DOI:10.5748/9788599693131-14CONTECSI/PS-4677

DIFICULTIES IN ADOPTION AND USAGE OF SCRUM METHOD IN NON-PROJECTIZED BRAZILIAN COMPANIES USING PLAN-DRIVEN PROCESSES: MULTIPLE CASE STUDIES

Daniel Medeiros de Assis (IPT - Instituto de Pesquisas Tecnológicas do Estado de São Paulo, São Paulo, Brasil) – daniel@arneam.com
Claudio L. C. Larieira (Fundação Getúlio Vargas, São Paulo, Brasil) – larieira@hotmail.com

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, and others. 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 specifically, hindering its adoption. This study presents the major difficulties in Scrum adoption in organizations that make use of plan-driven processes. To identify those difficulties, interviews were conducted and questionnaires were applied in three brazilian companies that make use of IT to support their business, with similar processes. After the result analysis, the major points of difficulty in Scrum adoption were consolidated in a way that allow companies to identify adjustment points in its processes to improve the adoption of Scrum.

Keywords: Agile Methods, Scrum in Organizations, Non-projectized Organizational Structures, Plan-Driven Process, Scrum Tailoring.

DIFICULDADES NA ADOÇÃO E USO DO METODO SCRUM EM EMPRESAS BRASILEIRAS NÃO PROJETIZADAS UTILIZANDO PROCESSOS PLAN-DRIVEN: ESTUDOS DE CASO MÚLTIPLOS

Empresas têm demonstrado interesse crescente sobre adoção de métodos ágeis em seus departamentos internos de desenvolvimento de software, visando obter benefícios relacionados ao aumento de produtividade, melhor relacionamento entre áreas de TI e de negócio, dentre outros. Contudo, já possuem processos consolidados, que muitas vezes entram em conflito com práticas sugeridas pelos métodos ágeis e pelo Scrum, dificultando sua adoção. Este estudo apresenta as maiores dificuldades na adoção de Scrum em organizações que fazem uso de processos plan-driven. Para identificar estas dificuldades, entrevistas foram conduzidas e questionários foram aplicados em três empresas brasileiras que utilizam departamentos de TI para suporte ao negócio, com processos similares. Após análise de resultados, os maiores pontos de dificuldade na adoção do Scrum foram consolidados de forma a permitir que empresas identifiquem pontos de ajustes em seus processos para melhorar a experiência de adoção do Scrum.

Palavras-chave: Métodos Ágeis, Scrum em Empresas, Estruturas Organizacionais Não Projetizadas, Processos Plan-Driven, Tailoring de Scrum.

1. Introdução

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 críticas, tais como baixa velocidade nas entregas e dificuldade em gerenciar mudanças em requisitos (Senepathi & Srinivasan, 2014). 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 e ramos variados. Constatou-se que 45% dos respondentes trabalham em empresas onde a maioria dos times usam métodos ágeis. 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). 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.

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.

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, Conboy & Yang, 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) (Senepathi, 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. Empresas que utilizam modelos *plan-driven* podem encontrar dificuldades na adoção de *Scrum,* pois métodos ágeis apresentam características opostas aos modelos *plan-driven,* como a possibilidade de mudanças constantes e rápidas, baseado em *feedbacks* (Fowler & Highsmith, 2001).

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 relacionados aos métodos ágeis (Cohn & Ford, 2003) (Hirsch, 2005) (Nerur et al., 2005) (Tolfo et al, 2011) (Waardenburg & Vliet, 2013) (Gren et al., 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.

Este artigo tem o interesse de apresentar como empresas brasileiras que desenvolvem software com abordagem *plan-driven* têm percebido dificuldades ao adotar o método *Scrum* e como tem atuado na customização de seus processos.

2. Fundamentos

2.1.Plan-Driven

O termo *plan-driven* tem sido empregado na literatura para descrever modelos de desenvolvimento de *software* mais rigorosos, em que um plano precede o desenvolvimento efetivo, geralmente envolvendo um processo estruturado e grande quantidade de artefatos e documentação. A Tabela 1 apresenta características de *plan-driven* por autores.

Tabela 1
Características de *plan-driven* por autores

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)
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 et al., (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)

Nota. Elaborado pelo autor

Plan-driven não é um modelo de ciclo de vida de *software*, mas sim uma categorização de modelos. 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 (Schwaber & Sutherland, 2013).

2.2. Método Scrum

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 Nonaka (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.

2.2.1. 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 (Schwaber & Sutherland, 2013).

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. Precisa ser respeitado pela organização em suas decisões(Schwaber & Sutherland, 2013).

O Time de Desenvolvimento é o grupo de pessoas que constrói e entrega o incremento de *software* definido no *Product Backlog* ao 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 (Schwaber & Sutherland, 2013).

