AS DIFICULDADES EM APLICAR O *FRAMEWORK SCRUM* EM UM AMBIENTE DOMINADO PELO CICLO DE VIDA EM CASCATA[1]

Yuri Petit Lobão Ferreira Tourinho

Resumo: Nos dias atuais, o grande desafio das empresas de desenvolvimento de *softwares* é, sem dúvida, a transição do modelo de desenvolvimento tradicional para uma metodologia ágil. Nesse contexto, este artigo tem por objetivo apresentar as dificuldades da aplicação do *framework Scrum* em uma empresa em que o ciclo de vida em cascata era utilizado. Para validar a análise, foi aplicado um questionário a 20 colaboradores da empresa, o *Scrum Team*. Os dados foram coletados no período de 1º a 2º de agosto de 2017 e estão apresentados em forma de tabelas descritivas. Os resultados da pesquisa apontaram que o modelo em cascata está fadado ao fracasso. Nesse sentido, a adoção do *Scrum* mostrou-se positiva e promissora na empresa pesquisada, apresentando-se mais adequado aos cenários de desenvolvimento atuais, visto que com menos gente e menos tempo, alcançou resultados com qualidade melhor e custos menores.

Palavras-chave: *Scrum*. Metodologia Ágil. Engenharia de *Software*.

1 INTRODUÇÃO

Atualmente, os termos Engenharia de *Software* e *Scrum* são recorrentes nas empresas de desenvolvimento de sistemas. Segundo Magela (2006), a Engenharia de *Software* abrange um conjunto de técnicas, métodos, ferramentas e processos, os quais são utilizados nas atividades de especificação, construção, implantação e manutenção de um sistema. Além disso, seu objetivo é garantir a gerência, o controle e a qualidade dos artefatos produzidos por meio de recursos humanos. Por sua vez, o *Scrum* é um *framework* ágil para gestão e planejamento de projetos de produtos complexos. Para Schwaber e Sutherland (2017), *Scrum* é um modelo utilizado para desenvolver e manter

1 Artigo apresentado como Trabalho de Conclusão do Curso de Especialização em Gerência de Projetos de Tecnologia da Informação, da Universidade do Sul de Santa Catarina, como requisito parcial para a obtenção do título de Especialista em Gerência de Projetos de Tecnologia da Informação.

produtos, possibilitando que as pessoas tratem e resolvam situações complexas e adaptativas enquanto produzem e entregam produtos com o mais alto valor possível.

Nesse sentido, as instituições desenvolvedoras de sistema, sejam elas públicas ou privadas, procuram adequar-se as melhores práticas do mercado. No cenário corrente, procuram adaptar-se às novas metodologias de desenvolvimento com o intuito de construir sistemas melhores de maneira mais eficaz.

No entanto, ao adotar uma metodologia ágil, ocorre uma quebra de paradigma. Isto é, o modelo de desenvolvimento que vinha sendo utilizado será substituído por outro. Na maioria dos casos, o modelo em cascata é substituído pelo *Scrum* e tal substituição traz grandes dificuldades e desafios para a organização, visto que seus recursos humanos estão acostumados ao antigo modelo.

Sendo assim, o presente trabalho analisou as dificuldades e os desafios dos recursos humanos em adotar uma metodologia ágil, ou seja, o *Scrum* em uma organização em que o modelo em cascata está institucionalizado. Especificamente, este trabalho identificou a estrutura organizacional da empresa pesquisada; identificou os recursos do *Scrum* que foram utilizados na empresa pesquisada; analisou as dificuldades e os desafios dos recursos humanos na adoção do *Scrum*; e comparou as práticas ágeis adotadas na empresa pesquisada com as boas práticas presentes na literatura.

Como metodologia, o presente trabalho analisou a adoção do *framework Scrum* em uma empresa de desenvolvimento de *softwares*. Nesse sentido, a pesquisa classificou-se como aplicada e empírica, uma vez que investigou e coletou fatos inerentes à empresa. Apesar do viés empírico, também foi classificada como teórica visto que envolveu uma etapa de revisão da literatura, a qual buscou o conhecimento no assunto tratado.

Quanto ao aprofundamento do estudo, considerou-se de origem descritiva pois expos as características da empresa, ou seja, descreveu sua realidade quanto ao *Scrum*. Sendo assim, a coleta de dados foi a partir de um estudo de caso realizado na empresa em questão.

Com o intuito de preservar a identidade e reputação da empresa, esta recebeu o nome fictício de “Casa do Rúgbi”. A Casa do Rúgbi é uma empresa de Tecnologia da Informação (TI) voltada para o desenvolvimento de *softwares*. Ela possui uma Subdivisão de Desenvolvimento e Manutenção (SDDM), onde foram extraídos os sujeitos da pesquisa. Sendo um total de 20 colaboradores (*Scrum Team*), cada um

desempenhando um dos seguintes papéis: *Product Owner*; *Scrum Master*; *Develop Team*, composto por: analistas de sistema, analistas de banco de dados, arquitetos, desenvolvedores, testadores e documentadores.

Por fim, para alcançar os objetivos propostos, foi realizada uma coleta de dados, por meio de questionários anônimos, junto ao *Scrum Team*. Além disso, ainda contou com as observações e análises do próprio pesquisador.

As demais seções deste artigo estão organizadas da seguinte forma: a seção 2 aborda os conceitos inerentes ao ciclo de vida em cascata e ao *framework Scrum*. A aplicação do *Scrum* na Casa do Rúgbi, os resultados do questionário aplicado e suas análises estão detalhados na seção 3. Finalmente, a seção 4 apresenta as conclusões a partir dos resultados obtidos.

2 DO CICLO DE VIDA EM CASCATA AO *FRAMEWORK SCRUM*

O surgimento das primeiras linguagens de programação de alto nível, como o FORTAN (1954), COBOL (1959) e SIMULA (1967), desencadeou projetos de *softwares* e sistemas nas áreas acadêmica, comercial, espacial, governamental e militar. Segundo Magela (2006), devido à popularização das linguagens de programação, o número de instruções na construção de *softwares* saltou de 0,1 milhão, em 1955, para 1,4 milhão, em 1970. Diante de tais números, a necessidade de organizar, gerenciar e controlar milhões de linhas de código tornou-se imprescindível para o desenvolvimento de sistemas.

