UNIVERSIDADE FEDERAL DO PARÁ

INSTITUTO DE CIÊNCIAS EXATAS E NATURAIS

PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

VITOR DE SOUZA CASTRO

Um *Framework* de Práticas Ágeis para apoiar a Implementação dos Processos de Construção da Solução constantes em Modelos de Qualidade de Software

Belém

2015

Vitor de Souza Castro

Um *Framework* de Práticas Ágeis para apoiar a Implementação dos Processos de Construção da Solução constantes em Modelos de Qualidade de Software

Dissertação de Mestrado apresentada para a obtenção do grau de Mestre em Ciência da Computação. Programa de Pós-Graduação em Ciência da Computação. Instituto de Ciências Exatas e Naturais. Universidade Federal do Pará.

Área de Concentração Engenharia de Software.

Orientador Prof. Dr. Sandro Ronaldo Bezerra Oliveira.

Belém

2015

Vitor de Souza Castro

Um *Framework* de Práticas Ágeis para apoiar a Implementação dos Processos de Construção da Solução constantes em Modelos de Qualidade de Software / Vitor de Souza Castro; orientador, Sandro Ronaldo Bezerra Oliveira - 2015.

Dissertação (Mestrado) - Universidade Federal do Pará, Instituto de Ciências Exatas e Naturais, Programa de Pós-Graduação em Ciência da Computação, Belém, 2015.

1. Engenharia de Software. 2 Modelos de Qualidade de Software. I. Oliveira, Sandro R. B orientador. II. Universidade Federal do Pará, Instituto de Ciências Exatas e Naturais, Programa de Pós-Graduação em Ciência da Computação. III. Título.

Vitor de Souza Castro

Um *Framework* de Práticas Ágeis para apoiar a Implementação dos Processos de Construção da Solução constantes em Modelos de Qualidade de Software

Dissertação de Mestrado apresentada para a obtenção do grau de Mestre em Ciência da Computação. Programa de Pós-Graduação em Ciência da Computação. Instituto de Ciências Exatas e Naturais. Universidade Federal do Pará.

Data da aprovação: Belém-Pa. __-__-____

Banca Examinadora

Prof. Dr. Sandro Ronaldo Bezerra Oliveira

Programa de Pós Graduação em Ciência da Computação - UFPA – Orientador

Prof. Dr. Eloi Luiz Favero

Programa de Pós Graduação em Ciência da Computação – UFPA – Membro Interno

Profª. Dr. Marianne Kogut Eliasquevici

Faculdade de Computação - Instituto de Ciências Exatas e Naturais- UFPA – Membro Externo

AGRADECIMENTOS

Agradeço os professores por todo o ensinamento e força para continuar sempre buscando ascenção intelectual.

Ao Prof. Sandro, orientador e amigo, por ter me guiado na busca dos meus objetivos, por ter sido paciente e por ter acreditado no meu trabalho.

Aos meus pais, Gumercindo Marra e Ana Maria, por seus ensinamentos, condutas e educação.

A minha irmã, Vivian Castro, pelo apoio e força para enfrentar os momentos ruins.

A minha futura esposa, Marcele Menezes, que me ajudou e me deu força para continuar sempre em busca dos meus objetivos.

Aos meus amigos e colegas de classe, que sempre estiveram a disposição para ajudar e apoiar.

E por fim, agradeço a Deus.

"Um navio no porto é seguro, mas não é para isso que os navios foram feitos"

William Shedd

RESUMO

Qualidade de software é um tema recorrente nas organizações especializadas. No Brasil, a SOFTEX criou o programa MPS.BR, tendo como objetivo de impulsionar a qualidade do software brasileiro. Adicionalmente, visando maior agilidade nos processos, muitas organizações estão adotando práticas ágeis para apoiar a implementação deste programa de qualidade. Assim, este trabalho visa apresentar um *framework* de práticas ágeis, cujo objetivo é apoiar a implementação dos modelos de qualidade de software, em destaque para o processo de projeto e construção do produto (PCP) do MPS.BR e para a área de processo solução técnica do CMMI. O conjunto de práticas ágeis tem sua origem em diversas metodologias: *eXtreme Programming*, FDD, TDD e Crystal. Dentre as práticas e os métodos ágeis mencionados nesse trabalho, não há um que alcance na totalidade de todos os objetivos do processo de solução técnica e projeto e construção do produto, surgindo assim a necessidade do relacionamento entre as diversas práticas para atendimento desses objetivos. Para o alcance desses resultados foi necessário um estudo aprofundando do tema e por conseguinte o desenvolvimento do mapeamento e do *framework* relacionando as práticas ágeis aos objetivos do processo. Para a avaliação tanto do relacionamento entre as práticas ágeis com os objetivos do processo, quanto do *framework* de práticas ágeis foi utilizada a técnica da revisão por pares.

PALAVRAS-CHAVE: Qualidade de Software, Modelos de Qualidade, MPS.BR, CMMI, Métodos Ágeis, Projeto e Construção do Produto, Solução técnica.

ABSTRACT

Software quality is a recurring theme in the specialized organizations. In Brazil, the SOFTEX created the MPS.BR program, with the aim to boost the quality of Brazilian software. Additionally, seeking greater flexibility in processes, many organizations are adopting agile practices to support implementation of this quality program. This work aims to present a framework of agile practices whose aims to support the implementation of software quality models, in particular the product design and construction process (PCP) of MPS.BR and the technical solution process area of CMMI. The set of agile practices has its origin in many methods: eXtreme Programming, FDD, TDD and Crystal. Among the practices and agile methodologies mentioned in this work, there is no method or agile practices that range in full all the objectives of the process, and with it, the need for relationship between the many practices to meet these goals. To achieve these results we needed a deeper study of the subject and therefore the development of mapping and framework relating the agile practices to the objectives of the process. To evaluate both the relationship between the agile practices and objectives of the process and the framework of agile practices was used to peer review technique.

KEYWORDS: Software Quality, Quality Models, MPS.BR, CMMI, Agile, Agile practices, Design Process and Construction Product and Technical Solution.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1. INTRODUÇÃO

1.1. Contexto

Organizações buscam com o apoio da tecnologia de informação diferenciais competitivos frente aos concorrentes. Para a construção de sistemas computacionais, softwares, aplicativos e outros, há a necessidade de um planejamento, desde a coleta de requisitos, construção da solução, implantação e por fim a manutenção e a continuidade dessa solução.

Empresas de desenvolvimento de software utilizam alguma técnica, mesmo que não seja alinhada a algum modelo de qualidade para realizar o desenvolvimento do software. Atividades e processos da engenharia de software são utilizados para a construção do produto.

No contexto relacionado à atividade de construção do produto, cujo objetivo é projetar, desenvolver e implementar soluções para atender aos requisitos (SOFTEX, 2013), existem inúmeros procedimentos, técnicas e metodologias para apoiar o desenvolvimento dessa atividade, para a obtenção da qualidade e da padronização do processo.

Visando à diferenciação no mercado, organizações estão buscando a implementação de modelos de qualidade, pois apresentam objetivos específicos a serem atingidos com o enfoque na qualidade de software, tendo como exemplo o modelo *Capability Maturity Model Integration* - CMMI (SEI, 2010) e o programa de Melhoria de Processo de Software - MPS.BR (SOFTEX, 2013).

Nesse cenário, a Associação para a Promoção da Excelência do Software Brasileiro – SOFTEX e o *Software Engineering Institute* – SEI apresentam um conjunto de resultados esperados/práticas (um resultado observável do sucesso do alcance do propósito do processo (ISO/IEC, 2008)), para implementar o processo de Projeto e Construção do Produto e da área de processo de Solução Técnica, além disso fornecem

guias com informações sobre formas de atingir esses resultados esperados/práticas, ou seja, guias de implementação.

Por outro lado, influenciados pelo Toyotismo, cujo princípio está baseado na produção "enxuta", os métodos ágeis apresentam-se como uma forma menos burocrática para a aplicação da engenharia de software. No cenário em que organizações estão em busca de processos "leves" e com qualidade, torna-se cada vez mais presente a aplicação dos modelos de qualidade com o apoio dos métodos ágeis.

O SEI (2014) publicou o "CMMI and Agile: Opposites Attract", como forma de evidenciar a possibilidade da implementação dos modelos de qualidade com os métodos ágeis. No MPS.BR também ocorre o mesmo, no trecho do guia de implementação, "*Além dessas técnicas, estórias de usuários também podem ser utilizadas quando se desenvolve utilizando métodos ágeis.*" (SOFTEX, 2013), pode-se notar a importância da citação relacionada à implementação utilizando métodos ágeis.

Essas possibilidades de implementação são descritas de maneira resumidas e na grande maioria das vezes não apresentam claramente a forma como realizar a implementação. Em SOFTEX (2013, p. 36), por exemplo, é descrito no trecho:

"*Um processo de seleção de alternativas possui, basicamente, as seguintes atividades: definição dos objetivos de seleção; estabelecimento dos critérios de seleção; desenvolvimento das soluções a serem avaliadas; avaliações das soluções com base nos critérios pré-estabelecidos e seleção da solução mais adequada.*".

No trecho citado anteriormente não há indicativo de quais objetivos podem ser estabelecidos, como por exemplo, o custo, plataforma de operação. Para organizações que se deparam com esse trecho pela primeira vez pode parecer obscuro o entendimento. Nesse cenário surge a figura da consultoria em processo para detalhar o "como fazer" e isso implica em custos para a organização.

Alinhado aos modelos de qualidade definidos pelo SEI e SOFTEX, o processo de Projeto e Construção do Produto e a área de processo de Solução Técnica foram escolhidas para ser objeto chave desse trabalho, porque inúmeros trabalhos desenvolvidos não exploram esse processo, como melhor detalhado na Seção 1.2, e além disso não há um indicativo para a implementação desse processo pelo uso dos

diversos métodos ágeis disponíveis na comunidade tais como Crystal (Cockburn, 2002), *Feature Driven Development* - FDD (Pressman, 2011) e *Test-Driven Development* - TDD (Beck, 2003). O método *eXtreme Programming* - XP (Beck, 2004), foi adicionado ao trabalho, fruto do resultado da revisão informal da literatura descrita na Seção 3.1, por apresentar um maior número de citações de implementação desse processo.

Assim, esta pesquisa tem o objetivo de apresentar um *framework* de práticas ágeis para apoio aos modelos de qualidade CMMI e MPS.BR, especificamente para a área de processo de Solução Técnica (TS), constante no CMMI, e o processo de Projeto e Construção do Produto (PCP), constante no MPS.BR. Esse framework faz parte do conjunto de estudos constantes no Projeto SPIDER (Oliveira *et al.*, 2011), cujo objetivo é criar um suíte de abordagens de software livre aderente ao MR-MPS-SW para reduzir os custos de implementação deste modelo.

A escolha da área de processo de Solução Técnica e do processo de Projeto e Construção do Produto foi justificada pelo mapeamento realizado em SOFTEX (2012b) dos modelos MR-MPS-SW e CMMI-DEV, contendo o relacionamento dos resultados esperados do MPS aos objetivos e práticas específicas do CMMI.

No contexto da aplicação do modelo MPS.BR, até Julho/2015, apenas 55 empresas já tiveram seus processos implementados nos resultados esperados do nível D, nível em que o processo PCP está inserido. Além disso, 594 empresas, totalizando 92% do total de empresas avaliadas, estão em níveis inferiores de maturidade (SOFTEX, 2015). Apesar do elevado índice de empresas avaliadas em níveis inferiores de maturidade, o percentual restante são de empresas que estão nos níveis A, B, C e D, e precisam manter as práticas do nível D alinhadas aos objetivos do programa MPS.BR para preservar a avaliação. Isso pode denotar que, nos próximos anos, as organizações que continuarem com o programa do MPS.BR estarão atingindo ou mantendo o nível D e com isso há necessidade de fornecer um *framework* de apoio para a implementação desse processo.

Para esse trabalho, a terminologia *framework* está relacionada a um conjunto de objetos, as práticas, que podem ser instanciadas, utilizadas, para atender os resultados do processo e da área de processo constante nos modelos de qualidade. Sendo que esses objetos podem estar associados a outras práticas para atender o mesmo objetivo. Já a

terminologia “construção da solução” será utilizada nesta dissertação para unir as terminologias da área de processo Solução Técnica e do processo de Projeto e Construção do Produto.

Os métodos ágeis utilizados para compor o *framework* foram: Crystal, *eXtreme Programming* - *XP*, *Feature Driven Development* - FDD e *Test-Driven Development* - TDD. A seleção destes métodos esteve pautada nas práticas ágeis, no apoio dessas práticas ao processo de PCP e TS e na baixa incidência de pesquisas relacionada a essas metodologias.

A principal contribuição dessa pesquisa é apresentar para a comunidade acadêmica e industria o *framework* de práticas ágeis que contempla todos os resultados esperados referentes à construção da solução e, além disso, fornecer exemplos de como poderá ser aplicado, por meio da forma de implementação da prática. Adicionalmente, em grande parte dos resultado esperados há o relacionamento entre as práticas ágeis, haja vista que muitas práticas são complementares para apoiar na sua totalidade o resultado esperado.

Para o alcance contemplando todos os resultados esperados e as práticas específicas contido nesse trabalho, o processo de harmonização dos dois modelos (MPS.BR e CMMI) especificamente para o PCP e o TS teve como referência o estudo realizado pela SOFTEX (SOFTEX b, 2012), cujo resultado é um documento que mapeia os dois modelos, apontando as relações entre eles, se satisfazem os requisitos do modelo de origem e se existem áreas de sobreposição com o modelo de destino, o que possivelmente permitirá a reutilização de dados e informações em processos de avaliação e implementação dos modelos.

1.2. Motivação

Em virtude das inúmeras publicações realizadas no contexto de métodos ágeis e modelos de qualidade pode-se destacar “A História da Tahini-Tahini: Melhoria de Processos de Software com Métodos Ágeis e Modelo MPS” (Boria, 2013), um livro que apresenta um caso de aplicação utilizando métodos ágeis e o modelo de qualidade MPS. Além desse trabalho, pode-se destacar nos últimos cinco anos nas principais conferências nacionais o SBQS e o WAMPS a incidência de 14 artigos que possuem

aplicação de métodos ágeis com modelos de qualidade. A Figura 1 traz o percentual de incidência por modelo de qualidade.

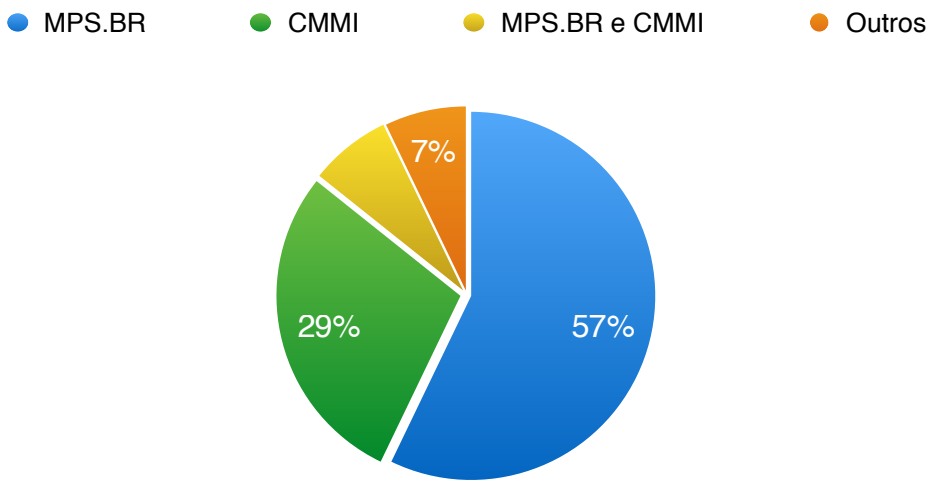

Figura 1. Métodos Ágeis e Modelos de Qualidade

Apesar da incidência do uso dos métodos ágeis para atendimento dos modelos de qualidade, o processo de PCP não possui grandes citações nos artigos nacionais resultantes da pesquisa.

Já no contexto internacional, em uma revisão informal da literatura realizada nas fontes: IEEE, ACM, EL COMPEDEX e SCOPUS; foram selecionados 26 artigos relacionados à aplicação de métodos ágeis apoiando a implementação dos modelos de qualidade. Na Figura 2 é apresentado o percentual de incidência dos métodos ágeis, sendo 52% para o SCRUM, 24% para o XP, 10% para a abordagem Lean e 14% para outras abordagens (Híbrido SCRUM/XP, Híbrido SCRUM/RUP, RUP ágil e RUP).

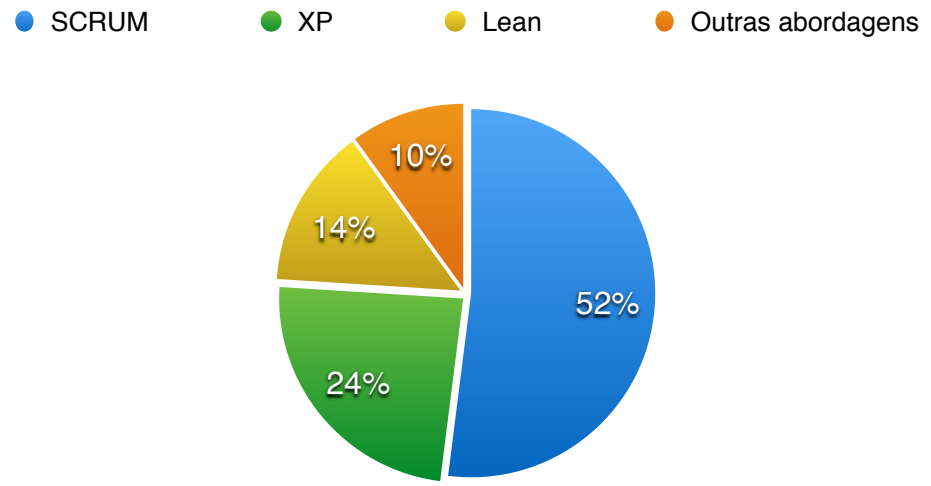

Figura 2. Percentual de Citação dos Métodos no Âmbito Internacional

O número de citações para o SCRUM e XP indicam que são os dois métodos mais utilizados para apoiar a implementação dos modelos de qualidade CMMI e MPS.BR. No SCRUM não há práticas relacionadas à área de processo de solução técnica e ao processo de projeto e construção do produto. Por esse motivo as equipes de desenvolvimento que fazem o uso desse método adicionam práticas ágeis relacionados à construção da solução na SPRINT (um *time-boxed* de um mês ou menos, durante o qual um "Pronto", versão incremental potencialmente utilizável do produto, é criado (Schwaber, 2013)). Já o XP possui diversas práticas que apoiam o processo de PCP e a área de processo de TS, sendo elas: programação por pares, projeto simples, testes, refatoração, integração contínua, padrões de codificação e documentação abrangente.

Os métodos Crystal, TDD e FDD não foram citados no contexto de aplicação de métodos ágeis e modelos de qualidade, porém não é conclusivo descartá-los desse contexto, pois inúmeras práticas contidas nesses métodos possuem relação direta ao processo de PCP e à área de processo de TS.

Um outro fator para fortalecer a motivação do trabalho é a publicação por parte do CMMI e da SOFTEX de orientações inseridas nos guias de implementação para organizações que buscam ou já utilizam métodos ágeis.

1.3. Objetivo Geral

A proposta principal para esse trabalho é apresentar o *framework* de práticas ágeis relacionadas ao processo de projeto e construção do produto e à área de processo de solução técnica. Para o objetivo geral, a proposta é reunir todas as práticas dos métodos ágeis indicados anteriormente e construir um processo de software que apoie a implementação de todos os resultados esperados do processo de PCP (Projeto e Construção do Produto) e as práticas específicas da área de processo de TS (Solução Técnica). Fornecendo para a indústria e academia um conjunto de práticas ágeis com suas definições e formas de aplicação, visando a agilidade no processo de implementação de modelos de qualidade por meio dos métodos ágeis.

1.4. Objetivos Específicos

- Avaliar a literatura relacionada às práticas ágeis e aos modelos de qualidade. Esta avaliação da literatura permitiu o conhecimento aprofundado das práticas ágeis e dos modelos de qualidade, fornecendo insumos para a análise da viabilidade de implementação dos modelos de qualidade pelo uso de práticas ágeis;
- Realizar um mapeamento das práticas ágeis e do processo de construção da solução. O mapeamento indica quais práticas ágeis podem ser utilizadas na implementação direta de cada resultado esperado constante neste processo;
- Avaliar, por meio de revisão por pares, o mapeamento. A avaliação por meio de revisão por pares foi a estratégia usada para ouvir a percepção de pesquisadores especialistas no assunto;
- Projetar um *framework* de práticas ágeis. O *framework* tem como objetivo o atendimento direto ao objetivo geral do trabalho;
- Avaliar, por meio de revisão por pares, o *framework*.

1.5. Metodologia

A metodologia empregada para a estruturação desse trabalho está dividida de acordo com as seguintes fases:

1. Pesquisa e Estudo inicial
    1. Estudo geral de trabalhos na área de Engenharia de Software.
    2. Estudo geral de modelos, normas e guias para processos de software.
    3. Estudo aprofundado de trabalhos na área de Construção da Solução.
2. Revisão informal da Literatura
    1. Estudo teórico do processo de revisão da literatura.
    2. Definição do protocolo para a revisão da literatura.
    3. Realização de pesquisa informal da literatura.
    4. Análise dos resultados obtidos na pesquisa.
3. Estudo do estado da arte do PCP e TS, e dos métodos ágeis

1. Estudo teórico dos processos de construção da solução (PCP e TS).
2. Estudo teórico aprofundado dos métodos ágeis e suas práticas.

4. Mapeamento das práticas dos métodos ágeis com os resultados esperados dos processos de construção da solução
    1. Associação dos resultados esperados/práticas às práticas ágeis.
    2. Construção do mapeamento.
5. Revisão por pares do mapeamento
    1. Elaboração do processo de revisão por pares.
    2. Avaliação junto ao revisor do mapeamento elaborado.
    3. Obtenção de pontos de melhorias no mapeamento.
    4. Construção de artigo contemplando o resultado dessa revisão.
6. Desenvolvimento do *framework* de práticas ágeis
    1. Elaboração dos itens contidos para cada prática ágil e resultado esperado.
    2. Desenvolvimento das práticas ágeis para apoio ao resultado esperado.
7. Revisão por pares do *framework* de práticas ágeis
    1. Definição do processo de revisão por pares para o *framework* de práticas ágeis.
    2. Realização da revisão por pares junto ao revisor selecionado.
    3. Análise e atualização do *framework* mediante às sugestões do revisor.
    4. Construção de artigo científico contemplando o resultado dessa revisão.

Segundo (Silva e Menezes, 2001) existem diversas formas de se classificar uma pesquisa, com base na literatura especializada. Sendo assim, a pesquisa desse trabalho pode ser caracterizado avaliando quatro pontos de vistas: natureza, abordagem do problema, aos objetivos e os procedimentos técnicos.

Relacionado ao ponto de vista da sua natureza, a pesquisa apresenta-se como um pesquisa aplicada, por objetivar a geração de conhecimentos para a aplicação prática a

fim de alcançar a solução de problemas específicos. Já do ponto de vista da abordagem do problema, a pesquisa pode ser definida como quantitativa e qualitativa, pois os aspectos quantitativos foram explorados principalmente na revisão por pares e qualitativamente quando da análise dos dados pelo pesquisador, de forma indutiva.

Quanto aos objetivos, trata-se de uma pesquisa exploratória em virtude do levantamento bibliográfico acerca do tema, proporcionando maior conhecimento e possibilitando a criação de hipótese para o problema em questão.

Por fim, quanto aos procedimentos técnicos, a pesquisa é classificada como uma pesquisa bibliográfica, pois a sua construção foi a partir de materiais disponibilizados nos repositórios de conhecimento, livros, periódicos e eventos científicos.

1.6. Estrutura da Dissertação

Esta dissertação está dividida em seis capítulos. Além do presente capítulo introdutório, o trabalho está organizado da seguinte forma:

- No Capítulo 2 é apresentada a fundamentação teórica, destacando a contextualização do processo de software, melhoria de processo de software, os modelos de qualidade MPS.BR e CMMI, o processo de projeto e construção do produto e a área de processo de solução técnica e, por fim, os métodos ágeis e os trabalhos relacionados;
- No Capítulo 3 é apresentada a revisão informal da literatura, destacando os principais artigos relacionados a pesquisa, bem como os trabalhos relacionados.
- Já no Capítulo 4 é apresentado o *framework* de práticas ágeis, contendo a metodologia da pesquisa, a estruturação do *framework*, o detalhamento do *framework* e a forma de uso;
- No Capítulo 5 a avaliação do *framework* é destacada, descrevendo a contextualização da revisão por pares, a avaliação do relacionamento das práticas ágeis aos resultados esperados e a avaliação do *framework*;
- Por fim, no Capítulo 6 são descritas as considerações finais, contribuições, limitações desta dissertação e os trabalhos futuros pretendidos.

2. FUNDAMENTAÇÃO TEÓRICA

2.1. Processo de Software

O contexto desta dissertação está relacionado à Engenharia de software que, segundo Pressman (2011), é uma tecnologia em camadas que inclui processo, métodos e ferramentas. Já a IEEE (2008) trata a Engenharia de Software como a aplicação de uma abordagem sistemática, disciplinada e quantificável para o desenvolvimento, operação e manutenção do software. Outra definição é de Sommerville (2011), sendo uma disciplina da engenharia que se ocupa de todos os aspectos da produção do software, desde os estágios iniciais de especificação do sistema até a manutenção desse sistema, depois que ele entrou em operação.

Já a terminologia de processo de software, segundo Pressman (2011), define como um arcabouço para as tarefas que são necessárias para construir um software de qualidade, isso é, define a abordagem que é adotada quando o software é elaborado. Em Sommerville (2011) um processo de software é um conjunto de atividades relacionadas que levam a um produto de software.

Para Pressman (2011) um processo genérico é aplicável à grande maioria dos projetos de software, sendo esse processo contemplado pelas seguintes áreas:

- Comunicação: alta comunicação e colaboração entre os envolvidos no projeto, atendendo também o levantamento de requisitos;
- Planejamento: são identificados os riscos prováveis, recursos necessários, cronogramas e os produtos de trabalho;
- Modelagem: criação de modelos para melhorar o entendimento acerca dos requisitos tanto pelo desenvolvedor quanto pelo cliente;
- Construção: geração de código-fonte, podendo ser manual ou automática, e os testes necessários para revelar os erros;
- Implantação: consiste na entrega para o cliente, podendo ser completa ou incremental.