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 ao lidar com os interesses e necessidades do *Product Owner*, do Time de Desenvolvimento, e da organização (Schwaber & Sutherland, 2013).

2.2.2. 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).

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*.

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.

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.

2.2.3. 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).

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 ou 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 *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.

2.3.Tailoring para adoção do Scrum em organizações plan-driven

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 et al., 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.

3. Metodologia

Este artigo procura entender como 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. Trata-se, portanto, de um estudo de caso, pois examina eventos contemporâneos (ao contrário da pesquisa histórica) sem manipular os eventos comportamentais (ao contrário do experimento) (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.

A seleção dos casos foi feita por meio da entrevista a um grupo de pessoas (atuantes na adoção de métodos ágeis em empresas), de forma a identificar se atuavam em empresa(s) adequadas(s) para os critérios estabelecidos nesta pesquisa. Os critérios de seleção adotados foram escolhidos de acordo com os principais aspectos em discussão neste trabalho: i) empresas utilizadoras de métodos *plan-driven; ii*) empresas interessadas na utilização de *Scrum* e que estejam encontrando dificuldades. Três empresas (A, B e C) foram selecionadas. Todas as empresas também compartilham um mesmo tipo de estrutura organizacional não projetizada, mas este aspecto não é aprofundado por não ser um tópico de interesse do presente artigo.

Na etapa de coleta de dados, foram adotadas as fontes de documentação e entrevistas. Fez-se uso de um roteiro de perguntas semi-estruturado (para verificar como a estrutura e o processo incorporavam o uso do *Scrum*) e de um questionário de perguntas estruturado com questões fechadas (sobre dificuldades na adoção do *Scrum*).

O tratamento de dados da documentação foi realizado considerando a técnica de análise de conteúdo de Bardin (2004), com as etapas de pré-análise, exploração dos materiais, tratamento dos dados e interpretação. Já para o questionário de perguntas fechadas, foi aplicado um método diferente: i) O questionário foi apresentado e respondido por diversos profissionais da empresa, de forma a 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; iii) foi aplicado o critério de maior média para seleção dos cinco temas mais significativos; iv) Os temas selecionados foram analisados detalhadamente. A análise propriamente dita foi realizada individualmente para cada estudo de caso e, posteriormente, de forma comparativa entre os casos.

4. Resultados e Análise

4.1. Empresa A

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.

Na pesquisa realizada na Empresa A, dezoito entrevistados responderam às questões fechadas sobre dificuldades na adoção de S*crum*.

4.2. Empresa B

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.

Na pesquisa realizada na Empresa B, dez entrevistados responderam às questões fechadas sobre dificuldades de *Scrum* na Empresa B.

4.3. Empresa C

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 departamento 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).

Na pesquisa realizada na Empresa C, cinco entrevistados responderam às questões fechadas sobre dificuldades de *Scrum* na Empresa C.

4.4. Análise comparativa dos resultados

Foi realizada uma seleção de cinco questões de cada empresa, pelo critério da maior média. Os temas relativos a estas questões, relacionados ao processo *plan-driven*, foram agrupados para comparação, sendo apresentados na Tabela 2.

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 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.

Tabela 2
Temas de maior dificuldade de adoção de *Scrum* nas empresas (*plan-driven*).

Tema	**Empresa**	**Empresa**	**Empresa**

	A	B	C
Pilar Inspeção em processo *plan-driven*	*X*		*X*
Backlog/requisitos em processo *plan-driven*	*X*		X
Reunião de revisão em processo *plan-driven*	*X*		*X*
Papel de Scrum Master em processo *plan-driven*		*X*	
Reunião de planejamento em processo *plan-driven*			*X*

Nota. Elaborado pelo autor

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.

Apresentação de dados gerais consolidados

A Tabela 3 apresenta os dados consolidados do questionário fechado, considerando os resultados de todas as questões para todas as empresas, ordenados pela maior média geral, dentro da ótica de processo *plan-driven*.

Tabela 3
Respostas para análise do questionário fechado de todas as empresas (*plan-driven*).

Número	Tema Relacionado	Média Empresa A	Média Empresa B	Média Empresa C	Média Geral
8	reunião de revisão em processo *plan-driven*	3,3	4,2	3,2	3,6
19	pilar Inspeção em processo *plan-driven*	3,9	3,8	2,4	3,4
2	*backlog*/requisitos em processo *plan-driven*	3,8	2,6	3,8	3,4
14	papel de *Scrum Master* em processo *plan-driven*	3,1	4,8	1,8	3,2
12	papel de *Product Owner* em	3,1	4,5	1,2	3,0

	processo *plan-driven*				
4	reunião de planejamento em processo *plan-driven*	3,1	2,3	3,4	2,9
10	reunião de retrospectiva em processo *plan-driven*	2,9	4,0	1,6	2,8
16	papel do time de desenvolvimento em processo *plan-driven*	3,0	3,8	1,0	2,6
6	reunião diária em processo *plan-driven*	1,8	2,8	1,2	1,9

Nota. Elaborado pelo autor

4.5. Conclusões da análise

A análise comparativa entre as três empresas permitiu observar que:

- Embora todas as empresas possuam semelhanças em processo, 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. 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 a empregar 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* 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 comumente 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.

