Pós-Graduação em Ciência da Computação

Mapeamento Sistemático sobre o uso do Auto-gerenciamento em Equipes de Desenvolvimento de Software

Por

Elisa Sattyam de Farias Cardozo

Dissertação de Mestrado

Universidade Federal de Pernambuco

posgraduacao@cin.ufpe.br

www.cin.ufpe.br/~posgraduacao

RECIFE, PE

2012

UNIVERSIDADE FEDERAL DE PERNAMBUCO

CENTRO DE INFORMÁTICA

PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

ELISA SATTYAM DE FARIAS CARDOZO

Mapeamento Sistemático sobre o uso do Auto-gerenciamento em Equipes de Desenvolvimento de Software

ESTE TRABALHO FOI APRESENTADO À PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO DO CENTRO DE INFORMÁTICA DA UNIVERSIDADE FEDERAL DE PERNAMBUCO COMO REQUISITO PARCIAL PARA OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIA DA COMPUTAÇÃO.

Orientador: **Prof. Fabio Queda Bueno da Silva, PhD**

Recife, PE

2012

Catalogação na fonte

Bibliotecária Jane Souto Maior, CRB4-571

Cardozo, Elisa Sattyam de Farias

Mapeamento sistemático sobre o uso do auto – gerenciamento em equipes de desenvolvimento de software / Elisa Sattyam de Farias. - Recife: O Autor, 2012.
xiii, 156 folhas: il., fig., tab.

Orientador: Fabio Queda Bueno da Silva.
Dissertação (mestrado) - Universidade Federal de Pernambuco. CIn, Ciência da Computação, 2012.
Inclui bibliografia.

1. Engenharia de software. 2. Gerenciamento de projetos. 3. Desenvolvimento de software. I. Silva, Fabio Queda Bueno da (orientador). II. Título.

005.1 CDD (23. ed.) MEI2012 – 067

Dissertação de Mestrado apresentada por **Elisa Sattyam de Farias Cardozo** à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título **"Mapeamento Sistemático sobre o Auto-gerenciamento de Equipes de Desenvolvimento de Software"** orientada pelo **Prof. Fabio Queda Bueno da Silva** e aprovada pela Banca Examinadora formada pelos professores:

_____

Prof. Sergio Castelo Branco Soares
Centro de Informática / UFPE

_____

Prof. Clauirton de Albuquerque Siebra
Departamento de Informática / UFPB

_____

Prof. Fabio Queda Bueno da Silva
Centro de Informática / UFPE

Visto e permitida a impressão.
Recife, 17 de abril de 2012

_____

Prof. Nelson Souto Rosa
Coordenador de Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

Dedico este trabalho aos meus pais Patrícia e Oswaldo e aos meus avós Carmelita e Oswaldo pela educação que me foi dada e por tudo o que proporcionaram para que eu chegasse até aqui.

AGRADECIMENTOS

Agradeço a Deus pela saúde concedida, por iluminar meu caminho, pelas graças alcançadas e por ter enviado pessoas tão especiais na minha vida como estas:

Aos meus pais Patrícia e Oswaldo e aos meus avós Carmelita e Oswaldo, que, juntos, representam minha base.

A TODA a minha família por todos os momentos especiais que passamos juntos, pelo acolhimento nos momentos em que precisei, pelos conselhos e palavras de conforto e apoio.

A Felipe, meu companheiro, amigo, namorado, pela compreensão, respeito, incentivo, críticas e apoio dados durante todo o período dos cursos de graduação e de mestrado.

A todos os meus amigos e amigas que sempre estiveram presentes me aconselhando e incentivando com carinho e dedicação.

Aos diretores, gerentes e colegas das empresas que já trabalhei, pela oportunidade prestada, pela confiança no meu trabalho e pelo grande aprendizado que pude adquirir.

Ao professor orientador Fabio Q. B. da Silva, pela confiança depositada em mim, pela oportunidade, incentivo, disponibilidade, atenção e ensinamentos prestados para a realização da pesquisa.

Aos membros do grupo de pesquisas HASE, pelas contribuições valiosas que direta ou indiretamente proporcionaram a este trabalho.

Finalmente, agradeço ao Projeto Samsung (CIn/UFPE) pela bolsa concedida durante o período do curso e pela atenção dos funcionários da organização.

A todos, muito obrigada!

“Não fiques em terreno plano.
Não subas muito alto.
O mais belo olhar sobre o mundo está a meia encosta."

Friedrich Nietzsche

Resumo Estruturado

Contexto – Muitos autores afirmam que o auto-gerenciamento de equipes está sendo cada vez mais adotado nas organizações. Uma das razões para esta popularidade é o fato de alguns autores considerarem que equipes auto-gerenciadas promovem um impacto positivo sobre alguns aspectos da efetividade, tais como o aumento da produtividade, a melhoria da qualidade do produto e também da qualidade de vida no trabalho. Porém, alguns pesquisadores afirmam que esses resultados são inconsistentes e podem variar de acordo com o tipo de trabalho. Na literatura de engenharia de software, pesquisadores indicam que há escassez sobre o tema.

Objetivo – Mapear o conhecimento existente na literatura sobre o autogerenciamento de equipes inseridas no contexto de desenvolvimento de software e analisar a força das evidências encontradas em apoio às conclusões.

Método – Um Mapeamento Sistemático da Literatura foi realizado para identificar, analisar e reunir o conhecimento encontrado na literatura atual sobre o auto-gerenciamento de equipes de software.

Resultados – A partir de buscas automáticas e manuais, foram retornados 2.646 estudos. Após passarem por uma seleção, 43 desses estudos foram incluídos nesta pesquisa para serem analisados em profundidade. Esses estudos permitiram a identificação de 25 características definidoras de uma equipe de software auto-gerenciada, 28 resultados promovidos por estas equipes e 93 fatores que podem afetar seu trabalho, dentre fatores técnicos, humanos e organizacionais. Os estudos incluídos também passaram por uma avaliação da qualidade que indicou que 91% deles possuem fortes evidências para apoiar as conclusões da pesquisa.

Conclusões – Apesar da quantidade significativa de estudos incluídos neste mapeamento, foram identificadas algumas lacunas no que diz respeito à profundidade da abordagem dada ao tema e à origem das evidências apontadas nos estudos. Portanto, ainda há escassez de estudos que abordem o tema com profundidade. Porém, é possível concluir que equipes de software auto-gerenciadas possuem diversas particularidades que, se não forem bem administradas, comprometem o auto-gerenciamento e podem prejudicar os resultados de desempenho.

Palavras-Chave: Auto-gerenciamento, Equipes, Desenvolvimento de Software, Mapeamento Sistemático da Literatura.

Abstract

Background - Many authors state that self-managing teams are being increasingly adopted in organizations. One reason for this popularity is the fact that some authors consider that self-managing teams promote a positive impact in some aspects of effectiveness, such as increased productivity, improved product quality, and also the quality of work life. However, some researchers argue that these results are inconsistent and can vary accordingly the type of work. In software engineering literature, some researchers indicate that there is a lack of data and studies on the subject.

Aim – To understand the existing knowledge in the literature on self-management teams within the context of software development and to analyze the strength of the evidence found to support the conclusions.

Method - A Systematic Mapping Study was conducted to identify, analyze, and systematically collect the knowledge found in the current literature on self-managing software teams.

Results - From automatic and manual searches, 2,646 studies were returned. After passing through a selection process, 43 studies were included in this study to be analyzed in depth. These studies allowed the identification of 25 defining characteristics of a self-managing software team, 28 results promoted by these teams, and 93 factors that may affect their work, among technical, human and organizational factors. The selected studies were also passed through a quality assessment, which indicated that 91% of them have strong evidence to support the research findings.

Conclusions - Despite the significant amount of studies included in this mapping, we identified some gaps with respect to the depth of approach used in the studies and the origin of the evidence presented. Therefore, there are few studies that address the topic in depth. However, it is possible to conclude that self-managing software teams have several particularities, which if not well managed may undermine self-management and affect performance results.

Key-Words: Self-management, Teams, Software Development, Systematic Mapping Study.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS/ACRÔNIMOS

SMWTs – Self-Managed Work Teams

EBM - Evidence-Based Medicine

EBSE - Evidence-based Software Engineering

SLR - Sistematic Literature Review

MS - Mapping Study ou Mapeamento Sistemático

QVT – Qualidade de Vida no Trabalho

ESEM - International Symposium on Empirical Software Engineering and Metrics

HSSE - Human and Social Factors of Software Engineering

EASE - Evaluation and Assessment of Software Engineering

SPE - Software Practice and Experience

TSE - IEEE Transactions on Software Engineering

ASD - Adaptive Software Development

XP - Extreme Programming

DSDM - Dynamic Systems Development Method

Sumário

1. Introdução

Este capítulo apresenta uma generalizada introdução sobre o auto-gerenciamento de equipes de software, com informações sobre as razões e natureza da pesquisa, sua importância, seus objetivos e a proposta para a elaboração.

Dessa forma, o capítulo está organizado como segue:

1.1 Motivação – Apresenta uma introdução ao tema, permeando por sua importância, por como o problema pesquisado foi definido, as questões utilizadas para guiar a pesquisa, os objetivos da pesquisa e o contexto no qual ela se insere.
1.2 Contribuição/Resultados Esperados – Apresenta algumas contribuições que a pesquisa espera trazer para a prática e para a ciência.
1.3 Estrutura do trabalho – Apresenta como esta dissertação está organizada.

1.1. Motivação

O auto-gerenciamento surgiu em contraponto ao gerenciamento tradicional e hierárquico, que possui uma estrutura de decisão centralizada, onde apenas uma pessoa toma todas as decisões. Enquanto a abordagem tradicional sugere que a liderança é uma função especializada que não pode ser compartilhada sem comprometer a eficácia da equipe e foca em profissionais com alta autonomia individual, mas com baixa autonomia concedida à equipe, membros de equipes de trabalho auto-gerenciadas possuem responsabilidade compartilhada, gerenciando e monitorando seus próprios processos e execução de tarefas (Moe et al., 2009). O auto-gerenciamento trata o conhecimento de forma descentralizada, fazendo com que todos os membros possuam o conhecimento de todo o processo de trabalho (DEVARO, 2007).

Sendo assim, o auto-gerenciamento propõe que a equipe de trabalho seja composta por indivíduos interdependentes que podem se auto-regular e ter autonomia para tomar decisões sem supervisão direta (GOODMAN e HUGHSON, 1988; MANZ e SIMS, 1993). Por vezes, equipes auto-gerenciadas são organizadas por função. Mas, geralmente, essas equipes são multifuncionais, possuindo uma variedade de habilidades para o trabalho (WALL et al., 1986; THOMS, 1996;).

O conceito de equipes de trabalho auto-gerenciadas vem sendo cada vez mais aplicado nas organizações (DRUSKAT e WHEELER, 2001; DRUSKAT e WHEELER, 2003 *apud* MUTHUSAMY et al., 2005). A popularidade dessas equipes se deve à tendência em se acreditar que podem melhorar drasticamente o desempenho. Essa crença é apoiada por algumas pesquisas, que sugerem que equipes de trabalho auto-gerenciadas devem ter um impacto positivo em alguns aspectos do desempenho (Harris, 1992 *apud* TATA e PRASAD, 2004; COHEN, 1993; Hammer e Stanton, 1995;), com a melhora na qualidade do produto, produtividade (ATTARAN e NGUYEN, 2000 *apud* TATA e PRASAD, 2004) e qualidade de vida no trabalho (QVT) (COHEN e LEDFORD, 1994), uma vez que promove funcionários mais satisfeitos, e diminui a rotatividade e absenteísmo (COHEN e BAILEY, 1997).

Com isso, acredita-se que equipes auto-gerenciadas podem contribuir para aumentar a eficácia organizacional, podendo ter efeito sobre todas as dimensões da eficácia como desempenho, atitudes e comportamentos (PASMORE et al.., 1982; COHEN, 1993;). O aumento da eficácia e a melhoria na qualidade dos serviços e produtos conseguidos através do auto-gerenciamento contribuem para impulsionar as organizações a alcançar um maior destaque no mercado, que está cada vez mais concorrido e disputado.

Na engenharia de software, apesar de não ser a única forma de trabalho para aplicação de equipes auto-gerenciadas, as metodologias ágeis são consideradas favorecedoras do auto-gerenciamento das equipes e, portanto, vêm ganhando muita popularidade nas indústrias de software (BEGEL e NAGAPPAN, 2007). O auto-gerenciamento é um dos 12 princípios seguidos por metodologias ágeis (Agile Manifesto, 2001) e alguns pesquisadores apontam que equipes auto-gerenciadas constituem o núcleo do desenvolvimento de software ágil (COCKBURN e HIGHSMITH, 2001; HIGHSMITH e FOWLER, 2001; SCHWABER, 2009).

O Manifesto Ágil[1] (2001) afirma que “as melhores arquiteturas, requisitos e projetos emergem de equipes auto-gerenciadas”. Além disso, pesquisadores afirmam que o auto-gerenciamento de equipes é um fator crítico de sucesso em projetos que utilizam metodologias ágeis (HODA et al., 2010).

[1] Manifesto Ágil é uma declaração de princípios que fundamentam o desenvolvimento ágil de software.

Para uma melhor organização, essa seção será dividida da seguinte maneira: Definição do Problema (1.1.1), Questões de Pesquisa (1.1.2), Objetivo Geral (1.1.3) e Objetivos Específicos (1.1.4).

1.1.1. Definição do Problema

Apesar do cenário positivo apresentado anteriormente, muitos pesquisadores indicam que os resultados são inconsistentes (COHEN e LEDFORD, 1994; TATA e PRASAD, 2004; MILLIKIN et al., 2010;). Em contraste com os resultados de maior eficácia e desempenho, também foi descoberto que a auto-gestão de equipes não aumentou todos os tipos de eficácia para todos os tipos de trabalho, podendo variar consideravelmente (BEEKUN, 1989; COHEN & LEDFORD, 1994; GUZZO & DICKSON, 1996 apud MILLIKIN et al., 2010; BERGMANN & DE MEUSE, 1996;. MOHRMAN et al., 1995 *apud* TATA & PRASAD, 2004; TATA & PRASAD, 2004). Da mesma forma, diferentemente do relatado anteriormente, foi descoberto que os membros de uma equipe auto-gerenciada têm maiores taxas de absenteísmo e rotatividade do que nas abordagens tradicionais (CORDERY et al., 1991 *apud* COHEN e LEDFORD, 1994).

Deste modo, podemos inferir que atribuir o rótulo de auto-gerenciamento a uma equipe não indica que ela terá, automaticamente, um melhor desempenho. Os resultados inconsistentes apresentados denotam a necessidade de um melhor entendimento sobre o auto-gerenciamento e os fatores que influem em seu sucesso. Segundo Cohen e Ledford (1994), esses resultados podem variar de acordo com o tipo de trabalho. No contexto de engenharia de software, pesquisadores afirmam que pouco se sabe sobre a natureza do auto-gerenciamento de equipes ágeis e seus desafios na prática, e que a literatura sobre o tema é escassa (HODA et al., 2010).

Portanto, ainda existem muitas lacunas sobre o tema pesquisado e há a necessidade de se produzir um melhor entendimento e interpretá-lo em uma única estrutura do conhecimento para que possa contribuir com praticantes e pesquisadores interessados.

Assim sendo, esta pesquisa irá realizar um estudo de mapeamento sistemático na direção da melhor compreensão do auto-gerenciamento em equipes de software e também do entendimento sobre os fatores que podem, de alguma forma, afetar o trabalho dessas equipes, tais como: fatores técnicos, humanos e organizacionais.

1.1.2. Questões de Pesquisa

Esta pesquisa visa identificar e analisar, a partir de um estudo de mapeamento sistemático, a literatura existente sobre auto-gerenciamento em equipes de software de forma a permitir encontrar respostas às seguintes **questões centrais** da pesquisa:

> Q1: O que se sabe atualmente sobre o auto-gerenciamento de equipes no desenvolvimento de software?
> Q2: Qual é a força da evidência em apoio a essas conclusões?

Para responder à questão principal Q1, foram elaboradas as três seguintes **questões específicas**:

> Q1.1: Como auto-gerenciamento é definido nos estudos da área de Engenharia de Software?
> Q1.2: Quais são os efeitos do auto-gerenciamento nas equipes de desenvolvimento de software?
> Q1.3: Quais são os fatores técnicos, humanos e organizacionais que podem afetar o trabalho de equipes de software auto-gerenciadas?

Mais detalhes sobre como essas questões foram formuladas estão descritos no APÊNDICE A.

1.1.3. Objetivo Geral

O objetivo desta pesquisa é investigar o conhecimento existente sobre o auto-gerenciamento de equipes de desenvolvimento de software e entender qual a força das evidências encontradas. Esse objetivo é apoiado pelo entendimento da definição do auto-gerenciamento de equipes, dos efeitos que o auto-gerenciamento pode trazer e dos fatores que podem influenciar a equipe de trabalho, sob o ponto de vista de pesquisadores no contexto de engenharia de software/desenvolvimento de software.

Essas evidências poderão contribuir com:

- A melhor implementação de equipes de software auto-gerenciadas nas organizações;
- O incentivo (ou não) ao uso do auto-gerenciamento em equipes de software;
- A adaptação das organizações aos fatores que podem afetar o trabalho de equipes de software auto-gerenciadas, no intuito de alcançar os benefícios do auto-gerenciamento;

- A identificação de possíveis lacunas e inconsistências do atual conhecimento;

1.1.4. Objetivos Específicos

Os objetivos específicos da pesquisa equivalem às metas que tiveram que ser alcançadas para que o objetivo final desta pesquisa fosse atingido. As metas definidas para esta pesquisa estão listadas como segue:

- Buscar e selecionar sistematicamente estudos sobre o auto-gerenciamento em equipes de software;
- Avaliar a qualidade desses estudos já selecionados;
- Extrair as informações relevantes desses estudos, baseando-se nas três questões específicas desta pesquisa;
- Categorizar e interpretar os dados de maneira que apóie as três questões específicas da pesquisa;
- Analisar as contribuições e limitações da pesquisa e as sugestões para trabalhos futuros;
- Combinar todos os resultados obtidos para apresentação do mapeamento.

1.1.5. Contexto

A engenharia de software é uma disciplina de natureza sócio-técnica baseada na interação entre fatores pessoais, sociais e técnicos (SAWYER, 2004). Equipes de engenharia de software podem ser consideradas compostas por trabalhadores do conhecimento, uma vez que os indivíduos que a compõem trabalham com a informação, convertendo-a em conhecimento para uso produtivo em seu (DRUCKER, 1959). De acordo com Drucker (1993), trabalhadores do conhecimento são indivíduos que detêm conhecimento analítico e teórico e a capacidade de conduzir esse conhecimento em diferentes contextos. Deste modo, Muthusamy et al. (2005) afirmam que “os trabalhadores do conhecimento não se limitam a determinados grupos ocupacionais ou níveis dentro de uma hierarquia organizacional” e que esses trabalhadores, geralmente, possuem mais conhecimento sobre sua área de trabalho do que seus gestores. Entretanto, para que seja possível a exploração do conhecimento tácito desses trabalhadores, são necessários interativos processos sociais que facilitam a comunicação e, portanto, o compartilhamento das informações (NONAKA, 1994).

Por envolver forte interação humana, os processos de desenvolvimento de software dependem significativamente do desempenho da equipe (MOE et al., 2008). De acordo com Pressman (2006), o desenvolvimento de software é conhecido por ser um processo complexo, existindo muitas variáveis que contribuem no sucesso ou fracasso dos projetos de software. Alguns pesquisadores têm sugerido que para melhorar o desempenho de projetos de software é necessário enfatizar questões pessoais mais do que conceitos metodológicos ou técnicos (DEMARCO & LISTER, 1999; SOMMERVILLE, 2007;). Segundo De Marco e Lister (1987), dentre as variáveis que podem afetar o desempenho das equipes, são destacados aspectos humanos, como a personalidade dos membros da equipe, e aspectos organizacionais, como o ambiente de trabalho, a composição e a organização das equipes.

1.2. Contribuição/Resultados Esperados

As principais contribuições deste estudo são:

- Para a prática do auto-gerenciamento em equipes de software, fazendo com que os colaboradores de uma companhia que adota o auto-gerenciamento compreendam melhor o que realmente significa uma equipe auto-gerenciada, os fatores que podem afetar o trabalho dessas equipes e os resultados, positivos e negativos, que a adoção do auto-gerenciamento em equipes de software pode trazer.
- Para a pesquisa científica na área de auto-gerenciamento em equipes de software, com a identificação de possíveis lacunas e inconsistências do atual conhecimento e a definição de novas estratégias de pesquisa sobre o auto-gerenciamento em equipes de software.

1.3. Estrutura do Trabalho

Esta dissertação está organizada como segue:

- No Capítulo 2 será apresentado todo o embasamento teórico e trabalhos relacionados que foram importantes para o melhor entendimento da pesquisa.
- O Capítulo 3 detalha como foi planejado e realizado o processo de condução da pesquisa, através do Quadro Metodológico e da definição da metodologia utilizada para a pesquisa.

- Capítulo 4 descreve todos os resultados encontrados com a execução, análise, interpretação da pesquisa e discussão.
- Capítulo 5 apresenta as considerações finais da presente pesquisa, descrevendo as limitações encontradas no decorrer da pesquisa e recomendações para trabalhos futuros.

2. Revisão da Literatura

Neste capítulo, será apresentada uma revisão da literatura, utilizada como fundamentação teórica para o melhor entendimento e condução desta pesquisa. O objetivo deste capítulo é a determinação do estado da arte, apresentando, por meio da literatura já publicada, o que se sabe sobre o tema pesquisado, as lacunas existentes e identificar os principais obstáculos teóricos ou metodológicos (LUNA, 1997).

O Capitulo está organizado nas seguintes seções: Referencial Teórico (2.1), Trabalhos Relacionados (2.2), Síntese (2.3).

2.1. Referencial Teórico

Esta seção irá descrever as teorias utilizadas para a condução da presente pesquisa. Todas as teorias aqui apresentadas foram obtidas através de um processo de levantamento de informações e análise dos estudos publicados, elaborado anteriormente à condução da pesquisa e aperfeiçoado no seu decorrer. Esse processo consistiu em realizar uma pesquisa em artigos, jornais, revistas científicas, dissertações e teses para a obtenção de uma gama de estudos suficiente para apoiar a pesquisa e relevante, utilizando-se os principais autores do tema pesquisado, para garantir a significância e confiabilidade da pesquisa.

Aqui será explorado o tema Auto-gerenciamento (na Seção 2.1.1), considerando sua base interdisciplinar, sendo apresentados conceitos pertinentes a diversas áreas do conhecimento e não apenas à Engenharia de Software. Isso também se deve ao fato de ainda não haver uma maturidade sobre o tema pesquisado no contexto da Engenharia de Software, como já mencionado no Capítulo 1.

Além do auto-gerenciamento, também será explorada a Engenharia de Software Baseada em Evidências (Seção 2.1.2), no intuito de apoiar a condução do método escolhido para a realização desta pesquisa, o Estudo de Mapeamento Sistemático.

2.1.1. Auto-gerenciamento

O auto-gerenciamento não é uma área de investigação recente. A história do auto-gerenciamento na literatura teve início em 1950, desde que Eric Trist e Ken Bamforth examinaram mineiros de carvão auto-regulados (TRIST, 1981; TATA e PRASAD, 2004). Desde então, o auto-gerenciamento (do inglês, *Self-management Work Teams* ou SMWT) vem recebendo diversos conceitos e termos diferentes, como auto-regulação (*self-regulation*), auto-organização (*self-organization*), auto-projetado (*self-designed*), grupos de trabalho autônomos (*autonomous workgroups*), grupos de trabalho semi-autônomos (*semi-autonomous work groups*), delegação de poderes (*empowerment*), auto-determinação (*self-determination*) entre outros (WALL et al., 1986; COHEN, 1993; COHEN e LEDFORD, 1994; GUZZO e DICKSON, 1996; COHEN et al., 1997).

O conceito do auto-gerenciamento vem sendo bastante implementado nas organizações (TATA e PRASAD, 2004), no intuito de melhorar a qualidade, produtividade, qualidade de vida no trabalho (QVT) e o desempenho do negócio (COHEN, 1993; COHEN e LEDFORD, 1994; TATA e PRASAD, 2004). Além disso, outra razão para essa popularidade do uso do auto-gerenciamento é o favorecimento de funcionários mais satisfeitos, menor rotatividade e absenteísmo (COHEN e BAILEY, 1997).

Equipes auto-gerenciadas (ou, do inglês, Self-Managed Work Teams - SMWTs) são grupos de indivíduos interdependentes, ricos em autonomia e identidade que têm poder de decisão sobre seu ritmo de trabalho e a distribuição de suas tarefas, podendo definir seus próprios objetivos e, muitas vezes, com autoridade para tomar decisões relativas a aspectos organizacionais ou de pessoal, como o recrutamento e treinamento de novos membros, além de outros assuntos do grupo. (Gulowsen, 1972 *apud* WALL et al., 1986; COHEN, 1993; CUMMING e GRIGGS, 1977, GOODMAN et al., 1988 *apud* COHEN et al., 1994; ; COHEN et al., 1997; HACKMAN et al., 1986). Ou seja, SMWTs assumem a responsabilidade das tarefas e todo o poder de decisão que tradicionalmente é reservado para a gestão (Banker et al., 1996; Guzzo e Dickson, 1996; Moorhead et al. , 1998 *apud* MILLIKIN et al., 2010).

Em geral, membros de uma equipe auto-gerenciada possuem variedade de habilidades para as tarefas e a equipe recebe *feedback* sobre o desempenho alcançado (WALL et al., 1994), como proposto por Wall ET AL. (1986), que afirmam que, para permitir esse auto-gerenciamento, a organização do trabalho geralmente tem três características:

1. Funcionários com tarefas inter-relacionadas que, coletivamente, são responsáveis por desenvolver um produto ou serviço;
2. Indivíduos multifuncionais para que possam executar a totalidade ou uma grande quantidade de tarefas;
3. Comentários e avaliações sobre o desempenho da equipe.

Com base em Goodman et al. (1988), Cohen (1993) reforça a importância da primeira característica apresentada por Wall et al. (1986). Porém, não menciona as outras duas e acrescenta outras:

1. Interação face a face;
2. Discrição dos funcionários sobre suas decisões, como atribuições de tarefas, métodos para execução e programação.

Hackman (1986) afirma que existem cinco sinais comportamentais que são previstos para equipes auto-gerenciadas de todos os níveis de maturidade:

- Os membros da equipe assumem a responsabilidade pessoal para os resultados de seu trabalho e demonstram que se sentem pessoalmente responsáveis pelos resultados obtidos;
- Os membros monitoram seus próprios desempenhos de maneira continua e sempre buscam dados e *feedback* para saber como eles estão realizando suas tarefas;
- Os membros gerem seu próprio desempenho, tomando medidas corretivas por sua própria iniciativa a fim de melhorar seu desempenho;
- Quando não possuem o que precisam para alcançar um bom desempenho, buscam ativamente da organização orientação, ajuda ou recursos que necessitam para que tenham um excelente desempenho – e fazem isso de maneira construtiva;
- Os membros tomam iniciativas para ajudar os colegas de outras áreas a melhorar seu desempenho, reforçando as políticas e desempenho da organização. Antes disso, os indivíduos se certificam de que suas próprias responsabilidades já foram cumpridas.

No entanto, a delegação de autoridade para a equipe, trazidas por essas definições, não é absoluta e o fato que uma equipe ser auto-gerenciada, não implica na ausência de uma supervisão (MANZ e SIMS, 1986 *apud* COHEN et al., 1997). SMWTs podem ou não ter supervisores diretos (COHEN, 1993). Em tendo um supervisor, seu papel é facilitar o desenvolvimento do auto-gerenciamento pelos funcionários para que eles possam administrar com sucesso suas atividades de trabalho com pouco controle organizacional (COHEN et al., 1997).

Apesar de concordarem com esta posição, Wall et al. (1986) afirmam que um supervisor é, muitas vezes, desnecessário e, curiosamente, foi observado por Cohen (1993) que, embora não houvesse uma diferença muito significativa, equipes sem supervisores são mais eficazes do que se trabalhassem com supervisores. Cohen (1993) conclui que talvez isso aconteça com equipes auto-gerenciadas mais maduras, com seu desempenho refletindo maior experiência e maturidade.

Independentemente de existir ou não um supervisor direto para a equipe, Hackman (1986) identificou cinco condições que, aliadas, podem promover a efetividade de equipes auto-gerenciadas. São elas:

- Direção geral do trabalho clara e engajadora;
- A estrutura da equipe promovendo um desempenho competente, através da concepção da tarefa, composição da equipe e expectativas enviadas relativas à gestão;
- Contexto organizacional que apoie um trabalho competente, através de recompensa, educação e estratégias de comunicação;
- Treinamento, com especialistas e consultas disponíveis, fornecidos em momentos apropriados;
- Recursos materiais adequados e disponíveis.

2.1.2. Engenharia de Software Baseada em Evidências

A pesquisa e prática baseada em evidências foi inicialmente desenvolvida na medicina. A Medicina Baseada em Evidências (do inglês, *Evidence-Based Medicine ou EBM*) ajuda profissionais e tomadores de decisão a basear suas decisões em evidências clínicas que já foram comprovadas e reunidas a partir do objetivo do estudo, ao invés de se basearem na opinião de um único especialista (BAILEY et al., 2008). Com isso, o paradigma baseado em evidencias teve

um considerável sucesso em se tratando de influenciar a pesquisa e prática na medicina clínica (BUDGEN et al., 2007).

Devido ao sucesso de EBM, as idéias e práticas nela envolvidas, em especial revisões sistemáticas da literatura, foram sendo adotadas e adaptadas por diversas outras áreas do conhecimento que dependem de dados empíricos para fins como a construção de teorias e obtenção de um entendimento da prática (BUDGEN et al., 2007). Em 2004, foi sugerido que esse paradigma baseado em evidencias poderia agregar valor à pesquisa e prática na área de Engenharia de Software, que, até então, tinha um fraco histórico do uso de dados empíricos para apoiar o desenvolvimento de modelos e métodos e também na adoção de práticas de outras disciplinas (WHITLEY, 1997, GLASS et al., 2002, GLASS et al., 2004 *apud* BUDGEN et al., 2007; KITCHENHAM, et al., 2004). Sendo assim, foi iniciado o uso da Engenharia de Software Baseada em Evidências, do inglês *Evidence-based Software Engineering* (EBSE). Mesmo com passos no processo semelhantes, isso não significa que os mecanismos subjacentes científico, tecnológico e organizacional aplicados em EBM são também aplicados em EBSE. Ambas possuem suas particularidades (KITCHENHAM et al., 2006).

O objetivo de EBSE é assegurar que as decisões estratégicas importantes são baseadas nas provas mais fiáveis disponíveis e integrar evidências atuais da pesquisa com experiência prática e valores humanos, para, com isso, melhorar a tomada de decisões relacionadas ao desenvolvimento de software e manutenção (KITCHENHAM et al.., 2004; DYBÅ et al.., 2005; BAILEY et al. 2007 ; DYBA et al.., 2008 ; KITCHENHAM et al.., 2008). Devido à possibilidade de existir uma grande distancia entre pesquisas investigativas e a prática, este é considerado um objetivo ambicioso (DYBA et al.., 2008). Porém, EBSE pretende fechar essa lacuna com o incentivo a uma maior ênfase no rigor metodológico enquanto mantém a concentração na relevância para a prática (DYBA et al., 2005 ; DYBA et al., 2008).

Para Kitchenham et al. (2004), a Engenharia de Software Baseada em Evidencias é importante pois fornece os mecanismos necessários para ajudar os profissionais de software a adotar as melhores práticas, usando as tecnologias apropriadas e evitando as tecnologias inadequadas. Budgen et al. (2007) acrescentam que EBSE tem o potencial de criar uma base muito mais sólida para os padrões, políticas e práticas. Por consequência, uma melhor qualidade do software é garantida, tranqüilizando e beneficiando os grupos interessados, como clientes e usuários (KITCHENHAM et al., 2004).

No entanto, existem algumas limitações da EBSE que merecem atenção e devem ser analisadas para que sejam minimizadas as consequentes ameaças à pesquisa que será conduzida. Para alguns autores, uma parte crítica de EBSE é encontrar todas as pesquisas relevantes a partir da enorme quantidade de trabalhos publicados (KITCHENHAM et al., 2004 ; BAILEY et al., 2007). Outros autores citam outras dificuldades que podem ser encontradas na EBSE:

> Um dos principais problemas para integrar os resultados do estudo em um corpo comum de conhecimento é a heterogeneidade dos relatórios dos estudos: (1) É difícil encontrar as informações relevantes, pois o mesmo tipo de informação está localizado em diferentes seções dos relatórios de diferentes estudos; (2) informações importantes são, por vezes, ausentes - por exemplo, informações do contexto são relatadas de forma diferente e desconsiderando a posterior generalização. Além disso, informações específicas de interesse para os profissionais são frequentemente ausentes, como uma discussão sobre o impacto global da tecnologia no projeto ou metas de negócios. (JEDLITSCHKA et al., 2008, p. 2, tradução nossa)

> Há problemas substanciais com evidências de forma sistemática, e não apenas porque o acúmulo de evidências de diferentes tipos de estudos é difícil. Um desafio específico na prática EBSE é que diferentes estudos empíricos sobre o mesmo fenômeno frequentemente relatam diferentes e, algumas vezes, contraditórios resultados. A menos que possamos compreender essas diferenças, integrar partes individuais de evidência é difícil. Isso aponta para a importância da divulgação de informações contextuais em estudos empíricos para ajudar a explicar os resultados conflitantes da investigação. (DYBA et al., 2005, p.2, tradução nossa)

De acordo com Dyba et al. (2005), Engenharia de Software Baseada em Evidências reúne e avalia as evidências encontradas através das cinco etapas seguintes:

1. Transformar o problema que se deseja solucionar ou a necessidade de informação em uma questão de pesquisa;
2. Pesquisar na literatura as melhores evidências disponíveis para responder a questão elaborada;
3. Avaliar criticamente as evidências, no tocante a sua validade, impacto e aplicabilidade;
4. Integrar as evidências avaliadas com experiências práticas, valores do cliente e as circunstâncias para tomar decisões sobre a prática;

5. Avaliar o desempenho até aqui alcançado e buscar formas de melhorá-lo.

Para a execução das etapas de 1 a 3 pode ser utilizado o Estudo de Mapeamento Sistemático ou uma Revisão Sistemática da Literatura, que são dois tipos de métodos da Engenharia de Software Baseada em Evidencias. Esses dois métodos são considerados estudos secundários por revisarem todos os estudos primários existentes relacionados à questão de pesquisa a ser investigada, tendo como objetivo a integração/síntese das evidências relevantes (KITCHENHAM, 2007). Abaixo, uma breve descrição dos estudos secundários mencionados:

- **Revisões Sistemáticas da Literatura** (do inglês – *Sistematic Literature Review* ou SLR), também referidas como Revisões Sistemáticas, identificam, avaliam e interpretam, de forma não enviesada, todas as evidências disponíveis relevantes para apoiar uma questão de investigação específica, área temática ou fenômeno (KITCHENHAM, 2007; DYBA et al., 2008 ; OATES e CAPPER, 2009). Esse método procura agregar resultados de uma determinada População (P), que faz o uso de uma Intervenção (I), inserida num Contexto (C) que pode ter um Resultado (O), comparando-o com a intervenção (C) (PETTICREW e ROBERTS, 2006). Portanto, uma SLR deve ser baseada na estrutura PICOC, do inglês: Population, Intervention, Context, Outcomes, e Comparison.