Imerso nesse cenário, a Engenharia de software e o processo de software reúnem um conjunto de ferramentas para o desenvolvimento de software com qualidade. Organizações que objetivam a aplicação do processo de software com qualidade utilizam os modelos qualidade, em destaque CMMI e MPS.BR, para tal propósito. Segundo Bartié (2002), um dos objetivos de se implantar um processo de qualidade de software é estabelecer um processo que garanta e gerencie o nível de qualidade do produto e do processo de desenvolvimento. Portanto, para o processo de desenvolvimento ter qualidade são necessárias abordagens que visam a melhoria do processo de software.

2.2. CMMI e MPS.BR

O *Capability Maturity Model* (CMM) definido pelo *Software Engineering Institute* (SEI) descreve uma estrutura de trabalho que possui todos os elementos necessários para tornar um processo de desenvolvimento de software mais eficiente e controlado (Bartié, 2002).

O CMM baseia-se em cinco níveis de maturidade de processo, sendo inicial, repetitivo, definido, gerenciado e otimizado. Organizações no nível inicial (1) têm processos imprevisíveis e pouco controlado; já no nível repetitivo (2), as tarefas "chaves" podem ser repetidas continuamente, no nível definido (3) o processo é caracterizado como bem entendido; no nível gerenciado (4), o processo já é medido e controlado e; por fim, no nível otimizado (5) o foco é o aperfeiçoamento do processo. Por ser uma estrutura em estágio, a organização não pode estar no nível (3) sem antes passar pelo (1) e (2).

Com a finalidade de integrar diversos modelos criados, o SEI evoluiu o CMM e desenvolveu o CMMI, com o objetivo de servir como guia para a melhoria de processos na organização (SEI, 2010). No Brasil, essa iniciativa foi realizada pela SOFTEX por meio do MPS.BR que tem o objetivo de impulsionar a melhoria da capacidade de desenvolvimento de software e serviços nas empresas brasileiras (SOFTEX, 2012a).

O CMMI possui quatro disciplinas: Engenharia de sistemas, Engenharia de software, Desenvolvimento e integração de produtos e processo, e fontes de aquisição

(SEI, 2010). Nesse modelo há duas representações: “por estágio” e “contínua”. Na representação por estágio, a mesma abordagem do CMM é utilizada, sendo dividida em cinco níveis de maturidade (inicial, gerenciado, definido, gerenciado quantitativamente, em otimização). Já na representação contínua é possível selecionar a sequência de melhorias que convém aos objetivos do negócio.

O CMMI-DEV – *CMMI for Development* (SEI, 2010) está estruturado em 22 áreas de processo (como visto na Quadro 1) que apresenta um conjunto de práticas que, quando executadas coletivamente, satisfaz um conjunto de objetivos. Para cada área de processo há objetivos específicos que identificam características únicas que descrevem o que deve ser implementado para satisfazer a área de processo. Sendo que para cada objetivo específico, são descritas práticas específicas, atividades importantes para atingir um determinado objetivo específico. A área de processo Solução Técnica encontra-se no nível 3 do CMMI.

Quadro 1. Áreas de Processo do CMMI-DEV

Áreas de Processo	**Nível de Maturidade**
Implantação de Inovações na Organização (OID)	Nível 5 - Em otimização
Análise e Resolução de Causas (CAR)	
Gestão Quantitativa de Projeto (QPM)	Nível 4 - Gerenciado Quantitativamente
Desempenho dos Processos da Organização (OPP)	
Solução Técnica (TS)	Nível 3 - Definido
Validação (VAL)	
Verificação (VER)	
Gestão de Riscos (RSKM)	
Desenvolvimento de Requisitos (RD)	
Treinamento na Organização (OT)	
Integração de Produto (PI)	
Definição dos Processos da Organização +IPPD (OPD +IPPD)	
Foco nos Processos da Organização (OPF)	
Gestão Integrada de Projeto +IPPD (IPM +IPPD)	
Análise e Tomada de Decisões (DAR)	

Áreas de Processo	Nível de Maturidade
Gestão de Contrato com Fornecedores (SAM)	Nível 2 - Gerenciado
Gestão de Requisitos (REQM)	
Garantia da Qualidade de Processo e Produto (PPQA)	
Planejamento de Projeto (PP)	
Monitoramento e Controle de Projeto (PMC)	
Medição e Análise (MA)	
Gestão de Configuração (CM)	

Fonte: SEI, 2010

Já o programa MPS.BR possui o MR-MPS-SW (Modelo de Referência MPS para Software), que define níveis de maturidade que são uma combinação entre processos e sua capacidade. Na Quadro 2 são apresentados os sete níveis de maturidade definidos no MR-MPS-SW e seus processos. Para cada nível de maturidade são definidos processos e para cada processo são indicados os propósitos e os resultados esperados. O propósito do processo define o objetivo geral da execução do processo. Convém que a implementação do processo forneça benefícios tangíveis aos envolvidos (ISO/IEC, 2004). O processo de projeto e construção do produto encontra-se no nível D do MPS.BR.

Quadro 2. Níveis de Maturidade do MR-MPS-SW

Níveis de Maturidade	Processos
Nível G - Parcialmente Gerenciado	Gerência de projetos - GPR Gerência de Requisitos - GRE
Nível F - Gerenciado	Medição – MED Garantia da Qualidade – GQA Gerência de Portfólio de Projetos – GPP Gerência de Configuração – GCO Aquisição – AQU
Nível E - Parcialmente Definido	Gerência de Projetos – GPR (evolução) Gerência de Reutilização – GRU Gerência de Recursos Humanos – GRH Definição do Processo Organizacional – DFP Avaliação e Melhoria do Processo Organizacional – AMP

Níveis de Maturidade	Processos
Nível D - Largamente Definido	Verificação – VER Validação – VAL Projeto e Construção do Produto – PCP Integração do Produto – ITP Desenvolvimento de Requisitos – DRE
Nível C - Definido	Gerência de Riscos – GRI Desenvolvimento para Reutilização – DRU Gerência de Decisões – GDE
Nível B - Gerenciado Quantitativamento	Gerência de Projetos – GPR (evolução)
Nível A - Em otimização	Não possui processos específicos

Fonte: SOFTEX, 2013

Para o contexto deste trabalho, a área de processo Solução Técnica (TS) do modelo CMMI e o processo de Projeto e Construção do Produto (PCP) do modelo MPS.BR foram escolhidas pois possuem papel importante dentro dos modelos de qualidade, haja vista que atendem diretamente a área de Construção da Solução (Pressman, 2011), além disso fornecem insumos para os processos de Verificação e Validação, processos esses que complementam a área de Construção.

2.3. Os Processos de Projeto e Construção do Produto e de Solução Técnica

A área de processo da Solução técnica tem o objetivo de fornecer subsídios para projetar, desenvolver e implementar soluções para os requisitos. Soluções, *designs* e implementações englobam produtos, componentes de produto e processos de ciclo de vida relacionados ao produto, seja de forma isolada ou em conjunto, conforme apropriado (SEI, 2010). Na Quadro 3 são apresentadas as 8 práticas específicas e seus objetivos desta área de processo.

Quadro 3. Práticas Específicas da Área de Processo de TS

Práticas Específicas	Objetivo
SP 1.1 Desenvolver Soluções Alternativas e Critérios de Seleção	Identificar e analisar soluções alternativas para possibilitar a seleção de uma solução equilibrada em termos de custo, prazo e desempenho técnico ao longo da vida do produto.

Práticas Específicas	Objetivo
SP 1.2 Selecionar Soluções de Componentes de Produto	Selecionar os componentes de produtos que melhor satisfazem aos critérios. Os requisitos detalhados são gerados a partir das alternativas selecionadas e utilizados para gerar o *design* do componente do produto.
SP 2.1 Desenvolver o Design do Produto ou dos Componentes de Produto	Desenvolver o design do produto contemplando as principais funcionalidades, características do produto e sua arquitetura para o design preliminar e estrutura com as funcionalidades dos componentes do produto para o design detalhado.
SP 2.2 Estabelecer Pacote de Dados Técnicos	Desenvolver um descrição detalhada do produto ou do componente do produto à medida que ele é desenvolvido. Também deve inclui diagramas, especificações, descrição do design e do modelo de dados, padrões, requisitos de desempenho e disposições relativas à garantia da qualidade.
SP 2.3 Projetar Interfaces Utilizando Critérios	A atividade de projetar interfaces deve considerar as linhas de serviços de comunicação do software, origem e destino e características elétricas, mecânicas e funcionais para o hardware. Os critérios para a definição da interface de ser ao menos investigado, para se certificar da aplicabilidade.
SP 2.4 Analisar Alternativas: Desenvolver, Comprar ou Reusar	Realizar uma análise da necessidade de projeto, iniciando durante a primeira interação de design, visando a tomada de decisão para desenvolver, adquirir ou reusar o produto.
SP 3.1 Implementar *Design*	Implementar o design como um componente de produto, devendo incluir alocação, refinamento e verificação de cada componente de produto.
SP 3.2 Elaborar Documentação de Suporte ao Produto	Elaborar e manter a documentação utilizada para instalar, operar e manter o produto.

Fonte: Elaborado pelo Autor

Já o processo de PCP (Projeto e construção do Produto) tem como propósito o de projetar, desenvolver e implementar soluções para atender aos requisitos. Para isso a SOFTEX (2013) definiu oito resultados esperados. Na Quadro 4 são apresentados os resultados esperados do processo de projeto e construção do produto e os objetivos.

Quadro 4. Resultados Esperados do Processo de PCP

Resultado Esperado	Objetivo
PCP1. Alternativas de solução e critérios de seleção são desenvolvidos para atender aos requisitos definidos de produto e componentes de produto.	Identificar possíveis soluções para a construção do produto com base em um processo de seleção de alternativas possui, basicamente, as seguintes atividades: definição dos objetivos de seleção; estabelecimento dos critérios de seleção; desenvolvimento das soluções a serem avaliadas; avaliações das soluções com base nos critérios pré-estabelecidos e seleção da solução mais adequada.
PCP2. Soluções são selecionadas para o produto ou componentes do produto, com base em cenários definidos e em critérios identificados	Realizar a seleção das soluções para o produto ou componente do produto, tendo como base cenários e critérios para que se possa selecionar a solução mais adequada.
PCP3. O produto e/ou componente do produto é projetado e documentado	Realizar e documentar o *design* do produto, essa documentação gerada servirá como base para o processo de desenvolvimento do produto
PCP4. As interfaces entre os componentes do produto são projetadas com base em critérios predefinidos	Definir critérios de seleção, identificar e projetar as interfaces, meio para o estabelecimento de comunicação entre os componentes do produto ou serviços.
PCP5. Uma análise dos componentes do produto é conduzida para decidir sobre sua construção, compra ou reutilização	Definir como será concebido o produto, podendo ser por desenvolvimento interno, aquisição ou reutilização.
PCP6. Os componentes do produto são implementados e verificados de acordo com o que foi projetado	Implementação do produto de acordo com o *design* estabelecido e sua documentação deve ser desenvolvida. A inspeção na implementação verificará se o componente do produto foi implementado conforme o projetado
PCP7. A documentação é identificada, desenvolvida e disponibilizada de acordo com os padrões estabelecidos	Identificar qual a documentação para o projeto e para o usuário final.
PCP8. A documentação é mantida de acordo com os critérios definidos	Manter a documentação consistente e organizada do projeto.

Fonte: Elaborado pelo Autor

A SOFTEX (2012b) realizou o mapeamento entre a área de processo Solução Técnica e o processo de Projeto e Construção do Produto, identificando que três resultados esperados são equivalentes às práticas específicas. O objetivo do

mapeamento realizado pela SOFTEX é identificar quais lacunas há entre os dois modelos a fim de prover, por meio do mapeamento, o relacionamento entre as práticas específicas e os resultados esperados, possibilitando a implementação dos dois modelos de qualidade conjuntamente. No mapeamento entre a TS e o PCP, nos casos em que não há equivalência direta entre a prática específica e o resultado esperado, a associação entre dois resultados esperados ou ampliação do resultado esperado já suportava as práticas específicas do TS.

Pode-se destacar as considerações para a não equivalência entre o mapeamento entre (SOFTEX, 2012b, p. 58): o PCP6 e a SP3.1, "*Enquanto o MR-MPS-SW exige a implementação e a verificação dos componentes do produto de acordo com o que foi projetado, o CMMI-DEV exige apenas a implementação do design, Só há referência a verificação dos componentes do produto nas subpráticas, o que não constitui uma obrigatoriedade.*"; e o PCP2 e a SP1.2, "*Enquanto o MR-MPS-SW exige a seleção de soluções para o produto ou componentes do produto com base em cenários definidos e em critérios identificados, o CMMI-DEV exige apenas a utilização de critérios. Só há referência ao uso de cenários para avaliar se as soluções atendem aos critérios nas subpráticas, o que não constitui uma obrigatoriedade.*". Isso demostra que apesar das objetivos não serem exatamente iguais, o resultado esperado alcançado, permite atender a prática específica.

Já entre o PCP7 e a SP2.2, "*O CMMI-DEV exige em SP 2.2 que a documentação associada ao projeto (design) seja estabelecida e mantida, mas sem fazer referência a documentação para o usuário final e aos padrões estabelecidos.*", pode-se considerar que a implementação do PCP7 contempla mais elementos que a implementação do SP2.2.

Por fim, entre o PCP8 e as práticas SP2.2 e SP3.2, "*PCP 8 complementa PCP 7 no atendimento à exigência de SP 2.2 de manutenção da documentação. Entretanto, o CMMI-DEV não faz referência a critérios.*", pode-se considerar que a referência para os critérios definidos no PCP8 são adicionais para a implementação das práticas SP2.2 e SP3.2.

2.4. Métodos Ágeis

Para a implementação desses modelos de qualidade surge a necessidade de se fornecer metodologias que apoiem essa implementação, buscando a eficiência nos processos.

O método ágil, além de ser um paradigma novo, promete melhorias na produção de software e em sua qualidade (Koscianski, 2006). Já Pressman (2011) afirma que o processo ágil atende a três suposições-chave: é difícil prever antecipadamente quais requisitos de software vão coexistir; é difícil prever o quanto de projeto é necessário antes que a construção seja usada para comprovar o projeto; e que análise, projeto, construção e testes não são tão previsíveis.

Em 2001 foi assinado o Manifesto Ágil que traz toda a essência do desenvolvimento ágil de software na seguinte filosofia: "Estamos descobrindo melhores modos de desenvolvimento de software fazendo-o e ajudando o outro a fazê-lo. Por meio desse trabalho passamos a valorizar:

Indivíduos e interações em vez de processos e ferramentas.

Software funcionando em vez de documentação abrangente.

Colaboração do cliente em vez de negociação de contratos.

Resposta a modificações em vez de seguir um plano.

Isto é, ainda que haja nos itens à direta, valorizamos mais os itens a esquerda." (Manifesto Ágil, 2001).

O manifesto trouxe o conceito da agilidade para o desenvolvimento de software, que já havia surgido com os métodos: Crystal, *eXtreme Programming* (XP), LEAN, *Test Driven Development* (TDD), *Feature Driven Development* (FDD) e SCRUM (Schwaber, 2013); que tiveram o seu surgimento antes da publicação deste manifesto. Para o contexto de aplicação desse trabalho, que se dá no processo de Projeto e Construção do Produto e na área de processo de Solução Técnica, dentre os métodos citados o SCRUM e o LEAN foram retirados por não apresentarem práticas diretamente relacionadas a esse processo, ou seja, práticas da engenharia de software relacionadas à construção da solução.

Segundo Cockburn (2002), o Crystal é uma família de métodos que enfatiza a "manobrabilidade", tendo como principal objetivo entregar softwares úteis funcionando. Os elementos centrais do Crystal são os papéis, os padrões de processo, produtos de trabalho e práticas específicas. Essa metodologia apresenta a prática da reunião de Exploração 360º, cujo objetivo é a realização do planejamento das tecnologias, metodologias e processos a serem utilizados no projeto. Além dessa prática, pode-se destacar o Esqueleto ambulante, a Rearquitetura incremental e a programação lado-a-lado.

Já o *eXtreme Programming* (Beck, 2004) define a codificação como a principal atividade de um projeto de software. É um método para times pequenos e médios, que desenvolve software com os requisitos vagos e que se modifiquem rapidamente. A prática Projeto simples tem a estratégia de identificar a solução mais simples e começar o desenvolvimento. A programação por pares, prática desse método, é uma técnica utilizada para a construção do produto sendo que duas pessoas estão trabalhando em um único problema juntas. Além das duas práticas mencionadas, o XP possui práticas como: Refatoração, Integração contínua, Padrões de codificação dentre outras.

O *Test Driven Development* (desenvolvimento dirigido por testes) é um método de desenvolvimento de software no qual deve-se escrever um teste unitário - teste a nível de componente - antes de escrever o código (Beck, 2003). No TDD a prática de projetar a solução visa a realização da construção de cenários de testes para a verificação do componente do produto.

Por fim, o *Feature Driven Development* (FDD) é um modelo prático de processo para a Engenharia de software orientado à objeto (Retamal, 2008). Possui a característica de desenvolvimento que as funções valorizadas pelo cliente devem ser passíveis de entrega em duas semanas ou menos. A *feature* é o principal objeto desse método, sendo que a *feature* representa a funcionalidade ou tarefa a ser desenvolvida no produto. No modelo do FDD pode-se destacar a prática Detalhar por *feature,* cujo propósito é, para cada *feature,* identificar e detalhar qual a melhor solução para o projeto ou *design* da *feature.*

3. REVISÃO INFORMAL DA LITERATURA

3.1. Metodologia

Uma revisão informal da literatura foi realizada nas principais bases de conhecimento do cenário nacional e internacional. Essa etapa da pesquisa é justificada para ampliar o conhecimento do pesquisador no que tange o assunto da pesquisa e observar quantitativamente e qualitativamente a produção pela comunidade científica e pela indústria de software. O principal benefício dessa etapa foi a identificação dos principais eixos de discussão e possíveis *gaps* do estudo.

O protocolo de revisão informal da literatura foi definido em um conjunto de fases visando a melhor estruturação da pesquisa, são elas: Definir os objetivos, a questão de pesquisa, os critérios de seleção e as fontes de dados; Compor a *string* de busca; Definir os critérios para seleção dos artigos; e Compilar os dados. A revisão informal tem o propósito de alcançar os seguintes objetivos:

- Identificar artigos e trabalhos científicos relacionados ao tema em questão;
- Possibilitar a análise das abordagens e metodologias utilizadas;
- Apresentar um conjunto extenso de práticas utilizadas pela indústria e academia no que diz respeito à área de processo de solução técnica e ao processo de projeto e construção do produto.

Para nortear a revisão informal da literatura e com intuito de investigar as propostas, a questão de pesquisa foi estabelecida: “Quais as metodologias ou práticas ágeis são utilizadas para a implementação do processo solução técnica e projeto e construção do produto ?”.

Além dessa questão principal pode-se identificar uma questão secundária: “Dentre as metodologias ágeis encontradas, quais contemplam a implementação de modelos de qualidade ?”.

Outra etapa desse protocolo de revisão informal é a escolha dos critérios para a seleção das fontes de dados. Para a escolha, foram estabelecidos os seguintes critérios:

- Disponibilidade de consulta via web;
- Disponibilidade de artigos em inglês ou português;
- Disponibilidade na íntegra dos artigos por meio de buscas na internet ou pelo domínio da Universidade Federal do Pará.

As fontes selecionadas foram: IEEE, ACM, EL COMPEDEX e SCOPUS. Além das bases internacionais, foram utilizados os anais do Simpósio Brasileiro de Qualidade de Software (SBQS) e o Workshop Anual do MPS (WAMPS) para contemplar os trabalhos relacionados ao modelo MPS, haja vista que, por exemplo, no WAMPS, os trabalhos contidos nos anais são na totalidade relacionados ao modelo MPS. Além disso, muitos dos trabalhos contidos nos anais do SBQS e do WAMPS não estão disponíveis nas bases internacionais pesquisadas e, sendo o MPS.BR um programa desenvolvido para atender principalmente as organizações no Brasil, o volume de trabalhos em português é mais significativo.

Para a realização das buscas foram utilizados dois métodos, o automático e o manual. O procedimento manual foi aplicado aos anais do SBQS e do WAMPS, haja vista que ambos não possuem máquina de busca disponível. Já o procedimento automático foi aplicado nas bases internacionais, isso por meio de *strings* de buscas, executadas na máquina de busca de cada base.

Para a execução do procedimento de busca foi necessária a definição de um conjunto de palavras-chave e a combinação delas formando assim uma *string* de busca. A *string* de busca tem por intuito filtrar os artigos com o objetivo de precisar o conteúdo esperado. Baseada na pergunta central, definida nessa seção, foram identificadas as seguintes palavras-chave para cada contexto:

- No contexto da Engenharia de software foram identificada as seguintes palavras:
    - Em português: software, processo, PCP, processo de projeto de construção do produto, desenvolvimento de software e solução técnica;
    - Em inglês: *software, process, PCP, project construction process of the product, software development e technical solution.*
- No contexto dos métodos ágeis foram identificada as seguintes palavras:

- Em português: metódos ágeis, ágil, agilidade, abordagem ágil e prática ágil;
- *o* Em inglês: *agile methods, agile, agility, agile approach e agile practice.*

- No contexto dos modelos de qualidade foram identificadas as seguintes palavras:
  - Em português: MPS,MPS.BR,CMMI e modelos de qualidade.
  - Em inglês: MPS,MPS.BR,CMMI e *quality models*.

Após a identificação das palavras-chave, a *string* de busca foi formada por: (palavras-chave no contexto da engenharia de software) AND (palavras-chave no contexto dos métodos ágeis) AND (palavras-chave no contexto dos modelos de qualidade)

Para a seleção do artigos após a execução das pesquisas, foi realizada a leitura do Resumo/*Abstract* para refinamento da seleção. O critério de seleção da leitura dos Resumos foi: relacionar modelos de qualidade com métodos ágeis e possuir indicativos do uso de práticas para a implementação do processo solução técnica ou projeto e construção do produto.

3.2. Resultados

Os resultados quantitativos da revisão informal são apresentadas no quadro 5. O total de artigos selecionadas foi 40 sendo a fonte de pesquisa IEEE com o maior quantitativo de artigos selecionados.

Dentre os artigos selecionados, 35% dos artigos selecionados a partir das fontes nacionais (Brasil) e 65% nas fontes internacionais. Sobre as metodologias utilizadas para apoiar modelos de qualidade, no âmbito nacional, 60% das citação foram sobre o SCRUM, 25% para o XP e 15% para outras abordagens (Modelagem ágil, Open UP e RUP).

Já no contexto dos artigos internacionais, 52% das citações para o SCRUM, 24% para o XP, 10% para a abordagem Lean e 14% para outras abordagens (Híbrido SCRUM/XP, Híbrido SCRUM/RUP, RUP ágil e RUP).

Quadro 5. Quantidade de artigos por fonte

Fontes	**Quantidade de Artigos**
ACM	6
EL COMPEDEX	1
IEEE	18
SCOPUS	1
SBQS	10
WAMPS	4
TOTAL	**40**

Dentre os trabalhos selecionados na revisão, pode-se destacar o trabalho de Oliveira *et al.* (2008), que apresenta uma implementação no nível F com as metodologias SCRUM e XP, mas não são evidenciadas quais as práticas e a sua relação com os resultados esperados, além disso, não se faz presente uma metodologia para a aplicação dos dois métodos. Já no trabalho de Santos *et al.* (2008), foi adotado um único método ágil para o mapeamento com as atividades da ISO9001:2000, não sendo apresentadas outras possibilidades para o atendimento da norma nos casos em que o método não obteve a aderência.

O trabalho de Catunda *et al.* (2011) está centrado na implantação do MPS.BR em uma organização. A abordagem das práticas é muito ampla no sentido que não houve a associação direta das práticas aos resultados esperados. Na etapa de análise, projeto e construção do produto não são evidenciadas práticas ou metodologia para a implementação. Outro ponto a destacar é que o método SCRUM foi utilizado e por se tratar de uma organização específica, os outros métodos ágeis usados neste trabalho não foram citados, podendo assim não ser possível avaliar se outras metodologias poderiam ser aplicadas com o objetivo da alcançar os resultados esperados.

Em Ramasubbu (2009) o objetivo é a investigação em equipes que escolheram a implantação do CMMI com métodos ágeis. Os dados coletados foram de 112 projetos e 34 delas utilizaram métodos ágeis (RUP ágil, XP e SCRUM). A avaliação do processo de implantação fez-se uso de cinco variáveis: Produtividade (Código fonte por pessoa-hora); Densidade de defeitos; Reutilização; Retrabalho (tempo gasto para correção de *bugs*) e Esforço em gestão de projeto. Aspectos relacionados ao grau de envolvimento

do cliente, conhecimento específico do cliente e familiaridade da equipe com a tecnologia a ser utilizada também foram analisados. Como resultado, foi identificado empiricamente que ao conduzir processos formais com o uso de práticas ágeis pode levar resultados de desempenho superior. Além disso, a extração de métrica para avaliar o processo de implantação do CMMI com os métodos ágeis.

No trabalho de Prikladnicki *et al.* (2010) são apresentadas práticas para a implementação do MPS.BR com métodos ágeis. O trabalho foi limitado a dois métodos ágeis e não se faz associação dos resultados esperados a cada prática. Além disso, há apenas informações superficiais das práticas para cada nível de implementação e não há uma perspectiva metodológica para a aplicação em conjunto das duas metodologias na implantação de cada nível do modelo MPS.BR.

Assim, com esta revisão fica claramente nítida a necessidade de pesquisas e experimentações na academia e na indústria de software sobre o uso integrado de métodos ágeis e modelos de qualidade para o alcance dos objetivos dos processos de Engenharia de Software, uma vez que dentre todos os trabalhos relatados sobre implementação de programas de qualidade, cerca de 60% tratavam da associação entre estes temas de pesquisa (Castro *et al.*, 2015). Isso pode ajudar a comunidade a desmistificar o seguinte pensamento: o desenvolvimento ágil busca tratar o processo de produção de software de maneira diferenciada, uma vez que entende que se trata de um processo criativo e que produz um bem intangível; já os modelos de qualidade baseiam-se no pensar fordista, tratando a produção de software de maneira parecida com os produtos manufaturados.

Em Castro *et al.*, 2015 estão descritos os demais artigos que não foram apresentados nessa seção. A principal contribuição da revisão foi a identificação de quais métodos ágeis, de acordo a literatura especializada, podem apoiar a implementação de modelos de qualidade.

