KAREN YUKO FUCHIGAMI

PROPOSTA DE APLICAÇÃO DO MODELO PSP NO DEPARTAMENTO DE SOFTWARE DE UMA INDÚSTRIA ALIMENTÍCIA

Monografia apresentada ao PECE – Programa de Educação Continuada em Engenharia da Escola Politécnica da Universidade de São Paulo como parte dos requisitos para conclusão do curso de MBA em Tecnologia de Software.

São Paulo
2014

KAREN YUKO FUCHIGAMI

PROPOSTA DE APLICAÇÃO DO MODELO PSP NO DEPARTAMENTO DE SOFTWARE DE UMA INDÚSTRIA ALIMENTÍCIA

Monografia apresentada ao PECE – Programa de Educação Continuada em Engenharia da Escola Politécnica da Universidade de São Paulo como parte dos requisitos para a conclusão do curso de MBA em Tecnologia de Software.

Área de Concentração: Tecnologia de Software

Orientador: Prof. Dr. Mauro de Mesquita Spinola

São Paulo
2014

DEDICATÓRIA

Dedico este trabalho aos meus pais Minoru e Edna que sempre incentivaram os meus estudos, ao meu irmão André que me apoiou em todos os momentos da minha vida e ao meu namorado Ricardo que esteve ao meu lado me dando forças para alcançar os meus objetivos.

AGRADECIMENTOS

À Universidade de São Paulo – USP, a Escola Politécnica da Universidade de São Paulo – EPUSP e ao PECE – Programa de Educação Continuada em Engenharia que me deram a oportunidade de receber o aprendizado de mestres e doudores.

RESUMO

O trabalho apresenta um plano de implantação para aplicação do framework PSP (Personal Software Process) em uma empresa brasileira do ramo de produção e exportação de proteína animal. A empresa apresenta 20% de retrabalho ao mês por problemas criados no desenvolvimento do sistema, para diminuir o retrabalho, o PSP aumenta a produtividade dos desenvolvedores e a precisão nas estimativas de projetos de desenvolvimento de software, melhorando a qualidade do código desenvolvido no tempo planejado. O trabalho mostra o contexto da empresa para que seja possível identificar o cenário e os problemas existentes, em seguida complementa com a fundamentação teórica, para que no fim seja exibido o plano de implantação a ser apresentada para a empresa.

Palavras Chave: PSP, Personal Software Process, Produtividade, Desenvolvimento, CMMI-DEV, Qualidade, Estimativa.

ABSTRACT

This paper presents a deployment plan to apply PSP (Personal Software Process) framework on a animal protein productor and exporter Brazilian company. The company presents rework of 20% per month due to software problems. To reduce rework, PSP increases developer productivity and precision estimates. Resulting in a high quality software completed on time. This paper shows the company context to understand the problems, and then a deployment plan is presented based on theoretical foundation.

Key words: PSP, Personal Software Process, Produtividade, Development, CMMI-DEV, Quality, Precision on estimation.

LISTA DE ILUSTRAÇÕES

Pág.

Figura 1 - Processos da Empresa (Fonte: Empresa, 2014) 14
Figura 2 - Entrada e Saída de Projetos (Fonte: Empresa, 2014) 15
Figura 3 - CMMI for Dev (CMMI Product Team) 17
Figura 4 - Restrição Tripla do Gerenciamento de Projetos (Fonte: elaborado pela autora) 19
Figura 5 - Fases do PSP (HUMPHREY, 2000) 27
Figura 6 - Estrutura do PSP (HUMPHREY, 2000) 29
Figura 7 - *Project Plan Summary* (HUMPHREY, 2000) 32
Figura 8 - Método PROBE (HUMPHREY, 2000) 33
Figura 9 - Programa que armazena as tarefas que ocorreram no dia (Fonte: Empresa, 2014) 39
Figura 10 - Erros categorizados (Fonte: Empresa, 2014) 40
Figura 11 - Lista de projetos (Fonte: Empresa, 2014) 41
Figura 12 - Fórum do projeto (Fonte: Empresa, 2014) 42

LISTA DE TABELAS

Pág.

Quadro 1 – Script de Processo (HUMPHREY, 2000).....32

Quadro 2 – Cronograma Proposto para Plano de Implantação (Fonte: elaborado pela autora).....43

LISTA DE ABREVIATURAS E SIGLAS

CMMI	Capability Maturity Model Integration
PSP	Personal Software Process
TSP	Team Software Process
PMBOK	Project Management Body of Knowledge

SUMÁRIO

1. INTRODUÇÃO 13
1.1 Motivações 13
1.2 Objetivo 16
1.3 Justificativas 16
1.4 Estrutura do Trabalho 18
2. REVISÃO BIBLIOGRÁFICA 19
2.1 Escopo 20
2.1.1 Pesquisa 21
2.1.2 Criatividade 21
2.1.3 Documentos 22
2.1.4 Observação 22
2.1.5 Apoio 23
2.2 Tempo 23
2.2.1 Análise Matemática 23
2.2.2 Compreensão da duração 24
2.2.3 Simulações 24
2.2.4 Nivelamento heurístico dos recursos 24
2.3 Custo 24
2.4 PSP 25
2.4.1 Etapas do PSP 26
2.4.2 Estrutura do PSP 29
2.4.3 Casos de Uso Aplicados em Outros Cenários 35
3. PROPOSTA 37
3.1 PSP0 38
3.2 PSP0.1 38
3.2.1 Relatório de Log do Tempo Gasto 38
3.2.2 Relatório de Log dos Defeitos Encontrados 40
3.3 PSP1 40
3.4 PSP1.1 40
3.5 PSP2 41
3.6 PSP2.1 42
3.7 PSP3 42
3.8 Plano de Implantação 43
3.8.1 Cronograma 43
3.8.2 Responsabilidades 44
5. CONSIDERAÇÕES FINAIS 47

5.1 Contribuições do Trabalho 47
5.2 Trabalhos Futuros 48

1. INTRODUÇÃO

Este capítulo apresenta as motivações que levaram a criação desse trabalho, seu objetivo, as justificativas e por fim a estrutura do trabalho.

1.1 Motivações

Esse estudo está baseado na melhoria do processo de desenvolvimento de uma empresa nacional de produção e exportação de proteína animal. A empresa atua no ramo de processamento de carne bovina, ovina, suína e de frango. Estando presente também no mercado de couro, higiene e limpeza, colágeno, embalagens metálicas, biodiesel, entre outros produtos. É uma empresa multinacional que pode ser encontrada em 24 (vinte quatro) países espalhados nos cinco continentes, atende mais de 300 mil clientes e possui mais de 185 mil funcionários..

Todos os processos da empresa são controlados por um sistema ERP (Enterprise Resource Planning). O sistema foi criado internamente e é evoluído e mantido até hoje. Utilizado por todos os departamentos, o sistema é dividido em quatro grandes módulos, sendo eles: Administrativo, Comercial, Fábrica e Financeiro. Todos os processos desde o abate do boi até o faturamento do pedido comercial passam pelo sistema. Na imagem abaixo é possível ter uma visão macro desses processos.

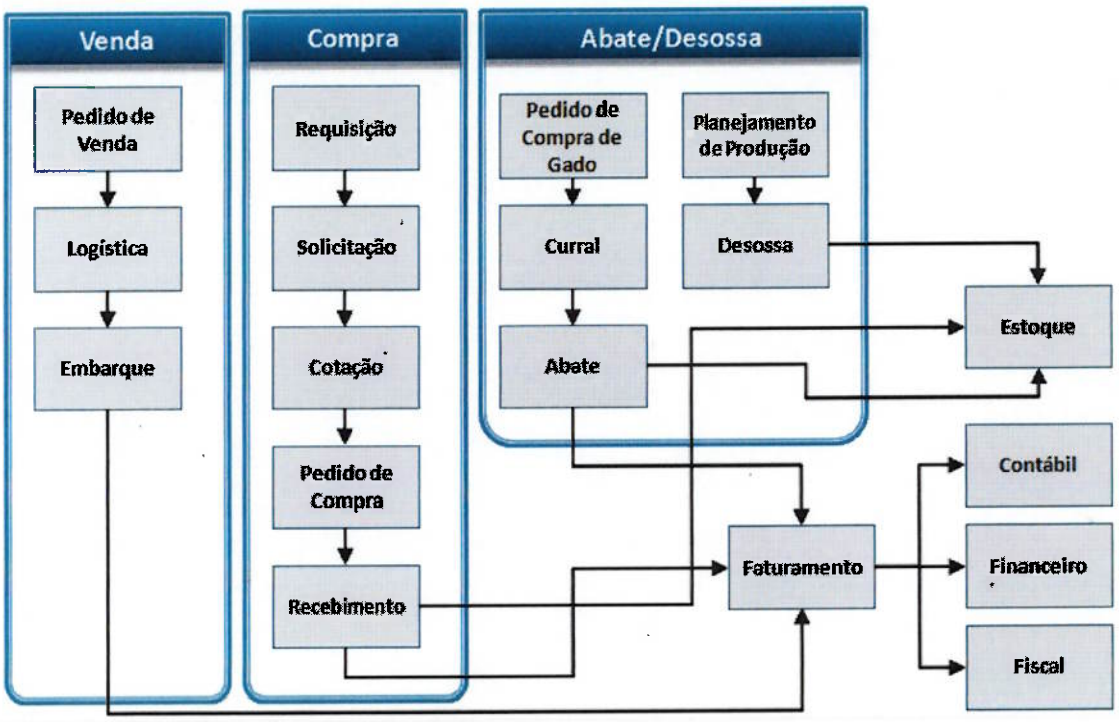

Figura 1 - Processos da Empresa (Fonte: Empresa, 2014)

Na imagem é possível observar que todos os Processos da Empresa são interligados e a falha em qualquer um deles pode interferir em todo o sistema.

O sistema ERP possui 1910 telas que são utilizadas por cerca de 9.200 usuários da empresa, gerando por volta de 15.000 transações de dados por segundo.

