Instituto de Pesquisas Tecnológicas do Estado de São Paulo

Daniel Medeiros de Assis

Dificuldades na adoção e uso de método Scrum em empresas brasileiras não-projetizadas com processos plan-driven: estudos de caso múltiplos

São Paulo
2016

Daniel Medeiros de Assis

Dificuldades na adoção e uso de método Scrum em empresas brasileiras não-projetizadas com processos plan-driven: estudos de caso múltiplos

Dissertação de Mestrado apresentada ao Instituto de Pesquisas Tecnológicas do Estado de São Paulo - IPT, como parte dos requisitos para a obtenção do título de Mestre em Engenharia de Computação.

Data da aprovação ____/_____/_____

_____
Prof. Dr. Cláudio L. C. Larieira (Orientador)
Mestrado em Engenharia de Computação

Membros da Banca Examinadora:

Prof. Dr. Cláudio Luís Carvalho Larieira (Orientador)
Mestrado em Engenharia da Computação

Prof. Dr. Ivanir Costa (Membro)
UNINOVE – Universidade Nove de Julho

Prof. Dr. Marcelo Novaes de Rezende (Membro)
FIPT – Fundação Instituto de Pesquisas Tecnológicas do Estado de São Paulo

Daniel Medeiros de Assis

Dificuldades na adoção e uso de método Scrum em empresas brasileiras não-projetizadas com processos plan-driven: estudos de caso múltiplos

Dissertação de Mestrado apresentada ao Instituto de Pesquisas Tecnológicas do Estado de São Paulo - IPT, como parte dos requisitos para a obtenção do título de Mestre em Engenharia de Computação.

Área de Concentração: Engenharia de Software

Orientador: Prof. Dr. Cláudio L. C. Larieira

São Paulo
Novembro/2016

Ficha Catalográfica
Elaborada pelo Departamento de Acervo e Informação Tecnológica – DAIT
do Instituto de Pesquisas Tecnológicas do Estado de São Paulo – IPT

A848d **Assis, Daniel Medeiros de**
Dificuldades na adoção e uso de método Scrum em empresas brasileiras não-projetizadas com processos plan-driven: estudos de caso múltiplos. / Daniel Medeiros de Assis. São Paulo, 2016.
130p.

Dissertação (Mestrado em Engenharia de Computação) - Instituto de Pesquisas Tecnológicas do Estado de São Paulo. Área de concentração: Engenharia de Software

Orientador: Prof. Dr. Cláudio L. C. Larieira

1. Scrum 2. Software orientado a um plano 3. Estrutura organizacional não-projetizada 4. Empresa 5. Método ágil 6. Tese I. Larieira, Cláudio L. C., orient. II. IPT. Coordenadoria de Ensino Tecnológico III. Título

17-01 CDU 004.41(043)

À minha querida amiga e esposa Laryssa,
Por todo apoio, incentivo e compreensão.

À meus pais,
Por toda a luta para me propiciar um ensino de qualidade.

À Sêneca, Thoreau, Kierkegaard e tantos outros,
Que têm trazido às trevas da alma humana a luz do conhecimento.

AGRADECIMENTOS

Ao Prof. Dr. Cláudio Luís Carvalho Larieira, pelo excelente trabalho de orientação, encorajamento, aconselhamento e ensino durante a elaboração deste trabalho.

Aos membros da banca, Prof. Dr. Ivanir Costa e Prof. Dr. Marcelo Novaes de Rezende, pelos conselhos e contribuções valiosos durante o exame de qualificação.

Ao Instituto de Pesquisas Técnológicas do Estado de São Paulo, pelo ensino de alto nível oferecido durante toda a duração do mestrado.

Aos participantes dos estudos de caso, pela contribuição valiosa.

E à todos aqueles que, de forma direta ou indireta, contribuíram para a realização deste trabalho.

RESUMO

Empresas têm apresentado interesse crescente na adoção de métodos ágeis em seus processos internos de desenvolvimento de software, visando obter ganhos relacionados ao aumento de produtividade, à melhoria na qualidade de *software*, a uma maior previsibilidade em entregas, a um maior alinhamento das áreas de TI e de negócios, a uma maior velocidade nas entregas e a uma maior habilidade em gerenciar mudanças de requisitos. Contudo, as mesmas empresas possuem estruturas organizacionais e processos consolidados, que muitas vezes entram em confito com os valores e práticas exigidos pelos métodos ágeis e pelo *Scrum,* dificultando sua adoção. Este estudo apresenta as maiores dificuldades na adoção de *Scrum* em empresas que fazem uso de estruturas organizacionais não-projetizadas e processos de software orientados a um plano (*plan-driven*). Para a identificação destas dificuldades, foram realizadas entrevistas e aplicados questionários com profissionais da área de desenvolvimento de software em três empresas brasileiras que fazem uso de TI para suporte ao negócio, com estrutura e processo similares. Após a análise dos resultados, foram identificados os maiores pontos de dificuldade na adoção de *Scrum* em cada uma das empresas, que foram consolidados de forma a oferecer insumos para que as empresas fossem capazes de identificar pontos de ajustes necessários em seus processos para uma melhor adoção do *Scrum*.

Palavras-chave: Métodos Ágeis; *Scrum* em Empresas; Estrutura Não-Projetizada; Processo *Plan-Driven; Tailoring* de *Scrum*.

ABSTRACT

Dificulties in adoption and usage of Scrum method in non-projectized brazilian companies using plan-driven processes: multiple case studies

Companies have demonstrated growing concerns about the adoption of agile methods in their internal software development organization, aiming to get the benefits related to productivity increasing, software quality improvement, better delivery predictability, better relationship between IT and business areas, better delivery velocity, and a better way to handle requirements change. However, the same companies already have consolidated organizational structures and processes, which get into conflict with values and practices established by the agile methods and Scrum specificaly, hindering its adoption. This study presents the major dificulties in Scrum adoption in organizations that make use of non-projectized structures and plan-driven processes. To identify those dificulties, interviewes were conducted and questionnaires were applied in three brazilian companies that make use of IT to support their business, with similar structure and process. After the result analysis, the major points of dificulty in Scrum adoption were identified and then consolidated in a way that allow companies to identify adjustment points in its processes to improve the adoption of Scrum.

Key Words: Agile Methods; Scrum in Organizations; Non-projectized Organizational Structures; Plan-Driven Process; Scrum Tailoring.

Lista de Ilustrações

Lista de Tabelas

Lista de Abreviaturas e Siglas

ISO	International Organization for Standardization
PMI	Project Management Institute
QA	Quality Assurance
RUP	Rational Unified Process
SMA	Software Method Adoption
XP	Extreme Programming

SUMÁRIO

1 INTRODUÇÃO

Este capítulo introduz a dissertação, apresentando informações sobre a motivação, o objetivo e a justificativa, as contribuições, o método de pesquisa e a organização do trabalho.

1.1 Motivação

Nos últimos anos, os métodos de desenvolvimento ágil de *software* vêm ganhando força como uma alternativa às abordagens de desenvolvimento de *software* tradicionais pela sua proposta de lidar de forma mais eficaz com problemas e limitações críticos, tais como baixa velocidade nas entregas e dificuldade em gerenciar mudanças em requisitos (SENAPATHI; SRINIVASAN, 2014).

Os métodos ágeis vêm despertando cada vez mais o interesse de grandes empresas. A pesquisa global *State of Agile* (VERSIONONE, 2014) acompanhou um panorama crescente neste contexto entre os anos de 2009 e 2014. Na pesquisa de 2014, foram coletadas 3.925 respostas de profissionais relacionados à comunidade de desenvolvimento de *software* em empresas de portes variados (com 53% dos entrevistados relacionados a empresas com mais de 1.000 pessoas) e de diversos ramos (*software*, financeiro, saúde, governo, manufatura, telecom, mídia, transportes, dentre outros). Constatou-se que 45% dos respondentes trabalham em empresas onde a maioria dos times usam tais métodos. A mesma pesquisa, em 2009, havia identificado que apenas 31% trabalhavam em empresas onde havia no máximo dois times praticando métodos ágeis.

A pesquisa (VERSIONONE, 2014) identificou também que as características associadas a métodos ágeis que mais atraem a atenção das empresas são o aumento de produtividade (53%), a melhoria na qualidade de *software* (46%), a previsibilidade na entrega (44%) e a melhoria no alinhamento da área de TI e de

negócios (40%), além da maior velocidade nas entregas (59%) e melhor habilidade de gerenciar mudanças em requisitos (56%). Todos estes números, direta ou indiretamente, podem ser relacionados com a proposição seminal dos métodos ágeis: o Manifesto Ágil.

O Manifesto Ágil surgiu de uma iniciativa de dezessete pessoas ligadas à comunidade de desenvolvimento de *software* nos Estados Unidos, dentre os quais estavam autores de livros, profissionais conceituados no setor de desenvolvimento de *software* orientado a objetos, programadores e entusiastas de metodologias (FOWLER; HIGHSMITH, 2001). Estes profissionais foram unidos pelo interesse comum em entender novas abordagens de desenvolvimento de *software*, que emergiam naquele momento.

Este grupo de indivíduos definiu um conjunto de quatro valores e princípios que refletiam os interesses do grupo como a promoção de modelos organizacionais colaborativos e focados em pessoas e a construção dos tipos de comunidades profissionais nas quais gostariam de trabalhar (FOWLER; HIGHSMITH, 2001). Os valores e princípios do Manifesto Ágil vêm influenciando pessoas, processos e empresas desde então.

Dentre os vários métodos que ganharam impulso ou foram influenciados pelo Manifesto Ágil, o *Scrum* é aquele que mais tem chamado a atenção das empresas (VERSIONONE, 2014). Aponta-se que 56% das empresas entrevistadas adotam *Scrum*, contra menos de 10% de adoção de outros métodos ágeis, como Kanban, *Lean* e XP (VERSIONONE, 2014). O *Scrum* também incorpora as técnicas ágeis mais aplicadas pelas empresas, que, segundo a pesquisa, são as reuniões diárias (aplicadas em 80% das empresas pesquisadas), iterações curtas (em 79% das empresas), *backlogs* priorizados (79%), planejamento da iteração (71%) e retrospectivas (69%) (VERSIONONE, 2014). Embora observe-se crescente interesse, a adoção plena do *Scrum* não é um processo simples, pois muitas

empresas fazem uso prévio de modelos de desenvolvimento de software e de estruturas organizacionais conceitualmente opostos aos propostos pelo método ágil. Segundo Waarderburg e Vliet (2013), o estilo de desenvolvimento de software do *Scrum* sugere que problemas e soluções ótimas não precisam ser completamente especificados de antemão, e apontam dificuldades em conciliar este estilo em ambientes grandes e complexos de empresas que adotam uma abordagem mais racionalizada, baseada em soluções preditivas.

Nikitina e Kajko-Mattsson (2014) observam que métodos ágeis como *Scrum* chamam a atenção das empresas, contudo muitas delas lidam com problemas na adoção do método, por se tratar de um trabalho muito complexo, que inclui mudanças não apenas no processo de desenvolvimento de *software* mas também na cultura organizacional e nos padrões sociais e comportamentais dos patrocinadores envolvidos. Boehm e Turner (2003) analisam a diferença entre métodos ágeis e *plan-driven*, indicando que a comparação entre estes dois é difícil e imprecisa, devido à natureza complexa do desenvolvimento de software e à grande variedade de métodos. Apontam cinco fatores críticos de sucesso relacionados à decisão pelo uso de métodos ágeis ou plan-driven num projeto: (i) tamanho, (ii) criticidade, (iii) dinamismo, (iv) pessoal e (v) cultura. Observam que se o projeto não lida bem com apenas um fator, isto já é suficiente para que seja realizada uma avaliação de risco.

Cohn e Ford (2003) apontam diversos problemas na adoção do método *Scrum* em organizações, por meio de relato de experiência, ocorrida em sete organizações que faziam uso de métodos *plan-driven*, de quatro estados norte-americanos durante um período de quatro anos. Identificaram diversos problemas do ponto de vista dos desenvolvedores, como resistência à mudanças, insegurança na transição dos métodos e problemas com expectativas externas ao time em relação ao aumento de produtividade imediata, e do ponto de vista dos gestores, como dúvidas em como acordar novas funcionalidades do produto com os clientes, como

medir progresso, e preocupações com o impacto em outros grupos da empresa. Nerur, Mahapatra e Mangalaraj (2005) também identificam desafios na migração de métodos tradicionais para métodos ágeis, elencando um conjunto abrangente de tópicos-chave que precisam ser considerados, como gerenciamento e organização, processos, pessoas e tecnologia. Concluem que organizações voltadas à inovação tem mais capacidade de adotar métodos ágeis do que aquelas que trabalham com métodos formais.

Outra questão a ser considerada é a adoção do método ágil *Scrum* em estruturas organizacionais. A estrutura organizacional para o desenvolvimento de *software* nas empresas pode assumir um caráter projetizado (voltado a projetos), funcional (voltado a uma estrutura fortemente departamentalizada) ou matricial (um misto entre estrutura projetizada e funcional) (PMI, 2013) (KERZNER, 2009) (DINSMORE; CABANIS-BREWIN, 2006). A estrutura projetizada é a que mais se aproxima dos métodos ágeis e do *Scrum*, no sentido de que relaciona-se melhor com a necessidade de ter os profissionais do time trabalhando juntos e de forma exclusiva ou quase exclusiva durante o projeto (PMI, 2013) (KERZNER, 2009).

Conceitua-se que, em estruturas organizacionais não-projetizadas, os recursos de um dado projeto podem não ser alocados de forma exclusiva, cabendo ao coordenador do projeto negociar sua participação com responsáveis pelo recurso, sejam gerentes funcionais de áreas distintas ou outros papeis de autoridade dentro da empresa nos quais o recurso está vinculado (KERZNER, 2009). Este ponto pode ser de difícil implementação em projetos que adotam os princípios do Manifesto Ágil, como aqueles que afirmam que “Pessoas de negócio e desenvolvedores devem trabalhar juntos e diariamente durante o projeto” (FOWLER; HIGHSMITH, 2001, p. 3) e que “A forma mais eficiente e efetiva de comunicar informação com e dentro do time de desenvolvimento é a conversa presencial” (FOWLER; HIGHSMITH, 2001, p.4).

Nas estruturas organizacionais matricial e funcional é comum observar cenários onde a estruturação de departamentos da área de TI é realizada considerando as disciplinas de engenharia de *software*. Há, neste caso, uma estrutura para análise de requisitos, outra para construção de *software*, outra para testes de *software*, e assim por diante. Este tipo de estrutura, enquanto rígida, pode levantar barreiras em relação à adoção dos valores e princípios do Manifesto Ágil na organização, pois espera-se que diferentes pessoas com diferentes perfis e de diferentes departamentos possam atuar juntas, de forma colaborativa e constante, visando um objetivo comum (SCHWABER; SUTHERLAND, 2013).

Um time Scrum precisa ser multidisciplinar e auto-organizado (SCHWABER; SUTHERLAND, 2013). Num time multidisciplinar, a soma das competências dos profissionais do time deve ser suficiente para realizar todo o trabalho esperado, sem depender de profissionais de fora do time. Um time auto-organizado escolhe como irá realizar seu trabalho da melhor forma, mais do que ser dirigido por profissionais de fora do time. Acredita-se que, com isso, obtém-se otimização de flexibilidade, criatividade e produtividade. (SCHWABER; SUTHERLAND, 2013).

Muitas empresas implementam seus processos de desenvolvimento de *software* considerando modelos que podem ser descritos como *plan-driven,* ou orientados a um plano (PETERSEN; WOHLIN, 2010) (HIRSCH, 2005) (BOEHM; TURNER, 2003) (COHN; FORD, 2003) (ABRAHAMSSON et al., 2009) (NERUR; MAHAPATRA; MANGALARAJ, 2005) (WAARDENBURG; VLIET, 2013) (GREN; TORKAR; FELDT, 2014) (DINGSOYR et al., 2012) (BECK; BOEHM, 2003) (HIGHSMITH, 2002) (LEE; XIA, 2010) (TOLFO et al, 2011) (SENAPATHI; SRINIVASAN, 2014). De acordo com Petersen e Wohlin (2010), o modelo *plan-driven* é definido como aquele em que espera-se que os sistemas sejam totalmente especificados, preditivos, e que possam ser construídos por meio de planejamento extensivo. Os mesmos autores observam que *plan-driven* associa-se naturalmente aos modelos cascata de ciclo de vida de projeto e ao modelo RUP (*Rational Unified*

Process, da IBM) de ciclo de vida de produto. Outros autores concordam que o modelo cascata pode ser classificado como *plan-driven* (COHN; FORD, 2003) (HIRSCH, 2005) (GREN; TORKAR; FELDT, 2014). Empresas que utilizam modelos *plan-driven* podem encontrar dificuldades na adoção de *Scrum,* pois métodos ágeis apresentam características opostas aos modelos *plan-driven.* Uma destas características é a possibilidade de mudanças constantes e rápidas, baseado em *feedbacks* (FOWLER; HIGHSMITH, 2001). Esta característica opoẽ-se diretamente à natureza preditiva dos métodos *plan-driven.* (PETERSEN; WOHLIN, 2010).

Autores discutem haver um impasse conceitual entre os métodos ágeis e os modelos *plan-driven,* refletindo em dificuldades na adoção de métodos ágeis em empresas que já fazem uso de modelos *plan-driven,* mas que querem beneficiar-se dos ganhos de métodos ágeis (COHN; FORD, 2003) (HIRSCH, 2005) (NERUR; MAHAPATRA; MANGALARAJ, 2005) (TOLFO et al, 2011) (WAARDENBURG; VLIET, 2013) (GREN; TORKAR; FELDT, 2014). No sentido de eliminar este impasse, por meio da adaptação de métodos ágeis para adequação à um ambiente *plan-driven*, o *tailoring* poderia apresentar-se como uma opção. Dinsmore e Cabanis-Brewin (2006) descrevem o conceito de *tailoring* como sendo a capacidade de customizar o modelo de ciclo de vida de um projeto, considerando a realidade particular do projeto e as necessidades e restrições da organização, de forma a garantir que o mesmo atinja seu objetivo. Também se pode considerar iniciativas para adoção de métodos ágeis em empresas, como o modelo SMA (*Software Method Adoption*) (NIKITINA; KAJKO-MATTSSON, 2014). As autoras sugerem que, ao seguir as atividades deste modelo, as empresas são capazes de estruturar seu processo ágil de forma eficiente. Dentre alguns projetos estudados no trabalho do modelo SMA, ressalta-se em especial um caso na Suécia, onde a empresa possuía processo *plan-driven* com características similares a de empresa pesquisada neste trabalho. Contudo, embora as ideias de *tailoring* e do modelo SMA proponham-se a tratar do problema em questão, nenhuma delas o resolve completamente.

Elabora-se, portanto, com base nas motivações previamente analisadas, a seguinte pergunta de pesquisa: “como empresas brasileiras não-projetizadas e que desenvolvem software com abordagem *plan-driven* têm percebido dificuldades ao adotar o método *Scrum*?”

1.2 Objetivo

Este trabalho tem por objetivo identificar e analisar como empresas brasileiras que trabalham com a estrutura organizacional conhecida como não-projetizada e que desenvolvem software com abordagem *plan-driven* percebem dificuldades ao adotar o método *Scrum.*

Para este fim, foram cumpridos os seguintes passos: (i) identificadas e caracterizadas as dificuldades segundo informações da literatura, (ii) realizados estudos de caso em empresas brasileiras para levantamento do estado das práticas correntes e (iii) efetuada a devida consolidação e análise dos dados para entendimento de como as dificuldades são percebidas.

1.3 Justificativa

Grandes empresas têm sido atraídas pelos métodos ágeis de desenvolvimento por características tais como redução do tempo na produção de *software* (TOLFO et al., 2011), melhoria no gerenciamento de mudanças de requisitos e redução na formalização de documentação (SENAPATHI; SRINIVASAN, 2014), aumento de produtividade e de qualidade, melhor gerenciamento de prioridades e qualidade da entrega (BENEFIELD, 2008). As pesquisas relacionadas à adoção de métodos ágeis em organizações, contudo, são limitadas (NIKITINA; KAJKO-MATTSSON, 2014) (WAARDERBURG; VLIET, 2013). Observam-se esforços na implantação de métodos ágeis em gerências internas e projetos isolados dentro de grandes organizações, sendo tais iniciativas individuais no sentido de que não

seguem nenhum método ou roteiro específico (WEST; GILPIN; GRANT et al., 2011), (BENEFIELD, 2008).

A motivação para o entendimento do uso do método *Scrum* em grandes empresas foi elaborada considerando outros trabalhos que visam definir formas de se utilizar métodos ágeis em empresas de filosofia oposta, como o trabalho sobre *tailoring* de processo *Scrum* em grandes empresas com ênfase no papel de *Scrum Master* (BASS, 2014), a adoção de práticas ágeis em empresas tradicionais pela aplicação do método *Grounded Theory* (WAARDERBURG; VLIET, 2013), a aplicação de métodos ágeis (como o XP) em empresas onde existe ceticismo por parte das gerências (SVENSSON; HOST, 2005), a investigação a respeito da efetividade da adoção de métodos ágeis em empresas em termos de fatores aplicáveis (SENAPATHI; SRINIVASAN, 2014), a identificação de práticas ágeis adotadas na migração de uma abordagem *plan-driven* para uma abordagem ágil (PETERSEN; WOHLIN, 2010), a definição de um método para adoção de processos ágeis em empresas tradicionais (NIKITINA; KAJKO-MATTSSON, 2014), o interesse de empresas em adotar práticas ágeis ao mesmo tempo em que encontram desafios pela diferença conceitual em seus processos (NERUR; MAHAPATRA; MANGALARAJ, 2005), e o estudo de caso que considera os aspectos motivacionais da adoção de métodos ágeis em um organização não-ágil (GREN; TORKAR; FELDT, 2014).

As seguintes pesquisas estudam a adoção de métodos ágeis de forma implícita: a) Benefield (2008) apresenta uma experiência de adoção em uma grande empresa de tecnologia, por meio da exibição de dados consolidados, mas sem associação a métodos ou práticas; b) Hoda, Noble e Marshall (2012) desenvolveram o conceito de *Grounded Theory Method* (GLASER; STRAUSS, 1967) para *software*, de forma a aplicar um método empírico que permita analisar práticas adotadas em times ágeis auto-organizados; c) Waarderburg e Vliet (2013) aplicaram o mesmo conceito num departamento centralizado de TI, organizado de forma tradicional.

Nenhuma destas abordagens, contudo, procurou associar as práticas apresentadas àquelas utilizadas pelo *Scrum* (ou outros métodos ágeis consolidados).

1.4 Contribuições

A tônica dos trabalhos acima apresentados está relacionada à adoção de métodos ágeis em grandes empresas, seja de forma direta ou indireta. Os trabalhos apresentam estratégias empregadas e dificuldades encontradas, sendo esta informação importante para permitir um melhor entendimento do cenário em questão. Esta pesquisa estuda e considera este cenário, apoiando-se nesta literatura para elaborar um trabalho cuja contribuição esteja voltada para uma situação mais específica dentro do mesmo contexto .

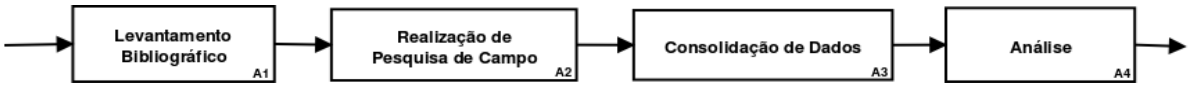
Do ponto de vista profissional, esta pesquisa contribui à indústria pela discussão sobre como o *Scrum* pode ser viabilizado a partir do entendimento de dificuldades em empresas que possuem estrutura e organização incompatíveis com os valores do Manifesto Ágil. Outras pesquisas já analisaram a adoção de práticas ágeis independentes e isoladas em empresas e da adoção do *Scrum* de forma ideal, mas não foram encontradas considerações acerca da adoção do método *Scrum* em cenários específicos como o aqui proposto, no qual a empresa possui uma estrutura organizacional não-projetizada e adota um processo *plan-driven*. Como não existe uma forma ideal de adoção do método *Scrum* em empresas com o cenário descrito neste trabalho, as empresas interessadas podem encontrar dificuldades em realizar uma adoção de resulte no atendimento às suas expectativas.

Do ponto de vista acadêmico, espera-se que a investigação realizada neste estudo de caso, considerando a experiência de profissionais de mercado, possa contribuir no entendimento acerca da adoção e aplicação do método *Scrum* em empresas com cenários similares.

1.5 Método de trabalho

Este trabalho é uma pesquisa de abordagem qualitativa, de caráter descritivo e corte transversal, baseada em estudos de caso múltiplos e que fez uso de entrevistas e questionários semi-estruturados como técnica de levantamento. A Figura 1 apresenta as macro-atividades.

Figura 1 – Macro-atividades do método de trabalho



Fonte: Elaborado pelo autor

1.6 Organização do trabalho

Os demais capítulos deste trabalho foram organizados em: Revisão Bibliográfica, que apresentou a literatura e teorias relevantes; Método de Pesquisa, onde se define a estratégia de pesquisa adotada, as atividades de levantamento e consolidação e o protocolo de estudo de dados; Estudo de caso, onde a pesquisa de campo é realizada juntamente com a devida coleta e tratamento dos dados; Análise dos estudos de caso, onde os resultados são interpretados à luz da teoria; e Conclusão, na qual foram apresentadas as conclusões e contribuições deste trabalho, bem como sugestões para futuras pesquisas relacionadas ao tema.

2 REVISÃO BIBLIOGRÁFICA

Este capítulo apresenta os principais fundamentos que servirão de base para testar a hipótese de que *Scrum* pode ser adotado em empresas organizadas de forma matricial. Sua utilidade está na preparação para a pesquisa de estudo de caso a ser desenvolvida.

2.1 Processo de *software* e *plan-driven*

2.1.1 Processos e atividades de *software*

Processo de *software* pode ser definido como um conjunto de passos ordenados que visam atingir ao objetivo de construir um *software* (FEILER; HUMPHREY, 1992). Cada passo é uma ação atômica que não tem subestrutura externamente visível, sendo uma atividade limitada, de duração finita e com um nível de abstração que depende do contexto. (FEILER; HUMPHREY, 1992). Processos de *software* são vistos como agentes de melhoria na efetividade de *software* em organizações, pois sua maturidade reduz o risco no custo imprevisto e nas estimativas de entrega (HUMPHREY; SNYDER; WILLIS, 1991).

O modelo de processo de *software* é definido de forma um pouco mais abrangente por Boehm (1988) onde, além da função primária de determinar a ordem dos estágios envolvidos no desenvolvimento e evolução de *software*, também estabelece o critério de transição de um estágio para o próximo, respondendo as questões: a) o que deve-se fazer em seguida?; b) quanto tempo deve-se continuar fazendo isto?. Ainda na visão de Boehm, um modelo de processo de *software* é diferente de um método de *software* (frequentemente chamado de metodologia) no sentido de que, enquanto o modelo de processo define a ordem de cada fase e as transições, o método define como navegar entre cada fase (envolvendo questões como particionamento de funções, alocação de recursos) e como representá-las

(com gráficos, diagramas) (BOEHM, 1988). Desta forma, os modelos tradicionais iterativo-incremental, *waterfall e* espiral são considerados modelos de processo porque definem a ordem e as transições, sem dizer como navegar e representá-las.

