versão intermediária.
- Testar e validar a arquitetura: esta fase envolve programadores do sistema, participação de analistas do sistema e dos usuários do sistema, afim de testar o sistema, se este está de acordo com os requisitos especificados e por fim validar o sistema, caso o mesmo passe pelos testes estipulados.

Tabela 6: PDS2 - Fase de Elaboração - Artefatos, descrição e responsáveis

Artefatos básicos	**Descrição**	**Responsável**
Matriz de Requisitos	Este documento contém todos os requisitos do sistema.	Analista de Sistemas
Modelo de Caso de Uso	Este documento registra todos os atores do sistema e suas funcionalidades.	Analista de Sistemas
Documento de Arquitetura	Este documento define toda a infra-estrutura de hardware e software do sistema.	Analista de Sistemas
Diagrama de Classes	Representação estática do sistema, contendo casos de uso e suas relações.	Analista de Sistemas.
Modelo do Banco de	Representa o modelo do banco de dados,	Projetista de Banco de

Dados	incluindo diagramas, usuários, permissões, dicionário de dados.	Dados
Versão do Sistema	Representa a validação do sistema ao final da fase.	Gerente de Configuração
Glossário	Este documento define os termos do projeto, tornando assim claro a sua compreensão pelas partes do projeto.	Analista de Sistemas

Fonte: Adaptação – http://pds.datasus.gov.br

2.2.2.3 Fase de Construção

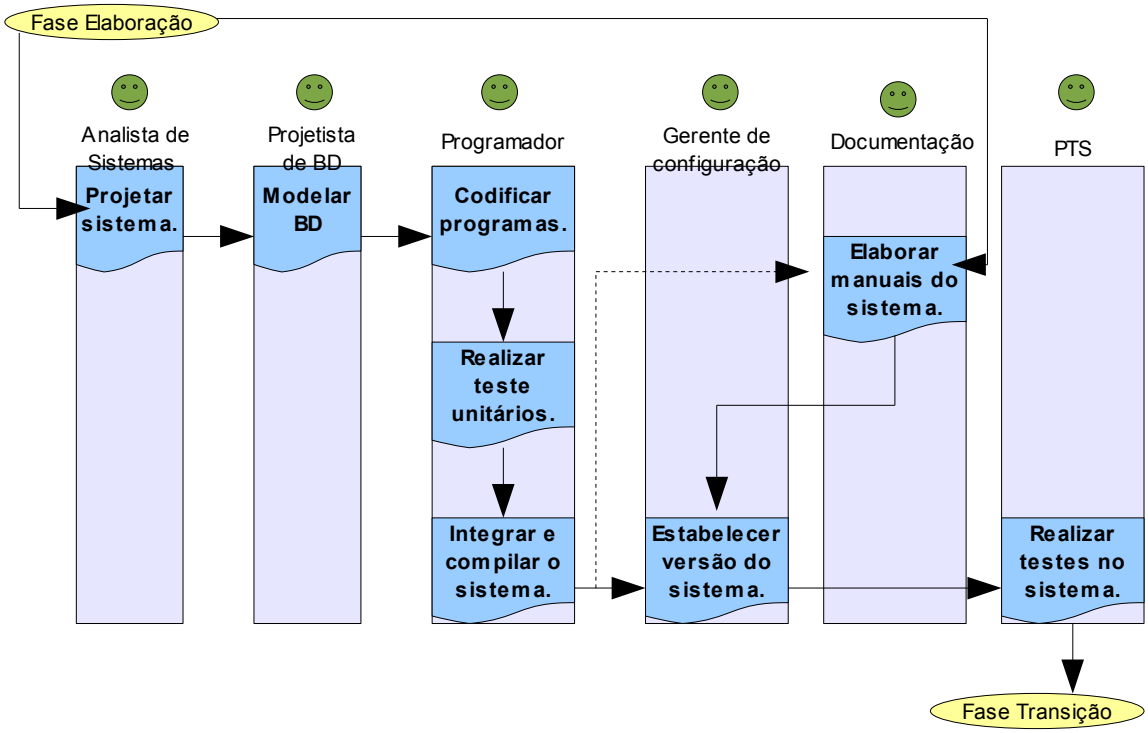

Figura 6: Fase de Construção

Fonte: Adaptação – http://pds.datasus.gov.br

- Projetar sistema: nesta fase os analistas irão validar o projeto de sistema proposto, os diagramas de caso de uso, diagramas de classes já foram elaborados durante a

fase de elaboração.

- Modelar banco de dados: será concluído a modelagem feita durante a fase de elaboração.
- Codificar programas: os programadores irão gerar os códigos fontes do sistema, *scripts*, *triggers* de banco de dados e *stored procedures*, com base na implementação intermediária da fase de elaboração.
- Realizar testes unitários: os testes unitários verificam se cada elemento do sistema implementado está de acordo com os requisitos especificados.
- Integrar e compilar o sistema: uma vez realizado os testes unitários, o sistema será integrado e compilado como um todo.
- Elaborar manuais do sistema: o manual tem por objetivo especificar os processos de instalação, uso, operação e administração do sistema para os usuários finais.
- Estabelecer versão do sistema: o gerente de configurações irá estabelecer a versão do sistema com a identificação de todos os itens de configuração do projeto que a compõem.
- Realizar testes no sistema: o PDS segue um padrão de testes próprio, que é o PTS, além de possuir um ambiente dedicado aos testes de softwares. Após realizados estes testes, têm-se a primeira versão operacional do sistema.

Tabela 7: PDS2 - Fase de Construção - Artefatos, descrição e responsáveis

Artefatos básicos	**Descrição**	**Responsável**
Diagrama de Classes	Representação estática do sistema, contendo casos de uso e suas relações.	Analista de Sistemas
Modelo do Banco de Dados	Representa o modelo do banco de dados, incluindo diagramas, usuários, permissões, dicionário de dados.	Projetista de Banco de Dados
Documentação de Usuário	Manuais, *releases* notes, guias.	
Versão do Sistema	Representa a validação do sistema ao final	Gerente de

	da fase.	Configuração
Pacote de Distribuição	Representa o empacotamento físico de uma versão do sistema.	Gerente de Configuração

Fonte: Adaptação – http://pds.datasus.gov.br

2.2.2.4 Fase de Transição

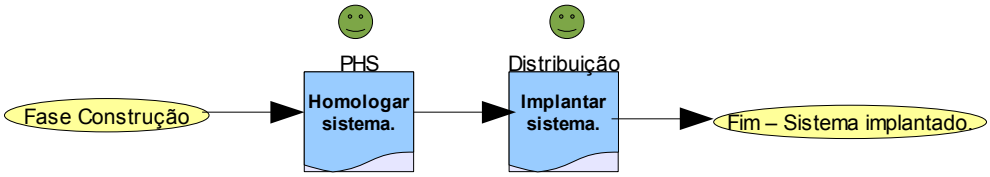

Figura 7: Fase de Transição

Fonte: Adaptação – http://pds.datasus.gov.br

- Homologar sistema: esse processo visa a verificar a conformidade do sistema com as normas e padrões estabelecidas no processo. O DATASUS utiliza-se do PHS (processo de homologação de software) para garantir a qualidade do produto.
- Implantar sistemas: tem por objetivo colocar o sistema em funcionamento em seu ambiente de operação.

