EXTREME PROGRAMMING: ANÁLISE SOBRE CONCEITO E BOAS PRÁTICAS DE DESENVOLVIMENTO

Rodolfo de Jesus Silva

Geiza Caruline Costa

RESUMO

Dentro do universo de desenvolvimento de software há programadores e equipes que buscam alcançar o produto ideal para atender uma necessidade específica. Existe meios, ou melhor, existem metodologias que contribuem para que essa necessidade seja alcançada. De um lado estão as metodologias clássicas e do outro lado as metodologias ágeis, essa que estão ganhando cada vez mais mercado devido suas incríveis soluções que diminuem extremamente as maiores complicações enfrentadas pelos programadores modernos. A metodologia XP por exemplo, vem ganhando destaque desde quando surgiu e sua capacidade de concluir projetos com a qualidade desejada é notória por toda comunidade de analistas e programadores de software.

Palavras-chave: programação extrema, XP, software, metodologia, ágil

ABSTRACT

Within the universe of software development there are programmers and teams that are able to achieve the ideal product to meet a specific need. There is a way, or better, there are methods that contribute to this need being met. On the one hand are the classic methodologies and on the other hand the agile methodologies, which are gaining more and more market share due to their solutions that greatly reduce the biggest complications faced by modern programs. The XP methodology, for example, has been gaining prominence since it appeared and its ability to complete projects with quality is notorious for the entire community of analysts and software developers.

Keywords: extreme programming, XP, software, methodology,agile

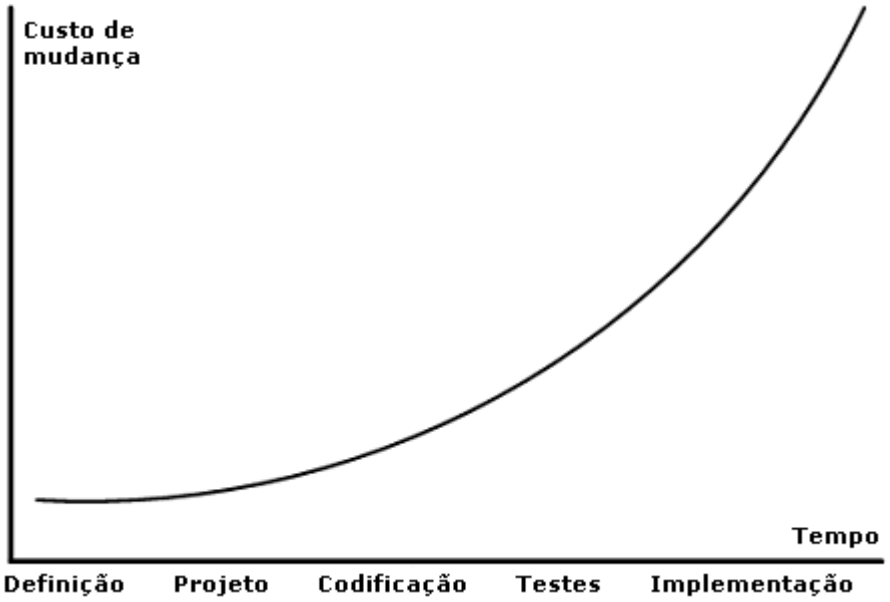
1. INTRODUÇÃO

> À medida que a importância do software cresceu, a comunidade de software tem continuamente tentado desenvolver tecnologias quer tornem mais fácil, mais rápido e menos dispendioso construir e manter programas de computadores de alta qualidade.
> (PRESSMAN, 2006, p.2)

O vasto mundo da programação de software vem se desenvolvendo a cada ano que se passa, e a necessidade de entregar um produto dentro do prazo, que atenda as necessidades dos clientes e que realmente funcione é imprescindível. Na década de 70, diante da imensa dificuldade de criação de produtos de software, muitas empresas desse ramo decidiram a utilizar metodologias para seus processos de desenvolvimento e criação de software, pois existia uma imensa dificuldade em concluir um projeto dentro do prazo estipulado, com os requisitos solicitados e com a qualidade desejada.

