RUBENS LOPES DA SILVA

ANÁLISE DE FATORES HUMANOS NOMÉTODO*EXTREME PROGRAMMING*

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

São Paulo
2014

RUBENS LOPES DA SILVA

ANÁLISE DE FATORES HUMANOS NO MÉTODO *EXTREME PROGRAMMING*

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

Área de Concentração: Tecnologia de Software

Orientador: Prof. Dr. Kechi Hirama

São Paulo
2014

DEDICATÓRIA

Dedico este trabalho primeiramente a Deus por seu amor e cuidado comigo. A minha noiva e minha família, pelo amor e apoio irrestrito.

AGRADECIMENTOS

Ao Professor e Doutor Kechi Hirama, braço amigo de todas as etapas deste trabalho. À minha noiva Bruna pela confiança e motivação. À minha família, pela força e incentivo no desenvolvimento deste trabalho e toda vibração positiva em relação a esta jornada. Aos professores e colegas de Curso, pois juntos trilhamos uma etapa importante de nossas vidas. Aos profissionais e Professores da Escola Politécnica da Universidade de São Paulo – EPUSP e ao PECE – Programa de Educação Continuada em Engenharia. A todos que, com boa intenção, colaboraram para a realização e finalização deste estudo. Aos que não impediram a finalização deste estudo.

RESUMO

A análise dos fatores humanos no XP é um agrupamento de heurística que busca aprimorar as práticas da metodologia com características comportamentais. Reforçam os princípios da metodologia ágil e tem como principal fonte de sucesso as pessoas que compõem a equipe de desenvolvimento, líderes que auxiliam e direcionam as equipes e os clientes que necessitam de soluções e melhorias proporcionadas por softwares. Deu-se devido à dificuldade e caso de insucesso das metodologias ágeis, causadas principalmente por barreiras humanas e as organizações que não reconhecem o fator humano como uma variável significativa nos processos. Dessa forma foram organizados características e comportamentos humanos necessários para a eficiência do desenvolvimento de software e adequados esses perfis nas práticas e agrupamento da metodologia XP. Além disso, tem-se observado que o direcionamento da valorização dos fatores humanos facilita a aplicação e a utilização de todas as práticas e valores sugeridos pelo XP. Com isso, o objetivo do trabalho é realizar uma análise dos fatores humanos que influenciam o resultado de um projeto de desenvolvimento de software. Estudos realizados sobre fatores humanos são considerados e aplicáveis ao método ágil *Extreme Programming* para então, propor uma análise desses fatores que geram impactos na produtividade e qualidade do software.

ABSTRACT

The analysis of human factors in XP is a heuristic grouping that seeks to improve the practice of methodology with behavioral characteristics. They reinforces the agile methodology principles, and its main successful source are people that make up the development team, leaders who assists and directs the teams and customers that need solutions and improvements provided by software. That came out due to the difficulty and failure of the agile methodologies, mainly caused by human barriers and organizations that do not recognize the human factor as a significant variable in the processes. In this way were organized human characteristics and behaviors necessary for the efficiency of software development and these profiles were adapted in practices and grouping of the XP methodology. However, it has been observed that the direction of the development of human factors facilitates the implementation and the use of all practices and values suggested by XP.Thus, the aim of this work is to analyze the human factors that influence the result of a software development project. Studies of human factors are considered and applicable to agile method Extreme Programming to then propose an analysis of factors that generate impacts on productivity and quality of software.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DSDM	*Dynamic Software Development Method*
FDD	*Feature Driven Development*
ASD	*Adaptive Software Development*
XP	*Extreme Programming*

SUMÁRIO

1. INTRODUÇÃO

Este capítulo apresenta as motivações desse estudo que consideram a importância de fatores humanos em processos de software, o objetivo, as justificativas e a estrutura do trabalho.

1.1. Motivações

Após estudo sobre o Manifesto Ágil, que surgiu em fevereiro de 2001, por um grupo de pensadores representante dos processos de *Extreme Programming*, SCRUM, DSDM, *Adaptive Software Development*, Crystal, entre outros que viam a necessidade de compartilhar ideias em comum. Esse mesmo grupo criou o manifesto com um conjunto de valores que promovem um modelo organizacional baseado em pessoas. (HIGHSMITH, 2001, p. 1).

Um dos princípios do Manifesto diz que a maior prioridade é a satisfação do cliente final, por empresas que fornecem softwares com entregas semanais e contínuas de forma interativa. O grupo criador do manifesto ágil discute diversos pontos sobre as pessoas envolvidas como clientes e desenvolvedores que devem trabalhar mais próximos, motivados, em um ambiente de apoio aos desenvolvedores, transmissão de informação, excelência técnica, simplicidade, equipe de desenvolvimento auto-organizada e tempo para reflexão sobre a eficiência e sintonia com os demais da equipe (HIGHSMITH, 2001, p. 1).

Porém, segundo Offner*et al*. (2011, p.1), cerca de 70% a 80% das iniciativas de mudança de outros processos para o processo ágil ocasionam falhas devido a barreiras a seus objetivos estratégicos. Dentre essas barreiras são citadas falta de colaboração e resistências dos funcionários.

Analisando pontos referentes somente à aplicação de métodos ágeis, Levy e Hazzan (2009) destacam que fatores humanos como, por exemplo, alta rotatividade

de pessoal, psicologia e sociologia entre os membros da equipe têm se caracterizado como o maior desafio nos projetos de software.

Peixoto e Silva (2009, p. 1) afirmam que o processo ágil tem como objetivo promover a adaptação evolucionária e iterativa das entregas, para adicionar valores e práticas que sustentam agilidade e resposta rápida a alterações. Porém, existem desafios como dificuldade de comunicação, disponibilidade do cliente, dúvidas que surgem durante a implementação de cada iteração e falta de reutilização de artefatos.

A análise dos processos a serem utilizados deverá levar em consideração questões de valores e cultura da organização pois Miller e Larson (2005, p. 1) afirmam que o uso de um processo ágil requer um julgamento subjetivo, ou seja, levar em considerações valores humanos acima de processos e ferramentas que não passam de artefatos inanimados.

Reis (2009, p. 6) discute sobre a qualidade de software, e o fator humano se destacou nos processo de software como uma das principais chaves de produtividade e sucesso de um processo. Mesmo após a evolução no desenvolvimento de software, existe a necessidade de um equilíbrio entre pessoas e processo. Portanto, a necessidade de construir um software com qualidade reflete a prática social que leva a um estudo e a compreensão do processo de aprendizagem e comunicação citado por Santos e Moura (2012, p. 1), o que comprova a deficiência de fatores humanos no processo de software.

1.2. Objetivo

O objetivo desse trabalho é realizar uma análise dos fatores humanos que influenciam o resultado de um projeto de desenvolvimento de software. Estudos realizados sobre fatores humanos são considerados e aplicáveis ao método ágil *Extreme Programming* para, então, propor uma análise desses fatores que geram impactos na produtividade e qualidade do software.

1.3. Justificativas

Estudos afirmam que os bons resultados e a eficiência na utilização das propostas por processos ágeis são apoiadas por indivíduos que interagem entre líderes, desenvolvedores e cliente(Santos e Moura 2012, p. 1). Existe uma linha de pesquisa sobre a interferência dos fatores humanos nos processos de desenvolvimento de software com destaque para o processo ágil, que consiste principalmente na interação e relacionamento interpessoal.

Assis (2010, p. 64) apresentou em um dos seus estudos, um conjunto de heurísticas do comportamento humano com o objetivo de aperfeiçoar um processo de engenharia de requisitos, no aspecto dos relacionamentos entre as partes interessadas.

O foco deste trabalho é o aprofundamento nessa linha de pesquisa, focando em um método ágil, o *Extreme Programming*. Assim como outros métodos, este sofre como principal barreira a cultura organizacional e outros tipos de resistências provocadas pelo fator humano como na análise de VersionOne (2013, p.10), destacada na Figura 1:

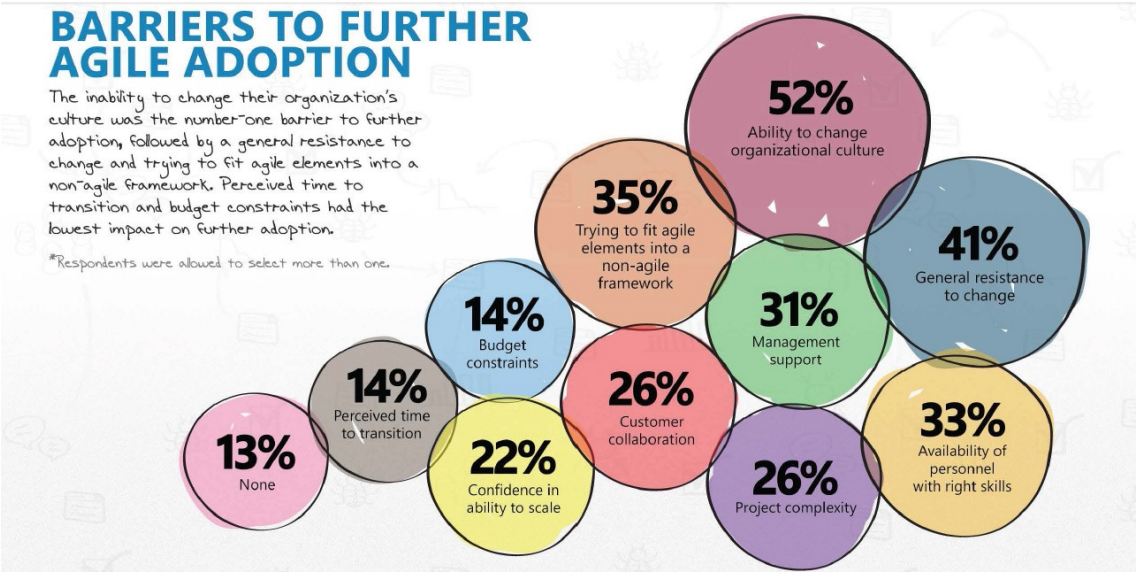

Figura 1. Barreiras da adoção do processo ágil (VersionOne 2013, p.10)

A análise desses fatores humanos permite identificar possíveis vulnerabilidades do processo ágil que podem comprometer o sucesso de sua aplicação em uma organização, ou que pode ser útil em outros processos de software (Esfahani 2010, p. 10).

1.4. Estrutura do Trabalho

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

O Capítulo 2, FATORES HUMANOS, apresenta a base teórica do trabalho com as principais referências e trabalhos acadêmicos sobre processos ágeis e estudos sobre fatores humanos.

O Capítulo 3, MÉTODO ÁGIL EXTREME PROGRAMMING, apresentada uma proposta dos fatores humanos e as subdivisões para o funcionamento eficiente do processo ágil com base nas referências bibliográficas apresentadas no capítulo 2.

O Capítulo 4, FATORES HUMANOS EM EXTREME PROGRAMMING, apresenta um estudo de caso realizado em uma empresa que utiliza o processo ágil. O estudo de caso consiste na análise do fator humano nas práticas do XP aplicado aos desenvolvedores, clientes e líderes que estão envolvidos no projeto de software.