Tabela 8: PDS2 - Fase de Transição - Artefatos, descrição e responsáveis

Artefatos básicos	**Descrição**	**Responsável**
Versão do Sistema	Versão do sistema homologada.	Gerente de Configuração
Pacote de Distribuição	Pacote de distribuição final.	Gerente de Configuração

Fonte: Adaptação – http://pds.datasus.gov.br

2.2.3 O Processo Praxis

O processo Praxis, Processo para Aplicativos eXtensíveis InterativoS, foi projetado com o intuito de apoiar projetos didáticos, como por exemplo em disciplinas envolvendo engenharia de software. Com esse propósito, o Praxis visa a atender projetos individuais e ou projetos com equipes menores, e que tenham duração entre 6 meses e um ano. Baseado nesse critério de aplicação do processo Praxis, considerou-se relevante o estudo desse projeto, uma vez que o presente trabalho é voltado para a implantação de um processo de software em uma cooperativa que possui características de projetos típicas às propostas pelo Praxis. Apesar de ser um processo voltado para área didática, esse processo pode ser perfeitamente adaptado à área comercial, podendo assim ser adotado por organizações, desde que feita suas adequações à realidade de cada organização. A versão a ser abordada do processo Praxis, será a versão Praxis 2.0, ao ser mencionado a palavra Praxis, considere se tratar do Praxis 2,0.

Assim como os demais processos apresentados, o Praxis é dividido em fases de Concepção, Elaboração, Construção e Transição. Segundo Filho (2003), cada fase é composta por iterações, onde estas iterações representam passos presentes em cada uma dessas fases. Assim como as fases (divisões orientadas para gestão do projeto), o Praxis possui fluxos (divisões orientadas por disciplinas da engenharia de software) que por sua vez são divididos em atividades. Ao final de cada passo do processo, artefatos são gerados. O Praxis classifica três tipos de artefatos, sendo eles modelo, documento, e relatório, nas Tabelas 9, 10 e 11, serão detalhados cada um desses artefatos, e na Tabela 12 serão relacionados as fases, com suas iterações e seus artefatos. Uma das diferenças do Praxis para os demais processo apresentados, é em relação ao seu ciclo de vida e a não definição de papéis durante o projeto. O ciclo de vida do Praxis é um ciclo de vida de entrega evolutiva. Diferentemente de um ciclo de vida iterativo e incremental, como dos demais processos estudados, o ciclo de vida de entrega evolutiva compreende o levantamento de todos os requisitos e toda a análise do sistema em uma iteração, propondo após, um desenho inicial do sistema, que será avaliado pelos usuários do sistema, até que se chegue em um desenho detalhado do sistema, e então inicia-se a fase de implementação.

Tabela 9: Documentos oficiais do Praxis

Nome	Sigla	Descrição
Proposta de Especificação do Software	PESw	Documento que delimita preliminarmente o escopo de um projeto, contendo um plano da fase de Elaboração.
Especificação dos Requisitos do Software	ERSW	Documento que descreve, de forma detalhada, o conjunto de requisitos especificados para um produto de software.
Plano de Desenvolvimento do Software	PDSw	Documento que descreve, de forma detalhada, os compromissos que o fornecedor assume em relação ao projeto quanto a recursos, prazos, riscos e outros aspectos gerenciais.
Plano de Qualidade do Software	PQSw	Documento que descreve, de forma detalhada, os procedimentos de garantia da qualidade que serão adotados no projeto.
Descrição do Desenho do Software	DDSw	Documento que escreve de forma detalhada, os aspectos mais importantes do desenho do software.
Descrição dos Testes do Software	DTWs	Documento que descreve, de forma detalhada, os planos e as especificações dos testes que serão executados.
Manual do Usuário do Software	MUSw	Documento que serve de referência para o uso do produto.

Fonte: Filho (2003) – Engenharia de Software – Fundamentos, Métodos e Padrões.

Tabela 10: Modelos oficiais do Praxis

Nome	Sigla	Descrição	Ferramentas aplicáveis
Cadastro dos Requisitos do Software	CRSw	Modelo que contém os requisitos levantados, assim como referências aos itens correspondentes dos modelos seguintes.	Planilha, banco de dados.

Modelo de Análise do Software	MASw	Modelo que detalha os conceitos do domínio do problema a resolver que sejam relevantes para a validação dos requisitos.	Ferramenta de modelagem orientada a objetos.
Memória de Planejamento do Projeto do Software.	MPPSw	Modelo que contém a informação necessária para o planejamento e o acompanhamento de tamanhos, esforços, custos, prazos e riscos do projeto.	Planilha, ferramenta de gestão de projetos.
Modelo de Desenho do Software.	MDSw	Modelo que detalha a estrutura lógica e física do produto, em termos de seus componentes.	Ferramenta de modelagem orientada a objetos.
Bateria de testes de Regressão do Software.	BTRSw	Conjunto dos scripts dos testes de regressão.	Ferramenta de desenvolvimento, ferramentas de testes.
Códigos Fontes do Software.	CFSw	Conjunto dos códigos fontes produzidos.	Ferramenta de desenvolvimento.
Códigos Executáveis do Software.	CESw	Conjunto dos códigos executáveis produzidos.	Ferramenta de desenvolvimento.

Fonte: Filho (2003) – Engenharia de Software – Fundamentos, Métodos e Padrões.

Tabela 11: Relatórios oficiais do Praxis

Nome	**Sigla**	**Descrição**	**Responsável**
Relatórios dos Testes de Software.	RTSw	Relatório que descreve os resultados dos testes realizados.	Grupo de testes do projeto.
Relatórios de Revisão do Software.	RRSw	Relatório que descreve as conclusões da inspeção de um artefato.	Grupo revisor do artefato.

Relatórios de Inspeção do Software.	RISw	Relatório que descreve as conclusões da inspeção de um artefato.	Grupo inspetor do artefato.
Relatórios das Auditorias da Qualidade do Software.	RAQSw	Relatório que descreve as conclusões de uma auditoria da qualidade realizada.	Grupo de Garantia da Qualidade.
Relatórios de Acompanhamento do Projeto do Software.	RAPSw	Relatório que descreve esforços, custos, prazos e riscos do projeto até a data corrente.	Gerente do projeto.
Relatório Final do Projeto do Software.	RFPSw	Relatório de balanço final do projeto.	Gerente do projeto

Fonte: Filho (2003) – Engenharia de Software – Fundamentos, Métodos e Padrões.