> Outra questão é que o cliente deve ser paciente, pois segundo o modelo clássico, ele só verá uma versão funcional no final do desenvolvimento. Desta forma, qualquer erro ou mal-entendido, se não for detectado durante o desenvolvimento do software, pode ser desastroso. (Sbrocco,2012, p.65)

O intuito dessas metodologias primordiais, chamadas “metodologias pesadas” era tornar os processos de desenvolvimento mais eficiente e disciplinada, porém nesse período era gerado uma vasta quantidade de documentos para acompanhar o produto de software, o que na maioria das vezes atrasava muito o projeto, principalmente em projetos que sofria quaisquer mudanças. Além do atraso, havia um excessivo gastos de recursos que superavam o orçamento, funcionalidades que não atendiam as necessidades dentre outros.



Fluiz(2010)

A imagem1 refere-se ao custo que ocorre quando surge a necessidade de realizar alterações no projeto enquanto se usa as metodologias pesadas. Observa-se que o custo é elevado de forma surpreendente.

Conforme os danos iam surgindo, Fluiz(2010) afirma que em meados dos anos 90 essas metodologias começaram a ser questionadas sobre sua real eficiência, o que resultou na elaboração das metodologia leves ou como são conhecidas metodologias ágeis de desenvolvimento. As metodologias leves prezam pela qualidade de software assim como as metodologias pesadas, porém seu foco é direcionado especialmente na elaboração de código e adaptação para mudanças de requisitos sem perda de recursos.

Segundo Pressman(2011, pag.87) O modelo XP(extreme programing) é a principal abordagem utilizada quando o assunto é metodologia ágil. Ela apresenta soluções robustas e eficientes para auxiliar a equipe de desenvolvimento a alcançar uma melhora significativa na entrega do software.

O XP é um processo de desenvolvimento que busca garantir que o cliente receba o máximo de valor de cada dia de trabalho da equipe de desenvolvimento. (TELES, Vinicius, 2014, p.24)

Diante dessa ideia, o conhecimento sobre alguns conceitos e práticas da XP se tornam praticamente essencial para todo profissional da área de tecnologia, além de saber como uma equipe consegue identificar quando e como fazer o uso do XP como a metodologia ideal para seus projetos.

1.1 OBJETIVOS E METODOLOGIA DA MONOGRAFIA

O objetivo desta monografia é apresentar as características essenciais da metodologia ágil de desenvolvimento e como ela pode nos auxiliar em nosso trabalho de desenvolvimento de software, tendo como foco principal o modelo Extreme Programing que é referência nesse área.

1.2 JUSTIFICATIVA

Segundo Pressman(2011, P.68) o mercado de software atual é bastante mutável, contendo prazos curtos para entrega sem deixar de lado a satisfação do cliente. A solução mais aceitável possível para atender essa demanda que o mercado exige foi a implantação de metodologias ágeis, e nesse meio o XP é que mais atrai atenção das organizações e equipes de desenvolvimento. Devido a esse fato muitos profissionais da área de programação e engenharia de software concordam que o conhecimento em XP é fundamental para sua carreira profissional.

Dessa forma acreditamos que esse artigo será de suma importância para a compreenção e entendimento deste tema.

2 METODOLOGIA ÁGIL DE DESENVOLVIMENTO

Metodologia ágil de desenvolvimento agrupam processos focados especificamente no desenvolver do projeto/produto em questão. Ela é caracterizada pela resposta a uma mudança e não seguir um plano com rapidez.

Essa ideia surgiu em 2001 através de 17 desenvolvedores e consultores da área de software. De acordo com Pressman(2011, p.81) a ideia era:

- Valorizar pessoas e interações antes dos processos e ferramentas
- Software funcional antes de uma vasta documentação
- Comunicação efetiva com o cliente ao invés de negociação contratual
- Manifestação à mudanças acima de seguir um plano

Dai então, nasceu o Manifesto para o Desenvolvimento Ágil de Software, o chamado "Manifesto Ágil" cujo site oficial se encontra em http://agilemanifesto.org/ (versão em inglês). De maneira nenhuma o manifesto ágil propõe largar a utilização de processos e ferramentas, negociação ou planejamento e muito menos deixar de seguir um plano. Porém a energia da equipe deveria ser usada principalmente em outros aspectos.

"Ou seja, embora haja valor nos itens à direita, valorizaremos os da esquerda mais ainda." (Pressman 2011, p.81)

De acordo com Sommerville (2013, p.40) os métodos ágeis fizeram com que a equipe de desenvolvimento focasse no software em si, e não em sua concepção e documentação. Ainda segundo Sommerville (2013, p.40) estes métodos propõem evitar sem burocracia os trabalhos que não geram valor ao cliente além evitar documentos que apesar de serem diligentemente elaborados, podem se quer ser usados.