Existe um grande número de atividades em desenvolvimento de *software*. Melhor do que definí-las, portanto, é considerar os grupos que as definem. No guia de engenharia de *sotware* do IEEE SWEBOK (2014), as atividades de *software* são relacionadas a tópicos, dentro de diferentes áreas de conhecimento. Estes tópicos são apresentados no Quadro 1.

Quadro 1 - Tópicos das áreas de conhecimento do SWEBOK (Continua)

Áreas de Conhecimento	**Tópicos relacionados**
Requisitos de *Software*	Fundamentos de Requisitos de *Software*, Processo de Requisitos, Elicitação de Requisitos, Análise de Requisitos, Especificação de Requisitos, Validação de Requisitos, Considerações Práticas e Ferramentas de Requisitos de *Software*
Design de *Software*	Fundamentos de Design de *Software*, Questões Chave em Design de *Software*, Estrutura e Arquitetura de *Software*, Design de Interface de Usuário, Análise e Avaliação de Qualidade do Design de *Software*, Notações de Design de *Software*, Estratégias e Métodos de Design de *Software* e Ferramentas de Design de *Software*
Construção de *Software*	Fundamentos de Construção de *Software*, Gerenciamento da Construção, Considerações Práticas, Tecnologias de Construção, Ferramentas de Construção de *Software*
Teste de *Software*	Fundamentos de Teste de *Software*, Níveis de Teste, Técnicas de Teste, Métricas Relacionadas a Teste, Processo de Teste, Ferramentas de Teste de *Software*
Manutenção de *Software*	Fundamentos da Manutenção de *Software*, Questões Chave em Manutenção de *Software*, Processo de Manutenção, Técnicas para Manutenção, Ferramentas de Manutenção de *Software*
Gerenciamento de Configuração de *Software* (GCS)	Gerenciamento do processo de GCS, Identificação da Configuração de *Software*, Controle da Configuração de *Software*, Acompanhamento do Status de Configuração de *Software*, Auditoria de Configuração de *Software*, Gerenciamento de Entrega e Release de *Software*, Ferramentas de Gerenciamento de Configuração de *Software*
Engenharia de *Software*	Definição do Escopo e Inicialização, Planejamento do Projeto de *Software*, Sanção do Projeto de *Software*, Revisão e Avaliação, Fechamento, Métricas de Engenharia de *Software*, Ferramentas de Gerenciamento de Engenharia de *Software*
Processo de Engenharia de *Software*	Definição do Processo de *Software*, Ciclos de Vida do *Software*, Avaliação e Melhoria do Processo de *Software*, Métricas de *Software*, Ferramentas de Processo de Engenharia de *Software*

Quadro 1 - Tópicos das áreas de conhecimento do SWEBOK (Conclusão)

Áreas de Conhecimento	**Tópicos relacionados**
Modelos e Métodos da Engenharia de *Software*	Modelagem, Tipos de Modelos, Análise de Modelos e Métodos de Engenharia de *Software*
Qualidade de *Software*	Fundamentos de Qualidade de *Software*, Processo de Gerenciamento de Qualidade de *Software*, Considerações Práticas e Ferramentas de Qualidade de *Software*
Prática Profissional de Engenharia de *Software*	Profissionalismo, Dinâmica de Grupo e Psicologia e Habilidades de Comunicação
Economia de Engenharia de *Software*	Fundamentos de Economia em Engenharia de *Software*, Ciclo de Vida de Economia, Risco e Incerteza, Métodos de Análise de Economia e Considerações Práticas

Fonte: Elaborado pelo autor com dados do SWEBOK 3.0 (2014)

2.1.2 Plan-driven

O termo *plan-driven* tem sido empregado na literatura para descrever modelos de *software* rigorosos, onde um plano precede o desenvolvimento efetivo, geralmente envolvendo grande quantidade de documentação, em oposição às abordagens dos métodos ágeis. O Quadro 2 apresenta características de *plan-driven* de acordo com diversos autores.

Quadro 2 - Características de *plan-driven* por autores (Continua)

Características de modelo *plan-driven*	**Autor**
Não incentiva que pessoas técnicas talentosas tenham muito controle sobre como trabalham e como iteragem com pares, gerentes e clientes.	Highsmith (2002)
Modelo onde planos de processos documentados são usados para comunicar e coordenar, sendo tais planos uma grande parte da documentação do projeto, e onde aplica-se fundamentalmente o conhecimento explícito documentado. Geralmente prefere especificações formais, completas, consistentes, rastreáveis e testáveis.	Boehm e Turner (2003)
As propriedades de um produto precisam ser conhecidas e precisamente especificadas antes do início de sua construção. Também possui uma forte crença na ação de planejar e prever em desenvolvimento de software, onde ocorra um planejamento antecipado detalhado.	Hirsch (2005)

Quadro 2 - Características de *plan-driven* por autores (Conclusão)

Características de modelo *plan-driven*	**Autor**
Modelo onde os desenvolvedores tratam documentação de design em UML (Unified Modeling Language, utilizada para geração de diagramas) como artefatos principais, enquanto em processos ágeis tais artefatos são usados apenas para apoiar a ação de escrever códigos-fonte. Também apresenta baixa quantidade de comunicação entre gestor e desenvolvedor, e tomadas de decisões mais lentas em comparação com métodos ágeis.	Cohn e Ford (2003)
Modelo tradicional, em detrimento de métodos ágeis.	Nerur, Mahapatra e Mangalaraj (2005)
Modelo cuja abordagem é baseada em engenharia, racionalizada, na qual problemas podem ser totalmente especificados e existem soluções preditivas para qualquer problema. Faz-se uso de planejamento rigoroso, processos rígidos e reuso para este fim.	Waardenburg e Vliet (2013)
Existe documentação extensiva e planejamento de requisitos altamente restritivos.	Senapathi e Shrinivasan (2014)

Fonte: Elaborado pelo autor

Plan-driven não é um modelo de ciclo de vida de *software*, mas sim uma categorização de modelos, utilizado para definir modelos opostos aos ágeis. Tanto o ciclo de vida em cascata quanto o ciclo em espiral podem ser considerados modelos *plan-driven:* o primeiro por exigir documentos completamente elaborados antes da construção efetiva do *software* e por preceder o desenvolvimento efetivo do *software* por outras fases, e o segundo por manter a mesma característica do modelo cascata em relação às fases que precedem a construção. A característica principal de um modelo *plan-driven* é a existência de fases precedendo a construção. Modelos ágeis procuram remover ou reduzir ao máximo esta separação, por meio de times multidisciplinares, responsáveis por executar todas as atividades de *software* em paralelo, sem que haja uma separação formal.

Abordagens iterativo-incrementais, de forma geral, são consideradas *plan-driven* apenas quando exigem planejamento, formalização ou fases e etapas antes do desenvolvimento efetivo; quando isto não ocorre, a abordagem pode ser vista como ágil. *Scrum* é um exemplo de método que usa um modelo iterativo-incremental ágil, no sentido de que adota práticas em que o desenvolvimento ocorre juntamente

com as demais atividades de *software*, por um único time multidisciplinar em sucessivas iterações.

2.2 Gestão de projetos e estruturas organizacionais

Um projeto, segundo Kerzner (2009), pode ser definido como uma série de atividades e tarefas que: a) tem um objetivo específico para ser completado dentro de certas especificações; b) tem datas de início e fim definidas; c) tem limites de fundos; d) consomem recursos humanos e não-humanos (por exemplo, dinheiro, pessoas, equipamentos); e) é multifuncional (por exemplo, abrange várias linhas funcionais).

Ainda de acordo com Kerzner (2009), gestão de projetos pode ser definida como o planejamento, organização, direção e controle de recursos da empresa para um objetivo de relativo curto prazo, que foi estabelecido para completar objetivos e metas específicos. Segundo o PMI (2013), trata-se da aplicação de conhecimento, habilidades, ferramentas e técnicas a atividades do projeto para atingir requisitos do projeto.

A gestão de projetos é realizada por meio de processos, como inicialização, planejamento, execução, controle e fechamento. O trabalho do time de projetos, que gerencia o trabalho dos projetos, de acordo com o PMI (2013), envolve: a) consideração de aspectos como escopo, tempo, custo, risco e qualidade; b) lidar com *stakeholders* com diferentes necessidades e expectativas; c) identificar requisitos. Quanto mais se sabe sobre um projeto, mais é possível que seja melhor gerenciado e, por conta disto, muitos dos processos da gestão de projetos são inerentemente iterativos, apoiados no conceito de elaboração progressiva durante o ciclo de vida do projeto (PMI, 2000).

Em gestão de projetos, uma fase é definida como a realização completa de um ou mais entregáveis (um trabalho tangível e verificável, tal como um estudo de viabilidade, um *design* detalhado, ou um protótipo). Um ciclo de vida de projeto serve para definir o início e fim de um projeto, considerando as atividades necessárias. A definição do ciclo de vida ajuda a determinar o fluxo de ações e as atividades a serem realizadas entre o início e fim do projeto, bem como a dar visibilidade sobre como o projeto poderá ser relacionado com outros projetos futuros da organização. A sequência de fases definidas num ciclo de vida de projeto envolve alguma transferência de conhecimento de tecnologia, como por exemplo, requisitos para *design* ou construção para operação. Entregáveis da fase anterior são geralmente aprovados antes que o trabalho da próxima fase inicie, mas trabalhos podem começar antes de tal aprovação se os riscos forem aceitáveis. (PMI, 2013).

Os conceitos de projeto e gestão de projetos precisam ser aplicados por meio do uso de recursos da empresa considerando a estrutura organizacional. Uma análise de como as estruturas organizacionais influenciam projetos pode ser visualizada na Figura 2.

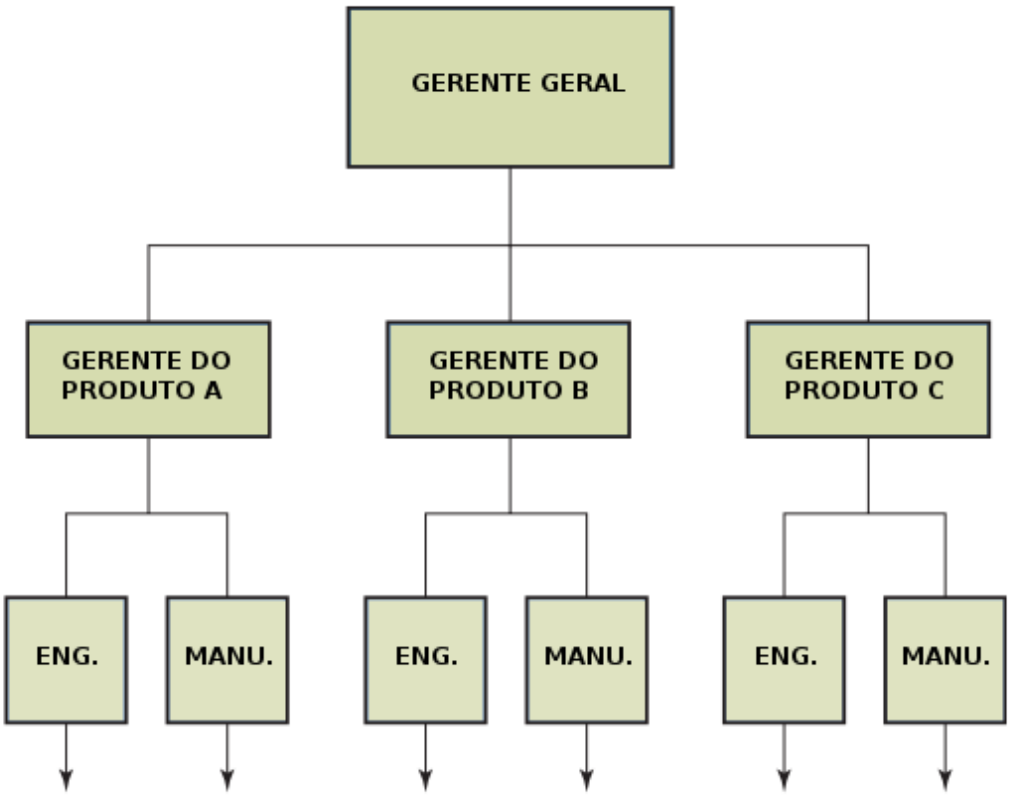
Figura 2 – Influência de estruturas organizacionais em projetos

Estrutura da organização / Características do projeto	*Funcional*	*Matricial*			*Por projeto*
		Fraca	Balanceada	Forte	
Autoridade do Gerente de Projetos	Pouco ou nenhum	Limitado	Baixo a Moderado	Moderado a Alto	Alto a Quase Total
Percentual de pessoas alocadas em projetos em tempo integral	Praticamente nenhum	0 - 25%	15 - 60%	50 - 95%	85 - 100%
Papel do Gerente de Projetos	Tempo parcial	Tempo parcial	Tempo integral	Tempo integral	Tempo integral
Títulos comuns para o papel do Gerente de Projetos	Coordenador de Projeto/Líder de Projeto	Coordenador de Projeto/Líder de Projeto	Gerente de Projeto	Gerente de Projeto/Gerente de Programa	Gerente de Projeto/Gerente de Programa
Pessoal Administrativo do Gerenciamento de Projeto	Tempo parcial	Tempo parcial	Tempo parcial	Tempo integral	Tempo integral

Fonte: PMI (2013) (tradução livre do autor)

Na estrutura organizacional projetizada, o gerente de projeto mantém controle total sobre o andamento do projeto. Os membros do projeto reportam apenas a um indivíduo, o que oferece ganhos em melhoria de comunicação comparado aos outros tipos de estruturas organizacionais. A Figura 3 apresenta esta estrutura.

Figura 3 – Estrutura projetizada (ou de projeto puro)



Fonte: Kerzner (2009) (tradução livre do autor)

Neste tipo de estrutura, as atividades são mantidas no prazo mas a tecnologia é impactada, uma vez que os departamentos funcionais não são fortes, não podendo investir em especialização tecnológica. Kerzner (2009) aponta outras desvantagens desta estrutura, tais como: o custo, pois os membros do projeto ficam totalmente alocados, sem possibilidade de serem reaproveitados em outros projetos visando redução de custo; a tendência de reter pessoas em projetos por mais tempo do que o necessário; a falta de oportunidade de troca tecnológica dos indivíduos entre projetos; falta de oportunidades de crescimento de carreira na área funcional específica.

Estruturas organizacionais funcionais são fortemente departamentalizadas, com cada departamento relacionada a uma especialidade. O papel de gerente de projetos possui pouca autoridade, concorrendo com os gerentes de departamentos por alocação de recursos. O gerente de projetos atua apenas parcialmente neste tipo de estrutura, sem disponibilidade para alocação de pessoas ao projeto em tempo integral. (PMI, 2013).

Estruturas matriciais são uma tentativa de combinar características das estruturas funcional e projetizada (KERZNER, 2009). Existem três tipos de estruturas matriciais: forte, fraca e balanceada. As estruturas matriciais fracas mantêm muitas das características de uma organização funcional, e o papel de gerente de projeto é mais voltado a um coordenador do que a um gerente; já na estrutura matricial forte, as características são mais voltadas a um modelo de organização projetizado, com gerentes de projeto alocados em tempo total e com autoridade considerável (PMI, 2013).

Observa-se ser uma prática comum de mercado que processos *plan-driven* sejam adotados sobre estruturas organizacionais não-projetizadas (matriciais e funcionais), em suas diversas variações, onde as diferentes disciplinas da engenharia de *software* (análise de requisitos, construção, testes, etc) são divididas em gerências distintas.

2.3 Métodos ágeis e *Scrum*

Waardenburg e Vliet (2013) definem o desenvolvimento de *software* ágil como o oposto dos modelos *plan-driven*: métodos caracterizados por capacidade em realizar mudanças rapidamente e aprender com elas. Este capítulo apresenta os conceitos e fundamentos por trás dos métodos ágeis, a partir da criação de iniciativas de métodos mais leves de trabalho, em contraposição ao *waterfall,*

culminando na elaboração do Manifesto Ágil, com seus valores e princípios. Dentre estes métodos, o *Scrum* é analisado detalhadamente.

2.3.1 Métodos ágeis: conceitos e fundamentos

Em 1986, o célebre artigo "*The New New Product Development Game*", de Hirotaka Takeuchi e Ikujiro Nonaka, foi publicado na Harward Business Review (TAKEUCHI; NONAKA, 1986). Neste trabalho, sugeriam uma abordagem diferente para lidar com desenvolvimento de produtos, considerando a nova ênfase do mercado por velocidade e flexibilidade. Utilizaram analogias com o esporte rugby para indicar a constante iteração entre times multidisciplinares, que emerge no processo de desenvolvimento de novos produtos. Como exemplo, sugerem que um grupo de engenheiros poderia começar a realizar o *design* de um produto antes que os testes de viabilidade deste produto estivessem finalizados. O rugby possui esta dinâmica, que não é linear, e sim adaptável. Relacionado a isto, rugby permite uma infindável variedade de táticas possíveis, assim como seria neste cenário de elaboração de novos produtos. Características apresentadas neste trabalho, como o conceito de times auto-organizados e multidisciplinares, as ideias de melhoria contínua e de adaptação, e o termo *scrum* (que significa uma jogada do rugby), seriam utilizados anos depois por Ken Schwaber e Jeff Sutherland para criar o modelo de processo de *software* ágil mais conhecido no mundo hoje: o *Scrum* (SCHWABER; SUTHERLAND, 2013).

Nos Estados Unidos, um conjunto de práticas leves de desenvolvimento de *software* começou a surgir depois do trabalho de Takeuchi e Nonaka, e alguns anos antes do próprio Manifesto Ágil. Durante os anos 1990, como reação aos problemas da prática do modelo *waterfall*, comumente utilizado, alguns modelos "ágeis" começaram a emergir, tendo mais destaque as Metodologias *Crystal*, *Feature-Driven Development* (FDD), *Extreme Programming (*XP) e *Scrum*. A crítica generalizada naquele momento era, em especial, relacionada às atividades do processo *waterfall*

de planejar, analisar e desenhar o *software* para só então efetivamente desenvolvê-lo. Outros autores não relacionam estas atividades diretamente ao *waterfall*, utilizando o termo mais genérico *plan-driven (*orientado a um plano).

Em relação a tais atividades de processos *plan-driven*, tanto *Scrum* como XP ofereceram uma mesma alternativa: realizá-las todas no momento do desenvolvimento, um pouco de cada vez, em ciclos iterativos e incrementais (SCHWABER, 1995) (BECK, 1999). Esta estratégia estaria alinhada com o Manifesto Ágil, que viria alguns anos depois para definir as bases formais dos novos métodos leves, popularizando-os.

No ano 2001, o Manifesto Ágil foi elaborado. Um conjunto de pessoas, dentre elas os criadores do *Scrum* e do *XP,* formou a *Agile Software Development Alliance* entre os dias 11 e 13 de fevereiro, criando, neste processo, o Manifesto. Este trabalho atestou que dezessete “anarquistas”, procurando encontrar melhores formas de desenvolver *software*, chegaram a quatro pontos principais de importância (FOWLER; HIGHSMITH, 2001):

- Indivíduos e iteração mais do que processos e ferramentas
- *Software* funcionando mais do que documentação compreensível
- Colaboração do cliente mais do que negociação de contrato
- Responder a mudança mais do que seguir um plano

Afirmaram que, em cada um dos pontos, os valores à esquerda devem ser mais valorizados do que os da direta (o que não significa que os valores da direta devem ser totalmente desconsiderados, como o seria numa atitude radical, diametralmente oposta).

Além destes quatro pontos, foram definidos um conjunto de doze princípios que norteiam esta mesma filosofia, a saber (FOWLER; HIGHSMITH, 2001):

- A maior prioridade é satisfazer o cliente por meio de entrega contínua e antecipada de *software* de valor;
- Mudanças em requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis sabem lidar com mudanças para a vantagem competitiva do cliente;
- Entregar *software* funcionando frequentemente, de um par de semanas a um par de meses, com preferência para a menor escala de tempo;
- Pessoas de negócio e desenvolvedores trabalham juntos e diariamente sobre o projeto;
- Construir projetos em torno de indivíduos motivados. Dêem a eles o ambiente e apóiem suas necessidades, e confie neles para fazer o trabalho;
- O método mais eficiente e efetivo de levar disseminar para e dentro do time de desenvolvimento é a conversa face a face;
- *Software* funcionando é a medida primária de progresso;
- Processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores, e usuários devem ser capazes de manter um passo constante indefinidamente;
- Atenção contínua à excelência técnica e bom design aumentam agilidade;
- Simplicidade – a arte de maximizar a quantidade de trabalho não-feito – é essencial;
- As melhores arquiteturas, requisitos e *design* emergem de times auto-organizados;
- Em intervalos regulares, o time reflete sobre como se tornar mais efetivo, e então ajusta seu comportamento de acordo.

Os métodos ágeis mais conhecidos, como *Scrum* e XP, estão alinhados com estes objetivos; seus criadores foram alguns dos dezessete membros da *Agille Alliance,* participantes da criação do Manifesto.

O Manifesto Ágil, portanto, não resultou na criação dos métodos ágeis mais conhecidos atualmente, mas contribuiu enormemente para sua popularização. Os valores e princípios do Manifesto Ágil têm servidor como referência àqueles que buscam a adoção de métodos ágeis em detrimento de processos *plan-driven*.

2.3.2 Método *Scrum*

2.3.2.1 *Scrum* como um framework e possibilidades de adaptação

O método *Scrum* é considerado um *framework.* Isto significa que define um conjunto básico de eventos, papeis e artefatos, ao mesmo tempo que permite que novas práticas sejam adicionadas, possibilitando extensão de acordo com cenários específicos (SCHWABER; SUTHERLAND, 2013). Portanto, não deve ser assumido como uma solução de processo fechada, a ser invariavelmente seguida. O *Scrum* precisa ser adaptado às características do ambiente, e oferece flexibilidade neste sentido.

2.3.2.2 Definição

Ken Schwaber e Jeff Sutherland definem *Scrum* como um *framework* para apoiar a construção de produtos complexos, onde a necessidade de adaptação é um requisito. Trata-se de um *framework* leve, simples de entender e difícil de dominar (SCHWABER; SUTHERLAND, 2013).

O *termo* Scrum originou-se no trabalho de Takeuchi e Nonata (1986), voltado ao desenvolvimento de novos produtos. Os autores apresentavam um método mais holístico do que a tradicional abordagem sequencial, que apresentava problemas num mundo que se tornava cada vez mais competitivo. A teoria foi apresentada fazendo diversas analogias ao rugby, um jogo originário da Inglaterra onde o objetivo é marcar o maior número de pontos num dado período de tempo, e onde a bola geralmente passa por todo o time para que os pontos sejam marcados. O trabalho cita o termo *Scrum*, uma jogada comum que ocorre após uma penalização ou jogada irregular, que consiste na disputa pela posse da bola por oito jogadores de cada time.

Em 1995, Ken Schwaber apresentou o trabalho *SCRUM Development Process*, descrevendo um conceito de trabalho oposto ao comumente praticado naquele momento - que valorizava o planejamento extensivo antes da construção do *software*. Este conceito, chamado *Scrum*, partia da premissa de que o processo de desenvolvimento de sistemas é algo complicado e imprevisível, e que precisava ser tratado por um processo que valorizasse essas características. O *Scrum* foi apresentado como sendo uma extensão do já existente ciclo de desenvolvimento iterativo/incremental (SCHWABER, 1995).

O *Scrum* é formado por papeis, artefatos, eventos e regras imutáveis, de forma que considera-se que implementações parciais do *Scrum* são possíveis, mas o resultado não é *Scrum* (SCHWABER; SUTHERLAND, 2013). Contudo, sua característica extensível permite que uma série de adaptações sejam feitas para atender à realidade de cada projeto; contanto que as definições gerais do *Scrum* sejam seguidas, as adaptações não descaracterizam o *Scrum*.

O *Scrum* adota uma abordagem empírica, reconhecendo que conhecimento vem da experiência, e que decisões devem ser tomadas com base no que se conhece. Possui três pilares, a conhecer: a) transparência: o processo precisa ser

visível para todos os envolvidos; b) inspeção: a cada iteração, o progresso e os artefatos são revisados em relação ao objetivo da iteração; a inspeção não pode tão constante de forma a atrapalhar o andamento da iteração, mas deve ocorrer; c) adaptação: baseado nos resultados da inspeção, pode ser necessário adaptar o processo atual e os artefatos sendo gerados.

O *Scrum* define quatro eventos formais onde a inspeção e adaptação podem ser aplicadas: *Sprint Planning, Daily Scrum, Sprint Review* e *Sprint Retrospective*. Estes eventos serão melhor explicados mais adiante, em capítulo específico.

2.3.2.3 O time *Scrum*

São três os papeis que compõem um time *Scrum*: *Product Owner*, *Scrum Master*, e Time de Desenvolvimento. Os times, idealmente, são auto-organizados e multidisciplinares. Times auto-organizados não necessitam de um controle rígido; o próprio time define como irá trabalhar, tendo em vista o objetivo, e assumindo a responsabilidade de atender às expectativas, que precisam ser claras e transparentes. Times multidisciplinares são aqueles que contém todos os perfis necessários para atender ao objetivo (pessoas com perfil de programação, de análise, de testes), e que não dependem de recursos de fora do time para atingir seus objetivos. Desta forma, o *Scrum* entende que flexibilidade, criatividade e produtividade são otimizados.

O *Product Owner* (dono do produto) é aquele que possui a responsabilidade de maximizar o valor do trabalho feito pelo Time de Desenvolvimento. A forma de fazer isto varia de acordo com a organização. Possui a responsabilidade de gerenciar o *Product Backlog (*a lista de funcionalidades desejadas para o sistema), podendo fazer ele mesmo ou delegar para o Time de Desenvolvimento. Trata-se de uma única pessoa, e não um grupo de pessoas. Pode representar os desejos de um comitê de pessoas, mas a manipulação do *Product Backlog* deve passar

obrigatoriamente por ele. Comumente faz a interface com o cliente em termos de funcionalidades a serem entregues, e negocia as estimativas e expectativas da entrega de funcionalidades com o Time de Desenvolvimento. Precisa ser respeitado pela organização em suas decisões.

O Time de Desenvolvimento é o grupo de pessoas que constrói e entrega o incremento de *software* definido no *Product Backlog* no fim de cada iteração. Apenas membros deste time podem entregar o incremento. A organização dá ao time poderes para organizar e gerenciar seu próprio trabalho. Desta forma, o controle de atividades e horas gastas por recurso não existe; as métricas não relacionam-se à quantidade de horas e atividades de cada pessoa, mas sim pelo incremento de *software* entregue pelo time ao fim de cada iteração. Com a sinergia causada por esta forma de trabalho, o *Scrum* afirma que a eficiência e eficácia do time são otimizadas. Quanto ao tamanho do time, o *Scrum Guide* (SCHWABER; SUTHERLAND, 2013) sugere um número entre três e nove, considerando que menos de três pessoas reduz a iteração e resulta em menores ganhos de produtividade, e mais de nove pessoas gera muita complexidade para que um processo empírico possa gerenciar.