Tabela 12: Praxis - Relações entre fases, iterações, artefatos e suas descrições.

Fase	**Iteração**	**Artefatos**	**Descrição**
Concepção	Ativação	PESw, MASw, RAPSw	Levantamento e análise das necessidades dos usuários e conceitos da aplicação, em nível de detalhe suficiente para justificar a especificação de um produto de software.
Elaboração	Levantamento dos Requisitos	ERSw, MASw, CRSw, PDSw, RAPSw	Levantamento das funções, interfaces e requisitos não-funcionais desejados para o produto.
	Análise dos Requisitos	ERSw, MASw, CRSw, DDSw, MDSw, PDSw, MPPSw, PQSw, RAPSw,	Modelagem conceitual dos elementos relevantes do domínio do problema e uso desse modelo para validação dos requisitos e planejamento detalhado da fase de Construção.

		RAQSw, RRSw	
Construção	Desenho Implementável	DDSw, MDSw, DTSw, RTSw, CRSw, PDSw, PQSw, CFSw, CESw, RAPSw, RAQSw, RRSw, RISw, MUSw	Definições interna e externa dos componentes de um produto de software, em nível suficiente para decidir as principais questões de arquitetura e tecnologia e para permitir o planejamento detalhado das liberações.
	Liberação 1	DDSw, MDSw, DTSw, BTRSw, RTSw, CRSw, CFSw, CESw, RAPSw, RAQSw, RISw, MUSw	Implementação de um subconjunto de funções do produto que será avaliado pelos usuários.
	Liberação ...	Idem.	Idem.
	Testes Alfa	DDSw, MDSw, DTSw, BTESw, RTSw, CRSw, CFSw, CESw, MUSw, RAPSw, RRSw, RIRw	Realização dos testes de aceitação, no ambiente dos desenvolvedores, juntamente com elaboração da documentação de usuário e possíveis planos de Transição.
Transição	Teste Beta	RTSw, RAPSw, RAQSw, RISw	Realização dos testes de aceitação no ambiente dos usuários.
	Operação Piloto	MUSw, RFPSw, RAQSw, RRSw	Operação experimental do produto em instalação piloto do cliente, com a resolução de eventuais problemas através de processo de manutenção.

Fonte: Adaptação de Filho (2003) – Engenharia de Software – Fundamentos, Métodos e Padrões.

2.3 Processos Ágeis

2.3.1 XP – eXtreme Programming

Os processos de desenvolvimento de software tidos como ágeis surgiram devido ao fato de que o desenvolvimento de software por meio de processos tradicionais tornou-se muito burocrático, e uma grande parte do esforço no desenvolvimento estava focado na parte de documentação e contratos, causando assim um grande aumento no tempo de desenvolvimento do projeto. As metodologias ágeis não são contra as práticas tradicionais, e nem mesmo que não sejam importantes para o projeto, no entanto, de acordo com Wells (2002) seu foco está principalmente na relação entre as pessoas do projeto, e na valorização de seus talentos. Esta metodologia surgiu por meio de um grupo de dezessete pessoas, que sentiram necessidade de criar uma metodologia alternativa para desenvolvimento de software. Ao grupo formado por essas pessoas se deu o nome de Aliança Ágil. Para eles uma metodologia ágil prega quatro valores principais:

- Indivíduos e interações acima de processos e ferramentas.
- Software funcionando acima de documentação abrangente.
- Colaboração com o cliente acima de documentação abrangente.
- Responder a mudanças acima de seguir um plano.

O Extreme Programming é uma das metodologias mais difundidas se tratando de metodologias ágeis. Segundo Wells (2002) sua primeira aplicação se deu em um projeto chamado C3 Payroll, dirigido pela empresa Chrysler, seu sucesso fez despertar a curiosidade de outros pesquisadores da área, e logo então outras empresas já estavam adotando esta metodologia como parte dos seus projetos. O XP garante muitas vantagens para um projeto, considerado um processo do tipo leve, ele permite atender seus clientes em um prazo mais curto, além de focar sempre nos indivíduos do projeto e contar com a participação efetiva do cliente no desenvolvimento do projeto.

O XP possui doze práticas que constitui o núcleo principal do processo, segundo

Kent Beck, citado por Vasconcelos (2005), estas práticas não são novidades, mas sim práticas que já vem sendo utilizadas há muitos anos, com eficiência em projetos de software. As doze práticas do XP, são descritas abaixo:

- Testes constantes – no XP existem dois tipos de testes, os testes de unidade e os testes funcionais, os teste de unidade são automaticamente executados assim que o programador escreve o código, e os testes funcionais são executados juntamente com o cliente para verificar a conformidade entre os requisitos e o funcionamento do sistema.
- Refatoramento – o refatoramento é uma técnica que visa a promover melhorias no código do sistema, tornado-o mais legível e otimizado, mas sem alterar as funcionalidades do sistema.
- Programação em pares – a implementação do sistema ocorre aos pares, enquanto uma pessoa fica responsável por codificar e pensar nos algoritmos, a outra pessoa fica observando, tentando encontrar melhorias no código, tornando-o mais simples.
- Propriedade coletiva do código – a propriedade coletiva torna a equipe mais unida, e mais focada em seus objetivos. Além de respeitarem padrões de codificação, a integridade do código pode ser garantida com o uso de ferramentas de controle de versão.
- Integração contínua – A integração é realizada constantemente, assim que uma parte é implementada e testada, ela será integrada ao sistema.
- Semana de quarenta horas – o XP não prega um carga horária fixa, no entanto é importante obedecer alguns limites, não afetando assim a produtividade da equipe.
- Cliente no local – uma pessoa por parte do cliente será integrada a equipe, e participará do desenvolvimento do sistema, respondendo as perguntas e dúvidas. Caso não se tenha uma pessoa por parte do cliente, pelo menos deve-se ter alguém com conhecimento do negócio.
- Padrões de codificação – é essencial que se obedeça certos padrões de codificação, uma vez que o XP realiza programação aos pares.
- O jogo do planejamento – o planejamento é constante no XP, defini-se o escopo de releases do sistema, que são implementadas, e novos planejamentos para novas releases serão estudados. Existe a participação de duas partes pelo lado do cliente,

uma pessoa que compreenda o negócio e a pessoa que atuará no sistema.

- Releases pequenas – uma release deve ser o quanto menor possível, atendendo aos requisitos mais importantes do sistema, isso garante um maior feedback para o cliente.
- Metáfora – uma metáfora consiste em fazer uma analogia entre o sistema que está sendo desenvolvido, e um outro sistema qualquer, não necessariamente um software, a fim de tornar mais compreensível o seu funcionamento por parte dos desenvolvedores e cliente.
- Projeto simples – deve-se atender as funcionalidades principais do sistema, e se focar nas necessidades imediatas do sistema, e não se preocupar em atender requisitos futuros, uma vez que os requisitos são inconstantes e podem se alterar a qualquer momento.