Além da metodologia eXtreme Programing que tem o foco nesse trabalho, outras metodologias incluem a abordagem ágil. Vamos abordar rapidamente alguns deles antes de focar no nosso tema principal.

2.1 DESENVOLVIMENTO DE SOFTWARE ADAPTATIVO (ASD)

Segundo(Pressman,2011, p.103), O ASD trabalha com processos repetitivos iterativos de iterações, métodos de levantamento de requisitos exigentes e um novo ciclo de desenvolvimento iterativo que se une a grupos focados nos clientes e busca constantes em feedback.

2.2 Scrum

Segundo(Pressman,2011, p.103), O Scrum da prioridade ao conjunto de padrões prédfinidas de software, esses padrões mostraram-se ser eficazes para projetos com cronogramas de curto tempo, requisitos mutáveis e características críticas de negócio.

2.3 Método de desenvolvimento de sistemas dinâmicos(DSDM)

Segundo(Pressman,2011, p.103), O DSDM faz uso de cronograma em tempos definidos e também traz a ideia de que somente o trabalho suficiente seja requisitado para cada incremento de software afim de facilitar o próximo incremento.

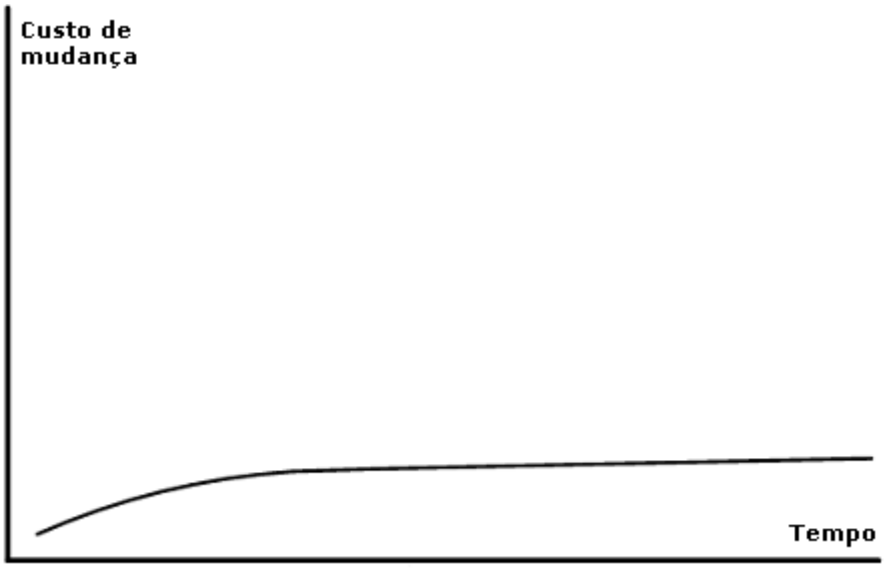
2.4 Crystal

Segundo(Sbrocco,2012, p.134), O Crystal é um conjunto de metodologias que possuem códigos relativos, isso possibilita o atendimento à diferentes tipos e tamanhos de projetos

2.5 Desenvolvimento dirigido a funcionalidades(FDD)

Segundo(Pressman,2011, p.103), Apesar do FDD ser um pouco mais formal, ele ainda se mostra bastante ágil fazendo com que a equipe foque nas funcionalidade do produto e que as funcionalidades podem ser rapidamente entregues.

Como menciona(Pressman,2011, p.94), Desses descritos acima e outras quem também compõem a proposta de desenvolvimento ágil, todos estão em comum acordo com o manifesto ágil e seguem sua proposta de desenvolvimento.



Fluiz(2010)

Ao analisarmos a imagem2, vemos que após adotada a metodologia ágil, o custo das mudanças permanecem extremamente baixos em comparação às metodologias pesadas, e se tornam quase que padrão após curto período.

3 EXTREME PROGRAMING

Segundo(Sbrocco,2012, p.145), O eXtreme Programming foi iniciado por Kent Beck em 1996 e assim como as demais metodologias citas acima, essa busca atender as necessidades dos clientes da forma mais simples possível sem deixar de perder a qualidade desejada.