A empresa tem um crescimento muito acelerado. Em apenas 61 anos, conseguiu chegar hoje a capacidade de abate de cerca de 90 mil cabeças de gado, 12 milhões de aves, 75 mil suínos, 25 mil carneiros e 13 milhões de peles por dia. Do último trimestre de 2013 até o primeiro trimestre de 2014 teve a receita líquida com aumento de 24,6% e 28,9% no lucro líquido. Devido à velocidade de crescimento da empresa e pelo estilo "agressivo" das tomadas de decisão, o sistema deve acompanhar as estratégias da empresa. Toda semana é gerada uma nova versão do sistema com melhorias e correções. Em média, são solicitados 224 projetos de clientes para alterar o sistema e são entregues por volta de 244 projetos por mês.

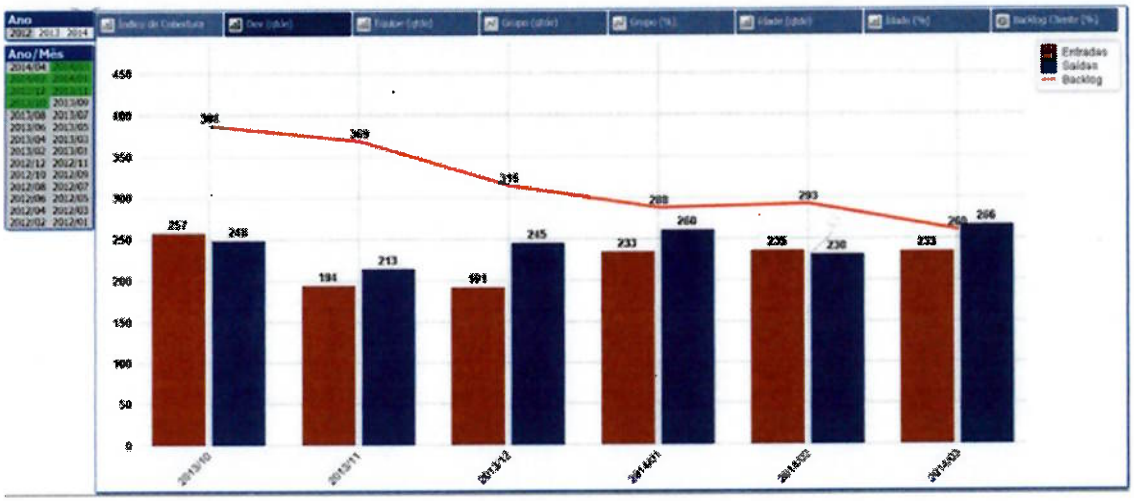

Figura 2 - Entrada e Saída de Projetos (Fonte: Empresa, 2014)

Apesar da grande quantidade de saída de demandas, a imagem mostra que a quantidade de demandas entregues é quase equivalente a quantidade solicitada por mês. Dificultando a diminuição do *Backlog* (demandas solicitadas que ainda não foram entregues) existente.

Todos os departamentos da empresa solicitam projetos para a área de Tecnologia da Informação. Desde simples manutenções até grandes projetos que modificam processos ou os adaptam para uma determinada situação. Os projetos são entregues após passar por especificação, desenvolvimento, testes, aprovação do cliente e publicação de versão.

Conforme dito anteriormente, o sistema controla todos os processos do inicio ao fim. Uma única alteração faz com que toda a aplicação seja republicada em produção. Como existem publicações semanais, toda semana existe um receio da gerencia de ocorrer erros em produção.

Será utilizado o mês Setembro de 2013 como base para demonstrar a quantidade de erros que podem ocorrer. Um mês de trabalho de 43 desenvolvedores resultou em 13 erros em produção, 18 erros encontrados pela equipe de Garantia de Qualidade e 13 erros de Banco de Dados. Variando entre erros simples até erros com maior impacto em produção.

Um erro que tenha impacto no módulo de Fábrica pode comprometer o Abate ou a Produção. Fazendo com que todos os funcionários tenham que estender o turno de trabalho, resultando em horas extras de todos os funcionários para cada fábrica. Em

um mês são abatidas cerca de 3.500.000 de cabeças de gado e produzidas 45.000.000 de caixas.

Erro no módulo Comercial pode afetar a entrada de pedidos que por mês o sistema recebe cerca de 300.000 pedidos. Pode afetar também os Suprimentos que impacta na produção.

Erro no módulo Financeiro pode afetar o Faturamento. Se um pedido não for faturado, o produto não pode ser embarcado para distribuição. Muitas empresas possuem horário de recebimento de mercadorias, caso o produto não seja entregue no horário determinado, o cliente pode cancelar a compra e até mesmo optar por não realizar mais pedidos, deixando de ser um cliente. Se a entrega não for realizada, existem ainda outros problemas, como gasto desnecessário de frete de entrega e a perda do valor da diária do caminhão.

Erro no módulo Administrativo pode afetar todo o sistema. Podendo bloquear ações que usuários deveriam realizar ou disponibilizar informações a usuários que não poderiam ter acesso.

1.2 Objetivo

Apresentar proposta de imaplantação do PSP (Personal Software Process) para auxiliar a empresa a garantir a qualidade do software com mais precisão em sua estimativa. Causando menos perda de dinheiro e desenvolvendo profissionais e produtos mais maduros.

1.3 Justificativas

Para evitar todas essas situações, o departamento de Tecnologia da Informação vem se preocupando cada vez mais com a qualidade do produto entregue. Mas também não abre mão da velocidade na entrega de projetos para o negócio. O sistema deve continuar acompanhando o enorme e veloz crescimento da empresa. Por este motivo a solução proposta deve ser pouco intrusiva, flexível e de fácil adaptação para os desenvolvedores.

Surgiu a necessidade de melhorar o processo de desenvolvimento. O CMMI é uma das soluções existentes para melhoria de processos e possui uma parte específica

para o desenvolvimento de sistemas, chamado CMMI-DEV. Baseado em boas práticas, cobre todo o processo de desenvolvimento desde a criação até a manutenção dos produtos e serviços.

No documento "CMMI® for Development" (CMMI Product Team) existe a definição e maior detalhamento sobre o CMMI para desenvolvimento de sistemas. A imagem abaixo mostra os 3 grandes focos que as empresas costumam ter: Pessoas, Procedimentos e Métodos e Ferramentas e Equipamentos.

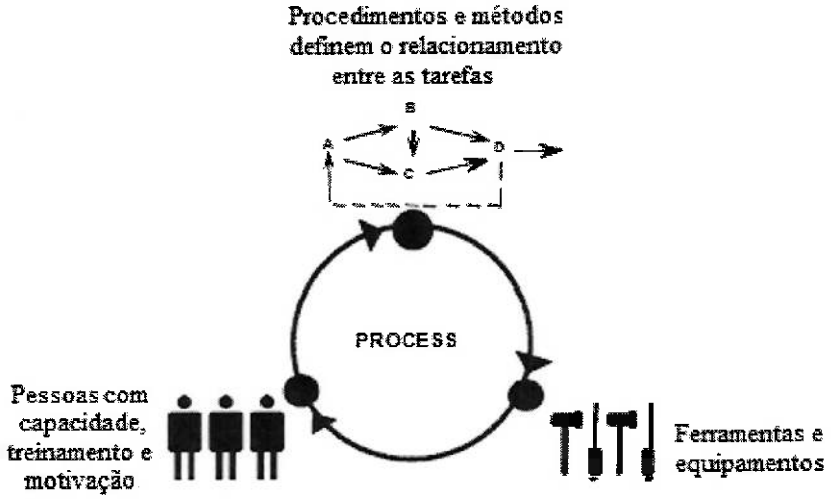

Figura 3 - CMMI for Dev (CMMI Product Team)

O CMMI para desenvolvimento foca principalmente no Processo que une as três partes. A melhoria do processo faz com que a empresa tenha infraestrutura e estabilidade para suportar mudanças tecnológicas e de profissionais.

Apesar de existirem diferentes implantações do CMMI nas empresas, a empresa deve estar disposta a sofrer uma grande mudança nos processos. Analisando o cenário agressivo da empresa em questão, não seria viável aplicar uma mudança no processo nesse momento. A maioria dos projetos são pequenas e desenvolvidas por um único desenvolvedor. De acordo com o texto "Results of Applying the Personal Software Process" (FERGUSON, HUMPHREY, KHAJENOORI, MACKE, & MATVYA, 1997), em 1989 o SEI (Software Engineering Institute) começou a ser questionado como aplicar o CMMI em pequenos projetos. Então surgiu o PSP (Personal Software Process) criado pelo Watts Humphrey na Universidade Carnegie Mellon.

O PSP não substitui o CMMI, mas coloca foco nos profissionais que irão desenvolver o software. Os torna mais disciplinados no trabalho, auxilia a estimar tamanho e

tempo de projeto e também como planejar o seu trabalho. Acompanhando o trabalho realizado conseguem visualizar os problemas e melhorar a qualidade do produto final.

O PSP é um framework que auxilia no planejamento e gerenciamento da produção de software com alta qualidade. Estimula a disciplina dos desenvolvedores resultando na diminuição de desperdício, erro e ineficiência.

Diferentemente das outras áreas de conhecimento, o desenvolvimento de software não propõe processos que vão auxiliar no trabalho dos desenvolvedores. Qualidade é raramente medida e o planejamento é medido através de pura estimativa.

De acordo com o artigo (FERGUSON, HUMPHREY, KHAJENOORI, MACKE, & MATVYA, 1997), o PSP é introduzido através de um curso, onde durante o curso pode-se ver melhora de 20% na produtividade e diminuição de mais de cinco vezes a quantidade de problemas gerados pelo software.

1.4 Estrutura do Trabalho

O Capítulo 1 INTRODUÇÃO apresenta as motivações, o objetivo, as justificativas e a estrutura do trabalho.

O Capítulo 2 REVISÃO BIBLIOGRÁFICA apresentaa fundamentação teórica para apresentar embasamento para a criação da proposta de implantação, casos práticos para que fossem utilizados métoos bem sucedidos e evitar problemas que outras empresas enfrentaram.