Paralelamente ao crescimento do código-fonte, Magela (2006) ainda destaca a inversão nos custos de *hardware* e *software*, onde, em 1955, o custo do *software* representava 13% ante aos 87% do *hardware*, crescendo para 68% ante aos 32%, em 1970. Para Paula Filho (2009), tal aumento é fruto de dois principais fatores: falta de técnicas para um mapeamento correto das necessidades do usuário. Ou seja, o que o usuário solicitava era entendido diferente por quem desenvolvia; e aumento na complexidade do *software*. Isto é, um número maior de problemas começou a ser abordado pela comunidade de desenvolvimento de *software*.

Na década de 70, muitos projetos das áreas acadêmica, comercial, espacial, governamental e militar apresentaram problemas em suas execuções, como: orçamentos além do previsto; prazos vencidos; baixa qualidade na codificação; baixa adesão aos requisitos; e cancelamento devido à inviabilidade técnico-financeira. Tais fatos,

segundo Magela (2006, p. 21), fez surgir o termo Crise do *Software*, uma vez que:

> […] as empresas gastavam milhões, e seu retorno de investimento (ROI) era muito baixo. E, pior ainda, quando terminavam de pagar um software, ele já estava obsoleto e, se um dia havia chegado a atender às necessidades da empresa, não era mais capaz de fazer isso.

Como pode ser observado, desenvolver *softwares* e sistemas nunca foi uma atividade trivial no âmbito da computação. Durante as fases de especificação, construção, implantação e manutenção, vários problemas podem ocorrer e levar ao insucesso do projeto como um todo. Dentre esses problemas, pode-se citar: mudanças de escopo/requisito; especificação das necessidades pelo cliente; elicitação dos requisitos pelo analista; orçamento; prazo; e outros.

Com vistas a mitigar tais problemas, as empresas de desenvolvimento de sistemas passaram a adotar conceitos e práticas inerentes à Engenharia de *Software*, como os modelos de desenvolvimento e a metodologia ágil. Nesse sentido, faz-se necessário discorrer acerca de tais temas, respectivamente, como o ciclo de vida em cascata, amplamente utilizado em um passado não muito distante, e o *framework Scrum*, em voga na atualidade.

2.1 Ciclo de Vida em Cascata

O ciclo de vida em cascata ou modelo cascata (*waterfall*) é um modelo de ciclo de vida de um processo, o qual, segundo Magela (2006, p. 27), "representa uma abordagem específica de desenvolvimento de *software* baseada no relacionamento entre etapas e atividades de um processo". Magela (2006, p. 26) complementa esse conceito ao definir processo como "coordenação na aplicação de técnicas, métodos e ferramentas e recursos humanos durante a especificação, construção, implantação e manutenção de um *software*". Por fim, Paula Filho (2009) destaca que o ponto de partida de um processo de desenvolvimento de *software* é a escolha de um modelo de ciclo de vida.

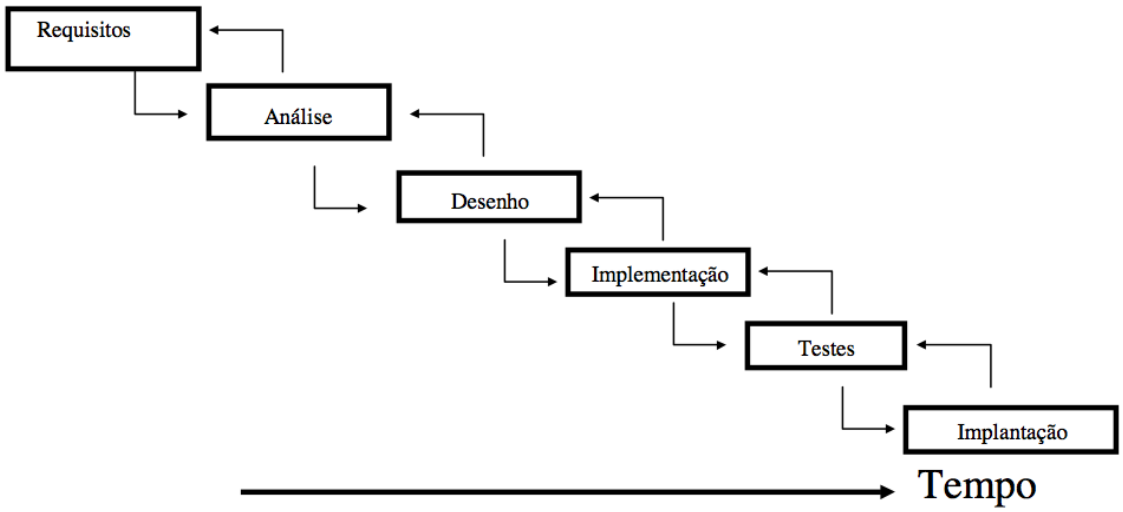
O ciclo de vida em cascata nasceu de uma visão industrial da manufatura de produtos (Magela, 2006). Segundo Paula Filho (2009), o ciclo de vida em cascata é um modelo de desenvolvimento de *software* sequencial, isto é, os principais subprocessos são executados em estrita sequência, conforme a Figura 2.1.1.

Após a finalização das atividades de um subprocesso, outro subprocesso iniciará suas atividades, por exemplo, o subprocesso análise inicia após a conclusão do

subprocesso requisitos, até que todos os subprocessos sejam executados e o produto final seja entregue ao cliente.

Para Paula Filho (2009), a característica sequencial torna o processo rígido e burocrático, visto que as atividades de requisitos, análise e desenho têm de ser muito bem desenvolvidas a fim de evitar erros. Além disso, o modelo é de baixa visibilidade para o cliente, o qual terá acesso ao produto final apenas na conclusão do projeto.

Figura 2.1.1 – O modelo de ciclo de vida em cascata



Fonte: Paula Filho (2009, p.24).

Para Magela (2006, p. 29), “esse modelo já nasceu com críticas”. Como os subprocessos de requisitos, análise e desenho dependem da elicitação das necessidades do cliente, necessidades essas que podem ser fornecidas pelo cliente de maneira ambígua ou percebidas pelo analista de forma equivocada, as maiores contradições e inconsistências são descobertas durante o subprocesso de implementação, forçando uma retomada e uma reanálise do problema. Outrossim, a natureza dinâmica dos *softwares* atuais, os quais estão em constante transformação e adequação, torna quase impossível a utilização dessa abordagem (Magela, 2006).