Ela utiliza um modelo incremental ou seja, na medida que o software for utilizado, novas melhorias são implementadas. Isso é uma característica positiva para o cliente, pois ele sempre terá um produto ou parte dele para ser utilizado ou testado de acordo com suas expectativas. Caso o cliente decida mudar certo requisito ou funcionalidade em seu produto, a equipe de desenvolvimento não terá problemas em realizar essas alterações.

O modelo XP foca bastante nos fatores humanos durante todo o processo de desenvolvimento para os projetos em execução. Para o programador por exemplo, a XP preza pela sua saúde física e mental além de proteger a dignidade de cada um. Para o cliente por exemplo, o XP mantém o respeito por ele sempre transmitindo informações consistentes e de fácil entendimento.

Segundo (Sbrocoo, 2012) a equipe XP é composta por 5 papeis bem importantes para o bom resultado final do produto: Gerente de projeto, Coach, Desenvolvedor, Analista de testes, redator tecnico.

i. Gerente de projeto: Ele deve seguir a risca os valores e práticas da XP e certificar que todos da equipe façam o mesmo. Através dele o pessoal da programação poderá se comunicar com o cliente, e através dele será estipulado prazos e custos para o projeto.

ii. Coach: Segundo Scbrocco(2012) o Coach é quem tem o mais alto nivel em conhecimento de XP, assim ele atuará como um mentor para toda a equipe e estar sempre presente no decorrer do projeto. Duvidas, respostas e questão importantes sobre o XP são passadas por ele

iii. Desenvolvedor: Apesar de que no mercado atual a area de desenvolvimento pode ser divididas por várias funções, Sbrocco(2012) explica que no XP o desenvolvedor exerce diversos papeis como programador, analista, analista de banco de dados, projetista e até mesmo designer.

iv. Analista de testes: O analista de teste ira realizar os testes de cada interação junto com o cliente sem ter conhecimento do código feito pelo programador. Um dos papeis fundamentais dessa função é identificar as possíveis falhas antes mesmo que elas apareçam, e dessa forma forma, o produto chegue ao cliente em perfeitas condições

v. Redator técnico: Mesmo que o eXtreme Programming evite o excesso de documentação, existe o redator técnico que é responsável por realizar a documentação do sistema, isso contribui para que os programadores foquem ao máximo na codificação.

Nos sub-capítulos a seguir veremos a estrutura geral do XP e poderemos saber a resposta do porque o eXtreme Programming é a metodologia leve mais usada no mercado atual?

3.1 VALORES DO XP

Segundo(Sbrocco,2012, p.146), Caso uma equipe decida optar por XP como metodologia em seu projeto, essa equipe deve seguir a risca cada um dos valores existentes dessa metodologia, pois como menciona (Pressman, 2012,p.87), eles servem com o um guia das atividades, ações e tarefas especificas da XP, do contrario não será caracterizado com XP. São eles: comunicação, simplicidade, feedback, coragem e respeito. A seguir vamos abordar cada umas dessas virtudes que os integrantes da equipe de desenvolvimento devem ter.

3.1.1 COMUNICAÇÃO

De igual modo a toda matéria de estudo, a comunicação é de suma importância para o bom funcionamento do XP, através da boa comunicação(direta, formal e presencial) entre os clientes e desenvolvedores, os integrantes do projeto podem transmitir e receber conceitos importantes além de receber o feedback(próximo valor a ser abordado) continuo de cada interação. Isso contribui para uma das características principais do XP que é evitar a quantidade excessiva de documentos.

Segundo(Sbrocco,2012, p.146) a comunicação deve ser realizada afim de não haver mal-entendidos, especulações, dúvidas ou coisas semelhantes, caso haja, todas são sanadas continuamente usando a comunicação que ainda segundo Segundo(Sbrocco,2012, p.146) deve ser face a face. Assim as chances do projeto fracassar são mínimas.

Um dos fatores positivos sobre a comunicação em momento de desenvolvimento com o cliente é que segundo(Sbrocco,2012, p.146) o cliente passa a se sentir um item essencial para a conclusão do projeto além de estabelecer fatores de confiança entre os solicitante do projeto e os desenvolvedores.

3.1.2 FEEDBACK

A cada integração feita, é dado um retorno sobre as novas funcionalidades realizadas.