O *Scrum Master* é responsável por garantir que o processo *Scrum* seja entendido e que seja aplicado por todos os envolvidos. Regras, práticas e teorias do *Scrum* precisam ser seguidas, e este papel é quem garante isto. Usa-se o termo líder-servidor para descrever a ação do *Scrum Master*, sendo aquele que direciona o time a trabalhar da forma correta, ao mesmo tempo que ensina àqueles fora do time a entender as iterações do time e como relacionar-se com esta forma de trabalho. O *Scrum Master* iterage de formas diferentes e específicas para lidar com os interesses e necessidades do *Product Owner*, do Time de Desenvolvimento, e da organização.

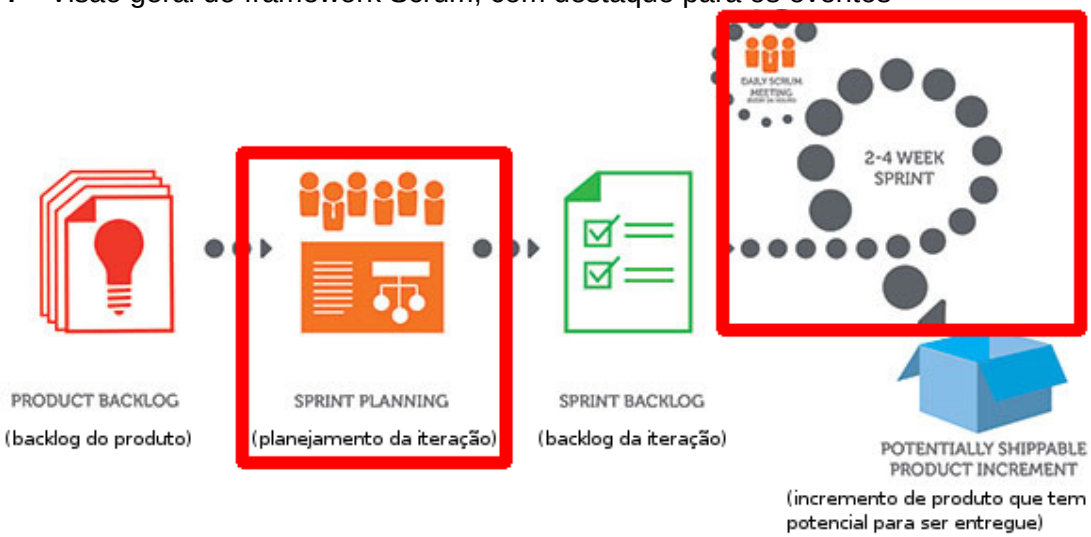
2.3.2.4 Os eventos do *Scrum*

Scrum usa eventos para minimizar a necessidade de reuniões, ao mesmo tempo em que cria uma regularidade dentro do time. Todos os eventos tem duração máxima. Sua duração é definida quando do momento da iteração, e não pode ser alterada. Cada um dos eventos é uma oportunidade para inspeção e adaptação, com exceção do *Sprint*, que é a iteração em si, propriamente dita. Os eventos aqui descritos foram extraídos do *Scrum Guide* (SCHWABER; SUTHERLAND, 2013).

O *Sprint* é a iteração do *Scrum*. Dura um período de tempo fixo, que pode ser de duas a quatro semanas, onde o incremento de *software* é construído e entregue. Um novo *Sprint* inicia imediatamente após a conclusão do *Sprint* anterior. O *Sprint* é composto de *Sprint Planning* (planejamento da iteração), *Daily Scrum Meetings* (reuniões diárias), o trabalho de desenvolvimento, o *Sprint Review* (revisão da Sprint) e o *Sprint Retrospective* (retrospectiva da Sprint).

A Figura 4 apresenta uma visão geral do atual framework *Scrum*, com destaque para o *Sprint Planning*, as *Daily Scrum Meetings* e o *Sprint*, com duração de duas a quatro semanas.

Figura 4 – Visão geral do framework Scrum, com destaque para os eventos



Fonte: Adaptado de ScrumAlliance.org

Durante o *Sprint*, algumas regras devem ser obedecidas, a conhecer: a) não podem ser feitas mudanças em escopo que impactem no objetivo do *Sprint*; b) os objetivos de qualidade não podem ser reduzidos; c) o escopo pode ser melhor entendido e renegociado com o *Product Owner,* conforme o time aprende mais sobre o requisito a construir. Um *Sprint* pode ser cancelado antes que termine, se assim o *Product Owner* desejar. Isto pode ocorrer em casos onde o objetivo do *Sprint* torne-se obsoleto antes da finalização do *Sprint*. Não se trata de uma prática comum.

Sprint Planning é o planejamento do *Sprint*. Para um mês de *Sprint*, a reunião de *Planning* deve durar um máximo de oito horas; para *Sprints* menores, este tempo reduz proporcionalmente. O *Scrum Master* é responsável por explicar o propósito ao time, e garantir que o time realize este rito. Na *Planning*, o *Product Owner* apresenta os requisitos priorizados do *Product Backlog*, definindo o objetivo do *Sprint* e explicando cada uma das funcionalidades, em alto nível. O Time de Desenvolvimento contribui com questionamentos para melhor entendimento do que deve ser feito. Para identificar o quanto de requisitos o time pode assumir para entrega na iteração, apóiam-se na *performance* da iteração anterior e na capacidade

projetada do time, também definida com base em iterações anteriores. Apenas o Time de Desenvolvimento pode definir o que entra na iteração, com base em seus números de capacidade e no entendimento do requisito. Também é definido o objetivo do *Sprint*, que ajuda o Time de Desenvolvimento a entender o motivo pelo qual os requisitos devem ser implementados. Então, o Time de Desenvolvimento trabalha no *design* dos requisitos definidos para entrada no *Sprint*. Cria-se um *Spring Backlog* (ou o escopo da iteração), contendo o esforço para os primeiros dias do *Sprint*; durante o *Sprint*, este *backlog* será refinado com base em melhor entendimento. No fim da reunião de *Planning*, o time deve ser capaz de explicar ao *Product Owner* e *Scrum Master* qual será a estratégia para construção dos requisitos.

Daily Scrum é a reunião diária. Este evento dura no máximo quinze minutos, e serve para sincronizar atividades e criar um plano para as próximas vinte e quatro horas. Na reunião, cada participante deve responder três questões: 1) o que fiz ontem que ajudou o Time de Desenvolvimento a atingir o objetivo do *Sprint*?; 2) o que farei hoje para ajudar o Time de Desenvolvimento a atingir o objetivo do *Sprint*?; 3) Vejo algum impedimento que me impede ou ao Time de Desenvolvimento de atingir o objetivo do *Sprint*? A reunião é realizada todo dia no mesmo horário e local, para reduzir complexidade. O *feedback* rápido oferecido por esta reunião permite que problemas sejam rapidamente identificados e que ações possam ser tomadas para que a entrega do *Sprint* não seja comprometida. O *Scrum Master* precisa garantir que o time realize as reuniões, mas cabe ao time conduzí-las. O *Scrum Master* também ensina ao time como manter a reunião num limite máximo de quinze minutos, e reforça a necessidade de todos os membros do time participarem. Segundo o *Scrum*, as reuniões diárias são uma reunião-chave para a aplicação dos pilares de inspeção e adaptação: melhoram comunicação, eliminam outras reuniões, identificam impedimentos, promovem rápida tomada de decisão, e melhoram o nível de conhecimento do Time de Desenvolvimento.

Sprint Review é a reunião de revisão. Acontece no final do *Sprint*, e tem duração de quatro horas para um *Sprint* de um mês (devendo ser reduzida proporcionalmente para durações menores). Durante esta reunião, é apresentado o que foi construído no *Sprint*. Trata-se de uma reunião informal, que incita o *feedback* entre os envolvidos e promove colaboração. Os participantes da reunião são o Time *Scrum* e outros *stakeholders* que o *Product Owner* queira convidar. O Time de Desenvolvimento explica o que deu certo no *Sprint*, quais problemas foram encontrados, e como os problemas foram resolvidos. O Time de Desenvolvimento responde perguntas sobre o incremento de *software* sendo entregue. O time colabora sobre o que pode-se fazer em seguida, de forma que esta reunião provê insumos para a reunião de *Sprint Planning*. O resultado da reunião é um *Product Backlog* revisado que define os itens do *backlog* que serão provavelmente requeridos no próximo *Sprint*.

Sprint Retrospective é a retrospectiva do *Sprint*. Nesta reunião, o time realiza uma auto-inspeção, de forma a: i) analisar os problemas do *Sprint* relacionados a pessoas, ferramentas, processos, relacionamentos; ii) identificar e ordenar os itens mais importantes que deram certo e potenciais melhorias; iii) criar um plano para melhorias na forma como o time realiza o trabalho. Trata-se de uma reunião de três horas para um *Sprint* de um mês (devendo ser reduzida proporcionalmente para durações menores), que ocorre após a reunião de *Sprint Review* e antes do *Sprint Planning*. No fim da reunião, o time deve ter identificado melhorias gerais a serem implementadas nas próxima iteração. Esta reunião é uma oportunidade de inspeção e adaptação para o time como um todo.

2.3.2.5 Os artefatos do *Scrum*

O *Scrum* define apenas dois artefatos: *Product Backlog* e *Sprint Backlog*. Por se tratar de um *framework* extensível, permite que cada projeto adote outros artefatos de acordo com a necessidade, e não define como seus artefatos devem

ser implementados, deixando o trabalho a cargo dos próprios times. Este capítulo apresenta a descrição dos artefatos definidos pelo *Scrum*, de acordo com o *Scrum Guide* (SCHWABER; SUTHERLAND, 2013). Os artefatos podem ser identificados na Figura 5.

Figura 5 – Visão geral do framework Scrum, com destaque para seus artefatos

Fonte: Adaptado de ScrumAlliance.org

O *Product Backlog* é uma lista ordenada de funcionalidades desejadas. O *Product Owner* é responsável pelo *Product Backlog*, incluindo seu conteúdo, disponibilidade, e ordenação. Pode ser representado de várias formas, como em um quadro físico e uma planilha Excel, mas o importante é que seja, de alguma forma, visível para todos os envolvidos. O *Product Backlog* nunca está completo, pois evolui com o produto, e é dinâmico, pois pode mudar constantemente de forma a garantir que o produto mantenha-se apropriado, competitivo e útil. São listadas todas as funcionalidades, requisitos e melhorias que constituem as mudanças a serem feitas em futuras entregas. Possui os atributos de descrição, ordem, estimativa e valor. Múltiplos times *Scrum* podem atuar num mesmo produto e, consequentemente, compartilhar um mesmo *Product Backlog*. Seu refinamento

consiste em adicionar detalhe, estimativa e ordem a seus itens. O *Product Owner* e o Time de Desenvolvimento precisam colaborar nesta atividade. O refinamento geralmente não consome mais do que 10% da capacidade do Time de Desenvolvimento. Contudo, os itens do *backlog* podem ser atualizados a qualquer momento pelo *Product Owner*. Os itens no topo da lista definem as funcionalidades com maior expectativa de realização. Desta forma, os itens no topo devem apresentar mais clareza e detalhes do que os itens do fim da lista. O Time de Desenvolvimento é responsável por todas as estimativas, e pode ter o apoio do *Product Owner* para melhor entendimento.

O *Sprint Backlog* define o escopo da iteração, estando diretamente ligado ao objetivo do *Sprint*. Este *backlog* é criado e mantido apenas pelo Time de Desenvolvimento, tendo por base as funcionalidades e requisitos a serem implementados na iteração, e precisa ser atualizado de forma a refletir o esforço de desenvolvimento utilizado para atender ao objetivo do *Sprint*. Esta atualização ocorre constantemente durante a iteração, não precisando se restringir ao momento da *Daily Meeting*. Novos esforços detectados são adicionados a este *backlog*, e conforme o trabalho for sendo completado, o trabalho restante estimado é atualizado. Quando identificam-se que elementos do *backlog* não são realmente necessários, os mesmos são removidos. Este *backlog* é um retrato do que o time pretende realizar no *Sprint*, e pertence apenas ao Time de Desenvolvimento. Nas reuniões diárias, o time monitora o progresso do *Sprint*, considerando o trabalho restante para que o objetivo do *Sprint* seja atingido. Ao fim da iteração, o Time de Desenvolvimento entrega um incremento de *software*, que é a soma de todos os itens do *Product Backlog* completados durante o último *Sprint,* bem como o acúmulo dos incrementos de todos os *Sprints* anteriores.

O Quadro 3 apresenta a relação entre eventos e artefatos do *Scrum*. O *Scrum* não limita os artefatos ao *Product Backlog* e o *Sprint Backlog*, apenas; por se

tratar de um *framework*, permite que novos artefatos sejam incorporados ao processo, dependendo da realidade e necessidade da empresa.

Quadro 3 - Relação entre eventos e artefatos do Scrum

Evento do Scrum	Artefatos	Ações
Sprint	*Sprint Backlog*	Execução das tarefas necessárias para cumprir o *Sprint Backlog*
Sprint Planning	*Product Backlog, Sprint Backlog*	Selecionam-se itens do *Product Backlog* para compor o *Sprint Backlog*, considerando prioridade e estimativa
Daily Scrum Meetings	*Sprint Backlog*	Discutem-se ações sobre tarefas que visam atender ao *Sprint Backlog*
Sprint Review	*Sprint Backlog*	Apresentam-se incrementos de *software* criados por meio de tarefas que visavam atender ao *Sprint Backlog*
Sprint Retrospective	--	Evento para melhoria de processo interno do time; não faz alusão a nenhum dos artefatos

Fonte: Elaborado pelo autor

2.3.3 Tailoring de processo de software

A norma ISO/IEC 12207 (ISO, 2008) define o processo de *tailoring* como aquele capaz de adaptar os processos atuais de ciclo de vida de *software* para satisfazer circunstâncias particulares ou fatores que: a) permeiam a organização que está aplicando o processo; b) influenciam um projeto que requer o uso do processo; c) refletem a necessidade da organização de suprir produtos ou serviços. Esta definição está de acordo com a visão de Dinsmore e Cabanis-Brewin (2006), que afirmam que o gerente de projetos é responsável por realizar o *tailoring* do processo de acordo com as necessidades do projeto e da organização. Espera-se, segundo esta norma ISO, que um processo de *tailoring* gere, como resultado de sua aplicação, processos de ciclo de vida modificados, para atingir os propósitos esperados na organização.

São cinco as atividades sugeridas pela norma ISO citada para que se realize um processo de *tailoring* de ciclo de vida de *software*:

1. Identificar e documentar as circunstâncias que influenciam *tailoring*, tais como (mas não restritas a): a) estabilidade de ambientes operacionais; b) riscos, tanto em termos comerciais quanto de performance, relacionados ao interesse das partes; c) Inovação, tamanho e complexidade; d) data de início e duração da utilização; e) questões de integridade, como segurança, privacidade, usabilidade, disponibilidade; f) oportunidade de tecnologias emergentes; g) orçamento e recursos organizacionais disponíveis; h) disponibilidade dos serviços de sistemas capacitados; i) regras e responsabilidades do ciclo de vida geral do sistema; j) necessidade de conformidade com outros padrões;
2. Em casos de aspectos críticos existentes no sistema, considerar as estruturas de ciclo de vida recomendadas ou obrigatórias por padrões relevantes à dimensão da criticalidade;
3. Obter retorno de todas as partes afetadas pelas decisões de *tailoring*, incluindo (mas nao limitado a) *stakeholders* do sistema, partes interessadas e funções organizacionais contribuidoras;
4. Realizar decisões de *tailoring* de acordo com o processo de gerenciamento de decisão da empresa;
5. Selecionar os processos de ciclo de vida que requerem *tailoring* e remover atividades, tarefas e artefatos de saída não relevantes.

Neste capítulo do trabalho, busca-se reunir literatura relacionada à aplicação de *tailoring* de modelos ágeis em empresas que adotam modelos *plan-driven*.

2.3.4 Tailoring do *Scrum*: análise de práticas de métodos ágeis

Neste capítulo, o assunto a ser considerado envolve tanto a análise de práticas que fazem parte do *Scrum* quanto de práticas que não são relacionadas com o núcleo do framework *Scrum,* mas que podem ser utilizadas para estendê-lo.

Petersen e Wohlin (2010) estudaram o caso de uma grande empresa (Ericsson, na Suécia) onde o processo mudou de *plan-driven* para iterativo-incremental, pelo uso de práticas ágeis. As etapas do processo estavam divididas em análise de requisitos, projeto e implementação, teste, entrega e manutenção. Este caso é similar ao desta dissertação pela característica de transição de *plan-driven* para processo ágil, mas diferente no sentido de que consegue englobar todos os setores de TI da empresa dentro do modelo incremental – o que não foi possível de ser feito nos estudo de caso deste trabalho. As práticas do processo estabelecido na Ericsson foram relacionadas com práticas de alguns processos, entre ágil e incremental, por meio de uma lista elaborada com base em informações providas por Larman (2003). Esta lista, conforme utilizada por Petersen e Wohlin (2010), é exibida no Quadro 4.

Quadro 4 - Comparação do processo na Ericsson com modelos de processos gerais (Continua)

Princípios	**ID**	**XP**	**SC**	**C**
Iterações e incrementos	X	X	X	X
Entregas (releases) internas e externas	X			X
Duração da iteração (Time boxing)	X	X	X	X
Sem mudança em projetos iniciados	X		X	X
Cliente alocado no time		X	X	
Frequente iteração face-a-face		X	X	X
Times auto-organizados		X	X	
Processo empírico		X	X	
Disciplina sustentável		X		
Backlog de produto flexível		X	X	X
Tomada de decisão rápida			X	
Integração frequente			X	X

Quadro 4 - Comparação do processo na Ericsson com modelos de processos gerais (Conclusão)

Princípios	ID	XP	SC	C
Simplicidade de design		X		
Refatoração		X		
Propriedade de código do time		X		

Fonte: Petersen e Wohlin (2010)

Legenda: C: práticas em uso pela companhia antes da adoção do modelo ágil;
ID: modelo iterativo de desenvolvimento; XP: *Exreme Programming*; SC: *Scrum*

Este quadro foi utilizado pelos pesquisadores para determinar o modelo ágil que melhor se adequaria à empresa Ericsson por conta da similaridade com o ambiente existente. Embora nesta dissertação o método já esteja determinado (*Scrum*), uma utilização similar deste quadro foi útil para identificar a aderência das práticas do *Scrum* ao ambiente de pesquisa, com foco nas dificuldades de adoção.

O Kanban é uma ferramenta de desenvolvimento ágil cujas práticas podem ser incorporadas pelo *Scrum*. Kniberg e Skarin (2009) apresentam uma relação entre os métodos ágeis *Scrum* e Kanban, no sentido de extrair o melhor de ambos (KNIBERG; SKARIN, 2009). Práticas do Kanban como o quadro físico e os conceitos de limitação de tamanho da fila (KNIBERG; SKARIN, 2009) são úteis na composição do *framework Scrum*, por serem uma forma eficaz de implementar valores tais como transparência, gestão à vista e priorização de atividades visando entrega eficaz.

Dean Leffingwell (2011) escreve sobre o uso de histórias de usuário em times ágeis, como sendo uma breve descrição de uma intenção que informa algo que o sistema precisa fazer para o usuário[1]. Histórias de usuário são uma prática originada no método *Extreme Programming* (BECK, 1999), e comumente utilizadas em times

[1] Neste trabalho e no contexto de métodos ágeis, o termo ‘requisito’ será empregado como algo semelhante a histórias de usuário.

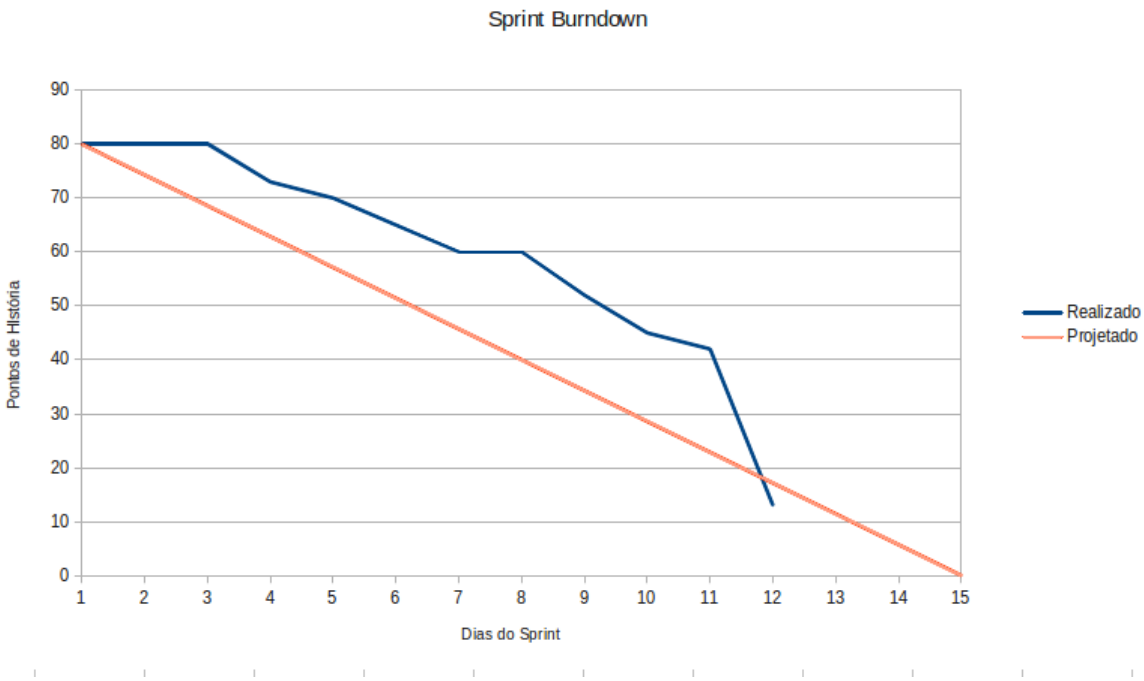
ágeis, dentre os quais aqueles que praticam *Scrum*. Segundo Leffingwell, o time ágil deve utilizar as histórias de usuário para ajudar a definição do escopo da iteração e servir de guia para a implementação. Portanto, por definirem o que deve ser feito, são utilizadas também em estimativas. As histórias de usuário podem ser utilizadas para compor o framework *Scrum*, porque o time necessita de um direcionamento em termos funcionais (o que o sistema deve fazer), e o conceito é adequado a este fim. Mike Cohn (2004) também escreve sobre histórias de usuário, detalhando como devem ser criadas, e sugerindo que a escrita das história deve ser elaborada de forma a ser Independente, Negociável, Agregadora de Valor, Estimável, Pequena e Testável.

Outro aspecto a ser considerado é o da estimativa, para fins de levantamento de quanto o time é capaz de produzir durante uma iteração. A estimativa comumente apóia-se em histórias de usuário. Como técnica de estimativa, métodos ágeis comumente fazem uso de *Planning Poker*. Esta técnica faz com que todos os desenvolvedores estimem uma dada história do usuário de forma independente, e então discutam suas estimativas, chegando assim, após algumas rodadas, a um denominador comum (HAUGEN, 2006). A pesquisa de Haugen (2006) concluiu que a estimativa de histórias de usuário pela prática de *Planning Poker* é melhor do que uma estimativa não estruturada, quando existe experiência prévia na tecnologia e funcionalidade envolvidas. A técnica de *Planning Poker* pode ser utilizada para compor o *framework Scrum*.

De forma a apresentar visibilidade da produtividade do time durante a iteração, times *Scrum* comumente fazem uso do gráfico de *Burndown*. Este gráfico permite acompanhar o avanço do time diariamente, em relação a uma linha de tendência projetada de forma linear com base na estimativa do time sobre o período da iteração. Segundo Sutherland (2001), um dos criadores do *Scrum*, os projetos que fazem uso de *Scrum* devem usar técnicas leves como gráfico de *Burndown* em detrimento de gráficos de *Gantt,* mais formais, porque o gráfico de *Gantt* não é bom

em cenários como o de desenvolvimento ágil, no qual o caminho crítico se torna obsoleto diariamente, por conta das mudanças constantes. Um gráfico de *burndown* de exemplo pode ser encontrado na Figura 6.

Figura 6 – Gráfico de Burndown



Fonte: Elaborado pelo autor

Neste exemplo, a iteração foi estimada com 80 pontos de história. A linha mais clara (projetado) distribui estes pontos em queda linear proporcional considerando a duração da iteração (de quinze dias, neste exemplo). A linha mais escura (realizado) é atualizada diariamente pelo time para demonstrar a evolução em termos de histórias realizadas. A linha mais escura nunca sobe no gráfico, mas pode-se manter estacionária (entre os dias 1 e 3, e 7 e 8, no exemplo), o que indica que não houve finalização de histórias nos dias em questão, e pode estar abaixo da linha projetada (o que indica que a entrega das histórias está sendo mais rápida do que a estimativa).

Outro método ágil bastante conhecido é o XP (*Extreme Programming*) (BECK, 1999), um processo eficaz que resulta em produtos de alta qualidade. Svensson e Host (2005) apresentam relatos de adoção de XP em grande empresa. Na pesquisa dos autores, observou-se que a complexidade da cultura organizacional exigiu que várias práticas tivessem de ser adaptadas. Para cada uma das práticas do XP, o trabalho relatou como a prática foi incorporada e executada pelo time, como a prática foi incorporada ao processo ágil e como o processo ágil gerado diferiu do processo XP original. Muitas destas práticas isoladas (como Programação em par, Teste Contínuo, Refatoração, dentre outras) são encontradas em muitos ambientes de construção de *software,* incluindo aqueles que fazem uso de *Scrum*. Este é um caso de abordagem localizada, no qual a empresa apresentou preocupação com o risco em fazer grandes mudanças em processo, e um caso no qual o projeto precisa iteragir com várias instâncias da organização. Pode-se identificar as práticas do XP que poderiam compor o *framework Scrum* por meio de uma análise em termos do que seria o ideal (de acordo com o XP) e de como a prática poderia ser adotada no ambiente de interesse para compor o framework *Scrum,* considerando a realidade da empresa. O Quadro 5 apresenta as doze práticas básicas do XP a serem consideradas.