O Capítulo 5, CONSIDERAÇÕES FINAIS, apresenta as contribuições do trabalho e analisa possíveis trabalhos futuros.

Em REFERÊNCIAS, apresenta uma lista de fontes pesquisadas e usadas neste trabalho.

2. FATORES HUMANOS

Os estudos dos fatores humanos começaram na disciplina de Engenharia de Software devido ao seu grande impacto em todo ciclo de vida dos sistemas. Conforme Brazier (1999, p. 1), as pessoas estão envolvidas em todas as fases de um ciclo de vida dos sistemas e são partes essenciais dos projetos, mesmo que seus erros causem falhas, a sua capacidade de diagnosticar problemas e ações adequadas podem levar à recuperação das falhas técnicas. Ou seja, os fatores humanos podem ter resultados positivos ou negativos independentemente da fase do projeto de sistemas.

Wang (2005, p. 2) descreve a importância da compreensão da natureza das necessidades humanas básicas porque é útil para previsão de motivações e identificação das forças motrizes. Wang (2005, p. 2) menciona que Sigmund Freud descreveu um ser humano motivado por estados de tensões internas como unidades que se acumulam até serem liberadas e, portanto foi sugerida uma lista de necessidades humanas que devem ser satisfeitas nível a nível:

- Necessidades fisiológicas: que são necessidades biológicas para a manutenção humana.
- Necessidades psicológicas: que são necessidades físicas e sociais de segurança, proteção e estabilidade.
- Necessidades cognitivas: que são necessidades de socialização.
- Necessidades sociais: que são necessidades de respeito, prestígio, reconhecimento, e auto-satisfação.
- Auto-realização: necessidade de crescer e cumprir seu potencial máximo em direção ao sucesso.

Dentro de uma equipe de projeto com o objetivo de produzir software, as pessoas assumem diversas funções, e essa diversidade é o que os torna componentes úteis no sistema, mas que introduz muitas chances para o fracasso. Por isso, Brazier (1999, p.2), considera que nem sempre é possível considerar os fatores humanos na primeira tentativa, e sim como parte da evolução ao longo do ciclo de vida do sistema.

Sampaio, Sampaio e Gray (2013, p. 1) ressaltam a importância das características humanas no processo de software, informando que os métodos não devem tratar as pessoas como atores ou simplesmente uma notação de modelagem, mas como seres conscientes e com todas as suas implicações na produção de software.

Segundo Wang (2005, p.3), a definição de fatores humanos é feita no sistema realizado por pessoas que apresentam fraquezas e incertezas, considerando-se que é uma restrição constante e importante em quase todas as disciplinas das ciências e engenharia, até que a maioria dos fatores ativos e dinâmicos seja considerada.

2.1. Definição de Fator Humano

A definição de fator humano no desenvolvimento de software é a reflexão sobre o desenvolvimento de software como uma prática social, passando além do quadro de engenharia técnica, que trata a compreensão de todo o processo em termos de aprendizagem, comunicação, foco nos indivíduos e as interações entre colaboradores e clientes (Santos e Moura. 2012, p.1).

Sampaio, Sampaio e Gray (2013, p.1) afirmam que a falta de reconhecimento dos fatores humanos é uma das principais razões pela quais o processo de software é tão difícil de ser melhorado. E argumentam que o esclarecimento de questões tão importantes com essa ajuda a conseguir um avanço significativo no processo de software leva a significativas melhorias de qualidade, satisfação e desempenho.

Para Wang (2005, p. 3) a capacidade de desenvolvimento de software é a chave entre todas as limitações cognitivas, organizacionais e de recursos em engenharia de software, pois para desenvolver sistemas, antes de qualquer artefato, o desenvolvedor deve criar, dentro de seu cérebro, um modelo abstrato.

2.2. Detalhamento dos Fatores Humanos

A seguir, os fatores humanos são detalhados levando-se em consideração sua a importância para a organização. Os fatores humanos considerados baseiam-se nos

trabalhos deSampaio, Sampaio e Gray (2013), Pressman (2011, p.86), Levy e Hazzan(2009)e que são: envolvimento, honestidade, competência, objetivo comum, colaboração e cooperação, confiança e respeito, habilidade de tomada de decisão e habilidade de solução de problemas e conflitos e auto-organização.

2.2.1. Envolvimento

O Envolvimento é um dos fatores mencionados por Sampaio, Sampaio e Gray (2013, p.1), que é o comportamento que resulta em empenho e criatividade humana que são motivadas por anseios e desejos pessoais, e por isso, a gerência precisa incorporar os objetivos e expectativas dessas pessoas, sejam clientes ou desenvolvedores, nas decisões organizacionais. Por ser um assunto complexo, Sampaio, Sampaio e Gray (2013, p.1) afirmam que a participação e o envolvimento correlacionam-se com o sucesso do sistema, mas o envolvimento correlaciona mais fortemente com o sucesso do sistema. Em uma avaliação, o envolvimento pode ser definido como um estado psicológico que reflete as relevâncias pessoais no processo avaliado.

Entender a singularidade das organizações implica na solução que deve ser sempre adaptada a organização em questão. Na área de gerência, a diferenciação é considerada fundamental para a excelência e competitividade. Porém, para utilização de modelos de padronização de processo cada indivíduo possui características únicas que podem ser compartilhadas com nenhuma, algumas ou todas as pessoas. (Sampaio, Sampaio e Gray, 2013, p. 1)

2.2.2. Honestidade

Honestidade é o estado de ser honesto sobre as situações relevantes, disponibilizar o conhecimento, experiências ou resultados, pois as pessoas que participam da organização têm suas próprias percepções sobre o que deve ser melhorado. Sem essas informações, o processo voltado de uma única visão em uma situação futura pode dar origem a comportamentos disfuncionais nas organizações (Sampaio, Sampaio e Gray, 2013, p.1).

2.2.3. Competência

Ozkaya, (2010, p. 3) afirma que a concorrência das organizações e respostas rápidas para novas inovações tecnológicas devido à maior capacidade de absorver conhecimento exige uma equipe com aptidões específicas e conhecimento assimilado do processo e, principalmente, com capacidade de inovar e habilidades pessoais. Essa inovação é resultante de uma função de princípios de organização, tais como: a especialização, diferenciação funcional, profissionalismo, gestão de mudança, conhecimento técnico, comunicação interna e externa, e habilidade administrativa e de gestão. Pressman (2011, p.86) afirma que a Competência condiz com as habilidades relacionadas ao software e com conhecimento do processo escolhido pela equipe, habilidades pessoais e conhecimento de processo que deve ser compartilhado com a equipe.

2.2.4. Objetivo em Comum

Outro fator que é necessário a ser analisado, segundo Levy e Hazzan (2009, p. 2), é a questão da exclusividade versus o compartilhamento. Levy e Hazzan (2009 , p. 2) afirmam que é um erro comum membros que entendem que a sua ascensão depende da perícia, pois esta postura de reter informações gera menor evolução profissional, em comparação com aqueles que compartilham seus conhecimentos. Wang e Chen, (2004, p. 4) afirmam que esse fator auxilia a equipe na revisão e escolha de qual conhecimento deverá ser mantido e compartilhado para projetos futuros, pois o conhecimento é composto de vários componentes que são produzidos durante as atividades, por papéis diferentes e em momentos diferentes.

Pressman (2011, p. 86) afirma que mesmo que cada membro realize tarefas diferentes e disponha de habilidades diferentes para os projetos, todos devem estar focados em um objetivo comum que é a entrega de um software funcional no prazo, e realizar adaptações contínuas contribui para o ajuste do processo às necessidades da equipe.

Segundo Draghici, (2013, p. 2), o conceito central do trabalho em equipe é um conjunto de pessoas que trabalham para um objetivo comum, compartilhando

recursos e principalmente tarefas. As equipes geralmente são associadas com o conjunto profissional e usuários que estão trabalhando juntos para cumprir um objetivo e que compartilham uma visão, que são definidos como uma organização.

2.2.5. Colaboração e Cooperação

Santos e Moura (2012, p.3) afirmam que os processos de trabalho em equipe são realizados através da coordenação e cooperação, envolvendo membros das equipes que compartilham do mesmo objetivo, e, além disso,pode-se garantir benefícios de equipes para a melhoria da reputação, das competências dos seus membros e produtividade organizacional.

Segundo Levy e Hazzan(2009. p.4) a distribuição de responsabilidades na forma de papéis ajuda a controlar e gerir estas responsabilidades. Além disso, a distribuição de papéis implica que todos os membros da equipe estejam envolvidos em todas as partes do software desenvolvido. Segundo Miller e Larson (2005, p.4) o termo Colaboração é mais focado sobre as pessoas envolvidas, enquanto que Negociação pode enfatizar a participação nas consequências. As pessoas se sentem privilegiadas em trabalhar com um grupo de pessoas que possuem um conjunto de valores compatíveis, um conjunto de valores com base na confiança e respeito um para o outro. (Miller e Larson, 2005, p.5).

Segundo Pressman (2011, p.86), colaborar é o mesmo que avaliar, analisar e utilizar informações que auxiliarão a todos da equipe a compreender o trabalho e a construir informações que gerem valor de negócio para o cliente.

2.2.6. Confiança e Respeito

Segundo Marnewick (2011, p.4), o estabelecimento de uma relação de confiança entre os usuários e a equipe é essencial. Sem confiança não há comunicação e sem comunicação não há transferência de conhecimento. A falta de confiança contribui para que a equipe não compreenda os problemas, o que leva a uma falta de domínio do que se deve desenvolver. O conhecimento de domínio contribui para a qualidade,

o que significa que, se houver falta de domínio de conhecimento, a qualidade do projeto de software será baixa.

As relações de comunicação e de confiança entre os usuários e a equipe de implementação são os principais fatores que contribuem para a qualidade dos requisitos. Porém, a comunicação é um fator humano e não pode ser resolvido por uma solução de engenharia. Excelente comunicação é uma habilidade que pode ser aprendida por qualquer pessoa. (Marnewick , 2011, p.4).

As relações de confiança se estendem internamente dentro da equipe, o que pode afetar sua interação e produtividade, Pressman (2011, p.86) considera que a confiança mútua e o respeito em uma equipe a torna consistente e fortemente unida.

2.2.7. Habilidade de Tomada de Decisão

Zhang (2010, p.1)considera a Habilidade de Tomada de Decisão como resultante de uma organização que desempenha o papel de descobrir, criar, comunicar e transferir o conhecimento. Primeiramente, é necessário realizar a gestão de conhecimento centralizada, o que envolve fases como: criação, retenção, processamento, comunicação, transmissão, partilha e utilização de conhecimento nas organizações.

Existem dois tipos de obtenção de conhecimentos, o explícito que se refere à pesquisa de conhecimento por linguagens, personagens ou símbolos que estão disponíveis para consulta e acessíveis em repositórios de dados, e o conhecimento implícito que se refere ao conhecimento obtido por experiências, análises e inspirações dos pesquisadores (Zhang, 2010, p.2).