De forma a sintetizar as principais características do modelo em cascata, pontua-se as suas principais vantagens: os subprocessos são fundamentais e estão logicamente em ordem; funcionam como pontos de controle, facilitando a gestão do projeto; e cada um inicia após o término do subprocesso anterior, permitindo que todos os insumos necessários estejam disponíveis. Além disso, suas desvantagens: modificações nos

requisitos são desencorajadas, dados o custo de uma alteração tardia; problemas descobertos na fase de implementação e testes possuem um alto custo de correção; o produto final é visto tardiamente pelo cliente, que pode receber um produto diferente do desejado; e desaconselhável para desenvolvimento de *softwares* de grande porte, dado a constante evolução do sistema e a necessidade de modificações/inclusões de requisitos.

2.2 *Framework Scrum*

A terminologia *Scrum* tem origem no rúgbi, denotando uma jogada onde toda a equipe se junta pela disputa da bola contra a equipe adversária (Foggetti, 2014). Como *framework*, surgiu no início dos anos 90, criada por Ken Schwaber e Jeff Sutherland, com o propósito de gerenciar e desenvolver produtos. Mais especificamente, o *Scrum* é um *framework* para desenvolver, entregar e manter produtos complexos. Isto é, o *Scrum* não é um processo, uma técnica ou um método definitivo. Em vez disso, é um *framework* dentro do qual pode-se empregar vários processos ou técnicas (Schwaber e Sutherland, 2017). Em complemento, de acordo Foggetti (2014), o *Scrum* é um *framework* ágil para ser utilizado onde os requisitos mudam rapidamente e não há uma técnica específica para desenvolvimento, apenas um conjunto de regras e práticas de gestão.

Em relação ao conjunto de regras e práticas de gestão, o *framework* é composto por times associados a papéis, eventos e artefatos. Cada componente tem o seu propósito e é essencial para o uso e sucesso do *Scrum* (Schwaber e Sutherland, 2017).

O *Scrum Team* é auto-organizável e multifuncional. Segundo Prikladnicki, Willi e Milani (2014), consiste em 3 papéis: *Product Owner*, *Develop Team* e *Scrum Master*.

O *Product Owner*, também chamado de dono do produto, é a pessoa responsável por maximizar o valor de negócio do produto, bem como gerenciar o seu *Backlog*. O *Develop Team* é composto por profissionais multifuncionais que organizam e gerenciam seus próprios trabalhos, a fim de entregarem um *Product Increment* "pronto" ao término de cada *Sprint*. O *Scrum Master* é o responsável por promover e suportar o *framework* com processos e iniciativas para melhorar o trabalho, ajudando a todos a entenderem a teoria, as práticas, as regras e os valores do *Scrum*.

Quanto ao *Develop Team*, esse deve ser pequeno o suficiente para se manter ágil e grande o necessário para completar os trabalhos da *Sprint*. A quantidade de integrantes varia de 3 a 9 profissionais. Os papéis *Product Owner* e *Scrum Master* não integram

essa contagem (Schwaber e Sutherland, 2017).

Os eventos ou cerimônias *Scrum* ocorrem durante a *Sprint*. Considerada o coração do *framework*, a *Sprint* é um *time-boxed*, com duração de até 1 mês, onde um *Product Increment* "pronto" (desenvolvido, testeado e utilizável) será entregue. Ao longo de todo o desenvolvimento, as *Sprints* possuem a mesma duração e uma nova *Sprint* começa, somente, quando a anterior termina (Schwaber e Sutherland, 2017). Conforme Prikladnicki, Willi e Milani (2014), existem 4 eventos ou cerimônias: *Sprint Planning*, *Daily Scrum*, *Sprint Review* e *Sprint Retrospective*.

A *Sprint Planning* é uma reunião de planejamento onde será conhecida e planejada a meta da *Sprint*. É um *time-boxed* com no máximo 8 horas para uma *Sprint* com duração de 1 mês e responde a duas perguntas: a primeira, "o que pode ser entregue como resultado do *Product Increment* da próxima *Sprint*?", e a segunda, "como o trabalho necessário para entregar o *Product Increment* será realizado?".

As *Daily Scrum* são reuniões diárias realizadas em todos os dias da *Sprint* na mesma hora e no mesmo local. É um *time-boxed* de 15 minutos e cada participante do *Develop Team* deve responder a 3 perguntas: a primeira, "o que fiz ontem que ajudou o *Develop Team* a atingir a meta da *Sprint*?", a segunda, "o que farei hoje para ajudar o *Develop Team* atingir a meta da *Sprint*?", e a terceira, "vejo algum obstáculo que impeça a mim ou o *Develop Team* no atingimento da meta da *Sprint*?".

A *Sprint Review* é a revisão do *Product Increment* "pronto" entregue no final da *Sprint*, onde o incremento será inspecionado e o *Backlog*, se necessário, adaptado. É um *time-boxed* com no máximo 4 horas para uma *Sprint* com duração de 1 mês. A *Sprint Retrospective* é o momento para o *Scrum Team* inspecionar a si próprio e criar um plano de melhorias para as próximas *Sprints*. Ocorre após a *Sprint Review* e antes do *Sprint Planning* da próxima *Sprint*. É um *time-boxed* com no máximo 3 horas para uma *Sprint* com duração de 1 mês.

Os artefatos do *Scrum* representam o trabalho ou valor, fornecendo transparência e oportunidades para inspeção e adaptação. Além disso, permite que todos tenham o entendimento dos artefatos (Schwaber e Sutherland, 2017). Para Prikladnicki, Willi e Milani (2014), existem 3 artefatos: *Product Backlog*, *Sprint Backlog* e *Product Increment*.

O *Product Backlog* apresenta uma visão atualizada dos requisitos desejados para o produto. O *Sprint Backlog* trata de um planejamento estratégico e tático da próxima

Sprint em nível mais micro. Ou seja, é um conjunto de itens do *Product Backlog* selecionados para a *Sprint*. O *Product Increment* contêm as funcionalidades do produto "pronto". Isto é, a soma de todos os itens do *Product Backlog* completados durante a *Sprint* e o valor dos *Product Increment* de todas as *Sprints* anteriores.