Quadro 5 - As doze práticas básicas do XP

Prática do XP	Descrição
Jogo de Planejamento	Rapidamente determinar o escopo da próxima entrega, combinando prioridades de negócio e estimativas técnicas. O cliente decide escopo, prioridade e datas de uma perspectiva de negócio, e as pessoas técnicas estimam e controlam o progresso
Pequenas entregas	Colocar um sistema simples em produção rapidamente. Entregar novas versões em um ciclo mito curto (duas semanas)
Metáfora	Guia todo o desenvolvimento com uma história simples e compartinhada de como o sistema funciona de uma forma geral
Design simples	Realize o design da forma mais simples possível
Teste	Desenvolvedores continuamente escrevem testes unitários que devem rodar livres de falhas; clientes escrevem testes para demonstrar que funções estão finalizadas. “Teste, então codifique” significa que um caso de teste que falha é o critério de entrada para escrever código
Refatoração	Reestruturar o sistema sem mudar seu comportamento, para remover duplicação, melhorar comunicação, simplificar, ou adicionar flexibilidade
Programação em par	Todo código de produção é escrito por dois programadores em uma máquina
Propriedade coletiva	Qualquer um pode melhorar o código do sistema a qualquer momento
Integração contínua	Integrar e construir (build) o sistema muitas vezes por dia (a cada vez que uma tarefa foi finalizada). Testes contínuos de regressão evitam regressões de funcionalidade quando requisitos mudam
40-horas por semana	Trabalhe, na medida do possível, não mais do que quarenta horas por semana; nunca faça horas extras por duas semanas seguidas
Cliente no local	Tenha um usuário real no time disponível em tempo total para responder questões
Padrões de codificação	Tenha regras que reforcem comunicação por meio do código

Fonte: Paulk (2001)

2.3.5 Tailoring para adoção do *Scrum* em estruturas não-projetizadas

Kerzner (2009) sugere que organizações podem ser definidas como grupos de pessoas que coordenam suas atividades para atingir objetivos. Nesta visão, as estruturas organizacionais são estabelecidas para garantir que pessoas trabalhem da melhor forma.

Infere-se que o aspecto do *Scrum* mais diretamente relacionado às estruturas organizacionais diz respeito às pessoas – mais especificamente, à disponibilidade de pessoas nos times. *Scrum* espera que os times sejam auto-organizados e multidisciplinares, formados por pessoas atuando juntas na realização de projetos

pelo máximo de tempo possível. A incapacidade de oferecer estas premissas causa efeitos colaterais na organização do time, que visa executar as atividades de construção de *software* juntamente com outras atividades de desenvolvimento, como documentação e *design* – o que é a principal característica de distinção em relação aos modelos *plan-driven*.

Times auto-organizados e multifuncionais, conforme preza o *Scrum*, são melhor atendidos em uma estrutura organizacional projetizada (onde há a devida autoridade do gerente de projetos para formar times ideais), e precisam ser mais fortemente adaptados por *tailoring* nas demais estruturas organizacionais (matriciais e funcionais, onde há menor autoridade).

Portanto, o *tailoring* voltado a estruturas não-projetizadas procura adaptar o *Scrum* para a realidade na qual os times de projetos precisarão compartilhar recursos com as gerências funcionais, e podem até mesmo não tornarem-se multidisciplinares, em cenários onde não seja possível trazer recursos de determinadas gerências para os times.

2.3.6 Tailoring para adoção do *Scrum* em organizações *plan-driven*

O assunto deste capítulo diz respeito a como pode-se fazer uso de um modelo para adoção de *Scrum* em empresas organizadas de forma *plan-driven*. Há o interesse em saber como o *Scrum* pode ser adotado num cenário onde a empresa assume uma filosofia de processo oposta, mais tradicional. Busca-se um modelo para implantação, ao mesmo tempo que espera-se que ele possa ser implementado sem ocasionar mudanças na cultura organizacional.

A relação oposta entre métodos ágeis e *plan-driven* sugere sérias diferenças na forma como pessoas de diferentes papeis atuam no desenvolvimento de *software*, que, segundo Nerur, Mahapatra e Mangalaraj (2005), estão relacionadas

aos seguintes tópicos: *a) controle:* no modelo *plan-driven*, o controle é centrado em processos e no ágil é centrado em pessoas; *b) estilo de gerenciamento:* comando-e-controle no modelo *plan-driven*, e liderança-e-colaborativo no modelo ágil; c) gerenciamento de conhecimento: explícito no modelo *plan-driven*, e tácito no modelo ágil; d) atribuição de papeis: no modelo *plan-driven*, considera especializações individuais, e no modelo ágil, considera times auto-organizados e intercâmbio; e) comunicação: formal no modelo *plan-driven*, e informal no modelo ágil; f) papel do cliente: é importante no modelo *plan-driven*, e crítico no modelo ágil; g) ciclo do projeto: no modelo *plan-driven*, é guiado por tarefas e atividades, e no modelo ágil, é guiado por funcionalidades do produto; h) modelo de desenvolvimento: no modelo *plan-driven*, faz uso de *waterfall*, modelo espiral, ou variações, e no modelo ágil, faz uso de modelos evolucionários focados em entrega, como o *Scrum*; i) estrutura organizacional desejada: no modelo *plan-driven*, é mecanizada (burocrática com alta formalização), e no modelo ágil, é orgânica (flexível e participativa, encorajando cooperação social); j) tecnologia: no modelo *plan-driven*, não informa restrições, e no modelo ágil, favorece tecnologias orientadas a objeto (NERUR; MAHAPATRA; MANGALARAJ, 2005).

Portanto, tal conjunto numeroso de diferenças sugere que a transição de um modelo *plan-driven* para um ágil não é um trabalho trivial, sujeito à consideração de muitos fatores e variáveis. Segundo Nikitina e Kajko-mattsson (2014), empresas com organizacão *plan-driven* encontram problemas na adoção de métodos ágeis especialmente por não haver um método claramente definido para a adoção, ao mesmo tempo que sustentam que pesquisas relacionadas à adoção de métodos ágeis em empresas são limitadas.

2.4 Conclusão

Este capítulo apresentou os principais conceitos que embasam a pesquisa de estudo de caso desta dissertação. Foram analisadas literaturas relacionadas a

processos de *software* (juntamente com o conceito de processo *plan-driven*), projetos e gestão de projetos, estruturas organizacionais não-projetizadas, métodos ágeis (com ênfase em *Scrum*), bem como o conceito de customização de processos (*tailoring*).

Com base nestes estudos, elaborou-se um referencial teórico para pesquisa, onde os principais elementos componentes do método *Scrum* foram analisados à luz das dificuldades em sua adoção, considerando a estrutura não-projetizada e o processo *plan-driven*. Destas dificuldades foram derivadas questões para composição de um questionário fechado (apresentado no Anexo D deste documento). O referencial teórico é apresentado no Quadro 6.

Quadro 6 - Temas relevantes ao Scrum considerando estrutura e processo (Continua)

Tema relevante	Dificuldades selecionadas para observação	Questão
Pilar Transparencia em estrutura não-projetizada	Em *Scrum*, transparência está relacionada a aspectos significativos do método que devem estar visíveis aos responsáveis pelos resultados. (SCHWABER; SUTHERLAND, 2013). Numa estrutura não-projetizada, os departamentos possuem diferentes visões do projeto (PMI, 2013), que nem sempre são compartilhadas entre todos os departamentos envolvidos. A aplicação da transparência pode ser dificultada em estruturas não-projetizadas pelo fato de pessoas estarem atuando paralelamente em tarefas do time *Scrum* e de seu próprio departamento, e de não haver transparência dos envolvidos no projeto neste sentido.	Minha empresa tem dificuldades em deixar claros os papeis e responsabilidades das pessoas no cenário onde seu trabalho é dividido entre atuar no time *Scrum* e em seu próprio departamento.
Pilar Inspeção em estrutura não-projetizada	Em *Scrum*, o time deve inspecionar seu próprio método, mas de forma que a inspeção não seja tão frequente a ponto de atrapalhar a execução das tarefas (SCHWABER; SUTHERLAND, 2013). Numa estrutura não-projetizada, os profissionais são divididos em departamentos (PMI, 2013). A inspeção pode ser dificultada em estrutura não-projetizada se o método *Scrum* não for inspecionado de forma a melhorar o próprio método, mas sim de forma que cada membro considere melhorias visando apenas características relacionadas ao seu departamento de origem. Por exemplo, um analista de sistema sugere apenas melhorias relacionadas ao departamento de análise, ao invés de considerar melhorias que sejam boas a todo o time.	As pessoas do time *Scrum* tendem a inspecionar seu processo considerando mais o ponto de vista do seu departamento original do que o ponto de vista do time como um todo.
Pilar Inspeção em processo *plan-driven*	Em *Scrum*, o time deve inspecionar seu próprio método, mas de forma que a inspeção não seja tão frequente a ponto de atrapalhar a execução das tarefas (SCHWABER; SUTHERLAND, 2013). Em um processo *plan-driven*, as tarefas de análise de requisitos devem estar completamente definidas para que só então a próxima etapa do desenvolvimento do *software* possa prosseguir (PETERSEN; WOHLIN, 2010). A inspeção do método *Scrum* pode ser comprometida se as pessoas do time *Scrum* organizarem suas tarefas de forma linear, de acordo com a perspectiva *plan-driven*, quando as tarefas podem ser realizadas em paralelo.	As pessoas do time *Scrum* tendem a inspecionar o processo visando uma organização das atividades onde as tarefas de análise devem preceder as tarefas de construção, que por sua vez devem preceder as tarefas de teste.
Pilar Adaptação em estrutura não-projetizada	Em *Scrum*, o pilar Adaptação aponta que caso um ou mais aspectos do método *Scrum* sendo adotado sejam desviados para fora de limites aceitáveis, o método precisa ser ajustado. Esse ajuste pode ocorrer a qualquer momento na iteração (SCHWABER; SUTHERLAND, 2013). Numa estrutura não-projetizada, pessoas são dispostas em diferentes departamentos, com objetivos e interesses próprios (PMI, 2013). A adaptação pode ser comprometida caso o time não realize melhorias no método *Scrum* por valorizá-lo menos do que os processos de seus próprios departamentos de origem.	Minha empresa tem dificuldade em adotar o papel de *Scrum Master* porque este papel não existe formalmente em nenhum departamento de TI.
Papel de *Scrum Master* em estrutura não-projetizada	Em *Scrum*, o *Scrum Master* é responsável por garantir que o *Scrum* seja adotado corretamente na organização, tanto dentro quanto fora dos times (SCHWABER; SUTHERLAND, 2013). Numa estrutura não-projetizada, as pessoas são divididas em departamentos com especialidades distintas (PMI, 2013). O papel de *Scrum Master* pode ser comprometido numa estrutura não-projetizada num cenário onde a empresa não saiba em qual departamento o *Scrum Master* deve estar associado, e por isto não valorize o papel.	Minha empresa tem dificuldade em adotar o papel de *Scrum Master* porque este papel não existe formalmente em nenhum departamento de TI.
Papel de *Scrum Master* em processo *plan-driven*	Em *Scrum*, o *Scrum Master* é responsável por garantir que o *Scrum* seja adotado corretamente na organização, tanto dentro quanto fora dos times (SCHWABER; SUTHERLAND, 2013). Num processo *plan-driven*, as atividades de uma disciplina precisam estar completamente fechadas e formalizadas para que as próximas atividades possam iniciar (PETERSEN; WOHLIN, 2010) (HIRSCH, 2005) (BOEHM; TURNER, 2003) (WAARDENBURG; VLIET, 2013). O papel de *Scrum Master* pode ser comprometido num processo *plan-driven* em cenários onde outras disciplinas de fora do time não ofereçam apoio aos times *Scrum* em termos de sincronização ordenada das tarefas.	O *Scrum Master* da minha empresa tem dificuldades no cumprimento do processo Scrum por conta da falta de apoio de outros departamentos, que nem sempre respeitam os limites e regras do processo *Scrum*.

Quadro 6 -Temas relevantes ao Scrum considerando estrutura e processo (Continua)

Tema relevante	Dificuldades selecionadas para observação	Questão
Papel de *Product Owner* em estrutura não-projetizada	Em *Scrum*, o Product Owner é um membro do time *Scrum* responsável por prover informações do negócio para o time de desenvolvimento (SCHWABER; SUTHERLAND, 2013). Numa estrutura não-projetizada, as pessoas são divididas em departamentos com especialidades distintas (PMI, 2013). O papel de *Product Owner* pode ser dificultado numa estrutura não-projetizada num cenário onde a empresa não saiba em qual departamento o *Product Owner* deve estar associado, e por isto não valorize o papel.	Minha empresa tem dificuldades em adotar o papel de *Product Owner* porque este papel não existe formalmente em nenhum departamento.
Papel de *Product Owner* em processo *plan-driven*	Em *Scrum*, o Product Owner é um membro do time *Scrum* responsável por prover informações do negócio para o time de desenvolvimento (SCHWABER; SUTHERLAND, 2013). Num processo *plan-driven*, as atividades de uma disciplina (ou departamento) precisam ser completamente formalizadas para que só então a próxima disciplina (ou departamento dependente) possa iniciar suas tarefas (PETERSEN; WOHLIN, 2010) (HIRSCH, 2005) (BOEHM; TURNER, 2003) (WAARDENBURG; VLIET, 2013). O papel de *Product Owner* pode ser dificultado num processo *plan-driven* em cenários onde o profissional não consiga atuar em tarefas com o time de desenvolvimento quando o time necessita, pois suas atividades podem não estar completamente formalizadas antes que o time inicie a iteração.	Minha empresa não consegue fazer com que o *Product Owner* esteja envolvido com o time de desenvolvimento nos momentos em que a interação faz-se necessária.
Papel do time de desenvolvimento em estrutura não-projetizada	Em *Scrum*, o time de desenvolvimento consiste de profissionais que realizam o trabalho de entregar uma versão usável que potencialmente incrementa o produto ao final de cada iteração (SCHWABER; SUTHERLAND, 2013). Numa estrutura não-projetizada, os profissionais são organizados em departamentos distintos, com objetivos diferentes (PMI, 2013). O papel do time de desenvolvimento pode ser dificultado numa estrutura não-projetizada em cenários onde o time tenha dificuldades de trabalhar junto na iteração, por conta de interrupções para realização de tarefas emergenciais do departamento de origem que não têm relação com o objetivo da iteração.	Minha empresa tem dificuldades em manter o time de desenvolvimento trabalhando junto na iteração, pois membros do time precisam também trabalhar em tarefas de seus departamentos de origem.
Papel do time de desenvolvimento em processo *plan-driven*	Em *Scrum*, o time de desenvolvimento consiste de profissionais que compõem um time multidisciplinar, com competência para realizar todas as tarefas necessárias para a entrega de uma versão que potencialmente incrementa o produto ao final de cada iteração (SCHWABER; SUTHERLAND, 2013). Num processo *plan-driven*, as atividades de uma disciplina (ou departamento) precisam ser completamente formalizadas para que só então a próxima disciplina (ou departamento dependente) possa iniciar suas tarefas (PETERSEN; WOHLIN, 2010) (HIRSCH, 2005) (BOEHM; TURNER, 2003) (WAARDENBURG; VLIET, 2013). O papel do time de desenvolvimento pode ser dificultado em um processo *plan-driven* em cenários onde o time de desenvolvimento não consiga realizar as tarefas planejadas da iteração de forma eficiente, sendo forçado a realizar as tarefas da iteração de forma sequencial, em cascata.	O time de desenvolvimento entende que as tarefas do time multidisciplinar não podem ser feitas em paralelo, mas sim devem ser feitas em cascata, iniciando pelos requisitos, para que depois sejam feitas as tarefas de construção e só então as tarefas de teste.
Reunião de planejamento em estrutura não-projetizada	Em *Scrum*, a reunião de planejamento é um rito que define o que poderá ser entregue na iteração e como o trabalho para isto será realizado (SCHWABER; SUTHERLAND, 2013). Numa estrutura não-projetizada, os profissionais são organizados em departamentos distintos, com objetivos diferentes (PMI, 2013). A reunião de planejamento pode ser dificultada em uma estrutura não-projetizada em cenários onde as pessoas não conseguem comparecer à reunião por conta de outras prioridades de seus departamentos de origem, o que causa problemas ao correto início da iteração.	Minha empresa não consegue reunir todas as pessoas necessárias nas reuniões de planejamento (*Spring Planning*) por conta de compromissos das pessoas com tarefas de seus departamentos de origem.

Quadro 6 -Temas relevantes ao Scrum considerando estrutura e processo (Continua)

Tema relevante	Dificuldades selecionadas para observação	Questão
Reunião de planejamento em processo *plan-driven*	Em *Scrum*, a reunião de planejamento é um rito que define o que poderá ser entregue na iteração e como o trabalho para isto será realizado (SCHWABER; SUTHERLAND, 2013). Um processo plan-driven apenas inicia uma atividade quando está completamente formalizada, mesmo que já tenha sido suficientemente entendida (PETERSEN; WOHLIN, 2010) (HIRSCH, 2005) (BOEHM; TURNER, 2003) (WAARDENBURG; VLIET, 2013). A reunião de planejamento pode ser dificultada em um processo *plan-driven* em cenários onde a reunião não possa ser realizada a não ser que os requisitos estejam completamente fechados e formalizados.	Minha empresa só realiza a reunião de planejamento (*Spring Planning*) quando o entendimento dos requisitos é total e está devidamente formalizado.
Reunião diária em estrutura não-projetizada	Em *Scrum*, a reunião diária é o evento onde o time inspeciona seu trabalho realizado nas últimas vinte e quatro horas, e prevê o trabalho que será realizado até a próxima reunião diária (SCHWABER; SUTHERLAND, 2013). Uma estrutura não-projetizada tem pessoas dispostas em diferentes departamentos, com diferentes objetivos e prioridades (PMI, 2013). A reunião diária pode ser dificultada em uma estrutura não-projetizada em cenários onde os profissionais não conseguem comparecer à reunião por conta de estarem atuando em atividades de seus departamentos de origem.	Os profissionais do time Scrum não conseguem comparecer às reuniões diárias (*Scrum Daily Meetings*) por terem conflito de agenda por conta de compromissos em seus próprios departamentos.
Reunião diária em processo *plan-driven*	Em *Scrum*, a reunião diária é o evento onde o time inspeciona seu trabalho realizado nas últimas vinte e quatro horas, e prevê o trabalho que será realizado até a próxima reunião diária (SCHWABER; SUTHERLAND, 2013). No processo *plan-driven*, as atividades de um tema precisam estar completamente fechadas para que as próximas atividades possam ser iniciadas (PETERSEN; WOHLIN, 2010) (HIRSCH, 2005) (BOEHM; TURNER, 2003) (WAARDENBURG; VLIET, 2013). A reunião diária pode ser dificultada em um processo *plan-driven* em cenários onde o time não veja utilidade na realização da reunião, pois muitas atividades (como análise de requisitos e testes) são feitas fora do time, antes e após a atividade de construção pelo time *Scrum*.	Os profissionais do time *Scrum* não participam das reuniões diárias (*Scrum Daily Meetings*) porque não vêem utilidade, já que as tarefas dos analistas já foram feitas antes do *Sprint*, ou as tarefas dos testadores só irão ocorrer após o *Sprint*.
Reunião de revisão em estrutura não-projetizada	Em *Scrum*, a reunião de revisão é a oportunidade para que, ao final da iteração, o time e as partes interessadas colaborem sobre o trabalho realizado na iteração (SCHWABER; SUTHERLAND, 2013). Uma estrutura não-projetizada tem pessoas dispostas em diferentes departamentos, com diferentes objetivos e prioridades (PMI, 2013). A reunião de revisão pode ser dificultada em uma estrutura não-projetizada em cenários onde não existe um departamento ou cliente designado como responsável pelo produto, ou este responsável não demonstra interesse no produto sendo construído.	O time *Scrum* não realiza reunião de revisão (*Spring Review*) pois não há interesse do cliente ou há desconhecimento sobre quem é o cliente ou departamento representante do cliente para o produto sendo desenvolvido.
Reunião de revisão em processo *plan-driven*	Em *Scrum*, a reunião de revisão é a oportunidade para que, ao final da iteração, o time e as partes interessadas colaborem sobre o trabalho realizado no período (SCHWABER; SUTHERLAND, 2013). Um processo *plan-driven* apenas inicia uma atividade quando a atividade anterior está completamente formalizada, mesmo que já tenha sido suficientemente entendida (PETERSEN; WOHLIN, 2010) (HIRSCH, 2005) (BOEHM; TURNER, 2003) (WAARDENBURG; VLIET, 2013). A reuniao de revisão pode ser dificuldada num processo *plan-driven* em cenários onde o cliente só é envolvido quando todo o produto está finalizado, e não de forma incremental.	O time *Scrum* não realiza reunião de revisão (*Spring Review*) com o cliente, mas sim com um outro departamento intermediário, que recebe as demandas.

Quadro 6 -Temas relevantes ao Scrum considerando estrutura e processo (Continua)

Tema relevante	Dificuldades selecionadas para observação	Questão
Reunião de retrospectiva em estrutura não-projetizada	Em *Scrum*, a reunião de retrospectiva é uma oportunidade para o time *Scrum* inspecionar a si próprio e criar um plano para melhorias a serem aplicadas na próxima iteração (SCHWABER; SUTHERLAND, 2013). Uma estrutura não-projetizada tem pessoas dispostas em diferentes departamentos, com diferentes objetivos e prioridades (PMI, 2013). A reunião de retrospectiva pode ser dificultada em uma estrutura não-projetizada em cenários onde não existe muito interesse do time em melhorar o método *Scrum*, pois as atividades do departamento de origem tem maior peso e importância.	O time *Scrum* não realiza reunião de retrospectiva (*Spring Retrospective*) porque não há muito interesse na melhoria do processo ágil, já que as pessoas do time têm processos diferentes em seus próprios departamentos.
Reunião de retrospectiva em processo *plan-driven*	Em *Scrum*, a reunião de retrospectiva é uma oportunidade para o time *Scrum* inspecionar a si próprio e criar um plano para melhorias a serem aplicadas na próxima iteração (SCHWABER; SUTHERLAND, 2013). No processo *plan-driven*, as atividades de um tema precisam estar completamente fechadas para que as próximas atividades de outro tema possam ser então iniciadas (PETERSEN; WOHLIN, 2010) (HIRSCH, 2005) (BOEHM; TURNER, 2003) (WAARDENBURG; VLIET, 2013). A reunião de retrospectiva pode ser dificultada num processo *plan-driven* em cenários onde o time incorpora a mesma fragmentação de um processo plan-driven, separando as melhorias do método *Scrum* por interesses individuais separados, e não considerando ao melhoria do método para todos os envolvidos.	O time *Scrum* tem dificuldades na realização de uma boa reunião de retrospectiva (*Sprint Retrospective*) porque as melhorias no processo são discutidas de forma fragmentada visando interesses de diferentes áreas, e não visam a melhoria do processo para o time como um todo.
Backlog/requisitos em estrutura não-projetizada	Em Scrum, o *Backlog* do produto é uma lista ordenada de tudo o que deve ser necessário ao produto, e é uma origem única de requisitos para qualquer mudança a ser feita no produto (SCHWABER; SUTHERLAND, 2013). Uma estrutura não-projetizada tem pessoas dispostas em diferentes departamentos, com diferentes objetivos e prioridades (PMI, 2013). O *backlog* pode ser dificultado em uma estrutura não-projetizada em cenários onde o *backlog* é de responsabilidade de um departamento diferente daquela onde reside o time *Scrum.*	Minha empresa tem um departamento/pessoa responsável por requisitos/*backlog*, não relacionado com o time *Scrum*.
Backlog/requisitos em processo plan-driven	Em *Scrum*, o *Backlog* do produto é uma lista ordenada de tudo o que deve ser necessário ao produto, e é uma origem única de requisitos para qualquer mudança a ser feita no produto (SCHWABER; SUTHERLAND, 2013). No processo *plan-driven*, as atividades de um tema precisam estar completamente fechadas para que as próximas atividades de outro tema possam ser então iniciadas (PETERSEN; WOHLIN, 2010) (HIRSCH, 2005) (BOEHM; TURNER, 2003) (WAARDENBURG; VLIET, 2013). O *backlog* pode ser dificultado em um processo *plan-driven* em cenários onde todos os requisitos do backlog precisem ser totalmente definidos e formalizados para que só então o time possa iniciar o desenvolvimento.	Minha empresa detalha todos os requisitos/itens do *backlog* para que só então o time de desenvolvimento possa começar a desenvolver.
Timebox em estrutura não-projetizada	Em Scrum, o timebox equivale aos limites de duração de uma iteração, onde o time desenvolve uma versão incremental potencialmente utilizável do produto. O timebox das iterações deve ser coerente durante todo o esforço de desenvolvimento (SCHWABER; SUTHERLAND, 2013). Uma estrutura não-projetizada tem pessoas dispostas em diferentes departamentos, com diferentes objetivos e prioridades (PMI, 2013). O timebox é dificultado em uma estrutura não-projetizada em cenários onde há muita interrupção nas iterações dos times, para atender a esforços de fora do time *Scrum*.	Minha empresa tem dificuldade em manter um *timebox* no *Sprint*, pois membros do time constantemente são interrompidos para resolver problemas de seus departamentos de origem.

Fonte: próprio autor

3 METODOLOGIA DE PESQUISA

Este capítulo apresenta a metodologia de pesquisa aplicada na dissertação. Primeiramente, é justificada a escolha do estudo de caso múltiplo como estratégia de pesquisa. Em seguida, são descritas a seleção, coleta, análise e tratamento dos dados.

3.1 Estratégia de pesquisa

Para determinar a estratégia de pesquisa, faz-se relevante relembrar a pergunta de pesquisa previamente formulada. Conforme definida na introdução, a pergunta de pesquisa desta dissertação é: “como empresas brasileiras não-projetizadas e que desenvolvem software com abordagem *plan-driven* têm percebido dificuldades ao adotar o método *Scrum*?”

Para que fosse possível determinar um método de pesquisa eficaz para responder a esta pergunta, considerou-se a relação entre métodos e situações relevantes conforme proposto por Yin (2013), apresentada no Quadro 7.

Quadro 7 - Situações relevantes para diferentes métodos de pesquisa

Método	(1) Questões de forma de pesquisa	(2) Requer controle de eventos comportamentais?	(3) Enfoque em eventos contemporâneos?
Experimento	Como, por que?	Sim	Sim
Levantamento (*survey*)	Quem, o quê, onde, quantos, quanto?	Não	Sim
Análise de arquivos	Quem, o quê, onde, quantos, quanto?	Não	Sim/Não
Pesquisa histórica	Como, por que?	Não	Não
Estudo de caso	Como, por que?	Não	Sim

Fonte: Yin (2013)

Este trabalho procurou entender “como” (YIN, 2013) empresas brasileiras não-projetizadas e que desenvolvem software com abordagem *plan-driven* têm percebido dificuldades ao adotar o método *Scrum.* Na definição de Yin (2013), questões de pesquisa relacionadas com o “como” podem ser categorizadas de três formas: como experimento, pesquisa histórica ou estudo de caso.

Observou-se que este trabalho:

- Não se tratou de um experimento, pois não lidou com eventos naturais e não permitiu que o investigador pudesse manipular comportamentos sistematicamente (isto não ocorre em ambientes dinâmicos de desenvolvimento de *software)*;
- Não se tratou de um evento histórico, pois o interesse do trabalho girou em torno de entender a forma como empresas estavam lidando com a adoção *Scrum* no momento presente, e não em momentos anteriores;
- Tratou-se, portanto, de uma pesquisa de estudo de caso, capaz de examinar eventos contemporâneos sem manipular os eventos comportamentais. O estudo de caso permite entender um caso do mundo real, considerando que este entendimento é influenciado por condições de contexto (YIN, 2013).