O diferencial desse trabalho face aos resultados da revisão informal da literatura está inicialmente associado ao número de métodos utilizados, sendo que a maior parte dos trabalhos apresenta um ou dois métodos e nesse trabalho são apresentadas quatro. Outro diferencial está na associação das práticas ágeis diretamente para apoio a cada resultado

esperado ou prática específica constantes nos modelos de qualidade. Por fim, é apresentado neste trabalho um modelo para implementação de cada prática no contexto de apoio aos resultados esperados ou práticas específicas, sendo esse modelo descrito no *framework* de práticas ágeis.

4. *FRAMEWORK* DE PRÁTICAS ÁGEIS

4.1. Metodologia

A metodologia para o desenvolvimento do *framework* apresentou as seguintes atividades, como pode ser vista na Figura 3: Revisão da Literatura (descrita no capítulo 3); Estudo do estado da arte do PCP e dos métodos ágeis; Mapeamento das práticas dos métodos ágeis com os resultados esperados do PCP; Revisão por pares do mapeamento; Desenvolvimento do *framework*; e Revisão por pares do *framework*.

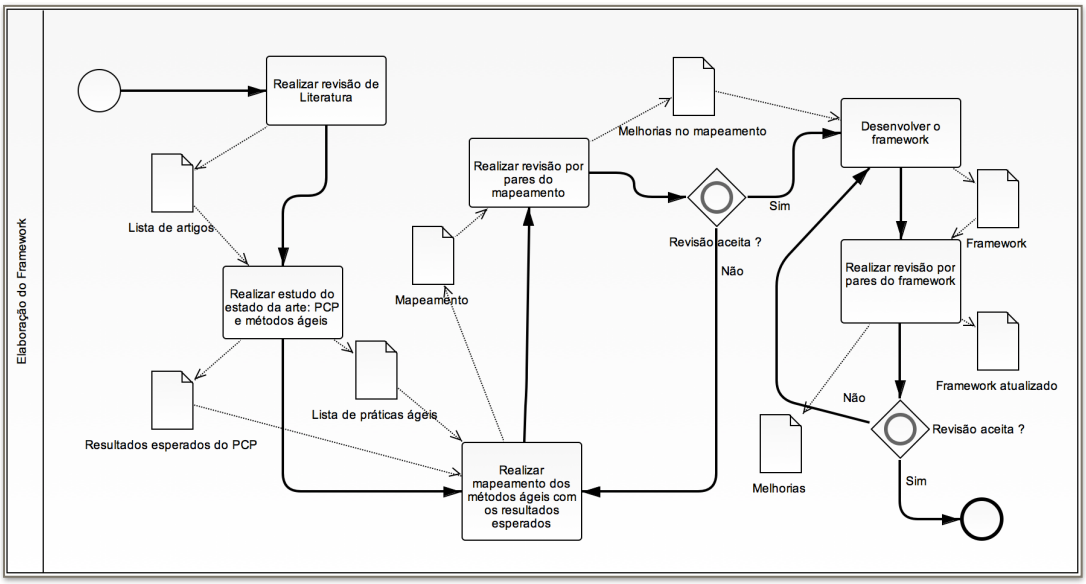

Figura 3. Metodologia de Desenvolvimento do *Framework*

Como resultado da revisão da literatura, um estudo do estado da arte sobre os métodos ágeis e dos processos de PCP e TS foi realizado. Dentre os métodos encontrados, foram identificados aqueles que possuem em suas práticas atividades relacionadas à implementação dos processos de PCP e TS. A partir dos métodos selecionados, foi realizado um estudo aprofundado de cada um dos métodos tendo como base as bibliografias e os artigos selecionados na etapa de revisão.

Além de se estudar os métodos ágeis, também foi necessário realizar um estudo do estado da arte sobre os processos de PCP e TS. Para a realização desse estudo foram utilizadas as referências da SOFTEX sobre o processo e dos guias definidos pelo SEI, além dos artigos selecionados na etapa de revisão. Esse estudo é justificado para a

identificação de quais resultados são esperados por cada uma dessas referências, fornecendo subsídios para a realização do mapeamento das práticas ágeis com o PCP e o TS.

Após o estudo sobre os métodos ágeis e dos processos de PCP e TS, foram realizados uma análise e um mapeamento entre as práticas que constam nos métodos ágeis com os resultados esperados e com as práticas específicas deste processo, no sentido de avaliar se a prática ágil ou um conjunto de práticas contemplam a execução do resultado esperado. Esse mapeamento será apresentado com maior detalhamento na Seção 4.2.

Para avaliar o mapeamento foi realizada a revisão por pares, que objetiva: avaliar os critérios utilizados para a comparação dos modelos; verificar a aderência entre os elementos presentes nas estruturas dos modelos, quanto a sua correspondência e interpretação dos elementos; e analisar se as considerações feitas esclarecem suas atribuições (Pavan, 2007). Na seção 5.3 será apresentado o processo de revisão por pares do mapeamento.

Nessa etapa do mapeamento, o principal benefício é a identificação de quais práticas ágeis poderão ser utilizadas no *framework*, e, além disso, quais associações entre as práticas poderão ser realizadas para o alcance dos resultados previstos no PCP e no TS.

O desenvolvimento do *framework* foi realizado tendo como base o resultado da revisão por pares e as demais atividades anteriores. Essa atividade teve como principais elementos: a estruturação de um formato de apresentação do *framework*; orientação para cada resultado esperado ou prática específica da implementação por meio das práticas selecionadas dos métodos ágeis contemplados neste trabalho; e, por fim, a elaboração de um relacionamento entre as práticas para o mesmo resultado do processo, cujo o objetivo é contemplar na totalidade dos resultado dos processos de PCP e TS a partir da junção de duas ou mais práticas dos métodos ágeis.

4.2. Relacionamento das Práticas Ágeis aos Resultados do Processo

Após a realização do estudo do estado da arte do PCP e dos métodos ágeis, foi necessário identificar os relacionamentos das práticas ágeis aos resultados do processo.

A Quadro 6. apresenta para cada resultado esperado quais práticas ágeis possuem relacionamentos, visando apoiar a implementação do resultado esperado.

Quadro 6. Relacionamento das Práticas aos Resultados do Processo

PCP1	Exploração 360º (Crystal), Rearquitetura incremental (Crystal) e Projeto Simples (XP)
PCP2	Esqueleto ambulante (Crystal) e Detalhar por feature (FDD),
PCP3	Padrões de codificação (XP), Projeto Simples (XP), Rearquitetura incremental (Crystal) e Detalhar por feature (FDD)
PCP4	Detalhar por feature (FDD) e Esqueleto ambulante (Crystal),
PCP5	Modelagem de Objeto conceitual (FDD) e Exploração 360º (Crystal),
PCP6	Construir por feature (FDD), Programação por Pares (XP), Testes (XP), Refatoração (XP), Integração contínua (XP), Esqueleto ambulante (Crystal), Programação lado a lado (Crystal) e Projetar solução (TDD),
PCP7	Documentação abrangente (XP), Manual do usuário (Crystal), Construir por feature (FDD) e Projetar solução (TDD),
PCP8	Documentação abrangente (XP), Detalhar por feature (FDD), Construir por feature (FDD) e Projetar solução (TDD)

No PCP1, foram identificadas duas práticas da metodologia Crystal: a Exploração 360º e a Rearquitetura Incremental. Para uma possível implementação da primeira prática é necessário desenvolver a reunião de Exploração 360º contendo o planejamento das tecnologias e metodologias e processos a serem utilizados no projeto. O uso da metodologia *shaping*, técnica utilizada para identificar pontos fortes e fracos da organização, pode ser expandido para a análise de alternativas de soluções para o projeto. Já para a prática de Rearquitetura Incremental é necessário avaliar alternativas de solução à medida que o desenvolvimento vai evoluindo, podendo, assim, ao se deparar com uma boa solução, atualizar a arquitetura do projeto de forma incremental.

Ainda para o PCP1, no XP a prática Projeto Simples objetiva identificar a solução mais simples e começar o desenvolvimento - simplicidade presumida. Assim, deve-se avaliar o menor investimento inicial que chegue na solução. Para determinar a solução, deve-se ter o pensamento de que cada parte do sistema precisa justificar a sua existência.

Já no PCP2, a prática de Esqueleto Ambulante remete ao desenvolvimento de um pequeno modelo para cada solução, esse modelo pode ser um protótipo funcional, a ser analisado para assegurar que haja a possibilidade da aplicação da solução. Com a prática Detalhar por *Feature* e de posse de um modelo abrangente, por exemplo um diagrama de classe ou modelo entidade relacionamento, para cada *feature* é necessário identificar e detalhar qual a melhor solução para o projeto da *feature*. O propósito é adequar e validar a solução para cada *feature*.

No PCP 3, a prática de Padrões de Codificação da XP possibilita a definição de um documento de arquitetura do projeto contendo os padrões a serem utilizados. Já a prática Projeto Simples consiste em conceber o projeto de maneira simples, ou seja, o *design* do projeto deve acompanhar a simplicidade, a complexidade do projeto sem necessidade deve ser retirada. A Rearquitetura Incremental propõe que à medida que o *design* for sofrendo alterações, o projeto arquitetural deverá acompanhar as modificações e ser devidamente documentado. Ao passo que o processo desenvolvimento vai ocorrendo, a adequação no modelo é necessária, sendo vital a sua documentação para garantir posterior manutenções. Já na prática Detalhar por *Feature*, o modelo abrangente fornece a base para o detalhamento da funcionalidade. Nessa prática a *feature* deve ser projetada e documentada. Alterações no modelo abrangente podem ocorrer em virtude de nova *feature* no projeto, sendo necessária a atualização do modelo.

No PCP 4, na prática Detalhar por *Feature* as interfaces e suas dependências devem ser identificadas a cada *feature* detalhada. Deve ser projetado como dar-se-á o comportamento diante dos outros componentes do produto, podendo fazer o uso dos diagramas de classes e de componente da UML. A prática Esqueleto Ambulante propõe o desenvolvimento de pequenos componentes e sua posterior integração aos demais componentes por meio do design evolutivo. As interfaces são identificadas à medida que os componentes da arquitetura são desenvolvidos.

No PCP5, a prática Modelagem de Objeto Conceitual tem o seguinte propósito: por meio da modelagem conceitual são extraídas as informações para subsidiar a decisão da construção, compra ou reuso de componentes. O modelo conceitual pode ser o diagrama

de classe, modelo de entidade e relacionamento ou diagrama de atividades em nível de sistema. A partir dessa visão a equipe poderá discutir como será concebido o produto. Já a Exploração 360º pode ser utilizada para a verificação e o estudo da viabilidade no desenvolvimento, compra ou reuso dos componentes. A metodologia *sharing* pode ser utilizada para a realização da análise. É recomendado que a equipe de desenvolvimento e os envolvidos com o projeto façam a análise juntos, objetivando buscar a melhor decisão.

No PCP6 , a prática Construir por *Feature* remete ao desenvolvimento das funcionalidades dos projetos passando pelo processo de construção e inspeção. Os componentes do produto devem atender a definição de pronto no projeto. A definição de pronto pode conter itens desde a atualização da documentação do projeto até testes da *feature*. O desenvolvimento da *feature* deve ser de acordo com o projetado.

A Programação por Pares é uma técnica utilizada para a construção do produto na qual duas pessoas realizam o desenvolvimento juntas. Nessa prática há dois perfis: piloto, que possui o controle da atividade de programação; e o co-piloto, que possui o papel de auxiliar e propor outras formas para atingir a solução. Os pares devem realizar rodízios para que haja a diversidade de concepções entre os membros envolvidos, gerando um melhor entendimento e projeto da solução.

Os Testes são realizados durante o desenvolvimento do componente de produto. Há uma interface dessa atividade com o processo de Verificação – VER, que é definida de acordo com a organização. A prática Refatoração é utilizada para otimizar o código interno do componente do produto sem a alteração do seu comportamento externo. Esse atividade pode trazer alterações na estrutura e na arquitetura do projeto devendo ser documentado.

Na Integração Contínua deve-se estabelecer uma política de integração dos componentes de produto por meio de ferramentas automotizadas que realizam o *merger* dos códigos fontes juntamente com a execução de todos os testes (unitário, integração e aceitação). Na prática Esqueleto Ambulante, o desenvolvimento dá-se por meio da criação de uma pequena aplicação de forma simples que pode-se integrar aos demais componentes. Essa prática garante o desenvolvimento da aplicação utilizando a

tecnologia definida e a arquitetura do projeto. A aplicação desenvolvida pode ser um protótipo funcional, que ao final do processo pode ser integrado ao produto.

Na Programação Lado-a-lado dois desenvolvedores trabalham lado a lado na mesma funcionalidade cada um em um terminal, possibilitando que a visualização do objetivo desenvolvido seja para os dois.

Na prática Projetar a Solução, deve ser realizada a construção de cenários de testes para a verificação do componente do produto. A verificação está relacionada à capacidade do componente do produto em atender o que foi projetado. Os cenários de teste são construídos de acordo com o projeto, garantindo o atendimento ao projetado.

No PCP 7, a prática da Documentação Abrangente remete a equipe de desenvolvimento a definir quais documentos serão necessários para o projeto. No Crystal, o manual do usuário é um produto de trabalho e deve estar integrado ao desenvolvimento de cada funcionalidade. Como alternativa de implantação da prática, a documentação citada pode fazer parte da definição de pronto do projeto.

No processo Construir por *Feature* no FDD, é necessária a documentação do *design* e do usuário, sendo que para cada nova feature toda a documentação deve ser atualizada. Por fim, a prática Projetar Solução do TDD rege que deve ser necessário atualizar as documentações do cenário de teste e das classes do projeto.

No PCP8, a Documentação Abrangente para esse resultado esperado tem o enfoque de manter somente a documentação necessária para o andamento do projeto. No Detalhar por *Feature* é proposto atualizar a documentação do *design* para cada *feature* do projeto. Já no Construir por *Feature* é necessário atualizar as documentações do projeto durante a construção do produto de acordo com a definição de pronto estabelecida. Para a prática de Projetar Solução, a documentação do projeto é desenvolvida e mantida a cada projeto de teste, por exemplo. Essa documentação de projeto é importante para dar suporte às mudanças que o software venha sofres ao longo do tempo.

4.3. Estruturação do *Framework*

O processo de estruturação de *framework* objetivou a padronização e a formatação do *framework*, visando a organização do documento. A estruturação do *framework* foi realizada tendo como base a prática ágil: Nome da prática e seu propósito.

O plano de implementação da prática, contemplando a forma de implementação, o grau de apoio e a ferramenta de apoio foram adicionados à estrutura do *framework* para demonstrar em forma de fluxo de processo como a prática pode ser aplicada e justificada pela necessidade de apresentar detalhadamente o "como" fazer o uso da prática. O grau de apoio foi inserido para indicar para as organizações qual o grau de apoio da prática com o resultado esperado. As ferramentas compõem o plano de implementação como sugestão de ferramentas que auxiliam a execução da prática ágil.

Visando a melhor eficiência na aplicação das práticas ágeis, os itens "Relacionamentos com outras práticas ágeis" e "Conflito com outras práticas ágeis" foram adicionados para indicar, ao usuário do *framework*, qual ou quais relacionamentos e conflitos essa prática pode ter.

As sugestões de perfil da organização e do projeto para a aplicação da prática ágil foram adicionadas para apresentar ao usuário do *framework* indicativos de cenários na organização e nos projetos para aplicação da prática ágil. Por fim, o item "Notas" apresenta informações adicionais para a aplicação da prática ágil.

O quadro 7 apresenta o formato de apresentação do *framework*, indicando os itens utilizados para descrever o processo de implementação da prática para o resultado esperado. Além disso, na referido quadro há o indicativo de que informações são inseridas em cada item.

Quadro 7. Estrutura do Framework

<table>
<tr><td colspan="2">1. Nome da Prática</td></tr>
<tr><td colspan="2">1.1 Origem de Prática: metodologia em que a prática está inserida</td></tr>
<tr><td colspan="2">1.2 Propósito: objetivo da prática</td></tr>
<tr><td colspan="2">1.3 Plano para a implementação da prática</td></tr>
<tr><td colspan="2">1.3.1 Forma de implementação: descrição da forma de implementação, exemplo e diagrama das atividades de implementação.</td></tr>
<tr><td>1.3.2 Grau de apoio: Podendo ser total ou parcial, devidamente justificada.</td><td>1.3.3 Ferramenta de apoio: Indicativo de quais ferramentas podem ser utilizadas para apoiar a implementação da prática.</td></tr>
<tr><td colspan="2">1.4 Relacionamento com outras práticas ágeis: Indicativo de quais práticas há relação para a implementação</td></tr>
<tr><td colspan="2">1.5 Conflito com outras práticas ágeis: Indicativo de quais práticas apresentam conflitos ou “choque” para a implementação</td></tr>
<tr><td colspan="2">1.6 Sugestão de perfis de organização e projeto para aplicação da prática</td></tr>
<tr><td colspan="2">Características organizacionais</td></tr>
<tr><td>Ambiente:</td><td>Equipe:</td></tr>
<tr><td>1) Provê a comunicação entre os membros da equipe de desenvolvimento
2) Facilita a cooperação da equipe de desenvolvimento</td><td>1) Nível de conhecimento em engenharia de software
2) Agregam mais de um perfil no escopo do processo de software</td></tr>
<tr><td colspan="2">Características em projeto</td></tr>
<tr><td>1) Grau de acesso aos usuários
2) Localização geográfica
3) Tamanho da equipe</td><td>4) Experiência da equipe com desenvolvimento em engenharia de software
5) Há necessidade de entregas de produtos intermediários</td></tr>
<tr><td colspan="2">1.7 Notas: Observações pertinentes a implementação da prática</td></tr>
</table>

A seção 4.4 apresenta o *framework* desenvolvido na totalidade, indicando para cada resultado esperado as práticas ágeis que fornecem apoio para à implementação do resultado.

Ao fim de cada resultado esperado há o item denominado “Relacionamento entre as práticas”, cujo o objetivo é reunir práticas ágeis relacionadas ao resultado esperado e apresentar uma forma de implementação contendo um conjunto de práticas ágeis para apoiar o resultado esperado.

4.4. O *Framework*

O desenvolvimento do *framework* tem como objetivo fornecer alternativas para apoiar a implementação do processo de projeto e construção do produto e da área de processo de solução técnica. O *framework* oferece um conjunto de práticas ágeis e principalmente, uma forma de "como" aplicar as prática ágeis indicando o contexto para essa aplicação. A figura 4 apresenta uma representação da concepção do framework.

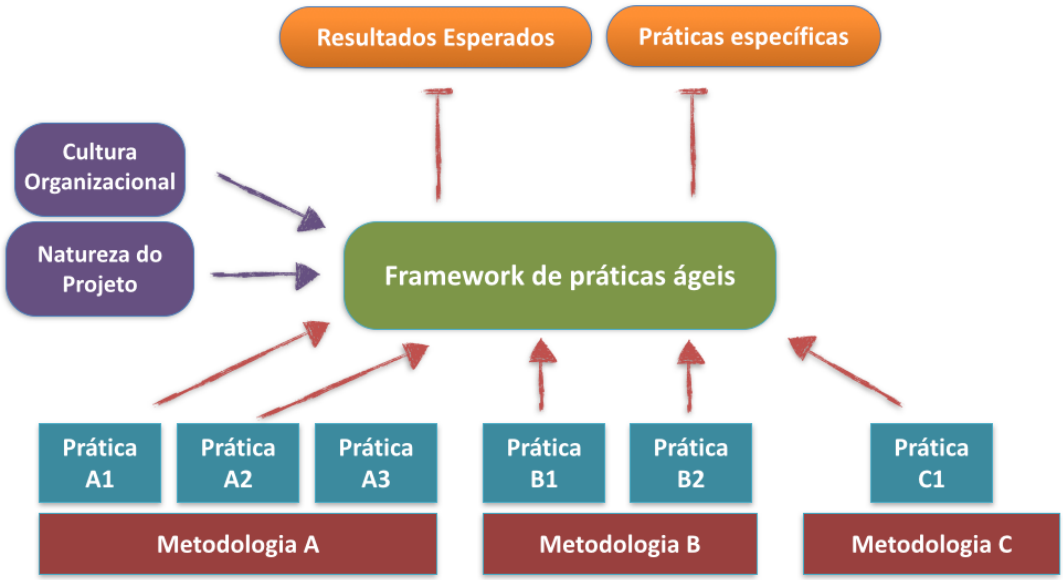

Figura 4. Representação do Framework

O uso do *framework* pode otimizar o processo de implementação, seja relacionado ao tempo ou esforço, pois a grande maioria das práticas descritas podem ser implementadas de maneira independente, com isso há a flexibilidade na sua utilização. As práticas ágeis podem ser escolhidas e associadas a outras práticas para satisfazer os resultados esperados e alguns aspectos culturais da organização.

Nesta seção será apresentado para cada resultado esperado as práticas ágeis que apoiam a implementação. Além disso, será apresentado os objetivos para cada resultado esperado. Os resultados esperados serão divididos em oito subseções, uma para cada resultado esperado.

4.4.1 PCP1 - Alternativas de solução e critérios de seleção são desenvolvidos para atender aos requisitos definidos de produto e componentes de produto

Objetivo: identificar possíveis soluções para a construção do produto com base em um processo de seleção de alternativas possui, basicamente, as seguintes atividades: definição dos objetivos de seleção; estabelecimento dos critérios de seleção; desenvolvimento das soluções a serem avaliadas; avaliações das soluções com base nos critérios pré-estabelecidos e seleção da solução mais adequada.

4.4.1.1 Exploração 360º

a) Origem da prática

Crystal

b) Propósito

Reunião de planejamento das tecnologias, metodologias e processos a serem utilizados no projeto.

c) Plano para a implementação da prática

c.1) Forma de implementação

A implementação da prática dá-se inicialmente pela realização de um *Brainstorm* para levantar alternativas de solução para o projeto. Nessa sessão de *brainstorm*, lideres de projetos e analistas de sistemas devem trazer inúmeras propostas para solução do projeto, aspectos relacionados a padrões de projeto e arquitetura devem fazer parte das discussões. Experiências em projetos correlatos podem apresentar indícios para o reuso de uma solução. Além disso, opções de tecnologias, linguagem de programação e formas de persistências devem ser listadas.

Para a avaliação da solução, aspectos positivos e negativos devem ser confrontados a fim de direcionar para a classificação da solução. Requisitos como performance, usabilidade, reuso podem ser utilizados como critérios de avaliação das soluções.

Na seleção das possíveis soluções, o uso da metodologia *shaping*, técnica utilizada para identificar pontos fortes e fracos da organização, pode ser expandido para a análise de alternativas de soluções para o projeto. As soluções que atendam aos critérios estabelecidos pela equipe do projeto devem ser selecionadas. A Figura 5 apresenta o processo de implementação da prática exploração 360º.

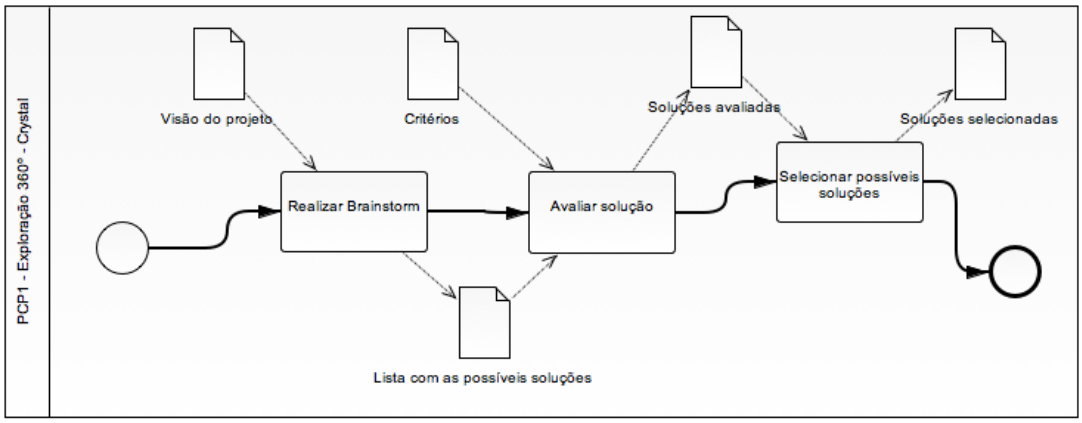

Figura 5. Processo para execução da prática Exploração 360º.

c.2) Grau de apoio

Total, pois por meio da atividade de realização de *brainstorm* as alternativas de soluções são listadas e posteriormente selecionadas pelos critérios estabelecidos.

c.3) Ferramenta de apoio

Uso de quadro branco, *post-its* e *flip-chart* são ferramentas que podem auxiliar na execução do *brainstorm*. Um exemplo do uso seria escrever em cada *post-it* o identificador da solução e colar no *flip-chart*, de modo que todos que estão participando da sessão de *brainstorm* possam visualizar de modo simples as soluções apresentadas como propostas. A mesma técnica pode ser aplicada na avaliação da solução, *post-its* com aspectos positivos e negativos podem ser agrupados em um quadro branco ou em um *flip-chart*.

Para equipes que estão trabalhando em locais diferentes o uso de ferramentas online para video-conferência é fundamental para que ocorra a interação da equipe.

d) Relacionamento com outras práticas ágeis

A implementação dessa prática pode auxiliar a Rearquitetura Incremental, pois fornece uma analise preliminar para a escolha de qual solução pode ser o *startup* para a evolução e expansão da solução.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto ou Médio
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Baixo
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto ou Médio
5) Há necessidade de entregas de produtos intermediários	Não

g) Notas

A visão do projeto pode ser apresentada nas formas definidas pela organização, podendo ser: *Product Data Sheet*, *Elevator Statement*, Documento de Visão, Lista de requisitos.

Os critérios para a avaliação das soluções deve ter como base as definições do projeto e da equipe.

4.4.1.2 Rearquitetura Incremental

a) Origem da prática

Crystal

b) Propósito

Descobrir a melhor solução à medida que o desenvolvimento está acontecendo, por meio de remodelagem do projeto.

c) Plano para a implementação da prática

c.1) Forma de implementação

Uma solução inicial pode ser proposta pelos líderes técnicos do projeto ou baseada na experiência de membros da equipe, a partir desse momento são realizadas curtas interações de desenvolvimento para avaliar a solução. Caso a equipe avalie como melhor solução, a arquitetura será reprojetada. Ao contrário, a equipe pode selecionar uma outra solução apresentada.