O cliente pode opinar sobre os novos recursos alegando tantos os fatores positivos como negativos.

Além do feedback do cliente, existe o feedback dado pela equipe de desenvolvimento, esse, segundo (Sbrocco, 2012,p.146) não acontece com a mesma frequência do feedback do cliente, mas mesmo diante desse fato, um não deixa de ser tão importante quando o outro.

Um ponto positivo do feedback é que o cliente pode se adaptar muito mais rapidamente ao uso do seu produto e evita fases de treinamento que segundo(Sbrocco, 2012,p.146), não é necessário usando XP

3.1.3 SIMPLICIDADE

O XP trata com seus desenvolvedores em construir uma funcionalidade que funcione e que possa ser facilmente implementado, e, se caso haja novas melhorias para serem realizados, o projeto poderá ser alterado posteriormente

"Essa simplicidade não significa facilidade e muito menos privar o cliente de recursos importantes; a ideia é atender o cliente com o necessário para que os requisitos do sistema possam ser completamente atendidos." (Pressman 2011, p.81)

Sendo assim, para entregar uma nova "release" a equipe foca em construir o simples e não focar seus esforços em elemento que podem ser realizadas/utilizadas futuramente. Esses elementos são chamados "perfumarias". Segundo(Sbrocco, 2012,p.146), a falta de simplicidade pode ocasionar falhas na comunicação e no feedback. Isso resultaria na perca total dos valores pregados pelo eXtremming Programming.

3.1.4 CORAGEM

Para os valores pregados pelo XP sejam seguidos corretamente, a equipe precisa ter coragem para emprega-los, pois como explica(Pressman, 2012,p.87), no decorrer do desenvolvimento do projeto, há sempre uma pressão para criação das "perfumarias", partindo das possíveis dificuldades que aparecem fazendo com que alguns integrantes queiram voltar às metodologias antigas e de que: "Se criarmos hoje, pouparemos tempo para amanhã".

Segundo(Pressman, 2012,p.87), a coragem poderia ser melhor definida como disciplina, para que a equipe de software entenda que quaisquer mudanças futuros podem resultar em um retrabalho árduo em relação ao projeto e ao código implementado.

Outro fator que exige a coragem da equipe é a tomada de decisões que em muitos casos são de auto-risco pois vão em sentido contrário ao que as metodologias pesadas pregam.

"Na verdade, trata-se de premissas que contrariam os moldes tradicionais de desenvolvimento. A ideia é que alguns itens devem sempre ser pensados ao longo do desenvolvimento, outros antes mesmo de começar a implementar". (Sbrocco, 2012, p.147)

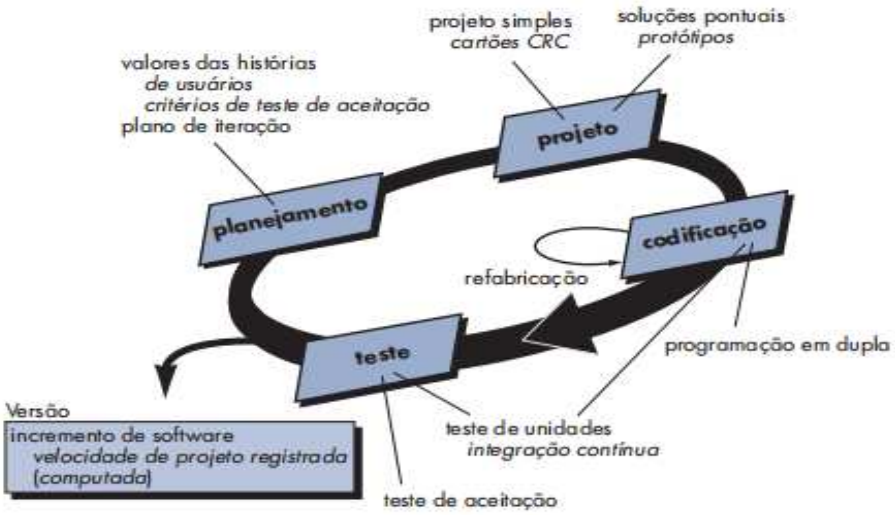
3.1.5 RESPEITO

O respeito trabalha simultaneamente com os demais valores do XP, trata-se de aceitar os feedbacks recebidos, efetivar a boa comunicação dando ouvidos aos que outros membros sugerem etc. Inclusive, conforme cada vez mais os valores são seguidos, cada vez mais haverá respeito pelos processo XP

3.2 PROCESSO XP

Como explica Bonato(2012) quando o objetivo é alcançar o máximo de benefícios oferecidos pela XP, a equipe deve empregar todas as práticas exigidas apesar de que se exercidas em pequenos conjuntos, é natural que aconteça resultados positivos.