Optou-se por um estudo de caso múltiplo pela desconfiança de que apenas uma verificação não daria a informação necessária. Este motivo é considerado válido por Yin (2013), que afirma que estudos de caso múltiplos são geralmente preferíveis a estudos de caso únicos pelo fato de que os benefícios em haver dois ou mais casos são substanciais em relação a apenas um caso.

Quanto ao objetivo de pesquisa, Yin (2013) categoriza estudos de caso em três categorias principais: a) exploratória: visa realizar investigações prévias, sem

que hajam informações completas sobre o assunto, de forma a identificar novas ideias, questões e hipóteses sobre os fenômenos; b) descritiva: visa narrar ou descrever fenômenos de forma mais precisa; c) explanatória: provê explicação causal de fenômenos conhecidos. Esta pesquisa teve caráter descritivo (YIN, 2013), pois analisou os detalhes dos processos das empresas em relação às dificuldades na adoção de *Scrum*, sem procurar explicar relações de causa e efeito.

Os dados empíricos desta pesquisa foram obtidos por meio de uma abordagem qualitativa, onde foram realizados estudos de caso múltiplos utilizando roteiros de entrevistas semi-estruturados, durante o período de cinco semanas.

Desta forma, a estratégia desta pesquisa consistiu na realização de um estudo de caso múltiplo, objetivando uma pesquisa descritiva e considerando uma abordagem qualitativa de coleta e análise dos dados.

Segundo Eisenhardt (1989), a estrutura da pesquisa é composta pelos seguintes passos:

- Início: onde é definida a questão de pesquisa e constructos são inicialmente definidos;
- Seleção de casos: onde os estudos de caso a serem considerados na pesquisa são devidamente selecionados;
- Elaboração de instrumentos e protocolo: onde são definidos métodos e instrumentos de coleta de dados, sejam de forma quantitativa ou qualitativa;
- Entrada em campo: onde a pesquisa é devidamente realizada em campo, e os instrumentos e métodos de coleta de dados são ajustados pela sobreposição de análise de dados com a coleta de dados, aliado à anotações de campo;

- Análise de dados: onde os dados coletados são devidamente analisados para o caso em questão, e padrões entre casos são procurados e identificados como evidências;
- Formulação de hipóteses: onde constructos são validados e refinados pelo uso de tabulação iterativa de evidências, lógica de replicação e identificação de evidências do “porquê” de relacionamentos;
- Comparação com literatura: onde conceitos emergentes, teorias ou hipóteses são comparados com a literatura, de forma a identificar aspectos similares, aspectos contraditórios, e o porquê das similaridades e contradições;
- Fechamento: onde as análises são finalizadas pela interrupção da adição de novos casos e de iteração entre teoria e dados.

Com exceção da Formulação de Hipóteses (não aplicável a esta pesquisa), os demais itens foram considerados. Alguns dos itens acima mencionados fazem uso de constructos, que não se aplicam neste trabalho pois não há busca quantitativa. Deve-se traduzir, neste trabalho, o termo *constructos* por *dificuldades* no contexto de adoção de *Scrum*.

3.2 Seleção dos casos

A seleção dos casos é um importante aspecto dos estudos de caso. Um conceito crucial é o da população, que determina o conjunto de elementos que será considerado como amostras de pesquisa, ao mesmo tempo que ajuda a definir os limites para generalização dos resultados, apoiando a validade externa (EISENHARDT, 1989). Também considera-se que, em pesquisas de estudos de caso, faz-se uso de amostragem teórica, no sentido em que os casos são escolhidos com base em motivos teóricos, e não estatísticos (EISENHARDT, 1989).

Com base nestas premissas, a definição da população deste trabalho seguiu o roteiro abaixo descrito:

- Foi elaborada uma lista de contatos atuantes na adoção de métodos ágeis em empresas, tanto por meio de experiências profissionais prévias com o pesquisador quanto por meio do contato com a comunidade;
- Cada um destes contatos participou de uma entrevista que procurou identificar se atuavam em empresa(s) adequadas(s) para os critérios estabelecidos nesta pesquisa. Um contato, se atuando sob regime de consultoria, poderia estar relacionado a mais de uma empresa;
- Os contatos e empresas identificadas como elegíveis foram selecionados como população válida para a pesquisa; os não-elegíveis foram descartados. Os critérios de seleção adotados foram escolhidos de acordo com os principais aspectos em discussão neste trabalho:
  - empresas utilizadoras de métodos *plan-driven;*
  - empresas organizadas com estrutura não-projetizada;
  - empresas interessadas na utilização de *Scrum* e que estejam encontrando dificuldades;

Como resultado, das cinco empresas pesquisadas, três foram selecionadas para pesquisa, aqui denominadas Empresa A, Empresa B e Empresa C.

A análise de casos de três empresas diferentes configura esta como sendo uma pesquisa de estudo de casos múltiplos, que permitem replicação literal entre os casos (YIN, 2013). A replicação literal aumenta a confiança na validade da pesquisa, por meio do maior grau de confirmação entre relacionamento dos constructos e as evidências obtidas (EISENHARDT, 1989).

3.3 Técnica de coleta de dados

De acordo com Yin (2013), evidências de estudo de caso podem vir de diversas fontes, dentre as quais as seis principais são: documentação, registros em arquivo, entrevistas, observação direta, observação como participante e artefatos físicos, com cada fonte sendo associada a uma lista de dados ou evidências. Nesta pesquisa, foram adotadas as fontes de documentação e entrevistas, por terem parecido mais aplicáveis à coleta de dados considerando: i) o interesse em saber como o processo *Scrum* era adotado nas empresas (entrevistas semi-estruturadas e documentação); ii) o entendimento da percepção das dificuldades (questionário).

A documentação foi, na medida do possível, analisada *in loco*, por conta de restrições de segurança das organizações, que não permitiram que documentos internos fossem levados para fora da empresa para estudo. Desta forma, foram realizadas anotações sobre os dados observados, para posterior análise. Foram analisados documentos de requisitos e processos, agendas e cronogramas.

Dois aspectos principais orientaram a realização de entrevistas: as características da empresa e do processo adotado, e as dificuldades na adoção do *Scrum*. Na coleta de dados, fez-se uso de um roteiro de perguntas semi-estruturado (Apêndices B e C) e de um questionário de perguntas estruturado com questões fechadas (Apêndice D), onde aplicou-se a escala *Likert* de 5 pontos (1=Discordo totalmente, 2=Discordo parcialmente, 3=Indiferente, 4=Concordo parcialmente, 5=Concordo totalmente).

Na entrevista guiada pelo roteiro de perguntas semi-estruturado, procurou-se identificar se o perfil da empresa estava aderente com os objetivos deste trabalho de pesquisa, e como a estrutura e o processo incorporavam o uso do *Scrum*. Foram entrevistadas cinco pessoas, cada uma relacionada a uma empresa diferente. As entrevistas foram realizadas presencialmente, em local público (café) e ambiente

informal. Em cada sessão, os pesquisados foram orientados a apresentar suas opiniões acerca de dois questionários, apresentados nos Apêndices B e C deste trabalho. Ao final de cada sessão, que variou de uma a duas horas, foi possível obter respostas a todas as questões.

Cada uma destas cinco pessoas foi, posteriormente, orientada a responder um questionário fechado, via ferramenta online de pesquisa (*TypeForm*), e divulgar este mesmo questionário a seus pares profissionais na empresa relacionada. Contudo, até a data de primeiro de Outubro, poucas pessoas haviam respondido ao questionário fechado: duas da Empresa A, duas da Empresa B, duas da empresa C, uma da empresa D, e nenhuma da empresa E. Assim sendo, como o pesquisador já havia exercido atividade profissional nas Empresas A e B, o mesmo reforçou individualmente o convite por contato em rede social (*Facebook*), e então o número de respostas ao questionário aumentou: dezoito da Empresa A, dez da Empresa B, cinco da Empresa C, uma da Empresa D, e nenhuma da Empresa E, totalizando trinta e quatro respostas.

O questionário fechado ainda recebeu quatro respostas relacionadas a empresas que não fazem parte do perfil deste trabalho. Estas respostas foram desconsideradas.

O procedimento de coleta dos dados considerou os seguintes pontos:

- *O acesso aos entrevistados*: foi feito presencialmente e pela Internet (*TypeForm*), para realização de entrevistas e questionários;
- *A disponibilidade de recursos para realização da pesquisa de campo*: houve total disponibilidade de recursos necessários, que foram i) o notebook do pesquisador, utilizado para realização de entrevistas e anotações; ii) *wi-fi* de ambientes públicos (cafés), também adotado na

realização de entrevistas; iii) recursos e ferramentas de Internet, para contatos gerais (e-mail) e aplicação de questionários fechados (pela ferramenta gratuita online *TypeForm*);

- *A organização de agenda para atividades de coleta em períodos específicos*: foi considerado um período de cinco semanas (do início de Setembro até a o fim da primera semana de Outubro de 2016) para a devida coleta dos dados. Nas duas primeiras semanas do período, o pesquisador realizou encontros presenciais com os pesquisados, onde foram realizadas duas entrevistas guiadas por um roteiro de perguntas semiestruturado. Na semana subsequente, um questionário fechado foi elaborado pelo pesquisador com base na análise das entrevistas e inserido na ferramenta online *TypeForm.* Em seguida, os pesquisados foram orientados por e-mail a acessar a ferramenta e responder o questionário. A finalização da coleta de dados ocorreu por meio do fechamento da ferramenta online de pesquisa para o público.

Em termos de proteção dos indivíduos envolvidos, este trabalho considerou aspectos propostos por Yin (2013):

- informar claramente às pessoas envolvidas sobre a natureza do estudo e seu grau de exposição, e solicitar formalmente seu aceite voluntário na participação do estudo;
- proteger a privacidade e confidencialidade dos participantes de forma que, como resultado desta participação, eles não possam ser colocados em nenhuma posição indesejada.

Assim sendo, optou-se por não apresentar o nome das pessoas e empresas nesta pesquisa.

Conforme sugerido por Yin (2013), foi elaborado um protocolo de pesquisa de forma a auxiliar nos procedimentos de coleta de dados. Este protocolo foi dividido em quatro partes:

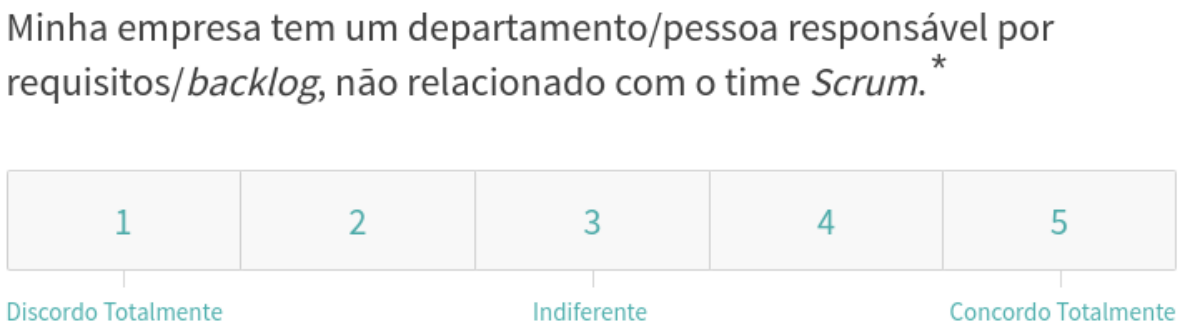
- *Apêndice A – E-mail de Apresentação*, que ofereceu uma visão geral da pesquisa ao entrevistado (bem como a explicação sobre regras de proteção à sua pessoa) e requeriu sua participação voluntária. Esta informação foi apresentada por meio de um e-mail;
- *Apêndice B – Roteiro de perguntas sobre características da empresa*, que apoiou a condução de entrevista presencial onde buscou-se identificar as características da empresa do entrevistado, verificando seu grau de aderência com a linha de pesquisa deste trabalho. Também procurou-se identificar dados secundários da empresa, como quantidade de funcionários, ramo, etc.;
- *Apêndice C – Roteiro de perguntas sobre processo*, que apoiou a condução de entrevista presencial onde buscou-se entender o processo estabelecido na empresa, bem como a relação com as tentativas de adoção do *Scrum.* Procurou-se identificar i) como se configurava o processo plan-driven da empresa, ii) como se configurava a estrutura organizacional da empresa, iii) quais eram os ganhos que a empresa gostaria de ter com a adoção de Scrum e iv) se havia dificuldades na adoção do Scrum para atingir os ganhos esperados;
- *Apêndice D – Questionário sobre dificuldades na adoção do Scrum,* construído a partir das respostas das entrevistas anteriores, que pretendeu aprofundar o entendimento de dificuldades relacionadas a características centrais do processo *Scrum:* i) adoção de times multidisciplinares; ii) adoção de times auto-organizados; iii) adoção de planejamento com base em requisitos; iv) cumprimento dos ritos estabelecidos e acordados. O questionário foi estruturado com questões

fechadas, e embasado na revisão da literatura de forma considerar pontos do método *Scrum.*

Todas as questões do questionário fechado foram redigidas de forma que a resposta “Concordo Totalmente” (valor 5 da escala *Likert*) indicasse que o tema da questão está muito incorreto em relação ao definido pelo método *Scrum*. Portanto, quanto maior a nota, maior a diferença entre o que a empresa adota e o que é esperado pelo *Scrum*. Num cenário ideal de adoção do *Scrum*, a resposta a todas as perguntas seria 1- “Discordo Totalmente”.

A Figura 7 apresenta uma das questões do questionário fechado, conforme apresentada para o entrevistado na ferramenta *TypeForm,* onde fica clara a relação de cada pergunta com as respostas na escala *Likert*.

Figura 7 – Questão apresentada aos entrevistados na ferramenta TypeForm



Fonte: Elaborado pelo autor

Cada questão apresentada no Apêndice D foi relacionada a um tema de pesquisa para análise, que considera a adoção de *Scrum* em cenários de estrutura não-projetizada e processo *plan-driven*. Esta relação foi definida no referencial teórico apresentado no Quadro 6.

3.4 Tratamento de dados

Em relação à documentação analisada, as informações coletadas foram agrupadas para análise.

Bardin (2004) oferece uma técnica para análise de conteúdo que foi adotada neste trabalho nas entrevistas presenciais (onde as questões são abertas) e na análise da documentação. Esta técnica consiste dos seguintes passos:

- Pré-análise: na seleção de documentação e entrevistas mais adequados ao entendimento do processo de desenvolvimento de *software*;
- Exploração dos materiais: no qual as entrevistas foram revisadas e validadas;
- Tratamento dos dados e interpretação: no qual os dados foram processados e interpretados de acordo com o conhecimento levantado no estudo bibliográfico.

Para o questionário estruturado com perguntas fechadas, a técnica de análise de conteúdo de Bardin não foi aplicada. Ao invés disto, o seguinte método foi empregado a todas as empresas:

i. O questionário fechado foi apresentado e respondido por diversos profissionais da empresa. Neste questionário, procurou-se analisar cada característica do *Scrum* individualmente, relacionando o grau de sua adoção enquanto influenciada tanto pelo processo *plan-driven* quando pela estrutura organizacional;
ii. As respostas foram consolidadas numericamente por média. Com a média, procurou-se obter um indicativo geral de onde estão as maiores dificuldades na adoção do *Scrum*;

iii. Com base nos dados consolidados, foi aplicado o critério de maior média para seleção dos cinco temas mais significativos.
iv. Os temas selecionados foram analisados detalhadamente, à luz dos estudos bibliográficos, das informações coletadas nas entrevistas semi-estruturadas e das documentações analisadas *in loco*.

A análise propriamente dita foi realizada individualmente para cada estudo de caso e, posteriormente, de forma comparativa entre os casos.

3.5 Análise do Modelo de Pesquisa

A coleta e análise de diferentes fontes de evidência (entrevistas e documentação) configuraram uma triangulação de dados, de forma a conferir maior validade aos constructos do estudo de caso (YIN, 2013).

Yin (2013) apresenta quatro critérios para validação da qualidade de um modelo de pesquisa: validade do constructo, validade interna, validade externa e confiabilidade. Destes quatro, descarta-se a validade interna, por ser aplicável apenas a estudos explanatórios ou causais, e não a estudos descritivos como o aqui realizado (YIN, 2013).

3.5.1 Validade do Constructo

A validade do constructo identifica medidas operacionais adequadas para os conceitos sendo estudados (YIN, 2013).

Este trabalho não utiliza constructos, pois não pretende validar um modelo. A lógica proposta é utilizada para validar a pesquisa em termos de dificuldades encontradas na adoção de *Scrum*.

A validade do constructo neste trabalho fez uso das seguintes táticas: i) múltiplas fontes de evidências, preferíveis a um estudo de caso único no sentido que melhoram a validade do constructo (YIN, 2013); ii) triangulação de dados, cuja validação convergente de múltiplos métodos fornece maior força à substância dos constructos (EISENBERG, 1989) (JICK, 1979).

3.5.2 Validade Externa

A validade externa define o domínio pelo qual as descobertas do estudo podem ser generalizadas, independentemente do método de pesquisa utilizado (YIN, 2013). Todo estudo de caso múltiplo deve seguir uma lógica de replicação, e não de amostragem (YIN, 2013). Este trabalho atingiu validade externa pela aplicação da lógica de replicação (YIN, 2013), de forma a estudar um primeiro caso e então replicando o experimento pela condução de demais estudos de caso (considerando que as empresas A, B e C compartilham similaridades). Buscou-se duplicar as condições exatas da primeira pesquisa nas pesquisas subsequentes.

3.5.3 Confiabilidade

A confiabilidade minimiza os erros de um estudo ao definir que, se um segundo pesquisador conduzir o mesmo estudo de um primeiro pesquisador seguindo os mesmos procedimentos, deve chegar às mesmas conclusões (YIN, 2013). Para este fim, este trabalho de pesquisa documentou o protocolo de pesquisa e os instrumentos como apêndices do trabalho, de forma que a pesquisa pudesse ser repetida por outros pesquisadores.

4 RESULTADOS DA PESQUISA E ANÁLISE

Este capítulo apresenta os estudos de casos realizados para três empresas distintas atuantes no mercado brasileiro, chamadas neste trabalho de Empresa A, Empresa B e Empresa C.

Para cada uma das seções voltadas às empresas, são apresentados: i) características da empresa e do processo adotado, tendo por base as informações obtidas nas entrevistas e na análise da documentação *in loco*; ii) dificuldades na adoção do *Scrum*, onde os dados coletados no questionário, apoiados pelas informações das entrevistas, são analisados e interpretados de forma a identificar os elementos de maior dificuldade para a adoção de *Scrum* na empresa.

4.1 Empresa A

Esta seção apresenta a análise da Empresa A, considerando suas características gerais, estrutura organizacional e processo adotados, bem como as dificuldades e conflitos na adoção do *Scrum*.

4.1.1 Características da empresa e do processo adotado

A Empresa A é uma multinacional brasileira de grande porte do ramo de energia, com décadas de atuação no mercado. Apresenta algo em torno de quatrocentos profissionais entre funcionários e prestadores de serviço no setor de TI em São Paulo. Emprega perto de oitenta mil funcionários.

Trata-se de uma empresa que não tem seu cerne de negócio em tecnologia da informação, fazendo uso desta disciplina para suporte às suas operações. Na pesquisa global *State of Agile*, que considera o interesse de indústrias da

comunidade de desenvolvimento de *software* global em relação à adoção de métodos ágeis, observou-se este tipo de característica em setenta e cinco por cento das empresas pesquisadas (VERSIONONE, 2014).

Na entrevista e nos documentos analisados, identificou-se que a estrutura organizacional adotada poderia ser classificada como não-projetizada. Segundo o PMI (2013), algumas características associadas a uma estrutura projetizada são a alta autoridade do gerente de projetos, o alto tempo de alocação de profissionais alocados em projeto (de 85% a 100%), e a existência do papel do gerente de projetos como atividade de tempo integral. Estes critérios não são adotados na organização do setor de TI da Empresa A. O entrevistado relatou que a estrutura é formada por diferentes departamentos (um para cada diferente disciplina de engenharia de software), com cada profissional inserido hierarquicamente abaixo de um dos departamentos. De forma transversal, existe uma organização de projeto por times com profissionais de diferentes departamentos, mas as tarefas do time concorrem com as tarefas do próprio departamento, de forma que não há alocação exclusiva. Além disto, embora o gerente de projetos seja um papel de tempo integral, sua autoridade concorre com a autoridade dos gerentes de departamentos, especialmente em termos de alocação de profissionais.

Identificou-se, também, que o processo poderia ser classificado como *plan-driven*. Análise de documentos *in loco* e relatos do entrevistado evindenciaram a adoção do processo RUP (*Rational Unified Process*, da IBM) em cascata. A forma pela qual o processo em cascata é adotado na Empresa A é diferente do proposto por Royce (1970), onde uma disciplina do processo poderia iteragir com a disciplina anterior. Na Empresa A, uma disciplina apenas pode iteragir com a próxima na cadeia do processo. Uma consequência direta disto é que os requisitos devem estar completamente definidos para que só então possam ser construídos, o que é uma definição de *plan-driven* (PETERSEN; WOHLIN, 2010). A Empresa A consegue flexibilizar esta forma de trabalho sequencial quando os profissionais estão atuando

em times *Scrum*, contudo, a diretriz oficial do setor de TI é que o processo deve ser RUP em cascata. O setor de TI da Empresa A não segue nenhum modelo de qualidade em processo de software, como CMMI.

A empresa lida, portanto, com o cenário no qual o processo *plan-driven* concorre com o método *Scrum* no setor de TI, numa estrutura não-projetizada.

4.1.2 Dificuldades na adoção de Scrum

Dezoito entrevistados responderam às questões fechadas sobre dificuldades de *Scrum* na Empresa A. Estas questões são apresentadas no Apêndice D deste documento, e cada uma das respostas coletadas são encontradas no Apêndice E.

A Tabela 1 apresenta a consolidação e ordernação das respostas por média.

Tabela 1 – Respostas para análise do questionário fechado da Empresa A (Continua).

Número	Questão	Tema relacionado	Média
19	As pessoas do time *Scrum* tendem a inspecionar o processo visando uma organização das atividades onde as tarefas de análise devem preceder as tarefas de construção, que por sua vez devem preceder as tarefas de teste.	pilar Inspeção em processo *plan-driven*	3,9
2	Minha empresa detalha todos os requisitos/itens do backlog para que só então o time de desenvolvimento possa começar a desenvolver.	*backlog*/requisitos em processo *plan-driven*	3,8
1	Minha empresa tem um departamento/pessoa responsável por requisitos/*backlog*, não relacionado com o time Scrum	*backlog*/requisitos em estrutura não-projetizada	3,5
18	As pessoas do time *Scrum* tendem a inspecionar seu processo considerando mais o ponto de vista do seu departamento original do que o ponto de vista do time como um todo.	pilar Inspeção em estrutura não-projetizada	3,3
8	O time *Scrum* não realiza reunião de revisão (*Spring Review*) com o cliente, mas sim com um outro departamento intermediário, que recebe as demandas.	reunião de revisão em processo *plan-driven*	3,3
20	Minha empresa não aplica melhorias ao processo pois valoriza o método *Scrum* menos do que seus próprios processos departamentais individuais.	pilar Adaptação em estrutura não-projetizada	3,2
11	Minha empresa tem dificuldades em adotar o papel de *Product Owner* porque este papel não existe formalmente em nenhum departamento.	papel de *Product Owner* em estrutura não-projetizada	3,2
12	Minha empresa não consegue fazer com que o *Product Owner* esteja envolvido com o time de desenvolvimento nos momentos em que a interação faz-se necessária.	papel de *Product Owner* em processo *plan-driven*	3,1
14	O *Scrum Master* da minha empresa tem dificuldades no cumprimento do processo *Scrum* por conta da falta de apoio de outros departamentos, que nem sempre respeitam os limites e regras do processo *Scrum*.	papel de *Scrum Master* em processo *plan-driven*	3,1
4	Minha empresa só realiza a reunião de planejamento (*Spring Planning*) quando o entendimento dos requisitos é total e está devidamente formalizado.	reunião de planejamento em processo *plan-driven*	3,1
3	Minha empresa não consegue reunir todas as pessoas necessárias nas reuniões de planejamento (*Spring Planning*) por conta de compromissos das pessoas com tarefas de seus departamentos de origem.	reunião de planejamento em estrutura não-projetizada	3,0
16	O time de desenvolvimento entende que as tarefas do time multidisciplinar não podem ser feitas em paralelo, mas sim devem ser feitas em cascata, iniciando pelos requisitos, para que depois sejam feitas as tarefas de construção e só então as tarefas de teste.	papel do time de desenvolvimento em processo *plan-driven*	3,0
10	O time *Scrum* tem dificuldades na realização de uma boa reunião de retrospectiva (*Sprint Retrospective*) porque as melhorias no processo são discutidas de forma fragmentada visando interesses de diferentes áreas, e não visam a melhoria do processo para o time como um todo.	reunião de retrospectiva em processo *plan-driven*	2,9

Tabela 1 – Respostas para análise do questionário fechado da Empresa A (Conclusão).

Número	Questão	Tema relacionado	Média
7	O time *Scrum* não realiza reunião de revisão (*Spring Review*) pois não há interesse do cliente ou há desconhecimento sobre quem é o cliente ou departamento representante do cliente para o produto sendo desenvolvido.	reunião de revisão em estrutura não-projetizada	2,6
17	Minha empresa tem dificuldades em deixar claros os papeis e responsabilidades das pessoas no cenário onde seu trabalho é dividido entre atuar no time *Scrum* e em seu próprio departamento.	pilar Transparencia em estrutura não-projetizada	2,5
21	Minha empresa tem dificuldade em manter um *timebox* no *Sprint*, pois membros do time constantemente são interrompidos para resolver problemas de seus departamentos de origem.	*timebox* em estrutura não-projetizada	2,3
13	Minha empresa tem dificuldade em adotar o papel de *Scrum Master* porque este papel não existe formalmente em nenhum departamento de TI.	papel de *Scrum Master* em estrutura não-projetizada	2,3
5	Os profissionais do time *Scrum* não conseguem comparecer às reuniões diárias (*Scrum Daily Meetings*) por terem conflito de agenda por conta de compromissos em seus próprios departamentos.	reunião diária em estrutura não-projetizada	2,2
9	O time *Scrum* não realiza reunião de retrospectiva (*Spring Retrospective*) porque não ha muito interesse na melhoria do processo ágil, já que as pessoas do time têm processos diferentes em seus próprios departamentos.	reunião de retrospectiva em estrutura não-projetizada	1,8
6	Os profissionais do time *Scrum* não participam das reuniões diárias (*Scrum Daily Meetings*) porque não vêem utilidade, já que as tarefas dos analistas já foram feitas antes do *Sprint*, ou as tarefas dos testadores só irão ocorrer após o *Sprint*.	reunião diária em processo *plan-driven*	1,8
15	Minha empresa tem dificuldades em manter o time de desenvolvimento trabalhando junto na iteração, pois membros do time precisam também trabalhar em tarefas de seus departamentos de origem.	papel do time de desenvolvimento em estrutura não-projetizada	1,6

Fonte: Próprio autor

4.1.3 Interpretação dos resultados

Esta seção analisa os dados obtidos. Foram selecionadas as cinco questões do questionário fechado que obtiveram a maior média. O tema relacionado a cada uma destas cinco questões foi então analisado, considerando os elementos levantados pelos questionários abertos e a revisão bibliográfica.