Para apoiar a implementação do resultado esperado na sua totalidade é necessário definir critérios para a aplicação da solução inicial. Como exemplos pode-se citar:

- Há suporte tecnológico para a solução inicial na organização ?
- A equipe atual possui habilidade para desenvolvimento da solução inicial ?
- Há projetos semelhantes no mercado que faça o uso dessa solução ?
- A solução inicial já foi utilizada em outro contexto de projeto ?

A utilização de requisitos cujo a complexidade seja de grau não simples pode ajudar a projetar uma arquitetura que atende o escopo do projeto. A Figura 6 apresenta o processo de implementação da prática de rearquitetura incremental.

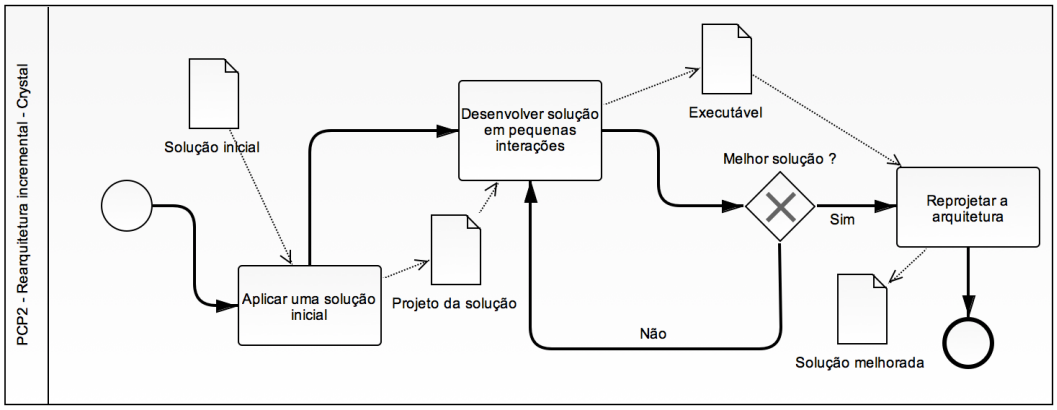

Figura 6. Processo para execução da prática Rearquitetura incremental.

c.2) Grau de apoio

Parcial, pois para inicio da prática uma solução inicial já precisa está definida. Nesse caso a avaliação de alternativas de solução não é contemplada.

c.3) Ferramenta de apoio

Ambiente integrado de desenvolvimento (IDE) e Ferramentas de modelagem.

d) Relacionamento com outras práticas ágeis:

Exploração 360º pode ser utilizada para selecionar a solução inicial identificada.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Médio
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

O executável pode ser definido como uma solução em operação.

4.4.1.3 Projeto Simples

a) Origem da prática

Extreme Programming (XP)

b) Propósito

Identificar a solução mais simples e começar o desenvolvimento.

c) Plano para a implementação da prática

c.1) Forma de implementação

Nessa prática, inicialmente deve-se listar as possíveis soluções para o projeto. Na atividade de avaliação é necessário identificar qual solução possui o menor investimento inicial. O contexto de menor investimento está no sentido da simplicidade de implementação e custo.

Cada item da solução deve possuir um papel definido e ser devidamente justificado. A Avaliação do custo versus benefício da solução deve ser levado em consideração. Alguns critérios podem ser utilizados para a avaliação da solução:

- Custo (menor custo melhor);
- Experiência da equipe de desenvolvimento com a solução (mais experiência com a solução melhor);
- Flexibilidade de solução (na solução há possibilidade de extensão, se houver melhor);
- Curva de aprendizagem para novos membros (menor curva melhor).

A Figura 7 apresenta o processo de implementação da prática projeto simples.

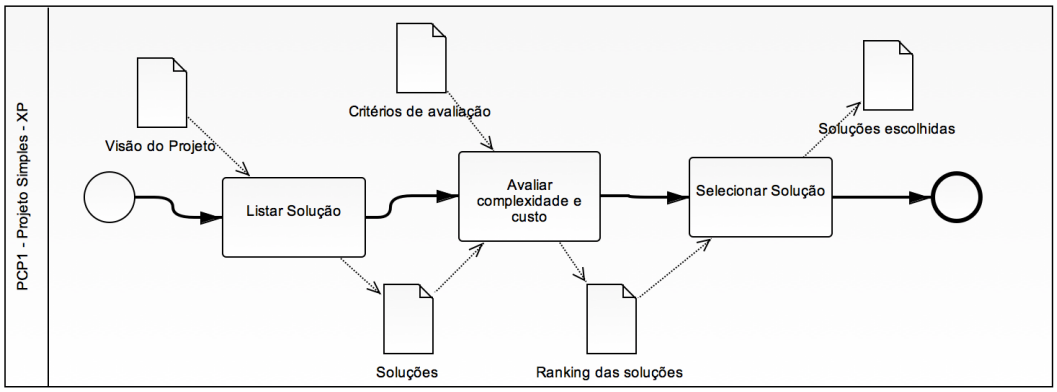

Figura 7. Processo para execução da prática Projeto Simples

c.2) Grau de apoio

Total, pois as soluções são escolhidas por meio de critérios de avaliação.

c.3) Ferramentas de apoio

Quadro branco pode ser utilizado para dar maior visibilidade do processo de listagem, avaliação e seleção das soluções para todos os membros da equipe.

d) Relacionamento com outras práticas ágeis

Esse prática pode ser associada à prática rearquitetura incremental, pois após a seleção da solução, a prática rearquitetura incremental irá auxiliar no refinamento da solução.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Médio
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena ou Média
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto ou Médio
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

O ranking das soluções pode ser construído baseado em média ponderada, sendo para cada critério da avaliação um peso é indicado e para cada solução uma nota é indicada.

4.4.1.4 Relacionamento entre as práticas

a) Propósito

Relacionar as práticas ágeis 1,2 e 3 descritas na seção do 4.4.1 (PCP1) visando o apoio na totalidade a implementação do resultado esperado.

b) Plano para a implementação da prática

b.1) Forma de implementação

A implementação híbrida consiste em uma proposta de apoio à implementação do resultado esperado. No PCP1 foram utilizado as atividades da Exploração 360º (Brainstorm, seleção das possíveis soluções), da Projeto Simples (Avaliação da complexidade e custo da implementação da solução, seleção da solução) e de Rearquitetura Incremental (Desenvolvimento em pequenas interações e Reprojetar a arquitetura). A Figura 8 apresenta o processo de implementação do relacionamento entre as práticas do PCP1.

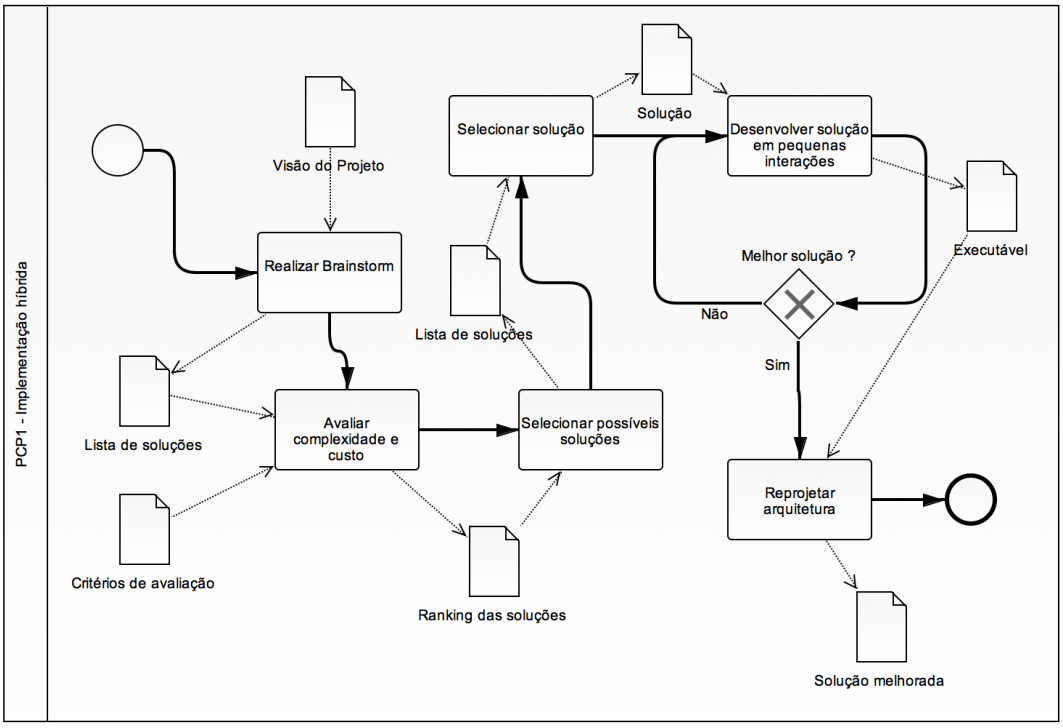

Figura 8. Implementação do relacionamento do PCP1

4.4.2 PCP2 - Soluções são selecionadas para o produto ou componentes do produto, com base em cenários definidos e em critérios identificados

Objetivo: realizar a seleção das soluções para o produto ou componente do produto, tendo como base cenários e critérios para que se possa selecionar a solução mais adequada.

4.4.2.1 Esqueleto Ambulante

a) Origem da prática

Crystal

b) Propósito:

Desenvolvimento de um pequeno modelo para cada solução, esse modelo pode ser um protótipo funcional, a ser analisado para assegurar que haja a possibilidade da aplicação da solução.

c) Plano para a implementação da prática

c.1) Forma de implementação

De posse da solução, oriunda da implementação do PCP1, a equipe de desenvolvimento deve projetar e implementar um protótipo funcional em uma interação pequena. O tempo da interação pode variar de acordo com o desempenho da equipe e a complexidade do projeto. Após esse desenvolvimento, é necessária a avaliação desse protótipo e por conseguinte da solução, mesmo que parcialmente.

Critérios de avaliação como velocidade de desenvolvimento, curva de aprendizagem da equipe e reuso podem ser utilizados para a avaliação. Caso a solução não atenda às necessidades, a equipe retoma ao passo de avaliar outra solução por meio de um protótipo funcional. A Figura 9 apresenta o processo de implementação da prática esqueleto ambulante.

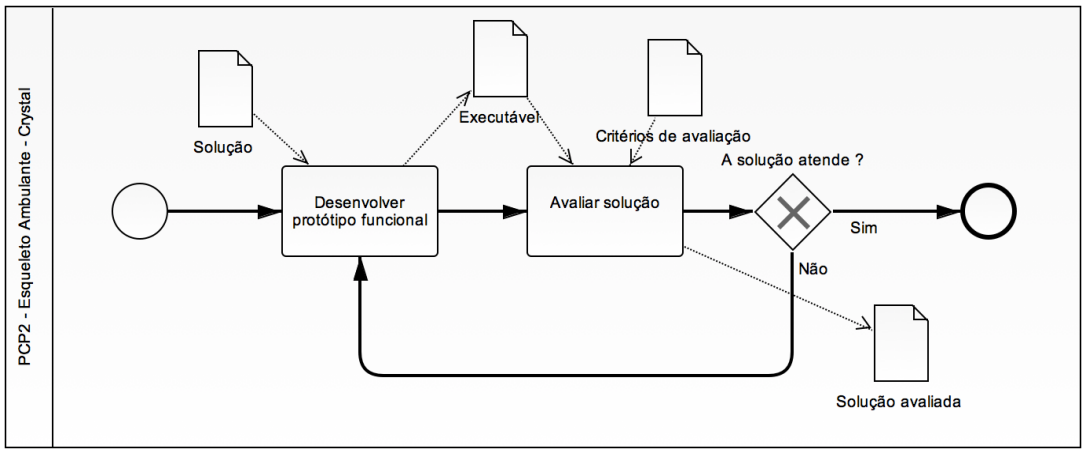

Figura 9. Processo para execução da prática Esqueleto ambulante

c.2) Grau de apoio

Parcial, pois apenas os critérios de avaliação são utilizados para determinar a escolha da solução.

c.3) Ferramenta de apoio

Ferramentas de controle de versão podem ser utilizadas para versionar os protótipos, a criação de *branches* para cada solução pode facilitar no processo de integração contínua para o andamento do desenvolvimento da solução.

d) Relacionamento com outras práticas ágeis

O relacionamento com as práticas Exploração 360º, Rearquitetura incremental e Projeto Simples no contexto da aplicação para implementação do PCP1, pois ambas fornecem a solução ou um conjunto de soluções devidamente avaliadas.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim

2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Alto
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

Entende-se como executável um pacote de código fonte que seja funcional, ou seja, que atenda mesmo que parcialmente (segurança, usabilidade, design) um dos requisitos do projeto., devendo ser passível de utilização pelo cliente final.

Os critérios de avaliação devem ser definido pela organização e equipe do projeto.

4.4.2.2 Detalhar por Feature

a) Origem da prática

Feature Driven Development (FDD)

b) Propósito

Identificar e detalhar qual a melhor solução para o projeto da *feature*. O desenvolvimento fazendo o uso da prática do FDD pressupõe a utilização dos requisitos no formato de *features* ou em pequenas tarefas.

c) Plano para a implementação da prática

c.1) Forma de implementação

De posse do modelo abrangente, que pode conter os diagramas de classes, diagrama de atividades e modelo entidade relacionamento da funcionalidade a ser projetada, a

equipe que for alocada como responsável pelo desenvolvimento dessa funcionalidade irá projetar gerando o diagrama de sequência e atualizar os detalhes da classes. A fase de projeto da funcionalidade pode gerar uma atualização na arquitetura da solução.

Na atividade de avaliar o projeto da funcionalidade a equipe realiza a inspeção na lista de tarefas geradas no projeto e, além disso, avalia o modelo desenvolvido em relação ao contexto da solução.

Apesar da prática detalhar por *feature* possuir outras atividades em sua definição, para o contexto de implementação do PCP2, as atividades destacadas na Figura 10 são as que geram as evidências esperadas para o alcance do resultado esperado.

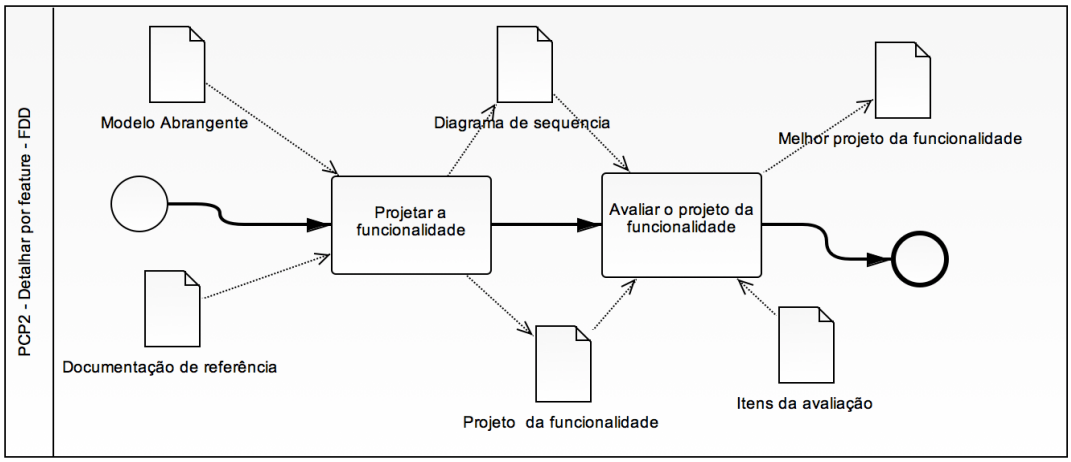

Figura 10. Processo para execução da prática detalhar por feature

c.2) Grau de apoio

Parcial, pois somente os critérios de avaliação são utilizados para a escolha da solução.

c.3) Ferramenta de apoio

Ferramentas de modelagem UML fornecem o apoio para construção do diagrama de sequência gerado no processo de Projetar a funcionalidade.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Esqueleto ambulante, pois o modelo utilizado para o desenvolvimento de uma solução é por meio de protótipo funcional, já na prática detalhar por *feature*, o enfoque está nas *features* do projeto.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Não
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Alto
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

O modelo abrangente deve conter diagramas e fluxograma que apresentem o projeto de maneira a demonstrar quais as principais atividades e suas relações dentro do projeto.

4.4.2.3) Relacionamento entre as práticas

a) Propósito

A implementação híbrida tem o propósito de relacionar as práticas ágeis 1 e 2 descritas na seção 4.4.2 (PCP2) visando o apoio a implementação do resultado esperado.

b) Plano para a implementação da prática

b.1) Forma de implementação

A relação das duas práticas está no contexto em que primeiramente é realizado o projeto da funcionalidade e sua avaliação e, posteriormente por meio da criação de um protótipo funcional, pode-se ter maior evidência se a solução atender as expectativas.

O resultado da atividade Projetar a funcionalidade fornece insumo da documentação da funcionalidade para desenvolver o protótipo funcional. A Figura 11 apresenta o processo de implementação do relacionamento das práticas do PCP2.

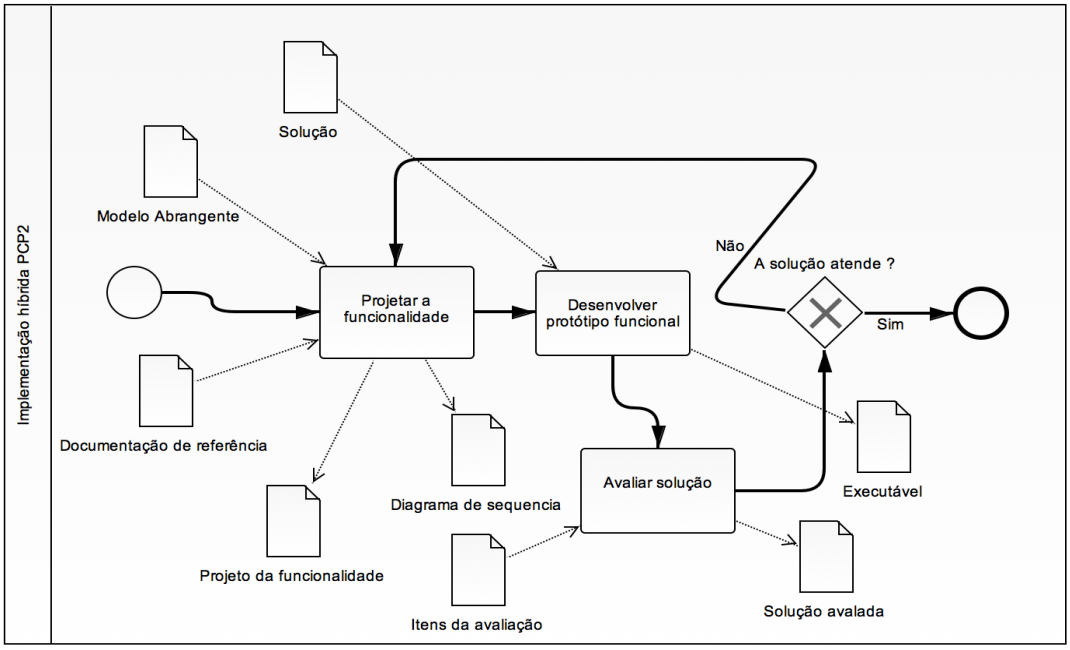

Figura 11. Implementação do relacionamento do PCP2.

4.4.3 PCP3 - O produto e/ou componente do produto é projetado e documentado

Objetivo: realizar e documentar o design do produto, essa documentação gerada servirá como base para o processo de desenvolvimento do produto.

4.4.3.1 Padrões de codificação

a) Origem da prática:

Extreme Programming (XP)

b) Propósito

A prática tem como propósito a definição de um documento de arquitetura de projeto contendo os padrões a serem utilizados.

c) Plano para a implementação da prática

c.1) Forma de implementação

Para a implementação da prática, a equipe dependo da sua experiência, já pode utilizar padrões de codificação. A consulta à comunidade especialista ou a materiais didáticos sobre padronização de código fonte deve ser um documento de entrada para a identificação de padrões a serem utilizados. A padronização da codificação está relacionada ao estilo de programação da equipe, que pode ser de diversos tipos: comentários padronizados, estrutura do projeto, organização dos pacotes, nome de classe, nome de variável, nome de método, estilo de indentação do código, utilização de padrões de projetos entre outros.

De posse da identificação dos padrões de codificação utilizados, a próxima tarefa é realizar a atualização do documento de arquitetura. Nesse documento deve existir uma seção reservada para os padrões de codificação do projeto. Esse processo de atualização deve ser uma atividade contínua, à medida que o projeto encaminha-se para o seu término. A Figura 12 representa o processo de implementação da prática.

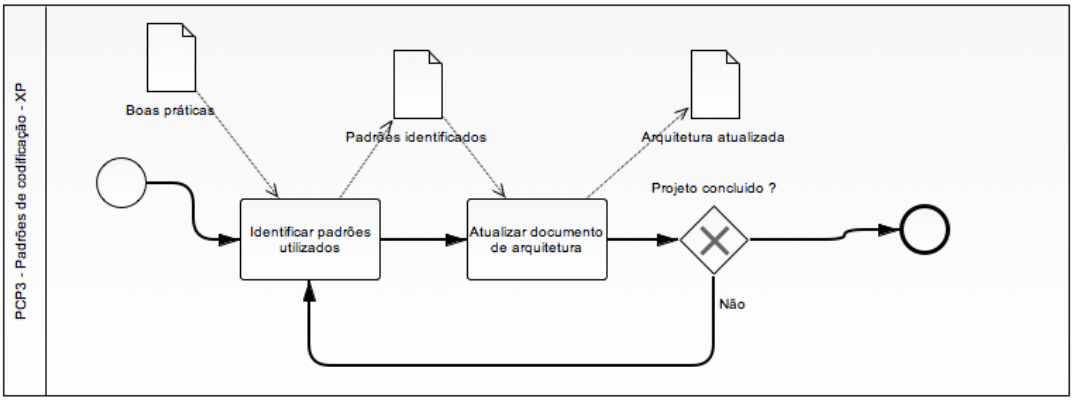

Figura 12. Processo para execução da prática Padrões de codificação.

c.2) Grau de apoio

Parcial, em virtude de não atender a etapa de documentação do projeto do componente.

c.3) Ferramenta de apoio

Para registro dos padrões de codificação, as ferramentas estilo *wiki* são recomendadas, pois fornecem a possibilidade de consulta e atualização rápida.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto ou médio
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Baixo
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto ou médio
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

As boas práticas podem surgir dos projetos anteriormente desenvolvidos pela organização, sendo a equipe de desenvolvimento a responsável por institucionalizar a sua utilização dentro do projeto.

4.4.3.2 Projeto simples

a) Origem da prática

Extreme Programming (XP)

b) Propósito

Conceber o projeto de maneira simples, ou seja, o design do projeto deve acompanhar a simplicidade. A complexidade do projeto sem necessidade deve ser retirada.

c) Plano para a implementação da prática:

c.1) Forma de implementação

A implementação da prática está pautada na simplicidade das ações dentro do projeto. Dada a lista de soluções identificadas no PCP2, é necessário identificar a solução mais simples. Depois da identificação, deve-se projetar a aplicação e realizar a atualização do projeto. A documentação do projeto pode estar associada a diagramas, modelos, protótipos, documentação do design. A Figura 13 apresenta o processo para a execução da prática projeto simples.

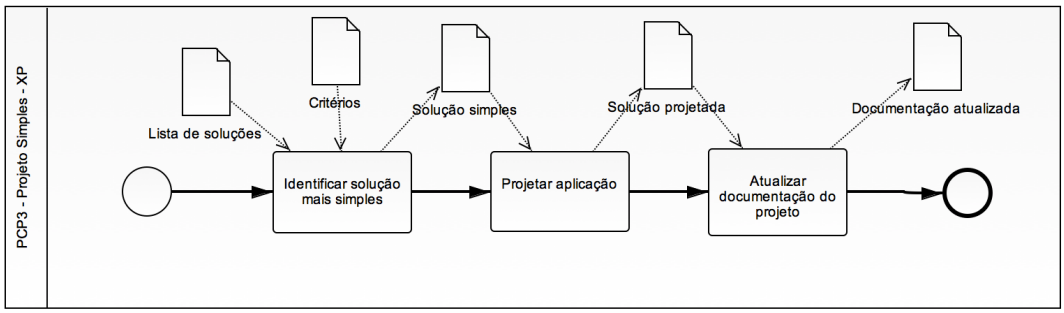

Figura 13. Processo para execução da prática Projeto simples.

c.2) Grau de apoio

Total, por conta do atendimento a etapa de projeto e documentação do componente do produto ou projeto. A atividade projetar aplicação atende ao projeto da solução (projetar a solução) e a atividade atualizar documentação do projeto atende a documentação desse projeto do produto.

c.3) Ferramenta de apoio

- Ferramenta de modelagem para realizar a atividade de projeto da aplicação.
- Editores de texto para atualizar as documentações, quando necessário.

- Quadro branco para projetar a aplicação, possibilitando a participação e visualização por todos os membros da equipe.

d) Relacionamento com outras práticas ágeis

A prática projeto simples pode ser associada à prática padrões de codificação no momento da atualização da documentação do projeto.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Médio
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto ou médio
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

Solução simples significa a solução que irá despender o menor esforço da equipe para o seu desenvolvimento.

4.4.3.3 Rearquitetura incremental

a) Origem da prática

Crystal

b) Propósito

À medida que o *design* for sofrendo alterações, o projeto arquitetural deverá acompanhar as modificações e ser devidamente documentado.

c) Plano para a implementação da prática

c.1) Forma de implementação

No decorrer do desenvolvimento, mudanças podem surgir na arquitetura do projeto. Caso essas mudanças ocorram, deve-se realizar a atualização do documento de arquitetura do projeto. A atualização da arquitetura do projeto pode estar relacionada à criação ou utilização de um novo componente do produto, algum padrão de projeto aplicado no âmbito da arquitetura do projeto. A Figura 14 apresenta o processo de implementação da prática rearquitetura incremental.