- **Estudo de Mapeamento Sistemático** (do inglês – *Systematic Mapping Study*) também é chamado de Estudo de Escopo (do inglês – *Scoping Study*), Revisão de Escopo (do inglês – *Scoping Review*) ou ainda Estudo Exploratório (*Exploratory Study*) (KITCHENHAM e CHARTERS, 2007 ; PETTICREW e ROBERTS, 2006 ; KITCHENHAM, 2007). Ao contrário de uma SLR, um Estudo de Mapeamento Sistemático não se destina a responder uma questão de investigação detalhada, devendo ser considerado apenas a População (P) e a Intervenção (I) (ARKSEY e O'MALLEY, 2005 ; BUDGEN et al., 2007 ; PETERSEN et al., 2007).

Para a realização da pesquisa aqui relatada, foi utilizado o Estudo de Mapeamento Sistemático, que, a partir daqui, poderá ser referido apenas como Mapeamento Sistemático ou MS. Maiores detalhes sobre esse método serão apresentados na seção seguinte (2.1.2.1).

2.1.2.1. Estudo de Mapeamento Sistemático

Como já mencionado na seção anterior (Seção 2.1.2), o Estudo de Mapeamento Sistemático é

projetado para fornecer uma visão mais ampla de pesquisa em uma área de tópico específico, muitas vezes com a utilização de múltiplas questões de pesquisa (KITCHENHAN, 2007). Um MS tem como objetivo identificar todas as evidências disponíveis sobre o tema, mapeando rapidamente os conceitos-chave que sustentam a área de pesquisa, além das principais fontes e tipos de evidência, no intuito de responder a uma questão de pesquisa (MAYS et al., 2001 apud ARKSEY e O'MALLEY, 2005 ; KITCHENHAN, 2007 ; BUGEN et al., 2007).

Apesar de essa definição dar muita ênfase à cobertura abrangente da literatura, em um Estudo de Mapeamento Sistemático pode haver graus muito diferentes de profundidade (ARKSEY e O'MALLEY, 2005). Além disso, um MS pode ser realizado como projeto autônomo, principalmente se a área a ser investigada é complexa ou ainda não foi revisada exaustivamente anteriormente (MAYS et al., 2001).

De acordo com Arksey e O'Malley (2005), existem, pelo menos, quatro razões para que seja realizado um Mapeamento Sistemático:

1. Para examinar a extensão, alcance e natureza do fenômeno a ser investigado. Essa é considerada uma maneira útil para o mapeamento de áreas de estudo, onde é difícil de visualizar a gama de materiais que possam estar disponíveis;
2. Para determinar se existe a necessidade da elaboração de uma completa revisão sistemática. Nesses casos, um mapeamento preliminar da literatura pode ser realizado a fim de identificar a relevância da elaboração de uma completa SLR;
3. Para resumir e divulgar resultados de pesquisas. Esse tipo de estudo de mapeamento pode descrever mais detalhadamente os resultados e a abrangência da pesquisa em determinadas áreas de estudo, proporcionando um mecanismo para resumo e divulgação dos resultados da pesquisa;
4. Para identificar lacunas de pesquisa na literatura existente. Esse tipo de Mapeamento Sistemático leva à disseminação de conclusões sobre o estado global da área investigada.

Os próprios autores Arksey e O'Malley (2005), que sugerem essas razões para um MS, enfatizam a importância da quarta e última razão, além de também ser bastante mencionada por outros autores, como Kitchenham (2007) e Budgen et al. (2007). Segundo Arksey e O'Malley (2005), um dos pontos fortes de um Mapeamento Sistemático é o fato de poder fornecer um método rigoroso e transparente, em um espaço de tempo menor do que em Revisões Sistemáticas.

O procedimento adotado para a condução do estudo de mapeamento sistemático da pesquisa aqui relatada será guiado por Arksey e O'Malley (2005), Kitchenham (2007), Budgen et al..(2007) e Petersen et al..(2007). De acordo com Arksey e O'Malley (2005), o processo de um MS não é linear, é interativo, o que exige que os pesquisadores se envolvam de forma atenta com cada fase da pesquisa e, quando necessário, repita os passos no intuito de garantir a maior cobertura da literatura. Após uma análise crítica sobre as etapas atribuídas a um Mapeamento Sistemático por cada um dos autores que serviram como guias desta pesquisa, adotamos as três etapas seguintes: Planejamento, Coleta de Dados e Análise. Essas etapas são resumidas abaixo.

Planejamento

O ponto inicial do processo de um MS é a identificação/definição da questão de pesquisa e atentar para que seja bem definida (ARKSEY e O'MALLEY, 2005 ; PETERSEN, 2007 ; OATES e CAPPER, 2009). Na verdade, esse primeiro passo é comum a todos os métodos de EBSE, como já apresentado na seção anterior (Seção 2.1.2) (DYBA et al.., 2005). Porém, para a formulação da questão de pesquisa de um MS, é melhor que seja utilizada uma estrutura resumida, como População (P) e Intervenção (I), a fim de introduzir pouca limitação e ampliar a pesquisa (PETERSEN, 2007).

Para uma melhor organização de como a pesquisa será procedida, um protocolo do estudo deve ser definido com antecedência (Kitchenham 2004). Porém, de acordo com Oates e Capper, 2009, na prática, isso é muito difícil de alcançar. Mas há a possibilidade de ser ajustado e aperfeiçoado, utilizando-se da documentação das alterações.

Coleta de Dados

Para a coleta dos dados, várias etapas foram encontradas. Primeiramente, **a busca deve ser conduzida** a fim de identificar estudos relevantes em bases de dados científicas ou, manualmente, em revistas e conferências da área temática (ARKSEY e O'MALLEY, 2005 ; PETERSEN, 2007). A **identificação desses estudos** deve ser feita com a leitura de seus resumos e palavras-chave (PETERSEN, 2007) .Os **critérios de inclusão e exclusão**, já definidos no protocolo, devem ser aplicados para selecionar os estudos relevantes para responder às questões de pesquisa (ARKSEY e O'MALLEY, 2005 ; PETERSEN, 2007).

Após selecionados, os estudos passam pelo processo de **extração dos dados**, onde as evidências mais relevantes avaliadas e organizadas (ARKSEY e O'MALLEY, 2005 ; PETERSEN, 2007).

Dentre esses passos, dois são também sugeridos por Dyba et al. (2005) para todos os métodos de EBSE, como apresentado na Seção 2.1.2. São eles: a condução da busca, que corresponde à etapa 2 do processo de EBSE, e a extração para avaliação das evidências, correspondente à etapa 3 do processo por ele sugerido. Porém, esta última tem suas particularidades. Por exemplo, a extração dos dados de uma SLR requer um nível de extração diferente do MS, que é muito mais amplo (KITCHENHAM, 2007). A extração em uma SLR trabalha com meta-análise continuamente com dados quantitativos coletados dos estudos (KITCHENHAM e CHARTERS, 2007).

Análise, Síntese e Apresentação dos dados

Os dados extraídos devem ser integrados e agrupados para serem analisados, baseando-se na freqüência de publicações em cada categoria identificada (ARKSEY e O'MALLEY, 2005; PETERSEN, 2007). Se relembrarmos as etapas de uma EBSE sugeridas por Dyba Et al. (2005) e apresentadas na Seção 2.1.2, encontramos semelhanças entre os passos relatados e a Etapa 4, que é constituída pela integração das evidências avaliadas para possibilitar a tomada de decisões.

Como resultado de todo esse processo, iniciado com o Planejamento (Seção 0), temos o Estudo de Mapeamento Sistemático, que é apresentado de forma consistente com as questões de pesquisa formuladas, fazendo uso de gráficos e tabelas para destacar similaridades e diferenças nos resultados (JACINTO, 2010).

2.2. Trabalhos Relacionados

Nesta seção serão descritos detalhadamente os dois estudos que são relacionados com a área temática investigada nesta pesquisa. Para cada estudo relacionado aqui apresentado, serão explorados os seguintes aspectos: Objetivos, Contexto, Procedimentos desenvolvidos, Análise dos resultados, Tratamentos de validade e suas Conclusões.

Nenhum trabalho que tente resolver o mesmo problema desta pesquisa, utilizando o mesmo método foi encontrado. Porém, foram encontrados três trabalhos que possuem objetivos semelhantes a algumas questões específicas desta pesquisa (Seção 1.1.2 do Capítulo 1).

Todos esses três trabalhos estudam o auto-gerenciamento em equipes de software que utilizam algum método ágil de desenvolvimento. Devido a esta ocorrência, este mapeamento sistemático irá incluir em sua análise estudos que abordam o auto-gerenciamento no contexto da abordagem ágil. Porém, ao contrário dos trabalhos que serão apresentados nessa seção, esta pesquisa abordará o auto-gerenciamento em equipes de software independentemente do uso de alguma metodologia, como, por exemplo, ágil. Isso porque, como apresentado na Seção 1.1.3 do Capítulo 1, o objetivo desta pesquisa é encontrar todas as evidências existentes sobre o auto-gerenciamento em equipes de software.

Além disso, os três trabalhos que serão apresentados aqui fizeram uso de métodos de pesquisa caracterizados como estudos primários, diferentemente da presente pesquisa, que irá conduzir um mapeamento sistemático, considerado um estudo secundário, como já apresentado na Seção 2.1.2.

Os estudos que serão explorados aqui são: *Organizing Self-Organizing Teams* (HODA et al., 2010), *Overcoming Barriers to Self-Management in Software Teams* (MOE et al., 2009) *Understanding Selforganizing Teams in Agile Software Development* (MOE et al., 2008).

Os estudos referenciados por esses três trabalhos que foram considerados possivelmente relevantes foram também analisados para a pesquisa.

2.2.1. Understanding Self-organizing Teams in Agile Software Development (MOE et al., 2008)

O objetivo do trabalho é estudar as barreiras enfrentadas com a introdução da auto-organização em equipes de software que utilizam o método ágil de desenvolvimento. Com isso, o estudo propõe a seguinte questão de pesquisa: Quais são as barreiras da introdução da auto-organização em equipes no desenvolvimento de software ágil?

Para tentar responder à questão da pesquisa, o autor desenvolveu um estudo de caso único com enfoque holístico, baseando-se por Yin (2003). O estudo foi conduzido por observação-participante entre os anos de 2006 e 2007, resultando em 45 observações que incluíam, além

da rotina de trabalho, fases e reuniões do processo de desenvolvimento do projeto. Durante essas observações, os pesquisadores tomavam nota sobre os diálogos, interações e atividades da equipe, para, em seguida, fazer a transcrição e integração das notas a fim de registrar detalhadamente cada sessão.

O estudo se passa em uma empresa de software localizada na Noruega que introduziu o método ágil Scrum[2] no intuito de aumentar a qualidade e melhorar a sensação e comunicação da equipe. Para que a equipe fosse habilitada a trabalhar com a metodologia Scrum, foi realizada uma oficina no ano de 2006. Portanto, o estudo foi iniciado antes da introdução da metodologia, o que permitiria a realização de uma pesquisa-ação[3].

Os resultados do estudo foram apresentados de acordo com os tipos de autonomia encontrados, sendo divididos em autonomia individual, autonomia interna e autonomia externa. Seguindo essa estrutura, os autores analisam os resultados obtidos e descrevem fatos que podem impactar na autonomia. Em seguida, o estudo faz um paralelo entre o que foi observado no caso estudado e o que outros pesquisadores propõem sobre o auto-gerenciamento de equipes.

A partir daí, o estudo conclui que a principal barreira da introdução ao auto-gerenciamento é a concentração em habilidades especializadas. Ou seja, o fato de uma equipe ter seus membros separados por suas especialidades e trabalharem individualmente nessas especialidades pode ser entendido como um fator que afeta o auto-gerenciamento. Por fim, o estudo sugere que sejam feitas mais pesquisas no sentido de descobrir novas barreiras na introdução a equipes ágeis. O estudo não apresentou limitações que podem ameaçar a validade da pesquisa.

Esse trabalho está relacionado a uma das questões específicas da pesquisa relatada nesta dissertação. A Q1.3, descrita na Seção 1.1.2 do Capítulo 1, que procura entender fatores técnicos, humanos e organizacionais que podem afetar o trabalho de equipes de software auto-gerenciadas.

[2] Scrum é uma das metodologias ágeis de desenvolvimento destinada ao gerenciamento de projetos de software.

[3] Pesquisa-ação é um método de procedimento de estudos que mistura pesquisa e participação com intervenção no contexto estudado.

2.2.2. Overcoming Barriers to Self-Management in Software Teams (MOE et al., 2009)

Produzido pelos mesmos autores do estudo apresentado na Seção 2.2.1, a questão central deste trabalho é: Como o trabalho em equipe deve ser organizado para o desenvolvimento de software?

Apesar de essa ser uma questão mais ampla, o estudo aponta para a direção do auto-gerenciamento, afirmando que a maioria dos métodos de desenvolvimento argumentam que as equipes devem se auto-organizar ou auto-gerir. Porém, o estudo se concentra nas principais barreiras e desafios ao nível da equipe e a nível organizacional na introdução de métodos de desenvolvimento que dependem de equipes de software auto-gerenciadas.

Desde o início, o estudo compara certos aspectos do auto-gerenciamento e do gerenciamento tradicional, dando mais ênfase ao auto-gerenciamento. Em seguida, o estudo destaca algumas vantagens trazidas pelo auto-gerenciamento, em relação ao gerenciamento tradicional, como a satisfação dos funcionários, menor volume de negócios e absenteísmo. Porém, o estudo indica que, apesar de a maioria das pesquisas apontarem para resultados positivos, existem algumas inconsistências. Além disso, o estudo cita a complexidade do auto-gerenciamento, aborda alguns desafios em sua adoção.

Para identificar o funcionamento dessas equipes e as barreiras e desafios que elas encontram, os pesquisadores investigaram, por três anos, cinco equipes em três empresas que introduziram o método ágil de desenvolvimento no intuito de aumentar a qualidade do software e melhorar o trabalho e a comunicação da equipe.

A coleta dos dados foi feita a partir de observações, com o apoio de imagens e anotações, e entrevistas, que foram gravadas e posteriormente transcritas. Todas as notas foram integradas, produzindo um registro detalhado de cada sessão acompanhada e foi usado o método de mate-etnografia para a análise e síntese dos dados das entrevistas transcritas, diálogos e anotações. O estudo não define qual o método de pesquisa foi utilizado para essa condução.
Os resultados obtidos, a partir de então, são apresentados de forma separada: barreiras encontradas a nível de equipe e as barreiras ao nível da organização. Para as equipes, o estudo afirma ter descoberto que as barreiras estão relacionadas ao compromisso individual, incapacidade de aprendizado e liderança individual e desenvolve aspectos sobre cada uma dessas três barreiras. Ao nível da organização, também foram encontradas três principais barreiras: recursos compartilhados (pessoas alocadas para mais de um projeto), controle organizacional e a cultura de especialistas.

Após o detalhamento do que constitui essas barreiras encontradas, o estudo indica algumas sugestões sobre o que pode ser feito para superá-las e pontua algumas dicas: organizar a multi-formação das pessoas, posicionar toda a equipe na mesma sala, apreciar indivíduos multifuncionais, construir confiança e compromisso e alocar as pessoas a um projeto por vez.

Por fim, o estudo indica as contribuições que espera ter dado para a prática e sugere que mudanças importantes devem ser identificadas e discutidas tanto no nível da equipe, quanto no nível da organização. Os autores apresentam ainda uma breve descrição de um trabalho que pretendem fazer futuramente.

Esse trabalho não incluiu discussões com comparações entre os resultados por ele obtidos e resultados de outros pesquisadores e também não indicou limitações que pudessem ameaçar a validade da pesquisa.

Assim como o trabalho relacionado apresentado na Seção 2.2.1, este estudo pode ser considerado relacionado à pesquisa relatada nesta dissertação por objetivar identificar fatores que podem afetar o trabalho das equipes de software auto-gerenciadas e, consequentemente, seus resultados. Portanto, o trabalho é relacionado à questão específica Q1.3 desta pesquisa.

2.2.3. Organizing Self-Organizing Teams (HODA et al., 2010)

Este trabalho foi motivado por entender que equipes de desenvolvimento de software que fazem uso de métodos ágeis são equipes auto-gerenciadas e, no entanto, a organização dessas equipes, na prática, não é bem compreendida. Portanto, o estudo visa investigar como esse tipo de equipe é organizada e propõe a pergunta: Como equipes ágeis auto-gerenciadas se organizam?

A pesquisa foi conduzida por meio de *Grounded Theory*[4]. De acordo com os autores, esse método foi escolhido por várias razões, entre elas: o fato de permitir o estudo das interações sociais e comportamentos, por ser adequado para áreas de pesquisa que ainda não foram muito exploradas e por se um método com crescente utilização no estudo da natureza social de

[4] Teoria Fundamentada ou, do inglês, Grounded Theory é uma metodologia científica que se refere à descoberta de uma teoria a partir da análise de dados.

equipes Ágeis. O estudo foi guiado pela abordagem dos desenvolvedores da *Grounded Theory*, Glaser e Strauss (1967).

Os dados foram coletados a partir de entrevistas face a face semi-estruturadas, utilizando perguntas abertas, e com o reforço de observações de várias práticas ágeis. A coleta dos dados e a análise dos dados ocorram de forma iterativa e, para isso, foi utilizado o método de comparações constantes[5].

Foram escolhidos participantes de duas culturas diferentes sem saber se esse fator iria influir nos resultados encontrados. Assim, 24 profissionais de 14 organizações de software diferentes, localizadas na Nova Zelândia e na Índia, foram entrevistados. Todas essas organizações faziam uso de métodos ágeis de desenvolvimento.

O estudo fornece uma base teórica para apoiar o desenvolvimento da pesquisa e fornece definições sobre o auto-gerenciamento no desenvolvimento ágil de software. Porém, dentre as referências utilizadas, somente uma não é um estudo sobre o auto-gerenciamento em métodos ágeis. Ou seja, o estudo não fornece embasamento sobre o auto-gerenciamento puro, sem o uso de uma metodologia que o favorece e que também pode afetar nos resultados obtidos pela equipe.

Como resultado, o estudo encontrou seis principais papeis que ocorreram nas equipes estudadas e descreveu como cada um deles trabalha e as percepções dos participantes entrevistados. Os papéis encontrados foram chamados de: Mentor, Coordenador, Tradutor, Campeão, Promotor e Exterminador. Segundo os autores, uma mesma pessoa pode atuar em mais de um desses papéis e também isso não indica que todos esses papéis estão presentes numa equipe ágil, mas organizações estudadas apresentaram pelo menos um deles e os papéis não foram afetados pelas diferentes culturas. Além dos papéis tradicionais, esses papéis são utilizados pelas organizações para facilitar a auto-organização das equipes.

Ao indicar os resultados obtidos através da análise dos dados coletados, o estudo discute alguns de seus resultados com resultados obtidos por outros pesquisadores, chegando a comparar um de seus resultados com o estudo de Moe et al.. (2008), descrito anteriormente na Seção 2.2.1 desta dissertação.

> A cultura da Nova Zelândia, apesar de ser individualista (ASTON et al.., 2008), não afetou negativamente a colaboração e a coordenação sobre essas equipes

[5] Um método para análise de dados que objetiva o desenvolvimento de uma Teoria Fundamentada, ou, do inglês, Grounded Theory. Neste método, os dados recém-coletados são constantemente comparados aos já coletados.

> ágeis. Isto está em contraste com os achados de Moe et al.. (2008) que descobriram que a alta autonomia individual provou ser uma barreira para a auto-organização em seu estudo de caso único. (HODA et al.., 2010, tradução nossa)

Algumas limitações atribuídas ao método de pesquisa escolhido (*Grounded Theory*) foram apresentadas, bem como as medidas tomadas para tentar minimizar seus efeitos na validade do estudo.

Por fim, o estudo conclui com a apresentação de sua pergunta de pesquisa, como forma de relembrar seu objetivo, um breve resumo sobre como cada um dos seis papéis encontrados trabalha e a contribuição esperada para a prática. Porém, o estudo não apresentou sugestões para possíveis trabalhos futuros.

- Mentor, que fornece orientação inicial, a compreensão, a confiança de métodos ágeis, e fomenta a adesão continua às práticas ágeis;
- Coordenador que coordena a comunicação e mudanças requeridas pelos clientes;
- Tradutor que traduz a linguagem de negócios dos clientes em terminologia técnica usada pela equipe, e vice-versa, a fim de melhorar a comunicação;
- Campeão que ganha o apoio da gerência sênior para criar equipes-piloto e para propagar mais equipes auto-organizadas em toda a organização;
- Promotor que assegura a colaboração e envolvimento do cliente para apoiar o funcionamento eficiente de equipes ágeis, e o
- Exterminador que remove os membros da equipe que dificultam a produtividade da equipe devido à sua incapacidade para se encaixar no meio do trabalho Ágil.

Este estudo pode ser considerado relacionado à pesquisa relatada nesta dissertação por tentar entender como funcionam as equipes auto-gerenciadas de software. Está mais diretamente ligado à Q1.1, que questiona como equipes auto-gerenciadas são definidas e visa encontrar também as características que a literatura indica estarem presentes em uma equipe de software auto-gerenciada.

2.3. Síntese dos Trabalhos Relacionados

Para encerrar este capítulo, a Tabela I irá apresentar uma síntese dos trabalhos apontados como relacionados à presente pesquisa.

Tabela I. Síntese dos Trabalhos Relacionados

	Understanding Selforganizing Teams in Agile Software Development (MOE et al., 2008)	**Overcoming Barriers to Self-Management in Software Teams (MOE et al., 2009)**	**Organizing Self-Organizing Teams (HODA et al., 2010)**
Objetivos	Estudar as barreiras enfrentadas com a introdução da auto-organização em equipes de software que utilizam o método ágil de desenvolvimento	Identificar barreiras e desafios ao nível da equipe e a nível organizacional na introdução de métodos de desenvolvimento que dependem de equipes de software auto-gerenciadas.	Compreender a organização de equipes auto-gerenciadas.
Questões de Pesquisa	Quais são as barreiras da introdução da auto-organização em equipes no desenvolvimento de software ágil?	Como o trabalho em equipe deve ser organizado para o desenvolvimento de software?	Como equipes ágeis auto-gerenciadas se organizam?
Método de Pesquisa	Estudo de caso único com enfoque holístico	Não apresentado	Grounded Theory
Coleta dos Dados	Observação-participante, incluindo, além da rotina de trabalho, fases e reuniões do processo de desenvolvimento do projeto.	Observações e Entrevistas	Entrevistas face a face semi-estruturadas e observações de várias práticas ágeis.
Contexto	Entre os anos de 2006 e 2007, uma equipe de uma empresa de software localizada na Noruega que introduziu o método ágil Scrum foi estudada.	Por três anos, foram estudadas cinco equipes em três empresas que introduziram o método ágil de desenvolvimento	24 profissionais de 14 organizações de software diferentes, localizadas na Nova Zelândia e na Índia, foram estudados.
Resultados	A barreira mais importante encontrada foi: **habilidades altamente especializada**s dos desenvolvedores e a correspondente divisão de trabalho. Além disso, a **redução da autonomia externa** pode ser uma barreira importante para a introdução de auto-	Ao nível de equipes, as barreiras estão relacionadas ao **compromisso individual**, **incapacidade de aprendizado** e **liderança individual**. Ao nível da organização, foram encontradas as principais barreiras: **recursos compartilhados** (pessoas	O estudo encontrou seis papéis presentes em equipes auto-gerenciadas de software: **Mentor**, que fornece orientação inicial, e fomenta a adesão continua às práticas ágeis; **Coordenador** que coordena a comunicação e

	organização de equipes.	alocadas para mais de um projeto), **controle organizacional** e a **cultura de especialistas**.	mudanças requeridas pelos clientes; **Tradutor** que traduz a linguagem de negócios dos clientes em terminologia técnica usada pela equipe, e vice-versa, a fim de melhorar a comunicação; **Campeão** que ganha o apoio da gerência sênior para criar equipes-piloto e para propagar mais equipes auto-organizadas em toda a organização; **Promotor** que assegura a colaboração e envolvimento do cliente para apoiar o funcionamento eficiente de equipes ágeis, e o **Exterminador** que remove os membros da equipe que dificultam a produtividade da equipe devido à sua incapacidade para se encaixar no meio do trabalho Ágil.
Limitações	Não apresentado	Não apresentado	Algumas limitações atribuídas ao método de pesquisa escolhido (Grounded Theory) foram apresentadas, como o fato de não permitir a generalização dos resultados.
Conclusão	Conclui que a principal barreira da introdução ao auto-gerenciamento é a concentração em habilidades especializadas. O estudo sugere que sejam	Indica as contribuições que espera ter dado para a prática e sugere que mudanças importantes devem ser identificadas e discutidas tanto no nível da equipe, quanto no nível da	O estudo é finalizado com um breve resumo sobre como cada um dos seis papéis encontrados trabalha e qual a contribuição esperada para a prática.

	feitas mais pesquisas no sentido de descobrir novas barreiras na introdução a equipes ágeis.	organização. Os autores apresentam uma breve descrição de um trabalho que pretendem fazer futuramente.	O estudo não apresentou sugestões para possíveis trabalhos futuros.

3. Metodologia

Este capítulo objetiva descrever o método utilizado para a condução desta pesquisa. A descrição detalhada do método é importante para garantir a confiabilidade da pesquisa, além de possibilitar a replicação do estudo por outros pesquisadores, a validação dos resultados aqui obtidos ou a comparação de resultados com o uso de outro contexto.

Para definir o método que será utilizado nessa pesquisa, primeiramente foi identificado o posicionamento filosófico que será seguido. Por concentrar-se mais na compreensão de como as diferentes pessoas fazem sentido do mundo e como atribuem significado às ações, do que na verificação de teorias, o Construtivismo foi considerado o posicionamento filosófico mais adequado (EASTERBROOK et al.., 2008). A partir dessa definição, foi feita a seleção das teorias, métodos e técnicas nele aceitos e que irão fazer parte do método de procedimento da pesquisa.

O Estudo de Mapeamento Sistemático foi, então, escolhido como método de procedimento desta pesquisa, devido ao fato de possibilitar a coleta de dados a partir da literatura existente e, portanto, o mapeamento do conhecimento atual sobre o auto-gerenciamento em equipes de software.

A abordagem metodológica utilizada para a condução dessa pesquisa é apresentada mais detalhadamente nas seguintes seções presentes neste capítulo: Quadro Metodológico (3.1), Premissas (3.2), Protocolo da Pesquisa (3.3) e Etapas da Pesquisa (3.4).

3.1. Quadro Metodológico

Segundo Marconi e Lakatos (2004), o quadro de referência metodológica confere rigor científico a um trabalho de pesquisa. Como mostrado na Tabela II, essa pesquisa optou por usar um método de abordagem **indutivo**, seguindo uma natureza das variáveis **qualitativa** com o apoio do método de procedimento de **Comparações Constantes**. Quanto ao escopo da pesquisa, optou-se por um **Mapeamento Sistemático da Literatura,** conforme classificação de Arksey O'Malley (2005), Bayley et al.. (2007) e Kitchenham (2007) e foi identificada a relação existente entre a **variável independente (X = Auto-gerenciamento)** e a **variável dependente (Y = Possíveis resultados do auto-gerenciamento de equipes de desenvolvimento de software)**.

Tabela II. Quadro Metodológico

Quadro Metodológico	
Método de Abordagem	*Indutivo*
Natureza das Variáveis	*Qualitativa*
Método de Procedimento	*Meta-Etnografia*
Quanto ao Escopo	*Mapeamento Sistemático da Literatura*
Variável Independente (X)	*Auto-gerenciamento*
Variável Dependente (Y)	*Possíveis resultados proporcionados pelo auto-gerenciamento de equipes de desenvolvimento de software*

Fonte: Elaboração Própria

O método indutivo, segundo Marconi e Lakatos (2007) é um processo mental por intermédio do qual, partindo de dados particulares, suficientemente constatados, infere-se uma verdade geral ou universal, não contida nas partes examinadas. Ainda de acordo com Marconi e Lakatos (2007), ao utilizar esse método de abordagem, devemos considerar três elementos/etapas fundamentais, são elas:

- **Observação dos fenômenos** - onde os fatos ou fenômenos são observados e analisados, objetivando o descobrimento das causas de sua manifestação.

- **Descoberta da relação entre os fenômenos** – onde se procura, por intermédio da comparação, aproximar os fatos ou fenômenos para descobrir a relação constante existente entre eles.
- **Generalização da relação** - onde a relação encontrada entre os fenômenos e fatos semelhantes é generalizada.

Apesar de alguns resultados da pesquisa apontarem para uma abordagem quantitativa, a natureza das variáveis da pesquisa é essencialmente qualitativa, pois não emprega procedimentos estatísticos para produzir resultados (STRAUSS e CORBIN, 1990) e procura entender fenômenos em um contexto específico com a utilização de uma abordagem mais naturalística (HOEPFL, 1997).

Esse tipo de abordagem foi selecionado para servir como base para essa pesquisa também por ser mais apropriada para obter um entendimento mais profundo e detalhado sobre as investigações, ambientes e comportamentos (MARCONI e LAKATOS, 2007). Nessa pesquisa, uma abordagem mais naturalística auxiliou na compreensão de como é definido o auto-gerenciamento em equipes de software, dos fatores que podem afetar o trabalho dessas equipes e dos possíveis resultados alcançados mediante a utilização do auto-gerenciamento em equipes de software. Esse entendimento foi possível devido à análise da literatura existente sobre o tema.

O método de procedimento chamado Meta-Etnografia, traduzido do inglês *Meta-ethnography*, foi baseado em Noblit et al. (1988) e é iniciado com a extração de conceitos-chave dos estudos encontrados, que fornecem informações relevantes para a pesquisa. A esses trechos, são associados códigos que indicam a que tipo ou categoria de informação o texto se refere.

O escopo foi definido com um estudo de Mapeamento Sistemático da Literatura, como forma de analisar e interpretar um conjunto de dados obtidos na literatura existente sobre uma questão de investigação particular, área temática ou fenômeno de interesse, baseando-se em evidências, como já abordado no Capítulo 2.

Para a condução de buscas automáticas neste Mapeamento Sistemático, uma String de Busca foi construída a partir das variáveis independente e dependente identificadas, como apresentado na Seção A4.1 do APÊNDICE A. De acordo com Marconi e Lakatos (2004), uma variável é classificada como independente quando é considerada fator determinante, condição ou causa para certo resultado, influenciando, determinando ou afetando uma outra variável,

chamada de variável dependente. Com isso, podemos identificar o auto-gerenciamento como a variável independente (X) da pesquisa e os resultados do auto-gerenciamento de equipes de software como a variável dependente (Y).

3.2. Premissas

Pesquisas realizadas no contexto de equipes de desenvolvimento de software descobriram que o desempenho da equipe está diretamente relacionado com a efetividade da coordenação da equipe (KRAUT et al.., 1995; HOEGL e GEMUENDEN, 2001;). Métodos de desenvolvimento mais atuais afirmam que as equipes devem se auto-organizar ou auto-gerenciar (Moe et al.., 2009). Uma equipe capaz de se auto-organizar é uma forma importante de auxiliar na colaboração do trabalho de conhecimento intensivo, tal como o desenvolvimento de software, e isso é especialmente favorecido em Métodos Ágeis, tais como XP e Scrum, propostos pelo Manifesto Ágil em 2001 (Moe et al.., 2009). O Manifesto Ágil (HIGHSMITH e FOWLER, 2001) afirma que as melhores arquiteturas, requisitos e projetos emergem de equipes auto-organizadas.

Com base nisso, a pesquisa parte da premissa de que o uso do auto-gerenciamento em equipes de software pode trazer resultados positivos de efetividade. Além disso, entendemos que os Métodos Ágeis têm sido apontados como os principais métodos a apoiarem o uso do auto-gerenciamento, no contexto de equipes de desenvolvimento de software.

3.3. Protocolo da Pesquisa

Como já mencionado neste capítulo, um Estudo de Mapeamento Sistemático (*Systematic Mapping Study*) será conduzido a fim de identificar e analisar a literatura existente sobre o tema pesquisado e selecionar os trabalhos mais relevantes para esta pesquisa, permitindo-nos, assim, responder às questões de pesquisa. Nesta seção, um resumo do protocolo utilizado para a condução de toda a pesquisa é apresentado. O protocolo completo encontra-se no APÊNDICE A deste trabalho.

3.3.1. Estratégia de Busca

No intuito de atingir uma maior cobertura da literatura existente, serão realizadas buscas automáticas e buscas manuais.

A busca automática é realizada a partir de uma String de Busca. Essa String de busca deve representar o conjunto de termos encontrados a partir das questões de pesquisa e influi na obtenção de uma maior cobertura da literatura existente.

A String de busca foi construída com base na combinação dos termos encontrados que foram concatenados através dos operadores booleanos “OR” e “AND”, como apresentado na Tabela III.

A busca manual não faz uso da String de Busca e é feita a partir de analises nos sites dos *journals* cada um dos anos de publicação, volumes e *issues*, além de uma busca por referências indicadas nos estudos já incluídos na pesquisa, chamada bola de neve (do inglês, *Snowball*).

Devido a suas particularidades, cada engenho de busca automática foi estudado e teve uma String de Busca adaptada a partir da string de busca original, a fim de garantir que nenhum estudo fosse omitido.

Tabela III. String de Busca

String de busca
("software team" **OR** "software application team" **OR** "software system team" **OR** "programming team" **OR** "software development team" **OR** "software work team" **OR** "software development work team" **OR** "software engineering team" **OR** "programming work team" **OR** "software project team" **OR** "software group" **OR** "software application group" **OR** "software system group" **OR** "programming group" **OR** "software development group" **OR** "software project group" **OR** "software engineering group" **OR** "software workgroup" **OR** "software development workgroup" **OR** "programming workgroup" **OR** "agile team" **OR** "agile development team" **OR** "agile group" **OR** "agile development group") **AND** ("empowered" **OR** "empowerment" **OR** "empowering" **OR** "autonomous" **OR** "autonomy" **OR** "semi-autonomous" **OR** "semi-autonomy" **OR** "self-managing" **OR** "self-managed" **OR** "self-management" **OR** "self-organizing" **OR** "self-organized" **OR** "self-organization" **OR** "self-manageable" **OR** "self-guided" **OR** "self-supervised" **OR** "self-coached" **OR** "self-coordinated" **OR** "self-determining" **OR** "self-determined" **OR** "self-designing" **OR** "self-designed" **OR** "self-sufficient" **OR** "self managing" **OR** "self managed" **OR** "self management" **OR** "self organizing" **OR** "self organized" OR "self organization" **OR** "self determining" **OR** "self determined" **OR** "self designing" **OR** "self designed" **OR** "self sufficient" **OR** "self manageable" **OR** "self guided" **OR** "self supervised" **OR** "self coached" **OR** "self coordinated")

Fonte: Elaboração Própria

3.3.2. Fontes de Busca

No intuito de atingir uma ampla cobertura dos estudos existentes sobre o tema de pesquisa, serão realizadas buscas automáticas e buscas manuais. Esta seção apresenta detalhadamente o procedimento para realização de cada uma dessas buscas, bem como os Engenhos de Busca utilizados.

3.3.2.1. Busca Automática

As buscas automáticas são feitas por meio de Engenhos de Busca, com o uso de uma String de Busca, como apresentado anteriormente na Seção 3.3.1. Para que um engenho de busca fosse selecionado, foram utilizados os seguintes critérios: (1) Deve ser acessado para consulta dos estudos através da Internet; (2) Deve ser considerado importante e relevante para a área de investigação;

A partir desses critérios, chegamos aos quatro engenhos de busca utilizados na pesquisa:

1. **ACM Digital library** (http://portal.acm.org/)
2. **IEEE Xplore Digital Library** (http://ieeexplore.ieee.org/)
3. **ScienceDirect** (http://www.sciencedirect.com/)
4. **Scopus** (http://www.scopus.com/)

3.3.2.2. Busca Manual

Para a busca manual, foram identificadas as principais revistas e anais de conferências em engenharia de software, além da busca e análise das referências dos estudos incluídos nesta pesquisa, processo denominado Bola de Neve (do inglês, *Snowball*).