4.1.3.1 Pilar Inspeção em processo *plan-driven*

O pilar Inspeção do *Scrum* sugere que o time deve inspecionar seu próprio processo. Embora deva ocorrer, a inspeção não deve ser tão frequente de forma a atrapalhar a execução das tarefas. Um momento ideal para realização da inspeção é nas Reuniões de Retrospectiva (*Sprint Retrospective*), onde o time pode inspecionar a si próprio e criar um plano de melhorias a serem aplicadas na próxima iteração (SCHWABER; SUTHERLAND, 2013).

Os times possuem liberdade para a organização interna de suas atividades. Ainda assim, escolhem organizar suas tarefas de forma que só depois que a construção tenha ocorrido as tarefas de teste devem ocorrer, por exemplo. Não há necessidade que isto seja feito desta forma, pois num time *Scrum*, tarefas que podem ser feitas pelos testadores (como escrita de critérios de aceitação, por exemplo) devem ocorrer ao mesmo tempo em que o código-fonte é construído, já que ambas derivam dos requisitos. A empresa adota um processo plan*-driven* RUP cascata, mas isto não significa que a divisão de tarefas dentro do time precise seguir este mesmo modelo.

Portanto, para esta questão, entende-se que o time tem liberdade para trabalhar o pilar Inspeção, mas o está fazendo com base em valores do processo *plan-driven*. A dificuldade de adoção identificada aqui não é relacionada à estrutura

organizacional ou ao processo *plan-driven*, mas sim da failta de conhecimento do time sobre os princípios e valores do *Scrum* e dos métodos ágeis.

4.1.3.2 Backlog/requisitos em processo *plan-driven*

O *Backlog* do produto é responsabilidade do *Product Owner*, que deve ser o único responsável por gerenciá-lo. O gerenciamento do *Backlog* inclui: i) expressar claramente os itens; ii) ordenar o *Backlog* para alcançar melhor as metas e missões; iii) garantir que o *Backlog* seja visível, transparente e claro para todos; iv) garantir que o time de desenvolvimento entenda os itens do *Backlog* no nível necessário. O *Product Owner* pode delegar este trabalho para o time de desenvolvimento, contudo ainda é o responsável por ele (SCHWABER; SUTHERLAND, 2013).

Em processos *plan-driven* os requisitos devem ser totalmente especificados antes de sua construção (PETERSEN; WOHLIN, 2010). Nesta empresa, observou-se que existe uma área externa ao time *Scrum* que é responsável por realizar a documentação. Contudo, para o *Product Owner*, não é necessário especificar totalmente um requisito, mas apenas garantir que o time de desenvolvimento entenda suficientemente o que precisa ser feito. Neste cenário, o *Product Owner* e os desenvolvedores do time podem até saber o que deve ser feito, mas não podem fazê-lo antes de estar completamente especificado.

O fato da empresa não manter a criação de requisitos juntamente com o time e de não adotar o papel de *Product Owner* para a organização e priorização dos requisitos são dois fatores que dificultam a adoção de *Scrum*.

4.1.3.3 Backlog/requisitos em estrutura não-projetizada

Esta seção trata do mesmo tema da seção anterior, mas do ponto de vista da estrutura não-projetizada. A empresa possui um departamento específico para tratar de requisitos que não se encontra no time *Scrum*. Não existe o papel formal do *Product Owner*, e o time recebe os requisitos já prontos.

Numa estrutura não-projetizada, projetos concorrem com departamentos pela disponibilidade de pessoas para execução de tarefas (PMI, 2013), e no caso desta empresa, existe um departamento para requisitos, distinto do departamento de desenvolvimento, que por sua vez é distinto do departamento de QA. Pessoas de diferentes departamentos conseguem trabalhar juntas em projetos em tempo parcial, mas os requisitos ficam fora do time. Com isto, os documentos de requisitos podem não ficar prontos em tempo hábil para o início da iteração, pois o departamento de requisitos tem suas próprias prioridades, que podem não estar alinhadas com as prioridades de todos os times. Isto pode gerar atrasos no desenvolvimento, afetando as datas de entrega acordadas.

A empresa está adotando práticas opostas às sugeridas pelo *Scrum* ao ter uma parte do trabalho do time externa ao próprio time. O ideal seria que os requisitos estivessem a cargo do *Product Owner* (membro do time *Scrum*), e que fossem gerenciados pelo time *Scrum* (SCHWABER; SUTHERLAND, 2013).

O fato da empresa manter os requisitos separados do time, podendo ocasionar atrasos por conta de prioridades distintas de diferentes departamentos, é um fator que dficulta a adoção de *Scrum*.

4.1.3.4 Pilar Inspeção em estrutura não-projetizada

Esta seção trata do mesmo tema de uma seção anterior (Inspeção), mas do ponto de vista da estrutura não-projetizada. As pessoas do time tendem a inspecionar o processo do time considerando mais o ponto de vista de seu departamento original do que a realidade do time *Scrum*. Isto significa que um profissional do time *Scrum* que vem do departamento de análise de requisitos, por exemplo, apenas propõe sugestões de melhoria do processo que tenha a ver com atividades de análise de requisitos.

Scrum espera que um time seja multidisciplinar e auto-organizado, escolhendo qual a melhor forma de completar seu trabalho, sem que sejam dirigidos por outros de fora do time. A multidisciplinaridade proposta não significa que cada membro do time seja capaz de realizar todas as atividades, mas sim que o time, como um todo, tenha todos os elementos que permitam que as tarefas sejam realizadas. Além disto, para o *Scrum* não existem analistas de requisitos, programadores e testadores: o único tiulo que existe dentro de um time *Scrum* é o de Desenvolvedor (SCHWABER; SUTHERLAND, 2013).

Na Empresa A, os times multidisciplinares são compostos por pessoas de diferentes departamentos especializados. Contudo, as pessoas ainda mantém seus títulos do departamento de origem (testador, programador, analista), e restringem-se a propor melhorias no processo apenas acerca de sua especialidade, sem enxergar o time como um todo. O *Scrum* espera que todos os membros do time melhorem o processo do time independentemente da posição e cargo que ocupem fora do time.

Esta é uma questão de atitude interna do time, não relacionada à estrutura organizacional ou ao processo *plan-driven*. A dificuldade de adoção identificada aqui é relacionada à falta de conhecimento do time sobre os princípios e valores do *Scrum* e dos métodos ágeis.

4.1.3.5 Reunião de revisão em processo *plan-driven*

Na Empresa A, o cliente não participa das reuniões de revisão (*Sprint Review*), que são aquelas onde o resultado do trabalho da iteração é apresentado. Contudo, o *Scrum* não exige que o cliente esteja presente, mas sim que estejam presentes as partes interessadas, que podem ser outros *stakeholders* além do cliente final (SCHWABER; SUTHERLAND, 2013).

Em entrevista, observou-se que existe uma área que faz a interface entre o cliente e o time *Scrum,* e que profissionais desta área geralmente participam das reuniões de revisão. Estes *stakeholders* não fazem parte do time: estão do lado de fora, validando o trabalho realizado em cada iteração.

No processo RUP cascata adotado na Empresa A, o *stakeholder* deveria apenas ser envolvido ao final do desenvolvimento. Mas com o método *Scrum*, o *stakeholder* é envolvido a cada iteração, na reunião de revisão. O *stakeholder* atende aos objetivos do *Scrum* de fornecer feedback rápido e obter comentários que possam promover a colaboração (SCHWABER; SUTHERLAND, 2013).

Nesta questão, entende-se que os times *Scrum* estão estruturados para funcionar conforme esperado pelo método *Scrum*. Desta forma, não há dificuldades ou conflitos observados.

4.2 Empresa B

Esta seção apresenta a análise da Empresa B, considerando suas características gerais, estrutura organizacional e processo adotados, bem como as dificuldades e conflitos na adoção do *Scrum*.

4.2.1 Características da empresa e do processo adotado

A Empresa B é uma seguradora multinacional antiga e consolidada no mercado. Possui perto de cem funcionários na área de TI estudada, e emprega milhares de funcionários ao redor do mundo. A área de TI estudada não representa toda a área de TI da empresa, mas apenas uma parte, que é responsável pelo conjunto de *softwares* utilizados pelas filiais de toda a América Latina.

Assim como a Empresa A, não tem seu cerne de negócio em tecnologia da informação, fazendo uso desta disciplina para suporte às suas operações.

Na entrevista e nos documentos analisados *in loco*, observou-se que a estrutura da TI é segmentada em três setores principais: um voltado para evoluções e manutenções do *software*, outro para entregas e outro para infraestrutura. No setor de evoluções e manutenções, existem departamentos de arquitetura e desenvolvimento. No setor de entregas, existem departamentos para suporte à produção e entrega. No setor de infra-estrutura, a organização é mais simples, com um único departamento. As pessoas são dispostas nestes departamentos, e não em projetos, o que é uma das características de uma estrutura organizacional não-projetizada (PMI, 2013).

A área de TI tem como objetivo principal a manutenção do *software* existente e a entrega de novas *releases* deste *software* para os clientes internos da companhia, distrubuídos em diferentes países. Toda a comunicação e rastreamento das atividades é feita pela ferramenta Jira, da empresa australiana Atlassian. O setor de entregas é quem define quais destas tarefas serão entregues na próxima *release*, e a partir daí, um processo cascata é assumido, na seguinte ordem: i) o setor de entregas escreve um requisito de alto nível e um documento de regras, com base na iteração com o cliente; ii) os setores de desenvolvimento e arquitetura elaboram uma solução técnica e sua devida implementação de acordo com os dados fornecidos

pelo setor de entregas; iii) o departamento de testes do setor de entregas recebe o produto desenvolvido, e efetua os testes funcionais para validar a entrega; iv) a *release* é aprovada e disponibilizada para os clientes em ambiente de testes; v) depois de aprovada pelos clientes, a *release* entra em produção.

Este tipo de modelo em cascata, onde a construção só pode iniciar depois do setor de entregas escrever uma documentação e onde a equipe de testes somente é envolvida após a construção, caracteriza-se como *plan-driven* (PETERSEN; WOHLIN, 2010).

Também observou-se que é prática comum que os profissionais atuem em regime de força-tarefa para cumprimento dos prazos. Neste momento, o modelo em cascata é interrompido para que pessoas trabalhem efetivamente juntas visando a entrega, mas a forma como isto ocorre é desorganizada e sem nenhum método ou processo observável, sacrificando a qualidade em detrimento do prazo.

Kerzner (2009) sugere que a prática da força-tarefa traz problemas de integração e coordenação, por não haver autoridade para responder pelo projeto. A cultura de suspender formas de trabalho previamente definidas e adotadas para atender à necessidade pontual do cliente no prazo mais rápido possível por meio de inicitiavas de forças-tarefa é muito forte, antiga e valorizada na Empresa B.

Esta empresa, a exemplo da Empresa A, também lida com o cenário no qual o processo *plan-driven* concorre com o método *Scrum* no setor de TI, numa estrutura não-projetizada, com eventuais ações de força-tarefa.

4.2.2 Dificuldades na adoção de *Scrum*

Dez entrevistados responderam às questões fechadas sobre dificuldades de *Scrum* na Empresa B. Estas questões são apresentadas no Apêndice D deste documento, e cada uma das respostas coletadas são encontradas no Apêndice F.

A Tabela 2 apresenta a consolidação e ordenação das respostas por média.

Tabela 2 – Respostas para análise do questionário fechado da Empresa B (Continua).

Número	Questão	Tema relacionado	Média
13	Minha empresa tem dificuldade em adotar o papel de *Scrum Master* porque este papel não existe formalmente em nenhum departamento de TI.	papel de *Scrum Master* em estrutura não-projetizada	5,0
11	Minha empresa tem dificuldades em adotar o papel de *Product Owner* porque este papel não existe formalmente em nenhum departamento.	papel de *Product Owner* em estrutura não-projetizada	4,8
14	O *Scrum Master* da minha empresa tem dificuldades no cumprimento do processo *Scrum* por conta da falta de apoio de outros departamentos, que nem sempre respeitam os limites e regras do processo *Scrum*.	papel de *Scrum Master* em processo *plan-driven*	4,8
17	Minha empresa tem dificuldades em deixar claros os papeis e responsabilidades das pessoas no cenário onde seu trabalho é dividido entre atuar no time *Scrum* e em seu próprio departamento.	pilar Transparencia em estrutura não-projetizada	4,8
21	Minha empresa tem dificuldade em manter um *timebox* no *Sprint*, pois membros do time constantemente são interrompidos para resolver problemas de seus departamentos de origem.	*timebox* em estrutura não-projetizada	4,5
12	Minha empresa não consegue fazer com que o *Product Owner* esteja envolvido com o time de desenvolvimento nos momentos em que a interação faz-se necessária.	papel de *Product Owner* em processo *plan-driven*	4,5
20	Minha empresa não aplica melhorias ao processo pois valoriza o método *Scrum* menos do que seus próprios processos departamentais individuais.	pilar Adaptação em estrutura não-projetizada	4,5
18	As pessoas do time *Scrum* tendem a inspecionar seu processo considerando mais o ponto de vista do seu departamento original do que o ponto de vista do time como um todo.	pilar Inspeção em estrutura não-projetizada	4,4
15	Minha empresa tem dificuldades em manter o time de desenvolvimento trabalhando junto na iteração, pois membros do time precisam também trabalhar em tarefas de seus departamentos de origem.	papel do time de desenvolvimento em estrutura não-projetizada	4,3
8	O time *Scrum* não realiza reunião de revisão (*Spring Review*) com o cliente, mas sim com um outro departamento intermediário, que recebe as demandas.	reunião de revisão em processo *plan-driven*	4,2
10	O time *Scrum* tem dificuldades na realização de uma boa reunião de retrospectiva (*Sprint Retrospective*) porque as melhorias no processo são discutidas de forma fragmentada visando interesses de diferentes áreas, e não visam a melhoria do processo para o time como um todo.	reunião de retrospectiva em processo *plan-driven*	4,0
3	Minha empresa não consegue reunir todas as pessoas necessárias nas reuniões de planejamento (*Spring Planning*) por conta de compromissos das pessoas com tarefas de seus departamentos de origem.	reunião de planejamento em estrutura não-projetizada	3,9

Tabela 2 – Respostas para análise do questionário fechado da Empresa B (Conclusão).

Número	Questão	Tema relacionado	Média
19	As pessoas do time *Scrum* tendem a inspecionar o processo visando uma organização das atividades onde as tarefas de análise devem preceder as tarefas de construção, que por sua vez devem preceder as tarefas de teste.	pilar Inspeção em processo *plan-driven*	3,8
16	O time de desenvolvimento entende que as tarefas do time multidisciplinar não podem ser feitas em paralelo, mas sim devem ser feitas em cascata, iniciando pelos requisitos, para que depois sejam feitas as tarefas de construção e só então as tarefas de teste.	papel do time de desenvolvimento em processo *plan-driven*	3,8
7	O time *Scrum* não realiza reunião de revisão (*Spring Review*) pois não há interesse do cliente ou há desconhecimento sobre quem é o cliente ou departamento representante do cliente para o produto sendo desenvolvido.	reunião de revisão em estrutura não-projetizada	3,6
9	O time *Scrum* não realiza reunião de retrospectiva (*Spring Retrospective*) porque não ha muito interesse na melhoria do processo ágil, já que as pessoas do time têm processos diferentes em seus próprios departamentos.	reunião de retrospectiva em estrutura não-projetizada	3,3
1	Minha empresa tem um departamento/pessoa responsável por requisitos/*backlog*, não relacionado com o time *Scrum*.	*backlog*/requisitos em estrutura não-projetizada	3,2
6	Os profissionais do time *Scrum* não participam das reuniões diárias (*Scrum Daily Meetings*) porque não vêem utilidade, já que as tarefas dos analistas já foram feitas antes do *Sprint*, ou as tarefas dos testadores só irão ocorrer após o *Sprint*.	reunião diária em processo *plan-driven*	2,8
2	Minha empresa detalha todos os requisitos/itens do *backlog* para que só então o time de desenvolvimento possa começar a desenvolver.	*backlog*/requisitos em processo *plan-driven*	2,6
4	Minha empresa só realiza a reunião de planejamento (*Spring Planning*) quando o entendimento dos requisitos é total e está devidamente formalizado.	reunião de planejamento em processo *plan-driven*	2,3
5	Os profissionais do time *Scrum* não conseguem comparecer às reuniões diárias (*Scrum Daily Meetings*) por terem conflito de agenda por conta de compromissos em seus próprios departamentos.	reunião diária em estrutura não-projetizada	2,2

Fonte: Próprio autor

4.2.3 Interpretação dos resultados

Esta seção analisa os dados obtidos. Foram selecionadas as cinco questões do questionário fechado que obtiveram a maior média. O tema relacionado a cada uma destas cinco questões foi então analisado, considerando os elementos levantados pelos questionários abertos, análise de documentação *in loco* e revisão bibliográfica.

4.2.3.1 Papel de Scrum Master em estrutura não-projetizada

O *Scrum* espera que cada time possua um profissional exercendo o papel de *Scrum Master.* O *Scrum Master* é responsável por garantir que o *Scrum* seja entendido e aplicado na organização, tanto dentro quanto fora dos times. Deve guiar o time *Scrum* dentro das regras e das práticas do *Scrum*, ao mesmo tempo que deve conscientizar as pessoas de fora do time *Scrum* em relação a como devem iteragir com o time. Deve liderar, ensinar e treinar as pessoas para trabalharem devidamente com o método (SCHWABER; SUTHERLAND, 2013).

O papel de *Scrum Master* apenas existe no time *Scrum*, e não faz parte de nenhum departamento de TI. Na Empresa B, o papel formal de *Scrum Master* não está formalmente definido. Os coordenadores de desenvolvimento costumam assumir o papel de *Scrum Master*, mas documentos *in loco* demonstram que as atividades exercidas não estão totalmente de acordo com as atividades esperadas de um *Scrum Master*.

Uma estrutura não-projetizada oferece dificuldades ao papel do *Scrum Master.* A organização de pessoas em setores distintos gera o problema de disponibilidade de pessoas para atuação em times (PMI, 2013), de forma que a adoção de um time multidisciplinar (SCHWABER; SUTHERLAND, 2013) não consegue ser devidamente atendida. O *Scrum Master* tem dificuldades em trabalhar

quando a participação dos profissionais no time concorre com sua participação em outros departamentos.

Um dos coordenadores de desenvolvimento, entrevistado sobre este tema, afirmou que vê ganhos na utilização do *Scrum* mas encontra problemas em obter condições mínimas para manter um time unido, por falta de apoio da alta gestão.

Além disto, a autoridade do *Scrum Master* é reduzida pelo desinteresse da alta gestão da Empresa B. As iniciativas de adoção de *Scrum* limitam-se ao departamento de desenvolvimento do setor de evoluções, por iniciativa própria. Os demais setores e departamentos não participam, o que torna ainda mais distante a elaboração de times *Scrum* multidisciplinares e a influência do *Scrum Master* entre os setores e departamentos.

O fato de não haver um maior apoio do método *Scrum* na estrutura organizacional, e do papel de *Scrum Master* como consequência, são dificuldades para a adoção de *Scrum*.

4.2.3.2 Papel de Product Owner em estrutura não-projetizada

O *Product Owner* deve ser o profissional responsável por prover informações do negócio para o time de desenvolvimento. Este tipo de papel não cabe a nenhum departamento específico da Empresa B, mas sim ao próprio time *Scrum* (SCHWABER; SUTHERLAND, 2013).

Uma estrutura não-projetizada oferece dificuldades ao papel do *Product Owner.* Na Empresa B, as atividades que seriam associadas ao *Product Owner* - definição de requisitos e iteração com o cliente – cabem a um outro departamento, que não tem interesse na adoção do método *Scrum*.

A impossibilidade de manter um método *Scrum* por muito tempo, em detrimento das força-tarefa emergenciais, aliada à questão da falta de interesse de outros departamentos, são elementos que contribuem para que este papel não seja valorizado e adotado.

O fato de não haver um maior apoio do método *Scrum* na estrutura organizacional, e do papel de *Product Owner* como consequência, são dificuldades para a adoção de *Scrum*.

4.2.3.3 Papel de Scrum Master em processo *plan-driven*

Este tema também trata da falta de valorização do método *Scrum* pela empresa. Os entrevistados entendem que os *Scrum Masters* não conseguem adotar corretamente o método *Scrum* por falta de apoio de outras áreas, que estão pouco propensas a colaborar e que interferem nos times. O papel do *Scrum Master* na Empresa B não tem a autoridade suficiente para bloquear estas intereferências.

O setor de entregas estabelece compromissos de prazo com o cliente de forma unilateral, sem participação do departamento de desenvolvimento. No processo *plan-driven* da empresa, o departamento de desenvolvimento é pressionado a entregar o produto mais rapidamente para que haja mais tempo para o setor de entregas realizar seus testes, e o setor de entregas ignora o *timebox* dos times *Scrum* do departamento de desenvolvimento, considerando apenas a data final.

O fato do *Scrum Master* não conseguir atuar propriamente na aplicação de regras do método *Scrum* por falta de apoio da alta gestão e colaboração de outros departamentos, que acabam por ocasionar interferências externas prejudiciais, é uma dificuldade para a adoção de *Scrum*.

4.2.3.4 Pilar Transparência em estrutura não-projetizada

Os profissionais do time de desenvolvimento atuam num cenário onde seu uso de *Scrum* é limitado apenas a seu departamento, de forma que existem muitos conflitos relacionados a expectativas de outras áreas, que não desejam colaborar. Além disto, estes profissionais frequentemente possuem sua forma de trabalho afetada, indo de um método *Scrum* para iniciativas de força-tarefa, e vice-versa.

Por conta dos conflitos e da instabilidade na forma de trabalho, os profissionais apontam dificuldades em entender quais são exatamente seus papeis e responsabilidades neste ambiente. Em determinados momentos, atuam com o método *Scrum* e, em outros, atuam sem nenhum processo claramente definido, respondendo apenas às necessidades pontuais do departamento para atendimento da força-tarefa.

O fato da empresa não esclarecer quais são os papeis e responsabilidades esperados dos programadores em relação à sua atuação em times *Scrum* num ambiente onde o método é continuamente interrompido é uma dificuldade para a adoção de *Scrum*.

4.2.3.5 Timebox em estrutura não-projetizada

O *timebox* no *Scrum* define os limites de duração de uma iteração (*Sprint*). Não deve ultrapassar um mês, pois quando o horizonte da iteração é muito longo, a definição do que será construído pode mudar, a complexidade pode aumentar e o risco pode crescer. O *Scrum* emprega uma abordagem iterativa e incremental para aperfeiçoar a previsibilidade e o controle de riscos, e o *timebox* de duração fixa é parte essencial deste método (SCHWABER; SUTHERLAND, 2013).

Na estrutura não-projetizada da Empresa B, os times *Scrum* são muito influenciados por atividades de fora do time, que surgem como necessidades de atendimentos emergenciais para o setor de entregas. Assim sendo, o *timebox* da iteração acaba sendo prejudicado, pois é constantemente interrompido para atendimento às emergências, quando nunca deveria ser interrompido.

Ao não adotar um *timebox* de duração fixa, o time perde a capacidade de fazer uso de outras práticas comuns da comunidade ágil que complementam o *Scrum*, como o *Planning Poker* (para determinar a quantidade de esforço que poderá ser realizada na iteração) (HAUGEN, 2006) e o gráfico de *Burndown* (que mede o progresso diário do time dentro da iteração) (SUTHERLAND, 2001). Desta forma, o time *Scrum* não consegue medir o quanto é capaz de produzir a cada iteração.

O fato do *timebox* ser constantemente interrompido por conta de influências externas ao time é uma dificuldade para a adoção de *Scrum*.

4.3 Empresa C

Esta seção apresenta a análise da Empresa C, considerando suas características gerais, estrutura organizacional e processo adotados, bem como as dificuldades e conflitos na adoção do *Scrum*.

4.3.1 Características da empresa e do processo adotado

A Empresa C é uma empresa de terceirização de várias áreas de *e-commerce*, realizando atividades de logistica, estocagem, campanhas de *marketing* e fidelização de produto via programas de bônus. Possui menos de duzentos funcionários dispostos em dois andares, englobando áreas como atendimento ao

cliente, marketing, compras, cadastro, logística, BI, ERP, finanças, recursos humanos e TI. Assim como as demais empresas pesquisadas neste trabalho, não tem seu cerne de negócio em tecnologia da informação, fazendo uso desta disciplina para suporte às suas operações.

A estrutura organizacional da empresa poderia ser classificada como não-projetizada. De acordo com o entrevistado, a estrutura de TI está organizada a partir de um *CTO* (*Chief Technology Officer*, ou diretor técnico), que possui um gerente de TI, com seus coordenadores de infra-estrutura e de desenvolvimento. As áreas de BI e ERP ficam de fora da área de TI, dentro do departmento financeiro. Existe um escritório de projetos, mas que atua apenas na área de operações e logística. Na gestão de TI, os departamentos de infra-estrutura e desenvolvimento trabalham próximos e adotam *Scrum*, pois o gestor é entusiasta do método. Contudo, outros departamentos atuam de forma mais distante, sem consideração por *Scrum* e métodos ágeis, com diferentes objetivos e necessidades que nem sempre estão alinhados com a realidade dos times de desenvolvimento.

O processo na área de TI consegue organizar pessoas para atuação de forma multidisciplinar e auto-organizada, em times. Contudo, esta organização cabe apenas à área de TI, que situa-se dentro de um processo maior. Os requisitos chegam por meio de diferentes áreas, que não possuem processos claramente definidos. Um profissional atua como *Product Owner*, mas depende que toda a informação seja previamente definida para que só então possa ser levada ao *Product Backlog*. Trata-se de um processo *plan-driven* no sentido de que os requisitos de outras áreas precisam estar definidos e fechados para que só então possam chegar ao time *Scrum* (PETERSEN; WOHLIN, 2010).

4.3.2 Dificuldades na adoção de *Scrum*

Cinco entrevistados responderam às questões fechadas sobre dificuldades de *Scrum* na Empresa C. Estas questões são apresentadas no Apêndice D deste documento, e cada uma das respostas coletadas são encontradas no Apêndice G.

A Tabela 3 apresenta a consolidação e ordenação das respostas por média.

Tabela 3 – Respostas para análise do questionário fechado da Empresa C (Continua).