5. Conclusões

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 artigo propôs-se a estudar as dificuldades na adoção de *Scrum* em empresas com características similares em processo (*plan-driven,* ou orientado a um plano).

Os estudos desta pesquisa apontaram indícios de que a adoção do *Scrum* sofreu influência direta de características 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 3 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*: 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 aos profissionais envolvidos com o método *Scrum,* para que o método possa ser adotado da melhor forma possível (o *tailoring* do método *Scrum* deve ser aplicado de acordo com o nível de poder concedido pela empresa aos profissionais). Estes dois itens sugerem indícios de que é importante que empresas de processos *plan-driven* considerem 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*.

6. Recomendações

Empresas que possuam métodos *plan-driven* e queiram adotar *Scrum* (para usufruir dos benefícios dos métodos ágeis) devem saber que encontrarão desafios diversos. De forma a tornar a experiência de adoção mais assertiva, recomenda-se que tais empresas, antes de iniciar qualquer adoção (e também durante o processo) capacitem e reciclem o conhecimento dos profissionais envolvidos com o *Scrum*, de forma que verifique-se haver um real entendimento do método (e não um conjunto de valores e práticas assumidos como sendo *Scrum*, mas que na verdade não o são). Além disso, empresas devem revisar sua política de concessão de poder, considerando especificamente os papeis de *Scrum Master* e *Product Owner*. Estes papeis necessitam de algumas liberdades e poderes que a empresa precisa endossar para que o trabalho possa ser corretamente realizado, de acordo com os limites aceitáveis da organização.

Desta forma, recomenda-se que o tailoringde *Scrum* para empresas com processo *plan-driven* considere, especialmente, os dois aspectos principais aqui mencionados: a capacitação e a concessão de poderes.

Referências

Abrahamsson, P., Conboy, K. & Yang, X. (2009). 'Lot's done, more to do': the Current State of Agile System Development Research. *European Journal of Information Systems, 18*(4), 281-284.

Ahmed, F., Robinson, S. & Tako, A. A. (2014). Using the Structured Analysis and Design Technique (SADT) in Simulation Conceptual Modeling. *Proceedings... IEEE Press,* 1038-1049. Winter Simulation Conference, 2014, Savanna, Estados Unidos.

Bardin, L. (2004). Análise de Conteúdo. *Edições 70*, 229p.

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

Beck, K. (1999, october) Embracing Change with Extreme Programming. *IEEE Computer Society, 32*(10), 70-77.

Beck, K. & Boehm, B. (2003, june). Agile throught Discipline: A Debate. *IEEE Computer Society*, *36*(6), 44-46.

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

Boehm, B. W. (1988, may). A Spiral Model of Software Development and Enhancement. *Computer Magazine*, 21(5), 61-72.

Boehm, B. W. & Turner, R. (2003) Balancing Agility and Discipline: A Guide for the Perplexed. *Addison-Wesley/Pearson Education*, 304p.

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

Cohn, M. & Ford, D. (2003, june). Introducing an Agile Process to an Organization. *IEEE Computer Society, 36*(6)*,* 74-78.

Dingsoyr et al. (2012, june). A decade of Agile Methodologies: Towards explaining Agile Software Development. *The Journal of Systems and Software*, *85*(6), 1213-1221.

Dinsmore, P. & Cabanis-Brewin, J. (2006). The AMA Handbook of Project Management. *AMACOM*, 512p.

Eisenhardt, K. M. (1989, october). Building Theories from Case Study Research. *Academy of Management Review*, *14*(4), 532-550.

Feiler, P. H. & Humphrey, W. S. (1993). Software Process Development and Enactment:

Concepts and Definitions. *Proceedings... of the 2nd International Conference on the Software Process.* pp.28-40. SECOND INTERNATIONAL CONFERENCE ON THE SOFTWARE PROCESS, 1993, Berlin, Germany.

Fowler, M. & Highsmith, J. (2001, august) The Agile Manifesto. *Software Development Magazine*, *9*, p.28-35.

Glaser, B. & Strauss, A. (1967). The Discovery of Grounded Theory: strategies for qualitative research. *Aldine Transaction*, 271p. Chicago.