O momento da descoberta de conhecimento refere-se à absorção de conhecimento relacionado à pesquisa científica das organizações através de levantamento e análise. Posteriormente é feita a integração, armazenamento e compartilhamento do conhecimento, para que posteriormente, ocorra a transformação do conhecimento através da descoberta e conclusões. Essas descobertas serão a base para as discussões e tomada de decisões.Para medir a qualidade de tomada de decisão é

feita uma média de resultados das decisões bem e mal sucedidas de acordo com os diferentes papéis (Zhang, 2010, p.2),

Zhang, (2010, p.2)descreve quatro tipos de papéis para tomada de decisão:

- Baseado na experiência de consultorias ou dos membros especialistas;
- Baseado na condução da visão dos líderes, que desempenham o papel dominante;
- Baseado em debates entre os membros, incluindo os líderes, quando suas opiniões têm o mesmo peso das opiniões dos demais membros;
- Baseado nas experiências dos membros mais antigos, que desempenham o papel dominante.

Segundo Osatuyi, Hiltz e Fjermestad (2012, p .1), cada membro da equipe contribui disponibilizando a informação ou parte dela, para que seja utilizada na tomada de decisão em uma discussão em grupo. A importância da informação pode afetar a forma como a informação é processada para a tomada de decisões durante a discussão da equipe, porém a distribuição da informação está disposta de tal modo que o grupo não tem todas as informações necessárias para tomar a melhor decisão.

Os tomadores de decisão devem dispor de capacidade de processamento das informações, que são utilizadas para analisar problemas de forma aprofundada e com o resultado tomar as melhores decisões, com o apoio dos membros da equipe que contribuíram significativamente com informações úteis, como por exemplo, em uma tarefa de *brainstorming*. Devem ser levados em conta fatores relacionados como características comportamentais e sociais que moldam as interações humanas com as informações discutidas, preferências e opiniões(Osatuyi, Hiltz e Fjermestad 2012, p.2).

Estudos de Osatuyi, Hiltz e Fjermestad (2012, p.6) também mostram que a discussão de informações repetidas pode sugerir a importância dessa informação, se é ou não é de fato importante, com base nestas descobertas, a hipótese de que as equipes vão focar a atenção em partes de informação que são consideradas mais

importantes, que são enumeradas em dois elementos: MIC (mais importante e compartilhado), e MINC (mais importante e não compartilhado).

Existe uma crença de que os indivíduos tendem a concordar quando todas as informações necessárias são fornecidas para tomar uma decisão, porém as informações disponíveis à equipe podem estar incompletas para a melhor tomada de uma decisão. De forma geral as informações trocadas entre os membros da equipe são selecionadas pela importância dessas informações. A análise preliminar dos dados pode realmente reduzir o tempo de discussão, fornecendo um mecanismo para que os tomadores de decisão avaliem a importância ou a relevância da discussão e pontos do processo de tomada de decisão. (Osatuyi, Hiltz e Fjermestad 2012, p.3).

2.2.8. Habilidade de Solução de Problemas e Conflitos

A Alta Gerência de uma Organização deve reconhecer que a equipe terá que enfrentar ambigüidades geradas por mudanças. E, deve reconhecer que as lições aprendidas pela solução de um problema podem ser benéficas. (Pressman, 2011, p.86)

Khan, *et al*.. (2011, p.5) afirmam que os conflitos geralmente surgem quando os membros da equipe entram em conflitos por causa de divergências provocadas por informações inconsistentes ou ambigüidades na comunicação ou registro dessas informações, por exemplo. Os conflitos precisam ter uma negociação para a sua resolução, que tem um impacto positivo de mudança no meio de comunicação. Em seus estudos eles afirmam que até a mudança de seqüência e abordagem do meio de comunicação é responsável por 93,1% da variação na resolução de conflitos.

Para In, Roy(2001, p.1) o cliente tem melhor percepção dos conflitos nos requisitos do produto que devem ser identificados e resolvidos em fase inicial do ciclo de vida do projeto para alcançar uma solução adequada aos projetos, que é um fator chave de um projeto bem-sucedido.

Uma abordagem sistemática para identificar conflitos chaves é analisar os perfis dos interessados. Um grupo colaborativo não tem a mesma experiência sobre as questões que estão sendo discutidas, portanto podem existir partes interessadas que possuem as mesmas posições, ou haver diferentes métodos de avaliações ou modelos, ou diferentes compreensões das questões e opções discutidas. Do ponto de vista da gerência de riscos, a discussão entre grupos chaves é importante e, em alguns casos, deve haver um consenso entre as partes interessadas (In, Roy 2001, p.2).

2.2.9. Auto-Organização

Segundo Hoda, Noble e Marshall (2013, p.2), as equipes auto-organizadas são compostas, por exemplo, por indivíduos que realizam a gestão da sua carga de trabalho, da jornada de trabalho do grupo e da tomada de decisão. Tendo como base equipes com foco comum, confiança mútua, respeito e capacidade de se organizar para enfrentar novos desafios.

A equipe se organiza para o trabalho, organiza o processo para seu ambiente, organiza o cronograma para cumprir a entrega, que ajudam a melhorar a colaboração e a moral da equipe, pois ela é auto-gerenciável.

O objetivo é que as equipes auto-organizadas escolham as tarefas a partir de uma lista de prioridades de necessidades dos clientes, de forma que as primeiras características do produto desenvolvidas são de mais elevada importância para o cliente. As equipes são definidas e que todos os membros sejam considerados de mesmo nível de responsabilidades, sem uma hierarquia explícita. Hoda, Noble e Marshall (2013, p.2) definem que, embora o sucesso de um processo dependa das pessoas, a capacidade para alcançar os objetivos depende do apoio recebido dos usuários, clientes e gestores.

2.3. Considerações do Capítulo

Nessa sessão foi apresentada a importância do estudo dos fatores humanos na disciplina de engenharia de software, e com as pessoas são componentes fundamentais que direciona os projetos para o sucesso ou fracasso. O fator humano no desenvolvimento de software é composto de práticas sociais que precisam considerar os processos de aprendizagem, comunicação e interação.

O conjunto de fatores humanos discutidos neste capítulo é considerado para aprimorar e sugerir ações que beneficiam as práticas do método *Extreme Programming*.

3. *EXTREME PROGRAMMING*

Criado por desenvolvedores no final dos anos 90 o método de desenvolvimento ágil surgiu como um estilo de produção de software colaborativo, iterativo e incremental.Esse processo foi escrito no "Manifesto Ágil" que contém valores como: interação das pessoas com processos e ferramentas de trabalho, documentação sobre software, colaboração do cliente negociado em contrato, respostas rápidas a mudança de escopo, entregas contínuas de trabalho, encorajamento de uma comunicação eficaz e equipes auto-organizadas. (Hoda, Noble e Marshall, 2013p. 2).

O processo ágil foi à resposta aos fracos resultados dos modelos tradicionais (custo maior - Figura 2) de desenvolvimento de software, e tem como diferencial poder acompanhar as mudanças por meio de um estilo iterativo e incremental de desenvolvimento que concentra um pequeno conjunto de funcionalidades priorizadas como importantes pelos clientes. Com custo menor conforme a Figura 3. (Hoda, Noble e Marshall, 2013, p.7).

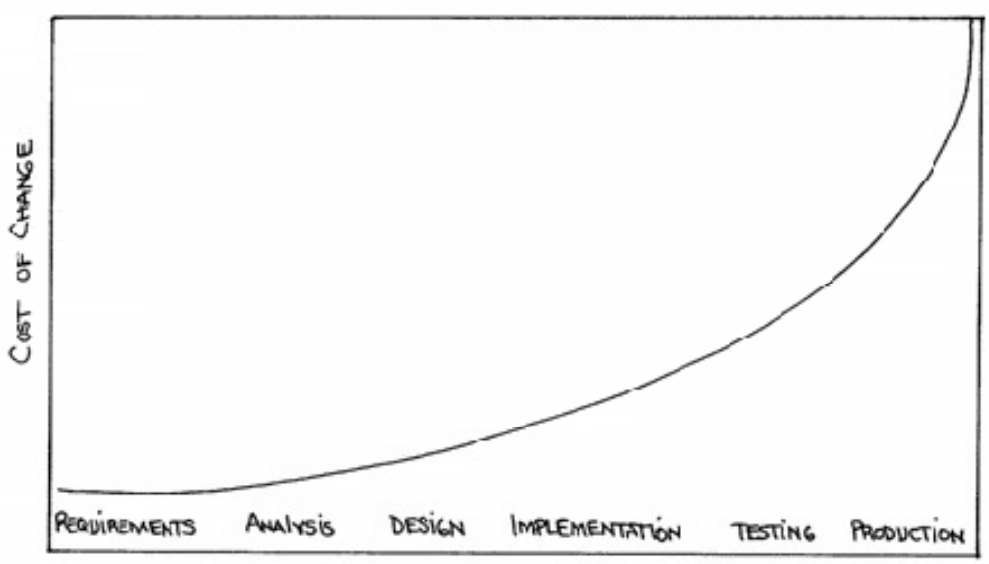

Figura 2. Custo da mudança na metodologia tradicional. (Beck, 1999, p. 24)

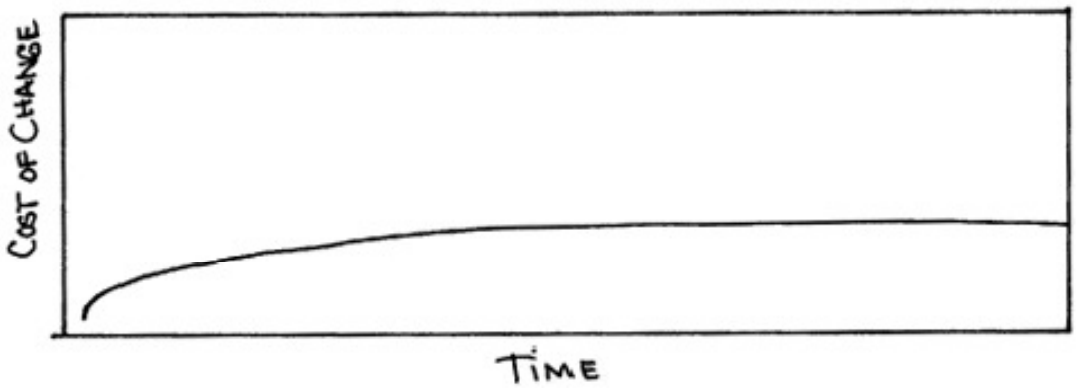

Figura 3. Custo da mudança na metodologia ágil. (Beck, 1999, p. 26)

O incentivo do envolvimento e o *feedback* do cliente é primordial para o processo ágil.O primeiro processo ágil *foi Dynamic Software DevelopmentMethod* (DSDM) e *Crystal* que consistem em uma série de métodos e princípios para personalizar projetos baseado em recursos, o *FeatureDrivenDevelopment* (FDD) que incide sobre o trabalho e o *Adaptive Software Development* (ASD) que se concentra em conceitos e na cultura organizacional (Hoda,Noble eMarshall,2013, p.1).