Número	Questão	Tema relacionado	Média
2	Minha empresa detalha todos os requisitos/itens do *backlog* para que só então o time de desenvolvimento possa começar a desenvolver.	*backlog*/requisitos em processo *plan-driven*	3,8
4	Minha empresa só realiza a reunião de planejamento (*Spring Planning*) quando o entendimento dos requisitos é total e está devidamente formalizado.	reunião de planejamento em processo *plan-driven*	3,4
8	O time *Scrum* não realiza reunião de revisão (*Spring Review*) com o cliente, mas sim com um outro departamento intermediário, que recebe as demandas.	reunião de revisão em processo *plan-driven*	3,2
19	As pessoas do time *Scrum* tendem a inspecionar o processo visando uma organização das atividades onde as tarefas de análise devem preceder as tarefas de construção, que por sua vez devem preceder as tarefas de teste.	pilar Inspeção em processo *plan-driven*	2,4
3	Minha empresa não consegue reunir todas as pessoas necessárias nas reuniões de planejamento (*Spring Planning*) por conta de compromissos das pessoas com tarefas de seus departamentos de origem.	reunião de planejamento em estrutura não-projetizada	2,4
7	O time *Scrum* não realiza reunião de revisão (*Spring Review*) pois não há interesse do cliente ou há desconhecimento sobre quem é o cliente ou departamento representante do cliente para o produto sendo desenvolvido.	reunião de revisão em estrutura não-projetizada	2,2
14	O *Scrum Master* da minha empresa tem dificuldades no cumprimento do processo *Scrum* por conta da falta de apoio de outros departamentos, que nem sempre respeitam os limites e regras do processo *Scrum*.	papel de *Scrum Master* em processo *plan-driven*	1,8
10	O time *Scrum* tem dificuldades na realização de uma boa reunião de retrospectiva (*Sprint Retrospective*) porque as melhorias no processo são discutidas de forma fragmentada visando interesses de diferentes áreas, e não visam a melhoria do processo para o time como um todo.	reunião de retrospectiva em processo *plan-driven*	1,6
5	Os profissionais do time *Scrum* não conseguem comparecer às reuniões diárias (*Scrum Daily Meetings*) por terem conflito de agenda por conta de compromissos em seus próprios departamentos.	reunião diária em estrutura não-projetizada	1,4
1	Minha empresa tem um departamento/pessoa responsável por requisitos/*backlog*, não relacionado com o time *Scrum*.	*backlog*/requisitos em estrutura não-projetizada	1,4
11	Minha empresa detalha todos os requisitos/itens do *backlog* para que só então o time de desenvolvimento possa começar a desenvolver.	papel de *Product Owner* em estrutura não-projetizada	1,4
9	O time *Scrum* não realiza reunião de retrospectiva (*Spring Retrospective*) porque não ha muito interesse na melhoria do processo ágil, já que as pessoas do time têm processos diferentes em seus próprios departamentos.	reunião de retrospectiva em estrutura não-projetizada	1,4
6	Os profissionais do time *Scrum* não participam das reuniões diárias (*Scrum Daily Meetings*) porque não vêem utilidade, já que as tarefas dos analistas já foram feitas antes do *Sprint*, ou as tarefas dos testadores só irão ocorrer após o *Sprint*.	reunião diária em processo *plan-driven*	1,2
12	Minha empresa não consegue fazer com que o *Product Owner* esteja envolvido com o time de desenvolvimento nos momentos em que a interação faz-se necessária.	papel de *Product Owner* em processo *plan-driven*	1,2

Tabela 3 – Respostas para análise do questionário fechado da Empresa C (Conclusão).

Número	Questão	Tema relacionado	Média
13	Minha empresa tem dificuldade em adotar o papel de *Scrum Master* porque este papel não existe formalmente em nenhum departamento de TI.	papel de *Scrum Master* em estrutura não-projetizada	1,2
15	Minha empresa tem dificuldades em manter o time de desenvolvimento trabalhando junto na iteração, pois membros do time precisam também trabalhar em tarefas de seus departamentos de origem.	papel do time de desenvolvimento em estrutura não-projetizada	1,2
16	O time de desenvolvimento entende que as tarefas do time multidisciplinar não podem ser feitas em paralelo, mas sim devem ser feitas em cascata, iniciando pelos requisitos, para que depois sejam feitas as tarefas de construção e só então as tarefas de teste.	papel do time de desenvolvimento em processo *plan-driven*	1,0
17	Minha empresa tem dificuldades em deixar claros os papeis e responsabilidades das pessoas no cenário onde seu trabalho é dividido entre atuar no time *Scrum* e em seu próprio departamento.	pilar Transparencia em estrutura não-projetizada	1,0
18	As pessoas do time *Scrum* tendem a inspecionar seu processo considerando mais o ponto de vista do seu departamento original do que o ponto de vista do time como um todo.	pilar Inspeção em estrutura não-projetizada	1,0
20	Minha empresa não aplica melhorias ao processo pois valoriza o método *Scrum* menos do que seus próprios processos departamentais individuais.	pilar Adaptação em estrutura não-projetizada	1,0
21	Minha empresa tem dificuldade em manter um *timebox* no *Sprint*, pois membros do time constantemente são interrompidos para resolver problemas de seus departamentos de origem.	*timebox* em estrutura não-projetizada	1,0

Fonte: Próprio autor

4.3.3 Interpretação dos resultados

Esta seção analisa os dados obtidos. Foram selecionadas as cinco questões do questionário fechado que obtiveram a maior média. O tema relacionado a cada uma destas cinco questões foi então analisado, considerando os elementos levantados pelos questionários abertos e a revisão bibliográfica.

4.3.3.1 Backlog/requisitos em processo *plan-driven*

O gerenciamento do *Backlog* cabe ao profissional com papel de *Product Owner,* que pode eventualmente delegá-lo para o time de desenvolvimento, ainda que permaneça responsável por ele (SCHWABER; SUTHERLAND, 2013).

Em processos *plan-driven* os requisitos devem ser totalmente especificados antes de sua construção (PETERSEN; WOHLIN, 2010).

Na Empresa C, todos os requisitos são detalhados por áreas externas ao Time *Scrum*, para que só então possam chegar ao *Product Owner*, que limita-se a apresentá-los aos desenvolvedores. Com isto, o time de desenvolvimento alterna entre momentos de muito trabalho (quando todos os requisitos estão completamente detalhados e precisam ser construídos imediatamente) e de trabalho nenhum (quando os desenvolvedores estão aguardando a definição completa de todos os requisitos). O ideal seria que o *Product Owner* tivesse autoridade suficiente para exigir uma lista de requisitos priorizados entre todas as áreas demandantes, sem nenhum detalhamento, e que a partir daí obtivesse o detalhamento necessário para cada requisito de acordo com a prioridade. Desta forma, poderia alimentar o *Backlog* do time de forma que a construção fosse mais contínua e estável, respeitando assim os limites do *timebox* das iterações.

O fato da empresa não detalhar os requisitos de acordo com sua prioridade e não dar poder ao *Product Owner* para realizar este gerenciamento são dois fatores que dficultam a adoção de *Scrum*.

4.3.3.2 Reunião de planejamento em processo *plan-driven*

A reunião de planejamento é um evento com o máximo de oito horas de duração (para uma iteração de um mês), realizada no início da iteração e que visa identificar o que pode ser feito durante a iteração e como o trabalho necessário será realizado. A entrada desta reunião é o *Backlog* do produto priorizado. O time de desenvolvimento determina o objetivo da iteração, selecionando itens do *Backlog* do produto que poderão ser construídos e entregues. A saída da iteração é, no mínimo, uma explicação, pelo time de desenvolvimento ao *Scrum Master* e ao *Product Owner*, sobre como irá se organizar para completar o objetivo da iteração (SCHWABER; SUTHERLAND, 2013).

Em processos *plan-driven* os requisitos precisam ser fechados e formalizados para que só então os desenvolvedores possam implementá-los (PETERSEN; WOHLIN, 2010).

Na Empresa C, os requisitos precisam ser formalizados e assinados para que só então possam entrar no *Backlog*. Apenas depois de entrar no *Backlog* os requisitos podem ir para a reunião de planejamento, e entrarem na iteração para construção. Identificaram-se, na entrevista, cenários onde o entendimento do requisito estava claro e o item poderia entrar no *Backlog* para análise pelos desenvolvedores na reunião de planejamento, mas isto não ocorreu porque o requisito não foi formalizado. Com isto, o time *Scrum* acabou tendo que aguardar as formalizações, atrasando o inícios de seus trabalhos.

O fato da empresa necessitar a formalização completa do requisito para que só então o time possa iniciar o planejamento de sua construção, mesmo que o requisito esteja claro para o *Product Owner*, é um fator que dificulta a adoção do *Scrum*.

4.3.3.3 Reunião de revisão em processo *plan-driven*

O método *Scrum* espera que a reunião de revisão aconteça no final da iteração para inspecionar o incremento de *software* sendo entregue, bem como para adaptar o *Backlog* do produto se necessário. Durante a revisão da iteração, o time *Scrum* e as partes interessadas devem colaborar sobre o que foi realizado na iteração. Com base nisto e em qualquer mudança no *Backlog* do produto durante a iteração, os participantes colaboram nas próximas coisas que podem ser feitas para otimizar valor. Esta é uma reunião informal (não uma reunião de status), e a apresentação do incremento destina-se a motivar e obter comentários e promover a colaboração (SCHWABER; SUTHERLAND, 2013).

Na Empresa C, a reunião de revisão não é sempre adotada, pois os times *Scrum* (restritos aos departamentos de desenvolvimento e infra-estrutura) têm dificuldade de identificar e de envolver os *stakeholders* (de outros departamentos). Contudo, quando ocorre, a reunião não é feita para o cliente final, e sim para uma outra área, distinta do desenvolvimento, que recebe as demandas e só então as repassa para uso pelos clientes. O método *Scrum* não vê isto como uma restrição, afirmando que a reunião de revisão pode receber *stakeholders* chave, e não necessariamente o cliente final (SCHWABER; SUTHERLAND, 2013).

O fato da reunião de revisão não envolver o cliente final diretamente é algo considerado válido pelo método *Scrum*, de forma que não representa um item de conflito ou de dificuldade em sua adoção. Contudo, o fato da reunião de revisão não acontecer sempre é um problema para a adoção do *Scrum*.

4.3.3.4 Pilar Inspeção em processo *plan-driven*

A Inspeção é um dos três pilares do *Scrum* (além da Transparência e da Adaptação), que visa acompanhar o progresso do time e apontar melhorias e necessidades de ajustes. Deve ser adotado de forma que não seja tão frequente a ponto de atrapalhar a execução das tarefas da iteração. A reunião de retrospectiva é a oportunidade adequada para consolidação dos pontos de inspeção do método e para os devidos ajustes para a próxima iteração (SCHWABER; SUTHERLAND, 2013).

Os entrevistados consideram que as pessoas dos times *Scrum* da Empresa C inspecionam o método de seu time de forma a considerar uma ordem de execução de tarefas que obedece a uma organização *plan-driven*: as tarefas de análise devem preceder as de desenvolvimento, que devem preceder as de teste. O *Scrum* sugere que os times tem a liberdade de auto-organizar suas tarefas internas do time. Adotar um modelo de tarefas internas ao time que siga um modelo linear pode afetar a produtividade, onde existem tarefas que podem ser executadas em paralelo.

A auto-organização dos times *Scrum* em tarefas que obedecem a um conceito *plan-driven* não é um problema da estrutura organizacional ou do processo da empresa, mas sim do entendimento dos próprios integrantes do time acerca da liberdade que possuem para a melhor forma de auto-gestão dentro do time.

4.3.3.5 Reunião de planejamento em estrutura não-projetizada

Na reunião de planejamento, o time *Scrum* se reune para selecionar, de forma conjunta, quais os itens do *Backlog* do produto que podem ser desenvolvidos na iteração. Nesta reunião, que dura um máximo de oito horas (para uma iteração de um mês), todos os desenvolvedores colaboram para definir o objetivo do *Sprint*, e compartilham a responsabilidade de entregar os requisitos selecionados. Neste

evento, o *Product Owner* tem a responsabilidade de apresentar o *Backlog* do produto, esclarecendo as dúvidas dos desenvolvedores (SCHWABER; SUTHERLAND, 2013).

Na Empresa C, os desenvolvedores estão todos abaixo de uma mesma gestão, de forma que conseguem trabalhar com certo nível de auto-organização e multidisciplinaridade em times. Contudo, o *Product Owner* não está na mesma gestão, e a reunião de planejamento não ocorre corretamente por conta da indisponibilidade deste profissional. Observou-se em entrevista que frequentemente este profissional não está disponível para as reuniões de planejamento, por conta de outras prioridades de seu departamento de origem. Isto atrasa o início da iteração do time, pois sem o *Product Owner* não há quem possa apresentar e esclarecer dúvidas do *Backlog*, e assim sendo não é posível realizar a reunião de planejamento e nem iniciar a iteração.

O fato do *Product Owner* nem sempre poder comparecer às reuniões de planejamento do time *Scrum* por conta de outras prioridades, impedindo que os desenvolvedores tenham elementos para realizar a reunião de planejamento, é uma característica que dificulta a adoção do *Scrum*.

5 ANÁLISE COMPARATIVA DOS RESULTADOS

Esta seção apresenta uma análise comparativa das Empresas A, B e C, com base nos dados individuais de cada empresa, previamente analisados.

5.1 Análise numérica das empresas

A média é um indicador geral da opinião dos entrevistados acerca das questões. Nesta pesquisa, o valor da média é diretamente proporcional à percepção dos entrevistados sobre a dificuldade da adoção do *Scrum* em sua empresa.

Conforme apresentado na Tabela 2, a Empresa B possui várias questões avaliadas entre 4,5 e 5,0, sendo a empresa que obteve as maiores médias. Já a Empresa C, cujos dados do questionário consolidados são apresentados na Tabela 3, apresentou os menores números, com diversos valores na faixa de 1,0. Observa-se, portanto, que os profissionais da Empresa B apresentam maior insatisfação na adoção do *Scrum,* seguidos pelos profissionais da Empresa A, que consideram a adoção mais equilibrada, de acordo com os dados apresentados na Tabela 1. Os profissionais da Empresa C consideram que a adoção de *Scrum* está mais adequada, embora várias dificuldades sejam observadas, com médias próximas de 4,0.

A análise numérica aqui descrita não possui pretensões estatísticas, buscando servir apenas como referência para a análise qualitativa dos resultados.

5.2 Análise comparativa de temas críticos

Foi realizada uma seleção de cinco questões de cada empresa, pelo critério da maior média. Os temas relativos a estas questões foram agrupados para comparação, sendo apresentados na Tabela 4.

Tabela 4 – Temas de maior dificuldade de adoção de *Scrum* nas empresas.

Tema	Empresa A	Empresa B	Empresa C
Pilar Inspeção em processo *plan-driven*	*X*		*X*
Backlog/requisitos em processo *plan-driven*	*X*		X
Backlog/requisitos em estrutura não-projetizada	*X*		
Pilar Inspeção em estrutura não-projetizada	*X*		
Reunião de revisão em processo *plan-driven*	*X*		*X*
Papel de Scrum Master em estrutura não-projetizada		*X*	
Papel de Product Owner em estrutura não-projetizada		*X*	
Papel de Scrum Master em processo *plan-driven*		*X*	
Pilar Transparência em estrutura não-projetizada		*X*	
Timebox em estrutura não-projetizada		*X*	
Reunião de planejamento em processo *plan-driven*			*X*
Reunião de planejamento em estrutura não-projetizada			*X*

Fonte: Próprio autor

Embora as três empresas sejam similares em termos de estrutura e processo, as respostas às questões sugerem que possuem perspectivas muito distintas em relação a como adotam *Scrum*. Não há unanimidade sobre nenhum tema entre as empresas, e há concordância de temas entre duas empresas em apenas três dos doze temas considerados.

As Empresas A e C possuem características bastante distintas. Atuam em ramos de negócio diferentes (energia e terceirização de e-commerce, respectivamente), tem portes diferentes (a primeira é uma grande multinacional e a

segunda é uma empresa com poucos funcionários) e ainda que possuam uma estrutura não-projetizada e um processo *plan-driven*, estes elementos são adotados de forma bastante diferente (a estrutura da Empresa A é hierarquicamente mais rígida e o processo *plan-driven* da Empresa A é mais rigidamente estabelecido). Ainda assim, ambas as empresas compartilham três temas da pesquisa, relacionados ao processo *plan-driven*: o pilar Inspeção, a reunião de revisão e *Backlog*/requisitos.

Em ambas as empresas, as dificuldades observadas em relação ao pilar Inspeção não estão diretamente relacionadas com a estrutura não-projetizada ou com o processo *plan-driven* (como era esperado), mas sim com o próprio conhecimento dos profissionais acerca das possibilidades do método *Scrum* e das recomendações dos métodos ágeis. Os times tem a possibilidade de se auto-organizar de forma diferente, mas acabam optando por adotar a mesma orientação *plan-driven*, separando as tarefas de uma disciplina para serem executadas antes de outra, de forma sequencial, quando poderiam ser executadas em paralelo.

Outro tema comum a estas duas empresas é relacionado à reunião de revisão em processo *plan-driven*. Os entrevistados de ambas as empresas apontam que não apresentam seu trabalho diretamente para o cliente, mas sim para uma outra área ou departamento, que faz o papel de interface com o cliente. Contudo, o *Scrum* não reconhece isto como sendo um problema.

As Empresas B e C são mais similares no sentido de que são grandes multinacionais, empregando milhares de funcionários, e com estruturas hierárquicas mais rígidas. Contudo, não compartilham nenhum tema nesta pesquisa.

5.3 Apresentação de dados gerais consolidados

A Tabela 5 apresenta os dados consolidados do questionário fechado, considerando os resultados de todas as questões para todas as empresas, ordenados pela maior média geral.

Tabela 5 – Respostas para análise do questionário fechado de todas as empresas (Continua).

Número	Questão	Tema Relacionado	Média Empresa A	Média Empresa B	Média Empresa C	Média Geral
8	O time *Scrum* não realiza reunião de revisão (*Spring Review*) com o cliente, mas sim com um outro departamento intermediário, que recebe as demandas.	reunião de revisão em processo *plan-driven*	3,3	4,2	3,2	3,6
19	As pessoas do time *Scrum* tendem a inspecionar o processo visando uma organização das atividades onde as tarefas de análise devem preceder as tarefas de construção, que por sua vez devem preceder as tarefas de teste.	pilar Inspeção em processo *plan-driven*	3,9	3,8	2,4	3,4
2	Minha empresa detalha todos os requisitos/itens do backlog para que só então o time de desenvolvimento possa começar a desenvolver.	*backlog*/requisitos em processo *plan-driven*	3,8	2,6	3,8	3,4
14	O *Scrum Master* da minha empresa tem dificuldades no cumprimento do processo *Scrum* por conta da falta de apoio de outros departamentos, que nem sempre respeitam os limites e regras do processo *Scrum*.	papel de *Scrum Master* em processo *plan-driven*	3,1	4,8	1,8	3,2
11	Minha empresa tem dificuldades em adotar o papel de *Product Owner* porque este papel não existe formalmente em nenhum departamento.	papel de *Product Owner* em estrutura não-projetizada	3,2	4,8	1,4	3,1
3	Minha empresa não consegue reunir todas as pessoas necessárias nas reuniões de planejamento (*Spring Planning*) por conta de compromissos das pessoas com tarefas de seus departamentos de origem.	reunião de planejamento em estrutura não-projetizada	3,0	3,9	2,4	3,1
12	Minha empresa não consegue fazer com que o *Product Owner* esteja envolvido com o time de desenvolvimento nos momentos em que a interação faz-se necessária.	papel de *Product Owner* em processo *plan-driven*	3,1	4,5	1,2	3,0
4	Minha empresa só realiza a reunião de planejamento (*Spring Planning*) quando o entendimento dos requisitos é total e está devidamente formalizado.	reunião de planejamento em processo *plan-driven*	3,1	2,3	3,4	2,9
18	As pessoas do time *Scrum* tendem a inspecionar seu processo considerando mais o ponto de vista do seu departamento original do que o ponto de vista do time como um todo.	pilar Inspeção em estrutura não-projetizada	3,3	4,4	1,0	2,9
20	Minha empresa não aplica melhorias ao processo pois valoriza o método *Scrum* menos do que seus próprios processos departamentais individuais.	pilar Adaptação em estrutura não-projetizada	3,2	4,5	1,0	2,9
10	O time *Scrum* tem dificuldades na realização de uma boa reunião de retrospectiva (*Sprint Retrospective*) porque as melhorias no processo são discutidas de forma fragmentada visando interesses de diferentes áreas, e não visam a melhoria do processo para o time como um todo.	reunião de retrospectiva em processo *plan-driven*	2,9	4,0	1,6	2,8
17	Minha empresa tem dificuldades em deixar claros os papeis e responsabilidades das pessoas no cenário onde seu trabalho é dividido entre atuar no time *Scrum* e em seu próprio departamento.	pilar Transparencia em estrutura não-projetizada	2,5	4,8	1,0	2,8

Tabela 5 – Respostas para análise do questionário fechado de todas as empresas (Conclusão).

Número	Questão	Tema relacionado	Média Empresa A	Média Empresa B	Média Empresa C	Média Geral
13	Minha empresa tem dificuldade em adotar o papel de *Scrum Master* porque este papel não existe formalmente em nenhum departamento de TI.	papel de *Scrum Master* em estrutura não-projetizada	2,3	5,0	1,3	2,8
7	O time *Scrum* não realiza reunião de revisão (*Spring Review*) pois não há interesse do cliente ou há desconhecimento sobre quem é o cliente ou departamento representante do cliente para o produto sendo desenvolvido.	reunião de revisão em estrutura não-projetizada	2,6	3,6	2,2	2,8
1	Minha empresa tem um departamento/pessoa responsável por requisitos/backlog, não relacionado com o time *Scrum*.	*backlog*/requisitos em estrutura não-projetizada	3,5	3,2	1,4	2,7
21	Minha empresa tem dificuldade em manter um *timebox* no *Sprint*, pois membros do time constantemente são interrompidos para resolver problemas de seus departamentos de origem.	*timebox* em estrutura não-projetizada	2,3	4,5	1,0	2,6
16	O time de desenvolvimento entende que as tarefas do time multidisciplinar não podem ser feitas em paralelo, mas sim devem ser feitas em cascata, iniciando pelos requisitos, para que depois sejam feitas as tarefas de construção e só então as tarefas de teste.	papel do time de desenvolvimento em processo *plan-driven*	3,0	3,8	1,0	2,6
15	Minha empresa tem dificuldades em manter o time de desenvolvimento trabalhando junto na iteração, pois membros do time precisam também trabalhar em tarefas de seus departamentos de origem.	papel do time de desenvolvimento em estrutura não-projetizada	1,6	4,3	1,2	2,4
9	O time *Scrum* não realiza reunião de retrospectiva (*Spring Retrospective*) porque não ha muito interesse na melhoria do processo ágil, já que as pessoas do time têm processos diferentes em seus próprios departamentos.	reunião de retrospectiva em estrutura não-projetizada	1,8	3,3	1,4	2,2
5	Os profissionais do time *Scrum* não conseguem comparecer às reuniões diárias (*Scrum Daily Meetings*) por terem conflito de agenda por conta de compromissos em seus próprios departamentos.	reunião diária em estrutura não-projetizada	2,2	2,2	1,4	1,9
6	Os profissionais do time *Scrum* não participam das reuniões diárias (*Scrum Daily Meetings*) porque não vêem utilidade, já que as tarefas dos analistas já foram feitas antes do Sprint, ou as tarefas dos testadores só irão ocorrer após o Sprint.	reunião diária em processo *plan-driven*	1,8	2,8	1,2	1,9

Fonte: Próprio autor

5.4 Conclusões da análise

A análise comparativa entre as três empresas permitiu observar que:

- Embora todas as empresas possuam estrutura não-projetizada e processo *plan-driven*, possuem problemas distintos em relação à adoção de *Scrum*, com poucos pontos de similaridade. Mesmo empresas similares em porte e rigidez hierárquica geralmente não apresentam as mesmas dificuldades;
- Os problemas observados na adoção de *Scrum* nas empresas nem sempre têm relação direta com processo e estrutura organizacional. Mesmo em cenários onde os times tiveram autonomia para tomada de decisão (como na organização interna dos times), optaram por práticas que não estão em conformidade com os valores do *Scrum* e dos métodos ágeis, apoiando-se em práticas relacionadas à abordagem *plan-driven* mais tradicional;

As empresas desta pesquisa, estruturadas de forma não-projetizada e que empregam processo *plan-driven,* entendem que os temas de maior dificuldade na adoção do *Scrum* estão relacionados, em ordem de dificuldade, aos seguintes pontos:

- Ritos do *Scrum,* em termos da correta execução da Reunião de revisão, considerando um processo *plan-driven:* Este ponto foi considerado um dos cinco mais inadequados pelos respondentes das Empresas A e C. Contudo, embora a reunião de revisão não aconteça com o cliente, a mesma ocorre com seu(s) representante(s), o que é considerado uma prática válida pelo *Scrum*;
- Pilares do *Scrum,* em termos da correta aplicação do Pilar Inspeção, considerando um processo *plan-driven*: Este ponto também foi

considerado pelas Empresas A e C como um dos cinco que apresentam maior inadequação ao *Scrum*. Os desenvolvedores do time organizam suas tarefas internas de forma que as tarefas de requisitos precedam as de construção, que por sua vez precedem as de teste, quando as tarefas poderiam ser paralelizadas. Este pensamento linear, característico da perspectiva *plan-driven*, não ocorre por imposição de processo ou estrutura da empresa, mas sim na própria organização interna dos times, por opção dos desenvolvedores. Trata-se de outro cenário onde a dificuldade na adoção do *Scrum* está diretamente relacionada ao mal-entendimento dos valores dos métodos ágeis e da proposta do *Scrum*, e não a limitações impostas pela empresa;

- Artefatos do *Scrum,* em termos da correta elaboração e gerenciamento de um *backlog* de produto, considerando um processo *plan-driven*: Este ponto também foi considerado pelas Empresas A e C como um dos cinco que apresentam maior inadequação ao *Scrum*. Em ambas as empresas, a elaboração de requisitos não era responsabilidade do time *Scrum*, e sim de outro departamento. O *Product Owner*, neste contexto, ou não era um papel existente ou era um papel que tinha muito pouca autoridade para questionar os requisitos, esclarecê-los propriamente, e priorizá-los. Em quaisquer dos cenários, o problema observado foi o mesmo: o time recebia requisitos pouco claros e sem prioridade, e precisava assumi-los para não atrasar o início da iteração. Como resultado, os times entregavam funcionalidades que não atendiam completamente às expectativas do cliente;
- Papeis do *Scrum,* em termos da correta adoção dos papeis de *Product Owner* (em uma estrutura não-projetizada) e de *Scrum Master* (em um processo *plan-driven*): os problemas em ambos os papeis foram apontados pela Empresa B como sendo um dos cinco elementos que causam as maiores dificuldades na adoção do *Scrum*. De forma geral, os

> times que tentam adotar *Scrum* na Empresa B sofrem constantes interrupções para atender necessidades emergenciais e de outros setores, e os papeis de *Scrum Master* e *Product Owner* não tem o apoio necessário da alta gestão para sustentar o método ágil.