Serão utilizadas cinco fontes de busca manual:

1. **Information and Software Technology (IST)** http://www.informatik.uni-trier.de/~ley/db/journals/infsof/

2. **IEEE Transactions on Software Engineering (TSE)**
   http://www.computer.org/portal/web/csdl/transactions/tse#1

3. **Software Practice and Experience (SPE)**
   http://onlinelibrary.wiley.com/journal/10.1002/%28ISSN%291097-024X

4. **ACM Transactions on Software Engineering Methodology (TOSEM)**
   http://portal.acm.org/pub.cfm?id=J790&CFID=15631611&CFTOKEN=81304600

5. **Journal of System and Software (JSS)**
   http://www.journals.elsevier.com/journal-of-systems-and-software/

3.3.3. Seleção dos Estudos

Para a realização do Mapeamento Sistemático, são pesquisados na literatura existente os estudos considerados relevantes para esta pesquisa. Para auxiliar na seleção desses estudos, alguns critérios de exclusão e inclusão foram definidos.

Os seguintes tipos de estudos serão excluídos:

1. Estudos não escritos em inglês.
2. Artigos convidados, tutoriais, *key-note speech*, relatórios de workshop, teses, dissertações, relatórios técnicos, livros.
3. Artigos que expressam apenas pontos de vistas pessoais, opiniões de especialistas ou relatos de experiências.
4. Documentos que não sejam artigos completos (e.x. apresentação de PowerPoint ou resumo estendido).
5. Artigos duplicados, que já foram encontrados em outras fontes.
6. Estudos relacionados a Ciência da Computação que não sejam claramente da área de Engenharia de Software ou Sistemas de Informação.
7. Estudos que não respondem a nenhuma das questões da pesquisa;
8. Trabalhos que não tiveram seus arquivos encontrados/não recuperados;
9. Estudos secundários que obtiverem pontuação menor do que 1 na Avaliação da Qualidade;

Os estudos que obedecerem ao primeiro critério abaixo e a um dos seguintes serão incluídos na pesquisa:

1. Artigos completos publicados em revistas ou conferências revisadas, que relatem o auto-gerenciamento no desenvolvimento de software e respondam a, pelo menos, uma das questões de pesquisa específicas;
2. Estudos experimentais ou *Empirical Studies*;
3. Estudos Secundários, aqueles que dependem de estudos primários;
4. Estudos Teóricos, que apresentem conceitos baseados em um entendimento de uma área, referenciando outros trabalhos;

3.3.4. Avaliação da Qualidade

Cada um dos estudos considerados relevantes para esta pesquisa será avaliado no que se refere à sua qualidade. O intuito dessa avaliação é compreender a força das evidências encontradas na literatura sobre o auto-gerenciamento em equipes de software, e, com isso, responder à segunda questão central desta pesquisa (Q2), apresentada no Capítulo 1 Seção 1.1.2.

Essa etapa será realizada por dois pesquisadores individualmente, com o uso de formulários pré-definidos. Após a finalização da avaliação individual, ambos os pesquisadores discutirão cada uma das divergências e chegarão a um acordo comum sobre elas, gerando uma única lista contendo a pontuação final de cada estudo e posterior normalização, permitindo a comparação.

De acordo com os critérios de inclusão já apresentados na seção anterior, esta pesquisa deverá incluir estudos do tipo empírico, teórico e secundários. Como mencionado no Capítulo 2 desta dissertação, existem diferenças consideráveis entre esses tipos de estudos e, portanto, foram elaborados formulários com critérios distintos para os três casos.

O formulário utilizado para a avaliação da qualidade de estudos empíricos contém 10 critérios, baseados em Dyba et al. (2008). O formulário para estudos teóricos contém apenas 2 critérios de qualidade, que foram baseados em Beecham et al. (2006). Já o formulário de avaliação dos os estudos secundários possui quatro critérios, obtidos de *Centre for Reviews and Dissemination* (2007). Todos esses formulários estão disponíveis na Seção A6 do APÊNDICE A.

3.3.5. Extração dos Dados

A Extração dos Dados será realizada pelos pesquisadores, individualmente, com o

preenchimento de um formulário, desenvolvido no Microsoft Excel ™. O Formulário D contém informações detalhadas sobre a identificação de cada um dos estudos incluídos na pesquisa, como, por exemplo: ano de publicação, autores, tipo de estudo, método de pesquisa, foco do estudo, etc. No Formulário E serão documentadas informações sobre os estudos excluídos na segunda etapa de seleção da pesquisa, como: título, local de publicação e critério usado para exclusão. No Formulário F, serão preenchidos os dados relevantes para apoiar as respostas das questões desta pesquisa. Todos esses formulários são apresentados na Seção A7 do APÊNDICE A.

3.3.6. Análise e Síntese dos Dados

Para proceder a síntese dos dados obtidos a partir da extração será realizada uma Meta-Etnografia, delineada por Noblit et al. (1988).

Resumidamente, esse procedimento consistirá em identificar conceitos-chave dos estudos que são relevantes para a pesquisa e categorizá-los em forma de tabelas de acordo com as questões de pesquisa, para que a informação seja interpretada.

3.4. Etapas da Pesquisa

A pesquisa foi iniciada com a execução de uma revisão bibliográfica, buscando os estudos mais relevantes sobre o conceito de auto-gerenciamento. Posteriormente, foi realizada uma breve pesquisa com a finalidade de identificar alguns estudos sobre o auto-gerenciamento de equipes de software. O resultado destas etapas da pesquisa foram apresentados no Capítulo 2 do presente trabalho.

A partir de então, a etapa de planejamento da pesquisa foi iniciada. Nela, as questões e as estratégias da pesquisa foram formuladas, como apresentado no Capítulo 1 e na Seção 3.3.

Após o planejamento da pesquisa, a coleta dos dados foi iniciada com a condução das buscas automáticas e manuais, reportando os estudos existentes. Esses estudos passaram pela Seleção 1, que resultou nos estudos considerados potencialmente relevantes para a pesquisa. Em seguida, passaram pela Seleção 2, que objetivou identificar os estudos realmente relevantes para a pesquisa, para que, então, a avaliação da qualidade e a extração dos dados contidos nos estudos fossem realizadas. Quando necessário este ciclo era refeito.

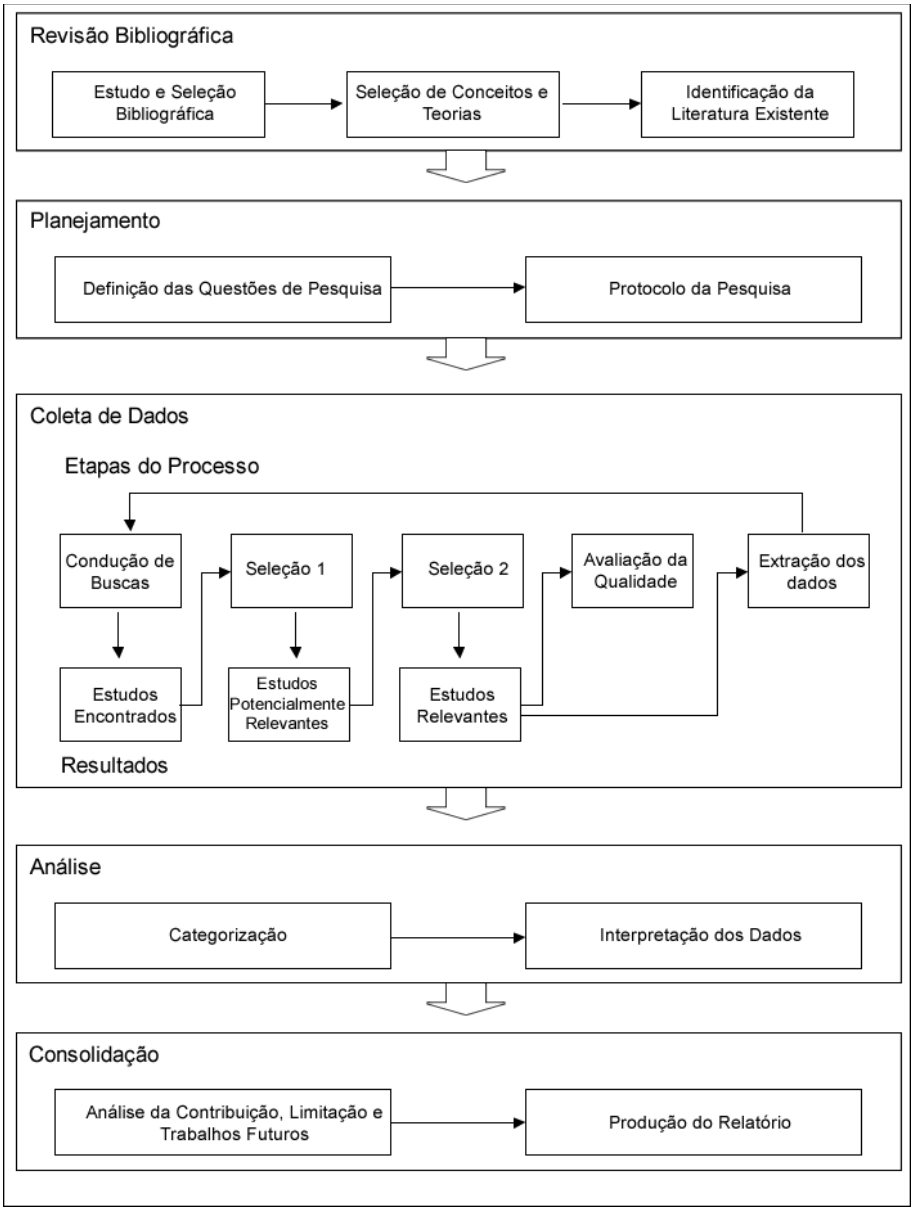
Com os dados já coletados, foi feita a análise e síntese desses dados para posterior categorização e interpretação, resultando, finalmente, no mapeamento sistemático, que reúne o conhecimento existente sobre o tema pesquisado. Os resultados obtidos com a conclusão do mapeamento sistemático são apresentados no Capítulo 4.

Essas três últimas etapas (Planejamento, Coleta de Dados e Análise e Síntese) foram previstas na Seção 2.1.2.1 do Capítulo 2 deste relatório.

Após a finalização do mapeamento sistemático, o estudo foi consolidado com a análise de suas contribuições, limitações e trabalhos futuros, apresentada no Capítulo 5.

Todas essas etapas da pesquisa estão sintetizadas na Figura 1. Para apoiar a documentação dos dados nos processos de seleção, extração, avaliação da qualidade e síntese dos desses dados, foram utilizadas ferramentas como Mendeley (www.mendeley.com) e Dropbox (www.dropbox.com).

Figura 1. Etapas da Pesquisa



Fonte: Adaptado de Petersen et al. (2007) e de França (2009)

4. Resultados

Este capítulo tem como objetivo apresentar os resultados obtidos a partir desta pesquisa. Para isso, o capítulo abordará três partes distintas, de acordo com o proposto por Jedlitschka e Pfahl (2005), como segue:

- Execução – Apresenta como foram realizados o processo de coleta dos dados e descreve a equipe envolvida nesta pesquisa. Durante a descrição do processo de coleta dos dados, serão consideradas as alterações ocorridas no Protocolo, apresentado no Capítulo 3.
- Análise – Apresenta os dados gerais obtidos no estudo de mapeamento, como o total de estudos retornados, as principais fontes que contribuíram para a pesquisa, os estudos incluídos no mapeamento, a distribuição temporal desses estudos, entre outros. Além disso, será apresentado o mapeamento realizado a partir das evidências encontradas e o desenho da conclusão e validação.
- Interpretação – Apresenta a interpretação feita a partir dos dados apresentados na Análise e uma discussão, comparando os resultados aqui obtidos com os já consolidados na literatura. Além disso, serão apresentadas as limitações do estudo.

4.1. Execução

Esta seção irá descrever como se deu o passo a passo desta pesquisa. As etapas de Amostragem, Processo de Coleta de Dados e Processo de Validação serão aqui detalhadas, ressaltando os desvios do Protocolo de Pesquisa, apresentado no Capítulo 3, que ocorreram durante todo o processo e as medidas que foram tomadas para tratar esses desvios.

4.1.1. Processo de Coleta de Dados

Com o andamento da pesquisa, realizada entre Outubro de 2010 e de 2011, foi identificada a escassez de trabalhos relevantes retornados das nove fontes de busca (4 para Busca Automática e 5 para Busca Manual) anteriormente selecionadas para análise. Portanto, houve a necessidade da seleção de novas fontes, seguindo os critérios de importância e acessibilidade já apresentados em na Seção 3.3.2 do Capítulo 3.

Desse modo, foram acrescentadas 5 novas fontes de busca, sendo 2 automáticas e 3 manuais, além de uma busca informal e da análise dos trabalhos incluídos em outras 3 SLRs do grupo HASE relacionadas (CARDOZO et al., 2010; FRANÇA et al., 2011).

A Busca Informal foi realizada no Google Scholar a princípio como um teste para saber se os estudos retornados por ela já haviam sido identificados nas outras fontes de busca. Esse procedimento não seria documentado, pois não houve rigorosidade em sua condução. Porém, mesmo sem a documentação e análise de todos os estudos retornados, foram identificados 3 novos estudos considerados potencialmente relevantes. Dentre os novos estudos encontrados, um estudo foi considerado relevante e incluído na pesquisa.

Também entre as novas fontes analisadas, a Agile Conference foi selecionada devido ao grande número de estudos relevantes já encontrados que focam no uso de metodologias Ágeis e que indicam seu favorecimento ao auto-gerenciamento. Também por esse motivo, as buscas manuais foram limitadas entre o ano de 2001 e 2010, já que o primeiro foi o ano em que o Manifesto Ágil (Agile Manifesto, 2001) foi divulgado.

Com isso, a pesquisa contou com um total de 16 fontes de busca. A Tabela IV lista as fontes que não haviam sido planejadas na Seção 3.3.2 do Capítulo 3.

Tabela IV. Lista de Novas Fontes de Busca Analisadas

Tipo	Engenhos Selecionados Posteriormente	Endereço de Acesso
Automática	1. Jstor	http://www.jstor.org/
Manual	2. International Symposium on Empirical Software Engineering and Measurement (ESEM)	http://ieeexplore.ieee.org/xpl/conhome.jsp?punumber=1001929
Manual	3. Evaluation and Assessment of Software Engineering (EASE)	http://www.scm.keele.ac.uk/ease/archive.html
Manual	4. Human and Social Factors of Software Engineering (ICSE - HSSE)	http://portal.acm.org/event.cfm?id=RE228&tab=pubs&CFID=15631611&CFTOKEN=81304600
Manual	5. Agile Conference	http://www.agileconference.org/
Manual	6. Outras Revisões Sistemáticas Relacionadas	CARDOZO, E. S. F. ; ARAÚJO NETO, J. BENITO ; BARZA, A.; FRANCA, A. C.; DA SILVA, F. Q. B. SCRUM and Productivity in Software Projects : A Systematic Literature Review. EASE'2010. p.1-4, 2010.

	FRANÇA, A. C. C.; GOUVEIA, T. B.; SANTOS, P. C. F.; SANTANA, C. A.; SILVA, F. Q. B. Motivation in Software Engineering : A Systematic Review Update. EASE'2011, p. 1-10, 2011.
7. Busca Informal	scholar.google.com.br/

Fonte: Elaboração Própria

Os 43 estudos relevantes para a pesquisa retornados das fontes de busca selecionadas foram analisados em sua totalidade e os trechos considerados mais importantes para responder cada uma das questões desta pesquisa foram documentados em uma planilha Excel™ e depois sintetizados, como apresentado nas Seções 3.3.5 e 3.3.6 do Capítulo 3. Além dos trechos-chave, também foram coletados dados de identificação dos estudos, como: autores, ano de publicação, tipo de estudo, etc (descrito na Seção 3.3.5).

4.1.2. Equipe Envolvida

Segundo Dyba et al. (2008), o possível viés introduzido pelos pesquisadores na seleção dos estudos realizada em estudos secundários, como o mapeamento sistemático, é uma das principais limitações desse método. Portanto, para a realização de uma pesquisa de maior qualidade e validade, com exceção da análise e síntese dos resultados, todas as etapas subjetivas da pesquisa contaram com, pelo menos, dois pesquisadores. Em casos de discordâncias, um terceiro pesquisador foi inserido na etapa para resolver o conflito. Desse modo, a pesquisa contou com um total de 7 pesquisadores, além do autor desta dissertação. Em diferentes níveis de participação, cada pesquisador colaborou em, pelo menos, uma das etapas da pesquisa, como: a seleção dos estudos potencialmente relevantes na busca automática, a avaliação da qualidade dos estudos incluídos e a verificação dos desacordos de dois pesquisadores.

Dentre os pesquisadores que colaboraram com a pesquisa, 3 são pós-graduados em Ciência da Computação, pelo Centro de Informática (CIn) da Universidade Federal de Pernambuco (UFPE) e 4 são mestrandos também em Ciência da Computação, na mesma instituição. Dentre os 8 pesquisadores envolvidos, 7 já haviam participado de outros estudos de Mapeamento Sistemático ou Revisão Sistemática da Literatura.

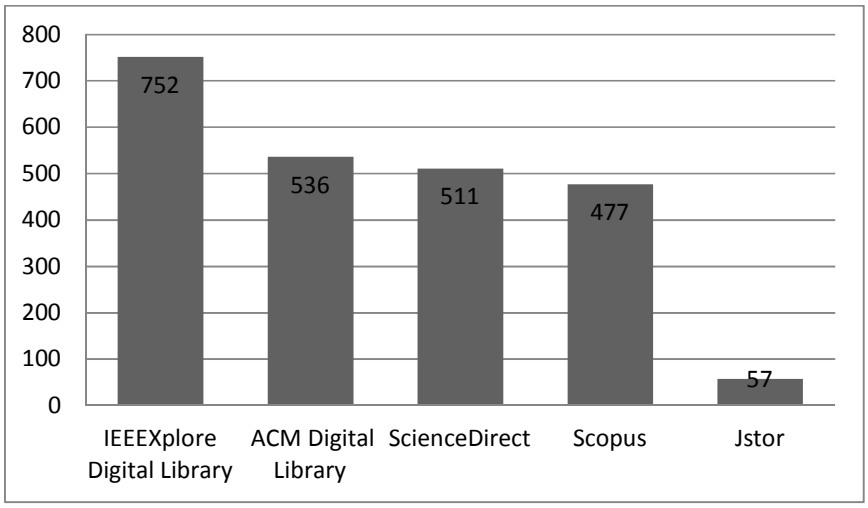
4.2. Análise

Esta seção está dividida em Apresentação e Análise dos Dados Gerais (Seção 4.2.1), onde alguns dados obtidos são apresentados para apoiar algumas conclusões a nível estatístico. Nesta seção também será apresentado o Mapeamento das Evidências (Seção 4.2.2), onde os dados serão organizados de acordo com as questões de pesquisa, e o Desenho da Conclusão e Validação (Seção 4.2.3), que irá indicar o que realmente os dados significam e suas implicações com as questões de pesquisa.

4.2.1. Apresentação e Análise dos Dados Gerais

A partir das Fontes de Busca, apresentados anteriormente na Seção 3.3.2 do Capítulo 3 e atualizados na Seção 4.1.1 deste Capítulo, a busca primária retornou um total de 2646 trabalhos. Dentre esses, 2333 são provenientes da Busca Automática e 313 da Busca Manual. Na Busca Automática, foram identificados 752 trabalhos no IEEE Xplorer, 536 no ACM Portal, 511 no Science Direct, 477 no Scopus e 57 no Jstor. A Figura 2 apresenta essa distribuição.

Figura 2. Distribuição de estudos retornados por engenho de Busca Automática

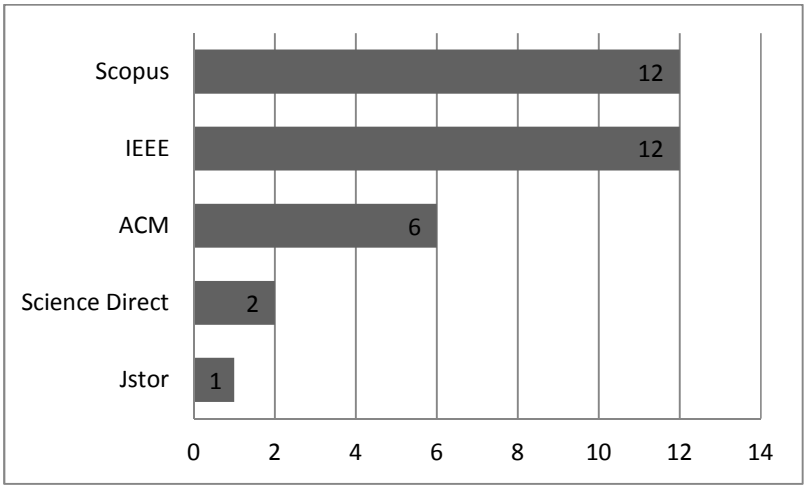


Fonte: Elaboração Própria. A partir dos dados do Mapeamento Sistemático.

Apesar de o IEEE Xplorer, ACM e o Science Direct terem retornado maior quantidade de estudos na busca primária, os engenhos de busca automática mais representativos foram o Scopus e o

IEEE. Juntos, os dois engenhos contribuíram com 55% dos estudos incluídos nesta pesquisa. Em seguida, o ACM representou 13%, o Science Direct 4% e o Jstor com apenas 2% da representatividade para a pesquisa. A Figura 3 apresenta a quantidade de estudos selecionados por cada Engenho de Busca Automática.

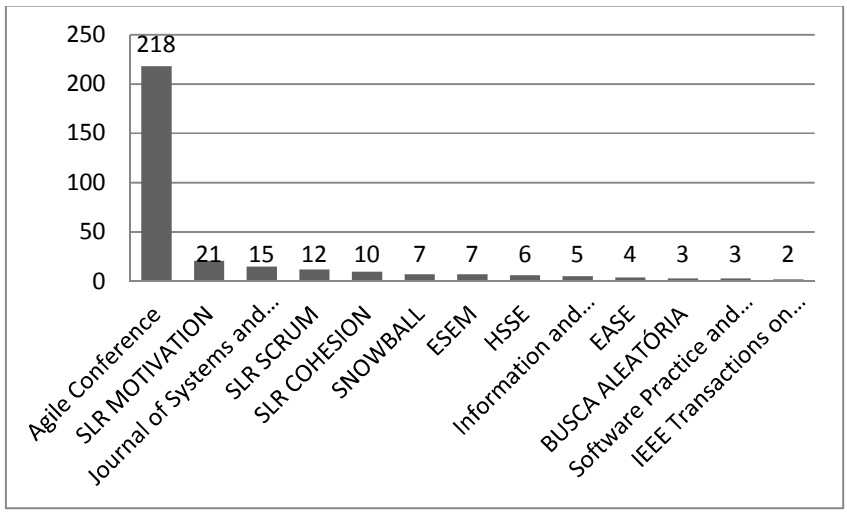
Figura 3. Distribuição de Estudos incluídos na pesquisa a partir da Busca Automática



Fonte: Elaboração Própria. A partir dos dados do Mapeamento Sistemático.

Já na Busca Manual, a fonte de busca que teve maior retorno foi a Agile Conference, com 218 estudos identificados, seguido pela SLR de Motivação (FRANÇA et al., 2011), com 21 estudos. Em seguida, os que mais retornaram foram o *Journal of System and Software (JSS)*, com 15 estudos, a SLR de Scrum (CARDOZO et al., 2010), com 12 estudos e a SLR de Coesão, com 10. A técnica de Bola de Neve e o *ESEM* vieram logo em seguida, ambos com 7 estudos retornados. Depois, o *HSSE* com 6, o IST com 5, o EASE com 4, a Busca Aleatória com 3, o SPE com 3 e finalmente o *TSE* com apenas 2 estudos retornados como apresentado na Figura 4.

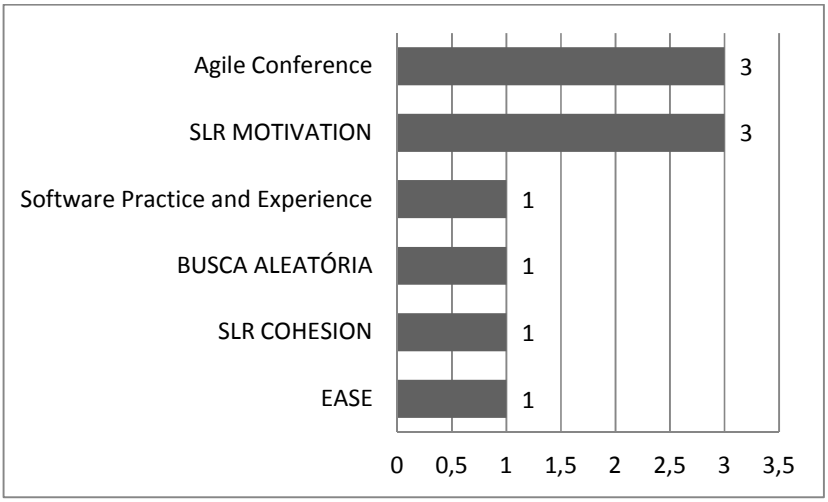
Figura 4. Distribuição de estudos retornados por engenho de Busca Manual



Fonte: Elaboração Própria. A partir dos dados do Mapeamento Sistemático.

Apesar da grande diferença na quantidade de artigos retornados, a Agile Conference e a SLR de Motivação (FRANÇA et al., 2011) tiveram igual representatividade, pois ambas contribuíram com 3 estudos cada, num total de 13% dos estudos incluídos. As outras fontes analisadas manualmente tiveram baixa representatividade ou nenhuma representatividade. Somente outras 4 fontes manuais contribuíram. Porém, com apenas 1 estudo incluído em cada, como mostra a Figura 5.

Figura 5. Distribuição de Estudos Incluídos na pesquisa a partir da Busca Manual



Fonte: Elaboração Própria. A partir dos dados do Mapeamento Sistemático.

Como mostra a Tabela V, após a primeira etapa de seleção dos estudos, descrita na Seção 3.3.3 do Capítulo 3, 576 trabalhos foram considerados potencialmente relevantes a esta pesquisa. Dentre eles, 9 não tiveram acesso disponível através da licença concedida pela UFPE e 533 foram excluídos pelos critérios de exclusão, também apresentados na Seção 3.3.3. Sendo assim, 43 trabalhos foram considerados realmente relevantes à pesquisa. Os detalhes dos trabalhos incluídos e excluídos estão disponíveis no APÊNDICE B e APÊNDICE C deste trabalho, respectivamente.

Com isso, é possível verificar a grande discrepância entre o número de trabalhos retornado da Busca Primária (n=2646) e a quantidade de trabalhos incluídos na pesquisa (n=43). Porém, essa redução nos números é considerada comum em estudos secundários, como SLRs e MS. Isso se deve às características dos engenhos de busca automáticos disponíveis (KITCHENHAM, 2007).

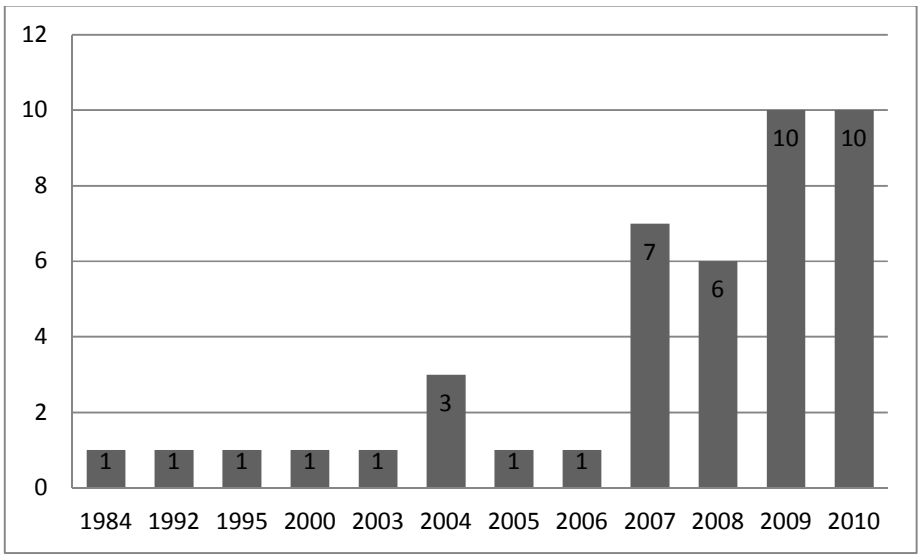
Tabela V. Processo de Seleção dos Estudos

	Eng. Busca	Estudos Retornados	1ª Seleção (Título, Abstract e Palavras-chave)	2ª Seleção (Introdução, Metodologia, Resultados e Conclusão)			
			Potencialmente Relevantes	Excluídos			Incluídos
				Não Relevantes	Duplicados	Não Recuperados	
AUTOMÁTICA	**ACM Digital Library**	536	57	42	6	3	6
	IEEEXplore Digital Library	752	104	89	3	0	12
	ScienceDirect	511	41	34	4	1	2
	Scopus	477	58	41	4	1	12
	Jstor	57	3	0	0	2	1
	Total	**2333**	**263**	**206**	**17**	**7**	**33**
MANUAL	Information and Software Technology (IST)	5		3	2	0	0
	IEEE Transactions on Software Engineering (TSE)	2		2	0	0	0
	Software Practice and Experience (SPE)	3		2	0	0	1
	ACM Transactions on Software	0		0	0	0	0

	Engineering Methodology (TOSEM)					
	Journal of System and Software (JSS)	15	13	2	0	0
	International Symposium on Empirical Software Engineering and Metrics (ESEM)	7	4	3	0	0
	Evaluation and Assessment of Software Engineering (EASE)	4	2	0	1	1
	Human and Social Factors of Software Engineering (ICSE - HSSE)	6	5	1	0	0
	SLRs HASE*	43	31	8	0	4
	Agile Conference	218	209	5	1	3
	Snowball	7	6	1	0	0
	Pesquisa Aleatória	3	2	0	0	1
	Total	**313**	**279**	**22**	**2**	**10**
	TOTAL	2646	485	39	9	43

Mesmo com a limitação da Busca Manual com ano de publicação entre 2001 e 2010, descrita na Seção 4.1.1, 4 estudos publicados antes do ano 2001 foram incluídos na pesquisa a partir da busca automática, sendo o mais antigo publicado em 1984. A Figura 6 apresenta como se deu essa distribuição e evidencia o considerável aumento da produção de trabalhos sobre o tema pesquisado entre os anos de 2007 e 2010. Juntos, esses 4 anos totalizaram 76% dos trabalhos incluídos nesta pesquisa.

Figura 6. Distribuição Temporal dos Estudos Selecionados



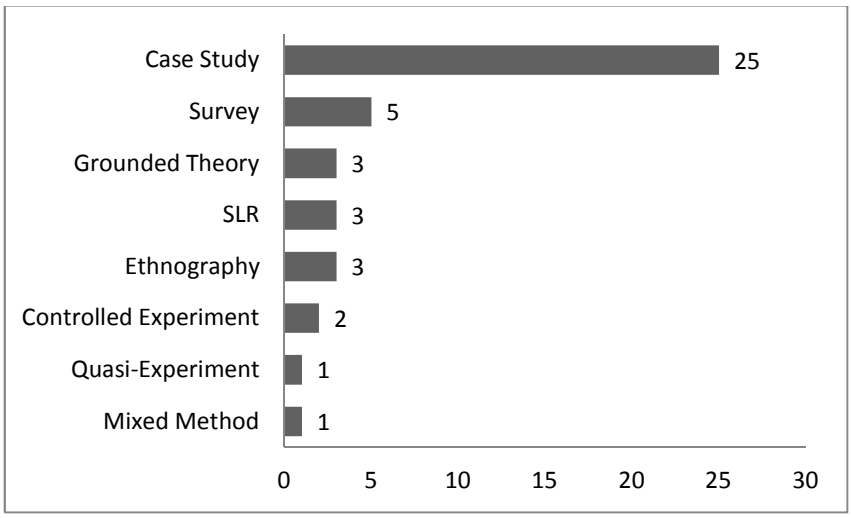
Fonte: Elaboração Própria a partir dos dados do Mapeamento Sistemático

Este mapeamento identificou o envolvimento de 81 diferentes autores entre os estudos selecionados. Dentre os que publicaram mais estudos relevantes a esta pesquisa, estão, respectivamente: Moe, N. B. (9 estudos); Dingsøyr, T. (7 estudos); Dybå, T. (6 estudos) e Sharp, H. (3 estudos).

Foram identificadas instituições de 16 países entre os estudos relevantes a esta pesquisa. Dentre os que mais contribuíram, estão, respectivamente: Noruega (11 estudos), Estados Unidos (9 estudos), Austrália (6 estudos), Reino Unido (4 estudos), Irlanda (4 estudos) e Finlândia (3 estudos). O Brasil contribuiu com apenas 1 estudo, elaborado por alunos do Centro de Informática da Universidade Federal de Pernambuco (UFPE), juntamente ao Professor Silvio Meira.

Dos estudos incluídos nesta pesquisa, 40 são estudos empíricos e outros 3 estudos secundários. Todos os 3 estudos secundários encontrados optaram pela Revisão Sistemática da Literatura como método de seu procedimento. Já os estudos empíricos foram distribuídos entre 7 métodos diferentes, como apresenta a Figura 7.

Figura 7. Distribuição dos Métodos de Pesquisa dos Estudos Empíricos Incluídos



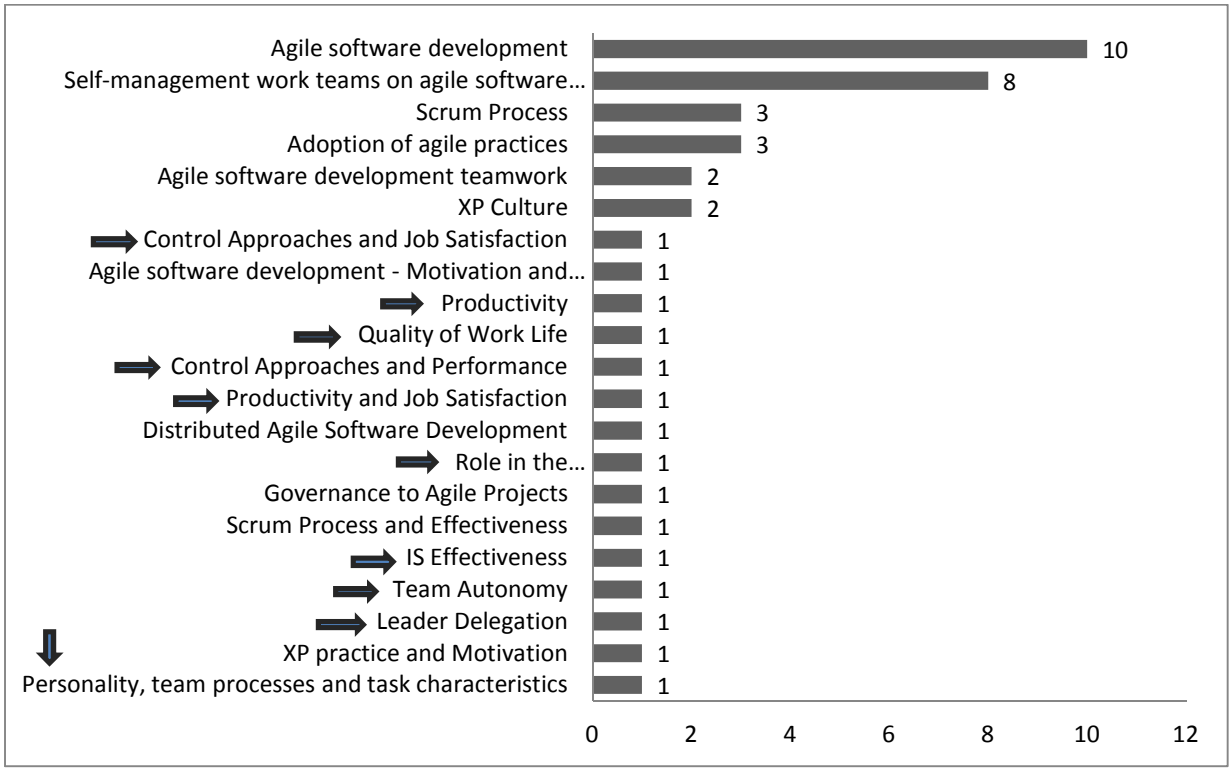
Fonte: Elaboração Própria. A partir dos dados do Mapeamento Sistemático.