O eXtreme Programming encapsula algumas regras e práticas dentro de 4 atividades padronizadas do modelo. Segundo Kent Beck, essas 4 atividades são denominadas: Planejamento, Projeto, Codificação e Teste.



Pressman(2012)

> "Você codifica porque se você não codificar você não terá nada. Você testa porque se você não testar você não saberá quando você terminou de codificar. Você ouve porque se você não ouvir você não saberá o que codificar ou o que testar. E você projeta para que você possa codificar, testar e ouvir indefinitivamente" (BECK, 2004).

3.2.1 Padrões de desenvolvimento

Quando formada, a equipe de desenvolvimento estabelece algumas praticas a serão padronizadas em todo o decorrer do projeto. Como por exemplo, como as variareis e funções serão declaradas, se serão usados CamelCase ou não, como o código será indentado dentre outros. Com isso, todos terão uma maior facilidade de entender o código feito por cada um, além de deixar a estrutura do código fonte mais profissional.

"Levando em conta que a comunicação é um dos valores da XP, essa prática possibilita a interação via código dos desenvolvedores". (Sbrocco, 2012, p.148)

3.2.2 Design Simples

Como já mencionada, essa prática visa aplicar soluções de códigos de forma mais simples possível para atender a solicitação do cliente em um primeiro momento, tendo em mente que os projetos podem mudar e essas mudanças podem ocasionar em um elevado custo de manutenção e um aumento considerado no prazo para entrega do produto. Sbrocco(2012)

explica que, caso o projeto for desenvolvido usando a programação orientado a objetos, essa questão do design simples pode ser muito bem aproveitado.

3.2.3 Cliente sempre presente

A metodologia XP foca muito nos ativos humanos e prioriza a participação do cliente junto a equipe de desenvolvimento durante todo tempo de construção do projeto. Essa prática facilita inclusive possíveis percepções de mudança do projeto e retorno imediato de feedbacks.

"Em um processo de Extreme Programming, o cliente é um membro da equipe de desenvolvimento". (Sommerville, 2013, p.148)

3.2.4 Jogo de planejamento

Reunindo o grupo de cliente e o grupo da equipe de desenvolvimento, são apresentadas as "historias de usuário", nelas são informadas as características, funcionalidades e resultados dos requisitos desejados pelos clientes. Junto delas, os clientes informam o valor ou a prioridade de cada funcionalidade. Através dessas informações, a equipe XP avaliam cada historia para estipular custos e prazos para entrega de cada release.

> "Para representar um dia de programação, a XP utiliza uma unidade específica, chamada de "ponto", que orienta o desenvolvedor a se dedicar plenamente ao desenvolvimento, não executam do outras tarefas que não objetivem atender a funcionalidade do projeto".(Sbrocco, 2012, p.148)

Como explica (Sommerville, 2013, p.88), se por ventura o prazo para entender a uma historia de usuário ultrapasse três semanas, o cliente deverá para dividir a sua solicitação em partes menores e então os cálculos serão realizados novamente.

3.2.5 Reuniões em pé

O objetivo disso é trocar experiência entre a equipe de desenvolvimento, tentar

encontrar soluções para as dificuldade encontrada em cada requisito, informar o que foi feito no dia anterior e informar o que será feito no decorrer do dia atual..

A reunião é feita pé pois deverá ser breve e objetiva, sem levantamento de duvidas ou hipóteses.

3.2.6 Programação em pares

Literalmente a programação em pares exige: Dois programadores trabalhando em um único computador. As duplas são formadas dinamicamente, não sendo necessário uma duplas definida para todo o decorrer do projeto. Essa forma de trabalho garante mais comprometimento no exercício das atividades uma vez que enquanto um profissional digita o código, o outro profissional está revisando afim de evitar erros e sugerir possíveis melhorias.

Essa prática ajuda no desenvolvimento dos programadores juniores quando formado dupla a programadores seniores ou programadores com dificuldades em um certo tipo de prática.