Baseado nos itens do capítulo 1 e 2, o Capítulo 3 PROPOSTA apresenta o plano de imaplantação do PSP, utilizando as técnicas e adaptando o cenário para a empresa em questão.

O Capítulo 5 CONSIDERAÇÕES FINAIS descreve a conclusão final do trabalho e trabalhos futuros que poderão ser iniciados baseados nesse trabalho.

REFERÊNCIAS relaciona todas as citações e referências do trabalho com relatórios, artigos de revista, livros e manuais.

2. REVISÃO BIBLIOGRÁFICA

O PSP tem como objetivo dar a habilidade de estimar e planejar o trabalho a ser realizado pelo desenvolvedor. Permite que o profissional consiga acompanhar a evolução da entrega e analisar o trabalho realizado. O desenvolvedor aprenderá a se conhecer melhor, facilitando a aplicação de melhorias contínuas ao seu método de trabalho, pois conhecerá os seus defeitos e poderá atuar sobre estes nos próximos projetos. Tendo o conhecimento do tempo e esforço necessário para desenvolver um determinado projeto, minimiza o insucesso dos projetos de desenvolvimento de software..

Muitos projetos são mal sucedidos pela incapacidade de estimativa do esforço e tempo que será gasto em um projeto.

De acordo com o PMBOK, existem restrições no Gerenciamento de qualquer projeto. Antes da 4ª edição era citada a famosa "restrição tripla", composta pelas restrições: Tempo, Custo e Escopo.

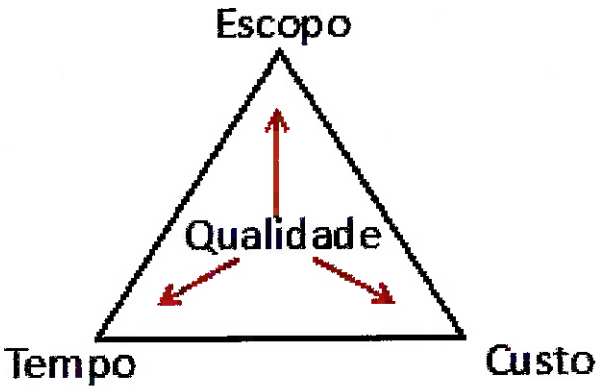

Figura 4 - Restrição Tripla do Gerenciamento de Projetos (Fonte: elaborado pela autora)

A partir da 4ª edição as restrições se tornaram: escopo, qualidade, cronograma, orçamento, recursos e riscos.

A estimativa de um trabalho a ser realizado está em ambas às versões do PMBOK como restrição de um projeto. Sem saber quanto tempo será gasto, a chance das demais restrições serem afetadas é extremamente grande. A falha na estimativa de

tempo e esforço em conjunto com a falta de qualidade pode causar atraso na entrega do projeto e até leva-lo ao insucesso.

A partir do momento em que os desenvolvedores conseguirem realizar as estimativas apuradas e desenvolver código com qualidade, o planejamento estratégico dos projetos poderá ser muito mais efetivo. Aumentando o comprometimento com os clientes que solicitam os projetos. Haverá também diminuição do retrabalho, que hoje na empresa citada é de 20%.

Antes de iniciar o detalhamento do framework PSP, os itens que compõe a restrição tripla do gerenciamento de projetos serão introduzidos para uma melhor base de entendimento.

2.1 Escopo

Os requisitos são a base para o escopo do projeto. São as características ou comportamentos solicitados e/ou esperados pelo cliente que devem compor o produto a ser criado.

Os desenvolvedores são as pessoas que possuem a visão mais clara do tempo que será gasto no desenvolvimento de um requisito e são responsáveis por estimar o tempo e esforço a ser aplicado em determinada tarefa. Na empresa em que será aplicada essa proposta, além da estimativa, os desenvolvedores são responsáveis por criar a análise dos requisitos antes de implementá-los. Por este motivo, este capítulo irá englobar técnicas de elicitação de requisitos que ajudarão nas estimativas das tarefas.

Existem diversas técnicas que podem ser utilizadas para realizar a elicitação de requisitos. O livro (RUPP & POHL, 2011) utilizado como base da certificação CPRE-FL (Certified Professional for Requirements Engineering) classifica as técnicas nos seguintes grupos:

- Pesquisa
- Criatividade
- Baseadas em documentos
- Observação
- De apoio

2.1.1 Pesquisa

Técnica utilizada por engenheiros de requisitos que possuem clientes que estejam comprometidos em investir tempo e esforço para expressar suas necessidades e problemas.

Um exemplo de pesquisa é o questionário, onde podem ser colocadas questões de uma ou múltipla escolha, para obtenção de muitos dados em um curto intervalo de tempo. Pode atingir uma grande quantidade de pessoas, sendo comum a utilização de formulários online.

Outra prática é a entrevista, nessa técnica as perguntas são elaboradas especificamente para os envolvidos que irão participar das entrevistas. Exige tempo do cliente para que sejam respondidas as questões previamente criadas pelo engenheiro, porém é possível obter informações mais específicas e cobrir assuntos mais abrangentes. As perguntas iniciais dão fluxo à conversa, mas durante a entrevista outros pontos podem ser discutidos de acordo com a resposta dos entrevistados.

2.1.2 Criatividade

Não são muito adequados para elicitar requisitos precisos sobre o comportamento de um sistema, mas costumam ser utilizados para criar requisitos inovadores e inesperados. Pode também ser utilizado como uma forma de criar a ideia inicial de um produto ou sistema.

Brainstorming é uma técnica conhecida por ser muito simples de ser aplicada. Basicamente existe um moderador que irá ter a função de coletar todas as ideias desenvolvidas pelos integrantes da reunião. Todas as pessoas presentes tem o direito de dar ideias sem que estas sejam julgadas, podem utilizar a ideia de outras pessoas para criar novas e até mesmo mesclar com alguma que já teve. Após um período de tempo todas as ideias passam por uma análise rigorosa para que sejam filtradas as mais aplicáveis ao negócio.

Brainstorming-paradoxy utiliza a mesma técnica que o brainstorming, porém reúne medidas para evitar que determinados riscos ocorram.

A Analogia utiliza situações da natureza para que sejam feitas comparações com problemas que o projeto enfrenta.

A Mudança de Perspectiva muito conhecida como *Six Thinking Hats* (seis chapéus pensantes) apresenta uma técnica que mostra as diferentes perspectivas dos envolvidos no projeto, cada integrante do grupo que irá participar da atividade "coloca um chapéu" de um determinado *stakeholder*. Cada um é responsável por expor o seu cenário com pontos de vista e problemas.

2.1.3 Documentos

Técnica utilizada quando já existe um sistema em funcionamento que terá suas funcionalidades replicadas à outro sistema.

A técnica da Arqueologia extrai informações de documentações ou código-fonte.

Na Leitura em Perspectiva as informações também são retiradas de documentações e/ou códigos-fonte, porém apresenta a diferença de que a leitura é realizada a partir da perspectiva de um determinado papel. Por exemplo, um implementador ou testador. Portanto a leitura é realizada com foco nos pontos mais relevantes para aquele papel.

Documentos já existentes também podem ser utilizados na técnica de Reutilização. Os documentos anteriormente escritos são atualizados para poderem ser utilizados na nova implementação.

2.1.4 Observação

As técnicas de observação são mais recomendadas quando os stakeholders não possuem disponibilidade para passar os requisitos ou até mesmo não sabem expressar e representar o seu conhecimento.

Pode ser realizada a Observação de Campo, onde engenheiros de requisitos acompanham o especialista ou usuário até o local de trabalho e documentam todos os processos e procedimentos que são realizados.

No apprenticing o engenheiro de requisitos aprende e pode até realizado o trabalho do stakeholder para conseguir entender e questionar qualquer procedimento que não esteja claro ou que seja muito complexo.

2.1.5 Apoio

Essas técnicas apoiam as demais técnicas. Reforçam os pontos fracos e desvantagens de outras técnicas.

Mapa Mental é uma técnica utilizada em conjunto às técnicas de criatividade, pois é necessário manter uma organização de ideias que são ditas aleatoriamente. Auxilia no agrupamento de informações para melhor entendimento e facilita as formações de conceitos.

A técnica de observação possui um problema, após ser aplicada pode ter deixado itens importantes no esquecimento. Por isso pode ser utilizada a Gravação, que permitirá a verificação desses itens observados posteriormente para que não seja necessária uma nova visita.

Protótipos são criados quando o usuário não possui uma visão muito clara do que deseja. Entrevistas e questionários podem deixar o usuário confuso e incerto sobre determinadas decisões, nesse momento os protótipos são excelentes para demonstrar o que o engenheiro de requisitos entende e até propõe. Alinhando os entendimentos e expectativas.

2.2 Tempo

O tempo é outra ponta da restrição tripla. Influencia diretamente no produto que será entregue. Se o tempo previsto para o projeto não for calculado de forma assertiva, o escopo, o custo e/ou a qualidade será diretamente afetada. Existe um tópico no guia PMBOK (Project Management Institute, 2013) que detalha o gerenciamento de tempo de um projeto. Nesse tópico do trabalho não será exposto o gerenciamento do tempo, mas algumas técnicas utilizadas para realizar a estimativa do tempo que resulta na principal, que é o cronograma do projeto.

O PMBOK mostra cinco técnicas/ferramentas para desenvolver um cronograma:

2.2.1 Análise Matemática

Calcula os períodos em que cada atividade pode ser executada. A junção de todos esses períodos não reflete o cronograma, mas essa técnica permite que seja

visualizada as janelas onde as atividades podem se encaixar. Auxiliando na montagem do cronograma.