Em uma *Sprint*, os itens do *Sprint Backlog* recebem o nome de tarefas. Essas tarefas são anotadas em *post-its* e coladas em um quadro conhecido por Kanban. O quadro é divido em colunas: "A fazer", "Fazendo" e "Feito". No início da *Sprint*, o *Scrum Team* cola, na coluna "A fazer", a maior quantidade possível de *post-its* que eles acreditam que conseguirão realizar. À medida que os dias passam, os integrantes da equipe pegam as tarefas e deslocam para a coluna "Fazendo". Quando uma atividade é finalizada, ela é movida para "Feito". Assim, o Kanban tem o intuito de prover a gestão à vista, isto é, dispor as informações relevantes à vista dos colabores e gestores (Sutherland, 2016).

Para o *Scrum*, requisitos são bem-vindos e o *Product Backlog* é sempre passível de mudanças como afirmam Schwaber e Sutherland (2017, p. 14):

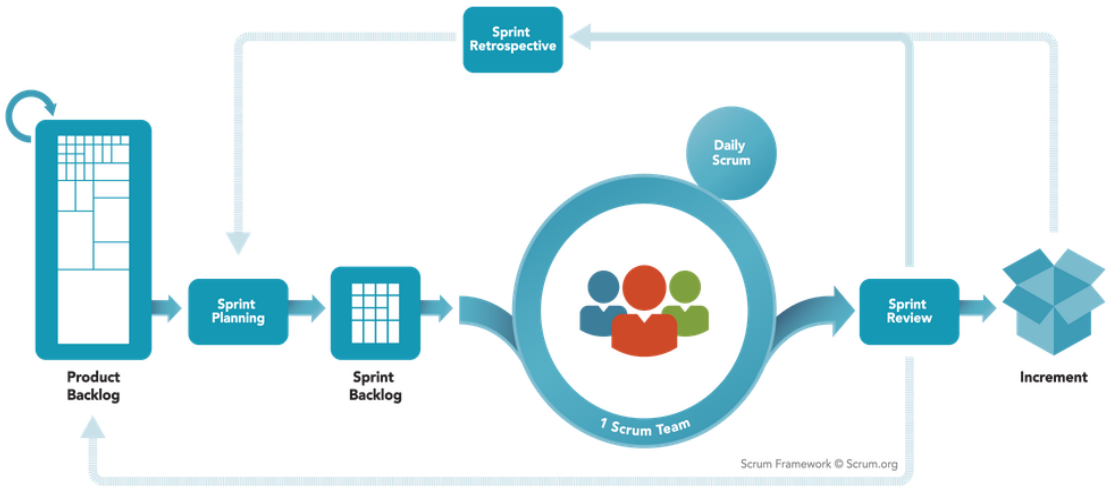
> Um *Backlog* do Produto nunca está completo. Os primeiros desenvolvimentos estabelecem os requisitos inicialmente conhecidos e melhor entendidos. O *Backlog* do Produto evolui tanto quanto o produto e o ambiente no qual ele será utilizado evoluem. O *Backlog* do Produto é dinâmico, mudando constantemente para identificar o que o produto necessita para ser mais apropriado, competitivo e útil. Se um produto existe, seu *Backlog* do Produto também existe.

A Figura 2.2.1 ilustra o *framework Scrum*, onde o *Product Backlog* é definido. Uma *Sprint Planning* é realizada e a partir dela é priorizado um *Sprint Backlog*. Esse *Sprint Backlog* será desenvolvido pelo *Develop Team* durante o período da *Sprint* e diariamente ocorrerão as *Daily Scrum*. Ao final da *Sprint*, é realizada a *Sprint Review* e o *Product Increment* é entregue. Por fim, é realizada a *Sprint Retrospective*, a última cerimônia da *Sprint*.

Nesse sentido, o *framework* torna a entrega de produtos mais eficaz, posto que adapta-se às realidades das mudanças. Os requisitos de maior valor são desenvolvidos primeiro. Caso haja necessidade de mudanças, o *Scrum Team* poderá facilmente mudar as prioridades. Nesse sentido, Cohn (2011) e Prikladnicki, Willi e Milani (2014) destacam as vantagens do *Scrum*: equipes pequenas e multidisciplinares que produzem versões incrementais de um produto em iterações curtas, sendo auto-organizáveis para

planejar e desenvolver as *Sprints*; trabalho em equipe facilitado pelo *Scrum Master*, que remove impedimentos e reforça os pontos centrais do *Scrum* ao longo do desenvolvimento do produto; trabalho organizado a partir do *Product Backlog*, constantemente revisado e priorizado; e comunicação e cooperação intensa entre a equipe ao longo do desenvolvimento do produto.

Figura 2.2.1 – O *framework Scrum*



Fonte: Scrum.org. Disponível em: <https://www.scrum.org>. Acesso em: 3 dez. 2017.

3 APLICAÇÃO DO *SCRUM* NA CASA DO RÚGBI

Historicamente, a Casa do Rúgbi apresenta, segundo o *Project Management Body of Knowledge* – PMBOK® (PMI, 2004), uma organização matricial balanceada, sendo composta por uma Subdivisão de Desenvolvimento e Manutenção (SDDM). Abaixo dessa, existem quatro seções, são elas: Seção de Análise de Negócios (SAN), Seção de Arquitetura e Implementação (SAI), Seção de Teste (STE) / Seção de Documentação (SDO) e Seção de Administração de Dados (SAD). Por sua vez, cada seção possui um grupo de colaboradores especialistas nas respectivas áreas. A estrutura organizacional da Casa do Rúgbi está ilustrada na Figura 3.1.

Antes da adoção do *Scrum*, quando algum projeto era iniciado, um colaborador de alguma das seções da SDDM exercia o papel de Gerente de Projeto. Em seguida, uma equipe era formada por colaboradores de cada seção, cada qual em sua especialidade. A organização matricial balanceada da Casa do Rúgbi está ilustrada na

Figura 3.2.

Figura 3.1: Estrutura Organizacional da Casa do Rúgbi.