2.3.2 Fases e Ciclo de Vida de um Processo XP

O ciclo de vida de um processo XP é muito curto comparado aos processos tradicionais, de acordo com Oshiro (2001) citado por Bona (2002), essa questão em relação ao ciclo de vida do XP é muito discutida entre diversos autores da área, no entanto o XP é um processo que se adequa à projetos onde se têm requisitos muito instáveis, quando necessita-se rever requisitos que foram previstos no início do projeto, ou mesmo acrescentar novos requisitos durante o decorrer do projeto. O XP compreende as fases a seguir:

- Exploração – nesta fase deve-se entender o propósito do sistema, de acordo com Bona (2002) o cliente deve relatar o sistema na forma de histórias, e os programadores por sua vez irão estimar as histórias contadas. Segundo Wake (2002) citado por Bona (2002) as histórias devem ser quebradas em histórias menores, isso diminui os riscos do projeto. Enquanto o cliente relata as histórias, os programadores devem avaliar possíveis arquiteturas para o sistema, isso o auxiliará nas decisões de implementação diante das histórias apresentadas pelo cliente.

- Planejamento – o objetivo da fase de planejamento é definir o menor cronograma possível e o maior número de histórias. Segundo Bona (2002) deve-se em primeiro momento selecionar as histórias essenciais para a primeira versão do sistema. O cliente irá avaliar as historias de acordo com o grau de prioridade e os programadores de acordo com os riscos apresentados. As histórias serão divididas em tarefas e alocadas entre os programadores, que por sua vez irão estimar o tempo previsto de cada tarefa de acordo com a experiência dos mesmos.
- Iteração para a primeira versão – o sistema é desenvolvido ao longo de várias iterações, em cada iteração serão realizadas as tarefas estipuladas, e ao final de cada iteração serão realizados os testes. Segundo Bona (2002) na primeira iteração será testada a arquitetura previamente definida, por essa razão convêm-se que sejam escolhidas um número de tarefas mínimas que consigam representar o sistema como um todo. Ao final das iterações o cliente terá testado todo o sistema e o mesmo estará pronto para ser produzido.
- Produção – a produção inicia-se quando encerra-se o feedback por parte do cliente, no entanto, nesta fase novos testes podem ser feitos pela equipe desenvolvedora para validar realmente a funcionalidade do sistema.
- Manutenção - diferentemente de outros processos, esta fase no XP é normalmente executada dentro do ciclo de vida de um projeto. Durante esta fase novas funcionalidades podem ser incorporadas ao sistema, pode-se questionar pela atualização da tecnologia anteriormente adotada, e é natural que ocorra uma mudança na equipe do projeto, devido ao fato de que agora não serão necessários tantos programadores. É importante ressaltar que o sistema já está em funcionamento, as novas funcionalidades serão desenvolvidas de forma paralela.
- Fim do projeto – o projeto chega ao fim quando o cliente não tem mais histórias para relatar, nesse momento é importante que a equipe desenvolvedora do projeto elabore um documento relatando as funcionalidades do sistema, e algumas informações básicas, caso seja necessário fazer algumas alterações no sistema futuramente. O ponto principal é que se tenha conseguido satisfazer as expectativas do cliente, e também é necessário que a equipe se reúna, para avaliar o trabalho realizado, e assim adquirir novos aprendizados para trabalhos futuros.

3 METODOLOGIA

Este trabalho baseou-se no estudo comparativo de alguns processos de software, foram analisadas as características presentes em cada processo estudado, podendo dessa forma fazer uma filtragem das principais características observadas, e então estar propondo um modelo de processo de software que atenda às principais necessidades do grupo em questão. O tipo de pesquisa utilizada se classifica como aplicada quanto a sua natureza, uma vez que se tem como finalidade a aquisição de um processo de desenvolvimento de software, e que o mesmo possa ser aplicado na cooperativa em questão. Segundo Jung (2004), uma pesquisa aplicada ou tecnológica se caracteriza por uma pesquisa que gera produtos ou processos além de conhecimentos com finalidade imediata.

Quanto aos seus objetivos, trata-se de uma pesquisa de caráter exploratório, pois visa à proposição de um novo processo de desenvolvimento de software, a partir de modelos de processos já existentes, e que serão analisados. Ainda segundo Jung (2004), uma pesquisa de caráter exploratório busca a descoberta de novas teorias e práticas que modifique as existentes. Não necessita de grandes teorizações e sim de experimentação para coleta de dados.

A pesquisa baseou-se em referências bibliográficas e pela Internet, foram referenciados livros, artigos e páginas da internet contendo modelos de processo de software existentes. Foram analisados quatro tipos de processos de software além de conceitos em torno dessa área. Decidiu-se pela escolha desses processos, uma vez que temos o RUP como um dos processos mais difundidos na área, o PDS2 é uma sintetização do RUP, voltado para projetos menores e equipes menores, justificando assim a sua abordagem nesse trabalho, uma vez que a cooperativa em questão possui características semelhantes aos projetos que o PDS2 aborda. O PRAXIS também está voltado para o desenvolvimento de projetos menores, onde se têm equipes menores, além de apresentar algumas características diferentes dos demais, como seu ciclo de vida, e a não definição de papéis dentro do projeto, sendo esses os motivos da sua escolha para esse trabalho. Dentre os processos ágeis, o XP é um dos mais difundidos, e por se tratar de um processo diferente dos demais apresentados nesse trabalho, por se enquadrar dentro de uma metodologia ágil de desenvolvimento, decidiu-se pela sua abordagem nesse trabalho.

Estudado estes quatro processos, foi realizada uma apresentação dos mesmos para a

cooperativa TecnoLivre. Na apresentação que teve uma duração de uma hora apresentou-se as características principais de cada processo, além de um quadro comparativo entre os processos, permitindo assim uma maior distinção entre os mesmos. Ao final foi aberto um espaço para esclarecimento de dúvidas, que puderam ser esclarecidas. Uma vez tendo realizado a apresentação, o grupo foi submetido a um questionário, que objetivou coletar de cada participante sua opinião em relação aos processos apresentados e em relação ao processo ideal na visão de cada um para a TecnoLivre. Obtido esses resultados, foi proposto o modelo de software para a cooperativa, que futuramente será avaliado de uma forma prática.

4 RESULTADOS E DISCUSSÃO

Neste capítulo serão apresentados os resultados alcançados, será feita uma discussão em torno dos mesmos, e a definição de uma proposta de processo de software será lançada.