2.2.2 Compreensão da duração

Contendo as informações dos recursos que serão utilizados, a disponibilidade destes no período do projeto e a análise matemática, é realizada a compressão da duração. São gerenciados os períodos para que se encaixem no menor tempo possível. Deve ser levado em consideração que existem diversas atividades que não podem ser simplesmente comprimidas no menor tempo existente, existem atividade que dependem de outras e atividades que não podem ser realizadas em paralelo. Para isso é necessário analisar as restrições e possíveis colisões.

2.2.3 Simulações

As simulações contemplam os possíveis riscos que o projeto pode passar. Desde problemas técnicos até a diminuição do tempo de participação dos recursos no projeto. É realizada a analise do risco e calculado o tempo extra que seria somado ao projeto caso esse problema ocorresse.

2.2.4 Nivelamento heurístico dos recursos

Utilizado para manter o cronograma conforme esperado. Pode-se realocar recursos para atividades mais críticas, utilizar datas e horários do dia que não estavam programados para os recursos trabalharem (finais de semana, feriados, horas extras, etc.), aumentar a produtividade através de ferramentas, tecnologias ou métodos.

2.3 Custo

No PMBOK (Project Management Institute, 2013) também existe a gerencia de custos. Onde é realizado o cálculo do custo do projeto baseado no valor dos recursos envolvidos no projeto e no tempo que irão trabalhar. Muitos projetos

ultrapassam o custo inicial que foi estimado para o projeto, pois o esforço é mal calculado.

Na empresa utilizada para realizar esse trabalho, o esforço de um projeto é estimado pelo analista que irá desenvolvê-lo. Pela falta de visibilidade de tamanho e complexidade acaba fazendo com que o custo do projeto aumente. Por esse motivo será apresentado o framework PSP no próximo tópico do trabalho. Tendo como objetivo a melhora na estimativa de projetos, mantendo a qualidade, e refletindo diretamente no custo do projeto.

2.4 PSP

O PSP por ser um framework simples e independente de tecnologia pode ser aplicado em qualquer projeto de desenvolvimento de software por profissionais com diversos níveis de conhecimento e experiência. A flexibilidade e adaptação do PSP podem ser vistas até nos modelos mais atuais, como a proposta de aplicação do PSP utilizando o BDD (*Behaviour-Driven Development*) no desenvolvimento de aplicativos para celular (GERBALDO & SANTOS, 2014). Estimula a abstração do código e criação de design modularizado, levando a um design limpo com partes reutilizáveis. Gera entregáveis progressivamente e ajuda a minimizar os riscos. Faz com que os desenvolvedores façam o planejamento, sigam o planejado, inspecionem constantemente e melhorem o processo para os próximos projetos. Dando atenção a qualidade desde o inicio do desenvolvimento.

De acordo com o "Technical Report do SEI" (HUMPHREY, The Personal Software Process (PSP), 2000) O PSP possui os seguintes princípios:

- Cada desenvolvedor é diferente, portanto é necessário que cada um planeje o seu trabalho de acordo com seus próprios dados;
- Para constantemente melhorar a sua performance, precisam utilizar processos bem definidos e que possibilitem a medição do resultado;
- Para produzir produtos com qualidade devem se sentir responsáveis pela qualidade de seus produtos;
- É mais barato encontrar e corrigir os erros antes no processo do que depois;
- É mais eficiente prevenir defeitos do que encontrar e corrigir depois;

- O correto é sempre a maneira mais rápida e barata para executar o serviço;

Assim como o CMMI, com a disciplina criada através do processo, a melhoria continua será mais fácil de ser atingida. Sempre identificando os problemas e propondo melhorias. O PSP é um framework que auxilia na análise e medição para caracterizar o seu processo de desenvolvimento através de passos definidos, formulários e padrões.

2.4.1 Etapas do PSP

Conforme dito anteriormente, o PSP surgiu da necessidade de aplicar um modelo de processos em pequenos projetos. Por ter sido baseado no CMMI, segue a ideia parecida com os níveis de maturidade do CMMI, porém o PSP possui sete níveis.
O conteúdo do PSP é dividido em sete etapas, conforme pode ser observado na imagem abaixo. O último item não faz parte do PSP, é a etapa seguinte. Envolve o processo de desenvolvimento em uma equipe, porém como o foco deste trabalho consiste no desenvolvimento de projetos individuais, este tópico não será abordado.
Cada um dos itens presentes em cada etapa está explicado nos tópicos seguintes desse trabalho. Em cada etapa é aplicado de 1 a 3 itens, quando o desenvolvedor segue para a próxima etapa deve aprender da fase atual e continuar aplicando os itens aprendidos nas etapas anteriores.

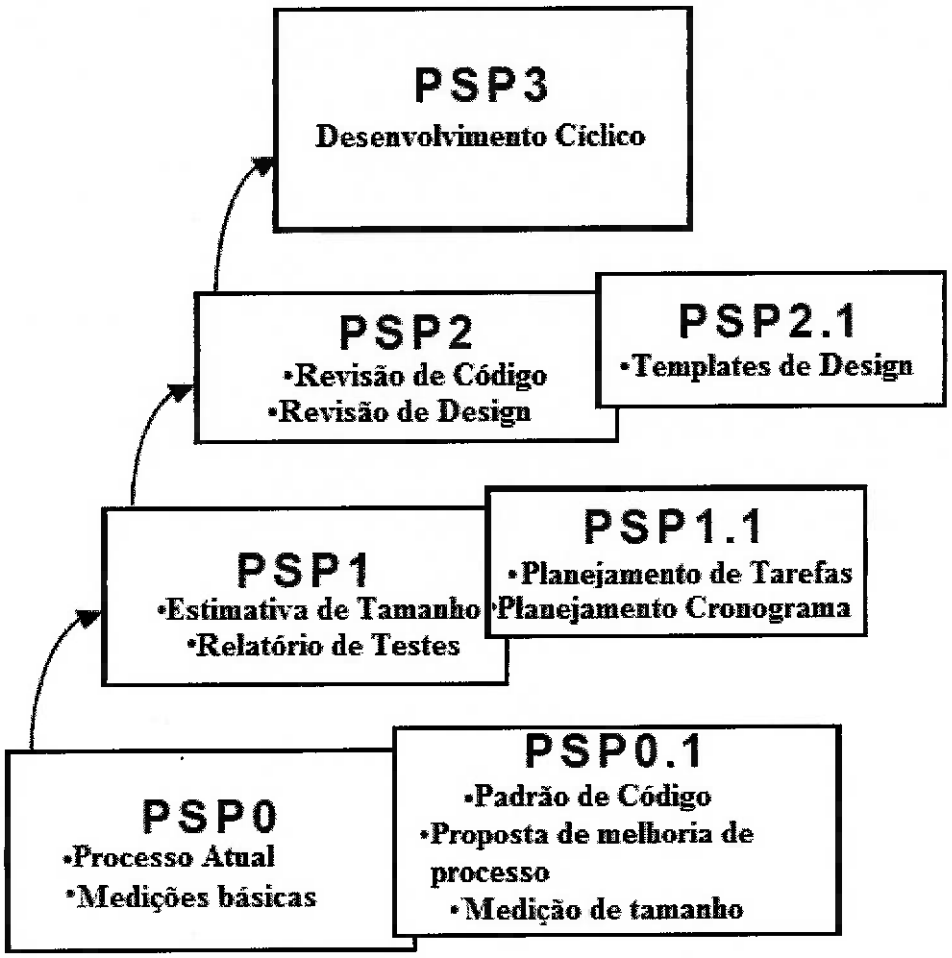

Figura 5 - Fases do PSP (HUMPHREY, 2000)

Ao aplicar o PSP, o desenvolvedor inicia no PSP 0 com a forma de trabalho que sempre seguiu, dessa forma é possível obter informações importantes para descobrir o que precisa ser melhorado no processo, de forma pouco intrusiva.

No PSP 0.1 inicia a padronização do código para que seja possível realizar comparações entre projetos e os dados sejam confiáveis. Permite também que o desenvolvedor possa criar propostas de melhoria para o processo (PIP – Process Improvement Proposal).

Com a padronização é possível seguir para o PSP 1 onde é iniciado o processo de planejamento, utiliza-se o método PROBE que será explicado na parte do planejamento. Também é criado o relatório de testes, onde são incluídos os defeitos encontrados e a fase em que foram inseridos.

Após passar pelo PSP 1 o desenvolvedor já terá a quantidade de linhas desenvolvidas e poderá comparar o resultado com as estimativas dadas no inicio do

projeto. No PSP 1.1 já será possível criar cronogramas mais assertivos com tarefas mais específicas, pois nessa fase o desenvolvedor já terá melhor conhecimento do tempo que gasta em cada tarefa.

No PSP 2 o foco é na qualidade do produto. São utilizados checklists de código e design para que sejam realizadas revisões que minimizem os problemas no produto. É recomendado que cada desenvolvedor crie e adapte seu próprio checklist.

O PSP 2.1 não exige que seja seguido um template de design, mas são oferecidas propostas de modelagem. A verificação do design é realizado antes mesmo do inicio do desenvolvimento. Os modelos propostos são: operacional, funcional, modelo de estados e pseudocódigo.

E finalmente chegando ao PSP 3, onde o processo já está definido. O desenvolvedor possui o material necessário para seguir com o processo. Sempre tendo a possibilidade de adaptá-lo conforme a necessidade. Podendo agora pensar em formas de evoluir e melhorar o que já existe.

2.4.2 Estrutura do PSP

Para facilitar o entendimento do fluxo do processo a imagem abaixo descreve como o PSP é aplicado.