Pode-se observar, com base nesta análise, que a percepção sobre as duas maiores dificuldades na adoção do *Scrum* está equivocada:

- o maior ponto de dificuldade está relacionado à não-realização de reunião de revisão com o cliente, mas o *Scrum* não define que deve ser necessariamente realizado com o cliente, mas sim com ele ou seu(s) representante(s), o que ocorre;
- o segundo maior ponto está relacionado com a organização do processo interno do time, que adota a perspectiva *plan-driven* para organização sequencial de tarefas, quando nada os obriga a isto.

Também pode-se observar que a percepção sobre as três dificuldades seguintes está relacionada a requisitos, papeis e responsabilidades:

- requisitos: em relação a seu adequado gerenciamento, priorização e esclarecimento. Dificuldades existem pelo fato desta disciplina estar fora do time;
- papeis e responsabilidades: em relação ao correto incentivo e apoio aos profissionais envolvidos, para que os papeis do *Scrum* possam ser desempenhados de forma adequada.

6 CONCLUSÃO

A adoção de *Scrum* em sua forma ideal não é realizada pelas empresas, que precisam realizar customizações para adequar o método *Scrum* à sua realidade. Considerando esta premissa, este trabalho de pesquisa propôs-se a estudar as dificuldades na adoção de *Scrum* em empresas com características similares em estrutura e processo. A pergunta de pesquisa que procurou ser respondida foi a seguinte: “como empresas brasileiras não-projetizadas e que desenvolvem software com abordagem *plan-driven* têm percebido dificuldades ao adotar o método *Scrum*?”

Três empresas diferentes participaram do estudo de caso. Estas empresas possuíam características distintas (como porte, ramo de negócio, número de funcionários), mas eram similiares no sentido que compartilhavam estrutura e processo parecidos, ao mesmo tempo em que realizavam iniciativas para adoção de *Scrum*.

Os estudos desta pesquisa (a revisão bibliográfica, a interpretação e análise dos resultados de entrevistas e questionários e o estudo das conclusões) apontaram indícios de que a adoção do *Scrum* sofreu influência direta de características da estrutura organizacional e do processo *plan-driven,* produzindo dificuldades, conforme esperado. Também constatou-se, conforme esperado, que esta influência não foi total, mas sim parcial: cada empresa realizava o *tailoring* de seu método *Scrum* de acordo com o possível dentro das restrições de seu ambiente, de forma que esperava-se que alguns elementos do *Scrum* poderiam ser incorporados com sucesso e outros não (os dados de média geral da Tabela 5 comprovaram isto, com variações entre 1,9 e 3,6). Contudo, não era esperado constatar que houvessem dificuldades significativas não relacionadas ao processo e à estrutura em si, mas sim a falhas de entendimento do próprio método *Scrum* e dos valores e princípios dos métodos ágeis pelos profissionais envolvidos.

Dois elementos principais foram observados como estando diretamente relacionados às maiores dificuldades de adoção do *Scrum*:

i. a capacidade dos profissionais em realizar a adoção de forma correta: uma correta adoção do *Scrum* deve aplicar o *tailoring* ao método *Scrum* com base em conhecimento teórico fundamentado, além do entendimento das características únicas da empresa. Além disto, os conceitos, práticas, ritos e valores do *Scrum* precisam ser disseminados e acompanhados de perto, por meio de um trabalho de ensino e evangelização dos profissionais responsáveis pela adoção aos profissionais atuantes no desenvolvimento de *software*;
ii. o nível de apoio que a empresa concede aos profissionais: a empresa precisa conceder poder, na medida do possível, aos profissionais envolvidos com o método *Scrum.* O ideal é que os profissionais possam atuar em times auto-organizados e multi-funcionais que trabalhem juntos visando um objetivo claro, e a concessão de poder vem no sentido de garantir esta organização. O *tailoring* do método *Scrum* precisa ser aplicado de acordo com o nível de poder concedido pela empresa aos profissionais.

Em termos de contribuição teórica, esta pesquisa analisa a adoção de *Scrum* em cenários bastantes específicos de processo e estrutura, avaliando as dificuldades da adoção num contexto comum e real de mercado. Este tema não é muito explorado, embora entenda-se ser de extrema importância para empresas que têm interesse em adotar *Scrum* mas não podem alterar sua estrutura ou processo para tal fim.

Em termos de aplicação prática, a pesquisa deste trabalho pode ser replicada em empresas não-projetizadas e de processo *plan-driven* para ajudar a determinar

as maiores dificuldades de adoção de *Scrum*. Ao interpretar os resultados, as empresas podem considerar a devida concessão de poder aos profissionais envolvidos, bem como a devida capacitação em *Scrum,* para que possa ser realizado um *tailoring* que equilibre os limites impostos pelas características da empresa e as premissas do método *Scrum*.

Esta pesquisa pode ser estendida academicamente pela análise dos mecanismos para realização de *tailoring* do método *Scrum* dentro dos limites de uma organização com estrutura não-projetizada, considerando o grau de poder concedido e o nível de capacitação teórica e prática de *Scrum* requeridos. Também pode ser estendida pela aplicação em métodos ágeis diferentes do *Scrum*, como o *XP*. Outro caso possível está voltado à pesquisa por dados mais quantitativos em relação às principais dificuldades aqui apresentadas.

REFERÊNCIAS

ABRAHAMSSON, P; CONBOY, K; YANG, X. ‘Lot’s done, more to do’: the Current State of Agile System Development Research. **European Journal of Information Systems**, v.18, n.4, p.281-284, Aug. 2009.

AHMED, F.; ROBINSON, S.; TAKO, A. A. Using the Structured Analysis and Design Technique (SADT) in Simulation Conceptual Modeling. In: Winter Simulation Conference, 2014, Savanna, Estados Unidos. **Proceedings...** IEEE Press, 2014. p. 1038-1049

BARDIN, L. **Análise de Conteúdo**. Edições 70, 2004. 229p.

BASS, J. Activities in Scrum Master Teams: Process Tailoring in Large Enterprise Projects. In: 9th International Conference on Global Software Engineering, 2014, Shanghai, China. **Proceedings...** IEEE Computer Society, ICGS, 2014. p. 6-15

BECK, K. Embracing Change with Extreme Programming. **IEEE Computer Society**, v.32, n.10, p.70-77, Oct. 1999.

BECK, K.; BOEHM, B. Agile throught Discipline: A Debate. **IEEE Computer Society**, v.36, n.6, p.44-46, Jun. 2003.

BENEFIELD, G. Rolling out Agile in a large enterprise. In: HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES, 2008, Waikoloa, Hawaii, United States. **Proceedings...** of the 41st Annual. p. 461

BOEHM, B. W. A Spiral Model of Software Development and Enhancement. **Computer Magazine**, v.21, n.5, p.61-72, May 1988.

BOEHM, B. W; TURNER, R. **Balancing Agility and Discipline: A Guide for the Perplexed**. Addison-Wesley/Pearson Education, 2003. 304p.

COHN, M. **User Stories Applied: For Agile Software Development**. Addison Wesley Longman Publishing Co., Inc., 2004. 268p.

COHN, M.; FORD, D. Introducing an Agile Process to an Organization. **IEEE Computer Society**, v.36, n.6, p.74-78, June. 2003.

DINGSOYR et al. A decade of Agile Methodologies: Towards explaining Agile Software Development. **The Journal of Systems and Software**, v.85, n.6, p.1213-1221, Jun. 2012.

DINSMORE, P.; CABANIS-BREWIN, J. **The AMA Handbook of Project Management**. AMACOM, 2006. 512p.

EISENHARDT, K. M. Building Theories from Case Study Research. **Academy of Management Review**, v.14, n.4, p.532-550, Oct. 1989.

FEILER, P. H.; HUMPHREY, W. S. Software Process Development and Enactment: Concepts and Definitions. In: SECOND INTERNATIONAL CONFERENCE ON THE SOFTWARE PROCESS, 1993, Berlin, Germany. **Proceedings...** of the 2nd International Conference on the Software Process. p.28-40.

FOWLER, M.; HIGHSMITH, J. The Agile Manifesto. **Software Development Magazine**, v.9, p.28-35, Aug. 2001.

GLASER, B.; STRAUSS, A. **The Discovery of Grounded Theory: strategies for qualitative research.** Chicago: Aldine Transaction, 1967. 271p.

GREN, L.; TORKAR, R.; FELDT, R. Work Motivational Challenges Regarding the Interface Between Agile Teams and a Non-Agile Surrounding Organization: A case study. In: AGILE CONFERENCE, 2014, Florida, United States. **Proceedings...** IEEE Computer Society, AGILE'14, 2014. p.11-15.

HAUGEN, N. An Empirical Study of Using Planning Poker for User Story Estimation. In: AGILE CONFERENCE, 2006, Minneapolis, United States. **Proceedings...** IEEE Computer Society, AGILE'06, 2006. p.34-43.

HIGHSMITH, J. What is Agile Software Development? **Crosstalk: The Journal of Defense Software Engineering**, v.15, n.10, p.4-9, Oct. 2002.

HIRSCH, M. Moving from a Plan Driven Culture to Agile Development. In: ICSE '05 INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 2005, St. Louis, United States. **Proceedings...** ICSE '05 Proceedings of the 27th International Conference on Software Engineering, 2005. p.38.

HODA, R. **Self-Organizing Agile Teams**: A Grounded Theory. 2011. 262 f. Tese (Doutorado) – Victoria University of Wellington, Wellington, Nova Zelândia, 2011.

HODA, R.; NOBLE, J.; MARSHALL, S. Developing a grounded theory to explain the practices of self-organizing Agile teams. **Empirical Software Engineering**, v.17, n.6, p.609-639, Dec. 2012.

HUMPHREY, W. S.; SNYDER, T. R.; WILLIS, R. R. Software Process Improvement at Hughes Aircraft. **IEEE Software**, v.8, n.4, p.11-23, July 1991.

IEEE COMPUTER SOCIETY. **SWEBOK v3.0 – Guide to the Software Engineering Body of Knowledge**. United States, 2014. 335p.

ISO. **ISO/IEC 12207:2008:** Systems and software engineering – Software life cycle processes. Genebra, Suíça, 2008. 124p.

JICK, T. D. Mixing Qualitative and Quantitative Methods: Triangulation in Action. **Administrative Science Quaterly**, v.24 n.4, p.602-611, Dec. 1979.

KERZNER, H. **Project Management: A Systems Approach to Planning, Scheduling and Controlling**. John Wiley & Sons, Inc., 2009. 1094p.

LARMAN, C. **Agile and Iterative Development: A Manager's Guide**. Addison-Wesley Professional, 2003. 368p.

LARMAN, C.; BASILI, V. R. Iterative and Incremental Development: A Brief History. **IEEE Computer Society**, v.36, n.6, p.47-56, June 2003.

LEFFINGWELL, D. **Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise**. Addison-Wesley Professional, 2011. 560p.

NERUR, S.; MAHAPATRA, R.; MANGALARAJ, G. Challenges of Migrating to Agile Methodologies. **Communications of the ACM**, v.48, n.5, p.73-78, May 2005.

NIKITINA, N.; KAJKO-MATTSON, M. Guiding the Adoption of Software Development Methods. In: ICSSP '14 INTERNATIONAL CONFERENCE ON SOFTWARE AND SYSTEM PROCESS, 2014, Nanjing, China. **Proceedings...** ACM Publications, ICCSP, 2014. p.109-118.

PAULK, M. Extreme Programming from a CMM Perspective. **IEEE Software**, v.18, n.6, p.19-26, November 2001.

PETERSEN, K; WOHLIN, C. The effect of moving from a plan-driven to an incremental software development approach with agile practices. **Empirical Software Engineering**, v.15, n.6, p.654-693, Dec. 2010.

PROJECT MANAGEMENT INSTITUTE. **A Guide to the Project Management Body of Knowledge (PMBOK Guide)**, 2013. 589p.

ROYCE, W. **Software Project Management: A Unified Framework**. Addison-Wesley Professional, 1998. 448p.

ROYCE, W. W. Managing the Development of Large Software Systems. In: Technical Papers of Western Electronic Show and Convention (WesCon), 1970, Los Angeles, United States. **Proceedings...** IEEE WESCON 26, August 1970. p.1-9.

SCHWABER, K. SCRUM Development Process. In: OOPSLA '95 Workshop on Business Object Design and Implementation, 1995, Austin, Texas, United States. **Proceedings...** Business Object Design and Implementation: OOPSLA '95 Workshop Proceedings, 1997. 167p.

SCHWABER, K.; SUTHERLAND, J. **The Scrum Guide.** 2013. 16p. Available online at:<http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf>. Accessed on: 24 Apr 2016.

SENAPATHI, M.; SRINIVASAN, A. An Empirical Investigation of the Factors Affecting Agile Usage. In: EASE '14 18th INTERNATIONAL CONFERENCE ON EVALUATION AND ASSESSMENT IN SOFTWARE ENGINEERING, 2014, London, United Kingdon. **Proceedings...** ACM Publications, EASE, 2014. article n.10.

SUTHERLAND, J. Agile Can Scale: Inventing and Reinventing SCRUM in Five Companies. **Cutter IT Journal**, v.14, n.12, p.5-11, 2001.

SVENSSON, H.; HOST, M. Introducing an Agile Process in a Software Maintenance and Evolution Organization. In: CSMR '05 9th EUROPEAN CONFERENCE ON SOFTWARE MAINTENANCE AND REENGINEERING, 2005, Manchester, UK. **Proceedings...** IEEE Computer Society, CSMR, 2005. p.256-264.

TAKEUCHI, H.; NONAKA, I. The New New Product Development Game. **Harward Business Review**, p.137-146, jan. 1986.

TOLFO, C. et al. Agile Methods and Organization Culture: Reflections about Cultural Levels. **Journal of Software Maintenance and Evolution: Research and Practice**, v.123 n.6, p.423-441, October. 2011.

VERSIONONE. **9Th Annual State of Agile Survey.** 2014. 17p. Available online at: <http://info.versionone.com/state-of-agile-development-survey-ninth.html>. Accessed on: 14 Jun 2015.

WAARDENBURG, G; VLIET, H. When agile meets the enterprise. **Information and Software Technology**, v.55, n.12, p.2154-2171, Dec. 2013.

WEST, D. Water-Scrum-Fall Is The Reality Of Agile For Most Organizations Today. **Forrester Research**, July. 2011. 17p. Available online at: <https://www.forrester.com/WaterScrumFall+Is+The+Reality+Of+Agile+For+Most+Organizations+Today/fulltext/-/E-RES60109?docid=60109>. Accessed on: 24 Apr 2016.

LEE, G.; XIA, W. Toward Agile: An Integrated Analysis of Quantitative and Qualitative Field Data on Software Development Agility. **MIS Quaterly**, v.34, n.1, p.87-114, Mar. 2010.

YIN, R. **Case Study Research: Design and Methods (Applied Social Research Methods)**. SAGE Publications, Inc., 2013. 312p.

APÊNDICE A – E-mail de Apresentação

Caro(a) Senhor(a),

Meu nome é Daniel Medeiros de Assis, e sou mestrando em Engenharia de Software pelo Instituto de Pesquisas Tecnológicas do Estado de São Paulo (IPT/USP).

Minha dissertação pesquisa as dificuldades que empresas estão encontrando na adoção do *Scrum*. O intuito é o de ajudar às empresas a entender quais são os principais pontos a serem tratados para que seja possível obter os ganhos propostos pelo *Scrum*.

Desta forma, gostaria de requisitar sua participação na resposta a algumas perguntas sobre sua percepção em relação à adoção de *Scrum* em sua empresa. De acordo com regras de proteção de dados adotadas na dissertação, sua pessoa e a empresa em questão não serão identificadas.

Agradeço desde já pela atenção e colaboração,

Atenciosamente,

Daniel Medeiros de Assis

daniel@arneam.com

APÊNDICE B – Roteiro de perguntas sobre características da empresa

De forma a realizar um estudo de caso relevante, é preciso conhecer mais sobre as características da empresa, de forma a analisar sua aderência aos pontos sendo pesquisados. Em relação às **características de sua empresa**, favor responder às seguintes questões:

1. Qual o ramo da empresa?
2. Quais produtos e serviços são oferecidos?
3. Conte uma história breve sobre sua empresa.
4. Quantos funcionários existem no total e no setor de TI?
5. Como se configura a estrutura organizacional da empresa?
6. Como você percebe o grau de formalismo nos processos em sua empresa?
7. Como as disciplinas de engenharia de software se relacionam com esta estrutura?
8. Comente sobre o histórico das tentativas de adoção do *Scrum* dentro de sua empresa.
9. Quantos projetos já foram entregues com o uso do *Scrum*?
10. Fale sobre as dificuldades na adoção de *Scrum* em sua empresa.
11. Fale sobre as expectativas ou benefícios na adoção de *Scrum* em sua empresa.

APÊNDICE C – Roteiro de perguntas sobre processo

Grandes empresas têm demonstrado cada vez mais interesse pelos ganhos sugeridos quando da adoção do *Scrum*, e o processo tradicional adotado na empresa é um fator de forte influência. Em relação às **tentativas de adoção do processo *Scrum* no setor de TI de sua empresa**, favor responder às seguintes questões:

1. Como pessoas são organizadas para realização de projetos?
2. Como os ritos do *Scrum* (planejamento da iteração, reuniões diárias, reunião de restrospectiva e reunião de revisão) são adotados?
3. Como o *Backlog* é gerenciado, priorizado, representado e acessado?
4. Como os impedimentos são removidos?
5. Como são as condições para realização do escopo da iteração dentro do *timebox* definido?
6. Como os times se organizam internamente para cumprimento das tarefas?
7. Como as influências externas ao time interferem na execução da iteração conforme planejada?
8. Como se dá a apresentação dos resultados do trabalho aos profissionais envolvidos no projeto e externos ao time?
9. Como o time se organiza para discutir e realizar melhorias no processo *Scrum*?

APÊNDICE D – Questionário sobre dificuldades na adoção do Scrum

Embora empresas observem ganhos com a adoção do *Scrum*, nem sempre conseguem adotar o método conforme proposto, por conta de características conflitantes com as práticas das empresas. Em relação às **dificuldades na adoção de Scrum no setor de TI da empresa**, qual sua opinião sobre as seguintes questões?

1. Minha empresa tem um departamento/pessoa responsável por requisitos/*backlog*, não relacionado com o time *Scrum*.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

2. Minha empresa detalha todos os requisitos/itens do *backlog* para que só então o time de desenvolvimento possa começar a desenvolver.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

3. Minha empresa não consegue reunir todas as pessoas necessárias nas reuniões de planejamento *(Spring Planning)* por conta de compromissos das pessoas com tarefas de seus departamentos de origem.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

4. Minha empresa só realiza a reunião de planejamento (*Spring Planning*) quando o entendimento dos requisitos é total e está devidamente formalizado.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

5. Os profissionais do time *Scrum* não conseguem comparecer às reuniões diárias (*Scrum Daily Meetings*) por terem conflito de agenda por conta de compromissos em seus próprios departamentos.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

6. Os profissionais do time *Scrum* não participam das reuniões diárias (*Scrum Daily Meetings*) porque não vêem utilidade, já que as tarefas dos analistas já foram feitas antes do *Sprint*, ou as tarefas dos testadores só irão ocorrer após o *Sprint*.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

7. O time *Scrum* não realiza reunião de revisão (*Spring Review*) pois não há interesse do cliente ou há desconhecimento sobre quem é o cliente ou departamento representante do cliente para o produto sendo desenvolvido.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

8. O time *Scrum* não realiza reunião de revisão (*Spring Review*) com o cliente, mas sim com um outro departamento intermediário, que recebe as demandas.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

9. O time *Scrum* não realiza reunião de retrospectiva (*Spring Retrospective*) porque não ha muito interesse na melhoria do processo ágil, já que as pessoas do time têm processos diferentes em seus próprios departamentos.

1	2	3	4	5
Discordo	Discordo	Indiferente	Concordo	Concordo

Totalmente	Parcialmente		Parcialmente	Totalmente

10. O time *Scrum* tem dificuldades na realização de uma boa reunião de retrospectiva (*Sprint Retrospective*) porque as melhorias no processo são discutidas de forma fragmentada visando interesses de diferentes áreas, e não visam a melhoria do processo para o time como um todo.

1	**2**	**3**	**4**	**5**
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

11. Minha empresa tem dificuldades em adotar o papel de *Product Owner* porque este papel não existe formalmente em nenhum departamento.

1	**2**	**3**	**4**	**5**
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

12. Minha empresa não consegue fazer com que o *Product Owner* esteja envolvido com o time de desenvolvimento nos momentos em que a interação faz-se necessária.

1	**2**	**3**	**4**	**5**
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

13. Minha empresa tem dificuldade em adotar o papel de *Scrum Master* porque este papel não existe formalmente em nenhum departamento de TI.

1	**2**	**3**	**4**	**5**
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

14. O *Scrum Master* da minha empresa tem dificuldades no cumprimento do processo *Scrum* por conta da falta de apoio de outros departamentos, que nem sempre respeitam os limites e regras do processo *Scrum*.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

15. Minha empresa tem dificuldades em manter o time de desenvolvimento trabalhando junto na iteração, pois membros do time precisam também trabalhar em tarefas de seus departamentos de origem.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

16. O time de desenvolvimento entende que as tarefas do time multidisciplinar não podem ser feitas em paralelo, mas sim devem ser feitas em cascata, iniciando pelos requisitos, para que depois sejam feitas as tarefas de construção e só então as tarefas de teste.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

17. Minha empresa tem dificuldades em deixar claros os papeis e responsabilidades das pessoas no cenário onde seu trabalho é dividido entre atuar no time *Scrum* e em seu próprio departamento.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

18. As pessoas do time *Scrum* tendem a inspecionar seu processo considerando mais o ponto de vista do seu departamento original do que o ponto de vista do time como um todo.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

19. As pessoas do time *Scrum* tendem a inspecionar o processo visando uma organização das atividades onde as tarefas de análise devem preceder as tarefas de construção, que por sua vez devem preceder as tarefas de teste.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

20. Minha empresa não aplica melhorias ao processo pois valoriza o método *Scrum* menos do que seus próprios processos departamentais individuais.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

21. Minha empresa tem dificuldade em manter um *timebox* no Sprint, pois membros do time constantemente são interrompidos para resolver problemas de seus departamentos de origem.

1	2	3	4	5
Discordo Totalmente	Discordo Parcialmente	Indiferente	Concordo Parcialmente	Concordo Totalmente

APÊNDICE E – Respostas do Questionário Fechado para a Empresa A

Respondente	Notas por questões																				
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21
Respondente 1	2	4	2	4	1	1	1	3	1	1	2	2	4	3	1	3	3	2	4	2	1
Respondente 2	5	2	5	1	4	3	4	5	3	5	5	4	4	5	2	4	5	5	5	4	3
Respondente 3	4	4	5	1	5	1	4	2	1	5	4	5	3	5	4	1	3	5	3	2	3
Respondente 4	5	4	2	2	1	1	2	5	2	1	4	1	1	2	1	2	1	1	5	1	2
Respondente 5	5	3	2	2	1	1	1	2	1	1	1	1	1	1	1	4	2	1	4	2	2
Respondente 6	2	4	4	3	4	2	2	4	2	4	4	4	2	4	1	4	2	2	4	2	1
Respondente 7	3	4	4	4	4	2	1	1	3	2	1	3	2	3	1	4	3	3	4	4	2
Respondente 8	5	5	4	4	4	2	2	2	2	2	2	4	3	2	2	3	3	4	4	4	2
Respondente 9	4	5	1	4	1	1	5	5	1	1	3	1	1	3	1	5	3	3	5	4	3
Respondente 10	5	5	1	4	2	1	1	2	1	2	1	1	2	1	1	2	2	2	2	4	2
Respondente 11	4	2	1	4	1	1	4	4	1	2	5	5	1	4	2	3	2	4	5	5	4
Respondente 12	2	1	4	2	1	1	1	4	2	2	4	4	4	3	3	3	3	4	4	4	3
Respondente 13	1	5	2	4	1	1	2	2	2	4	1	1	1	1	1	1	1	2	5	1	2
Respondente 14	4	5	5	5	3	4	4	5	2	4	5	5	3	4	1	1	2	3	1	3	2
Respondente 15	4	4	2	4	2	4	4	5	2	4	5	5	3	5	2	4	2	5	5	5	2
Respondente 16	1	4	1	1	1	1	1	1	1	3	1	1	1	1	1	1	1	4	1	3	2
Respondente 17	3	4	4	4	3	4	3	5	4	5	4	5	4	5	3	5	3	5	5	5	5
Respondente 18	4	3	5	2	1	1	4	2	2	4	5	4	1	3	1	4	4	4	5	2	1
Média	3,5	3,8	3,0	3,1	2,2	1,8	2,6	3,3	1,8	2,9	3,2	3,1	2,3	3,1	1,6	3,0	2,5	3,3	3,9	3,2	2,3

APÊNDICE F – Respostas do Questionário Fechado para a Empresa B

Respondente	Notas por questões																				
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21
Respondente 1	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5	5
Respondente 2	4	2	5	1	1	1	2	2	1	4	5	5	5	4	5	4	5	5	4	5	5
Respondente 3	4	3	4	4	1	1	1	5	4	3	4	5	5	5	1	5	5	3	4	5	5
Respondente 4	2	3	1	1	1	1	1	5	2	1	5	5	5	5	5	5	5	4	3	1	1
Respondente 5	4	3	4	2	3	1	5	5	2	4	5	5	5	5	4	2	3	5	3	5	5
Respondente 6	1	2	2	1	1	4	4	5	5	5	5	5	5	4	4	4	5	4	4	4	5
Respondente 7	5	2	4	1	1	3	4	5	3	5	5	5	5	5	5	5	5	5	5	5	5
Respondente 8	1	2	4	3	2	4	4	2	4	5	5	4	5	5	5	3	5	3	2	5	5
Respondente 9	1	1	5	1	5	5	5	5	5	5	5	5	5	5	5	1	5	5	5	5	5
Respondente 10	5	3	5	4	2	3	5	3	2	3	4	1	5	5	4	4	5	5	3	5	4
Média	3,2	2,6	3,9	2,3	2,2	2,8	3,6	4,2	3,3	4,0	4,8	4,5	5,0	4,8	4,3	3,8	4,8	4,4	3,8	4,5	4,5

APÊNDICE G – Respostas do Questionário Fechado para a Empresa C

Respondente	Notas por questões																				
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Q18	Q19	Q20	Q21
Respondente 1	2	4	4	4	2	1	2	4	1	2	1	1	1	2	1	1	1	1	2	1	1
Respondente 2	1	4	1	3	1	1	2	2	1	1	2	1	1	2	1	1	1	1	3	1	1
Respondente 3	1	4	2	4	2	1	2	4	1	2	1	1	1	2	1	1	1	1	2	1	1
Respondente 4	2	3	3	3	1	2	3	2	2	1	2	2	2	1	2	1	1	1	3	1	1
Respondente 5	1	4	2	3	1	1	2	4	2	2	1	1	1	2	1	1	1	1	2	1	1
Média	1,4	3,8	2,4	3,4	1,4	1,2	2,2	3,2	1,4	1,6	1,4	1,2	1,2	1,8	1,2	1,0	1,0	1,0	2,4	1,0	1,0