4.1 Apresentação dos Resultados

Os resultados a serem apresentados foram obtidos mediante o estudo de quatro processos de software anteriormente apresentados. A definição do processo de software a ser incorporado à TecnoLivre, teve como base a opinião das pessoas participantes do grupo da TecnoLivre e das necessidades observadas dentro do grupo. Foi apresentado ao grupo as características e os conceitos dos quatro processos estudados, durante uma apresentação realizada junto ao grupo, além de um estudo comparativo dos 4 processos, que pode ser observado na Tabela 13. Neste estudo comparativo analisou-se a complexidade de cada processo, no que diz respeito ao número de atividades presentes, número de artefatos gerados e em relação ao ciclo vida do processo. Obervou-se com isso que alguns processos são mais aplicáveis a depender do tipo de projeto e equipe em questão.

Tabela 13: Comparação de processos de software

	RUP	**PDS2**	**Praxis**	**XP**
Processo	Tradicional	Tradicional	Tradicional	Ágil
Papéis e atividades	Define 30 papéis e atividades específicas correspondentes.	Define 9 papéis e atividades correspondentes.	Define atividades, mas não define papéis.	Define 7 papéis, mas sem atribuições específicas dentro das atividades.
Ferramentas	Software próprio (privado).	Não especifica.	Não especifica.	Não especifica.
Artefatos	27 artefatos básicos ao	12 artefatos ao longo das 4	20 artefatos ao longo das 4	Apenas o código fonte.

	longo das 4 fases* do processo.	fases* do processo.	fases* do processo.	
Características de projeto e equipe	Projetos maiores e equipes maiores.	Simplificação do RUP – projetos menores e equipes menores.	Projetos menores e equipes menores (duração de 6 meses a 1 ano)	Equipes menores, e projetos onde os requisitos são instáveis.
Ciclo de vida	Iterativo e incremental	Iterativo e incremental	Entrega evolutiva	Iterativo e incremental

*Fases de Concepção, Elaboração, Construção e Transição

Para complementar os resultados, o grupo da TecnoLivre foi submetido à um questionário, que visou a absorver dos entrevistados as principais características, na opinião deles, presentes em um processo de software, tendo como base de resposta as realidades da Tecnolivre, de seus projetos e de suas equipes. Foram entrevistadas 18 pessoas. O questionário pode ser visto no Anexo A desse trabalho e os resultados obtidos podem ser conferidos a seguir:

Questionário Avaliativo de Processos de Software – Resultados

1 – Formação – área de atuação na Universidade

Tabela 14: Questionário - Questão1

Questão 1	
Professor	1
Graduado(a) em Ciência da Computação	2
Graduando(a) em Ciência da Comoutação	15
Outras áreas	0

2 – Você já cursou a disciplina de Engenharia de Software?

Tabela 15: Questionário - Questão2

Questão 2	
Sim	13
Não	5

3 – Como você classificaria o seu conhecimento da área de Processo de Software?

Tabela 16: Questionário - Questão3

Questão 3	
Excelente	0
Alto	2
Médio	7
Baixo	7
Nenhum	2

4 – Você já utilizou algum processo de software durante algum desenvolvimento de software?

Tabela 17: Questionário - Questão4

Questão 4	
Sim	6
Não	12

5 – Você já utilizou alguma ferramenta *CASE* ao desenvolver algum software?

Tabela 18: Questionário - Questão5

Questão 5	
Sim	14
Não	4

6 – Enumere as sentenças de acordo com as principais características de um bom processo.

Tabela 19: Questionário - Questão6

Questão 6	
Entrada/saída de atividades definidas	0
Responsabilidades definidas	4
Modelo de ciclo de vida adequado ao projeto	4
Fácil de ser utilizado	5
Produzir um volume de documentação adequado ao projeto	5
Outras áreas	0

7 – Classifique a importância da documentação de um projeto, incluindo documentação de requisitos, casos de usos, diagramas, entre outros.

Tabela 20: Questionário - Questão7

Questão 7	
Muito importante	10
Importante	8
Razoável	0
Pouco importante	0
Sem importância	0

8 – Em relação às práticas do XP, caso considere importante, enumere a(s) que considere importante(s) e aplicável(eis) ao processo, de acordo com o grau de prioridade.

Tabela 21: Questionário - Questão8

Questão 8	
Testes constantes	4
Refatoramento	1
Programação em pares	4
Propriedade coletiva do código	1
Integração contínua	0
Semana de quarenta horas	0
Cliente no local	1

Padrões de Codificação	6
Jogo de planejamento	0
Releases pequenas	0
Metáfora	0
Projeto simples	1

9 – Em relação ao ciclo de vida de um processo, marque a opção que melhor se encaixaria ao processo da TecnoLivre na sua opinião.

Tabela 22: Questionário - Questão9

Questão 9	
Iterativo e incremental	7
Seqüencial	2
Entrega evolutiva	9

10 – Em relação aos processos, apresentados, enumere-os de acordo com sua aderência ao perfil da TecnoLivre.

Tabela 23: Questionário - Questão10

Questão 10	
RUP	0
PDS2	5
Praxis	4
XP	9

Baseado nos resultados apresentados, foi proposto o modelo de processo de software apresentado a seguir. Suas definições e as justificativas dos resultados apresentados também serão discutidas a seguir.

4.2 O Processo Proposto

O processo proposto apresenta suas características de acordo com os resultados observados no questionário aplicado ao grupo da TecnoLivre, e também de acordo com as características consideradas importantes dentro de um processo de software. Analisando os resultados do questionário, observa-se a o desejo de se ter um processo simples e objetivo, mas por outro lado que seja também focado em documentação e na definição do papel de cada pessoa dentro do projeto. A princípio se poderia considerar contraditório o resultado alcançado, onde se têm como o processo que mais se adere a TecnoLivre, o eXtreme Programming, e também tendo como resposta dos entrevistados, a documentação como um fator muito importante em um processo, mas essa questão apenas enfatiza o desejo de se ter um processo focado em seus objetivos, que seja fácil de ser utilizado, que valorize a integração entre a equipe e os clientes, que priorize o desenvolvimento do que for essencial ao sistema e que além de todas essas características que são típicas de um processo XP, preze pela organização do projeto, de forma a definir as atividades, seus responsáveis, seus artefatos e que tudo isso seja documentado ao longo do projeto. Pode-se observar também que das pessoas entrevistadas, todas atuam na área de computação, a maioria possui uma certa familiarização com área de engenharia de software, sendo que destas, algumas possuem conhecimento da área de processo de software. A avaliação do grupo de pessoas a serem questionadas, quanto ao conhecimento da área em que está focado o trabalho é importante para validação dos resultados, o que neste caso prova que o grupo questionado apresenta um conhecimento razoável do tema abordado.