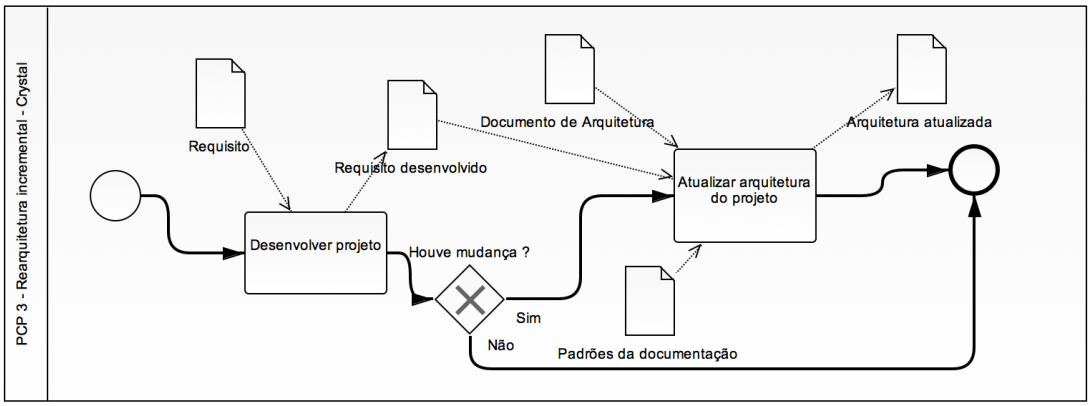

Figura 14. Processo para execução da prática Rearquitetura incremental

c.2) Grau de apoio

Total, por conta que nas práticas desenvolver projeto e na atividade de atualizar arquitetura do projeto as etapas de projetar e documentar a solução são contempladas.

c.3 Ferramenta de apoio

Ferramentas de controle de versão de arquivos pode ser utilizada para definir marcos de atualização do documento de arquitetura do projeto.

d) Relacionamento com outras práticas ágeis

A prática padrão de codificação pode ser utilizada na atividade de atualização da arquitetura do projeto, pois nesse momento novos padrões de codificação podem ser identificados.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto e médio
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Baixo
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto ou médio
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

A atividade de desenvolvimento do projeto é realizada de maneira contínua e com isso, para cada novo requisito desenvolvido, a atividade de atualização da arquitetura é iniciada.

O documento de arquitetura deve ser definido pela organização juntamente com a equipe de desenvolvimento do projeto.

Modelos já utilizados pela organização compõem os padrões de documentação.

4.4.3.4 Detalhar por *feature*

a) Origem da prática

Feature Driven Development (FDD)

b) Propósito

Documentar e projetar a feature.

c) Plano para a implementação da prática

c.1) Forma de implementação

De posse do modelo abrangente, que possui a documentação do modelo de alto nível, diagrama de classes de negócio, para cada *feature*, na tarefa projetar *feature*, todo o artefato gerado nessa atividade é subsidio para a atividade documentar *feature*.

A atividade projetar *feature* deve ser conduzida pelo desenvolvedor líder, que analisa o modelo abrangente e começa a traçar comportamentos inerentes à construção da *feature*.

O resultado dessa atividade é a geração do documento do projeto da *feature* que pode possuir: diagrama de classe, diagrama de sequência, modelo de entidade e relacionamento, diagramas de atividades. A Figura 15 apresenta o processo de implementação da prática detalhar por *feature*.

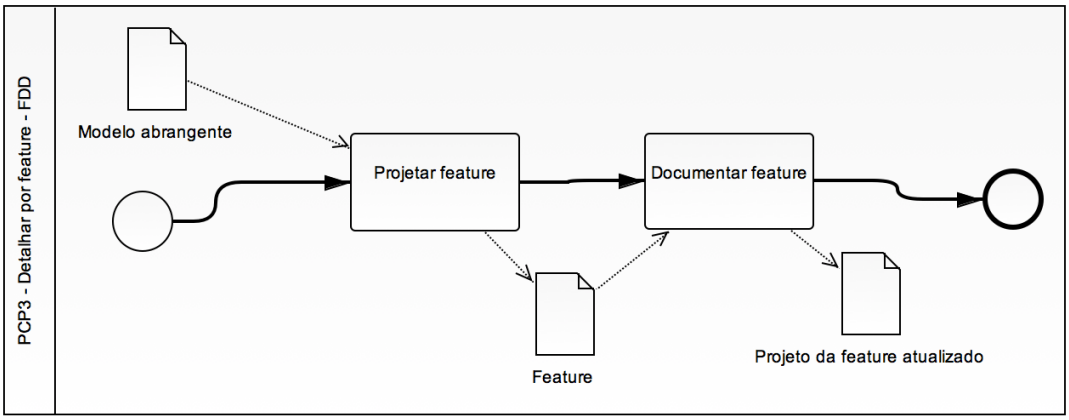

Figura 15. Processo de execução da prática Detalher por Feature.

c.2) Grau de apoio

Total, pois nas atividades de projetar feature o componente do produto é projetado e na atividade de documentar feature a sua documentação é atualizada.

c.3) Ferramenta de apoio

Ferramentas de modelagem UML e BPM podem auxiliar no processo de desenvolvimento da documentação e projeto da *feature*.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Médio
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

A prática é recomendada principalmente para organizações que fazem uso do FDD como metodologia ou utilizam algumas de suas práticas.

4.4.3.5 Relacionamento entre as práticas

a) Propósito

A implementação híbrida tem o propósito de relacionar as práticas ágeis 1 e 2 descritas na seção 4.4.3 (PCP3) visando o apoio a implementação do resultado esperado.

b) Plano para a implementação da prática

b.1) Forma de implementação

A prática Projeto Simples está relacionada à atividade do projeto da solução, já a prática padrões de codificação apoia a atividade de documentação da arquitetura do projeto. A junção das duas práticas fornece maior apoio a implementação do PCP3. A Figura 16 apresenta o processo de implementação do relacionamento entre as prática do PCP3.

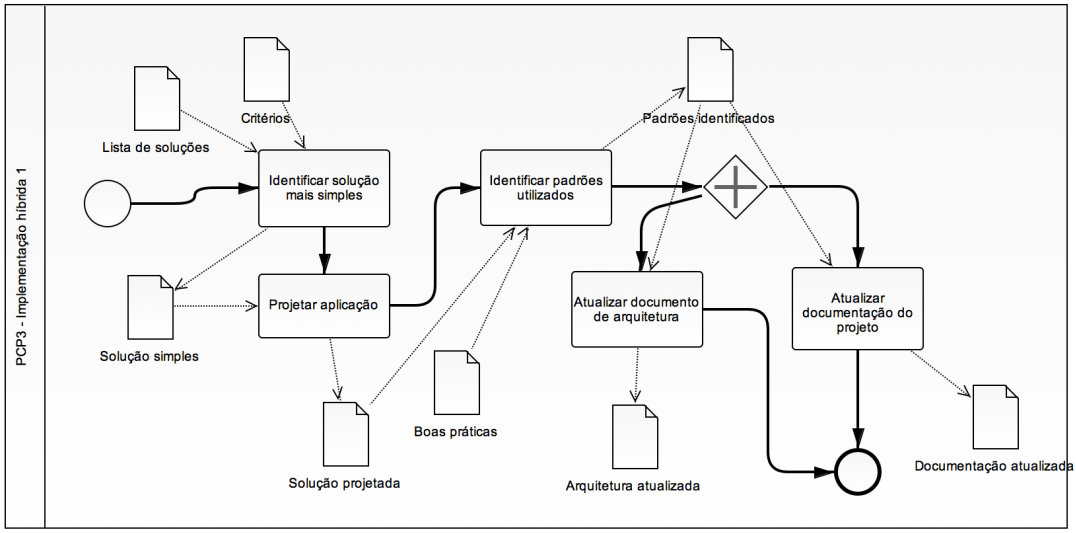

Figura 16. Implementação do relacionamento do PCP3

4.4.4 PCP4 - As interfaces entre os componentes do produto são projetadas com base em critérios predefinidos

Objetivo: definir critérios de seleção, identificar e projetar as interfaces, meio para o estabelecimento de comunicação entre os componentes do produto ou serviços.

4.4.4.1 Detalhar por Feature

a) Origem da prática

Feature Driven Development (FDD)

b) Propósito

As interfaces e suas dependências devem ser identificadas a cada *feature* detalhada.

c) Plano para a implementação da prática

c.1) Forma de implementação

A implementação da prática detalhar por *feature* no contexto de apoio ao PCP4 dá-se inicialmente pela identificação das interfaces da *feature*. Nesse momento, o modelo abrangente e o projeto das outras *features* são importantes, pois fornecem subsidios para a verificação dentro do projeto se existe algum componente no qual a *feature* a ser projetada precisa se comunicar.

Após a identificação dessas interfaces, faz-se necessário realizar o projeto da *feature*, que tem como característica a construção de diagramas que representam essas interfaces. Um produto gerado nessa etapa é o Diagrama de componente da UML. A Figura 17 apresenta o processo de implementação da prática detalhar por *feature*.

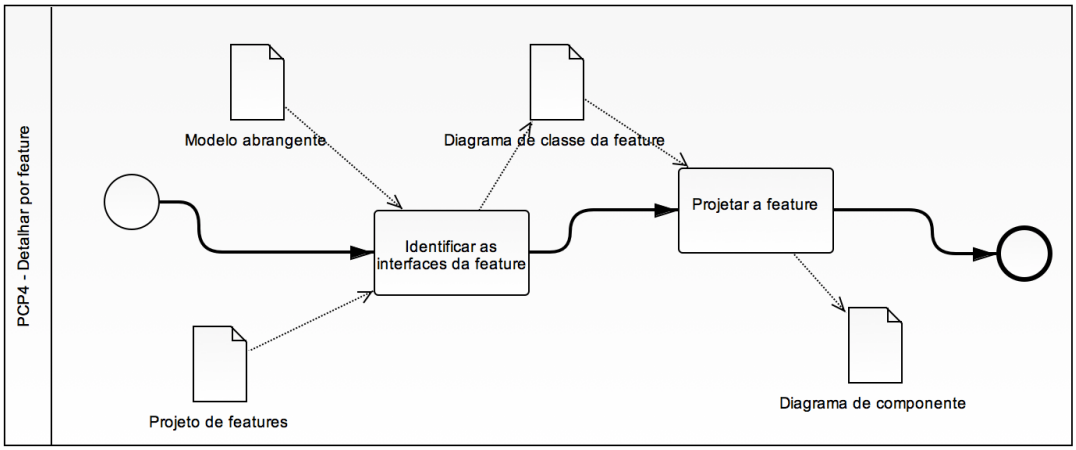

Figura 17. Processo de execução da prática Detalhar por *feature*

c.2) Grau de apoio

Total, pois para cada *feature* do projeto uma análise para identificação das interfaces é realizada.

c.3) Ferramenta de apoio

Ferramentas de modelagem em UML auxiliam no desenvolvimento dos diagramas de componente e de classes.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto ou médio
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Baixo
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto ou médio
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

O diagrama de classe da *feature* deve conter explicitamente os métodos e as interfaces, além do relacionamento entres as classes.

4.4.4.2 Esqueleto Ambulante

a) Origem da prática

Crystal

b) Propósito

Propõe o desenvolvimento de pequenos componentes e sua posterior integração aos demais componentes por meio do design evolutivo. As interfaces são identificadas à medida que os componentes da arquitetura são desenvolvidos.

c) Plano para a implementação da prática

c.1) Forma de implementação

O desenvolvimento dos requisitos é o primeiro passo para a execução da prática, para isso é necessário o uso do documento de arquitetura do projeto. Ele é a base para a evolução das interfaces entre os componentes do projeto, haja vista que é um artefato em constante atualização.

A próxima atividade é a identificação das interfaces dos componentes. Nessa identificação, o uso de um modelo simples ou uma modelagem adotando notação UML são recomendadas, pois será necessário evoluir esse modelo a cada desenvolvimento de novos requisitos. A Figura 18 apresenta o processo de implementação da prática esqueleto ambulante.

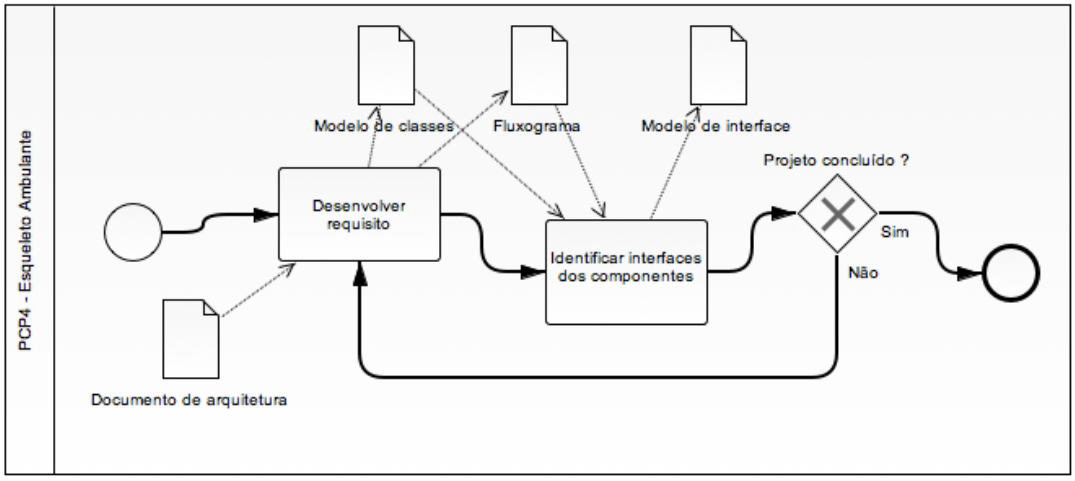

Figura 18. Processo de execução da prática Esqueleto Ambulante

c.2) Grau de apoio

Parcial, pois os critérios para a determinar as interfaces dos componentes não são estabelecidos.

c.3) Ferramenta de apoio

Ferramentas de modelagem UML são recomendadas, sendo os diagramas de classes, diagrama de componente e diagrama de sequência os mais indicados para a identificação das interfaces.

O uso de modelagem em um quadro branco pode ser utilizado desde que ao término da sessão seja persistido as informações, seja por meio de diagramas ou de uma foto.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Há um conflito com a prática Detalhar por feature, pois na implementação dessa prática, além da restrição por se trabalhar com requisitos no formato de feature, tem-se o enfoque na utilização de diagramas UML. Essa restrição limita o escopo de abrangência na utilização de ferramentas e no modo de se especificar requisitos.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto ou médio
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Baixo
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto ou médio
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

Requisitos podem ser especificados em qualquer formato conhecido: Caso de Uso, Estórias de Usuário, *Features*, Tarefas.

4.4.4.3 Relacionamento entre as práticas

Não se aplica.

4.4.5 PCP5 - Uma análise dos componentes do produto é conduzida para decidir sobre sua construção, compra ou reutilização

Objetivo: definir como será concebido o produto, podendo ser por desenvolvimento interno, aquisição ou reutilização.

4.4.5.1 Modelagem de Objeto Conceitual

a) Origem da prática

Feature Driven Development (FDD).

b) Propósito

Por meio da modelagem conceitual são extraídas as informações para subsidiar a decisão da construção, compra ou reuso de componentes.

c) Plano para a implementação da prática

c.1) Forma de implementação

Para iniciar o processo de execução dessa prática, recomenda-se que o documento de visão do projeto e uma lista de soluções do mesmo contexto de negocio ou semelhantes sejam disponíveis para as atividades de avaliação dos requisitos do projeto. Nessa avaliação, a equipe de desenvolvimento deve extrair da documentação disponível e da experiência da equipe qual a melhor alternativa para o desenvolvimento do projeto.

Na atividade de desenvolvimento do modelo conceitual, toda a atividade de avaliação é absorvida e uma proposta de modelo abrangente é identificada. O Modelo abrangente contém diagramas de atividades e diagramas de classes de domínio.

Após o desenvolvimento do modelo abrangente e da lista com as melhores alternativas, é realizada a avaliação do modelo. Nessa avaliação aspectos para determinar se a solução vai ser construída, reusada ou adquirida são tratados. Uma lista de critérios pode ser utilizada para a padronização da avaliação. Critérios como: Há uma ferramenta nesse modelo disponível no mercado ? O custo do desenvolvimento é maior que o custo de aquisição ? Na organização há projetos que possa ser reutilizado para atender as necessidades desse modelo abrangente ?, podem determinar ser a solução deve ser construída, comprada ou reutilizada.

A Figura 19 apresenta o processo de implementação da prática modelagem de objeto conceitual.

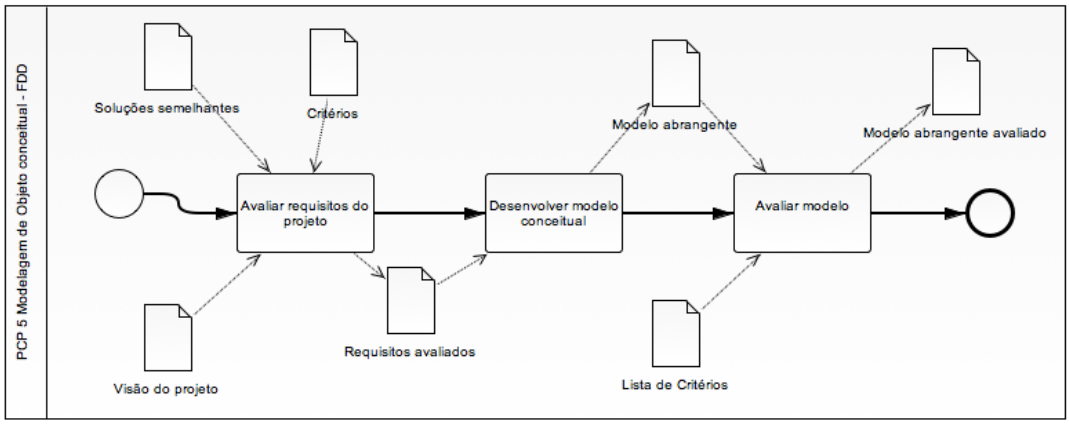

Figura 19. Processo de execução da prática Modelagem de objeto conceitual.

c.2) Grau de apoio

Total, pois na atividade de avaliação dos requisitos do projetos é realizada uma análise sobre como será concebido o projeto. Além disso, o modelo gerado nessa análise é submetido a uma avaliação, para sim identificar como será a execução do projeto.

c.3) Ferramenta de apoio

Ferramentas de modelagem UML são úteis para o desenvolvimento do modelo abrangente.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Médio
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto ou médio
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

Critérios para a avaliação do modelo devem ser definidos de acordo com o projeto e a organização, sendo que a relação custo versus benefício é um importante critérios a ser utilizado.

4.4.5.2 Exploração 360º

a) Origem da prática

Crystal

b) Propósito

Utilizada para a verificação e o estudo da viabilidade no desenvolvimento, compra ou reuso dos componentes.

c) Plano para a implementação da prática

c.1) Forma de implementação

O passo inicial para aplicação da prática é realizar a avaliação inicial dos requisitos e a lista de soluções. Nessa etapa o foco está em relacionar quais soluções realizam o atendimento de quais requisitos. Tendo como exemplo o desenvolvimento de um sistema de automação, cujos requisitos estão relacionados à venda de produto, uma possível solução para a geração da nota fiscal eletrônica seria a utilização de uma API para a comunicação. Como artefato de entrada para essa atividades, os documentos de visão do projeto, lista de requisitos e lista das soluções são necessários.

Após a criação dos relacionamentos entre requisitos e soluções, a equipe de desenvolvimento, juntamente com os principais interessados no projeto, realizam uma sessão de exploração. Essa sessão consiste em uma análise do relacionamento proposto, com o objetivo de identificar como será desenvolvida a solução indicada. Os interessados irão discutir além de aspectos técnicos da solução, os pontos fortes e fracos, e as implicações em termos do orçamento serão relacionados. Ao final dessa atividade, deveram ser decididas quais soluções serão desenvolvidas, adquiridas ou reutilizadas, devendo ser detalhada em um quadro final.

A Figura 20 apresenta o processo de implementação da prática exploração 360º.

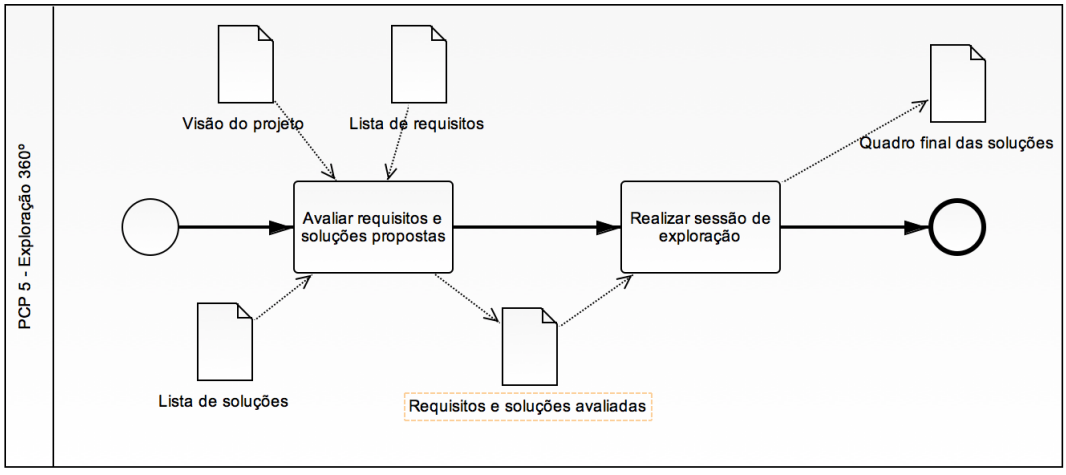

Figura 20. Processo de execução da prática exploração 360º no contexto do PCP5

c.2) Grau de apoio

Total, pois nas sessões de exploração é realizado uma avaliação abrangente sobre as possíveis soluções de como poderá ser concebido o projeto.

c.3) Ferramenta de apoio

O uso de planilhas eletrônicas para a geração do quadro final pode ser utilizada. Para a sessão de exploração, o uso de *flipchart* e *post-it* torna a atividade de identificação de pontos positivos e negativos visualmente mais detalhadas.

Para equipes que trabalham remotamente, o uso de ferramentas de video conferência é fundamental para a execução da sessão de exploração.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto ou médio
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Médio
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto ou médio
5) Há necessidade de entregas de produtos intermediários	Não

g) Notas

As sessões de exploração devem ter um membro para realizar a condução das atividades e das discussões, devendo-se canalizar os esforços para a obtenção dos resultados.

4.4.5.3 Relacionamento entre as práticas

Não se aplica.

4.4.6 PCP6 - Os componentes do produto são implementados e verificados de acordo com o que foi projetado

Objetivo: implementação do produto de acordo com o design estabelecido e sua documentação deve ser desenvolvida. A inspeção na implementação verificará se o componente do produto foi implementado conforme o projetado.

4.4.6.1 Construir por feature

a) Origem da prática

Feature Driven Development (FDD)

b) Propósito

Remete ao desenvolvimento das funcionalidades dos projetos passando pelo processo de construção e inspeção.

c) Plano para a implementação da prática

c.1) Forma de implementação

De posse da *feature* detalhada, documento que contém todas as especificação da *feature* (modelo de classe e diagrama de atividades), a equipe de desenvolvimento realiza a codificação da funcionalidade. A condução da implementação deve acontecer conforme o especificado no detalhamento da *feature*.

Após o desenvolvimento, outro membro ou equipe realiza a revisão do código da *feature*, nessa atividade uma lista de critérios para a revisão é utilizada para manter a padronização do código fonte. Exemplos de critérios: o nome dos métodos da classe estão utilizando o padrão camelCase ? o nome das variáveis são legíveis ? os métodos possuem uma quantidades de linhas do padrão ? a classe do mesmo pacote possui responsabilidades semelhantes ? há trechos de códigos repetidos ?. A equipe deve adotar

esse conjunto de critérios para que todos os membros desenvolvam utilizando o mesmo padrão. Caso haja alguma não conformidade no código da feature, a etapa de desenvolvimento deve ser reexecutada.

Nos casos em que a etapa de revisão não dectectou defeitos/não conformidade do codigo, a próxima atividade é a execução dos testes de unidade. Esses testes vão garantir que a adição da feature no sistema não ocasionou nenhum efeito colateral. A equipe que irá realizar essa atividade tem um roteiro de teste para seguir.

Por fim, é realizada a integração da nova *feature* desenvolvida ao projeto e uma versão para publicação é gerada. Atividades como geração de build, criação de pacotes para publicação são executadas. A Figura 21 apresenta o processo de implementação da prática construir por *feature*.

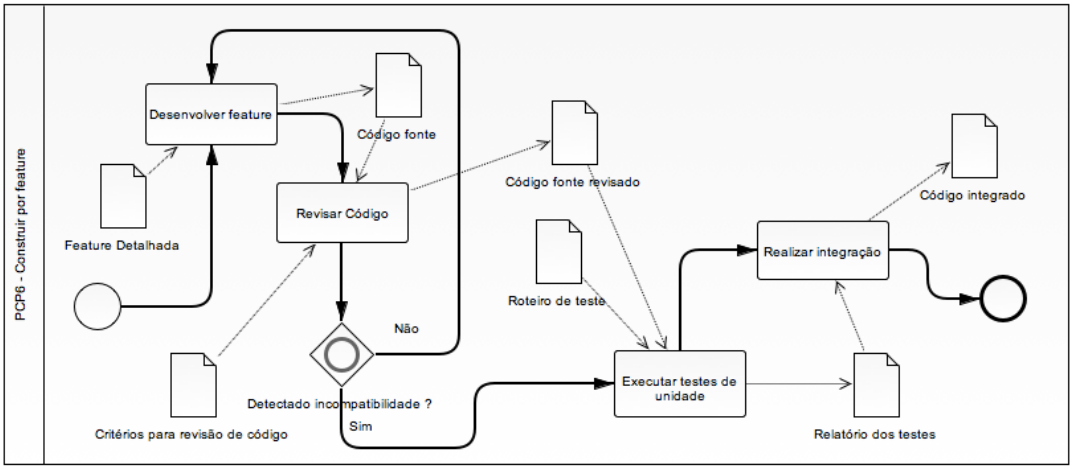

Figura 21. Processo de execução da prática construir por *feature*.

c.2) Grau de apoio

Total, pois a *feature* é implementada e o seu código passa por um processo de verificação para atestar que defeito/não conformidade não sejam entregues.

c.3) Ferramenta de apoio

Para a atividade de desenvolvimento da *feature*, a utilização de um ambiente integrado de desenvolvimento - IDE é essencial.

Para revisão de código, dependendo da linguagem de programação, há *plugins* que são adicionados na IDE que possibilitam a execução automática para a identificação de possíveis não conformidade no código fonte de acordo com o padrão definido.

Para a execução dos testes de unidade, as linguagens fornecem Bibliotecas para construção de testes unitário e de unidade.

Para a atividade de integração, há ferramentas específicas que realizam a geração do *build* de maneira contínua. Essas ferramentas são chamadas de ferramentas de integração contínua.

d) Relacionamento com outras práticas ágeis

Para a atividade de revisão de código fonte, a prática de programação em par do XP pode auxiliar na obtenção de melhores resultados da revisão.

e) Conflito com outras práticas ágeis

Práticas que utilizam outro modelo de especificação de requisitos, como por exemplo Caso de Uso.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Médio
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Média
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

Os roteiros de teste devem ser concebidos no momento do detalhamento da feature, atividade que não está contemplada nessa prática.