Os métodos mais amplamente utilizados são SCRUM e XP.O SCRUM foi desenvolvimento por Jeff Sutherland e formalizado por Ken Schuwaber. O processo foi derivado a partir do artigo de "*The New NewProductDevelopment Game*" de "*Harvard Business Review*" escrito por Takeuchi e Nonaka's em 1986.O SCRUM é caracterizado por ciclos curtos, tipicamente de duas a quatro semanas chamados de *sprints*, equipes auto-organizadas que escolhem as tarefas a partir de uma lista de prioridades e necessidades e que tenha maior importância para o cliente, e no final de cada ciclo uma parte do produto é entregue. (Hoda, Noble, Marshall,2013, p.2).

O *Extreme Programming* (XP) foi desenvolvido por Kent Beck e Ward Cunningham, com o apoio de Ron Jeffries e Martin Fowler.O XP é considerado um método leve para pequenas e médias equipes que trabalham com requisitos vagos ou que sofrem constantes mudanças (Hoda, Noble, Marshall, 2013, p.3).

O XP foi desenvolvido para enfrentar problemas do desenvolvimento clássico de software como defeitos de programação, projetos cancelados, incapacidade de solução de problemas de negócio. Esse método defende ciclos curtos e exige do

cliente selecionar a menor versão (quantidade de funções) que tenha o máximo de valor de negócio e requer participação do cliente na equipe para que forneça *feedbacks* rápidos. (Hoda, R. ;Noble, J. ; Marshall, S; 2013, p.3).

A diferença entre o SCRUM e o XP é que o XP se concentra em torno do desenvolvimento de atividades no nível de equipe e o SCRUM se concentra mais em gerenciamento de projetos (Hoda, Noble e Marchall, 2013, p.3) e por esse motivo, este trabalho foi direcionado para o XP.

3.1. Motivações do XP

Segundo Dyba, Tore e Dingsoyr, (2009, p.7), os benefícios do XP sãoa maior colaboração do cliente, os processos para o tratamento de defeitos, aprendizagem entre os desenvolvedores, melhora a visão do futuro para a gerência de projetos, concentração no trabalho atual para os engenheiros e estimativa de software.

A fim de oferecer uma solução adequada, reduzir os riscos e aumentar a previsibilidade dos sistemas, o XP veio e se tornou uma resposta aos problemas que ainda ocorrem quando se fornece soluções de software. O XP atraiu a atenção com sua negação feroz de muitas práticas bem aceitas da Engenharia de Software, e foi declarado como uma nova forma de desenvolvimento de software como uma metodologia leve, eficiente, de baixo risco, flexível, previsível e científica. Acima de tudo, oferece um horário flexível de implementação da funcionalidade do sistema que apóiam ativamente as necessidades do negócio mutáveis. Em suma, o XP promete reduzir os riscos do projeto e adaptação de negócios em constante mudança (Juric, Radmila, 2000, p.2).

Segundo Padberg, e Muller (2003, p.3), o XP considera suas técnicas mais benéficas se os requisitos são instáveis e o tempo de mercado é um fator decisivo. Por outro lado, se o tamanho da força de trabalho não permite executar o projeto com bastante pares(programação em pares), pode ser explorado um grau de paralelismo no projeto.Em média o par necessitaria entre 20% e 40% menos tempo para completar suas tarefas do que os desenvolvedores individuais. A programação

individual pode economizar no desenvolvimento, porém tem o dobro de custo durante a revisão de códigos.

Alguns estudos condenam a programação em pares como ineficiente, alegando que o XP funciona melhor com equipes de desenvolvimento experientes, e outros relataram outra limitação, que é a falta de atenção para projetar e questões arquitetônicas. (Dyba, Tore e Dingsoyr, 2009, p.7).

O XP prosperou em ambientes radicalmente diferentes, com pouco ou nenhum controle central, além disso, os pesquisadores têm analisado padronizações e mecanismos para criar uma consistência nas equipes e organização, como a boa relação interpessoal, habilidades e confiança que são características importantes para uma equipe de sucesso XP. (Dyba, Tore eDingsoyr, 2009, p.8).

Estudos sobre as percepções dos clientes relataram a satisfação do cliente como oportunidades para obter e dar *feedback*, no entanto, o papel do cliente no local pode ser estressante e insustentável por longos períodos. Empresas que usam XP relataram que os funcionários estão mais satisfeitos com seus empregos, com os softwares desenvolvidos e acreditam que este método melhora a produtividade da equipe, no entanto, eles informam que a programação em pares teve dificuldades devido às diferenças de habilidade entre os membros da equipe de projeto. Estudos concluíram que métodos ágeis podem incorporar as mudanças com mais facilidade e demonstrar o valor de negócio mais eficientemente do que projetos tradicionais (Dyba, Tore eDingsoyr, 2009, p.8).

O foco em humano e social é um fator necessário para ter sucesso, o alto nível de autonomia individual deve ser equilibrado, com um nível elevado de autonomia da equipe e responsabilidade corporativa. Também é importante para a equipe ágil, que as pessoas tenham fé em suas habilidades próprias, combinadas com boa habilidade interpessoal e confiança (Dyba, Tore eDingsoyr, 2009, p.8).

3.2. Princípios e Práticas do XP

O XP foi iniciado, segundo Wells (2009, p.1), em 6 de março de 1996, com os princípios de processo ágil e mostrou grande eficiência em empresa de diferentes tamanhos pelo mundo, e ganhou fama, pois seu sucesso se destacava pela satisfação do cliente. O XP é voltado para projetos que têm requisitos vagos e incertos, desenvolvimento orientado a objeto, equipes pequenas (desejável até 12 desenvolvedores) e desenvolvimento incremental. Segundo Teles (2004, p.41) a sua principal premissa é o aprendizado do cliente com o desenvolvimento, pois com o tempo os requisitos são reavaliados e é possível identificar suas prioridades e necessidades, dando maior valor de negócio ao software desenvolvido.

A comunicação da equipe de desenvolvimento com cliente garante menos distorções dos requisitos e maior agilidade, já que a informação é detalhada e clara. Outro ponto fundamental são os *feedbacks* que auxiliam no aprimoramento do projeto.

Sommerville (2010, p.264) descreve as seguintes etapas no processo, conforme apresentadas na Tabela 1.

Tabela 1. Práticas de Extreme Programming(Sommerville, 2010, p. 264).

Práticas	**Descrição**
Planejamento incremental	Os requisitos são registrados em cartões de estórias e as estórias a serem incluídas em um *release* são determinadas pelo tempo disponível e sua prioridade reativa. Os desenvolvedores dividem essas estórias em "tarefas".
Pequenos *releases*	O conjunto mínimo útil de funcionalidade que agrega valor ao negócio é desenvolvido primeiro. *Releases* do sistema são freqüentes e adicionam funcionalidade incremental ao primeiro release.
Projeto simples	É realizado um projeto suficiente para atender aos requisitos atuais e nada mais.
Desenvolvimento *test-first*	Um *framework* automatizado de teste unitário é usado para escrever os testes para uma nova parte da funcionalidade antes que esta seja implementada.
Refactoring	Espera-se que todos os desenvolvedores recriem o código continuamente tão logo os aprimoramentos do código forem encontrados. Isso torna o código simples e fácil de manter.
Programação em pares	Os desenvolvedores trabalham em pares, um verificando o trabalho do outro e fornecendo apoio para realizar sempre um bom trabalho.
Propriedade coletiva	Os pares de desenvolvedores trabalham em todas as áreas do sistema, de tal maneira que não se formem ilhas de conhecimento, com todos os desenvolvedores de posse de todo o código. Qualquer um pode mudar qualquer coisa.
Integração contínua	Tão logo o trabalho em uma tarefa seja concluído, este é integrado ao sistema como um todo. Depois de qualquer integração, todos os testes unitários do sistema devem ser realizados.
Ritmo sustentável	Grandes quantidades de horas extras não são consideradas aceitáveis, pois, no médio prazo há uma redução na quantidade do código e na produtividade.

Cliente *on-site*	Um representante do usuário final do sistema (cliente) deve estar disponível em tempo integral para apoiar a equipe de XP. No processo de XP, o cliente é um membro da equipe de desenvolvimento e é responsável por trazer os requisitos do sistema à equipe para implementação.

Segundo Juric (2000, p.2), o XP aborda os riscos do processo de desenvolvimento em todos os níveis do processo de desenvolvimento de software, que exige uma comunicação disciplinada dos programadores, gerentes e clientes. Um projeto com XP olha para problemas / riscos do processo de desenvolvimento e deriva soluções que ditam um conjunto de valores do XP:

- **Comunicações:**Problemas no projeto podem surgir se alguma informação importante ou mudança fundamental não é comunicada. O autor descreve diversas circunstâncias como medos, distrações, rejeições de informações relevantes que ocasiona a má comunicação .O Coach[1] tem o trabalho de perceber esses problemas e introduzir essas melhorias. (Beck,1999, p. 30).

  Simplicidade: O Coach solicita à equipe que sejam construídos artefatos de forma mais simples, mas o simples não significa fácil, e sim extrair o que é necessário para implementar no próximo prazo. O método XP afirma que é melhor apostar em algo simples no momento e implementar algo mais complexo depois, pois esse pode sofrer mudanças. (Beck, 1999, p. 31).

- ***Feedback*:** Existem *feedbacks* curtos de minutos e dias para programadores, testes e estórias feitas por clientes que são estimadas imediatamente ou podem ser realizadas em escala maiores de semana e meses para construção de testes para estórias, análise do estado atual do sistema, avaliar a velocidade de produção e ajustar os planos do projeto. Segundo Beck (ano, p. x),o *feedback* trabalha em conjunto com a comunicação e simplicidade, pois quanto mais se tem *feedback*, mais fácil de se comunicar, pois a

[1]Coach é o responsável técnico do projeto e assegura o bom funcionamento e melhorias da metodologia XP. (Teles, 2004, p.29).

comunicação se torna mais clara e simples contribuindo para a qualidade do desenvolvimento e testes da estórias. (Beck, 1999, p. 31).

- **Coragem:** Deve ser criada uma cultura que torna possível para programadores e clientes reconhecer seus medos e aceitar seus direitos e responsabilidades. Sem essas garantias, não se pode ter coragem. O XP incentiva a tomada de medidas ou ações drásticas e inesperadas, tais como jogar o código fora ou frear os testes que já foram executados e que possuem defeitos. (Beck, 1999, p. 32).

Os valores e as práticas do XP são a base para a construção de uma disciplina no desenvolvimento de software:

- Codificar: É a atividade básica XP que não é limitado ao ato de programar, mas de desenhar diagramas que geram código ou uso de ferramentas. Os códigos-fontes devem ser usados para tudo, para comunicar soluções, descrever algoritmos, testes, expressões,etc.Código é uma forma de comunicar de forma clara e concisa, pois uma ideia pode ser mal interpretada, mas quando codificada é possível ver na lógica precisão dessas ideias.(Beck, 1999, p.40).

- Testar: Os testes permitem pensar sobre o que deve ser feito independente de como será implementado, e quanto mais realizados os testes, a confiança no sistema aumenta com o tempo. Nesse processo deve se encontrar o nível de defeitos que estão dispostos a tolerar. São escritos testes funcionais que os clientes acreditem que o sistema, como um todo, funcione de maneira esperada.(Beck, 1999, p.41).