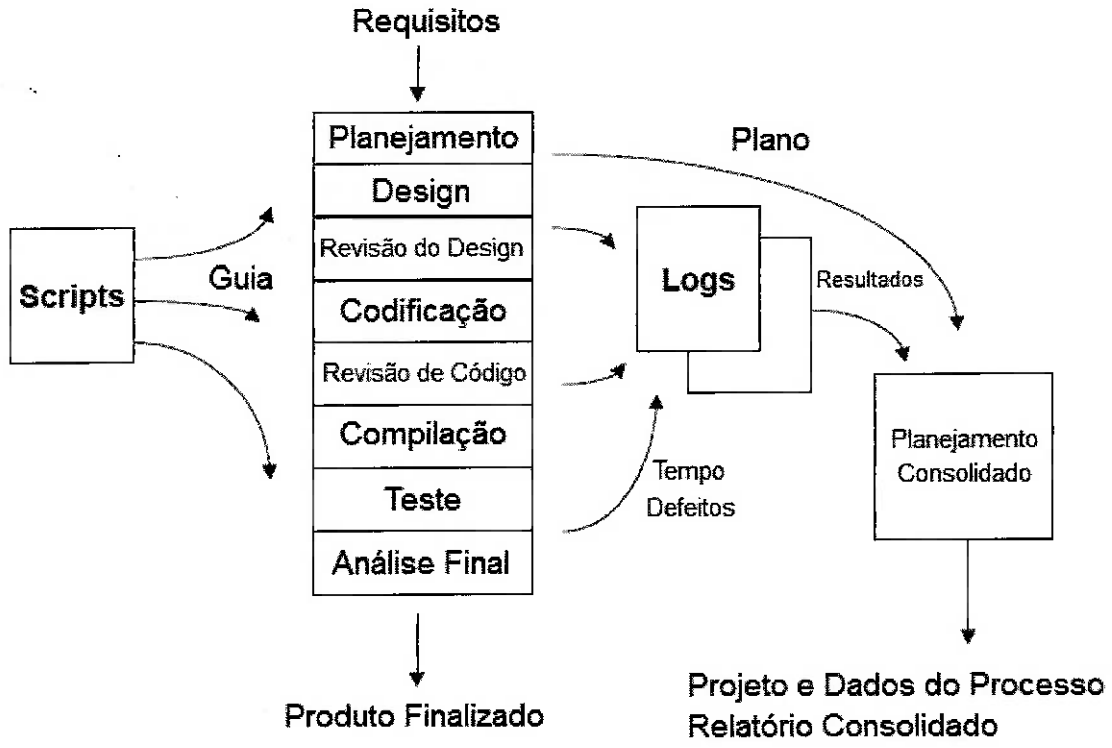

Figura 6 - Estrutura do PSP (HUMPHREY, 2000)

Os requisitos são utilizados para realizar o Planejamento. Os dados iniciais são inseridos no documento de Planejamento Consolidado (Summary Plan). O projeto passa por todas as etapas de um projeto de desenvolvimento (design, revisão do design, codificação, revisão do código, compilação e teste) seguindo os scripts que serão descritos no próximo tópico, a diferença é que nas etapas de revisão e análise final do projeto, dados sobre o tempo gasto e os defeitos encontrados são coletados para serem inseridos no documento de Planejamento Consolidado. O relatório e os dados são armazenados para os próximos projetos poderem consultar o histórico e se basear para as próximas estimativas.

2.4.2.1 Script do Processo

No quadro abaixo segue a recomendação do script detalhado para implantação do PSP de acordo com Watts Humphrey. As fases são dividas em Critério de entrada onde são descritos os passos de pré-requisito para iniciar o processo, seguindo com o Planejamento, Desenvolvimento, *Postmortem* e finalizando com os produtos de saída resultantes do processo que serão utilizados como referencia para projetos futuros.

Número da Fase	Objetivo	Para guiar no desenvolvimento de módulos de programa
	Critério de entrada	• Descrição do problema • PSP1 Formulário *"Project Plan Summary"* • Template de estimativa de tamanho • Histórico de tamanho estimado e real • Log de tempo e defeitos encontrados • Padrão de defeito • *Stop Watch* (opcional)
1	Planejamento	• Produzir ou obter os requisitos • Utilizar o método PROBE para estimar a Quantidade de Linhas de código requeridas • Completar o template de estimativa de tamanho • Incluir os dados no *"Project Plan Summary"* • Completar o log de tempo gasto
2	Desenvolvimento	• Fazer o design do programa • Implementar o design • Compilar o programa e corrigir e anotar todos os erros encontrados • Testar o programa e corrigir e anotar todos os erros encontrados • Completar o log de tempo gasto
3	*"Postmortem"*	• Completar o formulário *"Project Plan Summary"* com os dados reais

		de tempo, defeito e tamanho do projeto
	Critério de saída	• Programa completamente testado • Formulário *"Project Plan Summary"* completo com dados estimados e reais • Template de estimativa de tamanho completo • Template de relatório de teste completo • Formulários PIP completos • Logs de defeitos e tempo gasto completos

Quadro 1 - Script de Processo (HUMPHREY, 2000)

2.4.2.2 Formulário de planejamento do projeto

O formulário de planejamento do projeto (*Project Plan Summary*) contém informações básicas sobre o autor do programa e também sobre o projeto que está sendo desenvolvido. Dessa forma é possível buscar projetos com mesmas características para futuras referências.

Na sequência existem espaços reservados para que seja realizado o acompanhamento do projeto conforme for desenvolvido. O desenvolvedor é responsável por preencher com suas estimativas antes de iniciar o projeto e complementar no final do projeto com os dados reais.

Aluno		Data	
Programa		#Programa	
Instrutor		Linguagem	

Resumo LDC/Hora	**Planejado**	**Atual**	**Data Final**

Tamanho Programa (LDC):	**Planejado**	**Atual**	**Data Final**
Base(B)	(Medido)	(Medido)	
Excluído (E)	(Estimado)	(Contado)	
Modificado (M)	(Estimado)	(Contado)	
Adicionado (A)	(N – M)	(T – B + E – R)	
Reutilizado (R)	(Estimado)	(Contado)	
Total Novo & Modificado (N)	(Estimado)	(A + M)	
Total LDC (T)	(N + B – M – E + R)	(Medido)	
Total Novos Reutilizáveis			
Total Objeto LDC (E)			

Tempo na fase (min.)	**Planejado**	**Atual**	**Data Final**	**Data Final (%)**
Planejamento				
Design				
Codificação				
Compilação				
Testes				
Postmortem				
Total				

Defeitos Injetados	**Atual**	**Data Final**	**Data Final (%)**
Planejamento			
Design			
Codificação			
Compilação			
Testes			
Total Desenvolvido			

Defeitos Removidos	**Atual**	**Data Final**	**Data Final (%)**
Planejamento			
Design			
Codificação			
Compilação			
Testes			
Total Desenvolvido			
Após desenvolvimento			

Figura 7 - *Project Plan Summary* (HUMPHREY, 2000)

2.4.2.3 Planejamento

PROBE significa Proxy Based Estimating. Primeiro os desenvolvedores devem decidir os objetos que irão compor o design do produto de acordo com os requisitos que o cliente necessita. Definem o tipo e os métodos que cada objeto irá conter. Baseado no desenvolvimento de objetos anteriores é calculado o tamanho total do produto utilizando regressão linear. Para que seja possível utilizar a regressão linear é necessário que o desenvolvedor tenha dados históricos de pelo menos 3 projetos anteriores.

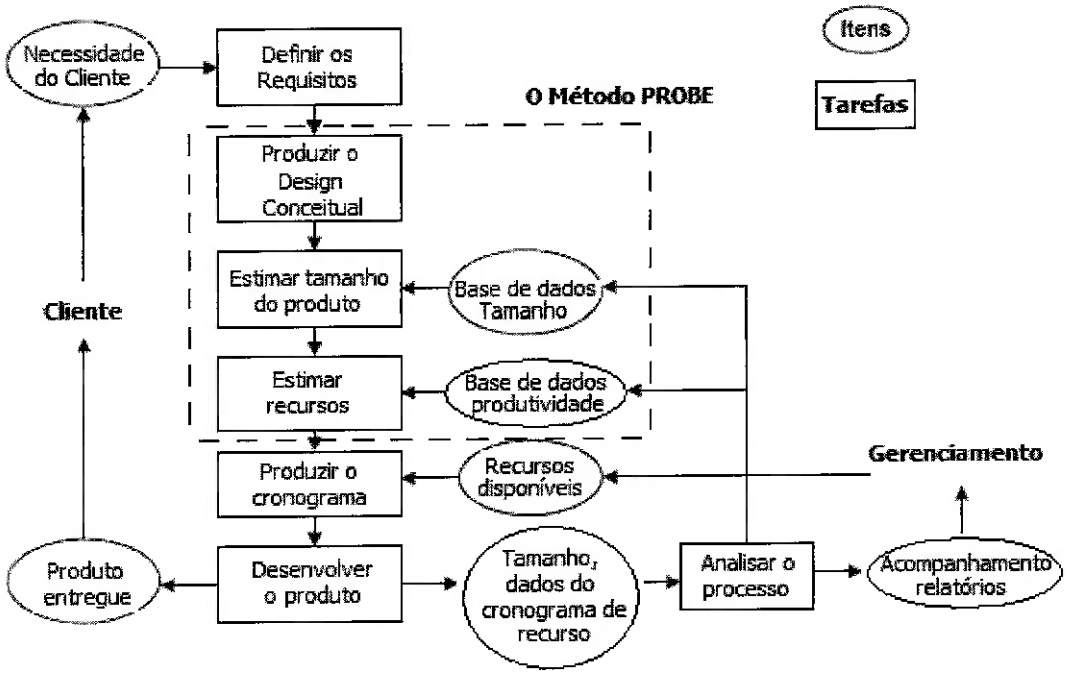

Figura 8 - Método PROBE (HUMPHREY, 2000)

Conforme observado na figura acima, o método PROBE deve ser aplicado na fase de design e estimativa de produto e recursos. O método auxiliará principalmente na fase das estimativas, onde o desenvolvedor apresenta ter muita dificuldade em saber como realizá-las.

2.4.2.4 Obtenção de Dados

Os dados obtidos no PSP são importantes para analisar o que foi previsto e o que realmente foi realizado, para que seja possível obter melhores resultados para os próximos projetos.

Um dos dados obtidos é o tempo, que é utilizado para monitorar o tempo gasto. É anotada a data e hora do inicio e do término das tarefas, incluindo as interrupções que podem ser desde paradas para um café até o desenvolvimento em outro projeto que está sendo realizado em paralelo.