Apesar de parecer um desperdício de recursos humanos, Bonato(2002) traz algumas características que provam que a programação em duplas pode trazer certos benefícios tais como:

i. . A tomada de decisões para cada parte do projeto é realizada por dois cérebros.

ii. . Ao menos duas pessoas estão familiarizadas com cada parte do sistema.

iii. . É bem menos provável que as duas pessoas deixem de realizar alguma tarefa importante no momento da codificação.

iv. . A formação de duplas dinâmicas contribui para o compartilhamento de conhecimento entre a equipe de desenvolvimento.

v. . O código esta sendo constantemente sendo revisado por pelo menos duas pessoas

I. Refactoring

É uma ótima prática para correção e melhoria de códigos sem alterar a funcionalidade final do software, ela resulta diretamente em um código de qualidade e de fácil manutenção.

O objetivo não é caçar erros a todo momento pois a identificação de erros ocorre na prática de testes, mas sim realizar mudanças estruturais no código afim de otimiza-los eliminando códigos duplicados, excesso de variáveis ou funções desnecessárias, atualização de códigos legados dentre outros.

Segundo Bonato(2002) a refatoração de código deve ser realizada pela equipe XP a todo momento sem exceções.

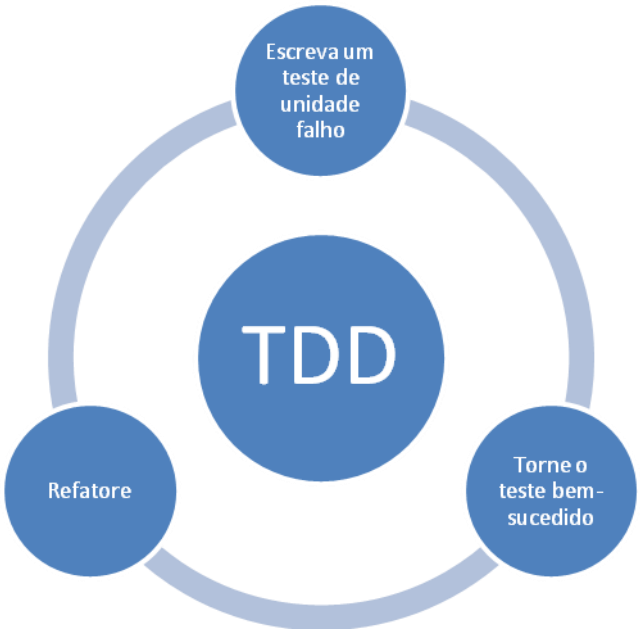
3.2.7 Desenvolvimento guiado por testes

Segundo Sbrocco(2012) toda e qualquer funcionalidade construida deverá ser acompanhada de um teste que poderá ser realizado usando teste de unidade, ou teste de aceitação. Basicamente os testes de unidade são feitos pelos programados no decorrer da elaboração do código enquanto os testes de aceitação são realizados pelos clientes para verificar se aquela funcionalidade atende sua necessidade atual.

O XP adota uma prática chamada desenvolvimento orientado a testes(TDD, em inglês Test Driven Development) e é conhecida por criar métodos de testes antes mesmo de criar qualquer funcionalidade.

"É importante ressaltar que o TDD não é um método para testar software, mas sim para construí-lo".(Sbrocco, 2012, p.236)

A imagem a seguir ilustra como é o clico de vida do TDD:



DevMedia(2012)

3.2.8 Código coletivo

Sbrocoo(2012) explica que no código coletivo, todos os programadores tenham acesso a todas as partes do código, podendo realizar correções ou modificar partes do código quando houver necessidade. Ou seja, independente do projeto o código nunca terá um dono e a responsabilidade por ele será de toda a equipe de forma coletiva.

Uma importante abordagem mencionada por Bonato(2002) é que apesar do código ser coletivo, a XP prega um ditado que diz: “Você estraga, você conserta”. Pois apesar do código ser coletivo ele não deixa de estar aos cuidados do programador.

3.2.9 Metáfora

A metáfora é um artificio para a boa comunicação entre a equipe de desenvolvimento e o cliente. A finalidade é usar temos simples do projeto para que o clientes possa entender claramente os processos, ou seja, usar analogias com os processos encontrados dentro da programação.