- Ouvir: Os programadores contribuem com o cliente quando entendem a complexidade do negócio e por isso a importância do ouvir. O feedback que é repassado fornece ajuda ao cliente a entender melhor os problemas, e dessa forma deve desenvolver regras que incentivam a comunicação estruturada e desencorajar a comunicação desnecessária. (Beck,1999, p.43).

- Projetar: faz parte do trabalho diário de todos os programadores em XP no meio de sua codificação e testes. Ele baseia-se no conceito de que a mudança de uma parte do sistema nem sempre requer uma mudança na outra parte. Em boa concepção cada pedaço de lógica no sistema tem só um lugar, o pedaço é colocado onde permite a extensão do sistema, com alterações em um único lugar. (Beck, 1999, p.43).

O principal aspecto do XP é estruturar as atividades e práticas brevemente listadas abaixo:

A. O Jogo do Planejamento

O desenvolvimento normalmente inicia com um plano mais superficial e é aperfeiçoado no decorrer do projeto, exigindo tempo e consultas aos clientes. As estimativas são fornecidas pelos programadores que avaliam as estórias do cliente e através dessas estórias é possível planejar os prazos e lançamentos das correções de erros. Durante todo o processo é feita a análise de possíveis alterações e oportunidades de melhorias nos sistemas junto com o Cliente. (Beck, 1999, p.54).

B. Lançamentos curtos

A vantagem de usar lançamentos curtos ou entrega com prazos menores, é pela possibilidade de trabalhar nas estórias mais valiosas e que seja funcional, e os testes para redução dos defeitos não necessitam de prazos longos. Assim é possível liberar versões com mínimo de funcionalidades que proporciona uma visão e a forma de como será conduzido o software para o cliente.(Beck, 1999, p.55).

C. Metáfora

As metáforas ajudam a orientar o desenvolvimento dos programas. A princípio pode ser uma metáfora global e seguindo para outras mais detalhadas, e o cliente expressa o comportamento desejado dos sistemas recebendo

rapidamente o *feedback* dos desenvolvedores, aperfeiçoando a compreensão do sistema e acompanhando a aplicação dessa estórias em código e em testes. (Beck, 1999, p.55).

D. Projeto Simples

O objetivo é realizar somente o necessário para aquele momento, pois projetos de desenvolvimento podem sofrer um trabalho desnecessário ou riscos pelo projeto ficar incapaz de evoluir. Para evitar esse tipo de perda os desenvolvedores necessitam de uma visão global e clara ou apoio da equipe e clientes para realizar o trabalho de forma mais otimizada. (Beck, 1999, p.55).

E. Teste

A simplicidade do design facilita os testes, pois as estórias direcionam para um teste adequado escritos pelos desenvolvedores e clientes, dando mais tranquilidade e conforto na execução dos testes. Beck afirma que o sucesso do XP depende de processos eficientes de testes.(Beck, 1999, p.56).

F. Refatoração

Refatorar o sistema é uma oportunidade de tornar o sistema mais simples, reduzir as duplicações de funcionalidades que realizam ações ou operações parecidas. Uma forma de comunicar com mais clareza e melhorar a administração é dividir o design em menores partes.Contribui no controle e no tempo, pois realizar todo o design de uma única vez custa mais esforços e tempo da equipe. (Beck, 1999, p.55).

G. Programação em Pares

Realizar a programação em pares pode ser um desafio para os programadores, pois várias práticas são oferecidas que permitem uma transição suave para programação em pares, incluindo a partilha de todos os

artefatos de programação, tais como design, codificação, revisão, foco nas tarefas, e *feedback* para melhora das habilidades pessoais. (Hoda, R. ;Noble, J. ; Marshall, S. ; 2013; p.4).

Apesar do custo pessoal, a programação em pares visa aumentar a produtividade da equipe e a qualidade do código, em comparação ao desenvolvimento convencional, o modelo baseia-se no padrão de valor presente líquido. Se a pressão do mercado é forte, o tempo é essencial e a programação em pares pode equilibrar o aumento do custo de profissionais, pois os pares de programadores são muito mais rápidos do que os desenvolvedores individuais.

Na programação em pares, todas as tarefas devem ser realizadas por pares que usam somente um monitor, teclado e mouse, com o objetivo de melhoria da qualidade do software, permitir que os desenvolvedores compartilhem suas ideias imediatamente para chegar a soluções rápidas e também ajuda a eliminar antecipadamente os defeitos. Pois, o par de programadores leva a uma revisão em curso do código pelo segundo desenvolvedor, ou seja, o modelo concentra-se no custo de desenvolvimento, e não no custo da operação. (Padberg, F. ; Muller, M.M.; 2003, p. 2).

A programação em pares atua na melhoria da qualidade do projeto, redução de defeitos, reduzindo riscos, melhorando habilidades técnicas, e melhorando a comunicação da equipe, porém o sucesso depende das características da personalidade dos pares, que deverão ser escolhidos com base nos traços de personalidades,Deve ser observada a comunicação, de forma eficaz, o conforto de trabalhar com outros, a autoconfiança e compromisso que são as quatro características essenciais para a programação em pares.(Chao, J. ;Atli, G.; 2006, p.1) .

H. Propriedade Coletiva

O código é uma propriedade da equipe que desenvolve, e todos devem ter autonomia e conhecimento, mesmo que não detalhado das funcionalidades

do sistema, pois todos têm a chance de melhorar o código e evoluir o sistema.Devido à prática de programação em par, teste e interações curtas a troca de informações ocorre com menos dificuldade e assim a equipe consegue definir um padrão de codificação, documentação e controle. (Beck, 1999, p.57).

I. Integração Contínua

Durante a integração ou implantação de ciclo podem surgir conflitos com outras partes do sistema. As novas entregas não podem interferir ou prejudicar entregas anteriores. O conjunto de testes contínuos ajuda a minimizar esses efeitos. O conhecimento disseminado entre a equipe e a prática de refatoração reduz o tempo de integração.Essa integração poderá ser feita em pouco tempo sem comprometer o sistema como um todo e poupando energia da equipe e o tempo do projeto.(Beck, 1999, p.58).

J. Semana de 40 Horas

A valorização da equipe requer uma preocupação com o bem estar e garante que a equipe tenha energia para todas as tarefas sem prejudicar sua capacidade produtiva. A realização do planejamento e a programação rápida contribuem para que as 40 horas semanais sejam respeitas sem nenhuma surpresa.(Beck, 1999, p.58).

K. Cliente Presente no Local

O cliente deve estar disponível para interações e validações, e tem o papel de contribuir na construção de cenários de testes, tomando decisões e realizando esclarecimentos durante a programação das estórias, porém respeitando o escopo e o planejamento do sistema.(Beck, 1999, p.58).

L. Padrões de Codificação

> Os programadores possuem estilos e visões diferentes sobre o mesmo problema.Caso a programação seja feita sem nenhum estilo ou sem padrões de codificação gera um atrito desnecessário na programação em pares e na refatoração. Nenhuma prática funciona bem se for feita por conta própria, por isso deve haver um padrão comum em que todos fazem parte de uma equipe vencedora.(Beck, 1999, p.59).

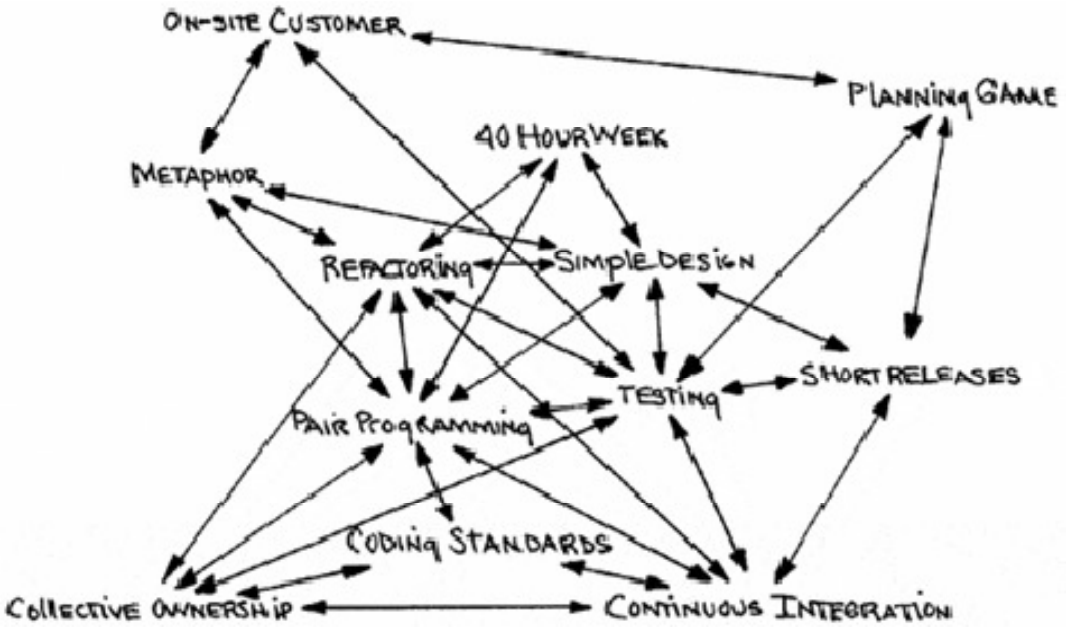

Figura 4. Diagrama que sumarizam as práticas.(Beck, 1999, p.59)

No XP todas as atividades são centradas em torno de programação, ou seja, tudo que se faz no XP parece com a programação. Começa com planejamento de iteração (a partir da estratégia de Planejamento) e incluem práticas como integração contínua de código escrito, propriedade coletiva de código e programação em pares, que ligam o processo de desenvolvimento em conjunto. Usa princípios com o pequeno investimento inicial, simplicidade, mudanças incrementais removendo todas as funções desnecessárias. Conforme as atividades de programação abaixo. (Juric, 2000, p.4):

- Exploração: preparara produção, estórias de usuários, estimativas e adquirir experiência com a tecnologia e programação de tarefas.

- Planejamento: executar o planejamento, acordar em conjunto menores as estórias para serem concluídas.

- Interação: produzir um conjunto de casos de teste funcionais que devem ser executados no final de iterações.

- Produção: realizar iterações de uma semana, e certificar-se que o software está pronto para a produção; implementar um novo teste e ajustar o sistema.

- Manutenção: produzir simultaneamente novas funcionalidades e continuar mantendo os sistemas existentes, refatorar se necessário ou migrar para uma nova tecnologia e experimentar novas ideias arquitetônicas.

3.3. Responsabilidades e Papéis no XP

O XP trabalha com iterações curtas ao longo de todo o projeto, e a cada fim de ciclo são realizadas as entregas de funções do software. A equipe deve se organizar em suas ações diárias, pois existe pouca margem de manobra. Na primeira interação será necessário se preocupar com três problemas: disponibilizar uma boa infra-estrutura de trabalho, fazer com que a equipe e o cliente tenham boas expectativas pelo processo, pois pode existir uma descrença que o processo irá funcionar. E por último, o desafio de estimar os primeiro pontos de função, sem estimativa anterior. Portanto, esse primeiro ciclo é o mais importante, e definirá o sucesso ou o fracasso do XP, pois o nível de confiança do cliente e da equipe será utilizado como base para as próximas interações (Teles, 2004, p.73).