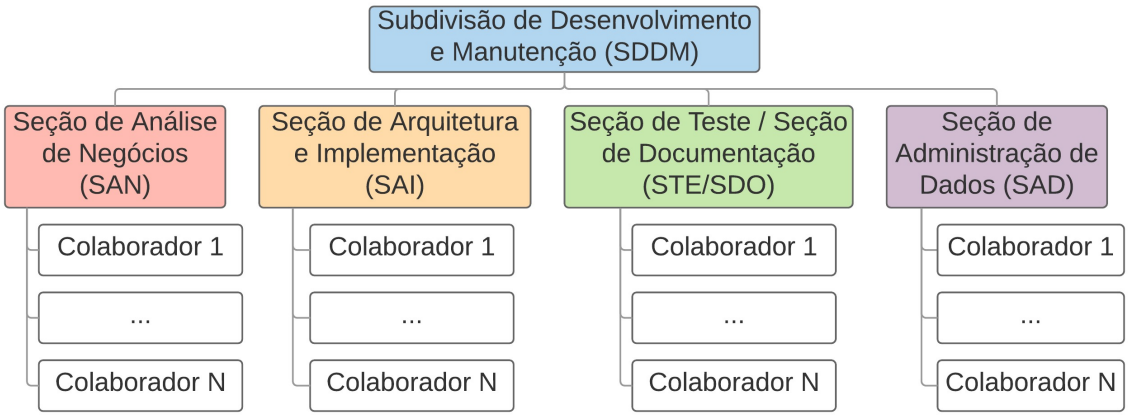
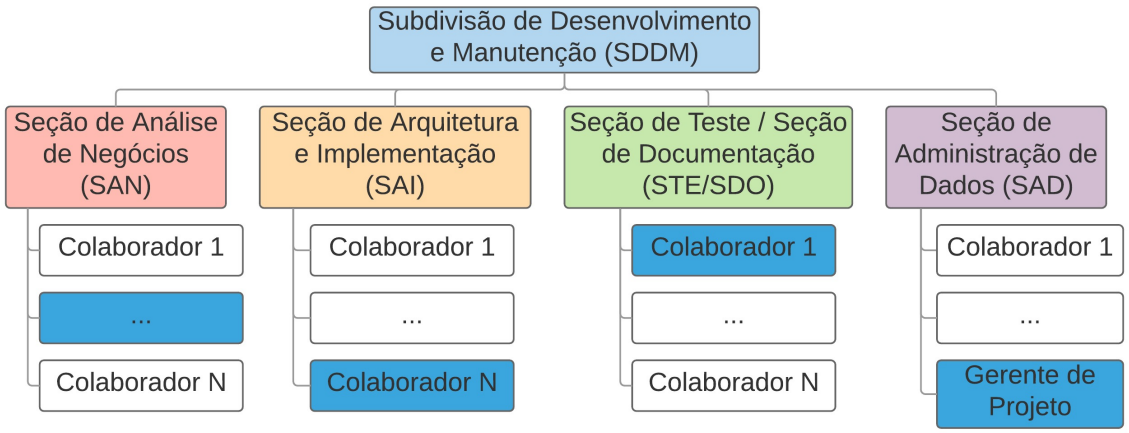


Figura 3.2: Organização matricial balanceada da Casa do Rúgbi.



3.1 Adoção do *Scrum*

Para a adoção do *Scrum* na Casa do Rúgbi algumas medidas foram elaboradas e aplicadas. Um dos colaboradores, detentor de vasto conhecimento em metodologias ágeis, foi selecionado para ministrar um treinamento acerca do *Scrum* aos demais membros da empresa. Esse treinamento contemplou os conceitos iniciais do *framework Scrum*, como a teoria, os papéis e os eventos.

Após a capacitação da equipe, um projeto-piloto foi selecionado para aplicar o *framework*. No entanto, a estrutura organizacional da empresa foi mantida e, agora, um colaborador de alguma das seções da SDDM recebeu o papel de *Product Owner* e outro

colaborador o de *Scrum Master*. Por fim, o *Develop Team* foi formado por uma equipe de colaboradores de cada seção, cada qual em sua especialidade.

A formação do *Develop Team* contradiz um dos pontos do *Scrum,* o qual, segundo Schwaber e Sutherland (2017), prega por um time multifuncional, isto é, um colaborador pode desempenhar mais de um papel (analista, desenvolvedor, testador e documentador) no *Develop Team*. Além disso, o *Develop Team* totalizou 18 colaboradores, contradizendo novamente o *Scrum*, o qual prega que o *Develop Team* seja "pequeno o suficiente para se manter ágil e grande o suficiente para completar um trabalho significativo dentro da *Sprint*" (Schwaber e Sutherland, 2017, p. 7), sugerindo um total de 3 a 9 colaboradores.

3.1.1 *Planejamento da Sprint*

Com um *Scrum Team* formado e um projeto selecionado, tiveram início as cerimônias do *Scrum*. Para cada *Sprint*, em seu início, a *Sprint Planning* era realizada. O intuito de tal evento era possibilitar ao *Develop Team* o planejamento de quais artefatos do *Product Backlog* seriam entregues na *Sprint* (com duração de duas semanas). Cada artefato recebia uma pontuação no *Planning Poker* e, em seguida, era disponibilizado no quadro Kanban por meio de um *ticket*. Tais *tickets* eram atribuídos aos membros do *Develop Team*, os quais ficavam incumbidos de implementar a solução computacional.

A pontuação por meio do *Planning Poker* ocorria por parte dos desenvolvedores e, devido ao fato do *Develop Team* não ser um time multifuncional, não era considerado o tempo do testador e do documentador. Logo o tempo da *Sprint* abrangia apenas o tempo de codificação.

3.1.2 *Execução da Sprint*

Diariamente, ocorriam as *Daily Scrum*. Essas cerimônias eram realizadas impreterivelmente às 10 horas com todos os membros do *Scrum Team*. A reunião possuía um tempo fixo de 15 minutos, com o objetivo de expor as atividades concluídas, as que seriam realizadas e os impedimentos.

Após a resolução de todos os *tickets*, o produto desenvolvido era disponibilizado para testes pela STE/SDO ao passo que a *Sprint Review* era realizada com o *Product Owner, Scrum Master e Develop Team*.

Nesse ponto, não existia a interação do produto com o cliente. Isto é, o cliente não tinha acesso às entregas parciais, o que resultou em mudanças de requisitos e retrabalho. Além disso, o produto entregue não havia sido testado. Ou seja, a definição de "pronto" contradiz com Schwaber e Sutherland (2017, p. 18), o qual deveriam incluir "critérios mais rigorosos de alta qualidade". Por fim, os testadores e documentadores não participavam da *Sprint Review*, uma vez que estavam executando os testes de aceitação, regressão e exploratórios.