A partir da documentação dos enfoques dados a cada um dos 43 estudos incluídos na pesquisa, foi possível identificar que muitos deles eram direcionados ao estudo de uma mesma área temática. Com isso, foram encontradas 19 áreas temáticas pesquisadas entre os estudos incluídos, apresentadas na Figura 8. Dentre essas, 10 são direcionadas a Metodologias Ágeis e 9 a áreas de Aspectos Humanos, que estão indicadas com uma seta ao lado. No total, 76% (33/43) estudos foram direcionados ao estudo de algum aspecto de metodologias ágeis de desenvolvimento de software e 23% (10/43) foram direcionados ao estudo de outros assuntos não relacionados a Métodos Ágeis.

É interessante observar que dentre esses 23% (10/43) estudos, 13% (6/43) foram publicados após a divulgação do Manifesto Ágil (2001) e, apesar de não ser o foco dos estudos, 33% (2/6) desses também aborda Metodologias Ágeis em sua pesquisa. Isso significa que 81% (35/43) dos estudos considerados relevantes a esta pesquisa, no mínimo, abordam Metodologias Ágeis em seu conteúdo. Porém, apenas 18% (8/43) dos estudos considerados relevantes à pesquisa são focados no auto-gerenciamento de equipes de software. Os outros 81% (35/43) foram incluídos na pesquisa por abordarem o auto-gerenciamento e apoiarem a resposta de pelo menos uma das questões de pesquisa, além de se enquadrarem nos outros critérios apresentados na Seção 3.3.3, porém não são focados no estudo do auto-gerenciamento.

As metodologias de desenvolvimento mais utilizadas nos estudos que abordaram metodologias ágeis foram respectivamente Scrum, com 48% (17/35) dos estudos e E*xtreme Programming* (XP), com 40% (14/35) estudos. Outras metodologias ágeis abordadas foram *Lean*, ASD e combinações de princípios de XP com Scrum e XP com DSDM.

Figura 8. Distribuição dos Focos dados aos Estudos Incluídos por área



Fonte: Elaboração Própria. A partir dos dados do mapeamento.

4.2.2. Mapeamento das Evidências

Nessa seção, são apresentados os resultados para cada questão de pesquisa. Para responder à primeira questão central desta pesquisa, visando o conhecimento de como o auto-gerenciamento de equipes de software é compreendido na literatura, serão apresentadas evidências de suas três questões específicas, como segue. Na Seção 4.2.2.1 são apresentadas as evidências quanto à definição do auto-gerenciamento em equipes de software. Na Seção 4.2.2.2 serão apresentadas as evidências que apoiam o conhecimento dos efeitos que o auto-gerenciamento pode causar em equipes de desenvolvimento de software. Na Seção 4.2.2.3, serão apresentadas as evidências que apóiam o conhecimento dos fatores técnicos, humanos e

organizacionais que podem afetar o trabalho de equipes de software auto-gerenciadas. Por fim, a Seção 4.2.2.4 apresenta os dados que apóiam o conhecimento da força da evidência dos estudos incluídos na pesquisa.

Todas as evidências aqui citadas são devidamente referenciadas pelos 43 estudos incluídos na pesquisa. Os números das referências são precedidos por SM (Self-Management), no intuito de identificar cada referência do mapeamento sistemático. Como anteriormente mencionado, as informações de cada estudo incluído estão disponíveis no APÊNDICE B.

4.2.2.1. Q1.1 Definição do auto-gerenciamento de equipes de desenvolvimento de software

Q1.1. Como o auto-gerenciamento é definido nos estudos da área de engenharia de software?

Esta questão específica da pesquisa visa apoiar a problemática apontada pela questão central Q1, onde é questionado "o que se sabe sobre o auto-gerenciamento de equipes no desenvolvimento de software". Aqui, o objetivo foi identificar as principais características que as pesquisas apontam como constituintes da definição de equipes de software auto-gerenciadas.

Após a análise e extração dos dados relevantes dos 43 estudos selecionados, 23 características utilizadas para definir equipes de software auto-gerenciadas foram identificadas. A Tabela VI apresenta o mapeamento das características identificadas e os estudos que as indicaram, distribuídas por ordem de importância/frequência. Como foi comum encontrar mais de uma característica utilizada para definir o auto-gerenciamento de equipes no mesmo estudo, este pode estar relacionado a mais de uma característica. Isso explica o percentual total dessas características ser maior do que 100%.

Tabela VI. Mapeamento das Evidências por Características encontradas (Q1.1)

Importância	Características Identificadas	Referências – SM (Self-Management)	Quantidade (%)
1 º	Autoridade Compartilhada de Decisão	[SM016] [SM023] [SM034] [SM040] [SM043] [SM060] [SM121] [SM124] [SM125] [SM129] [SM142] [SM185] [SM260] [SM262] [SM280] [SM369] [SM378] [SM382] [SM392] [SM400] [SM426] [SM428] [SM493] [SM506] [SM511] [SM528] [SM534] [SM541] [SM543] [SM544] [SM573]	72% (31/43)

2 º	Gerenciam sua carga de trabalho	[SM034] [SM043] [SM060] [SM119] [SM121] [SM124] [SM125] [SM129] [SM142] [SM206] [SM260] [SM262] [SM280] [SM378] [SM382] [SM400] [SM426] [SM430] [SM506] [SM528] [SM534] [SM541] [SM543] [SM544] [SM549] [SM573]	60% (26/43)
3 º	Responsabilidade Compartilhada	[SM016] [SM034] [SM043] [SM105] [SM125] [SM129] [SM140] [SM142] [SM185] [SM206] [SM260] [SM378] [SM392] [SM400] [SM426] [SM428] [SM493] [SM541] [SM543] [SM544] [SM573]	51% (22/43)
4 º	Autonomia	[SM040] [SM103] [SM121] [SM124] [SM125] [SM129] [SM142] [SM260] [SM280] [SM329] [SM378] [SM382] [SM400] [SM428] [SM493] [SM543] [SM544] [SM549] [SM573]	44% (19/43)
5 º	Redundância de Funções	[SM034] [SM060] [SM105] [SM121] [SM124] [SM129] [SM140] [SM142] [SM260] [SM378] [SM382] [SM400] [SM426] [SM428] [SM543] [SM544] [SM549] [SM573]	41% (18/43)
6 º	Constante Interação	[SM016] [SM034] [SM040] [SM105] [SM121] [SM124] [SM129] [SM142] [SM152] [SM426] [SM506] [SM543] [SM544] [SM573] [SM382] [SM140]	37% (16/43)
7 º	Feedback	[SM034] [SM105] [SM124] [SM142] [SM152] [SM378] [SM392] [SM400] [SM516] [SM528] [SM543] [SM573]	27% (12/43)
8º	Adaptabilidade	[SM016] [SM040] [SM124] [SM152] [SM369] [SM378] [SM426] [SM543] [SM544] [SM400] [SM124]	25% (11/43)
9º	Confiança Mútua	[SM034] [SM124] [SM400] [SM426] [SM493] [SM378], [SM369] [SM206] [SM040] [SM043]	23% (10/43)
	Foco Comum	[SM124] [SM378] [SM382] [SM400] [SM426] [SM493] [SM541] [SM543] [SM544] [SM105	
10º	Aprendizagem Contínua	[SM016] [SM034] [SM043] [SM124] [SM129] [SM142] [SM152] [SM400] [SM544]	20% (9/43)
	Liderança Distribuída	[SM023] [SM034] [SM129] [SM378] [SM382] [SM392] [SM400] [SM426] [SM544]	
11º	Flexibilidade	[SM043] [SM124] [SM378] [SM400] [SM511] [SM516] [SM543]	16% (7/43)
	Auxílio Mútuo	[SM016] [SM040] [SM121] [SM124] [SM129] [SM142] [SM543]	
12º	Propriedade Coletiva	[SM105] [SM140] [SM206] [SM493] [SM573]	11% (5/43)
	Liberdade	[SM124] [SM197] [SM493] [SM528] [SM544]	
13º	Respeito Mútuo	[SM124] [SM206] [SM516] [SM573]	9% (4/43)
14 º	Interdependência Mútua	[SM197] [SM280] [SM528]	6% (3/43)

	Auto-transcendência	[SM124] [SM260] [SM378]	
	Informações Compartilhadas	[SM016] [SM034] [SM573]	
16º	Estabelecem suas Metas	[SM124]	2% (1/43)
	Constante Auto-avaliação	[SM124]	
	Hierarquia Reduzida (Horizontal)	[SM280]	

Fonte: Elaboração Própria, a partir dos dados do Mapeamento Sistemático

Devido à grande quantidade de características encontradas e também de evidências que as apontam, apenas as características que obtiveram maior freqüência serão melhor descritas. Assim, serão apresentadas cinco transcrições extraídas dos estudos para descrever as cinco principais características: Autoridade Compartilhada de Decisão (72%), Gerenciamento da própria carga de trabalho (60%), Responsabilidade Compartilhada (51%), Autonomia (46%), Redundância de Funções (41%) e Constante Interação (32%).

Autoridade Compartilhada de Decisão

Dentre os estudos incluídos, 72% (31/43) deles consideram a autoridade compartilhada de decisão como uma das características que podem ser utilizadas para definir equipes de software auto-gerenciadas. Esses estudos afirmam que o auto-gerenciamento leva o processo de tomada de decisão ao nível de problemas operacionais. Ou seja, os indivíduos de uma equipe de software auto-gerenciada devem ter autoridade compartilhada para tomar decisões, ao invés de possuir uma estrutura de decisão centralizada, onde somente um indivíduo toma as decisões, ou descentralizada, onde cada indivíduo toma suas decisões individualmente. Segundo esses estudos, isso é uma das vantagens do auto-gerenciamento, se comparado ao gerenciamento tradicional, visto que aumenta a velocidade e precisão da resolução de problemas.

Seguem cinco das transcrições extraídas a partir da análise dos estudos incluídos nesta pesquisa que apontam para a autoridade compartilhada de decisão como característica de equipes de software auto-gerenciadas:

> *"(…) decision making are distributed in a team to promote self-organization."* [SM142]
>
> *"(…) self-management brings decision making authority to the level of operational problems, which increases the speed and accuracy of problem solving."* [SM378]
>
> *"(…) the degree to which all team members jointly share decision authority (…)"* [SM121]

"(…) it brings decision-making authority to the level of operational problems and uncertainties. (…) 'The team is accorded full authority to do whatever it decides is necessary to achieve the goal' (…) team members should share the authority to make decisions, rather than having: (a) a centralized decision structure in which one person (e.g. the team leader) makes all the decisions or (b) a decentralized decision structure in which all team members make decisions regarding their work individually and independently of other team members (…)." [SM034]

"Self-managing teams offer potential advantages over traditionally managed teams because they bring decision-making authority to the level of operational problems and uncertainties. (…)They typically share decision authority jointly, rather than having a centralized decision structure where one person makes all the decisions or a decentralized decision structure where team members make independent decisions. (…) Teams must be able to change the operating norms and rules within the team, as well as in the wider environment, to become and remain self-managed." [SM382]

Gerenciam sua carga de trabalho

O gerenciamento da própria carga de trabalho foi uma das características apontadas por 60% (26/43) dos estudos incluídos na pesquisa. De acordo com esses estudos, equipes auto-gerenciadas são responsáveis por gerenciar sua própria carga de trabalho, monitoram seus próprios processos e execução de tarefas. Os aspectos no qual as equipes gerenciam, estão ligados a planejamento, calendário e atribuição de tarefas. Ou seja, os indivíduos podem decidir "o que", "como" e "quando" farão suas atividades. Para alguns pesquisadores, isso pode influenciar bastante a efetividade da equipe.

Seguem cinco das transcrições extraídas a partir da análise dos estudos incluídos nesta pesquisa que apontam para o gerenciamento da própria carga de trabalho como característica de equipes de software auto-gerenciadas:

"Self-organizing teams manage their own work and organize around the details of their tasks. Self-organizing teams are composed of 'individuals [that] manage their own workload, shift work among themselves based on need and best fit, and participate in team decision making.' ." [SM124]

"Self-organizing teams manage their own work, and organize around the details of the tasks, and can greatly influence team effectiveness. (…) They manage their own work and organize around the details of their tasks." [SM378]

"Members of self-managed teams are responsible for managing and monitoring their own processes and executing tasks. (…) Self-managing teams are responsible for planning and scheduling their work" [SM382]

"In an organizational context, self-organization is the spontaneous coming together of a group to perform a task (or for some other purpose): the group decides what to do, how and when to do it, and no one outside the group directs those activities explicitly (Mittleton-Kelly 2003)" [SM142]

"Guzzo and Dickson (…) described such a team as a group of employees who (…) are given significant authorithy and responsiblity for many aspects of their work, such as planning, scheduling, assigning tasks to members, and making decisions with economic consequences. (…) In the agile approach, the self-organizing team itself decides how work is coordinated." [SM125]

Responsabilidade Compartilhada

Foram identificados 51% (22/43) estudos que indicaram a responsabilidade compartilhada como uma das características definidoras de equipes de software auto-gerenciadas. Os estudos indicam que, dentre outros aspectos, equipes auto-gerenciadas devem ser responsáveis pela obtenção de recursos, motivação da equipe e alinhamento das pessoas.

Apesar de bastante mencionada, esta característica não foi muito desenvolvida pelos estudos. Isso pode ser percebido nas cinco transcrições apresentadas, como segue:

"Agile teams are often compared to coaches of sports teams — responsible for setting direction, aligning people, obtaining resources, and motivating the teams (…)." [SM378]

"Self-management and team-discipline: shared responsibility for project management" [SM142]

"(…) Research has shown that (…) collective responsibility (is a) key feature of teams adopting XP (…)" [SM493]

"(…) a self-managing, self-organizing community with a culture that emphasized shared responsibility." [SM206] [SM105] [SM573]

"Team self-organization (…) principles: (…) boost performance through group accountability for results and shared responsibility for team effectiveness." [SM043]

Autonomia

Como previsto, a autonomia foi uma das características que mais foram mencionadas nos estudos incluídos, num total de 44% (19/43). Para muitos pesquisadores, a autonomia é um sinônimo do auto-gerenciamento e, por si só, representa o conceito de auto-gerenciamento. Alguns consideram que sem autonomia o auto-gerenciamento é apenas simbólico. Ou seja, se a equipe não possui autonomia, não pode agir como uma equipe auto-gerenciada. Porém, o foco da autonomia é importante. Ao contrário do gerenciamento tradicional, onde existe alta autonomia individual e baixa autonomia a nível da equipe, o auto-gerenciamento requer um nível elevado de autonomia individual e da equipe.

Abaixo são apresentadas cinco das transcrições extraídas a partir da análise dos estudos incluídos nesta pesquisa que apontam para a autonomia como característica de equipes de software auto-gerenciadas:

"Takeuchi and Nonaka describe self-organizing teams as exhibiting autonomy, cross-fertilization, and self-transcendence" [SM124]

"Moe et al.. (…) use one of these aspects — autonomy — to represent the concept of self-organizing teams and explore barriers to selforganization on a single project" [SM378]

"Self-organization is the ability of interconnected autonomous agents of a complex adaptive system to evolve into an organized form without external force." [SM142]

"We use the label 'self-organizing' teams as a synonym for 'autonomous team'. Team autonomy is at the heart of agile software development" [SM125]

"Traditional software teams consist of independently" focused self-managing professionals with high individual but low team autonomy. A challenge with introducing agile software development is that it requires a high level of both individual and team autonomy." [SM280]

"Without autonomy self-organization is symbolic only (…) meaning that if a team has no autonomy, it cannot really act like a self-organizing work unit." [SM129]

Redundância de Funções

Dentre os estudos incluídos nesta pesquisa, 41% (18/43) indicam a redundância de funções ou comportamento de *backup* como uma das características de equipes de software auto-gerenciadas. Os estudos afirmam que a redundância de funções é um importante pré-requisito para o auto-gerenciamento. Para eles, equipes auto-gerenciadas devem ser compostas por indivíduos generalistas, com diferentes especializações, processos de pensamento, padrões de comportamento, habilidades e talentos. Essas diferenças permitem a todos os membros a chance de desenvolves diferentes competências e compartilhá-las com os outros, o que leva a um comportamento de backup, onde os indivíduos podem ser substituídos de acordo com as necessidades. Alguns pesquisadores sugerem que isso pode criar espaço para a inovação. Outros afirmam que a falta desta característica nas equipes de software auto-gerenciadas foi encontrada como a principal barreira ao auto-gerenciamento.

Abaixo são apresentadas cinco das transcrições extraídas a partir da análise dos estudos incluídos nesta pesquisa que apontam para a redundância de funções como característica de equipes de software auto-gerenciadas:

"(…) it is composed of individual members with varying specializations, thought processes, and behavior patterns and (b) these individuals interact amongst themselves leading to better understanding of each other's perspectives. (…). Flexibility to work in multiple technical areas was welcomed by developers because it helped them maintain interest in their work." [SM124]

"(…) In addition, it considers important elements that are required in self-managed teams: (…) functional redundancy and backup behavior (…). (…)In the literature on self-managed teams, backup behavior has been identified as an important prerequisite for self-management (…). In our study, highly specialized skills and a corresponding division of work was the most important barrier to achieving backup and then self-management." [SM034]

"Each team member is able to assume all the roles (…)Total team involvement in all development activities with no separation of functional roles gives everyone a chance to

develop different competences and for these competences to be distributed among the developers, leading to an autonomous team that can work on any aspect of the development." [SM142]

"According to Morgan (…), any system with an ability to self-manage must have a degree of redundancy: a kind of excess capacity that can create room for innovation and development to occur. Members in a team need multiple skills so that they are able to perform (parts of) each other's jobs and substitute each other as circumstances demand." [SM125]

"(…) in the literature on self-organizing teams, we find that redundancy has been identified as an important prerequisite for self-organization (…). In particular, the "redundancy of functions", in which team members acquire multiple skills so that they are able to perform each other's jobs and substitute as the need arises, is essential for creating flexibility. The lack of such redundancy, and consequently, flexibility, is probably one of the most important reasons for several of the problems experienced by the case company." [SM543]

Constante Interação

A importância e frequência da comunicação e interação entre os indivíduos para que se tenha uma equipe auto-gerenciadas foi indicada em 32% (14/43) dos estudos incluídos na pesquisa. Alguns pesquisadores consideram este um componente indispensável ao auto-gerenciamento e afirmam que a constante interação permite que a equipe seja conduzida pelos próprios membros como se não existisse um supervisor. Isso porque a comunicação leva a um melhor entendimento sobre as perspectivas de cada indivíduo.

Seguem cinco das transcrições extraídas a partir da análise dos estudos incluídos nesta pesquisa que apontam para a constante interação entre indivíduos como característica de equipes de software auto-gerenciadas:

"(…) it is composed of individual members with varying specializations, thought processes, and behavior patterns and (b) these individuals interact amongst themselves leading to better understanding of each other's perspectives" [SM124]

"The roles played by individuals in an organization are therefore reshaped in the light of self-organization with emphasis placed on increased autonomy, more interactions with

other individuals and environment, and greater participation, especially in the decision-making process (…). (…) Interactions among team members, in the form of communication and collaboration, are an indispensable component of a self-managing team." [SM142]

"A self-organizing team needs the ability to demonstrate and communicate" [SM016]

"(…) the group interactions lead the team, i.e., the way team members communicate, reflect, and collaborate enables the team to lead itself as if there is no leader" [SM040]

"The main conclusions of the model construct and empirical evidence are that efficient communication (is one of) the foundational building element for effective self-organization." [SM129]

4.2.2.2. Q1.2 Efeitos do auto-gerenciamento em equipes de desenvolvimento de software

Q1.2. Quais são os efeitos do auto-gerenciamento nas equipes de desenvolvimento de software?

A segunda questão específica da pesquisa também visa apoiar a problemática apontada pela questão central Q1, onde é questionado "o que se sabe sobre o auto-gerenciamento de equipes no desenvolvimento de software". Aqui, o objetivo foi identificar os principais efeitos relacionados ao auto-gerenciamento de equipes apontados pelas pesquisas.

Após a análise e extração dos dados relevantes dos 43 estudos selecionados, 25 efeitos relacionados ao auto-gerenciamento foram identificados. A Tabela VII apresenta o mapeamento desses efeitos e os estudos que os apontaram, por ordem de importância/frequência. Identificar mais de um efeito em um mesmo estudo foi fato corriqueiro, o que explica o percentual total desses efeitos maior do que 100%.

Tabela VII. Mapeamento das Evidências por Efeitos relacionados ao auto-gerenciamento encontrados (Q1.2)

Importância	Efeitos do Auto-gerenciamento	Referências	Quantidade(%)
1º	Motivação dos Indivíduos	[SM260] [SM125] [SM400] [SM543]	39% (17/43)

		[SM329] [SM493] [SM541] [SM544] [SM206] [SM023] [SM516] [SM152] [SM369] [SM016] [SM197] [SM124] [SM142]	
2 º	Satisfação dos Indivíduos	[SM129] [SM034] [SM382] [SM125] SM400] [SM329] [SM206] [SM392] [SM142] [SM103] [SM369] [SM573] [SM197]	30% (13/43)
3 º	Produtividade	[SM129] [SM400] [SM543] [SM541] [SM544] [SM534] [SM382] [SM206] [SM430] [SM573] [SM260] [SM197]	27% (12/43)
4 º	Rápida Resolução de Problemas / Adaptação a Mudanças	[SM129] [SM125] [SM400] [SM543] [SM544] [SM124] [SM378] [SM382] [SM369] [SM430] [SM260]	25% (11/43)
5 º	Sensações Positivas para os indivíduos	[SM125] [SM543] [SM103] [SM206] [SM516] [SM152] [SM369] [SM016] [SM105] [SM573] [SM260]	23% (10/43)
	Consciência de Responsabilidade	[SM125] [SM543] [SM541] [SM544] [SM023] [SM124] [SM034] [SM516] [SM185] [SM430]	
6 º	Aumento na Qualidade	[SM129] [SM400] [SM543] [SM541] [SM544] [SM534] [SM382] [SM430]	18% (8/43)
7 º	Efetividade	[SM124] [SM378] [SM034] [SM125] [SM262] [SM544]	13% (6/43)
8 º	Comportamento de Colaboração	[SM125] [SM543] [SM541] [SM544] [SM400]	11% (5/43)
9 º	Eficácia	[SM206] [SM573] [SM260] [SM511]	9% (4/43)
	Aumento da Criatividade	[SM543] [SM541] [SM544] [SM392]	
	Redução do Estresse	[SM103] [SM206] [SM105] [SM573]	
	Menor Absenteísmo	[SM129] [SM034] [SM382] [SM103]	
	Maior vínculo emocional com a organização	[SM125] [SM543] [SM541] [SM544]	
	Inovação	[SM129] [SM400] [SM543] [SM034]	
10º	Confiança na Gestão	[SM129] [SM400] [SM206]	6% (3/43)
	Desempenho	[SM129] [SM400] [SM430]	
	Aumento da Coesão da Equipe	[SM400] [SM516] [SM121]	
	Menor Rotatividade (Turn over)	[SM129] [SM034] [SM382]	
11 º	Satisfação do Cliente	[SM129] [SM206]	4% (2/43)
	Reduz Custos	[SM382] [SM016]	
12 º	Maior Dedicação	[SM034]	2% (1/43)
	Maior Adaptabilidade dos Indivíduos	[SM400]	
	Melhor Planejamento	[SM430]	
	Melhoria na Comunicação	[SM260]	

Fonte: Elaboração Própria, a partir dos dados do Mapeamento Sistemático

Devido à grande quantidade de efeitos relacionados ao auto-gerenciamento encontrados, apenas os efeitos que obtiveram maior freqüência serão melhor descritos. Assim, serão apresentadas cinco transcrições extraídas dos estudos para descrever os cinco principais efeitos identificados: Motivação dos Indivíduos (34%), Satisfação dos Indivíduos (30%), Produtividade (27%), Sensações Positivas (25%) e a Rápida resolução de problemas (25%).

Motivação dos Indivíduos

Dentre os estudos incluídos na pesquisa, 39% (17/43) afirmam que o conceito de SMWTs leva os indivíduos de equipes de software a serem motivados. Porém, dentre esses, 2% (1/43) afirma que o auto-gerenciamento apenas manteve a motivação dos indivíduos. Outros 2% (1/43) afirma que, comparando ao uso do gerenciamento tradicional, o auto-gerenciamento diminuiu a desmotivação dos indivíduos. O restante 34% afirma que o auto-gerenciamento aumenta a motivação dos indivíduos. Estes estudos apontam para alguns fatores do auto-gerenciamento considerados significantes para o aumento da motivação, fatores tais como: autonomia, variedade de habilidades, significância de tarefas, identificação com as tarefas, habilidade para completar as tarefas e *feedback*. Ainda de acordo com os estudos encontrados, a participação da equipe no planejamento motiva os membros da equipe a trabalharem juntos e efetivamente.

Abaixo são apresentadas cinco das transcrições de trabalhos que se referem à motivação dos funcionários como sendo positiva nas equipes de software auto- gerenciadas:

> *" (…)Thus what can be seen as the major benefit of self-regulating teams – their ability to motivate team members to work together in a flexible manner towards collective goals – can be waylaid by over application of social controls (…)" [SM516]*

> *"Research in workplace psychology indicates that factors like autonomy, variety, significance, feedback, and ability to complete a whole task are significant factors to ensure motivation among workers. (...) In an ethnographic study of an XP team by Sharp and Robinson (…), (…) agile methods in fact contribute to create a work environment where the developers are highly motivated (…)" [SM329]*

> *"Leader delegation will then improve the team's sense of self-worth and motivate the team to work more effectively. An empirical study found that the autonomy of virtual team members in determining work objectives and methods improved the intrinsic motivation of the team […]. Picolli and Ives […] found that student virtual teams were more motivated and satisfied with less behavior control. (…) Also a team's decision*

acceptance is greatest when the decision is made by the group […]. Therefore, delegation to a competent virtual team would increase the team's motivation (…)" [SM369]

"(…)Use self-managed teams approach. The collective management motivates and it brings more commitment with project goals […] […]; Involve the whole team in the planning of each cycle. It provides visibility project stages and its impact on delaying each point […] […]. Participation in planning brings more commitment and motivation." [SM023]

"Hackman and Oldham […] indicate that employees will be motivated if they perceive their jobs to be characterized by high skill variety, task significance, task identity, autonomy, and feedback." [SM197]

Satisfação dos Indivíduos

Dos 43 estudos incluídos na pesquisa, 13 (30,23%) indicaram o aumento da satisfação dos indivíduos como um possível efeito do auto-gerenciamento de equipes de software. Alguns desses estudos enfatizaram *feedback* e outros a significância de tarefas e autonomia como principais fatores relacionados positivamente à satisfação dos indivíduos no trabalho. Porém, outros fatores também são considerados significantes para este aumento da satisfação, tais como: variedade de habilidades, identificação com as tarefas, habilidade de completar toda a tarefa e a variedade de tarefas. De acordo com esses estudos, indivíduos que percebem que possuem e maior nível de autonomia sobre seu próprio trabalho sentem-se mais satisfeitos com seu trabalho e com o produto.

Seguem cinco das transcrições extraídas dos estudos incluídos que constataram o aumento da satisfação dos indivíduos como um efeito relacionado ao auto-gerenciamento em equipes de software:

"One thing the team realizes is the importance of feedback on the positive aspects of the previous day, which can provide them with satisfaction and help keep them motivated. Sometimes the team also challenges the practices that work well, as the coach explains (…)" [SM142]

"Based on separate tests of the two facets of autonomy/control, we found that both types of autonomy/control had a direct effect that was significant ($p < 0.05$), based on one-tailed tests. Users who perceived that they had control over others in their work unit

and those who perceived a higher level of autonomy over their own work reported lower job dissatisfaction (or higher satisfaction). These results are consistent with what Van der Doef and Maes […] label the "strain hypothesis" of Karasek's research model, since they show that higher autonomy and control has a positive effect on work outcomes (reducing the level of mental strain, job dissatisfaction, and absenteeism), apart from any moderating effect. We examine the moderating effects of high autonomy/control in the next two hypotheses (…)"[SM103]

"The results of their study showed that the developers viewed XP practices favorably and indicated that developers who use XP practices are more comfortable with their job environment and more satisfied with their jobs than developers that do not use XP practices. (…)However, we also found that the role of on-site customer can be stressful and cannot be sustained for a long period. Developers are mostly satisfied with agile methods. Companies that use XP have reported that their employees are more satisfied with their job and that they are more satisfied with the product." [SM206]

"Five task characteristics (skill variety. task variety, task significance, autonomy, and feedback) are related to computer programmer productivity and job satisfaction. (…)In terms of the task characteristics, (…) two variables (taks significance and autonomy) were positively related to the computer programmer's level of job satisfaction." [SM197]

"Bivariate analyses indicated that managerial control, team-member control, and self-control were highly correlated with general satisfaction. Additional analyses indicated that self-control had strong positive effects on both intrinsic and extrinsic satisfaction, while, managerial control had a strong positive effect on extrinsic satisfaction. (…)In particular, IS researchers have found that each of the five core job dimensions--skill variety, task identity, task significance, autonomy, and feedback--has a positive effect on job satisfaction (Cheney, 1984; Couger et al.., 1979; Couger & Zawacki, 1981; Goldstein, 1989), and that the role stress variables--role conflict and role ambiguity--increase job related tension which, in turn, negatively affects several employee outcomes, including job satisfaction (Baroudi, 1985; Baroudi & Ginzberg, 1986; Bostrom, 1981; Goldstein & Rockart, 1984; Guimaraes & Igbaria, 1992)." [SM392]

Produtividade

Foram identificados 27% (12/43) dos estudos que indicam o aumento da produtividade como um efeito relacionado ao auto-gerenciamento de equipes de software. A produtividade relatada nos estudos é, por vezes, associada à efetividade e ao desempenho obtidos a partir de SMWTs.

Por este motivo, os estudos costumam apenas mencionar o aumento de produtividade juntamente a outros efeitos, também encontrados nesta pesquisa. Isso pode ser observado nas cinco transcrições apresentadas abaixo, que foram extraídas dos estudos incluídos que indicam o aumento da produtividade como um efeito relacionado ao auto-gerenciamento em equipes de software:

> *"Crossfunctional, and self-organizing teams have been found to have high productivity […] and increase the speed and accuracy of problem solving […]. However, research indicates that the effects of such teams are highly situationally dependent and that the effects depend on factors such as the nature of the workforce and the nature of the organization […]." [SM400]*

> *"SCRUM-PSP focuses on self-directed team, by which high performance (short iteration time, high productivity, high product quality, quick response to requirement changes and predictive delivery time) can be expected." [SM023]*

> *"In general, the related literature suggests that selforganizing teams bring many advantages for the organizations. One of the key benefits is performance effectiveness which means, for example, better productivity, lower response time, better quality and customer satisfaction, and more innovations […]." [SM129]*

> *"Regarding the relationship between project productivity and Scrum, Sutherland et al.. […] found that the use of Scrum in a company with CMMI level 5 almost doubled the productivity compared with projects using traditional processes. Furthermore, Benefield […] claimed that Yahoo had experienced 39% productivity gains after the introduction of Scrum in more than 150 teams worldwide. Richard et al.. […] concluded that much of the perceived increase in productivity from using Scrum at 3M came from the focus on delivering functional software. (…) Although several previous studies have concluded that agile process improve software quality and productivity, most studies, such as […], relied solely on quantitative data. Therefore, they could not figure out which Scrum practices impact software quality and productivity to guide industrial practitioners to adapt such practices effectively." [SM534]*

> *"(…) five task characteristics (skill variety. task variety, task significance, autonomy, and feedback) are related to computer programmer productivity and job satisfaction. (…) In*

terms of the task characteristics, only one variable (autonomy) was positively related to productivity (…).” [SM197]

Rápida Resolução de Problemas

Dentre os estudos incluídos, 25% (11/43) apontam para a rápida resolução de problemas como um efeito positivo relacionado ao uso do auto-gerenciamento em equipes de software. Segundo esses estudos, o fato de o auto-gerenciamento levar a autoridade de decisão ao nível de problemas operacionais permite que equipes auto-geridas competentes possam tomar as ações necessárias sem que tenham que esperar por decisões tomadas por um líder. Isso aumenta a velocidade e a precisão da resolução de problemas e, portanto, é indicado por estudos como um fator que pode diretamente influenciar na efetividade da equipe.

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos que fazem referência à rápida resolução de problemas como um efeito relacionado ao auto-gerenciamento em equipes de software:

“(…) as self-management brings decision making authority to the level of operational problems, which increases the speed and accuracy of problem solving.” [SM124] [SM378] [SM382]

“Self-management can also directly influence team effectiveness since it brings decision-making authority to the level of operational problems and uncertainties and, thus, increases the speed and accuracy of problem solving […].” [SM125] [SM543]

“Crossfunctional, and self-organizing teams have been found to (…) increase the speed and accuracy of problem solving […]. However, research indicates that the effects of such teams are highly situationally dependent and that the effects depend on factors such as the nature of the workforce and the nature of the organization […].” [SM400]

“Delegation allows the competent team to utilize their capabilities to adapt to the immediate opportunities and changes without waiting for decisions to be made by the distant leader. “ [SM369]

“Self-organizing teams can react to problems quickly since the decision-making is close to the problem […]. Instead of waiting for a manager’s approval, the team has the authority to take necessary actions by itself […].” [SM129]

Sensações Positivas

Algumas sensações positivas e redução de aspectos negativos no trabalho e fora dele foram indicadas por 23% (10/43) dos estudos incluídos na pesquisa. Esses estudos sugerem que aspectos do auto-gerenciamento das equipes estão relacionados a indivíduos julgarem o trabalho como sendo excitante e gratificante. Além dos aspectos trazidos pelo auto-gerenciamento das equipes, esses sentimentos positivos encontrados se referem ao reconhecimento das habilidades dos indivíduos, ao aumento de sua auto-estima e ao orgulho de pertencer à equipe do projeto. Porém, a alta demanda de trabalho e baixa autonomia dos indivíduos podem levar a resultados bastante negativos, tais como: o aumento do consumo de remédios, depressão, nervosismo, ansiedade e distúrbios do sono, como insônia.