Assim como apresentado na Figura 8, o processo proposto é focado em quatro fatores essenciais: as atividades, os artefatos, os papéis e o fluxo. Esse último nada mais é que o ciclo de vida do processo. Cada item desses fatores serão descritos e explicados a seguir:

- **Atividades**

  - Viabilização do Projeto – a viabilização do projeto compreende toda parte de negociação entre o cliente a empresa a desenvolver o projeto, inclui-se como parte da viabilização do projeto a proposta comercial, o contrato, entre outros documentos a vir a ser considerados essenciais. A viabilização do projeto não se inclui no processo em si, está é uma questão que faz parte da gerência de projetos.

  - Análise de Requisitos – Essa atividade é responsável pelo levantamento de todos os requisitos necessários ao sistema, os requisitos serão levantados mediante entrevistas junto ao cliente e/ou demais pessoas envolvidas na utilização do sistema a ser desenvolvido. Os requisitos compreendem requisitos funcionais e não funcionais, e podem ser classificados como essenciais, importantes ou desejados. Com o levantamento de todos os requisitos deve-se também definir o escopo do sistema.

  - Diagrama de Caso de Uso – O diagrama de casos de uso identifica a interação do usuário com o sistema, deve-se descrever cada caso de uso, os seus atores, as pré-condições, pós-condições e exceções.

  - Definição da Arquitetura – Deve-se descrever a arquitetura a ser utilizada, incluindo descrições detalhadas dos diversos mecanismos de arquitetura de software adotados.

  - Diagrama de Classes – O diagrama de classes descreve as classes do sistema, seus métodos e atributos, além da relação entre as classes, descrevendo também a divisão das camadas do sistema.

  - .Modelagem do Banco de Dados – identificar e modelar tabelas do banco de dados, seus campos, relacionamentos, permissões, chaves primárias.

  - Implementar Banco de Dados – Representa a concretização do modelo do banco de dados proposto, de forma a interagir com a aplicação a ser desenvolvida.

  - Codificar Programa – Codificar com base nos requisitos levantados e na

análise feita.

- Testes (Equipe de testes) – Testar a versão gerada na fase de implementação e verificar sua conformidade com os requisitos levantados, esse teste é realizado pela própria equipe participante do desenvolvimento do projeto.

- Testes (Junto ao cliente) – Após realizados os testes pela equipe, uma versão deve ser mostrada ao cliente, testada, e posteriormente aprovada, para então e iniciar a fase de implantação.

- Implantação – Esta atividade tem o objetivo de colocar o sistema funcionando em seu local de operação, atendendo todos os requisitos não-funcionais previstos anteriormente, e as definições de arquitetura levantadas.

- **Artefatos**

- Viabilização do Projeto – Compreende os documentos gerados durante a fase de negociação do projeto até a firmação do contrato.

- Documento de Requisitos - Documento com a descrição dos requisitos, sua classificação, seu grau de importância e a descrição do escopo do sistema.

- Diagrama de Caso de Uso – Diagrama UML de casos de uso.

- Documento de Arquitetura – Documento detalhado das definições de arquitetura adotadas.

- Diagrama de Classes – Diagrama UML de classes.

- Modelagem do Banco de Dados – Diagrama de modelagem do banco de dados.

- Banco de Dados Implementado – Criação do banco de dados dentro do Sistema Gerenciador de Banco de Dados estabelecido.

- Código Fonte – Representa a entrega do código fonte gerado pela atividade de implementação

- Relatório de Testes (Equipe de testes) – Relatório descrevendo os passos de testes realizados, resultados obtidos e solicitações de modificações que forem necessárias. Esses testes serão realizados no código fonte gerado, que deverá ser compilado, e deverá gerar uma versão executável do programa.

- Relatório de Testes (Junto ao cliente) - Relatório descrevendo os passos de testes realizados, resultados obtidos e solicitações de modificações que forem necessárias. Esses testes serão realizados em cima da versão executável gerada durante a atividade de testes realizada pela equipe de testes.

- Versão Intermediária (executável) – Resultado da atividade de testes feita pela equipe de testes, desde que atendido todos os critérios de testes previstos no teste junto à equipe de teste.

- Versão Final – Resultado da atividade de testes feita junto ao cliente, desde que atendido todos os critérios de testes previstos no teste junto ao cliente.

- Homologação do sistema – Documento descrevendo o encerramento do projeto, de acordo com as cláusulas contratuais previstas, e em conformidade com o propósito inicial previsto.

- **Papéis**

- Gerente de Projetos – Responsável por toda parte de negociação, planejamento e controle do projeto

- Analista de Sistemas – Responsável por toda parte de análise do projeto

- Projetista de Banco de Dados – Responsável por toda atividade relacionada à área de banco de dados presente no desenvolvimento do projeto.

- Programador – Responsável por realizar a implementação do modelo de análise proposto.

- Equipe de Teste – Responsável por realizar os testes internos e os testes

junto ao cliente, elaborando um relatório dos resultados encontrados.

- Equipe de Implantação – Responsável pela implantação do sistema.

- **Fluxo**

O fluxo de atividades representa a ordem cronológica de execução de cada atividade, respeitando a satisfação de atividades anteriormente previstas. Analisando esse conceito, entende-se por fluxo de atividades, o ciclo de vida do software. De acordo com os processos apresentados, dois modelos de ciclo de vida foram apresentados, sendo eles o modelo iterativo e incremental e o modelo de entrega evolutiva. Para o modelo proposto decidiu-se por adotar um ciclo de vida intermediário entre os dois modelos apresentados, uma vez que a princípio pretendia-se usar o modelo de entrega evolutiva, por ser esse o modelo pretendido de acordo com o questionário aplicado, no entanto observou-se alguns pontos insatisfatórios no modelo, como por exemplo o fato de considerar os requisitos como se fossem estáticos ao longo do processo. Sendo assim o ciclo de vida do modelo proposto segue seus passos da seguinte forma:

- Levantamento de todos os requisitos e realização de toda a análise possível inicialmente;
- Realizado o item anterior, inicia-se a implementação referente à análise anteriormente realizada;
- Serão realizados os testes iniciais, caso não sejam aprovados, existe a possibilidade de retornar à fase de implementação para correção de erros, ou até mesmo à fase de requisitos para solucionar algum tipo de requisito insatisfatório, conseqüentemente repetindo todo o ciclo novamente;
- Aprovado os testes iniciais, serão realizados os testes junto ao cliente, com possibilidade de retornar aos testes iniciais, à fase de implementação ou à fase de requisitos;
- Realizados todos os testes, inicia-se a fase de implantação, podendo ainda retornar

à fase de requisitos, caso algum tipo de requisito não seja atendido.