3.1.3 *Finalização da Sprint*

Findo os testes, a última cerimônia do *Scrum* era realizada, a *Sprint Retrospective*. Nessa reunião, os membros do *Scrum Team* inspecionavam a si próprios e sugeriam melhorias ao processo. Em suma, apontavam os erros e vislumbravam possíveis melhorias.

Uma das principais críticas nesse momento era a realização dos testes após a finalização *Sprint*. Isto é, um produto funcional e testado não era entregue ao cliente. O teste do produto ocorria na semana posterior ao término da *Sprint*. Tal fato tornava a *Sprint* com tamanho real de três semanas em vez de duas.

3.2 Resultados do Questionário Aplicado

Seguindo a metodologia proposta, este trabalho foi realizado por meio de um questionário qualitativo para conduzir a pesquisa. Esse questionário encontra-se no Apêndice A.

O questionário eletrônico foi aplicado ao *Scrum Team*, composto por 1 *Product Owner*, 1 *Scrum Master* e 18 *Develop Team*, totalizando uma amostra com 20 colaboradores da empresa Casa do Rúgbi, no período compreendido entre os dias 1° e 2° de agosto de 2017. Além disso, o questionário foi respondido de forma anônima com o intuito de permitir a livre resposta do pesquisado.

O questionário foi divido em 3 (três) partes: a primeira, com perguntas direcionadas ao contexto geral da transição para a metodologia ágil; a segunda, direcionada à implantação do *framework Scrum*; e a terceira, voltada às questões específicas referentes às outras técnicas ágeis utilizadas.

A elaboração das questões teve por base a adaptação da escala *Likert*, a qual é avaliada como de fácil entendimento e de muita clareza. A escala *Likert* é uma escala

amplamente utilizada, a qual exige que os entrevistados indiquem um grau de concordância ou discordância com cada uma das questões dos objetos de estímulo (Malhotra, 2004).

Neste trabalho, a escala *Likert* teve uma adaptação em que as opções de resposta variam, em sua maioria, de "totalmente positiva" até "totalmente negativa".

Os resultados do presente estudo evidenciaram que o questionário "Pesquisa Desenvolvimento de *Software*", idealizado para a avaliação subjetiva da aplicação do *framework Scrum* em um ambiente dominado pelo ciclo de vida em cascata, apresentaram os dados estatísticos a seguir.

Na Tabela 1, podem ser vistos os dados estatísticos relevantes do contexto geral da transição para a metodologia ágil. Nota-se que houve predomínio de 95% da amostra em reconhecer o uso do *framework Scrum* na instituição. Tal percentual pode ser justificado pelo fato dos participantes da pesquisa vivenciarem as cerimônias (reuniões) do *Scrum*, ou seja, as *Sprint Planning*, *Daily Scrum*, *Sprint Review* e *Sprint Retrospective*, conforme preconizadas por Schwaber e Sutherland (2017). Além disso, 70% ainda reconheceu o uso do quadro *Kanban*, visto que o *Scrum* o utiliza como auxiliar para a gestão à vista das tarefas selecionadas para a *Sprint*, por meio de *post-its*, como indicado por Sutherland (2016, p. 21).

Ainda na Tabela 1, os resultados estatísticos das perguntas 1.2, 1.3, 1.4 e 1.5 demonstraram que possuem correlação significativa, visto que a metade da amostra considera a transição do modelo tradicional para as metodologias ágeis parcialmente positiva, sugerindo que seriam necessárias intervenções para a melhoria da metodologia. Tal fato se confirma quando 80% dos colabores remetem à necessidade de algum processo tradicional necessário nesta transição, contradizendo Sutherland (2016, p. 15), o qual afirma que o modelo em cascata gasta centenas de milhões de dólares e, geralmente, não alcançam os resultados. Desse maneira, é percebido um possível equívoco da amostra haja vista que, além disso, 85% consideram que a organização estava parcialmente preparada para tal transição.

Mesmo existindo a necessidade de melhorias, 55% consideram totalmente positiva a mudança no processo de desenvolvimento, confirmando os resultados obtidos por Leidemer (2013), em que os entrevistados de sua pesquisa também consideram totalmente positiva a mudança no processo de desenvolvimento.

Na Tabela 2, encontram-se os dados estatísticos relevantes da implantação do

framework Scrum. Quanto ao gerenciamento de escopo e gerenciamento de configuração, respectivamente, 55% e 40% da amostra avaliam a mudança como parcialmente positiva. Tais percentuais diferem de Leidemer (2013), visto que em sua coleta de dados a amostra foi pequena, apenas 3 (três) indivíduos, sendo essa uma limitação do seu estudo. No entanto, Sutherland (2016, p. 16) afirma que o *Scrum* possibilita à equipe um posicionamento cuidadoso, unidade de propósito e clareza de objetivos. Dessa forma, essas avaliações podem ser um equívoco dos participantes por não compreenderem os propósitos do *framework* ou por este ter sido aplicado em um projeto-piloto e os mesmos estarem acostumados ao antigo processo, o modelo cascata.

Tabela 1: Contexto geral da transição para a metodologia ágil

Pergunta	Percentuais Relevantes
1.1 Qual(is) *framework(s)* adotados na composição da metodologia na sua organização?	95% - Scrum e 70% - Kanban
1.2 Como está sendo a transição do modelo tradicional para as metodologias ágeis na sua organização?	50% - Parcialmente positiva
1.3 Com relação a metodologia tradicional, existe algum processo que, na sua opinião, ainda é necessário nesta transição?	80% - Sim, parcialmente
1.4 Você considera que a mudança de processo de desenvolvimento está sendo, no geral?	55% - Totalmente positiva
1.5 Na sua opinião, a organização está ou estava preparada para transição de metodologias?	85% - Parcialmente preparada

Quanto ao gerenciamento de tempo, 60% dos pesquisados consideram totalmente positiva a mudança no processo. Esse indicador confirma a idéia de inspeção e adaptação proposta por Sutherland (2016, p. 17), onde é possível verificar em intervalos regulares se o projeto está no caminho certo e, ainda, se é possível aprimorar a forma como o trabalho está sendo realizado de modo a obter resultados melhores e mais rápidos.