4.4.6.2 Programação por Pares

a) Origem da prática

Extreme Programming (XP).

b) Propósito

Técnica utilizada para a construção do produto na qual duas pessoas realizam o desenvolvimento juntas.

c) Plano para a implementação da prática

c.1) Forma de implementação

Para a implementação da programação por pares, primeiramente é necessário identificar o requisito que irá ser desenvolvido por par. Na atividade de desenvolvimento por par os dois desenvolvedores precisam assumir o papel de piloto, que conduz o processo de escrita do código e realiza o desenvolvimento fazendo o uso de um nível de abstração mais baixo que o co-piloto, papel que tem a função de realizar a análise do desenvolvimento do requisito em um nível de abstração mais elevado que o piloto. O co-piloto tem que ter uma visão mais sistêmica, já o piloto uma visão mais funcional. O diálogo entre o piloto e o co-piloto é algo constante em membros que realizam a programação por pares.

Em intervalos de tempo, os dois desenvolvedores devem se questionar se há a necessidade da realização da programação por pares. Caso haja, deve-se inverter os papéis, pois essa troca possibilita que o desenvolvedor atinja os dois níveis de abstração no desenvolvimento do requisito. A Figura 22 apresenta o processo de implementação da prática programação por pares.

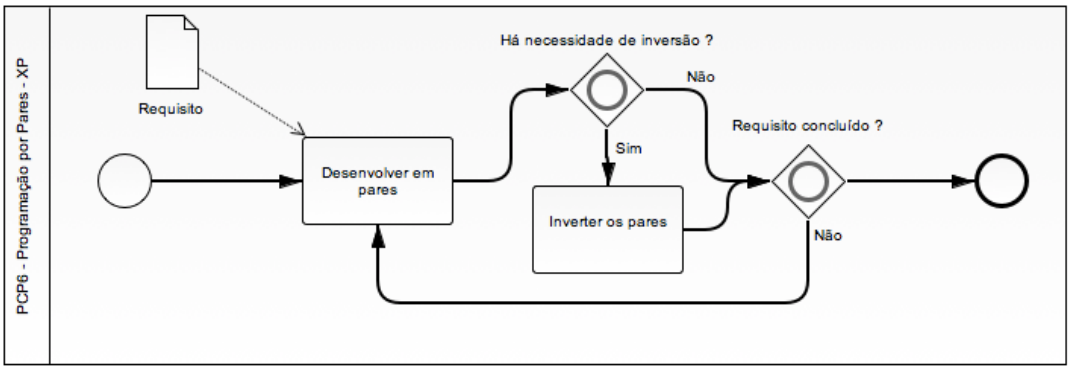

Figura 22. Processo de execução da Programação por Pares.

c.2) Grau de apoio

Parcial, pois há somente atividades de implementação do requisito, as atividades de verificação não estão evidentes na prática.

c.3) Ferramenta de apoio

Monitores maiores que o comum auxiliam no processo de leitura da código principalmente para o co-piloto. Utilização de mesas em formato de bancada favorece a conversa entre o par.

d) Relacionamento com outras práticas ágeis

A prática de programação em par pode ser associada a qualquer prática de desenvolvimento de requisitos.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto ou médio

2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Médio
2) Localização geográfica	Mesma sala
3) Tamanho da equipe	Média
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto ou médio
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

A programação por par deve ser utilizada para desenvolver de atividade complexas, para havendo interação entre o par. Não se recomenda a programação par para atividades simples e repetitivas.

4.4.6.3 Testes

a) Origem da prática

Extreme Programming (XP)

b) Propósito

Desenvolver e executar os testes durante o desenvolvimento do componente do produto.

c) Plano para a implementação da prática

c.1) Forma de implementação

A implementação dessa prática pressupõe que o componente do produto já está desenvolvido ou parcialmente desenvolvido. No caso de utilização de *UserStory* para especificação dos requisitos, os cenários para testes já estão inseridos na *UserStory*. Na atividade de especificar e projetar os testes, a equipe de testes, de posse do componente do produto a ser testado, realiza a especificação, onde podem ser descritos quais tipos de testes serão aplicados ao componente do produto. Testes de unidade e integração são projetados visando garantir a integração da aplicação.

De posse do projeto de teste, a execução dos testes é iniciada. Um relatório de incidente ou inconsistências dos testes é gerado. Caso haja alguma inconsistência, a equipe que desenvolveu o requisito é notificada. A Figura 23 apresenta o processo de implementação da prática testes.

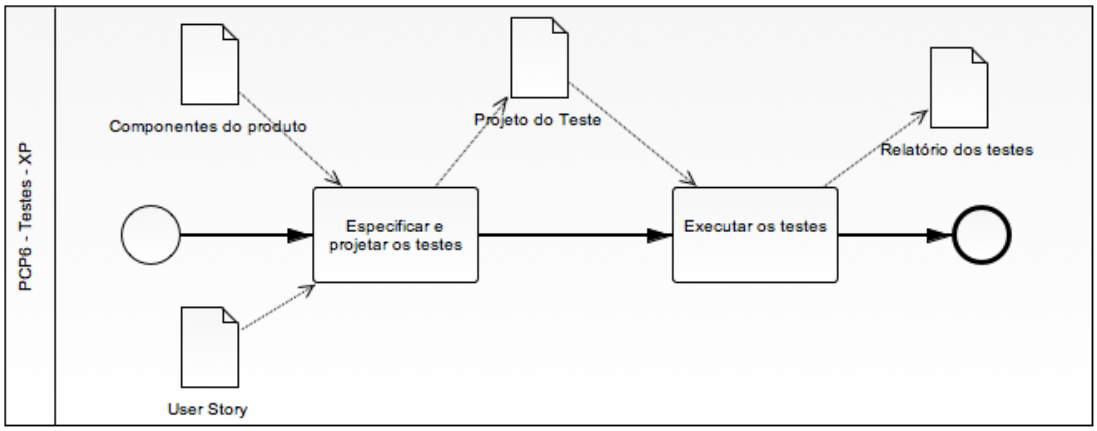

Figura 23. Processo de execução da prática Testes

c.2) Grau de apoio

Parcial, pois somente as atividades de verificação do componente do produto implementado são realizadas.

c.3) Ferramenta de apoio

Ferramentas de construção automática auxiliam na continuidade do desenvolvimento. A cada nova interação da equipe de desenvolvimento com a funcionalidade, o conjunto de testes é executado automaticamente.

As ferramentas de gestão de *bugs* (Ferramentas de *BugTracking*) podem ser utilizadas para registro das falhas ocorridas na execução dos testes.

d) Relacionamento com outras práticas ágeis

Programação por pares pode ser utilizada no processo de especificação e execução dos testes, dois desenvolvedores especificam e realizam a execução dos testes. Essa prática pode ser utilizada na realização das testes que envolvam uma revisão de código ou testes exploratórios.

e) Conflito com outras práticas ágeis

Não deve ser utilizada em ambientes que realizam especificação de requisitos com base em casos de usos, pois a entrada para a especificação dos testes é a *UserStory*.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Médio
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Média ou pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto ou médio
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

Testes de aceitação podem ser definidos e descritos no detalhamento da *UserStory*.

4.4.6.4 Refatoração

a) Origem da prática

Extreme Programming (XP)

b) Propósito

Otimizar o código interno do componente do produto sem a alteração do seu comportamento externo.

c) Plano para a implementação da prática

c.1) Forma de implementação

Para iniciar a prática da refatoração é necessário que haja um componente do produto em desenvolvimento ou concluído. Além disso, um conjunto de boas práticas de programação, padrões de projetos e estilos de programação definidos pela equipe dão a base para a identificação e o projeto da melhoria. Trechos de código que apresentam incompatibilidades, devidamente justificado pela equipe de desenvolvimento, estão sujeita à prática de refatoração.

Após a identificação e o projeto da melhoria, a atividade de implementação é iniciada. Os desenvolvedores ajustarão o código fonte da aplicação buscando atingir a melhoria proposta. Essa atividade deve ser documentada, mesmo que em código fonte, para preservar a melhoria. A Figura 24 apresenta o processo de implementação da prática refatoração.

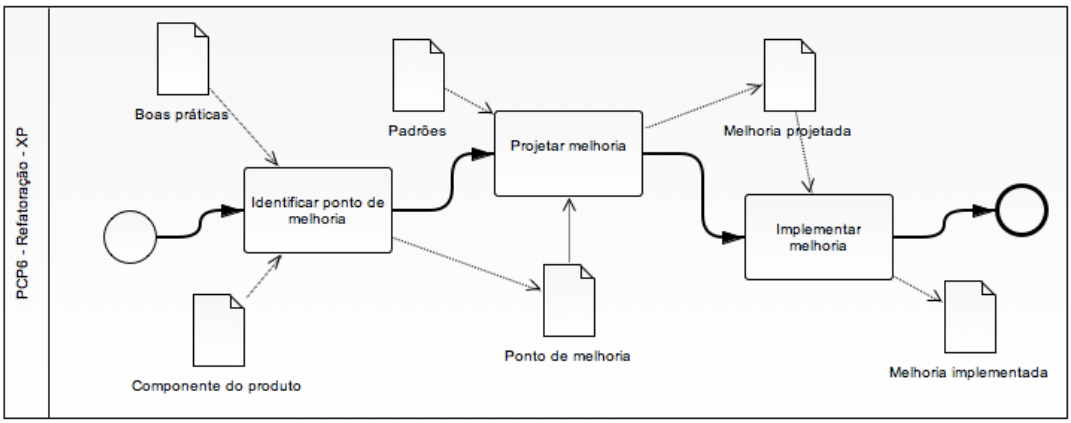

Figura 24. Processo de execução da prática de Refatoração

c.2) Grau de apoio

Parcial, pois as atividades de implementação do código fonte é realizada. Não tem-se evidência que o processo de verificação é realizado, por conta do objetivo principal da prática que é a permanência do comportamento externo do código, mesmo realizando melhorias/modificações internas do código.

c.3) Ferramenta de apoio

Ferramentas que identificam estilos de código automaticamente podem ser utilizadas na etapa de identificação do ponto de melhoria.

Para o projeto da melhoria, diagramas UML (classe, sequência e atividades) podem ser utilizados.

d) Relacionamento com outras práticas ágeis

A programação por pares pode ser aplicada em todas as atividades, pois a visão de dois desenvolvedores para discutir o projeto e a forma de implementação pode levar a melhor qualidade na execução da prática.

A associação com a prática de Testes pode ser executada para garantir que o comportamento externo do requisito não será alterado, sendo isso de forma automatizada.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Baixo
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Média ou pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

O Processo de refatoração deve ser priorizado entre os membros da equipe, pois a aplicação de inúmeras refatorações em sequência pode ocasionar um trabalho que "não tem fim", haja vista que se pode implementar um requisito de diversas maneiras.

4.4.6.5 Integração contínua

a) Origem da prática

Extreme Programming (XP)

b) Propósito

Estabelecer uma política de integração dos componentes do produto por meio de ferramentas automatizadas.

c) Plano para a implementação da prática

c.1) Forma de implementação

As tecnologias utilizadas no projeto são insumos para a atividade de projetar a integração contínua. Para cada tipo de tecnologia devem ser identificadas quais ferramentas poderão automatizar o processo de integração. A execução de integração contínua ocorre quando há um novo componente do produto no repositório do projeto ou quando há alguma modificação. Essa atividade, por ser automatizada e automática, em caso de falha, uma notificação é gerada. Processo de construção e execução das suítes de testes (unitário, unidade, integração e aceitação) fazem parte da atividade.

A Figura 25 apresenta o processo de implementação da prática integração contínua.

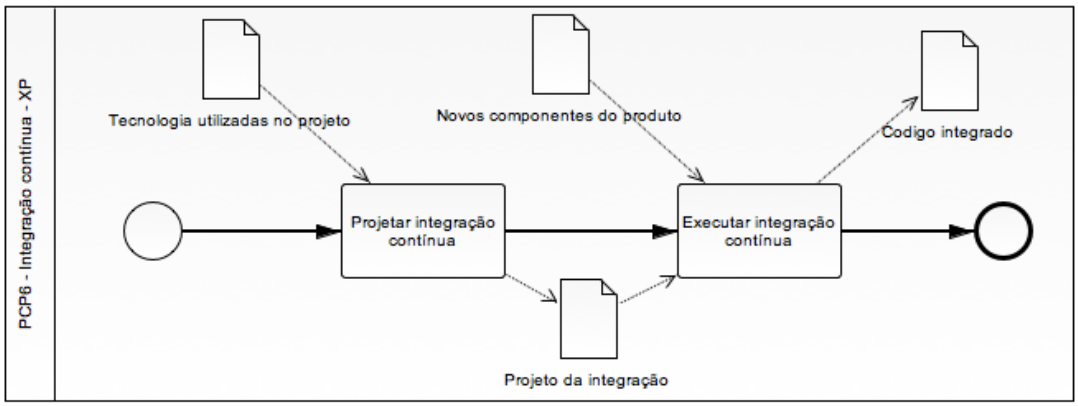

Figura 25. Processo de execução da prática Integração contínua

c.2) Ferramenta de apoio

Parcial, pois nessa prática as atividades de implementação do código fonte não são contempladas. Somente os procedimentos de verificação, por meio da execução da integração contínua são realizados.

c.3) Ferramenta de apoio

Ferramentas de construção automática, de controle de versão e de configuração das tecnologias do projeto são fundamentais para a aplicação dessa prática, isso por que o processo de integração contínua deve exigir o mínimo de intervenção da equipe de desenvolvimento na sua execução.

d) Relacionamento com outras práticas ágeis

A prática de Testes possui relação com a integração contínua no sentido da prática de teste ser colocada como uma sub-rotina na sua execução.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Baixo
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Grande ou Média
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

Ferramentas de construção automáticas são recomendadas para a execução dessa prática, pois diminuem o processo repetitivo evitando falha humana no processo.

4.4.6.6 Esqueleto ambulante

a) Origem da prática

Crystal

b) Propósito

Desenvolvimento por meio de pequenas aplicações de forma simples que pode se integrar aos demais componentes.

c) Plano para a implementação da prática

c.1) Forma de implementação

A implementação da prática inicia com o projeto de pequenos módulos do produto. Para isso o documento de visão do projeto, a estrutura analítica do projeto (EAP) e a analise de requisitos são importantes no intuito de fornecer a visão de quais conjuntos de funcionalidades podem ser reunidas para o desenvolvimento em um pequeno módulo. Ainda nessa atividade, as tecnologias e a arquitetura do projeto já são definidas.

Dado o conjunto de funcionalidades, agrupada em pequenos módulos, a equipe inicia o desenvolvimento. A filosofia dessa prática é a separação dos componentes para reduzir a sua complexidade.

Na atividade de integração as funcionalidade são incorporadas na versão atual e dessa forma o desenvolvimento evolui ao longo das interações. A Figura 26 apresenta o processo de implementação da prática esqueleto ambulante.

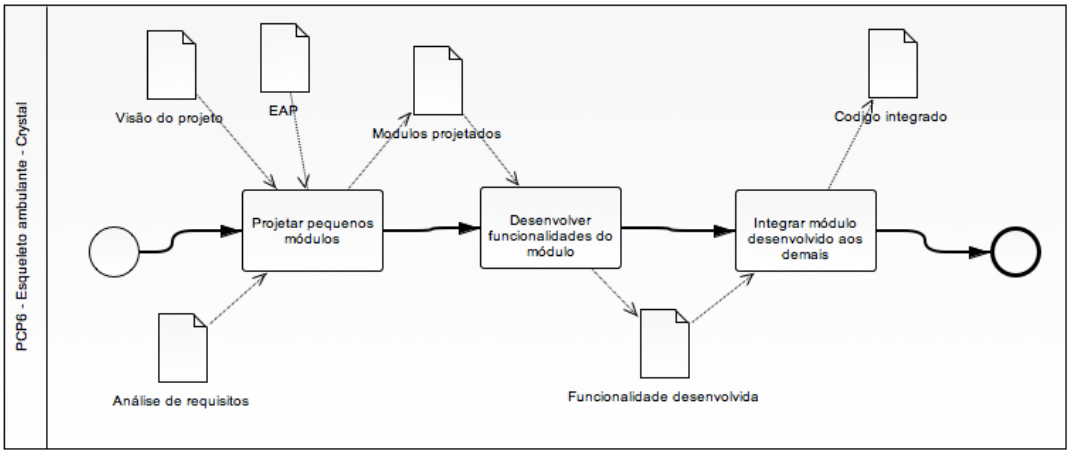

Figura 26. Processo de execução da prática Esqueleto ambulante

c.2) Grau de apoio

Parcial, pois à prática não possui atividades relacionadas a verificação dos componentes do produto.

c.3) Ferramenta de apoio

Utilização de componentes de software pode ajudar na integração da aplicação. A Construção de diagramas de componente auxilia no processo de identificação das dependências dos módulos.

d) Relacionamento com outras práticas ágeis

Para o desenvolvimento das funcionalidades a programação pode auxiliar no intuito de avaliar as melhores práticas visando a integração a posteriori. A aplicação das práticas de integração contínua e de testes, podem agilizar o processo de verificação da integração de maneira automatizada.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim

2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Média
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Qualquer
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

A estrutura analítica do projeto (EAP) pode auxiliar no processo de elaboração dos módulos, haja vista que ela já fornece uma separação em um maior nível de abstração, facilitando assim o projeto dos módulos.

4.4.6.7 Programação lado a lado

a) Origem da prática

Crystal

b) Propósito

Dois desenvolvedores trabalham lado a lado na mesma funcionalidade, cada um em um terminal, possibilitando que a visualização do objetivo desenvolvido seja para os dois.

c) Plano para a implementação da prática

c.1) Forma de implementação

A realização do projeto da funcionalidade e o detalhamento das tarefas são resultados da análise dos requisitos e dos diagramas do projeto. O par realiza a análise e projeto da funcionalidade juntos.

Para o desenvolvimento lado a lado, os dois desenvolvedores selecionam tarefas para fazer e sentam um ao lado do outro, trabalhando em computadores distintos. Cada desenvolvedor realiza a codificação da funcionalidade, mas pode visualizar a execução da atividade do par. Essa disposição facilita a comunicação e a visualização da codificação do par.

A Figura 27 apresenta o processo de implementação da prática programação lado a lado.

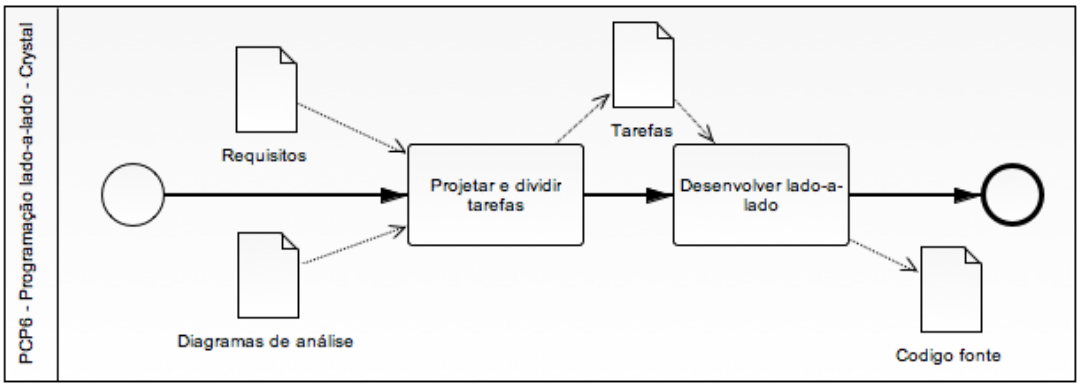

Figura 27. Processo de execução da prática Programação lado a lado

c.2) Grau de apoio

Parcial, pois na execução das atividades da prática não é evidente a execução de verificação da implementação do código fonte.

c.3) Ferramenta de apoio

Estações de trabalho em bancadas facilita a visualização das atividades desenvolvidas pelo par. Ferramentas de diagrama UML auxiliam na atividade de projetar a funcionalidade.

d) Relacionamento com outras práticas ágeis

Não aplica.

e) Conflito com outras práticas ágeis

Na Programação por par, os desenvolvedores estão fazendo o uso de uma única máquina.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Média
2) Localização geográfica	Mesma sala
3) Tamanho da equipe	Média ou pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

A prática da programação lado a lado pode ser aplicada em qualquer contexto de problema, pois além de possibilitar a divisão das tarefas entre os membros, facilita a comunicação direta.

4.4.6.8 Projetar solução

a) Origem da prática

Test Driven Development (TDD).

b) Propósito

Realizar a construção de cenários de testes para a verificação do componente do produto. A verificação está relacionada à capacidade do componente do produto em atender o que foi projetado

c) Plano para a implementação da prática

c.1) Forma de implementação

A execução da prática inicia pelo projeto e codificação do teste. Com base no requisito e seu teste de aceitação, são desenvolvido os testes, na sua maioria unitário antes mesmo do desenvolvimento da funcionalidade.

À medida que o desenvolvimento dos testes vai avançando, o desenvolvimento da funcionalidade é iniciado. A execução dos testes é um processo contínuo, pois a cada cenário, novos testes são adicionados e novos códigos da funcionalidade são implementados. A melhoria do código, por meio da refatoração, é executado a medida que o cenário de teste for contemplado e o código de desenvolvimento não esteja com inconsistências.

A Figura 28 apresenta o processo de implementação da prática projetar solução.

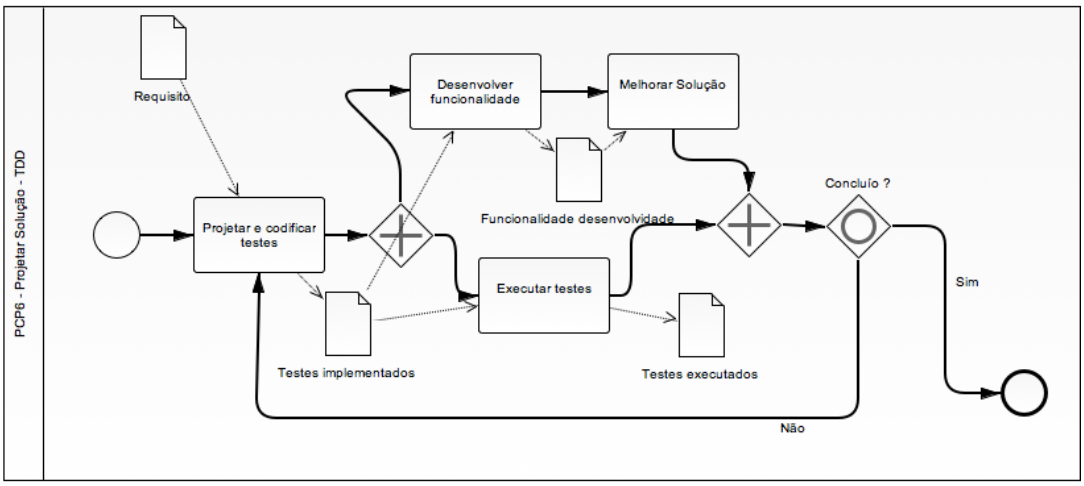

Figura 28. Processo de execução da prática Projetar Solução

c.2) Grau de apoio

Total, pois à medida que à progresso na implementação do código fonte, os testes são executados visando garantir a verificação do código fonte implementado.

c.3) Ferramenta de apoio

API de teste e ferramenta para execução dos teste fornecem apoio para a implementação dessa prática.

d) Relacionamento com outras práticas ágeis

A refatoração é aplicada na prática no momento em que os testes estão executado e código não apresenta inconsistências. A prática de Testes pode ser aplicada na etapa de execução dos testes.

A programação por pares e a programação lado-a-lado podem trazer bons resultados para a execução do desenvolvimento da funcionalidade ou dos testes. Na programação lado-a lado, um desenvolvedor pode construir o cenário de teste e o outro desenvolve a código que vai ser executado com sucesso.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alto
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Pequena
2) Localização geográfica	Mesma sala
3) Tamanho da equipe	Média ou pequena
4) Experiência da equipe com desenvolvimento em engenharia de software	Alto
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

Ferramentas para compilação e execução dos testes de forma simples e rápida devem ser utilizadas para possibilitar maior eficiência da prática.

4.4.6.9 Relacionamento entre as práticas

a) Propósito

Relacionar as práticas da seção do PCP6 visando o apoio a implementação do resultado esperado.

b) Plano para a implementação da prática:

b.1) Forma de implementação

O relacionamentos entre as práticas dá-se pela junção das práticas: Esqueleto ambulante, Programação por pares, Refatoração, Testes e Integração contínua. A Figura 29 apresenta o processo de implementação do relacionamento das práticas do PCP6.

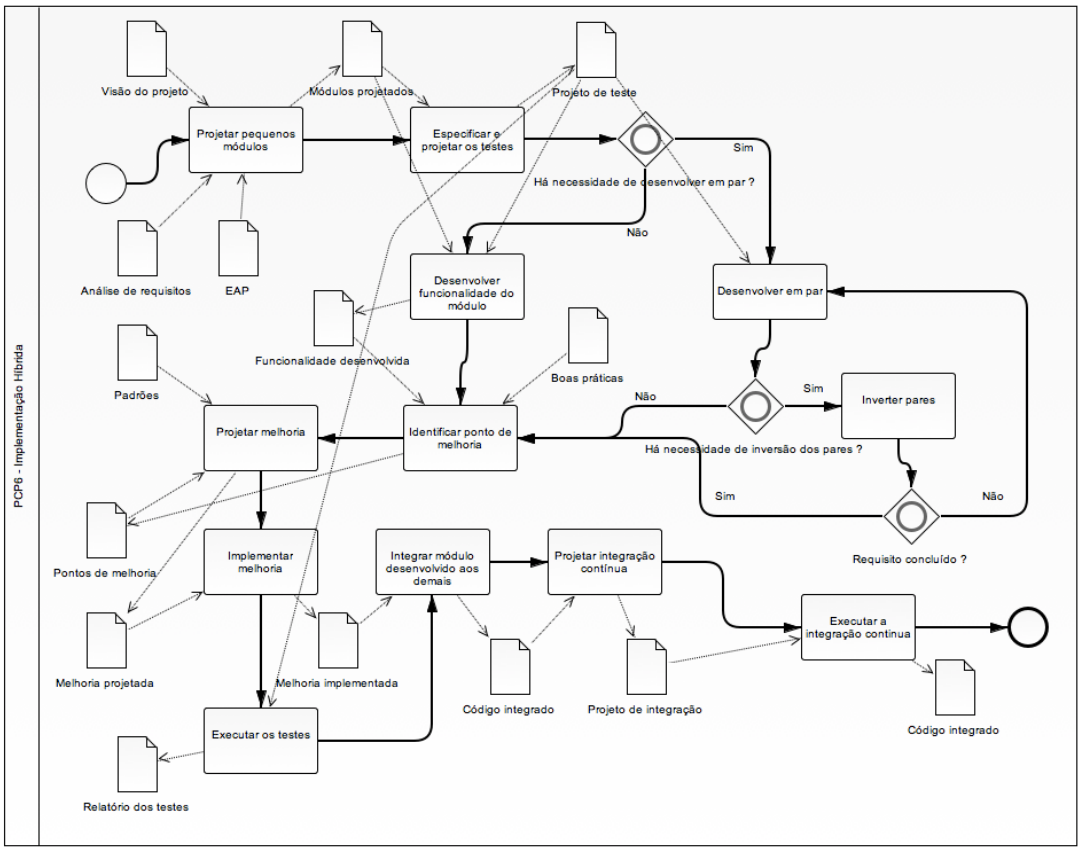

Figura 29. Implementação do relacionamento das práticas do PCP6

4.4.7 PCP7 - A documentação é identificada, desenvolvida e disponibilizada de acordo com os padrões estabelecidos

Objetivo: identificar qual a documentação para o projeto e para o usuário final.