O processo proposto além de possuir as características acima apresentadas, define algumas boas práticas a serem seguidas, práticas que são adotadas pelo processo XP e, de acordo com o resultado do questionário, foram consideradas importantes em um processo de software. Sendo assim define-se como práticas a serem seguidas:

- Programação em pares – Essa prática está pressente durante a atividade de implementação, e consiste em realizar a programação aos pares, enquanto um pessoa codifica a outra fica responsável por verificar a conformidade do código, além de apresentar possíveis soluções para o problema através de idéias.
- Padrões de codificação – Estabelecer um padrão de codificação garante o entendimento do código com uma maior clareza, permitindo assim que várias pessoas tenham posse do código, e mesmo assim mantendo um padrão comum para todos da equipe.
- Testes constantes – A prática de testes constantes, consiste na revisão de cada atividade realizada, podendo dessa forma evitar problemas futuros e então diminuir os riscos do projeto.

Resumindo o modelo proposto, a Tabela 24 descreve o processo de acordo com o número de itens que o constitui e algumas de suas características importantes.

Tabela 24: Características do Modelo de Processo Proposto

	Modelo de Processo Proposto
Processo	Tradicional.
Papéis e atividades	Define 6 papéis e as atividades específicas correspondentes.
Ferramentas	Não especificado.
Artefatos	13 artefatos.
Características de projeto e equipe	Projetos menores e equipes menores.
Ciclo de vida	Modelo intermediário entre o ciclo de vida de entrega evolutiva e o ciclo de vida iterativo e incremental.

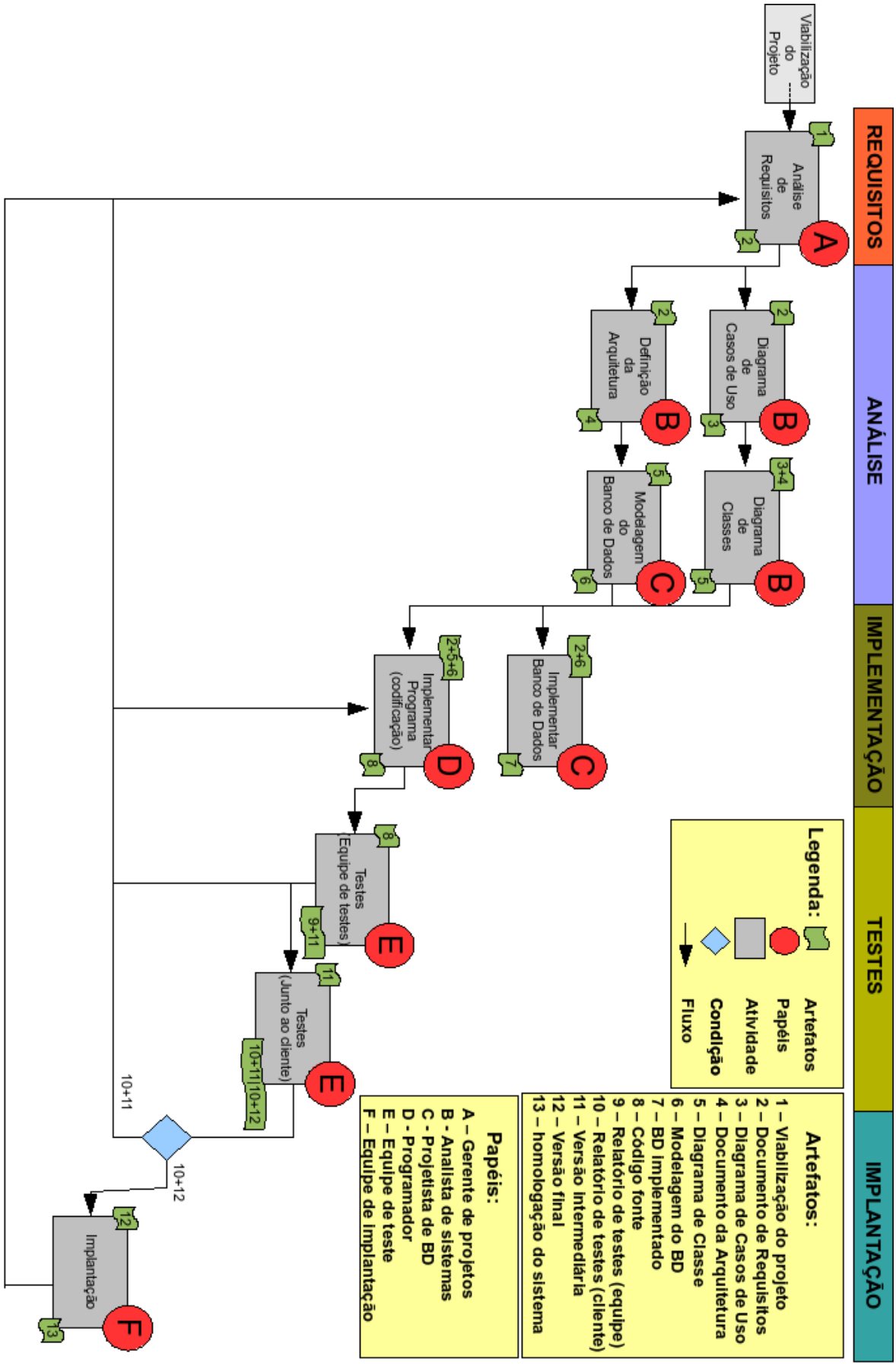

Figura 8: Processo Proposto

5 CONCLUSÕES E TRABALHOS FUTUROS

Neste capítulo serão apresentadas as conclusões alcançadas mediante aos objetivos propostos e também algumas proposta de trabalhos futuros que poderão aprimorar e incrementar o trabalho a ser apresentado.

5.1 Conclusões

Uma vez analisado alguns processos de software, entre tradicionais e ágeis, e com a participação de um grupo de pessoas pertencentes à cooperativa Tecnolivre, conseguiu-se definir um processo de software que fosse baseado na opinião dessas pessoas no que se refere processo de software. O fato de se ter um modelo de processo que foi construído a partir das próprias pessoas que irão utilizá-lo é um ponto importante para aderência e aceitação do mesmo, podendo assim reforçar uma característica essencial de um bom processo de software: ser um processo que é seguido e respeitado pelas pessoas que o utilizam, de forma a organizar e padronizar o desenvolvimento de seus trabalhos.

Alcançou-se um modelo de processo teórico, ou seja, ainda não é aplicado na prática. Esse foi o primeiro passo de muitos que poderão surgir, para que assim sejam atingidos os objetivos futuros, que é a real organização e padronização dos projetos da TecnoLivre. Ainda assim, considera-se realizado o objetivo proposto, que foi a definição de um processo de software, podendo o mesmo ser melhor avaliado após sua aplicação em um projeto que vir a surgir.