Para que seja possível dar a estimativa em tempo, é necessário saber o tamanho do projeto a ser desenvolvido. Para obter esse dado o desenvolvedor deve realizar a estimativa do tamanho inicial e após finalizar o desenvolvimento medir o tamanho para fazer a comparação entre eles. O tamanho do projeto pode ser medido através da quantidade de linhas de código (LDC) desenvolvidas, o cálculo dessa métrica pode variar de empresa para empresa, caso não exista o desenvolvedor pode utilizar a sua própria. Como esse calculo é um trabalho desgastante e visto como desnecessário pelos desenvolvedores, o PSP recomenda a automação dessa contabilização, através de um programa que fará esse calculo.

Existem seis categorias para uma linha de código:

- Base: quando um produto já existe e será modificado, são todas as linhas de código existentes antes de qualquer modificação;
- Incluídas: linhas adicionadas ao código já existente ou a novos programas;
- Modificadas: linhas já existentes que sofreram alguma alteração;
- Excluídas: linhas existentes que foram excluídas;
- Reutilizadas: linhas de código que pertencem a alguma biblioteca comum e é utilizada por mais de um programa;
- Novas Reutilizáveis: novas linhas de código que serão incluídas em bibliotecas compartilhadas ou novas bibliotecas que poderão ser utilizadas por outros programas posteriormente

Para verificar o tamanho de um projeto, só são contabilizadas as linhas incluídas e modificadas, pois estas que despendem maior esforço e tempo do desenvolvedor.

Outro dado importante a ser obtido são os defeitos encontrados. No relatório de log de defeitos informações como a fase onde o defeito foi injetado, a fase onde o

defeito foi encontrado e corrigido e o tempo que levou a correção do defeito são inseridos para analise.

2.4.2.5 Gerenciamento da Qualidade

O principal objetivo de qualidade do PSP é detectar o defeito antes mesmo da primeira compilação. Acredita-se que cada desenvolvedor repete o mesmo padrão de erros, portanto se tiver um checklist que ajude a encontrar esses erros que sempre comete, a tendência é que os erros cometidos diminuam e com o passar do tempo esses erros deixem de aparecer. Para isso o processo possui Revisão de Código que deve ocorrer antes mesmo de compilar ou executar testes unitários.
A melhor forma de gerenciar os defeitos é prevenindo que eles apareçam. É gasto muito mais tempo e esforço quando o defeito é encontrado no produto final pelo usuário do sistema. Portanto é recomendado que o desenvolvedor anote todos os erros encontrados e as soluções que resolveram os problemas. Esses dados serão analisados posteriormente para identificar formas de evita-los através do processo.
Outra forma de evitar defeitos seria aplicar um bom design adequado ao cenário do programa. Dessa forma o desenvolvedor leva mais tempo formulando o design e tendo um bom entendimento do programa.

2.4.3 Casos de Uso Aplicados em Outros Cenários

No artigo *"Results of Applying the Personal Software Process"* (FERGUSON, HUMPHREY, KHAJENOORI, MACKE, & MATVYA, 1997) os autores mostram a necessidade da gerência enfatizar a importância da qualidade do produto, planejamento e monitoramento efetivo. Um gerente da empresa US&S reforça a importância do PSP através de reuniões semanais com a sua equipe de projeto para revisar o status, planos e datas. Enquanto outro gerente não deu a devida importância acompanhando o processo com os desenvolvedores de software então eles voltaram a utilizar o método antigo de trabalho.
No artigo *"Using A Defined and Measured Personal Software Process"* (HUMPHREY, Using A Defined and Measured Personal Software Process, 1996) é recomendado que a gerência receba o treinamento do PSP antes dos desenvolvedores de software, dessa forma a gerência consegue visualizar a

importância do PSP e o trabalho que será necessário para alcançar os objetivos visados.

E por fim no artigo *"An Experiment Measuring the Effects of Personal Software Process (PSP) Training"* (PRECHELT & UNGER, 2000) o autor conclui que talvez o ponto mais importante da implantação do PSP seja que o ambiente em que os desenvolvedores de software estão. Se estes estiverem em um ambiente que os estimule a utilizar o PSP, será a chave para o sucesso. Da mesma forma o artigo *"Teaching disciplined software development"* (ROMBACH, MUNCH, OCAMPO, HUMPHREY, & BURTON, 2007) esfatiza que o sucesso do PSP não depende simplesmente de sua implantação, mas também da motivação dos desenvolvedores em utilizar um processo estruturado e o comprometimento na produção de um software com qualidade.

Em seguida o curso deve ser dado àqueles desenvolvedores de software com menos experiência no desenvolvimento de software, pois conforme o artigo *"An Experiment Measuring the Effects of Personal Software Process (PSP) Training"* (PRECHELT & UNGER, 2000) profissionais com mais experiência ou mais habilidade talvez tenham menos evoluções, pois tem um melhor conhecimento de seus defeitos. E profissionais com menos experiência estão mais abertos a novos conhecimentos e novas formas de se trabalhar.

Após os cursos serem ministrados para os gerentes e desenvolvedores de software com menos experiência, devem ser escolhidas ferramentas que obtenham as informações de forma automatizada, pois dois pontos que mais incomodaram os autores do artigo *"The Personal Software Process: A Cautionary Case Study"* (JOHNSON & DISNEY, 1998) foi a quantidade de formulários impressos que precisam ser preenchidos pelos profissionais e a possibilidade de manipulação dos dados obtidos. Os autores do artigo *"Assessing PSP effect in training disciplined software development: A Plan–Track–Review model"* (WEN-HSIANG, NIEN-LIN, & WEI-MANN, 2010) concordam que para ser aplicado na indústria é necessário facilitar a aplicação, porém reforçam o cuidado de manter o mínimo para cobrir os passos essenciais do PSP, a fim de conseguir alcançar os resultados esperados.

3. PROPOSTA

Conforme descrito na Fundamentação Teórica, através do resultado de diversos casos de uso foi citado como recomendação essencial para o sucesso da implantação do PSP o apoio da gerencia para que estes reforcem e incentivem a utilização das técnicas propostas pelo PSP pelos desenvolvedores de software.

Portanto o primeiro ponto da proposta é dar um curso para a gerência conseguir entender a importância do PSP e as mudanças no processo que serão necessárias para que a eficiência e qualidade desejadas sejam atendidas.

A empresa que receberá a proposta de aplicação do PSP já realiza grande parte da obtenção de dados necessários, porém não utiliza essas informações para a melhoria do processo. Hoje os dados são analisados pela gerencia e somente ações de nível gerencial são tomados para melhorar a qualidade do produto. Os desenvolvedores de software devem dar estimativas sem base nenhuma e não realizam analise dos defeitos encontrados.

Para que seja possível ter uma melhor aderência, menor intrusão e maior automatização do processo serão utilizadas as ferramentas e dados hoje existentes. Porém utilizando os dados obtidos de uma forma mais efetiva, complementando o que for necessário para a utilização do PSP.

Seguindo as orientações do PSP, todas as fases propostas serão iniciadas com o processo atual. Dessa forma será possível analisar os dados para entender o que precisa ser modificado. O próprio processo que foi proposto nesse trabalho pode sofrer alterações conforme a empresa sentir necessidade. O PSP é um framework que deve ser adaptado ao ambiente em que está sendo utilizado, por este motivo apresenta ter muita aderência. A estimativa iniciará da mesma forma como é realizada hoje, no passo seguinte, PSP 0.1, é criado o documento que irá armazenar os dados estimados para serem comparados ao esforço real final. Os primeiros três projetos passarão somente pelas duas primeiras etapas, criando dados históricos. A partir do quarto projeto o desenvolvedor de software passará para o PSP1, onde utilizará os dados no método PROBE, que auxiliará na estimativa do projeto. Os próximos passos serão melhor detalhados nos tópicos a seguir, utilizando ferramentas e técnicas para melhorar a forma de trabalho do desenvolvedor.

3.1 PSP0

O desenvolvedor de software mantém o processo atual. A partir dos requisitos, realiza o planejamento baseado em sua experiência, assim como tem feito até hoje. Durante o desenvolvimento deve registrar todos os defeitos encontrados e o tempo gasto.

3.2 PSP0.1

Nesse passo é adicionada a medição do tamanho do projeto ao formulário "Process Improvement Proposal", no qual os desenvolvedores utilizarão para anotar também todos os defeitos encontrados e ideias para resolvê-los.
Esses dados serão utilizados como benchmark para a análise e estimativa.

3.2.1 Relatório de Log do Tempo Gasto

Tempo gasto é determinado a partir do inicio de uma tarefa até o final dela, considerando todas as interrupções. As interrupções podem ser desde telefonemas até desenvolvimento em outros projetos. Cada momento deve ser medido para que no final tenha um resultado assertivo. E para isso a empresa já possui um programa que registra todo o tempo gasto pelo funcionário.
O programa é iniciado junto com a máquina. Enquanto o computador está em utilização, o programa registra o tempo como período de trabalho, em qualquer momento é possível clicar em um simples botão ou acionar teclas de atalho para indicar o início de um atendimento, chamada telefônica, ou qualquer outro tipo de interrupção. A partir do momento que o computador fica sem atividade por mais de 15 minutos, o programa registra estado de ausência. O funcionário pode modificar os registros posteriormente. O programa não funciona como um rastreador, ele foi criado com o intuito de o profissional saber quais projetos esteve trabalhando, quantas interrupções teve em um dia e dessa forma conseguir se organizar melhor ou até mesmo justificar algum atraso ou excesso de atendimento em um período.