Abaixo são apresentadas cinco das transcrições de trabalhos que se referem a sensações positivas como um efeito relacionado ao auto-gerenciamento das equipes de software:

> *"Members of cohesive agile teams noted positive feelings in association with recognition of their abilities within the team, and pride in project group membership" [SM516]*
>
> *"The product manager and self-motivated developers found the agile adoption exciting and rewarding." [SM152] [SM016]*
>
> *"Karasek […] examined two types of mental strain symptoms in his initial study: work exhaustion and depression. His findings showed that it was "primarily workers with jobs simultaneously low in job decision latitude [autonomy/control] and high in job demands who report exhaustion after work, trouble awakening in the morning, depression, nervousness, anxiety, and insomnia or disturbed sleep" […]. (…)In subsequent work, Karasek also examined several other consequences: absenteeism, pill consumption, and job dissatisfaction. The later findings were consistent with his earlier work: employees with high job demands and low autonomy/control reported higher levels of pill consumption, more sick days, and more job dissatisfaction […] (…)The latter result implies that employees who experience higher autonomy report fewer symptoms of depression although this result was only marginally significant, using a one-tailed test (p=0.057)." [SM103]*
>
> *"Leader delegation will then improve the team's sense of self-worth and motivate the team to work more effectively. (…)." [SM369]*
>
> *"The team atmosphere that we observed was one of calm, competence and confidence,*

embodied by a pervasive and harmonious rhythm. This rhythm operated at a range of levels and we'll concentrate on two that emerge from the data: one was a daily rhythm and one was a rhythm oriented around the three-week iterations Whilst the atmosphere was relaxed, much was happening (...) and there was no sense of frustration, such as might be associated with missed targets in a rigid schedule." [SM573]

4.2.2.3. Q1.3 Fatores que afetam o auto-gerenciamento em equipes de software

Q1.3. Quais são os fatores técnicos, humanos e organizacionais que podem afetar o trabalho de equipes de software auto-gerenciadas?

Esta terceira questão específica da pesquisa objetiva o conhecimento dos principais fatores indicados nos 43 estudos incluídos na pesquisa que podem afetar positivamente ou negativamente o trabalho das equipes de software auto-gerenciadas. Esta questão específica também visa apoiar o entendimento da questão central Q1, onde é questionado "o que se sabe sobre o auto-gerenciamento de equipes no desenvolvimento de software".

Assim como aconteceu nas questões específicas Q.1.1 e Q1.2, foi comum identificar mais de um fator influente em um mesmo estudo. Isso explica o percentual total de cada grupo de fatores identificados ser maior do que 100%. Além disso, também foi comum um mesmo fator estar associado a mais de um grupo de fatores. Ou seja, um mesmo fator pode estar associado a Fator Técnico, Humano e/ou Organizacional.

Os grupos de fatores (Técnicos, Humanos e Organizacionais) serão apresentados separadamente nesta seção, onde serão melhor detalhados os cinco principais fatores, por ordem de freqüência, a partir de cinco transcrições extraídas dos estudos. A não apresentação de todos os fatores se deve ao fato de ter sido identificada uma grande quantidade de evidências e fatores que podem afetar o auto-gerenciamento em equipes de software.

FATORES TÉCNICOS

Fator técnico foi entendido nesta pesquisa como tudo aquilo que se refere ao procedimento de execução de algum tipo de método utilizado pelas equipes e que é mencionado nos estudos como um fator importante para os resultados obtidos do auto-gerenciamento ou que pode afetar positivamente ou negativamente o trabalho de equipes de software auto-gerenciadas.

Após a análise e a extração dos dados relevantes dos 43 estudos incluídos na pesquisa, 22 fatores técnicos que podem afetar o auto-gerenciamento em equipes de software foram constatados. A Tabela VIII apresenta o mapeamento desses fatores e as referências que apontam para cada um desses fatores por ordem de importância/freqüência.

Dentre os fatores identificados aqui, muitos se referem a princípios das práticas ágeis. Por este motivo, alguns fatores encontrados podem parecer estranhos terem sido considerados fatores técnicos, mas o modo como eles são tratados permite entender que os princípios e práticas atribuídos a determinadas metodologias ágeis podem afetar o trabalho das equipes, positiva ou negativamente.

Tabela VIII. Mapeamento das Evidências por Fatores Técnicos encontrados (Q1.3)

Importância	Fatores Técnicos	Referências – SM (Self-Management)	Quantidade (%)
1º	Reuniões	[SM573] [SM543] [SM541] [SM129] [SM034] [SM142] [SM428] [SM206] [SM023] [SM534] [SM426] [SM260] [SM382] [SM430] [SM262] [SM124] [SM378] [SM185]	41% (18/43)
2 º	Processo Iterativo e Incremental	[SM543] [SM428] [SM034] [SM378] [SM124] [SM426] [SM060] [SM023] [SM573] [SM260] [SM129] [SM430] [SM541] [SM534] [SM493] [SM206]	37% (16/43)
3 º	Programação em Pares	[SM129] [SM124] [SM142] [SM329] [SM105] [SM573] [SM260]	16% (7/43)
4 º	Processo Visível a Todos	[SM034] [SM124] [SM428] [SM516] [SM040] [SM260]	13% (6/43)
5 º	Tamanho das Equipes	[SM034] [SM400] [SM544] [SM426] [SM541]	11% (5/43)
6 º	Direcionado pelo Valor de Negócio	[SM124] [SM378] [SM016] [SM105]	9% (4/43)
	Papel de Representante do Cliente	[SM124] [SM129] [SM428] [SM430]	
	Pouca Documentação	[SM105] [SM573] [SM260] [SM140]	
7 º	Foco na Comunicação	[SM016] [SM129] [SM140]	6% (3/43)
	Foco em Pessoas	[SM543] [SM426] [SM105]	
	Papel de Facilitador/Treinador	[SM378] [SM544] [SM060]	

8 º	Equipe Distribuída/Contígua	[SM369] [SM506]	4% (2/43)
	Treinamento da Equipe	[SM544] [SM129]	
	Foco na Criatividade	[SM543] [SM426]	
	Cultura de Compreensão Compartilhada	[SM034] [SM573]	
	Liderança Compartilhada	[SM129] [SM544]	
	Processo Flexível	[SM543] [SM541]	
9 º	Comunicação Face-a-face	[SM105]	2% (1/43)
	Tarefas Pequenas	[SM023]	
	Processo Interativo	[SM382]	
	Planejamento de Tarefas	[SM260]	
	Definição de "Feito"	[SM382]	

Fonte: Elaboração Própria, a partir dos dados do Mapeamento Sistemático

Os fatores que tiveram maior freqüência nos estudos serão melhor detalhados abaixo, a partir de cinco transcrições para cada um deles. Os principais fatores técnicos identificados foram, respectivamente: Reuniões (41%), Processo Iterativo e Incremental (37%), Programação em Pares (16%), Processo Visível a Todos (13%) e Tamanho das Equipes (11%).

Reuniões do Processo

Os diversos tipos de reuniões que comumente estão presentes na rotina de equipes auto-gerenciadas, principalmente quando seguem alguma metodologia ágil, constituem o fator técnico mais citado nos estudos, com 41% (18/43) dentre os incluídos. De acordo com alguns estudos, as reuniões do processo podem criar uma cultura de antecipação nos desenvolvedores, devido ao fato de incentivarem valores produtivos, como: a sensação de urgência, a partilha de conhecimentos, comunicações densas e a honestidade entre os indivíduos ([SM260]). Alguns estudos consideram que as reuniões do processo apóiam a orientação da equipe e a consciência dos membros da equipe sobre suas tarefas, além de ser uma ferramenta efetiva para verificar o aprendizado da equipe e o mecanismo mais importante para facilitar a estrutura de comunicação da equipe ([SM129], [SM124], [SM034], [SM142]). O rápido feedback e o conhecimento compartilhado atribuídos às reuniões do processo fazem com que melhore o entendimento do sistema a ser desenvolvido e antecipe o aprendizado a partir de erros por parte dos membros da equipe ([SM142], [SM534], [SM129]). Além disso, o feedback proporcionado pelas reuniões pode levar os indivíduos a manter a satisfação e motivação ([SM142]).

Porém, foi encontrado que as muitas reuniões do processo, se não cuidadas, também podem levar a equipe a perder a serenidade, além de baixar o compromisso compartilhado. Isso se deve ao fato de algumas pessoas considerarem as reuniões como uma "perda de tempo" ([SM573], [SM382], [SM543]).

A reunião diária, onde deve ser discutido o progresso da equipe, os planos e os potenciais problemas diariamente em um tempo máximo de 15 minutos é considerada o mecanismo de reunião mais importante ([SM426], [SM142], [SM428]). Porém, existem outras, que também são consideradas importantes, como a reunião de retrospectiva, que acontece ao final de cada iteração para se discutir o que se passou durante a iteração passada, reuniões de planejamento, reuniões de revisão e reuniões de estimativa de tarefas ([SM426], [SM129], [SM426], [SM185], [SM378]).

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para o processo iterativo e incremental como um fator que deve ser cuidadosamente coordenado, pois pode afetar o trabalho de equipes de software auto-gerenciadas:

> *"Regular weekly and daily meetings (planning game, steering, and feedback sessions) also facilitate and structure team communication. (…)One thing the team realizes is the importance of feedback (daily-meetings) on the positive aspects of the previous day, which can provide them with satisfaction and help keep them motivated. Sometimes the team also challenges the practices that work well, as the coach explains" [SM142]*

> *"A daily Scrum meeting is a short, stand-up, typically 15-minute meeting, during which the team members explain to each other what they accomplished since the last meeting, what they are going to do by the next meeting and what obstacles are on the way. (...) Greene […] reports that in one software development team at Intel they had to de-emphasize explicitly the "what did you do yesterday?" daily question, as people felt it was a general status meeting question and tended to lose too much time on it." [SM428]*

> *"During a sprint, the team holds short daily Scrum meetings to discuss progress, plans and potential problems. The daily scrum is important for understanding team members' tasks. The daily Scrum is the most important mechanism, but also the retrospective, planning meeting and review meetings provide feedback loops However, the daily meetings increased the overall communication Daily scrums address impediments, to discover other team-members problems." [SM426]*

"Scrum meetings help also in the creation of an "anticipating" culture […] because they encourage "productive values":

• They increase the overall sense of urgency.

• They promote the sharing of knowledge.

• They encourage dense communications.

• They facilitate honesty among developers since everyone has to give a daily status.

Daily Scrum of Scrums became the major Scrum meeting" [SM260]

"It seemed that some did not feel 100% committed to the project meetings. One reason was that some of the developers perceived these meetings as a waste of time." [SM543]

Processo Iterativo e Incremental

Dentre os estudos incluídos, 37% (16/43) apontaram para o processo iterativo e incremental, considerado princípio comum em equipes auto-gerenciadas ou ágeis, como um fator que pode afetar o auto-gerenciamento em equipes de software. De acordo com esses estudos, é nesse processo de desenvolvimento com várias iterações e incrementos que as equipes auto-gerenciadas ajustam as mudanças de requisitos dos clientes de acordo com as necessidades e tomam importantes decisões em conjunto ([SM124], [SM378], [SM382]). Esse processo é chamado por vezes como "Sprints", como em métodos ágeis, e devem possuir um prazo curto e pré-estabelecido, que deve iniciar com um planejamento e finalizar com uma revisão do que foi feito durante a iteração e deve durar de uma a quatro semanas ([SM543] [SM206] [SM534] [SM426], [SM426], [SM260], [SM034]). Esse processo é visto como *loops de feedback* que permitem que os desenvolvedores estejam sempre se adaptando às necessidades dos clientes e que apóiam o aprendizado da equipe ([SM543] [SM206] [SM534] [SM426] [SM060], [SM129]). Além disso, de acordo com estudos, quando bem executado, este processo pode dobrar a produtividade e entregar projetos no prazo, dentro do orçamento e ainda com as funcionalidades precisamente orientadas para as exigências do usuário final ([SM260]).

Porém, os estudos indicam que este processo deve ser cuidadosamente coordenado, pois ele pode criar uma perda de tempo quando a equipe se reorganiza entre as iterações e requerer

revisão periódica e ajustes pontuais devido à entrega de várias versões de software simultâneas ([SM260]). Além disso, é considerado também que quando ocorrem mudanças urgentes durante uma iteração a reação da equipe em um período de tempo aceitável é mais difícil ([SM573]).

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para o processo iterativo e incremental como um fator que deve ser cuidadosamente coordenado, pois pode afetar o trabalho de equipes de software auto-gerenciadas:

> *"Scrum focuses on project management in situations where it is difficult to plan ahead, with mechanisms for "empirical process control"; where feedback loops is the core element […]. Software is developed by a selforganizing team in increments (called "sprints"), starting with planning and ending with a review." [SM543] [SM206] [SM534] [SM426]*
>
> *"Agile software development promotes the development and delivery of completed work in short time-boxed iterations (which are also called Sprints) so that teams can obtain frequent feedback from stakeholders and adapt as needed. Here, the term time-box refers to a period of time in which to accomplish a task." [SM060]*
>
> *"Rising and Janoff […] described Scrum as a development process for small teams, which includes a series of short development phases, "sprints", which typically lasts from one to four weeks. (…)Scrum extends incremental software development to what is called "empirical process control"; where feedback loops is the core element Scrum is designed to adapt to change, by frequent feedback loops and replanning." [SM426]*
>
> *" (…) It was reported that generally customers are supportive of the XP approach because they had fast turnaround to issues. However when urgent changes arrived in the middle of an Iteration it became harder to react in an acceptable amount of time." [SM573]*
>
> *"In a Type A Scrum, all development occurs in an increment within the time box of Scrum iteration called a Sprint. A side effect of this approach is downtime between iterations when reorganizing for the next Sprint. Nevertheless, well executed Sprints can double productivity and repeatedly deliver projects on time, within budget, with functionality precisely targeted to end-user demands. Its downside is that it creates a loss of time*

between Sprints when the team is reorganizing for the next Sprint. Managing multiple simultaneous releases of software requires regular review and fine tuning of release schedules. Since every Sprint resulted in a production release of software, Sprints must be carefully coordinated." [SM260]

Programação em Pares

O mecanismo de programação em pares foi mencionada por 16% (7/43) dos estudos como sendo um fator que pode afetar resultados do auto-gerenciamento de equipes de software. Porém, esta prática só foi considerada como um fator que afeta positivamente os resultados do auto-gerenciamento. A prática da programação em pares consiste em um membro da equipe se juntar a outro que tenha mais experiência para adquirir novas habilidades, onde os indivíduos compartilham o mesmo computador, um utiliza o teclado, como um "*driver*", e outro o mouse, como um "navegador" ([SM142], [SM105]). Para melhoria do processo, é indicado que os pares sejam rotativos, de modo que seja incomum que um par trabalhe junto por mais de um dia. Isso permite que todos da equipe trabalhem juntos ao menos uma vez na iteração e que o cada membro da equipe trabalhe em uma grande parte do código em cada iteração ([SM573]).

De acordo com os estudos encontrados, a prática da programação em pares deve ser fortemente incentivada por gerentes, visto que faz com que a comunicação entre os indivíduos seja inevitável, é uma forma particular de colaboração e ajuda a melhorar a compreensão compartilhada e reforçar o propósito compartilhado ([SM329], [SM124], [SM142], [SM573], [SM129]).

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para a programação em pares como um fator que pode afetar positivamente o trabalho de equipes de software auto-gerenciadas:

"Pair programming, a particular form of collaboration, is a standard XP practice where developers work in pairs on every task […]." [SM124]

"Communication and collaboration are at the heart of agile software development. Another aspect is C&C inside the team. This is supported by the concepts of pair programming and open workspace in XP, and that of co-location in Crystal Clear […], […]. (…)Colocation enables the team to talk face-to-face anytime and pair programming

makes the communication between two developers inevitable. Also use of information radiators reveals project status to the team all the time." [SM129]

"Generally, the developers choose tasks they feel confident in completing, but they also pick up tasks that they are not so good at, and then pair up with a more experienced member to acquire new skills. (…)Pair programming is the main collaboration mechanism in Pongo. During development time, the developers always work in pairs on tasks. They physically sit together and share one desktop, one using the keyboard, as "driver," the other using the mouse, as "navigator." [SM142]

"As in other jobs, there are of course some limitations to autonomy for these developers. There are, for instance, architectural guidelines for the implementations, or the pair programming practice which is strongly encouraged by management. On the other hand, we see that confidence in people`s abilities made leaders trust them with advanced task" [SM329]

"Boundaries between pairs in pair programming also existed but were fluid and could be crossed without any great negotiation as peripheral awareness led to wider discussion as necessary. (…) Pairs would rotate so that it was unusual for a pair to stay together for longer than one day. This meant that everyone paired with everyone else at least once during an iteration. It also meant that everyone worked on a large portion of the code base in each iteration. This rotation helped to enhance the shared understanding and reinforce the shared purpose. Everyone in the development team pair programmed, including the graphic designer and the infrastructure person. " [SM573]

Processo Visível a Todos

O princípio de estabelecer um processo que seja visível e transparente a todos os envolvidos é citado como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas em 13% (6/43) dos estudos incluídos. De acordo com os estudos, o processo ser visível e transparente envolve a compreensão compartilhada do projeto e deve se estender por seus objetivos, planos, riscos, *status*, responsabilidades individuais e requisitos do sistema ([SM516], [SM040], [SM124]). Esse processo faz com que as informações de um projeto sejam acessíveis a qualquer pessoa nele envolvida, além de permitir que cada membro da equipe seja plenamente responsável e envolvido no processo de desenvolvimento ([SM040]). Com isso, ajudando a

equipe a coordenar os esforços e a cumprir os compromissos ou fazer ajustes, de acordo com a necessidade ([SM428]).

Porém, foi encontrado que algumas pessoas podem ficar entediadas com a necessidade de seu processo de monitoramento, como ter que alterar o status de cada tarefa sempre que ela for concluída, entre outros ([SM428]).

Alguns dos estudos sugerem práticas para tornar o processo visível e transparente, tais como: (a) Story Board, que consiste em um quadro afixado a uma parede onde são escritas as estórias que deverão ser realizadas durante a iteração, as tarefas que as constituem e o estado em que essas tarefas se encontram, além da prática do (b) Burndown chart, um gráfico que indica a proporção "tempo" *versus* "tarefas" que já foram feitas e que ainda deverão ser feitas ([SM124], [SM428]). Essas práticas são bastante comuns em metodologias ágeis, que devem trabalhar com equipes auto-gerenciadas.

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para o processo visível e transparente como um fator que pode afetar positivamente ou negativamente o trabalho de equipes de software auto-gerenciadas:

> *"Project goals, system requirements, project plans, project risks, individual responsibilities, and project status must be visible and understood by all parties involved […]." [SM034]*
>
> *"The story board - a physical paper chart posted on the wall; containing all stories, tasks, their states, and names of individuals working on them - provides high visibility and transparency about task assignments […] such that "when someone looks to the board they can see who is working on which task" (P9)." [SM124]*
>
> *"Burndown charts and tracking work in-sprint are supposed to help the team coordinate efforts and meet or adjust to sprint commitments, when needed. Unfortunately the results of the careful tracking take time to become visible; some people become bored with the need to play with the childish cards and tracking process." [SM428]*
>
> *"While agile practices instantiate team activity to be observable or transparent" [SM516]*

"This idea is expressed, among other ways, by the transparency of agile software development processes that makes information accessible to anyone and enables each team member to be fully accountable and involved in the development process." [SM040]

Tamanho das Equipes

Dentre os estudos incluídos na pesquisa, 11% (5/43) apontam para o tamanho da equipe como sendo um fator que pode afetar o trabalho de equipes de software auto-gerenciadas. Esses estudos afirmam se a liderança compartilhada é desenvolvida, as pessoas certas devem estar na equipe e a equipe deve ser pequena ([SM034], [SM426], [SM400], [SM544], [SM541]). Os líderes da equipe devem selecionar indivíduos de acordo com suas habilidades técnicas, de trabalho em equipe e liderança, para fazerem parte de uma equipe que deve conter entre 5 e 9 membros ([SM544], [SM541]).

Devido a um estudo possuir trecho igual ao de outro estudo, abaixo são apresentadas quatro transcrições dos cinco estudos que apontam para o tamanho das equipes como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas:

"Rising and Janoff […] describe Scrum as a development process for small teams, which includes a series of short development phases or iterations (''sprints")." [SM034] [SM426]

"If shared leadership is to be developed, the right people must be on the team, and the teams need to be small enough […]." [SM400]

"Insofar as possible, team leaders should select team members based on their technical, teamwork, and leadership skills. If shared leadership is to be developed, the right people must be on the team, and the teams need to be small […]." [SM544]

"The Scrum-master is in charge of solving problems that stops the Scrum-team (5-9 people) from working effectively." [SM541]

FATORES HUMANOS

Fator Humano foi entendido nesta pesquisa como tudo aquilo que depende diretamente dos indivíduos que fazem ou farão parte da equipe e é considerado um fator importante para os

resultados obtidos a partir do auto-gerenciamento ou que pode afetar positivamente ou negativamente o trabalho de equipes de software auto-gerenciadas.

Após a análise e a extração dos dados relevantes dos 43 estudos incluídos na pesquisa, 31 fatores humanos que podem afetar o auto-gerenciamento em equipes de software foram constatados. A Tabela IX apresenta o mapeamento desses fatores e as referências que os apontam, por ordem de importância/freqüência.

Tabela IX. Mapeamento das Evidências por Fatores Humanos encontrados (Q1.3)

Importância	Fatores Humanos	Referências	Quantidade (%)
1º	Capacidade Multifuncional	[SM400] [SM543] [SM428] [SM544] [SM043] [SM426] [SM129] [SM124] [SM034] [SM382] [SM142] [SM060] [SM573] [SM260] [SM392]	34% (15/43)
2 º	Senso de Responsabilidade	[SM493] [SM206] [SM043] [SM392] [SM016] [SM119] [SM105] [SM573] [SM260] [SM142] [SM382]	25% (11/43)
	Objetivos Comuns	[SM124] [SM378] [SM400] [SM543] [SM493] [SM426] [SM129] [SM544] [SM382] [SM541] [SM105]	
3 º	Equilíbrio entre Autonomia Individual e Interna	[SM124] [SM378] [SM034] [SM543] [SM544] [SM206] [SM129] [SM125]	18% (8/43)
4 º	Modelos Mentais Compartilhados	[SM060] [SM124] [SM034] [SM400] [SM426] [SM392]	13% (6/43)
5 º	Competência	[SM023] [SM430] [SM129] [SM544] [SM369]	11% (5/43)
	Capacidade de Tomar Decisões	[SM142] [SM369] [SM430] [SM119] [SM129]	
6 º	Capacidade de Aprendizado	[SM034] [SM544] [SM043] [SM382]	9% (4/43)
	Capacidade de Comunicação	[SM430] [SM573] [SM142] [SM016]	
	Capacidade de Adaptação	[SM493] [SM124] [SM378] [SM040]	
	Maturidade dos Indivíduos	[SM573] [SM124] [SM121] [SM129]	
	Cultura	[SM034] [SM573] [SM392] [SM060]	
	Auto-Confiança	[SM206] [SM105] [SM573] [SM400]	
	Capacidade de Trabalhar em Equipe	[SM400] [SM543] [SM544] [SM573]	
7 º	Afabilidade	[SM023] [SM430] [SM280]	6% (3/43)
	Capacidade de Auto-disciplina	[SM142] [SM119] [SM400]	
	Capacidade de Liderança	[SM382] [SM129] [SM142]	
8 º	Pro-atividade dos indivíduos	[SM124] [SM142]	4% (2/43)
	Honestidade	[SM260] [SM516]	
9 º	Capacidade de Observação	[SM142]	2% (1/43)
	Falta de Envolvimento	[SM382]	

Nível de Coesão	[SM511]
Capacidade de Auto-avaliação	[SM016]
Personalidade	[SM378]
Capacidade de Auto-aperfeiçoamento	[SM016]
Consciência	[SM280]
Capacidade de Influência Interpessoal	[SM392]
Disposição para Mudanças	[SM378]
Independência dos Indivíduos	[SM034]
Capacidade de Memorização	[SM573]
Criatividade	[SM543]

Fonte: Elaboração Própria, a partir dos dados do Mapeamento Sistemático

Os fatores que tiveram maior freqüência nos estudos serão melhor detalhados abaixo, a partir de cinco transcrições para cada um deles. Os principais fatores humanos identificados foram respectivamente: Capacidade Multifuncional (34%), Senso de Responsabilidade (25%) e Objetivos Comuns (25%), Equilíbrio entre autonomia individual e da equipe (18%) e Modelos Mentais Compartilhados (13%).

Capacidade Multifuncional

O fato de um indivíduo ser ou não capaz de possuir variedade de funções e talentos foi apontado por 34% (15/43) dos estudos incluídos como um fator humano que pode afetar o trabalho de equipes auto-gerenciadas. Esses estudos indicam que a variedade de habilidades e talentos é um pré-requisito importante para a auto-gestão e que, portanto, a seleção de indivíduos deve analisar seu potencial para desenvolver habilidades redundantes ([SM124], [SM034], [SM060], [SM382]). Isso significa que os membros de uma equipe auto-gerenciada devem ter a capacidade de trabalhar em outras áreas além de sua área de especialização. Essa capacidade multifuncional leva ao comportamento de backup, onde os membros da equipe possuem conhecimento de todas as áreas de trabalho e passam a não ter dependência em caso de falta, doença, demissão, entre outros ([SM034], [SM382]).

Indivíduos com habilidades especializadas foi encontrado por algumas pesquisas como sendo a principal barreira para o desenvolvimento do auto-gerenciamento em equipes de software ([SM382], [SM034], [SM543]). Entre os resultados de se trabalhar o auto-gerenciamento com indivíduos com habilidades especializadas, estão o baixo nível de comprometimento da equipe,

menos interação do que o necessário, redução da flexibilidade, redução da autonomia interna, redução da confiança na equipe e dificuldade em alinhar decisões a nível operacional, devido aos membros da equipe não saberem o que os outros estão fazendo ([SM382], [SM543], [SM544]).

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para a capacidade multifuncional como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas:

> *"Cross-functionality is the ability of team members to look beyond their area of specialization by taking an interest in activities outside their specialization." [SM124]*
>
> *"In the literature on self-managed teams, backup behavior has been identified as an important prerequisite for self-management […]. In our study, highly specialized skills and a corresponding division of work was the most important barrier to achieving backup and then self-management." [SM034]*
>
> *"The main reason for this low team-level commitment was specialization; in all the teams we studied, developers worked independently on particular modules according to specialization (for example, user interfaces, map interfaces, and databases). (...)had less interaction than needed, which made team-level commitment difficult. (…) In addition, several developers said it's difficult to commit to work they aren't involved in and that if team-members got sick or side-tracked, part of the iteration plan wouldn't get done. (...)Consequently, the highly specialized developers in companies A and B focused only on their "own" modules and had little interaction with others. This resulted in difficulties aligning decisions on the operational level because team members didn't know what others were doing. (…) As a result, this developer lost his team's trust. In addition, the scrum-master, who'd changed his behavior and seldom used his former decision-making authority, started to give him direct instructions. (…)When recruiting, select people with the potential to develop redundant skills." [SM382]*
>
> *"We found the most important barrier to be the highly specialized skills of the developers and the corresponding division of work. (…)This lack of redundancy reduced the flexibility and, thus, the internal autonomy of the team, and was therefore the most important barrier for self-organizing." [SM543]*

"Not every decision must be made jointly with equal involvement by every team member; rather the team can also delegate authority to individuals or subgroups within the team. One big threat against internal autonomy is highly specialized skills and corresponding division of work […]. This can result in a lack of redundancy, which reduce the flexibility and, thus, the internal autonomy of the team. But a paradox arises: how much redundancy should be built into a system?" [SM544]

Senso de Responsabilidade

O senso de responsabilidade dos indivíduos foi encontrado como um fator humano que pode afetar o trabalho de equipes de software auto-gerenciadas e teve sua importância mencionada por 25% (11/43) dos estudos incluídos na pesquisa. Fato é que, como apresentado na Seção 4.2.2.1, a responsabilidade compartilhada entre os indivíduos é vista como uma das mais importantes características do auto-gerenciamento de equipes de software. Isso indica que para que essa responsabilidade compartilhada funcione direito, cada indivíduo deve ter consciência de sua responsabilidade. Os estudos aqui identificados também sugerem que os membros das equipes devem compreender, aceitar e apresentar-se a suas responsabilidades com o trabalho e com a equipe ([SM060], [SM206], [SM573]). As responsabilidades dos indivíduos incluem: entrega das atividades, verificação do progresso, manutenção do progresso, alertar caso não haja progresso ([SM493], [SM573]). Se cada indivíduo possui a consciência de sua responsabilidade com a equipe e a assume, os efeitos de uma responsabilidade compartilhada são sentidos pela equipe ([SM140]).

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para o senso de responsabilidade como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas:

"User stories (simplified requirements) are an example of collective ownership in action, where the team has responsibility for delivery of all user stories, even in situations where individual team members accept responsibility for particular stories" [SM493]

"When creating a leadership team, it is important that the leadership team members understand and agree with expectations and their responsibilities. A key aspect is the time commitment." [SM060]

"(…) studies of XP indicate that successful teams manage to balance a high level of individual autonomy with a high level of team autonomy and corporate responsibility. (…)Respect for one's team members and a sense of responsibility were manifested via the way in which work was assigned; active agreement was required. ''Individuals clearly felt that they had the respect of their fellow team members and were therefore empowered to take on responsibility (…)Robinson and Sharp (S24) claim that agile development teams have faith in their own abilities, show respect and responsibility, establish trust, and preserve the quality of working life. Young et al.. (S33) used a technique called ''repertory grid analysis" to identify good personality characteristics for members of XP development teams" [SM206]

"A key feature of all the activity which we observed was that it exhibited evidence for a shared purpose, understanding and responsibility within the team. By that we mean that what work needed to be done was discussed, decided and agreed in a shared fashion, the detail of how that work might be executed was similarly a shared negotiation by the team, and responsibility for ensuring that the execution was satisfactorily carried out was collective. (…)The willingness to take responsibility, just to get on with things, not to be told by somebody else that you have to do certain things. It's very much about getting on with it. People see gaps and go to fill them. (…)This orchestration via the Chekov board was public and visible. Each developer had a responsibility, to themselves and to others, to check progress, to maintain progress, and to shout if progress wasn't happening." [SM573]

"They did not feel collectively responsible for the overall development. We call this problem the Notion of Responsibilities problem. It directly opposes the Agile principles of collective code ownership, self-organizing and cross-functional teams." [SM140]

Objetivos Comuns

A importância de todos os membros da equipe possuírem objetivos comuns foi relatada por 25% (11/43) dos estudos incluídos. De acordo com esses estudos, equipes auto-gerenciadas devem possuir objetivos comuns ao invés de objetivos individuais ([SM124], [SM378]). Para alguns pesquisadores, o que leva à prioridade a objetivos pessoais pelos membros é a falta de modelos mentais compartilhados, indicado por 13% (6/43) estudos como outro fator humano

que afeta o auto-gerenciamento, e isso reduz a possibilidade de monitoramento de desempenho mútuo ([SM426]). A prioridade a objetivos pessoais pode também levar os indivíduos a não assumirem responsabilidades e, portanto, constituir uma barreira à autonomia interna ([SM543]).

Por outro lado, o estudo [SM129] sugere que o fato de indivíduos enfatizarem objetivos da equipe ao invés de seus próprios pode prejudicar a autonomia individual e, portanto, a motivação do indivíduo.

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para objetivos comuns entre individuo e equipe como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas:

> *"Team orientation or collective orientation is often described as giving priority to team goals over individual goals […]." [SM400]*
>
> *"Uhl-Bieh and Graen […] found that if a cross-functional team is staffed with independently focused self-managing professionals, this may harm the organization and decrease the effectiveness. (…)Since problems were seen as personal and therefore seldom discussed in the meetings, it was difficult to have an equal influence over the solution of important problems. (…)The quagmire resulted in people not taking responsibility because their personal goals became more important than the team's goals. Therefore the quagmire can also be seen as a barrier for internal autonomy." [SM543]*
>
> *"This is also known as a lack of team orientation […]. For every team, particularly for a selforganizing team, the team's goals should be more important than the individual goals." [SM541]*
>
> *"Many researchers suggest that individuals should emphasize the team goals over their own […], […], […]. On one hand, losing individual autonomy is harmful for the individual's motivation […]. On the other hand, too much individual autonomy is a threat for team work […]. (…)According to the literature, team members should emphasize the team goals over their own goals […], […], […], which may be in contradiction with individual autonomy […], […]." [SM129]*

"For the project, participants said they had a better overview than in previous projects, but because of a lack of shared mental models there were individual goals and this reduced the possibility for mutual performance monitoring, since missing shared mental models leads to ineffective feedback […]." [SM426]

Equilíbrio entre autonomia individual e da equipe

O equilíbrio ou a falta de equilíbrio entre autonomia individual e de equipe foi constatado por 18% (8/43) dos estudos incluídos como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas.

De acordo com esses estudos, a autonomia individual se refere ao nível de liberdade que um indivíduo possui na realização de tarefas e define como um trabalho é realizado dentro da equipe ([SM544], [SM129]). Pesquisas científicas e trabalhos de campo têm demonstrado que o incentivo à autonomia individual pode trazer muitos efeitos positivos, tais como a motivação extrínseca, a satisfação no trabalho e atitudes positivas de trabalho ([SM125]).

Porém, indivíduos com alta autonomia individual são descritos como tendo poucas regras e restrições de procedimentos, alto controle sobre regras e procedimentos e alto controle sobre a natureza e o ritmo de seu trabalho ([SM544]). Por isso, pode constituir um efeito negativo para o desempenho da equipe, uma vez que faz com que a interação entre os indivíduos seja reduzida ([SM034], [SM544]).

Portanto, a alta autonomia individual foi encontrada por alguns estudos como uma barreira para o auto-gerenciamento, visto que esta pode entrar em conflito com a autonomia a nível da equipe ([SM124], [SM378], [SM034], [SM543], [SM206], [SM105]) e assim afetar a coesão da equipe e, indiretamente, reduzir a efetividade da equipe, fazendo com que a organização não consiga obter os resultados de uma equipe auto-gerenciada ([SM125], [SM543]). Deste modo, deve ser estudado uma forma de se alcançar o equilíbrio entre a autonomia individual e da equipe.

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para a existência do equilíbrio entre autonomia individual e da equipe como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas:

"Further, autonomy on the individual level may conflict with autonomy on the group level. (...) Markham and Markham [29] suggested that it may be difficult to incorporate

both individual autonomy and group autonomy in the same work group.(...) This created problems when the team members tried to change their normal way of working to become part of a self-managed team. Our findings confirm previous research by Langfred [], such that there can be a negative effect on team performance when teams are trying to function as a self-managed team when the team members have high individual autonomy." [SM034]

"Langfred […] argues that a software team probably needs group autonomy and individual autonomy, but that autonomy on an individual level may conflict with autonomy on a group level, producing countervailing influence on the cohesiveness and, indirectly, effectiveness of the team." [SM543]

"Individual autonomy refers to the amount of freedom and discretion an individual has in carrying out assigned tasks […]. Langfred […] describes individuals with high autonomy as individuals who have few rules and procedure constraints, high control over rules and procedures, and high control over the nature and pace of their work. As individual work becomes more independent, and as individuals exert more control over the scheduling and implementation of their own tasks, there will be less interaction between group members. Given reduced contact between group members, it may be a problem to achieve a high level of internal autonomy. Autonomy on an individual level may conflict with autonomy on a group level, because when individual goals are more important than the team's goals this reduces the internal autonomy […]." [SM544]

"Meanwhile, internal autonomy defines how the work is organized inside the team […]. The team may have substantial power to make decisions while some individuals have none […]. Great care should be taken to make sure that there really is internal autonomy instead of, for example, team leader autonomy […]. (…) individual autonomy, on its part, tells how much an individual has freedom to decide about his or her own work processes […]." [SM129]

"However, individual and team level autonomy may conflict, and reduce the effectiveness of the team. Therefore, there is a need to investigate how to achieve empowerment at the individual and team levels simultaneously. (…)It is worth noting that while agile practices, like Scrum and XP, often give the team as a whole a great deal of autonomy, it does not follow that the individual team members are given high levels of individual autonomy. (…)The fact that autonomy can simultaneously reside at both the group and

the individual level in a work group is often neglected in studies of self-management […]. A group may have considerable discretion in deciding what group tasks to perform and how to carry them out, but individual members within the group may have very little discretion or control over their jobs (…). (...)Encouraging individual autonomy has been shown to have many positive effects in both laboratory and field research […]. Offering workers greater individual autonomy increases: the degree to which extrinsic motivation offered employees is internalized […], job satisfaction and positive work attitudes […]. Greater internalization of extrinsic motivation has been shown to lead to effective performance on tasks requiring creativity, cognitive flexibility and conceptual understanding […]. These are the very characteristics of software development work. Practices that increase individual autonomy are, therefore, extremely valuable, especially if they can be leveraged to provide real value to the employer. (...) Autonomy at the individual level may conflict with autonomy at the group level, producing countervailing influence on the cohesiveness and, indirectly, effectiveness of the team. An organization could thus experience little or no results from empowering employees. " [SM125]

Modelos Mentais Compartilhados

Dentre os estudos incluídos, 13% (6/43) indicaram que o fato de a equipe possuir ou não modelos mentais compartilhados pode afetar o auto-gerenciamento. De acordo com esses estudos, modelos mentais compartilhados e comportamentos padronizados constituem um importante componente para as equipes, particularmente para um trabalho cooperativo, e alguns pesquisadores o relacionam ao aumento da qualidade do software e desempenho da equipe ([SM124], [SM034], [SM400]).