Detalhando a ideia de uma boa infra-estrutura, a empresa deverá determinar quais ferramentas necessárias, e identificar quais os benefícios, custos e tempo que a equipe necessita para se familiarizar. Para começar o projeto, deve assumir a simplicidade e o mínimo de ferramentas necessárias até atingir uma quantidade equilibrada que não comprometa a agilidade da equipe. O gerente deve se atentar

as essas questões que definirá o meio de desenvolvimento sem cometer exageros.(Teles, 2004, p.157).

No início do projeto pode haver uma preocupação na equipe, devido ao fato de os profissionais terem uma formação voltada ao planejamento e controle, totalmente diferente do que é pregado no XP, portanto a equipe poderá apresentar uma resistência ou desconhecimento do processo. O gerente deverá ter em mente que as visões das pessoas só mudam por meio de confronto de ideias e argumentos. Portanto, será necessário conversar diretamente e negociar com a equipe a passar por essa experiência. (Teles, 2004, p.248).

No XP os requisitos que são altamente mutáveis devem ser divididos em casos de teste mais simples, o trabalho é centrado em testes de unidades simples que são escritos antes do código de produção e executado cada vez que as novas alterações do código são produzidas (Juric, Radmila, 2000, p.4).

O código não pode ser duplicado e caso o mesmo código seja encontrado em dois locais diferentes, eles devem ser combinados.O conceito definido como refatoração irá garantir que cada método não tentará fazer mais do que deveria. No XP o código final encontrado na produção é tão preciso, reformulado e legível que poderia ser comparado como um modelo de arquitetura de software. No entanto, é mais provável que o código XP com pequenos métodos usando mensagens legíveis e nomes variados, poderia sim, contribuir para a documentação do projeto. (Juric, Radmila, 2000, p.4).

A equipe também tem o desafio de manter a simplicidade no seu trabalho de desenvolvimento. Mudança de hábitos e um trabalho contínuo e exaustivo.Segundo Sommerville (2010, p.262), a equipe se concentra na simplicidade do software que está sendo desenvolvido e no processo de desenvolvimento, sempre que possível, trabalha ativamente para eliminar a complexidade do sistema. Mudar o jeito de desenvolver projetos de forma simples e limpa contribui para o processo evolutivo e mudanças constantes de requisitos. Outro ponto na aplicabilidade no projeto utilizando o XP é a qualidade do software produzido, garantida por meio de testes

eficazes, baseados nos conhecimentos tanto técnico, como de negócio, e a utilização de ferramentas simples e eficazes.

Conforme Teles (2004, p.73) para a realização da estimativa será necessário definir um ponto. Conforme descrito anteriormente, cada estória determinará uma quantidade de cartão, ou grupo de funcionalidade que será entregue por interação delimitada por certo período como, por exemplo, em duas semanas. O gerente deve dividir cada cartão em pequenas funcionalidades, pois quanto maior a funcionalidade, maior será sua complexidade e imprecisão da estimativa. O autor define que um ponto é a produção diária de um par, e que cada cartão poderá consumir de um a três pontos, e a somatória desses pontos por interação é definida como velocidade. A velocidade de produção deve ser negociada com o cliente para não prejudicar a relação dele com a equipe, pois além da produção diária, a equipe tem outras atividades que podem tornar esse valor fora da realidade. Devem ser consideradas horas para reuniões, ajustes, manutenções, treinamentos etc.

Em questão do cliente, Teles (2004, p.248) descreve que mesmo bem intencionado, o cliente questiona muitas das práticas, e sugere uma ou outra técnica que pode ser alterada. Negociar com o cliente é fundamental para que sejam feitas experiências do processo de forma mais categórica possível. Para que finalmente sejam quebrados os paradigmas de um processo de desenvolvimento de software.

Sommerville (2010, p.262) ressalta a importância dessa relação com o cliente onde clientes devem ser profundamente envolvidos no processo de desenvolvimento. Seu papel é fornecer e priorizar novos requisitos do sistema e avaliar as interações do sistema.

Teles (2004, p.22) é enfático na necessidade da presença do cliente em todas as fases do projeto. O sucesso do XP está vinculado ao trabalho em conjunto com o cliente, por isso o gestor deve ter em mente que manter seu cliente informado e sempre interessado não será uma tarefa simples.

A comunicação em qualquer nível e fases do projeto XP parece ser a espinha dorsal de todas as práticas de XP, a obrigação de programar em pares, a fim de aproximar

unidade de testes e código de produção de forma mais dinâmica e compartilhar experiências e conhecimentos de outra pessoa. O código é tratado como propriedade coletiva e todos têm a responsabilidade de todo o sistema, que tem autonomia de alterar qualquer código em qualquer lugar. No entanto, as práticas de XP fazem exatamente o contrário: eles incentivam os programadores a envolver-se em qualquer fase do processo de desenvolvimento de software e de capturar os requisitos para a produção de software e quebra o isolamento dos programadores, que é a única maneira de obter soluções de software aceitáveis. (Juric, Radmila, 2000, p.6).

O XP proporciona a gestão eficaz das expectativas dos clientes motivando a equipe a gerir a sua própria carga de trabalho, jornada de trabalho com base na necessidade e melhor ajuste, e participar na tomada de decisão da equipe. A auto-organização das equipes deve ter foco comum, confiança mútua, respeito e capacidade de organizar repetidamente para enfrentar novos desafios. (Hoda,NobleeMarshall,2013, p.5)

SegundoGray, (et al. 2006, p.1), estudos empíricos mostraram que equipes compostas por pessoas com entendimentos e comportamentos semelhantes tendem a ter um melhor resultado e excedem o potencial de desempenho. Processos ágeis são informais e mais dependentes de pessoas e suas interações que os processos formais que se concentram mais em regras e estruturas. O XP prova ser mais útil em ambientes dinâmicos, onde os participantes envolvidos no projeto podem mudar com mínimo de efeito para o projeto. No entanto, esta força também pode ser um ponto fraco.Devido à sua dependência de pessoas, o processo XP pode não conseguir repetir resultados e ser visto como não confiável.

Segundo Hoda, Noble e Marshall (2013, p.10), existem seis papéis importantes no processo Ágil:

1. Mentor, que orienta e apoia a equipe inicialmente, ajuda a tornar confiante o uso e aderência a metodologia, e o mentor representado pelo Agile Coach ou Técnico ágil (conhecido como Coach no XP ou Scrum Master no Scrum), tem o

papel de realizar conversas informais com a equipe ou treinamento formal por um curto período e interagem principalmente com a Gerência Sênior e a equipe.

2. Coordenador, que atua como um representante da equipe para o cliente, atender as expectativas e coordenar a colaboração do cliente com a equipe. Esse papel surgir para facilitar a cooperação do cliente e expandir a atuação do cliente nas atividades, discutir características do produto, definir prioridades e receber *feedback* de forma rápida, o coordenador é representado pelo Técnico ágil, Analista de Negócio e Desenvolvedores que interagem com a equipe e clientes.

3. Tradutor, que entende e traduz entre a linguagem de negócios utilizada por clientes para as terminologias técnicas utilizadas pela equipe, melhorando a comunicação entre os dois, esse tradutor é representado pelo analista de negócios e interagem com a equipe e clientes.

4. Defensor, que defende o método ágil da gerência sênior, tem o papel de obter apoio da auto-organização da equipe e garantir a adoção bem sucedida do método e gerar interesse da gerência sênior em relação ao custo, eficácia e melhoria de processos, o defensor é representado pelo Técnico ágil e interagem exclusivamente com a gerência sênior.

5. Promotor, que promove o método ágil com os clientes e tentativas de garantir o seu envolvimento e colaboração para apoiar o funcionamento eficiente da organização da equipe ágil. Esse profissional tem o papel de diminuir a distancia entre o cliente e a equipe e quebrar a falta de compromisso de tempo por parte dos clientes. O promotor é representado pelo Técnico ágil e interagem exclusivamente com o cliente.

6. Avaliador,que identifica os membros da equipe que ameaçam o bom funcionamento e a produtividade da equipe ágil, esse papel utiliza apoio da gestão para remoção de tais membros que não está disposto a aprender e aplicar a metodologia. O avaliador é representado pelo Técnico ágil e interagem com a gerência Sênior e a equipe.

As equipes são destinadas a ser democráticas, onde todos os membros são considerados no mesmo nível, sem uma hierarquia estrita na prática, grupos menores são mais bem adequados às estruturas democráticas do que equipes maiores. Equipes de autogestão foram descritos como equipes composta de 10 a 15 pessoas que tomam as responsabilidades, cujas atividades diárias são guiadas pela visão corporativa da gerência sênior, com indivíduos treinados e que estabelecem os seus próprios horários de trabalho, que mostra um maior compromisso com a empresa. Quatro princípios de auto-organização são definidos como: especificação mínima crítica, a variedade necessária, redundância de funções, e aprendizagem. Não é de surpreender então que as equipes de auto-organização são vistos como melhorar a flexibilidade de uma organização em termos da sua capacidade para responder às mudanças e como um fator influente na melhoria a qualidade de vida no trabalho dos empregados. (Hoda, R. ;Noble, J. ; Marshall, S. ; 2013, p.4)

3.4. Considerações do Capítulo

O método ágil Extreme Programming é a resposta aos processos tradicionais iniciado pelo Manifesto Ágil aos métodos tradicionais desenvolvido por Kent Beck e Ward Cunningham para pequenas e media equipes que necessitam de maior colaboração dos clientes, práticas eficazes de desenvolvimento de software e flexibilidade nas implementações, pois sua maior virtude é de ser um processo eficiente, que atende requisitos mutáveis e com baixos riscos de projeto.

Outra grande mudança com o XP é a utilização de programação em pares que comprova a redução em 20% a 40% o tempo de desenvolvimento e diminui a incidências de defeito e falhas no código desenvolvido. E melhora a consistência na equipe tanto nas relações interpessoais como confiança e aumento de *feedback*.

Nessa sessão foram apresentadas as práticas do XP como:Planejamento incremental, Pequenos releases, Projeto simples, Desenvolvimento test-first, Refactoring, Programação em pares, Propriedade coletiva, Integração contínua, Ritmo sustentável e Cliente presente para execução e adaptação da metodologia

XP e os valores que devem ser incorporados na organização como: Codificar, Testar, Ouvir e Projetar.

O XP é um processo dinâmico, que exige envolvimento dos participantes e define a importância de papéis para organização das equipes e meios que viabilizem as práticas e valores do XP. Foram definidos seis papéis como: Mentor ou Coach, Coordenador (representante da equipe), Tradutor (traduz a linguagem de negócios para terminologia técnicas), Defensor (Responsável pela aplicação da metodologia), Promotor (Responsável pela interação com o cliente) e o Avaliador (Apoia na gestão de remoção do obstáculo da metodologia).

A seção 3.3 detalha as responsabilidades e papéis requeridos para o bom desempenho do XP, porém não explícita de que maneira as pessoas envolvidas podem se comportar para desempenhar satisfatoriamente para a aplicação de suas práticas.