Em relação ao gerenciamento da qualidade, 60% dos colaboradores consideram parcialmente positiva, confirmando a mesma avaliação de Leidemer (2013). Os motivos que levaram a tal percentual estão relacionados as atividades de teste. No caso, os testes eram realizados de forma manual e ocorriam na semana seguinte ao término da *Sprint*. Logo, não existia um produto pronto e testado, o que impedia a equipe de receber um *feedback* quase imediato do trabalho realizado, conforme defendido pelo *Scrum* (Sutherland, 2016, p. 21). Isso refletiu nos indicadores relacionados à utilização de

testes ágeis, onde 45% consideraram totalmente positiva e outros 45% consideraram parcialmente positiva. Ou seja, a utilização de testes automatizados auxiliou positivamente a velocidade na execução de testes, mas, ainda assim, os testes manuais eram realizados, acarretando lentidão ao processo.

Em atenção ao gerenciamento de risco, 35% da amostra avaliou como indiferente. Tal percentual deve-se ao fato do gerenciamento de risco estar concentrado no *Product Owner* e no *Scrum Master*, sendo assim desconhecido aos demais membros do time.

No tocante à comunicação entre os membros, 60% do *Scrum Team* avaliam como totalmente positiva, confirmando que o *Scrum* promove a união das equipes, facilitando a comunicação e permitindo enxergar a meta final (Sutherland, 2016, p. 27).

Tabela 2: Implantação do *framework Scrum*

Pergunta	Percentuais Relevantes
2.1 Como você avalia a mudança de processo no contexto do gerenciamento de escopo (análise das novas requisições, clareza nos requisitos, documentação)?	55% - Parcialmente positiva
2.2 Como você avalia a mudança de processo no contexto do gerenciamento de tempo (delegação de tarefas, estimativas de tempo, acompanhamento do desenvolvimento)?	60% - Totalmente positiva
2.3 Como você avalia a mudança de processo no contexto do gerenciamento da qualidade (validação das implementações, gerenciamento dos *releases*)?	60% - Parcialmente positiva
2.4 Como você avalia a mudança de processo no contexto do gerenciamento de configuração (controle de versões, registro das mudanças, integração contínua)?	40% - Parcialmente positiva
2.5 Como você avalia a mudança de processo no contexto do gerenciamento de risco (gestão dos riscos, tratamento de risco)?	35% - Indiferente
2.6 Como você avalia a comunicação entre os membros da equipe com a adoção do modelo ágil?	60% - Totalmente positiva
2.7 Como você considera a mudança de processo na utilização de testes ágeis?	45% - Totalmente positiva e 45% - Parcialmente positiva

Na Tabela 3, estão dispostos os dados estatísticos relevantes das questões específicas referentes às outras técnicas ágeis utilizadas. Observou-se que 80% avaliaram a adoção do quadro *Kanban* como totalmente positiva. Tal aceitação confirma a afirmação de Kniberg e Skarin (2018), onde os membros do *Scrum Team*, não enfrentarão uma revolução drástica no modo como trabalharão, ao invés disso, serão

encorajados a uma mudança gradual, isto é, a uma evolução gradual dos processos existentes alinhadas aos valores ágeis.

Por outro lado, 45% dos participantes mostraram-se indiferentes quanto a adoção da programação em par XP (*eXtreming Programming*). Esse percentual é justificado pelo fato do *Scrum Team* não ser multidisciplinar, conforme recomendado por Schwaber e Sutherland (2017), e, ainda, pelo fato de tal prática estar relacionada ao papel de desenvolvedor (Wildt *et. al*., 2015). Assim, como metade do time não desempenhava esse papel, logo não trabalhou com tal prática e, por tal motivo, mostraram-se indiferentes quanto a essa questão.

Tabela 3: Questões específicas referentes às outras técnicas ágeis utilizadas

Pergunta	Percentuais Relevantes
3.1 Caso utilize, como você avalia a adoção do quadro *Kanban* como parte da metodologia da sua organização?	80% - Totalmente positiva
3.2 Caso utilize, como você avalia a adoção da programação em Par XP (*eXtreming Programing*) como parte da metodologia da sua organização?	45% - Indiferente

4 CONCLUSÕES

Tendo em vista tudo que foi observado nesta pesquisa, conclui-se que o método em cascata, o qual as pessoas acreditam que tudo pode ser planejado com antecedência e que nada mudará ao longo do projeto, está fadado ao fracasso nos dias atuais devido à constante mudança nos requisitos.

A transição para a metodologia ágil mostrou-se positiva e promissora na empresa pesquisa. No entanto, ainda foi possível detectar tanto uma resistência à mudança quanto um não entendimento aos propósitos do *Scrum*.

A implantação do *framework Scrum* apresentou resultados positivos para os processos de gerenciamento de configuração, escopo, tempo e qualidade com base na avaliação realizada entre os membros do *Scrum Team*.

O *framework Scrum* mostrou-se mais adaptativo aos cenários de constante mudança de requisitos, uma vez que, por meio das *Sprints*, o *feedback* dos *stakeholders* foi praticamente imediato.

Ao comparar o passado e o presente do processo de desenvolvimento de *software* da empresa, notou-se melhorias trazidas pela metodologia ágil, como facilitadores para acompanhamento do projeto e a realização de testes (automatizados)

durante a *Sprint*.

No entanto, a comunicação entre os membros da equipe continuou prejudicada, haja vista que não foi respeitado o número de participantes do *Scrum Team*.

Por fim, o *Scrum* mostrou-se mais adequado aos cenários de desenvolvimento atuais, como o caso do projeto-piloto utilizado, uma vez que com menos gente e menos tempo, conseguiu mais resultados, com qualidade melhor e custos menores.

UNISUL

REFERÊNCIAS

COHN, Mike. **Desenvolvimento de Software com Scrum**: aplicando métodos ágeis com sucesso. Porto Alegre: Bookman, 2011.

FOGGETTI, Cristiano. **Gestão Ágil de Projetos**. São Paulo: Education do Brasil, 2014.

KNIBERG, Henrik.; SKARIN, Mattias. **Kanban e Scrum**: obtendo melhor de ambos. Disponível em: <https://www.infoq.com/br/minibooks/kanban-scrum-minibook>. Acesso em: 10 fev. 2018.

LEIDEMER, Rômulo Henrique. **Implantação de scrum em uma empresa de desenvolvimento de software, do curso de Sistemas de Informação da UNIVATES**. 2013. 129 f. Monografia (Graduação em Sistemas de Informação)-Centro Universitário Univates, Lajeado, 2013.

MAGELA, Rogério. **Engenharia de Software Aplicada**: princípios. Rio de Janeiro: Alta Books, 2006.