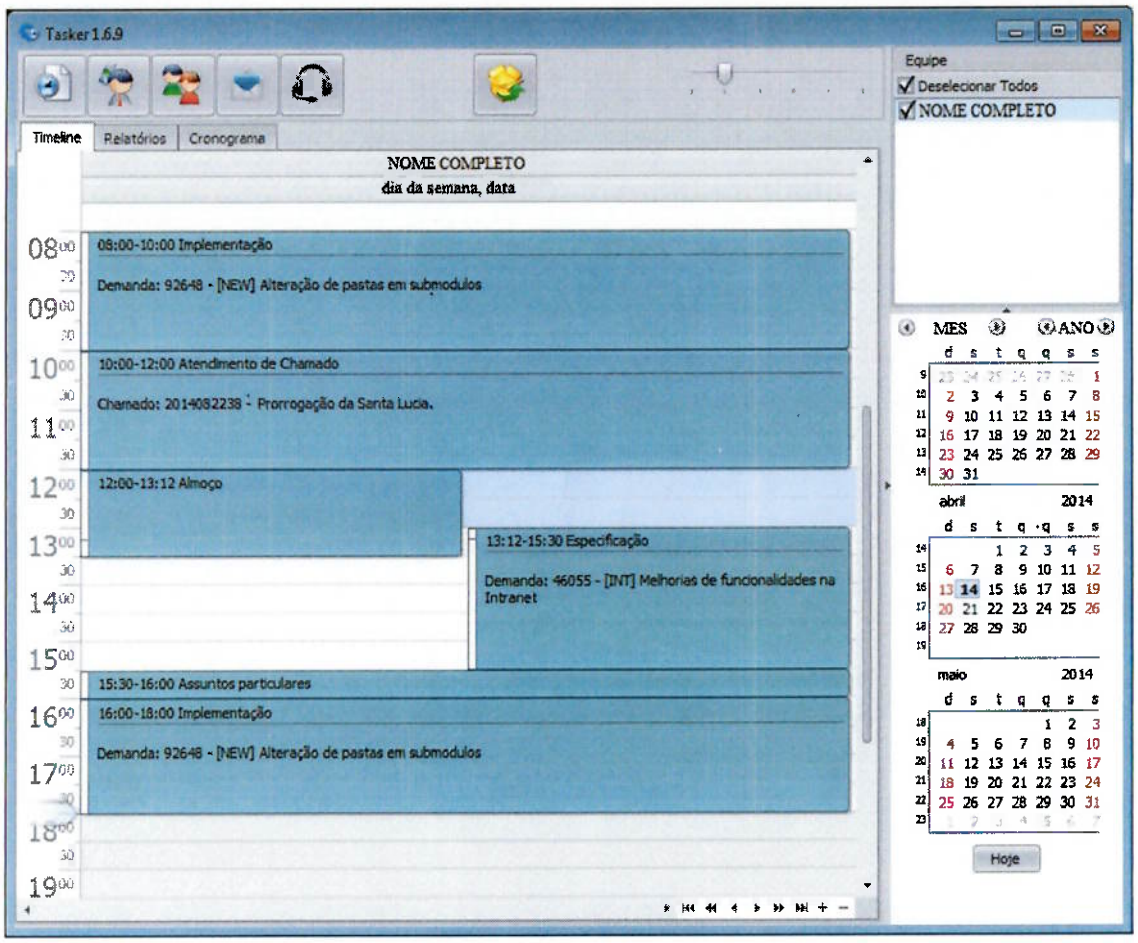

Figura 9 - Programa que armazena as tarefas que ocorreram no dia (Fonte: Empresa, 2014)

Na imagem é possível visualizar o modo de gerenciamento onde o profissional pode complementar os dados. No exemplo acima, o desenvolvedor de software iniciou o dia na tarefa de implementação do projeto 92648, parou as 10 a.m. para realizar o atendimento do chamado 2014082238, saiu para o almoço, iniciou a especificação do projeto 46055, fez um intervalo para resolver assuntos particulares e em seguida retornou ao desenvolvimento do projeto que estava desenvolvendo no inicio do dia.

Esses dados são utilizados pelo coordenador, para que sabia quanto tempo o desenvolvedor de software está utilizando em cada fase de cada projeto. O profissional somente fica atento se o seu apontamento de horas está em dia, não utiliza esses dados para analisar sua rotina de trabalho.

Através desse programa é possível saber o tempo gasto em cada projeto, com data e horário do inicio e termino, incluindo os intervalos e interrupções.

3.2.2 Relatório de Log dos Defeitos Encontrados

Hoje o desenvolvedor de software possui os dados de erros que ocorrem na Garantia da Qualidade, Implantação e em Produção.

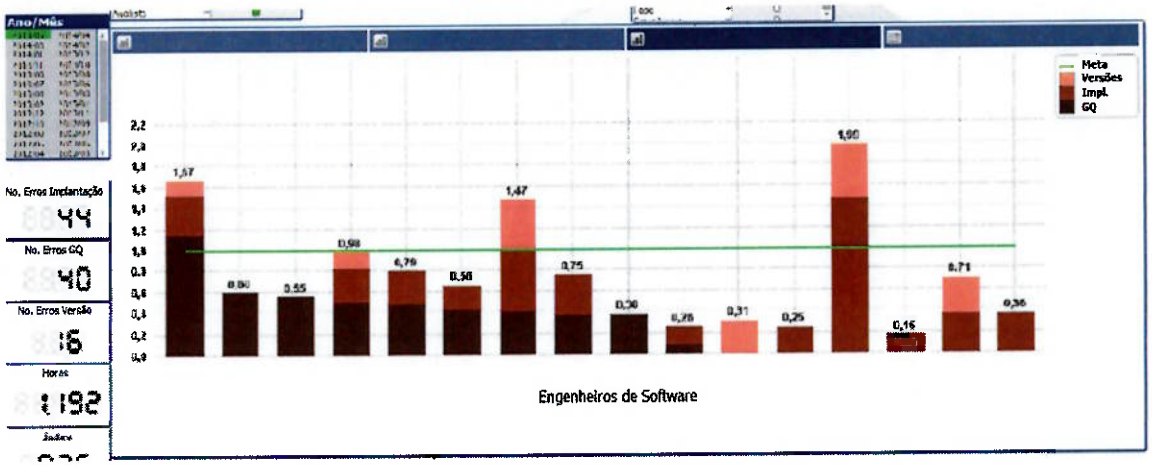

Figura 10 - Erros categorizados (Fonte: Empresa, 2014)

Será proposta a inclusão de um atalho na ferramenta de apontamento de horas que marcará o erro ocorrido e a solução aplicada para resolvê-lo. Sendo possível de forma rápida anotar essa informação para analise posterior.

3.3 PSP1

Nessa fase é iniciada a estimativa de tamanho e o relatório para testes. Para o item de planejamento utilizamos o método PROBE. Baseado no histórico até o momento. Para analisar o trabalho produzido, serão retiradas informações da ferramenta de repositório de código já utilizada na empresa. Nessa ferramenta é possível descobrir a quantidade de linhas modificadas de um determinado projeto já realizado.
O seguinte cálculo será levado em consideração:
Total LDC = Base (inicial) – Excluídas + Adicionadas + Reutilizadas

3.4 PSP1.1

Nessa etapa são adicionadas as estimativas de cronograma. Após a estimativa do tamanho de cada objeto, é possível criar o cronograma. Permitindo que o

desenvolvedor siga o cronograma estimado, utilizando o tempo estimado para aquele dia ou semana.

Os desenvolvedores de software tem dificuldade nessa etapa por não realizarem as tarefas na ordem em que as planejaram. Após a estimativa total do projeto, os desenvolvedores devem dividir o tempo entre as fases do projeto, baseando-se no histórico. Com o tempo dividido entre cada etapa do projeto, os desenvolvedores decidem o tempo que será gasto em cada tarefa em cada dia ou semana.

Para auxiliá-los, existe um relatório que exibe quais projetos deverão ser realizados no mês pelo desenvolvedor de software, qual é o tempo gasto, o tempo estimado para término e em qual fase o projeto se encontra.

Figura 11 - Lista de projetos (Fonte: Empresa, 2014)

Podendo se programar para realizar as demandas conforme a estimativa de entrega e esforço esperado.

3.5 PSP2

Os desenvolvedores criam e seguem checklists para realizar a revisão de design e de código. Utilizam os dados obtidos dos defeitos para realizar a medição e avaliação de qualidade.

Pelos checklists serem muito pessoais e variar muito de profissional para profissional, ficará aberto para o desenvolvedor criar o seu próprio. Será recomendado que seja incluído no fórum do projeto para que possa ser identificado como sendo um projeto que passou pelos requisitos de código e design.

Será criado um Identificador que representará o checklist. Este documento se tornará um dos documentos necessários para a implantação do sistema.

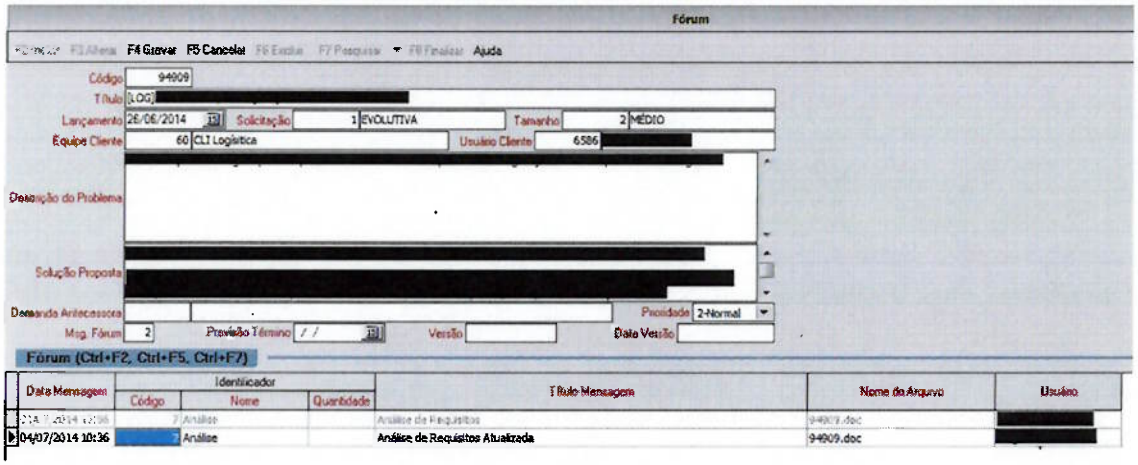

Figura 12 - Fórum do projeto (Fonte: Empresa, 2014)

Ficará armazenado no banco de dados para futuras referências. O desenvolvedor incluirá o *checklist* utilizado, mas terá flexibilidade de alterá-lo nos projetos seguintes conforme necessidade.