Gren, L., Torkar, R. & Feldt, R. (2014). Work Motivational Challenges Regarding the Interface Between Agile Teams and a Non-Agile Surrounding Organization: A case study. *Proceedings... IEEE Computer Society, AGILE'14,* pp.11-15. AGILE CONFERENCE, 2014, Florida, United States.

Haugen, N. (2006). An Empirical Study of Using Planning Poker for User Story Estimation. *Proceedings... IEEE Computer Society, AGILE'06,* pp.34-43. AGILE CONFERENCE, 2006, Minneapolis, United States.

Highsmith, J. (2002, october). What is Agile Software Development? *Crosstalk: The Journal of Defense Software Engineering*, *15*(10), 4-9.

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

Hoda, R. (2011) Self-Organizing Agile Teams: A Grounded Theory. 262p. Victoria University of Wellington, Wellington, Nova Zelândia.

Hoda, R., Noble, J. & Marshall, S. (2012, december). Developing a grounded theory to explain the practices of self-organizing Agile teams. *Empirical Software Engineering*, *17*(6), 609-639.

Humphrey, W. S., Snyder, T. R. & Willis, R. R. (1991, july). Software Process Improvement at Hughes Aircraft. *IEEE Software*, *8*(4), 11-23.

IEEE Computer Society. (2014). SWEBOK v3.0 – Guide to the Software Engineering Body of Knowledge. 335p. United States.

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

Jick, T. D. (1979, december). Mixing Qualitative and Quantitative Methods: Triangulation in Action. *Administrative Science Quaterly*, *24*(4), 602-611.

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

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

Larman, C. & Basili, V. R. (2003, june). Iterative and Incremental Development: A Brief History. *IEEE Computer Society*, *36*(6), 47-56.

Lee, G. & Xia, W. (2010, march). Toward Agile: An Integrated Analysis of Quantitative and Qualitative Field Data on Software Development Agility. *MIS Quaterly*, *34*(1), 87-114.

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

Nerur, S., Mahapatra, R. & Mangalaraj, G. (2005, may) Challenges of Migrating to Agile Methodologies. *Communications of the ACM, 48*(5), 73-78.

Nikitina, N. & Kajko-Mattson, M. (2014) Guiding the Adoption of Software Development Methods. *Proceedings... ACM Publications, ICCSP,* pp.109-118. ICSSP '14 INTERNATIONAL CONFERENCE ON SOFTWARE AND SYSTEM PROCESS, 2014, Nanjing, China

Paulk, M. (2001, november). Extreme Programming from a CMM Perspective. *IEEE Software, 18*(6), 19-26.

Petersen, K & Wohlin, C. (2010, december). The effect of moving from a plan-driven to an incremental software development approach with agile practices. *Empirical Software Engineering*, *15*(6), 654-693.

Project Management Institute (2013). A Guide to the Project Management Body of Knowledge (PMBOK Guide), 589p.

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

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

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

Schwaber, K. & Sutherland, J. (2013) The Scrum Guide**.** 16p. Available online at:<http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf>. Accessed on: 05 Feb 2017.

Senepathi, M. & Srinivasan, A. (2014). An Empirical Investigation of the Factors

Affecting Agile Usage. *Proceedings... ACM Publications, EASE,* article n.10. EASE '14 18th INTERNATIONAL CONFERENCE ON EVALUATION AND ASSESSMENT IN SOFTWARE ENGINEERING, 2014, London, United Kingdon.

Sutherland, J. (2001). Agile Can Scale: Inventing and Reinventing SCRUM in Five Companies. *Cutter IT Journal*, 14(12), 5-11.

Svensson, H. & Host, M. (2005). Introducing an Agile Process in a Software Maintenance and Evolution Organization. *Proceedings... IEEE Computer Society, CSMR*, pp.256-264. CSMR '05 9th EUROPEAN CONFERENCE ON SOFTWARE MAINTENANCE AND REENGINEERING, 2005, Manchester, UK.

Takeuchi, H. & Nonaka, I. (1986, january). The New New Product Development Game. *Harward Business Review*, pp.137-146.

Tolfo, C. et al. (2011, october). Agile Methods and Organization Culture: Reflections about Cultural Levels. *Journal of Software Maintenance and Evolution: Research and Practice*, *123*(6), 423-441.

VersionOne (2014). 9Th Annual State of Agile Survey**.** 17p. Available online at: <http://info.versionone.com/state-of-agile-development-survey-ninth.html>. Accessed on: 05 Feb 2017.

Waardenburg, G. & Vliet, H. (2013, december). When agile meets the enterprise. *Information and Software Technology*, *55*(12), 2154-2171.

West, D. (2011, july). Water-Scrum-Fall Is The Reality Of Agile For Most Organizations Today. *Forrester Research*, 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: 5 Feb 2017.

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