4.4.7.1 Documentação Abrangente

a) Origem da prática

Extreme Programming (XP)

b) Propósito

A equipe de desenvolvimento deve definir quais documentos serão necessários para o projeto.

c) Plano para a implementação da prática

c.1) Forma de implementação

Para a implementação da prática a equipe de desenvolvimento do projeto realiza uma reunião para identificar quais artefatos serão gerados para o projeto. O intuito dessa reunião é eliminar qualquer artefato que não agregue valor para o projeto, para isso o documento de visão do projeto faz-se necessário. Como exemplo de documentação pode-se citar: manuais do usuário final, diagramas e fluxogramas do projeto, tutoriais, glossário de dados, entre outros.

Após a identificação da documentação, a equipe reúne com os *stakeholders* do projeto para definição da lista final dos documentos. Esse reunião deve ser conduzida de modo a identificar os artefatos de suma importância para o cliente e para a equipe. Ao final, deve ser gerada esta lista com todos os artefatos que devem ser criados a mantidos.

A Figura 30 apresenta o processo de implementação da prática documentação abrangente.

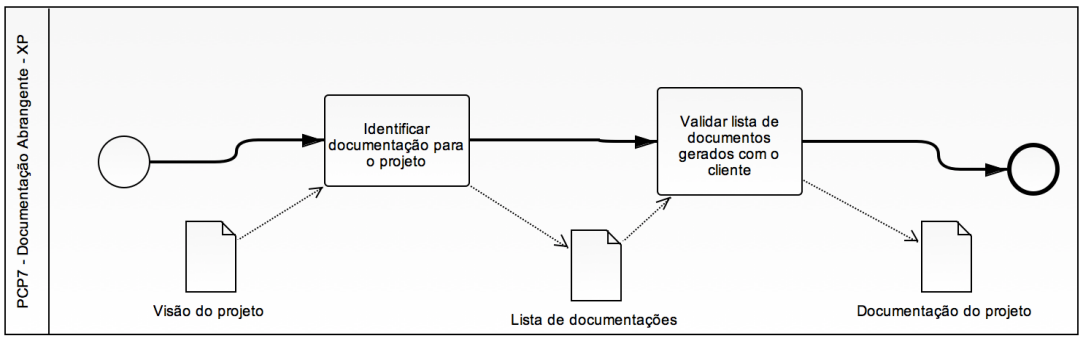

Figura 30. Processo para execução da prática documentação abrangente

c.2) Grau de apoio

Total, pois nas atividades da prática são identificados, desenvolvidos e validados com os interessados.

c.3) Ferramenta de apoio

Quadro branco pode ser utilizado para facilitar a exposição dos itens de identificação da documentação para o projeto.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alta
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Médio
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Qualquer
4) Experiência da equipe com desenvolvimento em engenharia de software	Alta
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

A documentação do projeto gerada como resultado da execução desta prática deve contemplar principalmente se a documentação é relevante para o projeto. Documentação que não for utilizada não deve ser desenvolvida.

4.4.7.2 Manual do Usuário

a) Origem da prática

Crystal.

b) Propósito

O manual do usuário é um produto de trabalho e deve estar integrado ao desenvolvimento de cada funcionalidade.

c) Plano para a implementação da prática

c.1) Forma de implementação

Essa prática é iniciada à medida que a funcionalidade for desenvolvida. A equipe de desenvolvimento realiza a construção da documentação do usuário. Ao final desse desenvolvimento, o manual do usuário é atualizado para a nova funcionalidade.

Após a construção, a equipe de teste realiza a verificação para identificar se a documentação está nos padrões definidos pela equipe para o projeto. Modelos adotados para a instituição podem ser utilizados para o projeto. Para a equipe que utilizar o conceito de "definição de pronto", a referida documentação pode ser adicionada.

A Figura 31 apresenta o processo de implementação da prática manual do usuário.

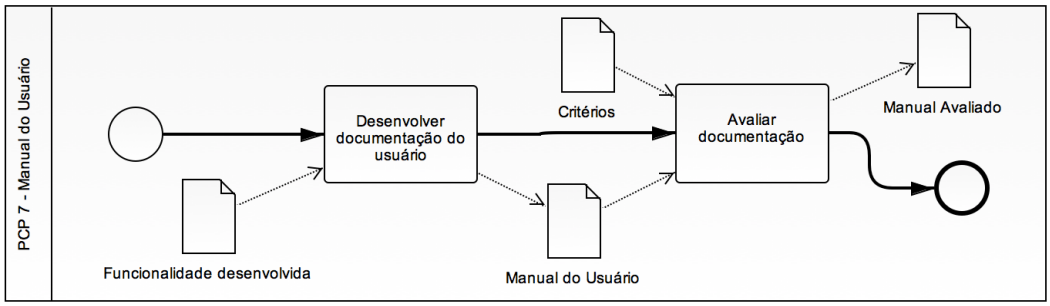

Figura 31. Processo para execução da prática Manual do Usuário

c.2) Grau de apoio

Parcial, pois a prática está diretamente relacionada ao desenvolvimento do manual do usuário, não levando em consideração outros tipos de documentações.

c.3) Ferramenta de apoio

Editores de texto, planilhas eletrônicas e ferramentas de edição de imagem e vídeo são úteis no processo de construção dos manuais.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Baixa
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Alto
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Qualquer
4) Experiência da equipe com desenvolvimento em engenharia de software	Baixa
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

Os critérios para a avaliação do manual do usuário são definidos pela equipe e respeitando as restrições de tecnológicas do projeto.

4.4.7.3) Construir por *Feature*

a) Origem da prática

Feature Driven Development (FDD)

b) Propósito

É necessário o desenvolvimento da documentação do *design* e do usuário, sendo que para cada nova *feature* toda a documentação deve ser atualizada.

c) Plano para a implementação da prática

c.1) Forma de implementação

Essa prática tem seu início à medida que as funcionalidades são desenvolvidas. A atividade de desenvolvimento da documentação da funcionalidade irá atualizar os diagramas de atividades, de classes e qualquer outro modelo utilizado na etapa de projeto da funcionalidade.

Já na atividade de atualização do documento de design do projeto, as atualizações das documentações geradas são incorporadas a essa documentação. O produto gerado por essa atividade é o documento de arquitetura atualizado.

Práticas relacionadas ao controle de versão da documentação podem ser utilizadas para promover melhor gestão dos artefatos atualizados. A Figura 32 apresenta o processo de implementação da prática construir por *feature*.

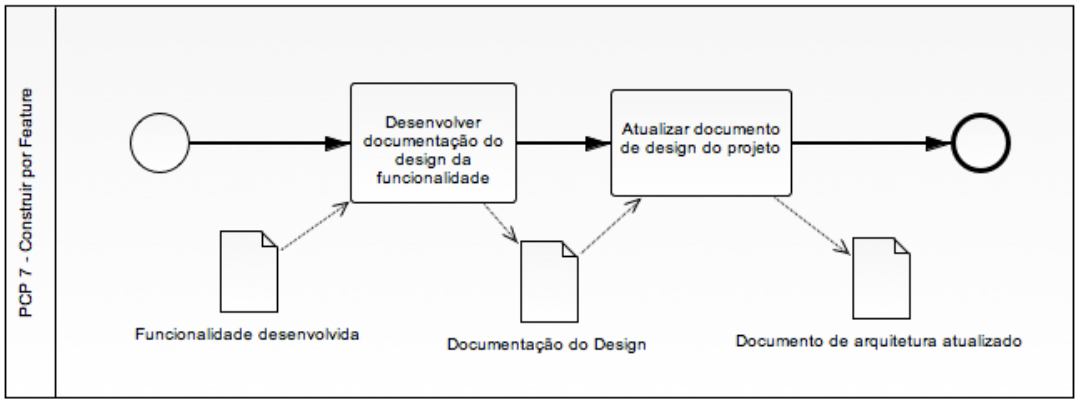

Figura 32. Processo de execução da prática construir por feature.

c.2) Grau de apoio

Parcial, pois a prática está relacionada à construção da documentação do design do projeto, não levando em consideração os outros tipos de documentações.

c.3) Ferramenta de apoio

Editores de texto, ferramentas UML e sistema de controle de versão.

d) Relacionamento com outras práticas ágeis

Essa prática possui relacionamento com a prática Manual do Usuário, pois pode ser aplicada em paralelo, realizando a atualização tanto da documentação técnica quando do usuário.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alta
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Baixo
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Média
4) Experiência da equipe com desenvolvimento em engenharia de software	Alta
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

A atualização do design do projeto deve atender aos padrões definidos para o projeto ou pela organização.

4.4.7.4 Projetar Solução

a) Origem da prática

Test Driven Development (TDD).

b) Propósito

Essa prática indica que deve ser necessário atualizar as documentações do cenário de teste e das classes do projeto.

c) Plano para a implementação da prática

c.1) Forma de implementação

Essa prática é iniciada quando o teste da funcionalidade é realizado. Essa atividade pode acontecer durante o desenvolvimento dos testes, haja vista que, à medida que a prática do desenvolvimento orientado aos testes é executada, uma documentação no código fonte já pode ser realizada. A documentação envolve além do código fonte, atualizações do cenário de teste, que podem evoluir ao longo do desenvolvimento.

Para garantir a utilização dos padrões definidos pela equipe, a atividade de inspeção da documentação ao final da sua construção remete à busca da qualidade presumida. Identificar inconsistências e lacunas na documentação é a principal tarefa dessa atividade. Ao final, a documentação do projeto é atualizada.

A Figura 33 apresenta o processo de implementação da prática projetar solução.

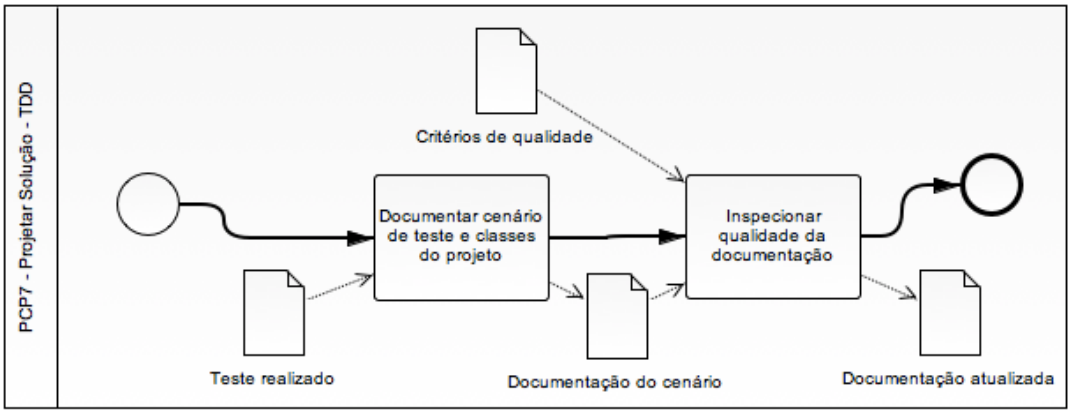

Figura 33. Processo de execução da prática Projetar Solução

c.2) Grau de apoio

Parcial, pois a prática atende somente à documentação dos cenários de teste e classes do projeto, não levando em consideração os outros tipos de documentações.

c.3) Ferramenta de apoio

- IDE para documentação do código fonte.
- Ferramenta *Wiki* para documentar os cenários de teste.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Práticas que não são baseadas em cenários de teste.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alta
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Média
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Pequena ou média
4) Experiência da equipe com desenvolvimento em engenharia de software	Alta
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

Critérios de qualidade devem ser descritos pela equipe de desenvolvimento.

4.4.7.5 Relacionamento entre as práticas

a) Propósito

Apoiar a implementação do PCP7, por meio das práticas Manual do Usuário e Construir por Feature do Crystal.

b) Plano para a implementação da prática:

b.1) Forma de implementação

A composição das práticas dá-se pela aplicação em paralelo, com isso aspectos do projeto e do usuário são contemplados. Nesse relacionamento das práticas temos a atividade de avaliação da documentação sendo executada para as duas práticas. A Figura 34 apresenta o processo de implementação do relacionamento das práticas do PCP7.

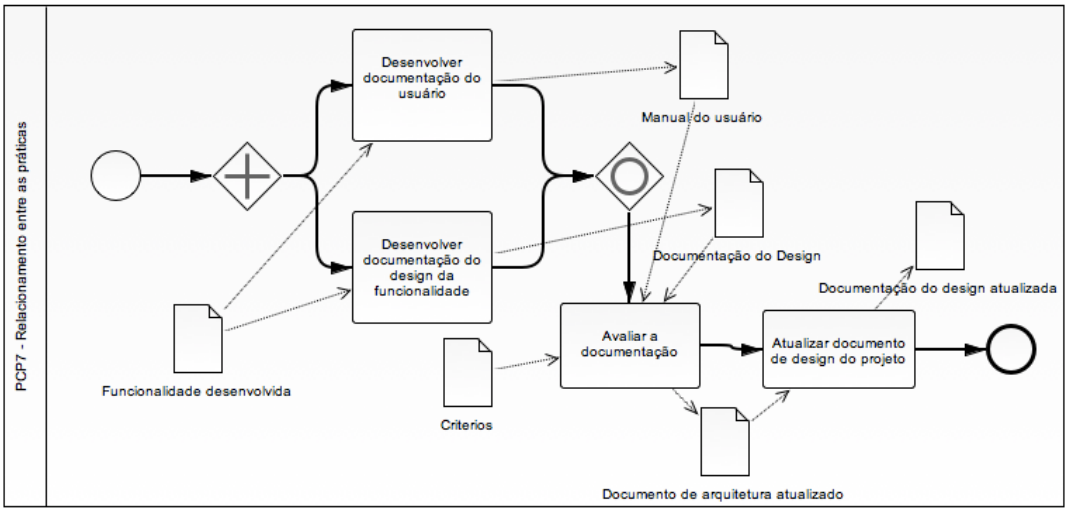

Figura 34. Implementação do relacionamento das práticas do PCP7.

4.4.8 PCP8 - A documentação é mantida de acordo com os critérios definidos

Objetivo: manter a documentação consistente e organizada do projeto.

4.4.8.1 Documentação Abrangente

a) Origem da prática

Extreme Programming (XP).

b) Propósito

A equipe de desenvolvimento deve manter a documentação necessária para o andamento do projeto.

c) Plano para a implementação da prática

c.1) Forma de implementação

Com base na lista de documentos, ou seja, artefatos que foram identificados como sendo os fundamentais para o projeto, é iniciada a atividade de identificação dos padrões a serem utilizados nessas documentações. Essa identificação consiste na definição de padrões, formas e modelos para a atualização da documentação. Para a documentação do código fonte, por exemplo, pode-se definir que para cada método de negócio tenha a descrição do seu comportamento. Como resultado dessa atividade são gerados os padrões para documentação.

Na atividade de atualização da documentação do projeto, para cada mudança no projeto que acarrete na atualização da documentação, são utilizados os padrões definidos na atividade anterior. O resultado dessa atividade produz a atualização da documentação do projeto.

A Figura 35 apresenta o processo de implementação da prática documentação abrangente.

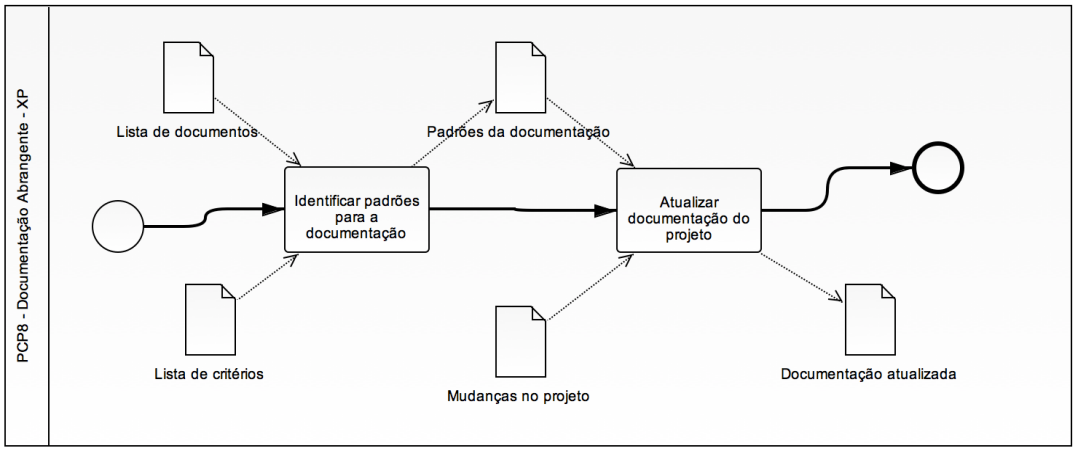

Figura 35. Processo de execução da prática Documentação abrangente

c.2) Grau de apoio

Total, pois os tipos de documentações do projetos são mantidas e os critérios são estabelecidos.

c.3) Ferramenta de apoio

Ferramentas *Wiki* auxiliam na atividade de disponibilização e manutenção dos padrões identificados para cada documentação do projeto.

Ferramenta de controle de versão auxilia na atividade de atualização da documentação, pois é responsável pelo armazenamento das versões da documentação do projeto.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alta
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Baixo
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Qualquer
4) Experiência da equipe com desenvolvimento em engenharia de software	Alta
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

A lista de critérios indicada como entrada para a atividade de identificação dos padrões para documentação do projeto deve ser obtida por meio de reunião entre os membros do projeto. Equipes podem adicionar na definição de "pronto" as atividades de atualização da documentação de acordo com os padrões estabelecidos.

4.4.8.2 Detalhar por *Feature*

a) Origem da prática

Feature Driven Development (FDD)

b) Propósito

O proposto para a execução da prática é atualizar a documentação do *design* para cada *feature* do projeto.

c) Plano para a implementação da prática

c.1) Forma de implementação

A prática possui apenas uma atividade, em que para cada *feature* detalhada é necessária a criação/atualização do *design* (projeto). A atualização deve ser realizada nos diagramas de classes, sequência e atividades. A Figura 36 apresenta o processo de implementação da prática detalhar por *feature*.

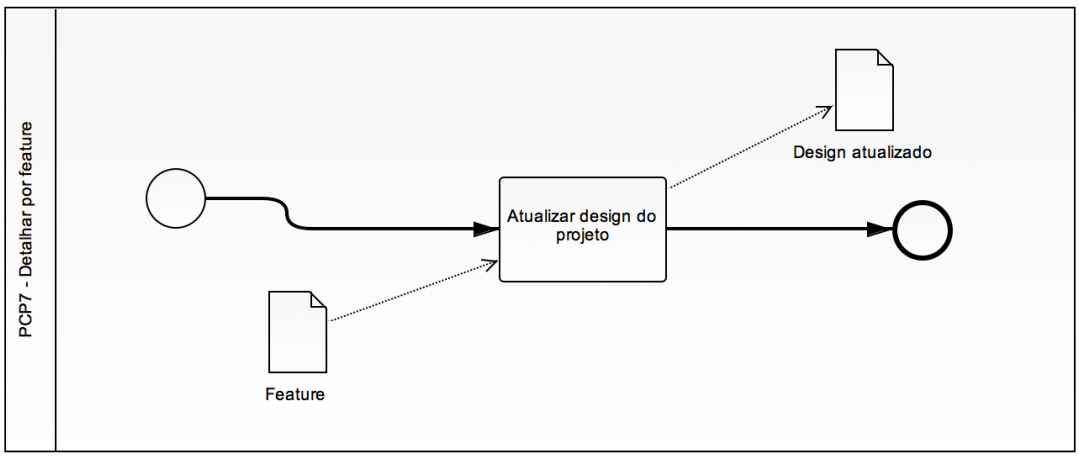

Figura 36. Processo de execução da prática Detalhar por feature

c.2) Grau de apoio

Parcial, pois a documentação mantida é somente o design do projeto.

c.3) Ferramenta de apoio

- Ferramentas de modelagem UML ajudam no processo de atualização dos diagramas.
- Ferramenta de controle de versão.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Práticas de documentação que não utilizam a *feature* como base.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alta
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Baixo
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Qualquer
4) Experiência da equipe com desenvolvimento em engenharia de software	Alta
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

Não se aplica.

4.4.8.3 Construir por *Feature*

a) Origem da prática

Feature Driven Development (FDD).

b) Propósito

Será necessário atualizar as documentações do projeto durante a construção do produto de acordo com a definição de "pronto" estabelecida.

c) Plano para a implementação da prática

c.1) Forma de implementação

Após o desenvolvimento da feature, ou mesmo durante o processo de desenvolvimento, é realizada a atualização da documentação do projeto. A definição de "pronto" indica quais os critérios mínimos aceitos para que a feature seja considerada "pronta". A atualização será realizada para o atendimento da definição de "pronto", que pode conter a documentação do código fonte, diagramas, fluxos, manuais dos usuários.

A Figura 37 apresenta o processo de implementação da prática construir por *feature*.

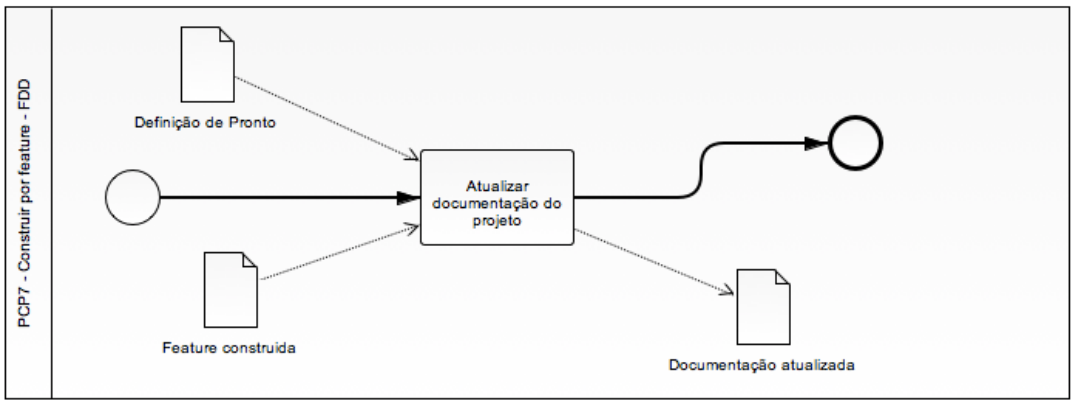

Figura 37. Processo de execução da prática Construir por feature

c.2) Grau de apoio

Parcial, pois não são identificados os critérios para a manutenção da documentação do projeto.

c.3) Ferramenta de apoio

Editores de texto, Ferramentas de controle de versão e Ferramentas de modelagem UML e BPMN.

d) Relacionamento com outras práticas ágeis

Há relacionamento com a prática detalhar por feature no contexto de apoio a implementação do PCP8, pois fazem parte da mesma origem da prática. Além disso, ambas apresentam como atividade a atualização de documentação, sendo a detalhar por *feature* do *design* do projeto e a construir por *feature* a documentação do projeto.

e) Conflito com outras práticas ágeis

Práticas de documentação que não utilizam a *feature* como base.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alta
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Baixo
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Qualquer
4) Experiência da equipe com desenvolvimento em engenharia de software	Alta
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

A definição de “pronto” deve ser estabelecida para cada projeto. Deve haver diálogo entre os membros da equipe de desenvolvimento com os *stackholders*.

4.4.8.4 Projetar Solução

a) Origem da prática

Test Driven Development (TDD)

b) Propósito

A documentação do projeto é desenvolvida e mantida a cada projeto de teste.

c) Plano para a implementação da prática

c.1) Forma de implementação

A atividade de desenvolvimento do cenário de teste para cada funcionalidade pode alterar outros cenários de testes. Para cada novo cenário, a documentação da funcionalidade é atualizada. A documentação do pacote de código fonte pode ser utilizada nessa atividade. Além disso, diagramas, cenários de aceitação, fluxogramas podem ser atualizados. Caso os cenários de testes projetados para a funcionalidade não tenham sido finalizados, um novo cenário é desenvolvido e por conseguinte a atualização da documentação é realizada.

A Figura 38 apresenta o processo de implementação da prática projetar solução.