4. ANÁLISE DOS FATORES HUMANOS NO XP

Com o objetivo de melhorar o desempenho do XP, buscou-se estudar as características e comportamentos humanos para melhorar a aplicação das práticas do XP. Para cada ação existem dificuldades e peculiaridades que devem ser contornadas pelas equipes e gerentes que acompanham de perto o desenvolvimento de software utilizando o XP e procura obter o melhor aproveitamento das práticas sugeridas.

Santos e Moura (2012. p.6)realizaram um agrupamento das práticas do XP, dividido em processo de desenvolvimento, gerenciamento, orientação a uso, excelência e desenvolvimento iterativo e incremental.Apesar de não ser o foco do estudo deste trabalho,esse agrupamento permitiu esclarecer onde essas práticas estão mais concentradas.

Tabela 2. Aspecto técnico evolutivo do XP (Santos e Moura, 2012. p.6)

Desenvolvimento iterativo e incremental		Mudança incremental (ritmo sustentável) Pequenos Lançamentos Integração contínua
Processo de Desenvolvimento	**Tarefas de trabalho**	Teste de unidade Outras tarefas
	Trabalho em equipe	Programação em Pares
	Definição de função	Desenvolvedores Clientes Consultor (opcional)
	Artefatos	Estórias do Usuário Metáfora
Processo de Gerenciamento	**Tarefas de trabalho**	Inspeções
	Trabalho em equipe	Reunião em pé Jogo do Planejamento
	Definição de função	Treinamento Acompanhamento
	Artefatos	Estórias do usuário metáfora
Orientação para uso		Práticas Definição Normas de codificação
Excelência		Projeto simples Desenvolvimento orientado a testes Refatoração

Esse agrupamento pode ser identificado por Orientação da Equipe para uso da prática de Programação no ciclo azul apresentado na seção4.1, a Práticas de Equipe no ciclo verde apresentado na seção4.2 e Questões de Definições no ciclo vermelho apresentado na seção4.3, conforme Figura 5.

Figura 5. Práticas do XP(site http://www.extremeprogramming.org)

Os fatores humanos foram alocados conforme as características de cada ciclo do XP apresentados nas próximas sessões.

4.1. Prática de Programação

As práticas de programação da equipe consistem na ação interna do XP como a ***programação em pares*** que se trata de compartilhar informações, dar e receber *feedback*, flexibilidade e compatibilidade entre as pessoas que a desenvolve. Nessa fase, deve-se desenvolver o sistema de forma ***simplificada*** e o **teste deve ser contínuo** e realizado durante o desenvolvimento. Outra prática importante é a ***refatoração*** onde são feitas alterações sem afetar o comportamento do sistema. A seguir são enumerados heuristicamente os fatores essenciais para as práticas mencionadas:

- **Envolvimento**: Este fator humano é exigido das equipes e gerentes, pois incorporar os objetivos da equipe, organização, cliente individual exigindo uma adequação dos interesses, que impõe aos participantes discutir os pontos de vistas, ideias e sugestões para o bom funcionamento da programação em pares e as práticas de simplicidade, pois evita foco em tarefas desnecessárias ou com pouca relevância.

- **Colaboração e Cooperação**: Estes fatores humanos auxiliam na adequação dos papéis e responsabilidades dentro da equipe, o que resulta na rapidez de organização e assimilação. O bom trabalho em equipe depende da utilização das informações e melhoria na comunicação que auxilia na dinâmica dos programadores em pares, eficiência no desenvolvimento e testes.

- **Confiança e Respeito**: Estes fatores humanos são importantes na dinâmica de produção de software, pois diminuem as diferenças técnicas e visões do sistema, contribuindo na diminuição de sobrecargas de certas pessoas na equipe. Ou seja, melhoria na transferência de conhecimentos técnico ou referente à parte de negócios do sistema, isso contribui na refatoração, testes e na codificação da equipe.

- **Auto-organização**: Este fator humano auxiliará na auto-análise da equipe em questão da produtividade interna e prioridades de desenvolvimento para que o prometido seja de fato cumprido. A equipe aprende a aceitar responsabilidades, e principalmente a visualizar as necessidades do cliente, assumindo o papel de coparticipante. Assim a equipe consegue realizar as práticas internas do XP de forma mais rápida e simples.

4.2. Práticas de Equipe

As práticas do ciclo do gerenciamento de produção auxiliam a equipe e o cliente no desenvolvimento de software com as práticas de ***propriedade coletiva do código*** em que todos têm autonomia sobre o código, práticas de ***padronização de código***

principalmente ao comentário de código que auxilia na leitura e entendimento do programa, práticas de ***ritmo sustentável***, práticas de ***metáforas*** que auxiliam no entendimento do sistema através de estórias e ***integração contínua*** do cliente com a equipe. A seguir, são enumerados heuristicamente os fatores essenciais para as práticas mencionadas:

- **Honestidade:** Este fator humano necessita de uma comunicação objetiva e direta essencial para que as práticas ocorram com menores índices de erros.As informações para os clientes devem ser precisas para que o mesmo esteja ciente do rumo do projeto, e os mesmos conceitos devem ser aplicados às equipes que manterão a uniformidade das informações entre os membros. Com essas ações será possível garantir a qualidade das informações trocadas com o cliente principalmente no contexto das metáforas, pois estórias ambíguas podem levar a codificações equivocadas e testes ineficientes. Dentro da equipe as informações sobre a coletividade e padronizações dos códigos devem conter somente informações relevantes para que outros membros possam dar continuidade e realizar somente manutenções nas partes necessárias no sistema.

- **Habilidade de Solução de Problemas e Conflitos:**Este fator humano auxilia principalmente nos problemas de requisitos e prioridade do ciclo.As utilizações de metáforas para tratarem regras dos sistemas minimizam esses conflitos entre a equipe e o cliente, e essas experiências adquiridas devem ser compartilhadas para que se mantenha um ritmo sustentável de produção. A equipe deve lidar com a prática de integração com o cliente que podem levar as discussões e análises de compreensões e posições diferentes que devem ser ponderadas conforme o que foi planejado.

- **Objetivo em Comum:** Este fator humano contribui para que todos tenham conhecimento do que devem realizar, das metas, dos padrões de produção, disseminação das metas, conhecimento dos prazos, enfim ajuda a manter uma padronização e o ritmo produtivo. As práticas de padronizações e propriedades coletivas devem expressar de forma explícita o conhecimento que deve ser compartilhado e disseminado entre a equipe. A relação com

cliente utiliza práticas de integração e metáfora que é a forma mais eficiente de expressar suas necessidades e entendimento do que o sistema deve realizar.O conhecimento e expectativas devem estar alinhados para que ambos tenham a mesma compreensão e torna o projeto eficiente. Com relação ao ritmo sustentável deve haver um consenso entre todos para que a rotina não comprometa a produtividade e o planejamento, e deve ser analisado pela gerência em um ritmo que não gere estresse ou desmotivação da equipe.

4.3. Questões de Definições

O ciclo de planejamento trata das metas e resultados esperados do extreme programming e por isso exige uma participação do cliente mais intensa, e outras práticas como manter uma ***equipe coesa***, realizar o ***jogo do planejamento*** que determina prazo para os próximos lançamentos. Práticas de entregas em ***pequenos lançamentos*** que são produzidos rapidamente e ***testes dos clientes*** que validam as estórias. A seguir, são enumerados heuristicamente os fatores essenciais para as práticas mencionadas:

- **Colaboração e Cooperação:** Estes fatores humanos exigem valores compatíveis e ajudam manter coesão entre os membros da equipe, cliente e melhora no fluxo de informação para o planejamento. Esse fator é importante para coesão da equipe, pois as pessoas devem sentir-se privilegiadas em poder trabalhar na equipe. E, contribuir com suas experiências nas estimativas e divisões no jogo de planejamento e nos lançamentos de entregáveis ao cliente.

- **Confiança e Respeito**: Estes fatores humanos contribuem na qualidade da extração de informações primordiais e na transferência de conhecimento melhorando a coesão da equipe, nas métricas e planejamento do projeto. A comunicação é primordial na fase de planejamento entre a equipe e clientes. Ganhar a confiança do cliente com entregas contínuas e rápidas passa maior segurança de que o produto é algo consistente e que pode ser acompanhado de perto, pois o cliente é comunicado e repassa informações que afetam o

sistema. Ao contrário do cliente que espera um longo período para receber a primeira entrega e que naquele momento a realidade já é diferente de quando foi solicitado.

- **Habilidade de Tomada de Decisão:** Este fator humano depende do estudo e do resultado de uma boa base de conhecimento, facilita ao gerente e a equipe a decidirem a melhor solução que seja benéfica para todos. Com a experiência de outros projetos ou ciclo de entrega a gerência, equipe e clientes conseguem decidir as prioridades de entregas e solucionar riscos que surgem e podem afetar os prazos e custos.

4.4. Mapeamento dos Fatores Humanos no XP

A análise do fator humano no XP tem descrito várias interações entre pessoas e práticas e de pessoas entre pessoas. Em um ambiente dinâmico com processos informais com alta dependência das pessoas (equipes e clientes).

O maior desafio tem sido adaptar as pessoas às práticas e adaptação do ambiente, pois é um processo interativo e que exige uma mudança na auto-organização das equipes e da interação com o cliente. O tratamento com as pessoas da equipe é democrático, onde todos possuem o mesmo respeito, responsabilidade, confiança e principalmente com indivíduos motivados e que aceitam novos desafios.

Através da análise dos fatores humanos foi possível fazer um mapeamento dos fatores humanos discutidos no XP, conforme apresentado na Figura 6 e Tabela 3.

Figura 6.Heurísticas das práticas do XP com os fatores humanos

	Práticas do XP	Fatores Humanos
Práticas de Programação	Programação em dupla Projeto Simples Refatoração Desenvolvimento Guiado por Testes	Envolvimento Colaboração e Cooperação Confiança e Respeito Auto-organização
Práticas de Equipe	Propriedade Coletiva Padrões de Codificação Ritmo Sustentável Metáfora Integração Contínua	Honestidade Habilidade de Solução de Problemas e Conflitos Objetivo em Comum
Questões de Definições	Jogo do Planejamento Testes do Cliente Pequenos Lançamentos Time Coeso	Colaboração e Cooperação Confiança e Respeito Habilidade de Tomada de Decisão

Tabela 3. Distribuição das práticas do XP com os fatores humanos.

O fator humano ***envolvimento*** é considerado como uma característica que afeta seus participantes a desenvolver software de forma simples, objetivo e em grupo. Os fatores ***colaboração e cooperação*** auxiliam no trabalho em equipe, divisão de papéis, responsabilidades, aprimora a comunicação da equipe e com o cliente melhorando no fluxo de informação para o planejamento.

Os fatores ***confiança e respeito*** melhoram a transferência de conhecimentos técnicos e de negócios do sistema e com isso os clientes ganham maior segurança no produto desenvolvido, pois é algo consistente, que pode ser acompanhado e comunicado sobre os impactos. O fator ***auto-organização*** auxilia na produtividade e no comprometimento da equipe com cliente,o fator ***honestidade*** melhora a utilização de informações relevantes e na transferência de conhecimento entre clientes e equipe.