O desenvolvimento de modelos mentais no auto-gerenciamento é suportado através do foco na visão do projeto, planejamento e pelas reuniões do processo, detalhado no tópico Fatores Técnicos desta seção ([SM426]).

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para modelos mentais compartilhados como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas:

"When creating a leadership team, it is important that the leadership team members understand and agree with expectations and their responsibilities. A key aspect is the time commitment. (AGILE)" [SM060]

"Nonaka et al.. found that a team possesses cross-fertilization when (a) it is composed of individual members with varying specializations, thought processes, and behaviour patterns and (b) these individuals interact amongst themselves leading to better understanding of each other's perspectives […]." [SM124]

"(...) important components, such as trust and shared mental models. (...) Working cooperatively requires the team to have shared mental models […]. Our results are also consonant with Salas et al..'s […] findings that without a shared understanding, the individual members may be headed toward different goals, which in turn will lead to ineffective/lack of feedback or assistance." [SM034]

"They found that high team vision preferences (or mental models) and high participative safety (degree of trust) perceptions of the team were related to improved software quality. (..)to develop shared mental models, in order to improve team performance (…). (…) Neither culture nor mind-sets of people are easily changed, making the move to agile methodologies all the more formidable for many organizations […]" [SM400]

"Shared Mental Models: are supported in Scrum through the involvement by the product owner, focus on the project vision, and the planning resulting in a backlog, and the retrospective meeting. The daily scrum is important for understanding team members' tasks. (…)In the case, because of highly specialized skills and a corresponding division of work, it was a problem to develop shared mental models both on a project level and on a task level. Without a clear understanding of the system being developed, planning was difficult. The planning meetings turned out to be discussions between the Scrum-master, product owner and the developer that was going to do the work. " [SM426]

FATORES ORGANIZACIONAIS

Fator Organizacional foi entendido nesta pesquisa como tudo aquilo que depende diretamente das organizações, sendo representadas também pela gerência e alta gerência e que é considerado um fator importante para os resultados obtidos do auto-gerenciamento ou que pode afetar positivamente ou negativamente o trabalho de equipes de software auto-gerenciadas.

Após a análise e a extração dos dados relevantes dos 43 estudos incluídos na pesquisa, 38 fatores organizacionais que podem afetar o auto-gerenciamento em equipes de software foram

constatados. A Tabela X apresenta o mapeamento desses fatores e as referências que os apontam, por ordem de importância/freqüência.

Tabela X. Mapeamento das Evidências por Fatores Organizacionais encontrados (Q1.3)

Importância	Fatores Organizacionais	Referências	Quantidade (%)
1º	Concessão de Autoridade	[SM105] [SM129] [SM034] [SM125] [SM400] [SM280] [SM543] [SM493] [SM121] [SM544] [SM382] [SM329] [SM428] [SM541] [SM016] [SM040] [SM043] [SM511] [SM140] [SM060] [SM378] [SM023] [SM573] [SM262]	55% (24/43)
2 º	Gerência Facilitadora e Colaboradora	[SM378] [SM142] [SM060] [SM428] [SM544] [SM506] [SM206] [SM534] [SM426] [SM129] [SM543] [SM541] [SM040] [SM430] [SM152] [SM016] [SM511]	39% (17/43) (39,53%)
3 º	Feedback	[SM129] [SM124] [SM378] [SM034] [SM400] [SM543] [SM262] [SM121] [SM060] [SM544] [SM206] [SM426] [SM528] [SM573] [SM516]	34% (15/43)
4 º	Nível de Controle	[SM129] [SM124] [SM378] [SM543] [SM516] [SM544] [SM382] [SM142] [SM369] [SM428] [SM206]	25% (11 /43)
	Nível de Autonomia Externa	[SM129] [SM400] [SM543] [SM121] [SM544] [SM428] [SM541] [SM426] [SM573] [SM260] [SM124]	
5 º	Sistema Formal de Apoio da Alta Administração	[SM378] [SM493] [SM124] [SM034] [SM382] [SM543] [SM043] [SM426] [SM511]	20% (9/43)
	Liderança	[SM129] [SM124] [SM378] [SM400] [SM541] [SM040] [SM382] [SM544] [SM426]	
	Recursos Disponíveis	[SM129] [SM124] [SM378] [SM400] [SM493] [SM544] [SM382] [SM142] [SM573]	
	Estrutura de Espaço Aberto	[SM129] [SM034] [SM382] [SM121] [SM043] [SM140] [SM260] [SM516] [SM142]	
	Compartilhamento de Informações	[SM121] [SM544] [SM129] [SM493] [SM185] [SM016] [SM040] [SM034] [SM573]	
6 º	Cultura de Especialistas/Generalistas	[SM129] [SM400] [SM426] [SM140] [SM382] [SM543] [SM105] [SM573]	18% (8/43)
	Crença nos Indivíduos e Equipes	[SM125] [SM516] [SM493] [SM129] [SM329] [SM043] [SM426] [SM511]	
7 º	Concessão de Livre Autonomia	[SM129] [SM280] [SM103] [SM549] [SM400] [SM142]	13% (6/43)
	Motivação aos Indivíduos	[SM124] [SM378] [SM152] [SM016] [SM023] [SM426]	
	Eficácia na Comunicação	[SM543] [SM541] [SM142] [SM185] [SM573] [SM140]	

8 º	Envolvimento do Cliente	[SM129] [SM378] [SM382] [SM060] [SM262] [SM378]	11% (5/43)
	Articular Confiança	[SM400] [SM544] [SM543] [SM369] [SM140]	
	Hierarquia	[SM142] [SM280] [SM121] [SM040] [SM140]	
	Composição/Formação da Equipe	[SM124] [SM544] [SM060] [SM511] [SM400]	
9 º	Treinamentos para a Equipe	[SM129] [SM544] [SM382] [SM543]	9% (4/43)
	Supervisão	[SM543] [SM400] [SM382] [SM544]	
	Concessão de Liberdade	[SM124] [SM493] [SM142] [SM129]	
	Concessão de Responsabilidade	[SM129] [SM034] [SM125] [SM541]	
	Regras Organizacionais	[SM142] [SM543] [SM262] [SM544]	
10 º	Fornecimento de Visão do que é necessário à Equipe (Articular Visão)	[SM400] [SM511] [SM544]	6% (3/43)
	Direção Clara	[SM400] [SM544] [SM034]	
	Respeito à Equipe	[SM382] [SM105] [SM516]	
11 º	Alinhamento dos Indivíduos	[SM124] [SM378]	4% (2/43)
	Exigências no Trabalho	[SM103] [SM493]	
	Relação da Alta gerencia com a Equipe	[SM382] [SM142]	
	Recursos Compartilhados	[SM382] [SM543]	
	Gestão descentralizada	[SM142] [SM544]	
	Sistema de Recompensa	[SM129] [SM382]	
	Flexibilidade de Trabalho	[SM573] [SM511]	
12 º	Intercâmbio Regular de Idéias	[SM185]	2% (1/43)
	Controle Organizacional	[SM382]	
	Foco das Reuniões	[SM543]	
	Concessão de Influência à Equipe	[SM129]	

Fonte: Elaboração Própria, a partir dos dados do Mapeamento Sistemático

Os fatores que tiveram maior freqüência nos estudos serão melhor detalhados abaixo, a partir de cinco transcrições para cada um deles. Os principais fatores organizacionais identificados foram respectivamente: Concessão de Autoridade (55%), Gerência Facilitadora e Colaboradora (39%), Feedback (34%), Nível de Controle (25%) e Nível de Autonomia Externa (23%).

Concessão de Autoridade

A concessão de autoridade à equipe pela gerência e alta gerência foi apontada por 55% (24/43) dos estudos incluídos como fator que pode afetar o trabalho de equipes de software auto-gerenciada. Para estes estudos, deve ser concedida à equipe plena autoridade para organizar seu trabalho da maneira que decidem ser necessário para alcançar os objetivos da maneira que acreditam ser mais efetiva ([SM043], [SM034], [SM125], [SM541], [SM543]). Isso significa que deve ser concedida à equipe auto-gerenciada significante autoridade para diversos aspectos de seu trabalho, tais como: planejamento, cronograma, atribuição de tarefas entre os membros e tomada de decisões ([SM034], [SM125], [SM541], [SM543]).

Portanto, de acordo com esses estudos, para se tornarem e permanecerem auto-gerenciadas e alcançarem seus benefícios, as equipes devem realmente afetar decisões gerenciais e ser capazes de alterar normas e regras de funcionamento dentro da equipe, assim como no ambiente mais amplo ([SM382], [SM125]). Com a autoridade sobre esses aspectos concedida à equipe, os membros tornam-se mais propensos a sentir que suas idéias são valorizadas pelos colegas e a desenvolver maiores níveis de apoio e confiança para com o outro. Assim, o compromisso com a equipe pode ser aumentado, resultando no aumento da coesão da equipe ([SM121]).

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para a concessão de autoridade à equipe como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas:

> *"A Scrum team is given significant authority and responsibility for many aspects of their work, such as planning, scheduling, assigning tasks to members, and making decisions: ''The team is accorded full authority to do whatever it decides is necessary to achieve the goal" […]." [SM034] [SM125] [SM541] [SM543]*
>
> *"If team members are allowed to make decisions jointly, they are also likely to develop higher levels of support and trust for each other (Sethi, 2000). Additionally, individuals making decisions jointly are more likely to feel that their ideas are worthy and valued by colleagues. As such, they may further their commitment to the team and their fellow teammates, resulting in increased team cohesion and mutual support. Furthermore, joint decision making improves team members' identification with the (joint) decision." [SM121]*

"Sprint is a period of time, when a team is focusing on meeting the sprint commitments. During this period of time the team is supposed to have full authority over its actions and no external influence from the Product Owner, or anybody else, is allowed." [SM428]

"The Scrum team has the authority to organize their own work in whatever way to seem most effective. They are, for instance, allowed to hire external experts." [SM043]

"Consequences of the authority problem were not explicitly mentioned in the case studies. However, when further analyzing them, we understood that these problems hindered communication, open knowledge transfer, process improvement and created divisions between team members and management." [SM140]

Gerência Facilitadora e Colaboradora

Dentre os estudos incluídos nesta pesquisa, 39% (17/43) indicaram que um dos fatores que afeta o trabalho de equipes auto-gerenciadas é o estilo de gerência facilitadora e colaboradora. Esses estudos afirmam que, no auto-gerenciamento, o papel do gerente de projeto é transformado de um controlador e planejador para ser facilitador e colaborador, mais como um coordenador ([SM378], [SM544], [SM016]). Portanto, é recomendado que os gerentes de projeto estejam prontos para renunciar algum controle, para, com isso, criar um ambiente que estimule o surgimento do auto-gerenciamento ([SM378], [SM142]). Isso significa que o gerente deve remover obstáculos que impedem o desenvolvimento da equipe, proteger a equipe contra ruídos externos e facilitar os diferentes processos utilizados pela equipe, tudo para que a equipe trabalhe de forma eficaz ([SM426], [SM428], [SM544]). Porém, o gerente deve deixar que a equipe se auto-organize e tome suas decisões ([SM428], [SM544]). Caso isso não aconteça e o gerente trabalhe mais como um controlador do que facilitador, ele irá reduzir a capacidade da equipe se auto-gerenciar ([SM426]).

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para o estilo de gerência facilitadora e colaboradora como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas:

"Agile methods require that the role of the project manager needs to change from being a controller and planner to becoming a facilitator and collaborator […]. It is recommended that managers on Agile projects be ready to relinquish some control […]. (…) A Scrum Master is seen as a facilitator and does not organize or manage the team

[…]. Similarly, an XP Coach is meant to lead the team towards self-organization by leaving the team alone as early as possible […].” [SM378]

“In a self-managed team, a manager’s role is more of a facilitator, creating an environment that fosters the emergence of self-organization.(...)Secondly, although effective communication with customers and between team members is essential to being agile, over-long communication and extensive discussion can lead to stasis and a consequent deterioration in the coevolutionary process. Third, the project manager has an on-going struggle o not be externalized from the project team; the project manager needs to continue to live inside the team at the same time as fulfilling the role as a facilitator of a self-organizing team. Consequently, the effectiveness of agile methods use, as Maruping et al.. (2009) argue, is contingent.” [SM142]

“Scrum and Agile development favour a leadership and collaboration style of management where the traditional project manager’s role is replaced with the Scrum-master’s role of a facilitator or coordinator […]. (...)The Scrum-master is in charge of solving problems that stops the Scrum-team (5-9 people) from working effectively. The Scrum-master is often described as a coach or facilitator. He or she does not organize the team (designers and developers); the team organizes itself and makes the decisions concerning what to do. (…) The Scrum-master works to remove the impediments of the process, runs and makes decisions in the daily meetings and validates them with the management […].” [SM428] [SM544]

“The product manager worked as a facilitator rather than as a manager and led the team by facilitating and governing them not only as a manager.” [SM016]

“The Scrum-master is presented as a coach, focusing on protecting the team against external noise, removing impediments and facilitating the different processes defined by Scrum. (…) Ability to direct and coordinate the activities of other team members, assess team performance, assign tasks, develop team knowledge, skills, and abilities, motivate team members, plan and organize, and establish a positive atmosphere (…). (...) Engage in preparatory meetings and feedback sessions with the team (…)When the Scrum master acted more like a project manager than a coach, he reduced the ability of the team to self-organize.” [SM426]

Feedback

O *feedback* ou retorno que é passado à equipe sobre o que é produzido foi apontado por 34% (15/43) dos estudos incluídos como fator que pode afetar o trabalho de equipes de software auto-gerenciadas. De acordo com os estudos, quando a equipe segue a prática de um processo iterativo e incremental, como descrito no tópico de Fatores Técnicos desta seção, o *feedback* dado à equipe ao final de cada iteração permite que a equipe reflita sobre como o trabalho pode ser melhorado e planeje melhor a próxima iteração, adaptando às necessidades ([SM060]). Além disso, o feedback concedido à equipe ajuda na conclusão bem sucedida do projeto e a incentivar a criatividade dentro da equipe ([SM400] [SM543] [SM121]). Por outro lado, nos casos em que o cliente não está disponível ou em que a equipe não possui abertura para discutir com o cliente, o aprendizado da equipe é comprometido, do ponto de vista do auto-gerenciamento ([SM129]).

Por isso, diversos estudos indicam a importância de feedback honesto e contínuo no desenvolvimento de projetos de software com equipes auto-gerenciadas ([SM129], [SM124], [SM034], [SM400] [SM543] [SM121], [SM543], [SM516], [SM206]).

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para *feedback* como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas:

> *"While some forms of team-external influence are sometimes beneficial because they provide important feedback to help project completion or encourage creativity within the team by discouraging groupthink, Hoegl and Parboteeah […] argue that the specific type of team-external influence considered here is detrimental to teamwork in projects e.g. developing software." [SM400] [SM543] [SM121]*
>
> *"(…)Finally, the importance of whole team awareness and feedback in agile software development teams has implications for the design of methods for distributed software development." [SM516]*
>
> *"At the end of each Sprint, the team demonstrates work completed to stakeholders, reflects on how it can improve, and plans the next Sprint. (...)Agile software development promotes the development and delivery of completed work in short Btime-boxed[iterations (which are also called Sprints) so that teams can obtain frequent feedback from*

stakeholders and adapt as needed. Here, the term time-box refers to a period of time in which to accomplish a task." [SM060]

"The team received feedback on their beliefs from successfully executing code, from a satisfied customer, and from support and encouragement from each other" [SM206]

"Similarly, the importance of honest and continuous feedback supports the existing literature […]. (...) The case studies emphasized the importance of good customer communication in software development projects. If the customer is not available or if the team and the customer are not able to discuss openly, the team cannot get the feedback needed. From the self-organization point of view, this deteriorates learning in the team." [SM129]

Nível de Controle

O nível de controle estabelecido pelos gerentes sobre a equipe foi apontado por 25% (11/43) dentre os estudos incluídos como um fator que pode afetar o trabalho de equipes auto-gerenciadas. De acordo com esses estudos, gerentes devem manter um controle sutil e adaptativo, concedendo substancial liberdade e pontos de inspeção regulares ([SM129], [SM124], [SM378]). Assim, o gerente deve evitar um controle do tipo rígido, que iniba a criatividade e a espontaneidade da equipe ([SM543], [SM544], [SM382]). Do contrário, pode resultar em altos níveis de estresse e descontentamento com esforços coletivos ([SM516]).

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para o nível de controle estabelecido pelos gerentes sobre a equipe como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas:

"Self-organizing teams are not leaderless, uncontrolled teams […]. Leadership in self-organizing teams is meant to be light-touch and adaptive […], providing feedback and subtle direction […]. Leaders of Agile teams are responsible for setting direction, aligning people, obtaining resources, and motivating the teams […]." [SM124] [SM378]

"To build trust in the whole organization, management should avoid any control that would impair creativity and spontaneity. The teams' need for continuous learning, not the company's need for control, should motivate the need for data collection." [SM382]

"Takeuchi and Nonaka […] found that although the members of a self-organized team are largely on their own, they are not uncontrolled. Management should establish enough checkpoints to prevent instability, ambiguity, and tension from turning into chaos. At the same time, same time, managers should avoid the kind of rigid control that impairs creativity and spontaneity. (…)" [SM543] [SM544]

"In his article, Tightening the Iron Cage: Concertive Control in Self Managing Teams, James Barker […] discusses the strong social forces present in post-bureaucratic structures such as self-regulating teams. Barker outlines a case study in which a selfregulating team suffers from normative (group) control that becomes constrictive to the individuals within it, resulting in high levels of stress and disenchantment with collective efforts. Barker describes a situation where heightened social awareness and involvement in a team eventually results in a heavy-handed system of social control, within which team-members suffer from extreme loss of freedom and reduction in job satisfaction." [SM516]

"A team must have a real possibility to influence relevant matters; otherwise self-organization is more symbolic than real […]. On the other hand, a team should not be left completely alone […]. Instead, while management should give a team substantial freedom, it should maintain subtle control and have regular checkpoints […]." [SM129]

Nível de Autonomia Externa

O nível de autonomia externa da equipe foi indicado por 25% (11/43) dos estudos incluídos como um fator que pode afetar o trabalho de equipes auto-gerenciadas. A autonomia externa pode ser entendida como o grau em que gerentes e outros indivíduos externos à equipe influenciam as atividades e decisões da equipe sobre estratégias de trabalho ou processos, metas do projeto e alocação de recursos ([SM129], [SM543], [SM544]).

Algumas formas de influência externa são por vezes benéficas à equipe, pois promovem importantes feedbacks que ajudam na conclusão do projeto e no incentivo à criatividade da

equipe ([SM543], [SM544]). Porém, quando os membros da equipe sentem que grande parte do projeto reflete as demandas externas e sentem que as decisões estão sendo impostas sobre eles, a tendência é que eles se tornem menos propensos a se identificar com o projeto e a contribuir com seus conhecimentos para a conclusão bem sucedida do projeto ([SM121]). Além disso, o grau de interferências externas sobre assuntos da equipe pode indicar para os membros que os gerentes não confiam na capacidade da equipe tomar decisões e, assim, resultar na redução da satisfação dos indivíduos e do compromisso com a equipe. Isso, por sua vez, pode ser associado à menor coesão e esforço ([SM121]). Portanto, o nível de autonomia externa da equipe pode representar uma ameaça à natureza do auto-gerenciamento ([SM124]).

Abaixo são apresentadas cinco das transcrições extraídas dos estudos incluídos na pesquisa que apontam para o nível de autonomia externa como um fator que pode afetar o trabalho de equipes de software auto-gerenciadas:

> *"They found that the management did not provide an environment conducive to self-organization that led to reduced external autonomy. (…) A team's failure to balance between cross-functionality and specialization, as in this example, can invite senior management intervention. Frequent senior management interference can pose a threat to the team's self-organizing nature." [SM124]*

> *"External autonomy is defined by Hoegl and Parboteeah […] as the influence of management and other individuals (outside the team) on the team's activities. Such influence can be deliberate actions from management to limit autonomy, such as requiring the team to make certain decisions regarding work strategies or processes, project goals and resource allocation. While some forms of team-external influence are sometimes beneficial because they provide important feedback to help project completion or encourage creativity within the team by discouraging groupthink, Hoegl and Parboteeah […] argue that the specific type of team-external influence considered here is detrimental to teamwork in projects e.g. developing software." [SM543] [SM544]*

> *"Sometimes, there is removal of team decision-making discretion because top management does not share an understanding of the product development process with the team (Clark & Wheelwright, 1992). This interference may take the form of higher levels of management requesting that they be consulted for major or minor (i.e., operational) decisions, thus representing a team-external influence on the project. Other reasons are that managers may not buy into the concept of team autonomy (Gerwin &*

Moffat, 1997a, 1997b).(…) Such information sharing and task coordination within the team, however, is likely reduced in cases of strong team-external influence over project decisions, as technical and coordinative information is more likely channeled to the team-external decision maker (e.g., team-external manager). (…)If the team members feel that the project reflects largely external demands, then they are less likely to identify with the project (Sethi, 2000). (…)If team members feel that decisions are being imposed on them, they are less likely to contribute their own knowledge to bringing the project to successful completion. Moreover, team-external influence on project decisions may also signal to team members that external managers do not buy into the team autonomy idea (Gerwin & Moffat, 1997a, 1997b) or do not trust the team to be able to make such decisions. (…) Both cases are likely to result in team members' lower satisfaction with the team (Kirkman & Rosen, 1999) and lower commitment to the team (Wall et al.., 1986). Cohen (1990) showed that top management's involvement in operational decision making undermined the team members' feelings of authority, responsibility, and accountability. This involvement, in turn, is likely to be associated with lower cohesion and effort. Furthermore, as self-reliance in more autonomous teams is replaced with team-external decision making in less autonomous teams, the likelihood for sincere and constructive discussion within the team on operational project matters is likely reduced (if not to say largely preempted) by outside decision making. Instead, team members turn to those influential team-external people or groups for task decisions as well as guidance and support in carrying out their work." [SM121]

"When decisions are made outside the team, such decisions reflect outside perspectives. If the team members feel that the project reflect largely external demands, then they are less likely to identify with the project […]. The level of external autonomy can be reduced by the lack of a system for support and people working on several projects because the team will then have problems protecting their resources. […]. Also the team need to be involved in the initially planning of the project (deadlines, contracts etc) […]." [SM544]

"The customer participates in the sprint meetings, but is not allowed to influence the team in between the meetings" [SM426]

4.2.2.4. Q2 Força das Evidências Encontradas

Q2. Qual a força da evidência em apoio às conclusões encontradas?

Esta questão central da pesquisa objetiva avaliar o quão forte são as evidências encontradas que apóiam as conclusões desta pesquisa. Para responder a esta questão, foram realizadas avaliações da qualidade dos 43 estudos incluídos. Essas avaliações servem para determinar a força das inferências e para atribuir notas a recomendações geradas (KHAN et al., 2001). Apesar de não existir uma definição universal do que é realmente a qualidade de um estudo, em geral, os *checklists* abordam questões que visam avaliar a extensão em que o viés é minimizado e a validade interna e externa são maximizadas (KHAN et al., 2001; KITCHENHAM, 2007 *apud* MONTEIRO, 2010).

Para a realização dessas avaliações, foram utilizados alguns critérios de qualidade a depender do tipo de estudo identificado. Aqui, foram considerados estudos experimentais, teóricos e secundários. Estudos experimentais, ou empíricos, são baseados em evidências diretas ou experimentos. Os estudos teóricos são conceituais e baseados em um entendimento de uma área, referenciando outros trabalhos relacionados. Já os estudos secundários reexaminam outros trabalhos ou fazem mapeamentos sistemáticos e revisões da literatura sistemáticas ou tradicionais.

O processo de avaliação da qualidade de cada um dos 43 estudos considerados relevantes para esta pesquisa se deu a partir da avaliação independente de dois pesquisadores. Após a finalização da avaliação individual, ambos os pesquisadores reavaliaram cada item em que houve divergência e chegaram a um acordo comum sobre elas. Quando isso não foi possível, um terceiro pesquisador fez sua avaliação e atribuiu sua nota ao critério divergente, assim gerando uma única lista contendo a pontuação final de cada estudo.

Dentre os estudos encontrados 93% (40/43) foram classificados como estudos empíricos e apenas 6% (3/43) foram classificados como estudos secundários. Os Formulários A e C, apresentados na Seção A6 do APÊNDICE A, foram utilizados para avaliar a qualidade de estudos empíricos e secundários, respectivamente. O Formulário B que contém os critérios referentes a estudos teóricos não foi utilizado, pois nenhum estudo classificado como teórico foi considerado relevante e, portanto, incluído nesta pesquisa.

Todos os critérios de qualidade, tanto para estudos empíricos quanto para os estudos secundários, foram elaborados em forma de perguntas. Aos critérios do Formulário A, específico para estudos empíricos, foram atribuídas as notas “0” ou “1”, respectivamente significando

"Não" e "Sim", para 10 critérios de qualidade baseados em Dyba e outros (2008). Aos 4 critérios do Formulário C, para estudos secundários, foram atribuídas as notas "0", "0,5" e "1", que significam respectivamente "Não", "Parcialmente" e "Sim", com base em no sugerido em *Centre for Reviews and Dissemination (2007*).

A pontuação total dos estudos foi dada a partir da soma de todas as notas a ele atribuídas. No Formulário C, de estudos secundários, caso o estudo avaliado não atingisse nota final igual ou superior a 1 deveria ser excluído, como aconteceu com 2 estudos que não puderam ser incluídos na pesquisa. Dessa forma, a pontuação total de um estudo secundário poderia variar entre 1 (25%) e 4 (100%) e a de um estudo empírico poderia variar entre 0 (0%) e 10 (100%).

Com base nessas informações, a fim de evitar atribuições arbitrárias e produção de viés, a extensão da pontuação foi dividida em quatro faixas, assim como feito por Belbin (1981), são elas: baixa, média, alta e muito alta. A Tabela XI apresenta como será feita a distribuição da pontuação entre as faixas estabelecidas e os tipos de estudos identificados.

Tabela XI. Distribuição da pontuação por Faixa de Qualidade

Qualidade	Baixa	Média	Alta	Muito Alta
Estudos Empíricos	$0 \leq N \leq 2$ (0% a 20%)	$3 \leq N \leq 5$ (30% a 50%)	$6 \leq N \leq 8$ (60% a 80%)	$9 \leq N \leq 10$ (90% a 100%)
Estudos Secundários	N=1 (25%)	$1,5 \leq N \leq 2$ (37,5% ou 50%)	$2,5 \leq N \leq 3$ (62,5% ou 75%)	$3,5 \leq N \leq 4$ (87,5% ou 100%)
Onde N = Nota atribuída na Avaliação				

Fonte: Elaboração Própria, a partir dos dados do Mapeamento Sistemático

Após a resolução das divergências e integração dos resultados, os estudos foram distribuídos entre as quatro faixas de qualidade. Por terem sido avaliados a partir de critérios diferentes, estudos empíricos só devem ter suas pontuações finais comparadas a de outro estudo empírico. O mesmo acontece entre os estudos secundários. Portanto, o resultado da avaliação da qualidade dos estudos é apresentado a partir de duas tabelas. A Tabela XII apresenta os resultados da avaliação dos 40 estudos empíricos incluídos na pesquisa e a Tabela XIII apresenta os resultados dos 3 estudos secundários. Apenas o resultado da força das evidências será apresentado reunindo a quantidade de estudos empíricos e secundários que dela fazem parte, de acordo com o estabelecido na Tabela XI. Nas Tabelas XII e XIII, apenas as referências dos identificadores dos estudos é indicada. Mais detalhes sobre eles podem ser encontrados no APÊNDICE B deste trabalho.

Tabela XII. Resultado da Avaliação da Qualidade dos Estudos Empíricos (Q2)

Referência	Pontuação	Força da Evidência	Quantidade de Estudos (%)
SM280	100%	Muito Alta	2% (1/43)
SM034	80%	Alta	48% (21/43)
SM142	80%		
SM346	80%		
SM493	80%		
SM121	70%		
SM125	70%		
SM197	70%		
SM378	70%		
SM382	70%		
SM392	70%		
SM428	70%		
SM511	70%		
SM516	70%		
SM528	70%		
SM534	70%		
SM103	60%		
SM124	60%		
SM129	60%		
SM369	60%		
SM541	60%		
SM544	60%		
SM105	50%	Média	34% (15/43)
SM119	50%		
SM152	50%		
SM185	50%		
SM400	50%		
SM426	50%		
SM543	50%		
SM016	40%		
SM043	40%		
SM262	40%		
SM329	40%		
SM573	40%		
SM040	30%		
SM430	30%		
SM506	30%		
SM260	20%	Baixa	6% (3/43)
SM060	10%		
SM549	10%		

Fonte: Elaboração Própria, a partir dos dados do Mapeamento Sistemático

Tabela XIII. Resultado da Avaliação da Qualidade dos Estudos Secundários (Q2)

Referência	Pontuação	Força da Evidência	Quantidade (%)
SM206	3,5 (87,5%)	Muito Alta	2% (1/43)
SM140	2,5 (62,5%)	Alta	2% (1/43)
SM023	1 (25%)	Baixa	2% (1/43)

Fonte: Elaboração Própria, a partir dos dados do Mapeamento Sistemático

Como pode ser observado nas tabelas, no total, incluindo estudos empíricos e secundários, 9% (4/43) estudos foram considerados de Baixa qualidade, 34% (15/43) tiveram sua qualidade avaliada como Média, 51% (22/43) foram considerados de Alta qualidade e 4% (2/43) de qualidade Muito Alta. Portanto, a maioria dos trabalhos (90% ou 39/43) foi avaliada como possuindo uma qualidade entre Média e Muito Alta, sendo 55% (24/43) deles considerados de qualidade Alta ou Muito Alta.

A maior qualidade de um estudo está relacionada à sua importância e força em apoiar as evidências. Sendo assim, os estudos que fornecem as mais fortes evidências deste trabalho são, respectivamente: [SM280], [SM206], [SM034], [SM142], [SM346], [SM493], [SM121], [SM125], [SM197], [SM378], [SM382], [SM392], [SM428], [SM511], [SM516], [SM528], [SM534], [SM140],[SM103], [SM124], [SM129], [SM369], [SM541], [SM544].

4.2.3. Desenho da Conclusão e Interpretação

A partir dos dados apresentados nas Seções 4.2.1 e 4.2.2, é possível concluir que:

- A Figura 8 indica que dentre os estudos incluídos apenas 18% (8/43) realmente são focados no estudo de algum aspecto do auto-gerenciamento de equipes de software. Isso ocorreu pois, mesmo não tendo o tema como objetivo de estudo, alguns trabalhos abordam o tema e respondem a questões desta pesquisa, seguindo os critérios de inclusão e exclusão pré-estabelecidos. Porém, o fato de ter incluído 35 estudos que não são focados no auto-gerenciamento e apenas o abordam pode dar a falsa impressão de que o tema investigado já é amadurecido na área. Esse pensamento não parece ser verdadeiro uma vez que grande parte das evidências não se aprofunda no tema investigado e aponta para outros estudos também incluídos na pesquisa ou estudos sobre o auto-gerenciamento de equipes que não são de desenvolvimento de software, o

que gera informações repetidas. Isso indica que, apesar do alto número de estudos incluídos nesta pesquisa, e, portanto, da alta quantidade de estudos que abordam e respondem de alguma maneira questões do auto-gerenciamento de equipes de software, ainda há escassez de estudos que abordem com profundidade o tema investigado. Essas informações repetidas podem ser observadas nas transcrições apresentadas nas Seções 4.2.2.1, 4.2.2.2 e 4.2.2.3.

- A Figura 8 também indica que a maioria desses estudos foi direcionada a algum aspecto de metodologias ágeis e, analisados internamente, outros 2 estudos mesmo não sendo direcionados a aspectos de metodologias ágeis também os abordaram. Isso indica que desde a divulgação das metodologias ágeis, em 2001, o auto-gerenciamento vem sendo fortemente relacionado ao uso dessas metodologias, que são apontadas como favorecedoras do auto-gerenciamento de equipes no desenvolvimento de software. Além disso, de acordo com o mapeamento das metodologias utilizadas para apoiar o auto-gerenciamento, apresentado na Seção 4.2.1, dentre as metodologias identificadas, todas são Ágeis. O que indica que, apesar da menor quantidade, alguns estudos abordam o auto-gerenciamento puro, sem quaisquer métodos utilizados para suporte ao auto-gerenciamento de equipes de software. Esse talvez seja um fator importante para a compreensão dos resultados encontrados para as questões desta pesquisa.

- A Tabela VI visa a resposta à questão específica Q1.1 desta pesquisa. Para isso, indica que, de acordo com os estudos incluídos nesta pesquisa, equipes de software auto-gerenciadas podem ser definidas como equipes que possuem plena autonomia e autoridade compartilhada para tomar as decisões, gerenciar sua carga de trabalho, estabelecer suas próprias metas e participar da definição das regras dos projetos. Assim, o auto-gerenciamento forma equipes com liderança distribuída e responsabilidade compartilhada entre os membros. Equipes de software auto-gerenciadas não são equipes de especialistas, onde cada indivíduo desempenha sua função específica. Equipes auto-gerenciadas são equipes possuidoras de funções redundantes que estão em constante interação e compartilham todas as informações sobre o projeto. Com isso, essas equipes estão sempre recebendo *feedback* sobre o que fazem, estão em constante aprendizado e se adaptam facilmente a situações de mudanças no projeto. Essas equipes possuem indivíduos mutuamente interdependentes que devem confiar uns nos outros e ter objetivos comuns.

- A Tabela VII visa a resposta à questão específica Q1.2 desta pesquisa e indica que equipes de software auto-gerenciadas podem contribuir para mudanças de atitudes e comportamentos dos indivíduos, como a maior consciência de suas responsabilidades, maior dedicação e disposição para colaborar com a equipe, maior adaptabilidade a mudanças, aumento da criatividade e da inovação e melhorias na comunicação, formando equipes mais coesas. Além disso, foi identificado que membros de equipes de software auto-gerenciadas tiveram aumento de sensações positivas, maior vínculo emocional, redução do estresse no trabalho, maior confiança na gestão, menor absenteísmo, menor rotatividade, maior motivação e satisfação com seu trabalho. Essas mudanças de atitudes e comportamentos dos indivíduos também podem explicar outros achados referentes à efetividade das equipes auto-gerenciadas. Apesar de ter sido mencionada em poucos estudos, a efetividade foi bem representada por algumas de suas dimensões, como a rapidez na resolução de problemas e o então aumento da produtividade, o aumento da qualidade, eficácia, desempenho e a redução de custos à organização. Dentre os resultados encontrados nos estudos incluídos nesta pesquisa, apenas um indicou um efeito do auto-gerenciamento que não é positivo, mas também não é negativo. Um estudo afirmou que o auto-gerenciamento das equipes não trouxe um melhor retorno do investimento à organização.