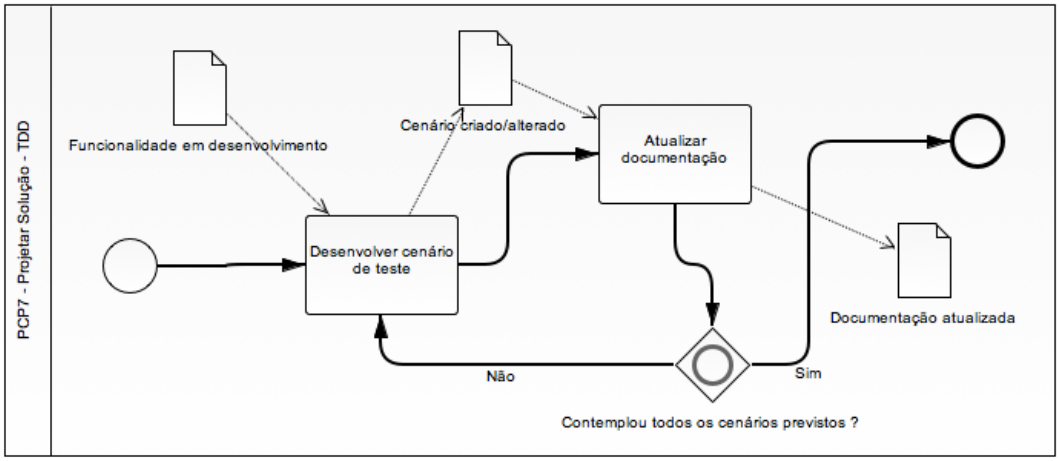

Figura 38. Processo de execução da prática Projetar solução

c.2) Grau de apoio

Parcial, pois somente a documentação relacionada aos cenários de testes são mantidos.

c.3) Ferramenta de apoio

IDE de desenvolvimento, Ferramentas e Bibliotecas de Teste de Software, Editores de texto e Ferramentas de diagramas UML.

d) Relacionamento com outras práticas ágeis

Não se aplica.

e) Conflito com outras práticas ágeis

Não se aplica.

f) Sugestão de perfil de organização e projeto para aplicação da prática

Características organizacionais	
Ambiente	
1) Provê a comunicação entre os membros da equipe de desenvolvimento	Sim
2) Facilita a cooperação da equipe de desenvolvimento	Sim
Equipe	
1) Nível de conhecimento em engenharia de software	Alta
2) Agregam mais de um perfil no escopo do processo de software	Sim
Caracteristicas em projeto	
1) Grau de acesso aos usuários	Médio
2) Localização geográfica	Qualquer
3) Tamanho da equipe	Qualquer
4) Experiência da equipe com desenvolvimento em engenharia de software	Alta
5) Há necessidade de entregas de produtos intermediários	Sim

g) Notas

A atividade de atualização da documentação está diretamente relacionada à adição de cenários de testes à documentação da funcionalidade.

4.4.8.5 Relacionamento entre as práticas

a) Propósito

Associar duas práticas do FDD, construir por feature e detalhar por *feature*, para atendimento do resultado esperado PCP8.

b) Plano para a implementação da prática:

b.1) Forma de implementação

A implementação dá-se pela junção das atividades de atualização do design do projeto seguida pela atividade de atualização da documentação do projeto. A Figura 39 apresenta o processo de implementação do relacionamento entre essas práticas do FDD no contexto de apoio a implementação ao PCP8.

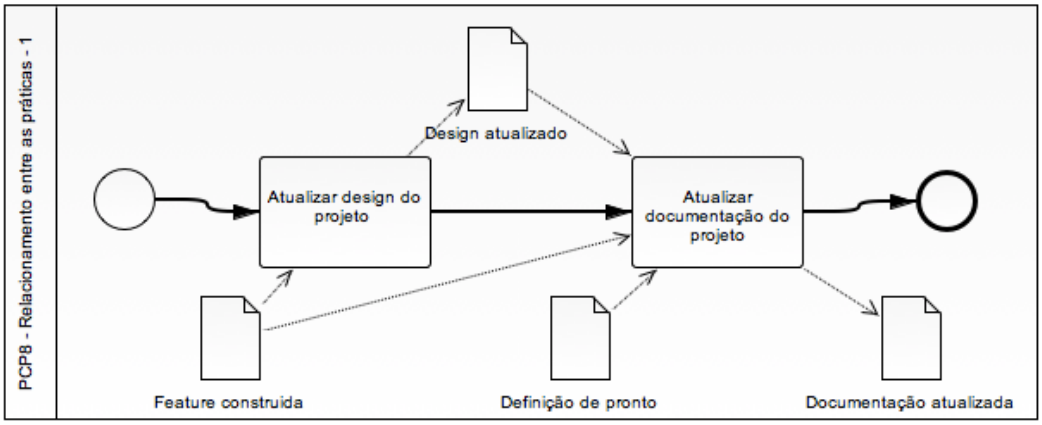

Figura 39. Processo de execução do relacionamento entre as práticas do PCP8

4.5. Como utilizar

A utilização do *framework* de práticas ágeis é bem simples, pois para a organização ou equipe que for utilizar, precisará inicialmente identificar qual resultado esperado deseja ter apoio e depois selecionar qual prática ágil utilizar. No plano de implementação da prática há um fluxo principal para execução da prática, e alinhado ao plano, o usuário pode identificar as ferramentas e o grau de apoio da prática para o resultado esperado.

Além do plano de implementação, o usuário poderá identificar os relacionamentos e os conflitos com as outras práticas ágeis. Esses relacionamentos ajudarão o usuário a compatibilizar a utilização das práticas, evitando conflitos de práticas na implementação.

Em caso de dúvida, o usuário pode consultar, para cada prática, a seção “Sugestão de perfil de organização e projeto para aplicação da prática”, para identificar aspectos relacionados ao perfil da organização e do projeto.

Objetivando um maior grau de apoio à prática, a seção “Relacionamento entre as práticas”, apresenta uma proposta que alinha duas ou mais práticas ágeis do mesmo resultado esperado para alcançar o apoio total.

5. AVALIAÇÃO DO FRAMEWORK

5.1. Contexto da Avaliação

No contexto desse trabalho a avaliação por pares esteve relacionada a dois momentos: na avaliação do relacionamento das práticas aos resultados esperados do processo; e na avaliação do *framework*.

5.2. O processo de revisão por pares

O processo de revisão por pares foi estruturado nas seguintes etapas: Identificação do Revisor; Desenvolvimento do Formulário de Revisão; Revisão inicial do trabalho; Ajustes/Correção dos itens indicados; e Revisão final.

Na etapa de identificação do revisor foi encaminhado o formulário para identificação do perfil do revisor para um pesquisador que atua na área de pesquisa desse trabalho. Anexo ao formulário de identificação do perfil do revisor, uma planilha contendo os critérios de avaliação foi encaminhada. O processo de avaliação por pares teve como categorias de avaliação os seguintes itens:

- **TA (Técnico Alto)**, indicando que foi encontrado um problema em um item que, se não for alterado, comprometerá as considerações;
- **TB (Técnico Baixo)**, indicando que foi encontrado um problema em um item que seria conveniente alterar;
- **E (Editorial)**, indicando que foi encontrado um erro de português ou que o texto pode ser melhorado;
- **Q (Questionamento)**, indicando que houve dúvidas quanto ao conteúdo das considerações;
- **G (Geral)**, indicando que o comentário é geral em relação às considerações.

- **N (Notas sobre as práticas)**, indicando que o comentário pode ser adicionado na forma de Notas para a prática ágil (somente aplicado para a avaliação do *framework*).

5.3. Avaliação do relacionamento das práticas ágeis aos resultados do processo

A principal justificativa para a execução dessa etapa é a avaliação do mapeamento realizado e assim fornecer insumos para o desenvolvimento da próxima atividade, a estruturação e o desenvolvimento do *framework*.

A revisão por pares do relacionamento das práticas ágeis aos resultados do processo foi realizada em Junho/2014, após a revisão da literatura (seção 3.1), do estado da arte dos métodos ágeis e dos resultados dos processo de construção da solução.

O formulário de avaliação enviado para o avaliador possuiu 9 questões, sendo 4 relacionadas à formação e experiência do especialista no processo de PCP e as demais relacionadas ao nível de conhecimento nos métodos indicados no trabalho (Crystal, FDD, TDD e XP).

De acordo com o formulário de avaliação, foi identificado que o revisor é avaliador líder inicial do MPS.BR, possui conhecimento alto na metodologia XP, médio conhecimento em TDD, FDD e Lean, e baixo conhecimento na metodologia Crystal. Sobre o PCP, o revisor atribuiu nível alto de conhecimento, por já ter implementado este processo em algumas unidades organizacionais da indústria de software nacional.

O resultado da revisão apontou o relacionamento dos resultados dos processos de construção da solução ao XP, TDD, FDD e Crystal como satisfatório e não indicou comentário para as categorias de avaliação.

5.4. Avaliação do *Framework*

O processo de revisão por pares aplicada ao *framework* teve como objetivo avaliar, tanto a estrutura quanto o conteúdo descritivo. Um formulário de avaliação com 16 questões foi enviado para o avaliador, sendo oito questões relacionadas à experiência do

avaliador relacionada aos modelos de qualidade e ao resultados dos processos de construção da solução, as demais questões foram específicas para identificar o nível de experiência do avaliador relacionado aos métodos contemplados neste trabalho.

Para realizar essa revisão foi escolhido um avaliador com experiência de mais de cinco anos em engenharia de software, conhecimento elevado sobre o processo projeto e construção do produto, tendo participado em três organizações da implantação do processo PCP. Além disso, possui certificação CMMI-DEV e MR-MPS-SW, já realizou avaliações de modelos qualidade em mais de seis organizações e tem experiência em avaliação de mais de cinco anos.

A escolha de um avaliador para realizar a revisão foi justificada pela indisponibilidade de pesquisadores para realização desse processo e pela complexidade e dimensão do framework de práticas ágeis. Além disso, o pesquisador selecionado possui bastante conhecimento e experiência na implementação de modelos de qualidade como MPS.BR e CMMI.

Foi fornecido ao avaliador um planilha eletrônica para ser utilizada como suporte para anotações acerca dos itens, contendo: a categoria (critério de avaliação); item em que a categoria foi aplicada; comentários com justificativa sobre o item avaliado; e sugestão para correção/melhoria.

O processo de avaliação por pares foi realizado no mês de Janeiro/2015, sendo em três momentos: no primeiro foi enviando para o avaliação o *framework* e a planilha eletrônica com os critérios de avaliação e um formulário de avaliação geral; no segundo momento, devido ao volume de informação contido no *framework*, foram realizadas reuniões remotas para a apresentação do seu conteúdo; e, por fim, o terceiro momento foi a apresentação por parte do avaliador da planilha de avaliação preenchida e o indicativo de sugestões/melhorias. O processo de revisão por pares teve duração de três meses, onde houveram dez encontros por vídeo conferência. Na Quadro 8 e na Figura 40 há o quantitativo indicado pelo avaliador por critério de avaliação.

Quadro 8. Dados da revisão por pares

Critérios	Quantidade
TA - Técnico Alto	2
TB - Técnico Baixo	18
E - Editorial	14
Q - Questionamento	0
G - Geral	3
N - Notas sobre as práticas	0

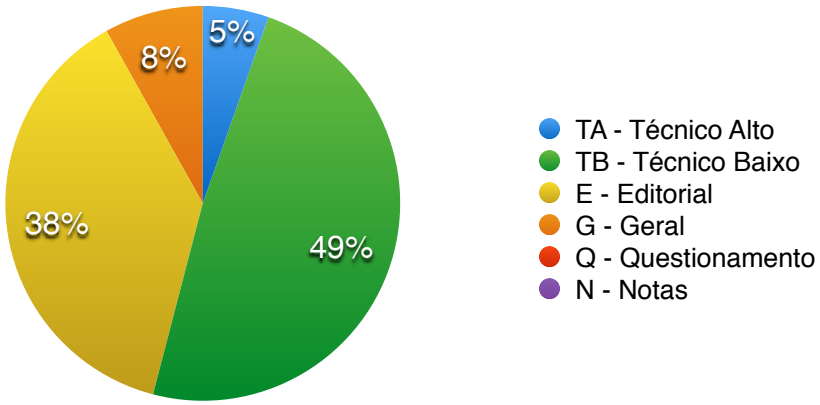

Figura 40. Gráfico do Resultado Revisão por pares

Após o recebimento da planilha de avaliação com os resultados da revisão por pares, todos os problemas identificados foram corrigidos, de acordo com a sugestão descrita pelo avaliador. Finalizadas as correções, o *framework* foi encaminhado novamente para o avaliador para aprovação. Todas as correções realizadas foram aprovadas pelo avaliador.

Diante de dois itens categorizados como TA - Técnico Alto, as seguintes correções foram realizadas, pois essa categoria denota que há um problema e se não for alterado, comprometerá as considerações, a saber:

- Comentário: “Justificar o item grau de apoio para todas as práticas”. Esse comentário foi considerado pertinente e para todas as práticas do *framework* foi justificado o grau de apoio, para ambos os casos (total ou parcial);

- Item: PCP3 - Prática padrões de codificação, comentário: “No item características em projeto “há a necessidade de entregas de produtos intermediário”, a resposta deve ser SIM, pois a prática gera produtos intermediários”. Nesse item a justificativa foi acatada e para o item a mudança de NÃO para SIM foi realizada.

Na Figura 41. foram identificados para cada resultado esperado os itens de avaliação TA - técnico alto, TB - técnico baixo e E - editorial. Nota-se que o PCP 6 possui o maior quantitativo de itens de avaliação TB(6) e E(7). Já o PCP 2 não contabilizou nenhum item de avaliação. A categoria G - geral não foi incluída como indicador para este figura, por causa de sua característica de ser aplicada a todos os resultados esperados. Alguns itens categorizados como TA, TB e E não estão indicados no gráfico pois foram aplicados a todo o *framework*.

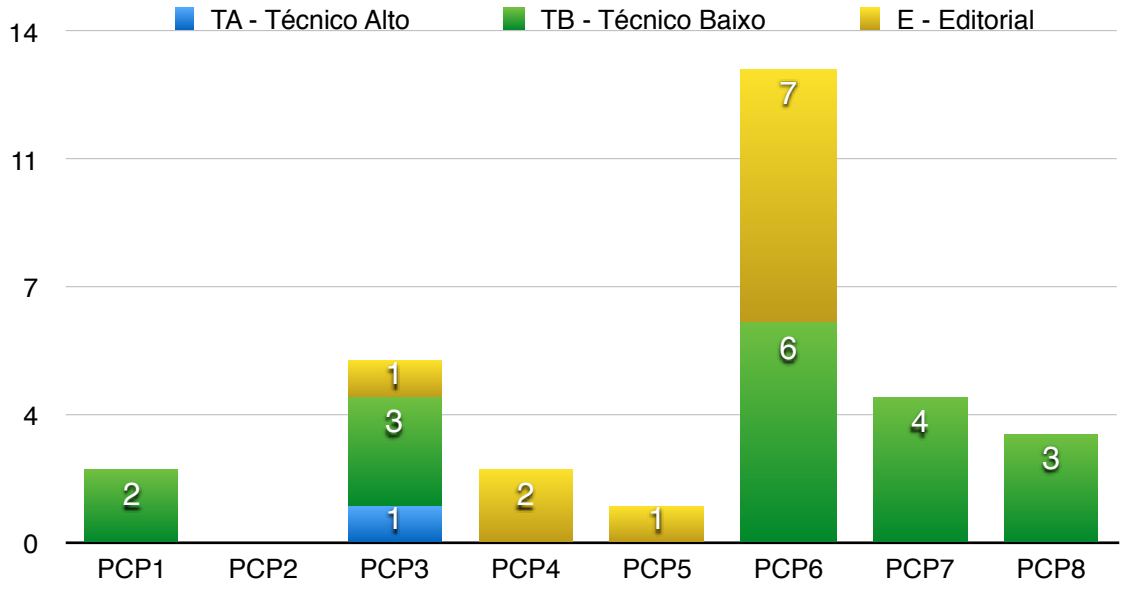

Figura 41. Itens de avaliação por resultado esperado

Sobre a formulário de avaliação geral, o avaliador classificou a proposta como “atende totalmente” e considerou a estrutura “completa”. Citou que o LEAN poderia ter sido contemplado no *framework*. Além disso, considerou a proposta como um referencial de apoio para a implementação do PCP com práticas ágeis e indicou que “Adoto atualmente na implementação do CMMI-DEV e MR-MPS-SW em uma organização de software”.

6. CONCLUSÕES

6.1. Consideração finais

Durante o desenvolvimento da dissertação notou-se a diversidade de métodos e práticas ágeis que podem ser utilizadas em um ambiente de desenvolvimento, mais especificamente para a implementação dos processos de PCP e de TS. O grande desafio foi definir o conjunto de práticas que deveriam ser utilizadas numa organização, desde a sua implantação, passando pela sua manutenção e evolução. A prática pode ser a mesma, mas a forma da implementação pode ser diferente de organização para organização, respeitando a cultura organizacional.

Apesar da especificação de cada prática apresentar uma simplicidade na sua implementação, não se pode assumir que sejam simples de serem aplicadas na práticas. Práticas de TDD, por exemplo, possuem a sua aplicação diretamente relacionada ao nível técnico da equipe de desenvolvimento. Percebeu-se, ainda, que não há um método ágil que atenda na totalidade todos os resultados dos processos de construção da solução. Para atingir ao propósito destes processos é necessária a integração de práticas de outros métodos.

Para esse entendimento, foi necessária a realização da revisão da literatura, estudo do estado da arte tanto dos resultados dos processos de construção da solução quanto das métodos ágeis, e o processo de revisão por pares, visando a avaliação dos relacionamentos entre os resultados dos processos de construção da solução e as práticas ágeis.

A estruturação e desenvolvimento do *framework* deu-se visando a aplicabilidade de cada prática ágil, destacando a forma de implementação e o grau de apoio da prática ágil ao resultado dos processos de construção da solução. Após a composição do *framework*, foi aplicado o processo de revisão por pares, visando a avaliação por um especialista e a posterior melhoria do trabalho.

6.2. Contribuições

A principal contribuição do trabalho é a apresentação de um conjunto de práticas, por meio de um *framework*, retratando como aplicar estas práticas para alcançar os objetivos dos resultados dos processos de construção da solução.

Outra contribuição do trabalho está relacionado ao fornecimento de um referencial teórico para organizações que estão implementando modelos de qualidade com métodos ágeis, especificamente o PCP do MPS.BR e o TS do CMMI. O detalhamento da forma de implementação de cada prática ágil apresenta para a organização, que faz o uso do *framework*, uma possível forma para a sua institucionalização.

No curso da pesquisa houve a publicação de três artigos científicos, a saber:

- o primeiro "MR-MPS-SW e Métodos Ágeis: Um Apoio à Implementação do Processo de Projeto e Construção do Produto" na Conferência "Computer on the beach" de 2014, cujo objetivo do artigo foi apresentar um apoio na forma de mapeamento e análise dos resultados esperados do processo de Projeto e Construção do Produto (PCP) do MR-MPS-SW com as práticas dos principais métodos ágeis com foco na Engenharia de Software (Crystal, FDD, TDD, Lean e eXtreme Programming);
- o segundo "Uma Análise da Literatura Especializada sobre a Aplicação de Métodos Ágeis no Contexto da Área de Processo de Solução Técnica constante em Modelos de Qualidade para Processo de Software" na Revista FSMA de 2015, com o objetivo de apresentar o resultado da revisão da literatura especializada no contexto da aplicação dos métodos ágeis para apoio à implementação dos modelos de qualidade CMMI e MPS.BR, especificamente para a área de processo de Solução Técnica e o processo de Projeto e Construção do Produto;
- o terceiro "Um *Framework* de Práticas Ágeis para Apoio à Implementação do Processo de Projeto e Construção do Produto do MR-MPS-SW" na Revista iSys de 2015, com o objetivo de apresentar o framework de práticas ágeis para apoio à implementação dos processos de construção da solução.

Por fim, no processo de revisão por pares o revisor indicou que “Adoto atualmente na implementação do CMMI-DEV e MR-MPS-SW em uma organização de software”, isso denota a qualidade do referido trabalho.

6.3. Limitações

Como limitação do trabalho pode-se destacar:

- não realização do mapeamento levando em consideração os processos que possuem relacionamentos com os resultados dos processos de construção da solução, tais como o desenvolvimento de requisitos, relacionado à concepção do produto, e os processos de Verificação e Validação, relacionados com o teste, homologação e revisão no produto;
- apesar da realização da revisão por pares do *framework*, um processo de experimentação não foi realizado, o que traria para o trabalho benefícios relacionados a melhorias no *framework*, pois a aplicação em um ambiente real de desenvolvimento valida a prática no uso *framework*;
- número de métodos no *framework*, pois no processo de revisão por pares houve o indicativo do LEAN para atendimento dos resultados dos processos de construção da solução. A expansão no número de métodos e por conseguinte as práticas ágeis daria maior amplitude para o *framework* no momento da utilização, pois haveria maior numero de possibilidades na aplicação do modelos de qualidades com o apoio de práticas ágeis.

6.4. Trabalhos Futuros

Como trabalhos futuros, pretende-se realizar a aplicação do *framework* no contexto das organizações que fazem uso de práticas ágeis para apoiar a implementação de modelos de qualidade. Essa aplicação traria para o trabalho pontos fortes, fracos e oportunidades de melhoria, pois na aplicação em ambiente de desenvolvimento haveria a possibilidade de verificar a eficiência na utilização das práticas.

Além disso, pretende-se realizar o estudo do LEAN no contexto de atendimento aos resultados esperados do MPS.BR, pois no processo de revisão por pares houve

indicativos de que práticas do LEAN poderiam atender resultados esperados dos processos de construção da solução.

Outro trabalho futuro refere-se à expansão do *framework* para os demais resultados esperados, tais como: Integração de Produto, Verificação e Validação. Essa expansão traria integração com as outras áreas e facilitaria a aplicação das práticas ágeis, pois essas práticas podem atender mais de um resultado esperado, conforme visto nesse trabalho. Além disso, na maioria das organizações esses processos relacionados à construção da solução são muita das vezes executados pela mesma equipe, o que facilitaria ainda mais o uso de práticas comuns às diversas áreas do processo da organização.

REFERÊNCIAS BIBLIOGRÁFICAS

BARTIÉ, A. **Garantia da qualidade de software: adquirindo maturidade organizacional**, 2002, Elsevier.

BECK, KEN. **Test Driven Development: By Example**. Pearson Education. 2003

BECK, KEN. **Programação extrema explicada: acolha as mudanças**. Bookman, 2004.

BORIA, JORGE LUIS; RUBINSTEIN, VIVIANA LEONOR; RUBINSTEIN, ANDRÉS OSCAR. **A História da Tahini-Tahini: Melhoria de Processos de Software com Métodos Ágeis e Modelo MPS**. Ministério da Ciência, Tecnologia e Inovação. 2013.

CASTRO V. S.; OLIVEIRA, S. R. B., VASCONCELOS, A. M. L. **Uma Análise da Literatura Especializada sobre a Aplicação de Métodos Ágeis no Contexto da Área de Processo de Solução Técnica constante em Modelos de Qualidade para Processo de Software**. Revista de Sistemas de Informação da FSMA, n 15, pp. 52-59. 2015.

CATUNDA, EDMAR; NASCIMENTO,CAMILA; CERDEIRAL, CRISTINA; SANTOS, GLEISON; NUNES, ELAINE; SCHOTS, NATÁLIA CHAVES LESSA; SCHOTS, MARCELO; ROCHA, ANA REGINA. **Implementação do Nível F do MR-MPS com Práticas Ágeis do Scrum em uma Fábrica de Software**. SBQS`2011.

COCKBURN, A. **"Crystal Clear - A Human-Powered Methodology For Small Teams, including The Seven Properties of Effective Software Projects"**. 2002

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION/ INTERNATIONAL ELECTROTECHNICAL COMISSION. ISO/IEC 15504-1: **Information Technology - Process Assessment – Part 1 - Concepts and Vocabulary**, Geneve: ISO, 2004.

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION/ INTERNATIONAL ELECTROTECHNICAL COMISSION. ISO/IEC 12207 Systems and software engineering– **Software life cycle processes**, Geneve: ISO, 2008.

KOSCIANSKI, ANDRÉ; SOARES, MICHEL DOS SANTOS. **Qualidade de Software**. Segunda edição. Novatec. 2006.

MANIFESTO ÁGIL. **Manifesto para o desenvolvimento ágil de software**. 2001. Disponível em <http://agilemanifesto.org/iso/ptbr/>. Acessado em 30 de janeiro de 2014.

OLIVEIRA, ANA CLÁUDIA; GUIMARÃES, FERNANDA ALVES; FONSECA, ISABELLA DE ARAÚJO. **Utilizando Metodologias Ágeis para atingir Certificação MPS.BR na Powerlogic**. SBQS`2008.

OLIVEIRA, S. R. B. et al. SPIDER – **Uma Proposta de Solução Sistêmica de um SUITE de Ferramentas de Software Livre de Apoio à Implementação do Modelo MPS.BR**. Revista do Programa Brasileiro da Qualidade e Produtividade em Software.

PBQP Software, SEPIN/MCT, 2011.

PAVAN, C., STUMPF, I. R. C. **Revistas Brasileiras de Ciência da Computação: procedimentos de avaliação pelos pares**. VIII ENANCIB – Encontro Nacional de Pesquisa em Ciência da Computação. 2007

PRESSMAN, ROGER. **Engenharia de Software**. Sétima Edição, MacGraw-Hill, 2011.

PRIKLADNICKI, RAFAEL; MAGALHÃES, ANA LIDDY. **Implantação de Modelos de Maturidade com Metodologias Ágeis: Um Relato de Experiências**. WAMPS`2010.

RAMASUBBU, N. BALAN, R. **"The Impact of Process Choice in High Maturity Environments: An Empirical Analysis"**. 2009

RETAMAL, A. **"Feature-Driven Development"**. 2008

SANTOS, ANTÔNIO F.; SILVA, VANDERMI J.; JUNIOR, VICENTE F; **Desenvolvimento de um Processo de Software Aderente à ISO 9001:2000 Baseado no Processo Ágil Scrum**. SBQS`2008.

SCHWABER, KEN; SUTHERLAND, JEFF. **The Scrum Guide**. Scrum.org, 2013.

SEI - Software Engineering Institute, CMMI-DEV, **CMMI for development**. version 1.3. 2010. Disponível em: <http://www.sei.cmu.edu/reports/10tr033.pdf>.

SEI - Software Engineering Institute, **"CMMI and Agile"**. Disponível em: <http://www.executivebrief.com/cmmi/agile-cmmi/>. Acessado em 02 de fevereiro 2014.

SILVA, E. L.; MENEZES, E. M. **Metodologia da Pesquisa e Elaboração da Dissertação**. 3. ed. rev. Atual – Laboratório de Ensino a Distância da UFSC, 2001.

SOMMERVILLE, IAN. **Engenharia de Software**. Nona edição. 2011.

SOFTEX "Avaliações MPS Publicadas". Disponível em: <http://www.softex.br/wp-content/uploads/2013/07/2Avaliacoes-MPSSW-Publicadas_12.maio_.2015_635.pdf>. 2015.

SOFTEX a. **Guia Geral MPS de Software**. 2012

SOFTEX b, **Guia de Implementação – Parte 11: Implementação e Avaliação do MR-MPS-SW:2012 em Conjunto com o CMMI-DEV v1.3**, 2012

SOFTEX. **Guia de Implementação – Parte 4: Fundamentação para Implementação do Nível D do MR-MPS**. 2013.