3.2.10 Ritmo sustentável

A XP prega que a equipe de desenvolvimento não deve trabalhar mais do que 40 horas semanais, Ponto!. Apesar das metodologias pesadas aceitarem um esforço maior dos programadores para trabalhar durante mais tempo, a XP entende que esse esforço pode ser um falso aumento de produtividade, pois Segundo (Scbrocco, 2012) após um tempo o cansaço físico e mental começam a aparecer e consequentemente irá gerar falta de concentração e falhas no andamento e na implementação do projeto.

Bonato(2002) afirma que, horas extras são um sintoma de problemas e nunca uma resposta a um problema.

3.2.11 Integração contínua

Afim de que software esteja sempre atualizado, a XP necessita que haja sempre uma integração no projeto. Claro que toda integração deverá antes ser seguida de testes e um backup para evitar possíveis danos.

Quando integrado continuamente, a causa de uma possível falha é mais clara e fácil de se encontrar, agora imaginemos o tempo e o trabalho de encontrar a casa da falha de uma integração que envolve diversas funcionalidade?.

“Grandes integrações criam uma grande carga de problemas todos de uma vez, e estes problemas podem ter centenas de causas possíveis.” (Bonato, 2002, p.6)

3.2.12 Release curtos

Trata-se de um parte específica e pequena do produto que deverá atender a uma necessidade do cliente e liberado na ordem definida pelo ele. Essa pequena “versão” do software será liberado para imediata utilização até que todo o projeto seja completamente concluído e satisfatório ao cliente.

4 CONSIDERAÇÕES FINAIS

Após a abordagem sobre o XP juntamente com suas práticas e valores podemos definir o porque o eXtreme Programming é a metodologia mais usada, em quais situações podemos

usá-las e em quais situações ele não se enquadra.

Em modo geral, recomenda-se usar o eXtreme Programming principalmente em projeto mutáveis, ou também em casos onde o projeto ainda causam incertezas tanto para o cliente como para a equipe XP. Mesmo assim, há algumas questões que se por algum motivo não seja possível praticar, o projeto pode falhar.

O XP não funcionará caso o cliente não esteja engajado a trabalhar junto com a equipe. O XP não atende projetos que requerem prazo de entrega, escopo e requisitos definido, projetos onde o cliente exige um orçamento logo de inicio e projetos que tenham grande risco financeiro também não serão bem atendidos. O XP não é recomendado para equipes extremamente grandes e nem para equipes que trabalham distanciados como o home office.

Com tantos contras para usa-la, o site oficial do eXtreme Programming explica seu enorme sucesso. Sem deixar a qualidade do software, o XP enfatiza a satisfação do cliente. Capacita os desenvolvedores a atender as mudanças com eficácia e eficiência. Todos os indivíduos são respeitados, tanto cliente, gerente ou desenvolvedores. Apesar do ambiente XP ser simples, é bem eficaz nos seus projetos, contribuindo para a alta produtividade das equipes.

5 REFERÊNCIAS

DeveMedia. **Introdução ao desenvolvimento guiado por teste(TDD) com JUnit**

Disponível em: https://www.devmedia.com.br/introducao-ao-desenvolvimento-guiado-por-teste-tdd-com-junit/26559

Fluiz. R. **Sem boas práticas de engenharia não há agilidade** Disponível em: https://ricardofluiz.wordpress.com/2010/10/18/sem-boas-praticas-de-engenharia-nao-ha-agilidade/

BECK, K. **Programação Extrema (XP) Explicada: Acolha as Mudanças**,

Bookman, 2004.

HIROKI, Kleber U. **METODOLOGIAS TRADICIONAIS E METODOLOGIAS ÁGEIS: ANALISE COMPARATIVA ENTRE RATIONAL UNIFIED PROCESS E EXTREME PROGRAMMING.** 2012

TELES, V. M. **Aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade.** São Paulo: Editora; Novatec,2014.

Henrique, C. **Carlos Henrique M. da Silva**, Disponível em: <https://s3.amazonaws.com/ppt-download/extremeprogrammingxp-141009125259-conversion-gate02.pdf?response-content-disposition=attachment&Signature=B9ksmnn9n2fuYDVP0EEN45tSWq4%3D&Expires=1587684048&AWSAccessKeyId=AKIAIA5TS2BVP74IAVEQ>. Acesso em: 23 abr. 2020.