- A Tabela VIII, a Tabela IX e a Tabela X objetivam o apoio à resposta da questão específica Q1.3 e apresentam todos os fatores que podem afetar o trabalho de equipes auto-gerenciadas sobre o aspecto técnico, humano e organizacional. No total, foram encontrados 93 fatores que podem afetar o trabalho de equipes auto-gerenciadas, sendo 22 Fatores Técnicos, 31 Fatores Humanos e 40 Fatores Organizacionais. Devido a essa grande quantidade de fatores identificados, podemos concluir que a implementação de equipes de software auto-gerenciadas é difícil e deve ser cuidadosa, pois se esses fatores não forem usados adequadamente, as equipes não terão os efeitos esperados do auto-gerenciamento, como os vistos no parágrafo acima. O aspecto organizacional foi o que teve maior freqüência de abordagem entre os estudos incluídos. Isso pode indicar que a organização exerce uma grande influência sobre os resultados alcançados por uma equipe auto-gerenciada. Os principais fatores apontados por estes estudos, ou seja, os que foram mais freqüentemente abordados nos estudos foram: a concessão de autoridade à equipe (Fator Organizacional), com 55% dos 43 estudos

incluídos; as reuniões do processo (Fator Técnico), com 41% dos estudos; o estilo de gerência facilitador e colaborador (Fator Organizacional), com 39% dos estudos; as iterações e incrementos do processo (Fator Técnico), com 37%; a capacidade que um indivíduo tem de ser multifuncional (Fator Humano), com 34%; o *feedback* concedido à equipe pelos *stakeholders*[6] (Fator Organizacional), com 34%.

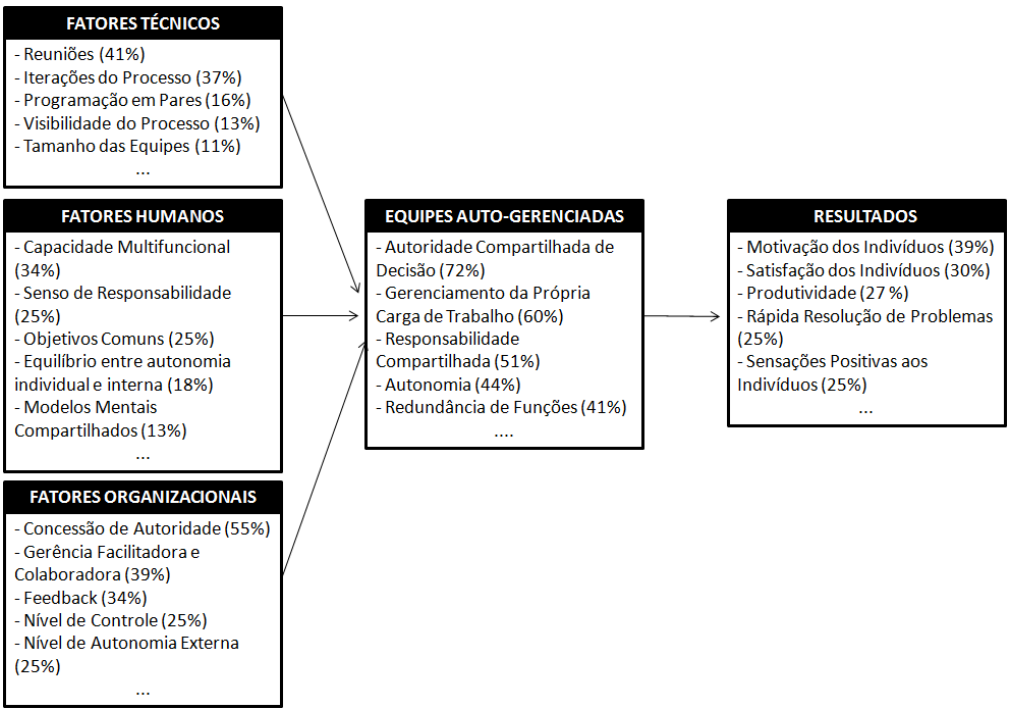
- A Tabela XII e a Tabela XIII apresentam, respectivamente, o resultado da avaliação da qualidade de estudos empíricos e o de estudos secundários. O procedimento de avaliação da qualidade dos 43 estudos incluídos nesta pesquisa foi realizado para responder à questão central de pesquisa Q2, onde é questionada a força das evidências encontradas. Mais da metade dos estudos encontrados (55%) apresentou uma qualidade considerada entre Alta e Muito Alta e 90% dos 43 estudos apresentou qualidade entre Média e Muito Alta. Isso significa que as evidências aqui encontradas possuem força suficiente para apoiar as conclusões.

A partir dos dados e das observações que apóiam as questões específicas Q1.1, Q1.2 e Q1.3 desta pesquisa, é possível o desenho da resposta à questão central Q1, que questiona o que se sabe sobre o auto-gerenciamento de equipes de desenvolvimento de software. Como é possível perceber nas tabelas, uma quantidade de dados superior à esperada foi encontrada. Porém, no detalhamento dos itens de cada categoria identificada foi possível notar que esses itens muitas vezes não foram abordados com mais profundidade nos estudos. Além disso, como já mencionado no início desta seção, houve uma enorme quantidade de frases e citações repetidas em diferentes estudos, indicando que na literatura identificada ainda há escassez de estudos que abordem com profundidade o tema sobre o auto-gerenciamento de equipes de software.

Apesar disso, as evidências encontradas nas questões específicas permitem que seja desenhado um modelo com os principais itens de cada categoria que pode ajudar indivíduos pertencentes a organizações de software que adotam o conceito do auto-gerenciamento, além de ser um resultado científico útil para o melhor entendimento da área temática. A Figura 9 resume os achados que apóiam a resposta à questão central Q1.

[6] Stakeholders: partes interessadas ou pessoas envolvidas no projeto.

Figura 9. Resumo dos principais itens encontrados na Q1.1, Q1.2 e Q1.3



Fonte: Elaboração Própria, a partir dos dados do Mapeamento Sistemático

4.3. Discussão

Os resultados encontrados nesta pesquisa mostraram-se consistentes com os resultados encontrados nos trabalhos relacionados apresentados no Capitulo 2, Seção 2.2, e com outros trabalhos de autores reconhecidos na literatura da auto-gestão.

Dentre os trabalhos relacionados apresentados anteriormente, os intitulados “Understanding Selforganizing Teams in Agile Software Development” (MOE et al., 2008) e “Overcoming Barriers to Self-Management in Software Teams” (MOE et al., 2009) estão diretamente relacionados à questão específica Q1.3 e abordam as principais barreiras para a auto-gestão de equipes. O primeiro encontrou como principais barreiras “habilidades altamente especializadas” e a “redução da autonomia externa”. O segundo desses estudos encontrou que as barreiras estão relacionadas ao “compromisso individual”, à “incapacidade de aprendizado” e à “liderança individual”, além de “recursos compartilhados”, “controle organizacional” e “cultura de especialistas”.

Todas essas barreiras estiveram presentes também nesta pesquisa, sendo, a maioria, também considerados aqui como principais fatores que podem afetar o auto-gerenciamento das equipes, como pode ser visto na Seção 4.2.2.3. As barreiras de “habilidades altamente especializadas” e “cultura de especialistas” nesta pesquisa estão relacionadas à “Capacidade Multifuncional”, considerada aqui o principal fator humano que pode afetar as equipes, com 34% dos estudos incluídos, e à “Cultura de Especialistas/Generalistas”, considerada aqui o 7º principal fator organizacional, com 18% dos estudos incluídos. A “redução da autonomia externa” foi considerada um dos 5 principais fatores organizacionais, sendo representada por “Nível de Autonomia Externa” presente em 25% dos estudos incluídos. O “compromisso individual” nesta pesquisa está relacionado a “Objetivos Comuns” entre equipe e indivíduo, aqui indicado como o segundo principal fator humano, com 25% dos estudos incluídos. O “controle organizacional” também foi identificado aqui como um fator organizacional representado por “Nível de Controle” com 25% dos estudos. Ou seja, esses fatores foram considerados principais nos dois trabalhos relacionados e também nesta pesquisa.

Outras barreiras encontradas nesses trabalhos também foram identificadas nesta pesquisa, porém com menor importância se comparados a outros fatores. Essas barreiras foram: a “incapacidade de aprendizado” e os “recursos compartilhados”, representados respectivamente em fatores humanos, com 9% dos estudos, e fatores organizacionais, com 4% dos estudos incluídos. A “liderança individual” foi encontrada aqui como uma característica utilizada para definir o auto-gerenciamento em equipes de software, representado por “Liderança Distribuída” com 20% dos estudos incluídos, como apresentado na Seção 4.2.2.1.

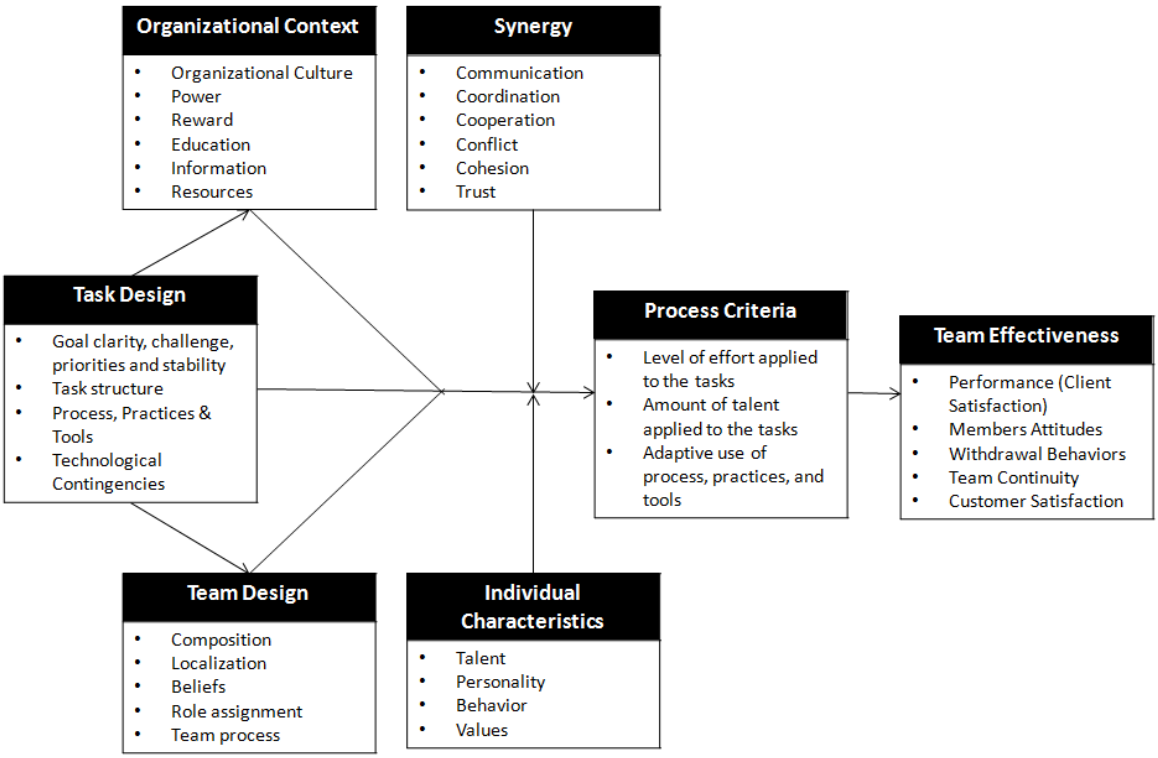
Os resultados desta pesquisa também foram comparados a outros trabalhos de autores reconhecidos na literatura da auto-gestão, como Hackman (1987) e Cohen (1993). Cada item dos modelos de efetividade de equipes auto-gerenciadas propostos por esses autores foi comparado a cada item encontrado no modelo proposto por esta pesquisa. Após essas comparações, foram reunidos modelos propostos por outros trabalhos, como o de Yeatts & Hyten (1998) e algumas pesquisas do grupo HASE para uma nova comparação que objetivou a construção de um novo modelo de efetividade de equipes auto-gerenciadas. A Figura 10 apresenta este modelo, escrito em inglês para evitar incoerências com os trabalhos no qual foi derivado.

A partir deste novo modelo, os resultados encontrados nesta pesquisa foram novamente comparados e adequados a cada item/grupo correspondente. Esta comparação permitiu

entender que alguns tópicos encontrados no novo modelo proveniente de Hackman (1987), Cohen (1993) e Yeatts & Hyten (1998) não puderam ser identificados nos resultados desta pesquisa e alguns itens identificados nesta pesquisa não tiveram seus correspondentes tópicos/grupos encontrados no novo modelo. Porém, a grande maioria dos itens aqui encontrados pôde ser enquadrada nos tópicos e grupos do novo modelo. O resultado dessa reunião e comparação entre os grupos e tópicos do modelo com os itens identificados nesta pesquisa é apresentado separadamente de acordo com os 7 grupos do modelo: “Task Design” (Figura 11), “Team Design” (Figura 12), “Individual Characteristics” (Figura 13), “Organizational Context” (Figura 14), “Synergy” (Figura 15), “Process Criteria” (Figura 16) e “Team Effectiveness” (Figura 17). Nas Figuras também é apresentada a quantidade de estudos incluídos nesta pesquisa que abordam cada item de resultado encontrado.

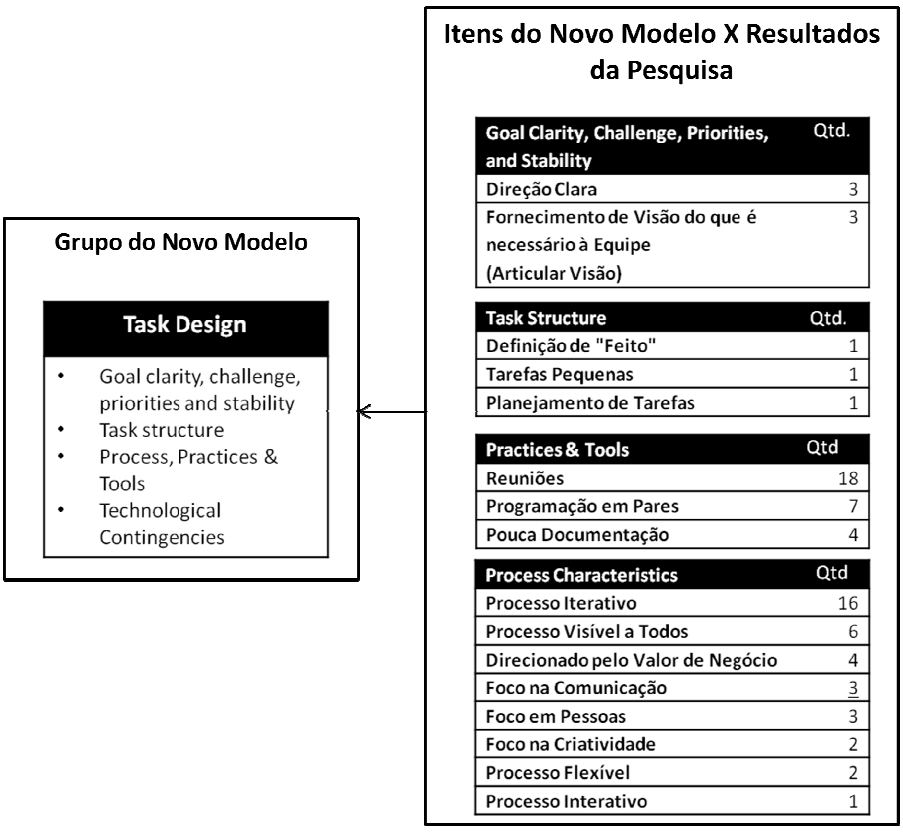
Essas comparações permitem entender que, em geral, os resultados encontrados nesta pesquisa são consistentes com os encontrados em outras pesquisas que focam no tema investigado.

Figura 10. Novo Modelo de Efetividade de Equipes Auto-gerenciadas Baseado em Hackman (1987), Cohen (1993) e Yeatts & Hyten (1998)



Fonte: SILVA et al. (in press)

Figura 11. Distribuição dos Resultados sobre o grupo "Task Design"



Goal Clarity, Challenge, Priorities, and Stability	Qtd.
Direção Clara	3
Fornecimento de Visão do que é necessário à Equipe (Articular Visão)	3

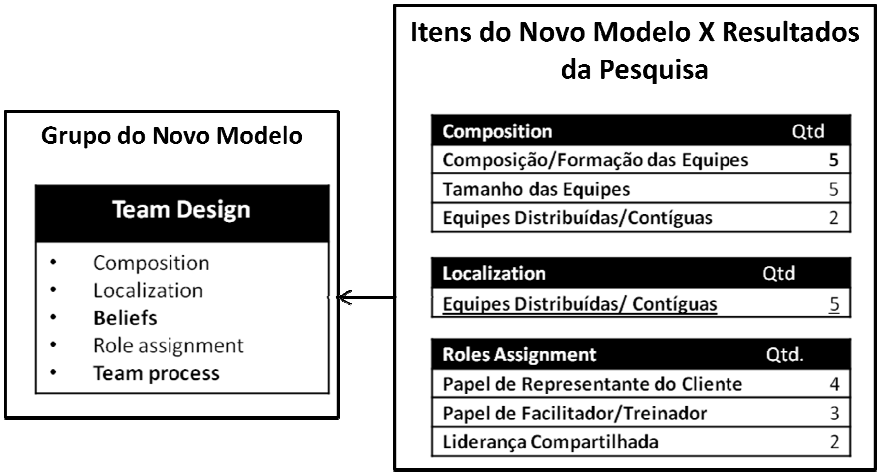
Task Structure	Qtd.
Definição de "Feito"	1
Tarefas Pequenas	1
Planejamento de Tarefas	1

Practices & Tools	Qtd
Reuniões	18
Programação em Pares	7
Pouca Documentação	4

Process Characteristics	Qtd
Processo Iterativo	16
Processo Visível a Todos	6
Direcionado pelo Valor de Negócio	4
Foco na Comunicação	3
Foco em Pessoas	3
Foco na Criatividade	2
Processo Flexível	2
Processo Interativo	1

Fonte: Elaboração Própria, a partir dos dados do Mapeamento

Figura 12. Distribuição dos Resultados sobre o grupo "Team Design"



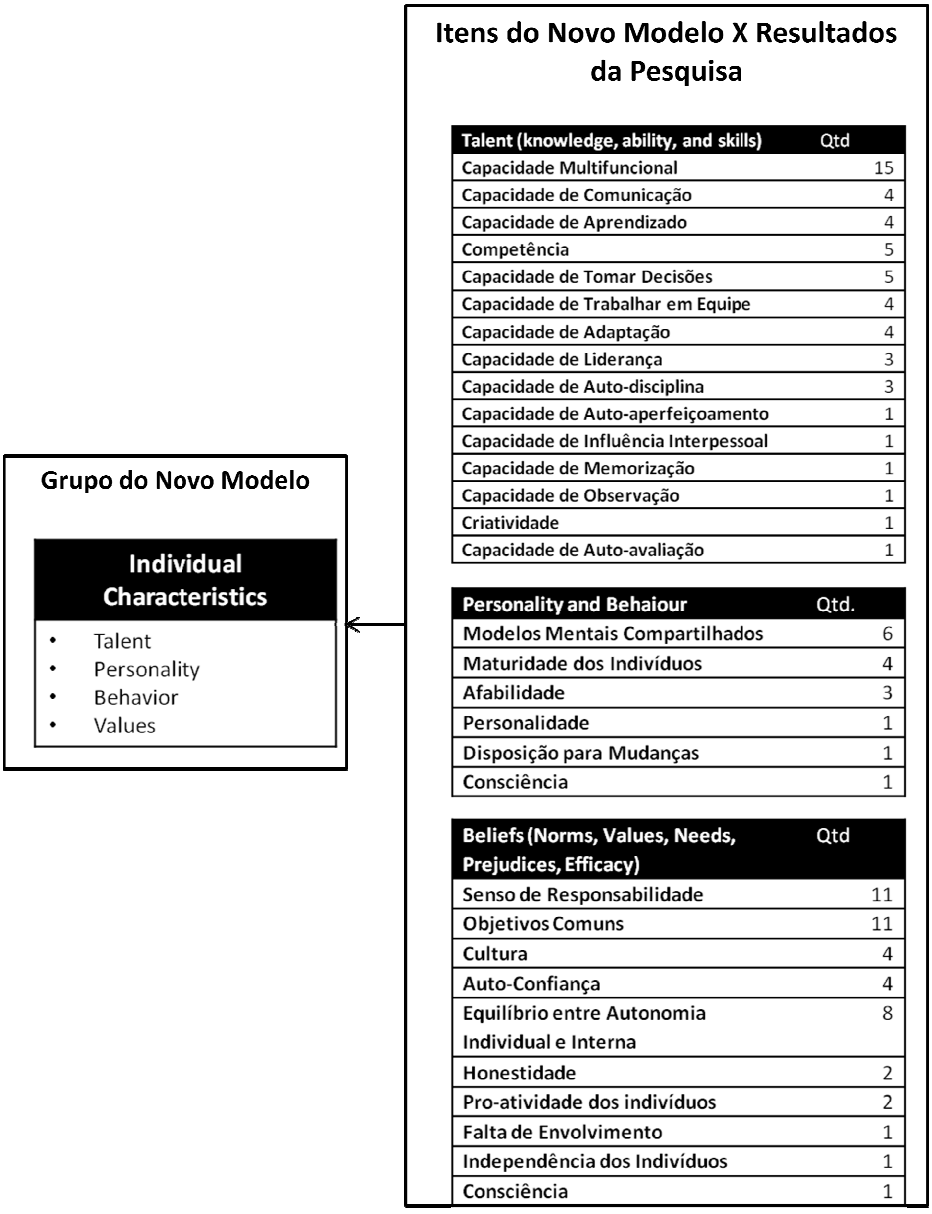
Composition	Qtd
Composição/Formação das Equipes	5
Tamanho das Equipes	5
Equipes Distribuídas/Contíguas	2

Localization	Qtd
Equipes Distribuídas/ Contíguas	5

Roles Assignment	Qtd.
Papel de Representante do Cliente	4
Papel de Facilitador/Treinador	3
Liderança Compartilhada	2

Fonte: Elaboração Própria, a partir dos dados do Mapeamento

Figura 13. Distribuição dos Resultados sobre o grupo "Individual Characteristics"



Itens do Novo Modelo X Resultados da Pesquisa

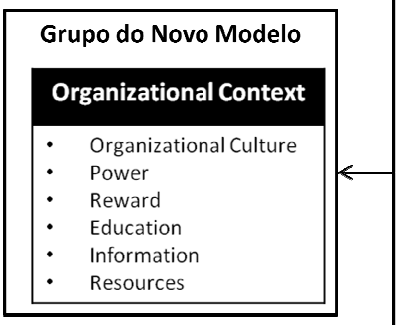
Talent (knowledge, ability, and skills)	Qtd
Capacidade Multifuncional	15
Capacidade de Comunicação	4
Capacidade de Aprendizado	4
Competência	5
Capacidade de Tomar Decisões	5
Capacidade de Trabalhar em Equipe	4
Capacidade de Adaptação	4
Capacidade de Liderança	3
Capacidade de Auto-disciplina	3
Capacidade de Auto-aperfeiçoamento	1
Capacidade de Influência Interpessoal	1
Capacidade de Memorização	1
Capacidade de Observação	1
Criatividade	1
Capacidade de Auto-avaliação	1

Personality and Behaiour	Qtd.
Modelos Mentais Compartilhados	6
Maturidade dos Indivíduos	4
Afabilidade	3
Personalidade	1
Disposição para Mudanças	1
Consciência	1

Beliefs (Norms, Values, Needs, Prejudices, Efficacy)	Qtd
Senso de Responsabilidade	11
Objetivos Comuns	11
Cultura	4
Auto-Confiança	4
Equilíbrio entre Autonomia Individual e Interna	8
Honestidade	2
Pro-atividade dos indivíduos	2
Falta de Envolvimento	1
Independência dos Indivíduos	1
Consciência	1

Fonte: Elaboração Própria, a partir dos dados do Mapeamento

Figura 14. Distribuição dos Resultados sobre o grupo "Organizational Context"



Itens do Novo Modelo X Resultados da Pesquisa

Organizational Culture	Qtd
Gerência Facilitadora e Colaboradora	17
Nível de Controle	11
Sistema Formal de Apoio da Alta Administração	9
Liderança	9
Cultura de Especialistas/Generalistas	8
Crença nos Indivíduos e Equipes	8
Regras Organizacionais	4
Supervisão	4
Respeito à Equipe	3
Relação da Alta gerencia com a Equipe	2
Exigências no Trabalho	2
Flexibilidade de Trabalho	2

Power	Qtd
Concessão de Autoridade	24
Nível de Autonomia Externa	11
Concessão de Livre Autonomia	6
Hierarquia	5
Concessão de Responsabilidade	4
Concessão de Liberdade	4
Gestão descentralizada	2
Controle Organizacional	1
Concessão de Influência à Equipe	1

Rewards	Qtd.
Sistema de Recompensa	2

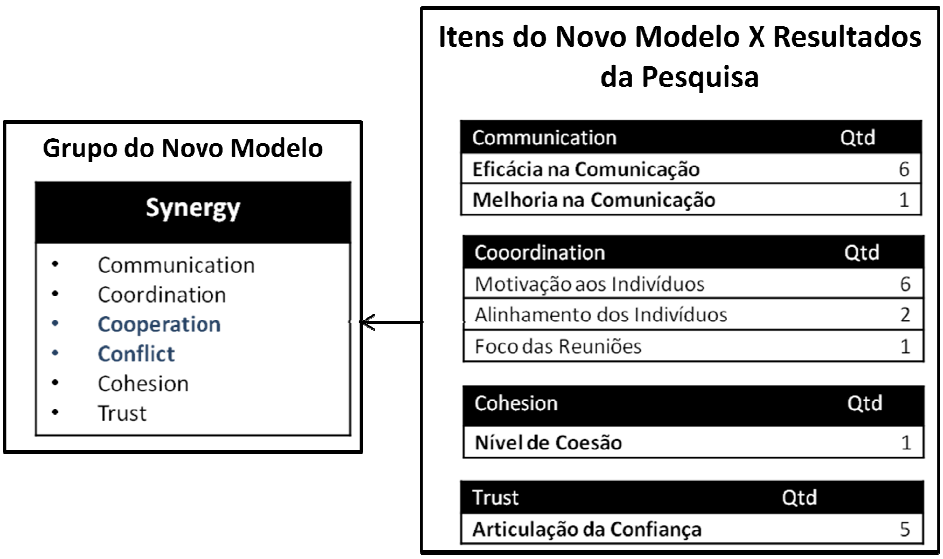
Education	Qtd.
Treinamentos para a Equipe	4

Information	Qtd.
Feedback	15
Compartilhamento de Informações	9
Intercâmbio Regular de Idéias	1
Envolvimento do Cliente no Projeto	5

Resources	Qtd.
Estrutura de Espaço Aberto	9
Recursos Disponíveis	9
Recursos Compartilhados	2

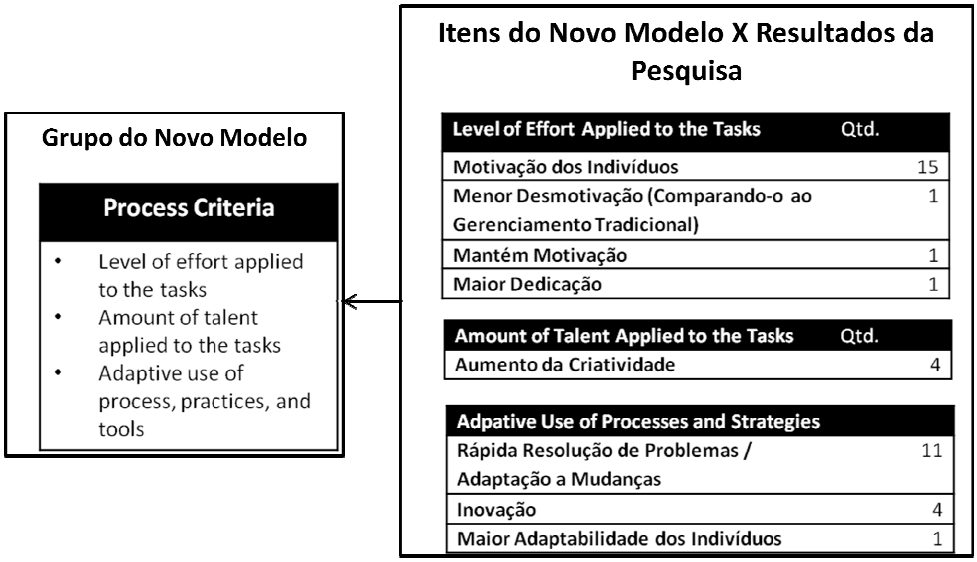
Fonte: Elaboração Própria, a partir dos dados do Mapeamento

Figura 15. Distribuição dos Resultados sobre o grupo "Synergy"



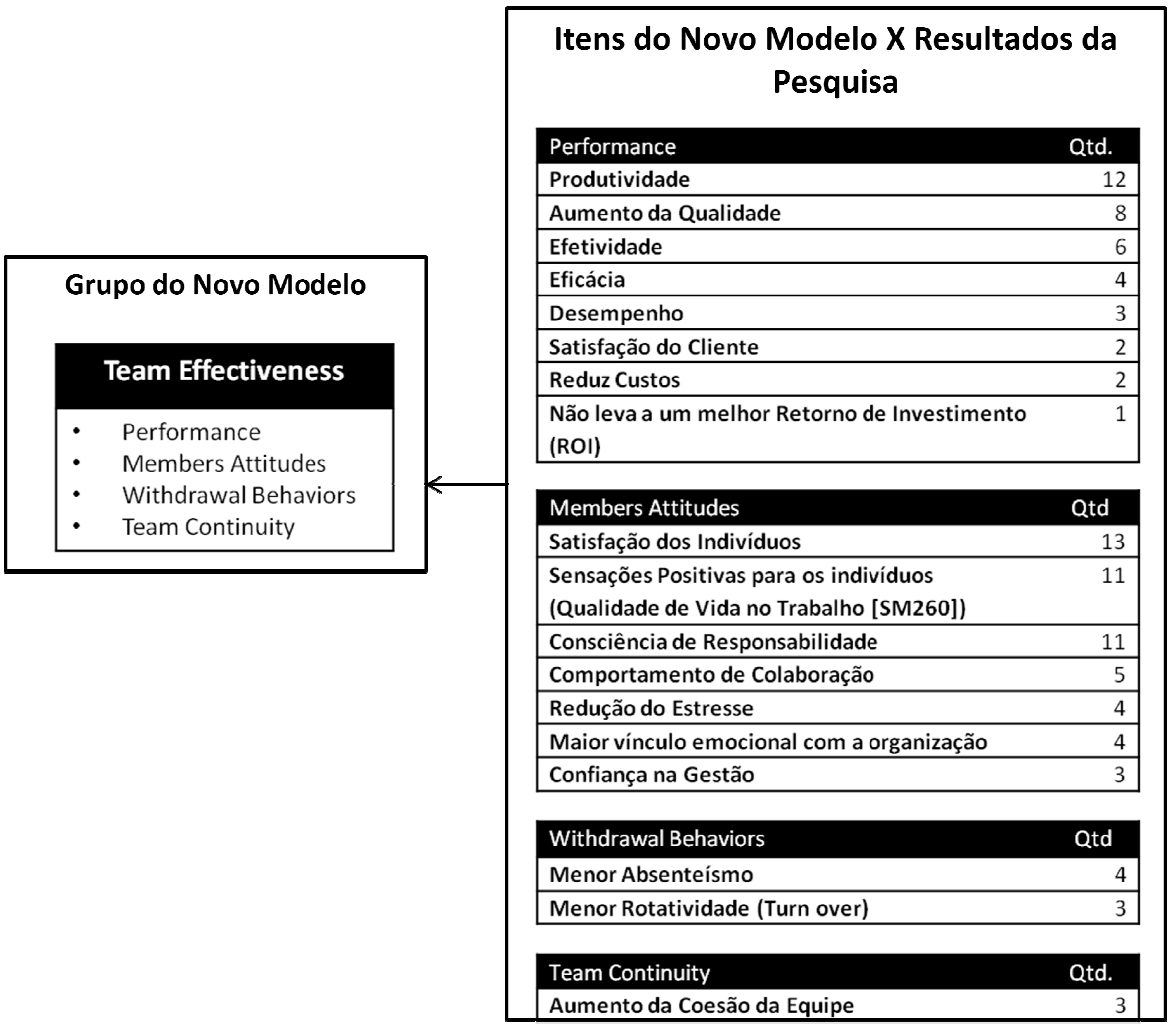
Fonte: Elaboração Própria, a partir dos dados do Mapeamento

Figura 16. Distribuição dos Resultados sobre o grupo "Process Criteria"



Fonte: Elaboração Própria, a partir dos dados do Mapeamento

Figura 17. Distribuição dos Resultados sobre o grupo "Team Effectiveness"



Itens do Novo Modelo X Resultados da Pesquisa

Performance	Qtd.
Produtividade	12
Aumento da Qualidade	8
Efetividade	6
Eficácia	4
Desempenho	3
Satisfação do Cliente	2
Reduz Custos	2
Não leva a um melhor Retorno de Investimento (ROI)	1

Members Attitudes	Qtd
Satisfação dos Indivíduos	13
Sensações Positivas para os indivíduos (Qualidade de Vida no Trabalho [SM260])	11
Consciência de Responsabilidade	11
Comportamento de Colaboração	5
Redução do Estresse	4
Maior vínculo emocional com a organização	4
Confiança na Gestão	3

Withdrawal Behaviors	Qtd
Menor Absenteísmo	4
Menor Rotatividade (Turn over)	3

Team Continuity	Qtd.
Aumento da Coesão da Equipe	3

Fonte: Elaboração Própria, a partir dos dados do Mapeamento

5. Considerações Finais

Esta pesquisa investigou o conceito de auto-gerenciamento de equipes no contexto do desenvolvimento de software. Surgido em contraponto ao gerenciamento tradicional, o conceito de equipes auto-gerenciadas teve início na literatura em 1950 e desde então vem sendo desenvolvido por pesquisadores e cada vez mais aplicado nas organizações. A popularidade do tema é apoiada por diversas pesquisas que sugerem que o auto-gerenciamento promove maior efetividade, com melhorias na qualidade do produto, na produtividade da equipe e na Qualidade de Vida no Trabalho. Porém, muitos pesquisadores apontam inconsistências nesses resultados.

Em contraste com esses resultados, pesquisas também sugerem que o auto-gerenciamento não aumentou todos os tipos de eficácia para todos os tipos de trabalho. Isso significa que os resultados podem variar consideravelmente por fatores como as particularidades de cada tipo de trabalho, como as encontradas no contexto de desenvolvimento de software.

Tendo isso em vista, esta pesquisa teve por objetivo entender o conhecimento existente na literatura sobre o auto-gerenciamento de equipes de desenvolvimento de software e qual a força das evidências encontradas. Para isso, foi identificado, analisado e reunido sistematicamente o conhecimento encontrado na literatura sobre o tema, por meio de um Mapeamento Sistemático da Literatura.

Após as buscas automáticas e manuais, 2.646 estudos foram retornados e posteriormente analisados, resultando em 43 estudos incluídos na pesquisa. Esses estudos permitiram a identificação de 25 características definidoras de uma equipe de software auto-gerenciadas, 28 resultados que equipes de software auto-gerenciadas podem promover e 93 fatores que podem afetar positivamente ou negativamente o trabalho dessas equipes, sendo distribuídos entre fatores técnicos, fatores humanos e fatores organizacionais.