3.6 PSP2.1

Este item depende do andamento das demais etapas. Torna-se uma fase mais prática e com menos documentações. São introduzidas técnicas de especificação de design e formas de prevenção de erros. Será analisada como será implantada conforme o andamento do projeto.

3.7 PSP3

O design é verificado e o processo é adaptado para o ambiente de trabalho do desenvolvedor. Nessa etapa o processo será revisado e analisado se a forma de trabalho está simples e pouco intrusiva. E deverá ser realizado um acompanhamento da quantidade de erros injetados e a assertividade das estimativas do projeto, identificando a efetividade do PSP. Ajustando se necessário.

Deve ser sempre reforçada a importância do acompanhamento gerencial e apoio aos desenvolvedores. Conforme o processo for se ajustando ao ambiente de trabalho da empresa, novos desenvolvedores deverão receber o treinamento e iniciar a utilização do framework PSP. Sempre em ciclos onde cada um deles se ajusta ao processo para melhor aderência e efetividade.

3.8 Plano de Implantação

A implantação do PSP passará por seis fases. A primeira será utilizada como projeto piloto para que seja possível identificar a forma de trabalho dos funcionários da empresa, os problemas que normalmente enfrentam e a melhor forma de adaptar a forma de trabalho. Na segunda fase serão selecionados cinco desenvolvedores, preferencialmente que possuam pouca experiência. Estes cinco profissionais e todos os coordenadores receberão um treinamento sobre o PSP e será apresentado este plano de implantação. A terceira fase será a aplicação do PSP, passando por todos os passos descritos anteriormente. Na quarta fase será solicitado feedback dos desenvolvedores e coordenadores sobre a experiência de aplicar o PSP no ambiente atual. Na quinta fase os feedbacks serão analisados e o processo será adaptado para corrigir os problemas que ocorreram na primeira turma. Na sexta e última etapa, será montado uma apresentação do novo processo e os dados contidos nesse estudo somado aos resultados obtidos na primeira turma que utilizou o PSP, para que seja possível montar o plano de implantação para toda a área de desenvolvimento.

3.8.1 Cronograma

No quadro 2 é possível observar o cronograma que será proposto para a empresa desse estudo.

	1ª Semana	2ª semana	3ª semana	4ª semana	5ª semana	6ª semana	7ª semana	8ª semana	9ª semana	10ª semana
Análise do ambiente atual										
Curso PSP										
Aplicação do PSP										
Feedback										
Análise e adaptação do PSP										
Apresentação do modelo para toda área de desenvolvimento										

Quadro 2 – Cronograma Proposto para Plano de Implantação (Fonte: elaborado pela autora)

As marcações em cinza representam as semanas em que serão realizadas as tarefas descritas. Abaixo serão detalhados os papéis responsáveis por cada uma dessas tarefas.

3.8.2 Responsabilidades

Existirão basicamente quatro papéis envolvidos na implantação dessa proposta:

- Responsável pela implantação;
- Gerente de desenvolvimento;
- Coordenador de desenvolvimento
- Desenvolvedor.

Nos tópicos a seguir serão apresentados os detalhamentos de cada papel com suas respectivas responsabilidades.

3.8.2.1 Responsável pela Implantação

O responsável pela implantação do novo modelo será o papel central da implantação. Terá como objetivo analisar o ambiente atual, realizar todas as tarefas necessárias para que a proposta possa ser implantada e analisar todos os problemas para poder solucioná-los da melhor forma. Terá como função:

- Participar da análise inicial do ambiente atual;
- Ser o responsável por procurar e selecionar um curso de PSP que será realizado in company;
- Acompanhar, orientar e observar a aplicação do PSP;
- Receber o feedback do coordenador e dos desenvolvedores;
- Adaptar o modelo de acordo com o feedback;
- Apresentar o modelo final ao gerente de desenvolvimento e para toda área de desenvolvimento.

3.8.2.2 Gerente de Desenvolvimento

O gerente de desenvolvimento será envolvido no inicio e final do processo de implantação. Conforme citado anteriormente, é de extrema importância que o gerente participe e entenda como o processo funciona, para que possa apoiar a mudança e entender todo o benefício que o seu departamento irá receber.
Terá como função:

- Contribuir com informações para a análise do ambiente atual;
- Participar do curso *in company* sobre PSP;
- Ser o ouvinte chave da apresentação do novo modelo.

3.8.2.3 Coordenador de Desenvolvimento

Assim como o gerente de desenvolvimento, deverá entender e participar do processo para que possa apoiar a mudança no departamento de desenvolvimento. Além disso, observará e analisará a primeira aplicação do processo com os desenvolvedores, reforçando a necessidade de utilizar o framework e apontar melhorias ao processo.
Terá como função:

- Contribuir com informações para a análise do ambiente atual;
- Participar do curso in company sobre PSP;
- Acompanhar os desenvolvedores na aplicação do processo;
- Dar feedback na visão gerencial do projeto;
- Propor melhorias no novo processo;
- Ser o ouvinte da apresentação do novo modelo.

3.8.2.4 Desenvolvedor

Responsável por aprender o processo PSP, aplicar as alterações, analisar o novo processo de trabalho, criticar e propor melhorias para o processo final.
Terá como função:

- Contribuir com informações para a análise do ambiente atual;
- Participar do curso in company sobre PSP;

- Aplicar os conhecimentos e técnicas aprendidas no curso;
- Dar feedback do processo utilizado;
- Propor melhorias no novo processo;
- Ser o ouvinte da apresentação do novo modelo.

5. CONSIDERAÇÕES FINAIS

Para finalizar, estão descritas as contribuições que este trabalho trouxe e sugestões de trabalhos futuros que poderão ser escritos com base neste trabalho.

5.1 Contribuições do Trabalho

Este trabalho apresenta um plano prático e automatizado para a implantação do PSP. Os trabalhos desenvolvidos e os casos aplicados no mercado utilizam muitos formulários manuais que consomem muito tempo dos desenvolvedores, dificultando a adoção do PSP. Com essa estratégia a curva de aprendizado será mais suave, com retorno muito mais rápido.

Durante a elaboração do trabalho foi possível identificar muita resistência em relação aos profissionais da área de desenvolvimento de sistemas para aplicação de técnicas para a melhoria no processo de desenvolvimento. Deve ser levado em consideração que a proposta não força a utilização das técnicas propostas, o trabalho sugere um plano de implantação onde com o apoio da alta gerência espera-se ganho na aproximação da estimativa dos projetos e melhora na qualidade do produto entregue.

Com os resultados da implantação desse plano será possível ter uma posição concreta dos resultados que podem ser obtidos.

A empresa que receberá a proposta apresentou uma enorme necessidade de se planejar devido a quantidade agressiva de projetos e ao mesmo tempo a exigência de qualidade do produto para que não existam impactos que parem a produção da industria ou outras áreas críticas.

O PSP está sendo proposto para auxiliar o desenvolvedor a realizar o trabalho com maior assertividade de prazo e qualidade de produto.

5.2 Trabalhos Futuros

Através do resultado dessa proposta, evoluí-la para chegar a um modelo aderente à empresa. Em trabalhos futuros existe a necessidade de criar um plano de implantação de técnicas para melhoria do desenvolvimento de grandes projetos, envolvendo mais de um desenvolvedor. O desenvolvimento em equipe difere do desenvolvimento individual e exige atenção à outros pontos. Para equipes, o estudo do TSP (*Team Software Process*) poderá ser utilizado como base de estudo.

REFERÊNCIAS

CMMI Product Team. (n.d.). *CMMI for Development - CMMI Institute* . Retrieved 02 25, 2014, from CMMI Institute: http://www.cmmiinstitute.com

FERGUSON, P., HUMPHREY, W. S., KHAJENOORI, S., MACKE, S., & MATVYA, A. (1997, Maio). Results of Applying the Personal Software Process. *IEEE*, pp. 24-31.

GERBALDO, G. B., & SANTOS, J. M. (2014). Aplicação do PSP (Personal Software Process) integrado ao BDD (Behavior-Driven Development) no desenvolvimento de um aplicativo Android. *Computer on the Beach 2014*, 424-426.

HUMPHREY, W. S. (1996). Using A Defined and Measured Personal Software Process. *IEEE Software*, 77-88.

HUMPHREY, W. S. (2000). *The Personal Software Process (PSP)*. Pittsburgh: Carnegie Mellon.

JOHNSON, P. M., & DISNEY, A. M. (1998). The Personal Software Process: A Cautionary Case Study. *IEEE Software*, 85-88.

Pomeroy-Huff, M., Cannon, R., Chick, T. A., Mullaney, J., & Nichols, W. (2009). *The Personal Software Process (PSP) Body of Knowledge, Version 2.0.* Hanscom, MA: Software Engineeting Institute.

PRECHELT, L., & UNGER, B. (2000). An Experiment Measuring the Effects of Personal Software Process (PSP) Training. *IEEE Transactions on Software Engineering*, 465-472.

Project Management Institute. (2013). *A guide to the Project Management Body of Knowledge (PMBOK® Guide)*. Project Management Institute.

ROMBACH, D., MUNCH, J., OCAMPO, A., HUMPHREY, W. S., & BURTON, D. (2007). Teaching disciplined software development. *Science Direct*, 747-763.

RUPP, C., & POHL, K. (2011). *Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam.* Rocky Nook.

WEN-HSIANG, S., NIEN-LIN, H., & WEI-MANN, L. (2010, 10 01). Assessing PSP effect in training disciplined software development: A Plan–Track–Review model. *Elsevier*, pp. 137-148.