MALHOTRA, Naresh K. **Pesquisa de Marketing**: uma orientação aplicada. 4. ed. São Paulo. Artmed, 2004.

PAULA FILHO, Wilson de Pádua. **Engenharia de Software**: fundamentos, métodos e padrões. 3. ed. Rio de Janeiro: LTC, 2009.

PMI. **Um Guia do Conjunto de Conhecimentos em Gerenciamento de Projetos** (Guia PMBOK®). 3. ed. Newtown Square: Project Management Institute, Four Campus Boulevard, 2004.

PRIKLADNICKI, Rafael.; WILLI, Renato.; MILANI, Fabiano. **Métodos Ágeis para Desenvolvimento de Software**. Porto Alegre: Bookman, 2014.

SCHWABER, Ken.; SUTHERLAND, Jeff. **Guia do Scrum**. Disponível em: <http://www.scrumguides.org/docs/scrumguide/v2017/2017-Scrum-Guide-Portuguese-Brazilian.pdf>. Acesso em: 20 nov. 2017.

SUTHERLAND, Jeff. **Scrum**: a arte de fazer o dobro do trabalho na metade do tempo. 2. ed. São Paulo. Leya, 2016.

WILDT, Daniel.; HELM, Rafael.; MOURA, Dionatan.; LACERDA, Guilherme. **Extreme Programming**: Práticas para o dia-a-dia no desenvolvimento ágil de software. São Paulo. Casa do Código, 2015.

APÊNDICE A

Pesquisa Desenvolvimento de Software

Desenvolvimento de Software

*Obrigatório

1. **1.1 Qual(is) Framework(s) adotados na composição da metodologia na sua organização ? ***

   Marque todas que se aplicam.

   - [ ] Scrum
   - [ ] XP (extreming programing)
   - [ ] Lean manufacturing
   - [ ] Kanban
   - [ ] TDD(Test-Driven Development)
   - [ ] RUP (Rational Unified Process)
   - [ ] PMBOK (Project Management Body of Knowledge)

2. **1 .2 Como está sendo a transição do modelo tradicional para as metodologias Ágeis na sua organização? ***

   Marcar apenas uma oval.

   - ◯ Totalmente positiva
   - ◯ Parcialmente positiva
   - ◯ Indiferente
   - ◯ Parcialmente negativa
   - ◯ Totalmente negativa

3. **1 .3 Com relação a metodologia tradicional, existe algum processo que, na sua opinião, ainda é necessário nesta transição? ***

   Marcar apenas uma oval.

   - ◯ Não, nenhum é necessário
   - ◯ Sim, a maioria
   - ◯ sim, Parcialmente

4. **1 .4 Você considera que a mudança de processo de desenvolvimento esta sendo, no geral: ***

   Marcar apenas uma oval.

   - ◯ Totalmente positiva
   - ◯ Parcialmente positiva
   - ◯ Indiferente
   - ◯ Parcialmente negativa
   - ◯ Totalmente negativa

5. **1.5 Na sua opinião, a organização esta ou estava preparada para transição de metodologias?** *
   Marcar apenas uma oval.
   - ◯ Totalmente preparada
   - ◯ Parcialmente preparada
   - ◯ Não estava preparada

6. **2.1 Como você avalia a mudança de processo no contexto do gerenciamento de escopo (análise das novas requisições, clareza nos requisitos, documentação)?** *
   Marcar apenas uma oval.
   - ◯ Totalmente positiva
   - ◯ Parcialmente positiva
   - ◯ Indiferente
   - ◯ Parcialmente negativa
   - ◯ Totalmente negativa

7. **2.2 Como você avalia a mudança de processo no contexto do gerenciamento de tempo (delegação de tarefas, estimativas de tempo,acompanhamento do desenvolvimento)?** *
   Marcar apenas uma oval.
   - ◯ Totalmente positiva
   - ◯ Parcialmente positiva
   - ◯ Indiferente
   - ◯ Parcialmente negativa
   - ◯ Totalmente negativa

8. **2.3 Como você avalia a mudança de processo no contexto do gerenciamento da qualidade (validação das implementações,gerenciamento dos releases)?** *
   Marcar apenas uma oval.
   - ◯ Totalmente positiva
   - ◯ Parcialmente positiva
   - ◯ Indiferente
   - ◯ Parcialmente negativa
   - ◯ Totalmente negativa

9. **2.4 Como você avalia a mudança de processo no contexto do gerenciamento de configuração (controle de versões, registro das mudanças, integração contínua)?** *
   Marcar apenas uma oval.
   - ◯ Totalmente positiva
   - ◯ Parcialmente positiva
   - ◯ Indiferente
   - ◯ Parcialmente negativa
   - ◯ Totalmente negativa

10. **2.5 Como você avalia a mudança de processo no contexto do gerenciamento de risco (Gestão dos riscos, tratamento do risco)?** *
Marcar apenas uma oval.

- ◯ Totalmente positiva
- ◯ Parcialmente positiva
- ◯ Indiferente
- ◯ Parcialmente negativa
- ◯ Totalmente negativa

11. **2.6 Como você avalia a comunicação entre os membros da equipe com a adoção do modelo Ágil?** *
Marcar apenas uma oval.

- ◯ Totalmente positiva
- ◯ Parcialmente positiva
- ◯ Indiferente
- ◯ Parcialmente negativa
- ◯ Totalmente negativa

12. **2.7 Como você considera a mudança de processo na utilização de testes Ágeis ?** *
Marcar apenas uma oval.

- ◯ Totalmente positiva
- ◯ Parcialmente positiva
- ◯ Indiferente
- ◯ Parcialmente negativa
- ◯ Totalmente negativa

13. **3.1 Caso utilize, como você avalia a adoção do quadro KANBAN como parte da metodologia da sua organização?** *
Marcar apenas uma oval.

- ◯ Totalmente positiva
- ◯ Parcialmente positiva
- ◯ Indiferente
- ◯ Parcialmente negativa
- ◯ Totalmente negativa

14. **3.2 Caso utilize, como você avalia a adoção da programação em Par XP (extreming programing) como parte da metodologia da sua organização?** *

Marcar apenas uma oval.

- ◯ Totalmente positiva
- ◯ Parcialmente positiva
- ◯ Indiferente
- ◯ Parcialmente negativa
- ◯ Totalmente negativa