O fator ***habilidade de solução de problemas e conflitos*** auxilia na análise de compreensões e posições diferentes sobre os sistemas e melhora nas soluções de conflitos de requisitos e diminui riscos de falhas de projetos.E por fim o fator ***objetivo em comum*** prioriza conhecimentos e expectativas que devem estar alinhadas para que todos tenham a mesma compreensão do sistema.

4.5. Considerações do Capítulo

Neste capítulo foram descritas as características e comportamentos humanos mais adequados às práticas do XP na etapa de programação que consiste em práticas para o desenvolvimento de software como: refatoração, programação em pares e testes dos fatores humanos tais como: Envolvimento, Colaboração, Cooperação, Confiança, Respeito e Auto-organização que estudados contribuem para compartilhamento de informação, flexibilidade, *feedback* e promove o espírito de equipe que são adequados e justificáveis para as práticas do XP.

Destaque nas práticas de equipe que focam na excelência da produção apoiando o desenvolvimento com referências, metas e padrões. As práticas são propriedade coletiva, padronização, ritmo sustentável, metáfora e integração contínua, que exigem características humanas como preocupação com a qualidade da informação, tratar conflitos, expressar e compartilhar conhecimentos exigindo fatores humanos como Honestidade, Objetivo em comum, Habilidade de solução de problemas e conflitos.

A base do XP são questões relacionadas às práticas de definições e questões de relacionamento com o cliente e organização da equipe e das atividades como: o jogo

do planejamento, pequenos lançamentos, formação de time coeso e testes de cliente. Portanto, exigem características que contribuem no relacionamento entre equipe, gerência e clientes, na troca de conhecimentos, na passagem segurança e nas escolhas do rumo do projeto de forma benéfica a todos. Os fatores humanos que mais se destacaram foram Colaboração, Cooperação, Confiança, Respeito e habilidade de tomada de decisão.

5. CONSIDERAÇÕES FINAIS

As práticas da metodologia ágil XP tem o objetivo de melhorar a qualidade do software focando no rápido desenvolvimento e com mais interação entre as pessoas a fim de diminuir erros de entendimento e falta de testes. No entanto essas atividades podem ser influenciadas por barreiras provocadas pelas pessoas envolvidas, que torna uma variável imprevisível, pois o sucesso depende do fator humano que atua em todo momento da metodologia XP.

5.1. Contribuições do Trabalho

Para avaliar o fator humano foi estudada a metodologia XP, as formas de interação entre o cliente, desenvolvedores e gestores, valores, interações e práticas e os principais fatores humanos e características para a qualidade de desenvolvimento de software. Baseado nestas informações pode-se extrair uma combinação de heurísticas de fatores humanos adequados e melhorar as práticas da Metodologia XP descrita por Kent Beck.

Esses fatores estão agrupados em níveis que fornece o aprimoramento dos processos de desenvolvimento de software iniciando na produção de software e em um grau acima em que o processo de práticas de equipe tem a preocupação com garantia da qualidade e o último processo de definição da produção ou planejamento.

Apesar de existir inúmeros fatores que influenciam no desenvolvimento de software, os fatores mencionados nesse estudo cobrem todas as práticas e descrevem uma justificativa do ponto de vista do fator humano: a realização das práticas da metodologia de forma mais detalhada.

5.2. Trabalhos Futuros

Como futuros trabalhos será possível avaliar as características necessárias para cada prática do XP, melhorando sua utilidade e aprofundando as heurísticas encontradas. Também, pesquisas podem ser realizadas por amostragens em

empresas com grau de experiências diferentes, porte e diferenças culturais, aplicando questionários aos gerentes, programadores e clientes para descobrir perspectivas diferentes sobre os fatores humanos nas práticas do XP.

REFERÊNCIAS

BECK, K. **Extreme Programming Explained: embrace change**. Boston: Addison-Wesley**,** US ed.1 edition, 1999. 224 p.

BRAZIER, A. Human factors in systems engineering. In: **Proceedings of IEEE Colloquium on System Sciences Successful Introduction of Systems Engineering into an Organization (Ref. No. 1999/037),**. London, IEEE Computer Society, 1999. p. 1 - 4.

CHAO, J.; ATLI, G. Critical personality traits in successful pair programming**. Agile Conference.** IEEE Computer Society, Minneapolis, 2006. p. 88 - 93.

DRAGHICI, A.; BURLOIU, C. A.; DEACONESCU, R.; KARLSSON, M.; MULLER, D. Teamwork: A Decentralized, Secure and Portable Team Management System. In: **Proceedings of 12th International Symposium on System Sciences Parallel and Distributed Computing (ISPDC), 2013,** IEEE Computer Society, Bucharest, 2013. p. 182 - 189.

DYBA, TORE; DINGSOYR, T. **What Do We Know about Agile Software Development?**, IEEE Software v.26, IEEE Computer Society, 2009. p. 6 - 9.

GRAY, A.; JACKSON, A.; STAMOULI, I.; TSANG, S.L. Forming successful extreme programming teams. In: **Agile Conference**, IEEE Computer Society, Minneapolis,2006. p. 389 - 399

HIGHSMITH, J. **History: The Agile Manifesto**. Disponível em: http://www.agilemanifesto.org/history.html acessado em 24 de novembro de 2013.

HODA, R.; NOBLE, J.; MARSHALL,S. **Self-Organizing Roles on Ágile Software Development Teams,** v.39, IEEE Computer Society, 2013. p. 422 - 444

HODA, R.; NOBLE, J.; MARSHALL, S. Organizing self-organizing teams. In: **Proceedings of 32nd International Conference on System Sciences Software Engineering, 2010** IEEE Computer Society, Cape Town, 2010. p. 285 - 294

IN, H.; ROY, S. Issues of visualized conflict resolution. In: **Proceedings of Fifth IEEE International Symposium on System Sciences Requirements Engineering, 2001. Proceedings**. IEEE Computer Society, Toronto, 2001. p. 314 - 315

JURIC, R. **Extreme programming and its development practices**; 2000;Disponívelem http://www.extremeprogramming.org/rules/commit.html, acessado em 18 de abril de 2014

KHAN, H. H.; MALIK, N.; USMAN, M.; IKRAM, N. Impact of changing communication media on conflict resolution in distributed software development projects. In: **Proceedings of 5th Malaysian Conference in Johor Bahru Software Engineering (MySEC), 2011**, IEEE Computer Society, Johor Bahru, 2011. p. 189 - 194.

LEVY, M.; HAZZAN, O. Knowledge Management in Practice: The Case of Agile Software Development. In: **Proceedings of ICSE Workshop on System Sciences Cooperative and Human Aspects on Software Engineering, 2009. CHASE '09**, IEEE Computer Society, Vancouver, 2009. p. 60 - 65.

MARNEWICK, A.; PRETORIUS, J.H.; PRETORIUS, L.A perspective on human factors contributing to quality requirements: A cross-case analysis. In: **Proceedings of International Conference on System Sciences Industrial Engineering and Engineering Management (IEEM), 2011,** IEEE Computer Society, Singapore, 2011. p. 389 - 393.

MILLER, K.W.; LARSON, D.K. Agile software development: human values and culture. **IEEE Technology and Society Magazine,** v. 24, IEEE Computer Society, USA, 2005. p. 36 - 42.

OFFNER, A.; SWINDLER, S.; PADULA, G.; KING, A.; FEDORA, J. Change Management: Developing a Tool to Foster Adaptive Collaboration. In: **Proceedings**

of International Conference on System Sciences Collaboration Technologies and Systems (CTS), 2011, IEEE Computer Society, Philadelphia, 2011. p. 606 - 611.

OSATUYI, B.; HILTZ, S.R.; FJERMESTED, J. The Impact of Importance and Distribution on Information Exchange in Team Decision Making: Preliminary Results. In: **Proceedings of 45th Hawaii International Conference on System Sciences System Science (HICSS), 2012**, IEEE Computer Society, Maui, 2012. p. 3786 - 3795.

OZKAYA, A. **R&D Team's Competencies, Innovation, and Growth With Knowledge Information Flow**. IEEE Transactions on System Sciences Engineering Management**,**, IEEE Computer Society, 2010. p. 416 - 429.

PADBERG, F.; MULLER, M.M. Analyzing the cost and benefit of pair programming. In: **Proceedings of Ninth International, Software Metrics Symposium, 2003.** IEEE Computer Society, 2003. p. 166 - 177**.**

PEIXOTO, C.S.A.; SILVA, A.E.A. A Conceptual Knowledge Base Representation for Agile Design of Human-Computer Interface. In: **Proceedings of Third International Symposium on Nanchang Intelligent Information Technology Application, 2009, IITA 2009,** IEEE Computer Society, Nanchang,2009. p. 156 - 160.

PRESSMAN, R.S. **Software Engineering: A Practitioner's Approach**, 7 ed., McGraw-Hill Higher Education, Universidade da Califórnia, 2010. 928 p.

SAMPAIO, A.; SAMPAIO, I.B.; GRAY, E. The need of a person oriented approach to software process assessment. In: **Proceedings of 6th International Workshop on San Francisco Cooperative and Human Aspects of Software Engineering (CHASE), 2013,**IEEE Computer Society, San Francisco, 2013. p. 145 - 148.

SANTOS, F.S.; MOURA, H.P. Analyzing the Intertwining of Social and Technical Aspects in Agile Methods.In: **Proceedings of International Conference on**

Lausanne Social Informatics (SocialInformatics), 2012, IEEE Computer Society, Lausanne, 2012. p. 320 - 327.

SOMMERVILLE,I. **Engenharia de Software**. Pearson Addison Wesley,8ª Edição, São Paulo,2010. 549 p.

TELES, V.M. **Extreme Programming**. Novatec Editora, São Paulo 2004. 320 p.

VERSIONONE, Agile Methods & Practices In: **Proceedings of 7th Conference Annual State of Agile Development Survey**. 2013. Disponível em:http://www.versionone.com/pdf/7th-Annual-State-of-Agile-Development-Survey.pdf. Acessado em 06 de julho de 2014.

WANG,Y.; CHEN,M. A collaborative knowledge production model for knowledge management in complex engineering domains. In: **Proceedings of International Conference on System Sciences Systems, Man and Cybernetics, 2004, volume 6,** IEEE Computer Society, China, 2004. p. 5050 - 5055.

WANG, Y. On cognitive properties of human factors in engineering. In: **Proceedings of Fourth IEEE Conference on System Sciences Cognitive Informatics, 2005, (ICCI 2005), ,**IEEE Computer Society, Canadá, 2005. p. 174 - 182.

WELLS,D. **Extreme Programming: A Gentle Introduction**, 2009, Disponível em http://www.extremeprogramming.org, Visitado em 03 de março de 2014.

ZHANG, C.; ZONG, G. Team decision making of scientific research organization based on knowledge management. In: **Proceedings of International Conference on Chengdu Advanced Management Science (ICAMS), 2010,** v.1**,** IEEE Computer Society, Chengdu, 2010. p. 252 - 255.