5.2 Trabalhos Futuros

Por ser um trabalho pioneiro dentro do grupo da TecnoLivre na área de engenharia de software, existem várias propostas de trabalhos futuros, sendo que algumas já se encontram em fase de inicial. De imediato têm-se como meta a utilização do processo proposto, e sua avaliação no projeto aplicado. Outras idéias podem surgir envolvendo esse trabalho, como a do estudo de ferramentas de apoio ao processo, de ferramentas para

análise, implementação, testes, e demais atividades do processo. Uma outra proposta consiste em desenvolver uma ferramenta *web* de suporte ao processo, de forma a apresentar o modelo do processo, descrição de suas atividades, artefatos, papéis e fluxo, além de disponibilizar *templates* dos artefatos gerados. Isso auxiliaria a equipe do projeto, sendo uma referência a ser seguida ao longo do projeto.

Como pode-se observar, a gama de possibilidades de trabalhos é ampla. Além dos citados, considera-se importante também o estudo e a viabilização de uma ferramenta para a gerência do projeto, podendo através da mesma gerenciar o cronograma do projeto, que englobaria as atividades, seus responsáveis, além de poder gerenciar várias outras áreas integrantes da gerência de projetos. Como foco inicial de trabalho futuro, está a aplicação do processo a um projeto real, sendo os demais trabalhos possibilidades interessantes e importantes para que possam complementar o trabalho apresentado.

ANEXO A - Questionário Avaliativo de Processos de Software

Questionário Avaliativo de Processos de Software

O presente questionário objetiva coletar informações referente às características de um processo de software, tendo como base a apresentação de quatro processos de software (RUP, PDS2, Praxis e XP) apresentada junto ao grupo em questão, permitindo assim a avaliação das necessidades observadas pelo grupo e o auxílio na formalização de um processo de desenvolvimento de software para a TecnoLivre.

Observações: Deve-se considerar que estas informações referem-se à um processo para ser aplicado na TecnoLivre, então considere-se como base de resposta as realidades da Tecnolivre, de seus projetos e das equipes pertencentes aos projetos.

1) Formação – área de atuação na Universidade:
( ) Professor(a) ( ) Graduado(a) em Ciência da Computação ( ) Graduando(a) em Ciência da Computação
( ) Outras áreas

2) Você já cursou a disciplina de Engenharia de Software?
( ) Sim ( ) Não

3) Como você classificaria o seu conhecimento da área de Processo de Software?
( ) Excelente ( ) Alto ( ) Médio ( ) Baixo ( ) Nenhum

4) Você já utilizou algum processo de software durante algum desenvolvimento de software?
( ) Sim ( ) Não

5) Você já utilizou alguma ferramenta *case* ao desenvolver algum software?

( ) Sim ( ) Não

Qual(s):_____

_____

6) Enumere as sentenças de acordo com as principais características de um bom processo:

( ) Entrada/saída de atividades definidas.

( ) Responsabilidades definidas.

( ) Modelo de ciclo de vida adequado ao projeto.

( ) Fácil de ser utilizado.

( ) Produzir um volume de documentação adequado ao projeto

( ) Outra _____

7) Classifique a importância da documentação de um projeto, incluindo documentação dos requisitos, casos de uso, diagramas, entre outros.

( ) Muito importante ( ) Importante ( ) Razoável ( ) Pouco importante ( ) Sem importância

8) Em relação às práticas do XP, caso considere importante, enumere a(s) que considere importante(s) e aplicável(s) ao processo, de acordo com o grau de prioridade:

( ) Testes constantes

( ) Refatoramento

( ) Programação em pares

( ) Propriedade coletiva do código

( ) Integração contínua

( ) Semana de quarenta horas

( ) Cliente no local

( ) Padrões de codificação

( ) Jogo de planejamento

( ) Releases pequenas

( ) Metáfora

( ) Projeto simples

9) Em relação ao ciclo de vida de um processo, marque a opção que melhor se encaixaria ao processo da TecnoLivre na sua opinião:
( ) Iterativo e incremental (RUP, PDS2, XP) ( ) Sequencial ( ) Entrega evolutiva (Praxis)

10) Em relação aos processos apresentados, enumere-os de acordo com sua aderência ao perfil da TecnoLivre:
( ) RUP ( ) PDS2 ( ) Praxis ( ) XP

6 REFERÊNCIAS BIBLIOGRÁFICAS

ANDRADE J., Avaliação de Processos de Software em Ambiente de Desenvolvimento de Software Orientado à Organização, 2005, disponível em <http://ramses.cos.ufrj.br/taba/index.php?option=com_docman&task=doc_view&gid=251>, acessado em 11/05/2007.

BONA C., Avaliação de Processos de Software: Um Estudo de Caso em XP e Iconix, 2002, disponível em: <http://teses.eps.ufsc.br/defesa/pdf/10816.pdf>, acessado em 11/05/2007.

CAMPELO R., XP-CMM2: Um Guia para Utilização de Extreme Programming em um Ambiente Nível 2 do CMM, disponível em: < http://www.di.ufpe.br/~hermano/download/dissertacoes/XP-CMM2.pdf>, acessado em 15/03/2007.

DATASUS, PDS-DATASUS v2.0 Processo de Desenvolvimento de Software Guia Completo 01/2007, disponível em: <http://pds.datasus.gov.br>, acessado em 02/03/2007.

FILHO W., Engenharia de Software, Fundamentos, métodos e padrões, 2ª edição 2003, LTC-Livros Técnicos e Científicos Editora S.A.

JAVAFREE.ORG, Obtendo qualidade de software com o RUP, disponível em: <http://www.javafree.org/content/view.jf?idContent=7>, acessado em 27/01/2007.

JUNG, C. F. . Metodologia para Pesquisa & Desenvolvimento: Aplicada a Novas Tecnologias, Produtos e Processos. 1. ed. Rio de Janeiro: Axcel Books do Brasil Ltda, 2004. v. 1. 328 p.

PRAXIS, Processo Praxis, disponível em: < http://www.wppf.uaivip.com.br/praxis>, acessado em 05/03/2007.

PRESSMAN R., Engenharia de Software, 5ª edição 2002, Mc Graw Hill.

R DENNIS GIBBS, Project Management with IBM Rational Unified Process: Lessons from the Trenches, IBM Press 2006.

REIS C., Caracterização de um modelo de processo para projetos de software livre, disponível em: <http://www.async.com.br/~kiko/quali_pres/>, acessado em 15/01/2007.

RUP – Rational Unified Process:Visão Geral, disponível em: < http://www.wthreex.com/rup/>, acessado em 25/01/2007.

TECNOLIVRE – Cooperativa de Tecnologia e Soluções Livres, disponível em: <http://www.tecnolivre.ufla.br>, acessado em 26/06/2007.

VASCONCELOS A., Processo de Desenvolvimento de Software 1, Textos Acadêmicos UFLA/FAEPE, 2005.