No entanto, apesar da quantidade considerável de estudos incluídos que apóiam a resposta às questões desta pesquisa, apenas 18% (8/43) são focados no tema investigado. Talvez isso explique o fato de os itens das categorias identificadas não terem sido muito explorados e de que muitas das evidências apontavam para outros trabalhos, citados repetidas vezes em diferentes estudos incluídos na pesquisa. Isso significa que, na literatura identificada, ainda há

escassez de estudos que abordem com profundidade o tema sobre o auto-gerenciamento de equipes de software.

Apesar disso, os resultados encontrados foram consistentes com a literatura existente e a grande maioria (90%) dos estudos incluídos nesta pesquisa teve suas evidências consideradas de força média a muito alta, a partir da avaliação da qualidade de cada estudo. As evidências aqui encontradas também permitem concluir que o auto-gerenciamento de equipes de software possui diversas particularidades que necessitam de cuidado e administração. Isso torna o auto-gerenciamento difícil de ser implementado, com o risco de ser comprometido e, assim, prejudicar os resultados de desempenho, caso essas particularidades não sejam bem administradas.

As principais contribuições deste trabalho foram a melhor compreensão a respeito do que se sabe sobre auto-gerenciamento de equipes de software, o conhecimento dos resultados que essas equipes podem promover e o conhecimento dos fatores que podem afetar as equipes e comprometer esses resultados.

5.1. Limitações e Ameaças à Validade

Apesar da atenção dada aos procedimentos metodológicos descritos no Capítulo 3, a pesquisa conta com algumas limitações que podem por em risco sua validade.

De acordo com Dyba et al. (2008), as principais limitações do método são o possível viés introduzido pelos pesquisadores na seleção dos estudos e a imprecisão na extração dos dados. A fim de tentar garantir imparcialidade no processo de seleção dos estudos, o protocolo do mapeamento foi definido e detalhado para servir como base para que os pesquisadores pudessem iniciar as etapas da pesquisa de forma padronizada, como apresentado no APÊNDICE A. Ainda na tentativa de maximizar a seleção imparcial, os pesquisadores documentaram suas razões para a inclusão/exclusão dos estudos em cada etapa da pesquisa, de acordo com o sugerido por Kitchenham e Charters (2007). Também visando minimizar o viés, como já apresentado na Seção 4.1.2 do Capítulo 4, todas as etapas da pesquisa, exceto a extração dos dados, foram realizadas por, pelo menos, dois pesquisadores. Nos casos em que houve conflito entre os dois pesquisadores responsáveis por cada estudo, foram realizadas reuniões para resolução, e, quando necessário, avaliação de um terceiro pesquisador.

Outra importante limitação de um Mapeamento Sistemático é a cobertura limitada de estudos para a área de investigação, onde, na construção da String de Busca, com a identificação das palavras-chave e sinônimos que a constituem, existe o risco de omissão de estudos relevantes para a pesquisa. Na tentativa de ampliar a cobertura dos estudos existentes, foi feita uma busca automática em 5 engenhos de busca, número considerado suficiente para uma cobertura aceitável (KITCHENHAN, 2007), e uma busca manual em 11 fontes diferentes, incluindo as principais revistas e anais dos principais congressos, como já descrito na Seção 4.1.1.

O período em que a pesquisa foi realizada, englobando estudos publicados até o ano de 2010, também pode ser considerado uma limitação. Estudos que poderiam ser relevantes para a pesquisa foram identificados no processo de busca informal, apresentado na Seção 4.1.1 do Capítulo 4. Porém, esses estudos não puderam ser incluídos na pesquisa, pois foram publicados em 2011, ano em que a pesquisa já estava em avançado nível de desenvolvimento e não cabia o retorno à etapa de coleta de dados. Apesar dessa possível limitação, foi possível o recolhimento de uma boa quantidade de estudos relevantes e grande quantidade de dados que apoiam a pesquisa.

5.2. Recomendações para Trabalhos Futuros

Sabendo-se que a relevância de um trabalho também pode ser avaliada pelas oportunidades de trabalhos futuros que ele provê, a seguir são propostos alguns direcionamentos para novas pesquisas identificados a partir deste trabalho:

- **Ampliação e atualização da base de dados:** um dos possíveis trabalhos a serem realizados a partir deste estudo é a extensão da pesquisa para inclusão dos anos seguintes ao ano a que a pesquisa foi limitada (2010). Nesta pesquisa de extensão também pode ser interessante mapear quais e quantas evidências estão partindo de conclusões do próprio estudo incluído, de inferência do autor ou de conclusões de terceiros apenas citadas no estudo. Isso pode permitir a identificação e explicação mais exata do motivo de termos encontrado tantas citações idênticas repetidas entre diferentes trabalhos, além de poder diminuir o erro na inferência quando a extração, análise e síntese dos dados é feita somente por um pesquisador. Além disso, uma extensão também pode ampliar o estudo sobre cada resultado encontrado, em se tratando das características do auto-gerenciamento, dos efeitos que o auto-

gerenciamento promove e dos fatores que podem afetar o trabalho das equipes. Isso traria um melhor entendimento sobre o por que de cada um desses itens ser mencionado nos estudos incluídos e também possibilitaria identificar, separar e analisar os itens que não são pertencentes unicamente a equipes auto-gerenciadas, mas também podem ser pertencentes a outros tipos de equipe, como Catedral e Bazar (MIRANDA, 2011).

- **Pesquisar a prática nas organizações:** outro possível direcionamento a um futuro trabalho é como ocorre a prática do auto-gerenciamento de equipes de software. Esse estudo pode ser realizado a partir de uma pesquisa qualitativa, onde devem ser analisadas as características práticas dessas equipes, os resultados que elas podem ter gerado e os fatores que podem ter afetado essas equipes, para, em seguida, comparar aos resultados encontrados aqui. Além disso, pode ser feito também um estudo comparativo com equipes gerenciadas tradicionalmente, um estudo com equipes que já foram auto-gerenciadas e passaram a ser tradicionalmente gerenciadas e também o contrário, equipes tradicionais que passaram a ser auto-gerenciadas.

5.3. Conclusões

Uma quantidade considerável de estudos foi incluída na pesquisa por abordarem o tema investigado e responderem ao menos uma das questões da pesquisa, além de se enquadrarem nos outros critérios de inclusão, apresentados na Seção 3.3.3. Porém, apenas 18% dos estudos incluídos foram elaborados com o objetivo de investigar algum aspecto do auto-gerenciamento de equipes de desenvolvimento de software. Isso explica a grande quantidade de estudos que apontam para outros estudos quando abordam o tema e indica que, na literatura identificada, ainda há escassez de estudos que abordem com profundidade o tema sobre o auto-gerenciamento de equipes de software.

Apesar disso, foi possível o desenho de um modelo que indica o conhecimento existente sobre o auto-gerenciamento de equipes de software, no que se refere aos fatores que afetam o trabalho das equipes, as características que definem o auto-gerenciamento de equipes de software e os efeitos que podem ser obtidos através dessas equipes. Além disso, os resultados encontrados são consistentes com os achados em outras pesquisas.

REFERÊNCIAS

AGILE MANIFESTO, 2001. Disponível em: </http://www.agilemanifesto.org/S>

ARKSEY, H.; O'MALLEY, L. Scoping studies: towards a methodological framework. International Journal of Social Research Methodology, v. 8, n. 1, p. 19-32, 2005. Disponível em: <http://www.informaworld.com/openurl?genre=article&doi=10.1080/1364557032000119616&magic=crossref||D404A21C5BB053405B1A640AFFD44AE3>. .

ASTON, J.; LAROCHE, L.; MESZAROS, G. Cowboys and Indians: Impacts of cultural diversity on agile teams. *AGILE '08:* 423–428, IEEE Computer Society, Washington, 2008.

ATTARAN, M.; NGUYEN, T. T. Creating the Right Structural Fit for Self-Directed Teams. Team Performance Management, 2000.

BANKER, R.; FIELD, J.; SCHROEDER, R; SINHA, K. Impact of work teams on manufacturing performance: A longitudinal field study. Academy of Management Journal, 1996.

BAILEY, J.; BUDGEN, D.; TURNER, M.; ET AL.. Evidence relating to Object-Oriented software design: A survey. First International Symposium on Empirical Software Engineering and Measurement (ESEM 2007), p. 482-484. Ieee. doi: 10.1109/ESEM.2007.58, 2007.

BAILEY, J.; ZHANG, C.; BUDGEN, D.; TURNER, M.; CHARTERS, S. Search Engine Overlaps : Do they agree or disagree?, 2007.

BEECHAM, S., BADDOO, N., HALL, T., & ROBINSON, H. Protocol for a Systematic Literature Review of Motivation in Software Engineering Systematic Review – Cover Sheet. Computer, 2006.

BEEKUN, R. I. Assessing the effectiveness of sociotechnical interventions: Antidote or fad? Human Relations, 47(10), 877-897, 1989.

BEGEL, A; NAGAPPAN, N. Usage and perceptions of agile software development in an industrial context: An exploratory study. *ESEM '07:* 255–264, IEEE Computer Society, Washington, 2007.

BELBIN, M. Management Teams: Why they succeed or Fail. Butterworth-Heinemann Ltd, 1981.

BERGMANN, T. J.; DE MEUSE, K. Diagnosing \\Tiether an Organization Is Truly Ready to Empower Workers: A Case Study." Human Resource Planning, 1996.

BUDGEN, D.; TURNER, M.; BRERETON, P.; KITCHENHAM, B.; Using Mapping Studies in Software Engineering, 2007.

CARDOZO, E. S. F. ; ARAÚJO NETO, J. BENITO ; BARZA, A.; FRANCA, A. C.; DA SILVA, F. Q. B. SCRUM and Productivity in Software Projects : A Systematic Literature Review. EASE'2010. p.1-4, 2010.

CENTRE FOR REVIEWS AND DISSEMINATION. What are the criteria for the inclusion of reviews on DARE?, 2007 Disponível em: <http://www.york.ac.uk/inst/crd/faq4.htm>, (acessado em 17/09/2010).

COCKBURN, A; HIGHSMITH, J.; Agile software development: The people factor. *Computer*, 34(11):131–133, 2001.

COHEN, S. G. **Designing Effective Self-Managing Work Teams. Advances**, 1, 1993.

COHEN, SUSAN G, LEDFORD, G. E. J. **The effectiveness of self-managing teams: A quasi-experiment. Human Relations**, 47, 1994.

COHEN, S.G.; BAILEY, D.E. **What makes teams work: group effectiveness research from the shop floor to the executive suite**, Journal of Management 23 (3) 239–290, 1997.

COHEN, SUSAN G; CHANG, LEI; LEDFORD, GERALD. **A hierarchical construct of self-management leadership and its relationship to quality of work life and perceived work group effectiveness**, 1997.

CORDERY, J. L., MUELLER, W. S., & SMITH, L. M. Attitudinal an behavioral effects of autonomous group working: A longitudinal field study. Academy of Management Journal, 34(2), 1991.

CUMMINGS, T. G., MOLLOY, E. S., & GLEN, R. A methodological critique of fifty-eight selected work experiments. Human Relations, 30, 675-708, 1977.

DEMARCO, T.; LISTER, T. **Peopleware productive projects and teams**. 2. ed. New York, USA: Dorset House Publishing Co, 1999.

DEVARO, J., & EAST, I. H.; The Effect of Self-Managed and Closely-Managed Teams on Labor Productivity and Product Quality : An Empirical Analysis of a Cross Section of Establishments The Effects of Self-Managed and Closely-Managed Teams on Labor Productivity and Product Quality : A. Employee Relations, 2007.

DRUCKER, P.F., **The Landmarks of Tomorrow**, Harper, New York, NY, 1959.

DRUSKAT, V. U., & WHEELER, J. V.; Managing from the boundary. Washington D.C., August, 2001.

DRUSKAT, V. U. & WHEELER, J. V. Managing from the boundary: The effective leader- ship of self-managing work teams. Academy of Management Journal, 2003.

DYBÅ, T.; KITCHENHAM, B.A.; JØRGENSEN, M. Evidence-Based Software Engineering for Practitioners, IEEE Software, 22(1): 58–65, 2005.

DYBA, T.; DINGSØYR, T.; Empirical studies of agile software development : A systematic review. Information and Software Technology. doi: 10.1016/j.infsof.2008.01.006, 2008.

EASTERBROOK S; SINGER J; STOREY M-A; DAMIAN D. **Selecting Empirical Methods for Software Engineering Research**, 2008.

FRANÇA, A. C.; Um Estudo sobre Motivação em Integrantes de Equipes de Desenvolvimento de Software. Recife: Editora Universitária – UFPE, 2010.

FRANÇA, A. C. C.; GOUVEIA, T. B.; SANTOS, P. C. F.; SANTANA, C. A.; SILVA, F. Q. B. Motivation in Software Engineering : A Systematic Review Update. EASE'2011, p. 1-10, 2011.

GLASER B.; STRAUSS, A. L.; *The Discovery of Grounded Theory*. Aldine, Chicago, 1967.

GLASS, R., VESSEY, I. & RAMESH, V.; 'Research in software engineering: An analysis of the literature', Information & Software Technology 44, 491–506, 2002.

GLASS, R., RAMESH, V. & VESSEY, I.; 'An Analysis of Research in Computing Disciplines', Communications of the ACM 47, 89–94, 2004.

GOODMAN, P. S.; DEVADAS, R.; HUGHSON, T. L.; Groups and productivity: Analyzing the effectiveness of self-managing teams. Em J.P. Campbell, R.J. Campbell, & Associates

(Eds.), Productivity in organizations, pp. 295-325. San Francisco: Jossey-Bass & Associates, 1988.

GULOWSEN, J.; A measure of work group autonomy. Em L. E. Davis & J, C, Taylor (Eds.). Design of jabs: 374-390. Harmondsworth. U, K,: Penguin, 1972.

GUZZO R.; DICKSON, M.; "Teams in organizations: Recent research on performance and effectiveness". Annual review of psychology, vol. 47, no. 1, p. 307–338, 1996.

HACKMAN, J. R.; The psychology of self-management in organizations. In M. S. Pollack & R. O. Perlogg (Eds.), Psychology and work: Productivity change and employment (pp. 85–136). Washington, 1986.

HACKMAN, J. R. The design of work teams. In J. Lorsch (Ed.), Handbook of organizational behavior. Englewood Cliffs, NJ: Prentice-Hall, 1987.

HARRIS, T. E.; Toward Effective Employee Involvement: An Analysis of Parallel and Self-Managing Teams. Joumal of Allied Business Research, 1992.

HAMMER, M.; STANTON, S. A.; The Reengineering Revolution, 1995.

HIGHSMITH, J.; M. FOWLER, M.; The Agile Manifesto. *Software Development Magazine*, 9(8):29–30, 2001.

HODA, RASHINA; NOBLE, JAMES; MARSHALL, STUART; Balancing Acts: Walking the Agile Tightrope, 2010.

HOEPFL, M.C. Choosing Qualitative Research: **A Primer for Technology Education Researchers**. Journal of Technology Education. v. 9, n. 1, 1997.

JACINTO, S. Um Mapeamento Sistemático da Pesquisa sobre a Influência da Personalidade na Engenharia de Software Ciência. Dissertação de Mestrado. Universidade Federal de Pernambuco, Recife, PE, Brasil, 2010.

JEDLITSCHKA, ANDREAS; PFAHL, DIETMAR; Reporting guidelines for controlled experiments in software engineering. Proceedings of ACM/IEEE International Symposium on Empirical Software Engineering, 2005, p. 95-104.

JEDLITSCHKA, A.; CIOLKOWSKI, M.; Reporting Experiments in Software Engineering, 2008

KHAN, K.S.; TER RIET, G.; GLANVILLE, J.; SOWDEN, A.J.; KLEIJNEN, J.;. Undertaking Systematic Review of Research on Effectiveness. CRD Report Number 4 (Second Edition), NHS Centre for Reviews and Dissemination, University of York, UK, 2001.

KITCHENHAM, B.; DYBA,T.; JøRGENSEN,M.; Evidence-Based software engineering. Proceedings of the 26th International Conference on Software Engineering (ICSE). IEEE Computer Society, Los Alamitos, CA, 273–281, 2004.

KITCHENHAM, B.A.; DYBA, T.; JORGENSEN, M. Evidence-based software engineering, 2006

KITCHENHAM, B. Guidelines for performing Systematic Literature Reviews in Software Engineering. Vol 2.3 EBSE Technical Report, EBSE-2007-01, 2007.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering, Technical Report EBSE-2007-01, School of Computer Science and Mathematics, Keele University, 2007.

KITCHENHAM, B; PEARLBRERETON, O; BUDGEN, D; TURNER, M; BAILEY, J; LINKMAN, S.; Systematic literature reviews in software engineering – A systematic literature review, 2008

LUNA, SERGIO VASCONCELOS DE. Planejamento de pesquisa: uma introdução. São Paulo: EDUC, 1997.

MANZ, C. C.; SIMS, H. P., JR.; Leading self-managed groups: A conceptual analysis of a paradox. Economic and Industrial Democracy, 7, 141-165, 1986.

MANZ, C. C.; SIMS, H. P., JR; Business Without Bosses, John Wiley & Sons, New York, 1993.

MARCONI, M. A.; LAKATOS, E. M.; Metodologia Científica. 4. ed. São Paulo: Atlas, 2004.

MARCONI, M. A.; LAKATOS, E. M. **Metodologia Cientifica: Ciência e Conhecimento Científico**. Atlas, 2007.

MAYS, N.; ROBERTS, E.; POPAY, J.; Synthesising research evidence. Em N. Fulop, P. Allen, A. Clarke and N. Black (eds) Studying the Organisation and Delivery of Health Services: Research Methods (London: Routledge), pp. 188-220, 2001.

MILLIKIN, JOHN P.; HOM, PETER W.; MANZ, CHARLES C.; Self-management competencies in self-managing teams: Their impact on multi-team system productivity, 2010

MIRANDA, R. C. G. ; Uma Revisão Sistemática Sobre Equipes de Desenvolvimento de Software : Tipologia , Características e Critérios de Formação. Dissertação de Mestrado. Universidade Federal de Pernambuco, Recife, PE, Brasil, 2011.

MOE, N. B.; DINGSØYR, T.; DYBÅ, T. Understanding Self-organizing Teams in Agile Software Development. n. 3, p. 76-85, 2008.

MOE, N. B.; DINGSØYR, T.; KVANGARDSNES, Ø. Understanding Shared Leadership in Agile Development : A Case Study. Learning, n. 7465, p. 1-10, 2009.

MOE, N. B.; DINGSØYR, T.; RØYRVIK, E. A. Putting Agile Teamwork to the Test – An Preliminary Instrument for Empirically Assessing. , p. 114-123, 2009.

MOHRMAN, S. A., COHEN, S. G., & MOHRMAN, A. M., JR. ; Designing team-based organizations. San Francisco: Jossey-Bass Publishers, 1995.

MONTEIRO, C. V. F. Impacto do uso de ferramentas de software nas fases iniciais do processo de inovação. Dissertação de Mestrado. Universidade Federal de Pernambuco, Recife, PE, Brasil, 2010.

MUTHUSAMY, S., WHEELER, J. V., & SIMMONS, B. L.; Self-Managing Work Teams : Enhancing Organizational Innovativeness Self-Managing Work Teams : Organization Development Journal, 23(3), 2005.

NOBLIT, GEORGE W.; HARE, R. DWIGHT; Meta-Ethnography: Synthesizing Qualitative Studies, Newbury Park, CA: Sage Publications, 9-37, 1988.

NONAKA, I.; **A dynamic theory of organizational knowledge creation**. *Organization Science,* 5(1), 14-37. 1994.

OATES, J. B.; CAPPER G. Using systematic reviews and evidence-based software engineering with masters students. International Conference on Evaluation & Assessment in Software Engineering, EASE, 2009.

PASMORE, W.; FRANCIS, C.; HALDEMAN, J.; SHANI, A.; Sociotechnical systems: A North American reflection on empirical studies of the seventies. Human Relations, 35, 1179-1204, 1982.

PETERSEN, K.; FELDT, R.; MUJTABA, S.; MATTSSON, M. Systematic Mapping Studies in Software Engineering. , p. 1-10, 2007

PETTICREW, M.; ROBERTS, H. Systematic Reviews in the Social Sciences. Blackwell Publishing, 2006.

PRESSMAN, R. **Software engineering: a practitioner's approach**, 6. ed., 2006.

SAWYER, S. Software Teams. Communications of the ACM, 47, 12, p. 95-99, Dec. 2004. 79

SCHWABER, K.; *Scrum Guide*. Scrum Alliance Resources, 2009.

SOMMERVILLE, I., **Engenharia de Software**, 8ª ed.SP: Pearson, 2007.

STRAUSS, A.; CORBIN, J. **Basics of qualitative research: Grounded theory procedures and techniques.** Newbury Park, CA: Sage Publications, Inc, 1990.

TATA, J.; PRASAD, S. "**Team Self-Management, Organizational Structure, and Judgments of Team Effectiveness,**" Jounal of Managerial Issues, vol. XVI, no. 2, pp. 248–265, 2004.

THOMS, P.; MOORE, K. S.; SCOTT, K. S. The relationship between self-efficacy for participating in self-managed work groups and the big five personality dimensions. Journal of Organizational Behavior, 17(4), 349-362. doi: 10.1002/(SICI)1099-1379(199607)17:4<349::AID-JOB756>3.0.CO;2-3, 1996.

TRIST, E.; The evolution of socio-technical systems: a conceptual framework and an action research program. Ontario Quality of Working Life Centre, Ontario, 1981.

WALL, T. D.; KEMP, N. J.; JACKSON, P. R.; CLEGG, C. W. **Outcome of Autonomous Workgroups: a Long-Term Field Experiment**. Academy of Management Journal, 29(2), 280-304. doi: 10.2307/256189, 1986.

WALL, T. D.; PARKER, S. K.; JACKSON, P. R. Job design and work orientations in modern manufacturing: A comparative case study. Em S. A. Robertson (Ed.), Contemporary ergonomics 1994 (pp. 411-416). New York: Taylor & Francis, 1994.

WHITLEY, K. N.; 'Visual Programming Languages and the Empirical Evidence For and Against', Journal of Visual Languages and Computing 8, 109–142, 1997.

YEATTS, D. E.; HYTEN, C.; High-performing self-managed work teams: A Comparison of Theory to Practice. Ed. Sage Publications, Inc., 1998.

YIN, R.K.; Case Study Research: design and methods. 3rd ed. Applied Social Research Methods. 2003: Sage Publications. 179.

APÊNDICE A. PROTOCOLO DO MAPEAMENTO SISTEMÁTICO

Este apêndice contém o protocolo do Mapeamento Sistemático conduzido nesse trabalho. Ele foi desenvolvido pela aluna de mestrado Elisa S. F. Cardozo e pelo orientador da pesquisa Fabio Queda Bueno da Silva.

Algumas informações já apresentadas neste trabalho ou de menor importância foram omitidas deste protocolo, tais como: Controle de Alterações, Sumário e o Background da pesquisa.

PROTOCOLO

Um Estudo de Mapeamento sobre o Auto-gerenciamento em Equipes de Engenharia de Software

Resumo

Este documento é um protocolo que tem por objetivo o planejamento e detalhamento de um Mapeamento Sistemático da Literatura, sobre os estudos de auto-gerenciamento em equipes de engenharia de software (ES) existentes na literatura. Espera-se com este Mapeamento responder questões sobre que permitam descobrir o quanto a área é consolidada, o que tem sido falado sobre a área, identificar campos de incerteza onde pouca ou nenhuma pesquisa consistente tenha sido feita e entender a força das evidências encontradas para apoiar as conclusões. Pretende-se mapear nos estudos sobre auto-gerenciamento em equipes de ES os objetivos de pesquisa, métodos de pesquisa utilizados e principais resultados. Entre os possíveis enfoques específicos a serem dados na análise dos estudos estão: definição de auto-gerenciamento; os fatores que podem afetar o auto-gerenciamento e os resultados obtidos com o uso do auto-gerenciamento em equipes de software. Para finalizar a análise do mapeamento e confirmar se as evidências encontradas nos estudos são convincentes será realizada uma avaliação da qualidade de todos os estudos incluídos no Mapeamento.

A1. Introdução

No mercado mundial de organizações de TI a crescente competitividade tem feito aumentar a busca das empresas por diferenciais que possibilitem seu destaque perante as outras. Por isso, cada vez mais há um aumento nas exigências de mercado para os produtos e serviços oferecidos, bem como a maneira de produzi-los. Uma das formas de alcançar o destaque pretendido por aquelas organizações é o aumento no desempenho das equipes de trabalho.

Com esse objetivo, as organizações de TI estão cada vez mais implementando o auto-gerenciamento em suas equipes de software. Isso se deve à promessa do auto-gerenciamento em melhorar o desempenho das equipes ou, no mínimo, impactar positivamente em algumas facetas de sua performance (Cohen, 1993).

O auto-gerenciamento propõe que a equipe de trabalho seja composta por indivíduos interdependentes que podem se auto-regular e ter autonomia para tomar decisões sem supervisão direta (Goodman, Devadas, e Hughson, 1988; Manz e Sims, 1993). Por vezes, equipes auto-gerenciadas são organizadas por função. Mas, geralmente, equipes essas equipes são multifuncionais, possuindo uma variedade de habilidades para o trabalho (Thoms, 1996; Wall et al., 1986). Além disso, o auto-gerenciamento trata o conhecimento de forma descentralizada, onde todos os membros da equipe possuem o conhecimento de todo o processo de trabalho (Devaro, 2007).

Equipes auto-gerenciadas podem ou não ter supervisores, mas são responsáveis por suas atividades e métodos de trabalho, com poder e autoridade de decisão sobre como e o que será feito (Cohen, 1993). Muitas vezes, opta-se por um líder ao invés de um supervisor. O papel de líder em equipes auto-gerenciadas é de um facilitador ou coordenador que presta assistência à equipe, ao invés de supervisioná-la (Manz e Sims, 1993).

Por consequência, equipes auto-gerenciadas podem contribuir para aumentar a eficácia organizacional, podendo ter efeito sobre todas as dimensões da eficácia como desempenho, atitudes e comportamentos (Cohen, 1993; Pasmore et al.., 1982). Dessa forma, o auto-gerenciamento pode contribuir para a melhoria na qualidade dos serviços e produtos, e, assim, impulsionar as organizações a alcançar um maior destaque no mercado.

A2. Questões de Pesquisa

A formulação das questões de pesquisa deve ser o primeiro passo para a condução de um método de Engenharia de Software Baseada em Evidências (EBSE), como o Mapeamento Sistemático (ARKSEY e O'MALLEY, 2005; OATES e CAPPER, 2009; PETERSEN et al., 2007).

De acordo com Petersen et al. (2007), as questões de pesquisa devem refletir os objetivos da pesquisa. No caso de um Mapeamento Sistemático, os principais objetivos são: fornecer uma visão geral de uma área de pesquisa, identificar a quantidade, tipos de pesquisa e resultados disponíveis sobre a área investigada e mapear a freqüência das publicações (PETERSEN et al.,2007).

A formulação das questões desta pesquisa foi guiada por Arksey e O'Malley (2005), que recomendam para um Mapeamento Sistemático uma abordagem ampla a fim de se ter uma maior amplitude de cobertura. Com isso, as autoras propõem que nas questões de pesquisa devem ser considerados a população, intervenções ou resultados estudados.

Portanto, esta pesquisa irá considerar a população e a intervenção que serão investigadas:

- **População** – Equipes de Engenharia de Software
- **Intervenção** – Auto-gerenciamento

Com base nos objetivos já mencionados e nas facetas da pesquisa consideradas, as **questões centrais** desse estudo são:

> Q1: O que se sabe atualmente sobre o auto-gerenciamento de equipes no desenvolvimento de software?
> Q2: Qual é a força da evidência em apoio a essas conclusões?

Para responder à questão principal Q1, foram elaboradas as três seguintes questões específicas:

> Q1.1: Como auto-gerenciamento é definido nos estudos da área de Engenharia de Software?
> Q1.2: Quais são os efeitos do auto-gerenciamento nas equipes de desenvolvimento de software?
> Q1.3: Quais são os fatores técnicos, humanos e organizacionais que podem afetar o trabalho de equipes de software auto-gerenciadas?

A3. Seleção dos Estudos

Para selecionar os estudos relevantes para apoiarem as questões desta pesquisa, duas etapas de seleção deverão ser realizadas. Na primeira etapa, os pesquisadores analisarão independentemente cada estudo retornado das buscas e avaliarão a relevância de cada estudo a partir de seus títulos, resumos e palavras-chave. Dessa forma, os estudos que passaram por essa seleção serão documentados em uma lista denominada de Estudos Potencialmente Relevantes, onde todos os estudos pré-selecionados por, pelo menos, um pesquisador serão incluídos.

Os estudos potencialmente relevantes deverão passar, então, por uma segunda seleção mais rigorosa, que resultará nos estudos considerados relevantes para esta pesquisa e, portanto, incluídos. Para isso, deverão ser aplicados os critérios de inclusão e exclusão pré-estabelecidos após a leitura de seus títulos, resumos, introduções, métodos, discussões e conclusões. Nessa etapa, todos os critérios utilizados para inclusão ou exclusão e os dados de cada estudo devem ser documentados, como sugerido por Kitchenham e Charters (2007). Nos casos de discordância entre os pesquisadores sobre a relevância do estudo, um terceiro pesquisador deverá fazer sua avaliação.

A seção A3.1 lista os critérios de exclusão e a seção A3.2 lista os critérios de inclusão que devem ser utilizados na segunda etapa de seleção dos estudos.

A3.1 Critérios de Exclusão

Os seguintes tipos de estudos serão excluídos:

1. Estudos não escritos em inglês.
2. Artigos convidados, tutoriais, *key-note speech*, relatórios de workshop, teses, dissertações, relatórios técnicos, livros.
3. Artigos que expressam pontos de vistas pessoais, opiniões de especialistas ou relatos de experiências.
4. Documentos que não sejam artigos completos (e.x. apresentação de PowerPoint ou resumo estendido).
5. Artigos duplicados, que já foram encontrados em outras fontes.
6. Estudos relacionados a Ciência da Computação que não sejam claramente da área de Engenharia de Software ou Sistemas de Informação.
7. Estudos que não respondem a nenhuma das questões da pesquisa;

8. Trabalhos que não tiveram seus arquivos encontrados/não recuperados;
9. Estudos secundários que obtiverem pontuação menor do que 1 na Avaliação da Qualidade;

A3.2 Critérios de Inclusão

Os estudos que obedecerem aos seguintes critérios serão incluídos na pesquisa:

- Artigos completos publicados em revistas ou conferências revisadas, que relatem o auto-gerenciamento no desenvolvimento de software e respondam a, pelo menos, uma questão de pesquisa;
- Estudos experimentais ou *Empirical Studies*;
- Estudos Secundários, aqueles que dependem de estudos primários;
- Estudos Teóricos, que apresentem conceitos baseados em um entendimento de uma área, referenciando outros trabalhos.

A4. Estratégia de Busca

No intuito de atingir uma maior cobertura da literatura existente, serão realizadas buscas automáticas e buscas manuais. O procedimento que deverá ser realizado para cada uma dessas buscas está descrito nas seções A4.1 e A4.2.

A4.1 Busca Automática

A busca automática é realizada a partir de uma String de Busca, que representa o conjunto de termos encontrados a partir das questões de pesquisa e que serão filtrados em todo o conteúdo dos estudos presentes na base de dados dos engenhos de busca, influindo na obtenção de uma maior cobertura da literatura existente.

Os termos utilizados para a busca automática desta pesquisa foram construídos em quatro etapas:

1. A partir das variáveis identificadas como independente (X) e dependente (Y), as palavras-chave são extraídas.

2. As palavras-chave são, então, traduzidas para o idioma Inglês, tendo em vista a predominância desse idioma nas fontes de busca utilizadas e também nos principais jornais e conferências da área de investigação.
3. A partir das palavras-chave já traduzidas, são identificados seus sinônimos, levando em conta sua relevância para a pesquisa, como apresentado na Tabela A1.
4. Assim, a String de Busca que será utilizada nos Engenhos de Busca é formada, a partir da combinação entre palavras-chave e sinônimos. Para isso, são usados os operadores *OR* entre os sinônimos identificados e *AND* entre as palavras-chave identificadas.

A etapas 3 e 4 são iterativas, ou seja, foram feitas várias revisões dos sinônimos utilizados, incluindo consultas com especialistas da área investigada, para se chegar à String de Busca final. A Figura A1 sintetiza todo esse procedimento. Na Tabela A2, podemos ver como ficou a versão final da String de Busca utilizada nesta pesquisa.

Devido a suas particularidades, cada engenho de busca automática foi estudado e teve uma String de Busca adaptada a partir da string de busca original, a fim de garantir que nenhum estudo fosse omitido.

Tabela A1 – Termos para construção da string de busca

Termo Identificado	Termos a serem usados	
Equipes de Engenharia de Software	• **Software Team** • **Software Group** • **Software Pair** • **Software Partner** • **Software Workgroup** • **Software Work Team** • **Software Development Group** • **Software Development Pair** • **Software Development** • **Software Development Partner** • **Software Development Workgroup** • **Software Development Work Team** • **Agile Team** • **Agile Group**	• **Agile Partner** • **Agile Workgroup** • **Agile Work Team** • **Agile Development Team** • **Agile Development Group** • **Agile Development Pair** • **Agile Development Partner** • **Agile Development Workgroup** • **Agile Development Work Team** • **Programming Team** • **Programming Group** • **Programming Pair** • **Programming Partner** • **Programming Workgroup** • **Programming Work Team**

	• **Agile Pair**	
Auto-gerenciamento	• **Self-managed** • **Self-manageable** • **Self-managing** • **Self-organized** • **Self-organizing** • **Self-directed** • **Self-guided** • **Self-supervised**	• **Self-coached** • **Self-coordinated** • **Self-sufficient** • **Self-designed** • **Self-headed** • **Empowered** • **Autonomous**

Fonte: Elaboração própria

Figura A1 – Procedimento de Construção da String de Busca

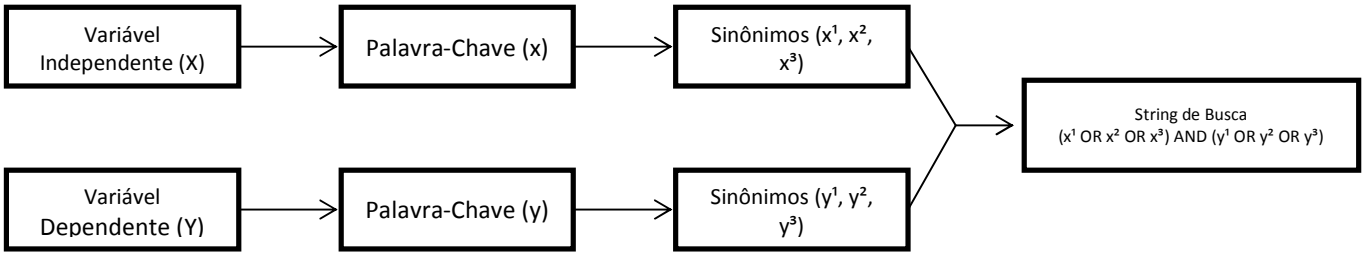


Tabela A2 – String de Busca

String de busca
("software team" **OR** "software application team" **OR** "software system team" **OR** "programming team" **OR** "software development team" **OR** "software work team" **OR** "software development work team" **OR** "software engineering team" **OR** "programming work team" **OR** "software project team" **OR** "software group" **OR** "software application group" **OR** "software system group" **OR** "programming group" **OR** "software development group" **OR** "software project group" **OR** "software engineering group" **OR** "software workgroup" **OR** "software development workgroup" **OR** "programming workgroup" **OR** "agile team" **OR** "agile development team" **OR** "agile group" **OR** "agile development group") **AND** ("empowered" **OR** "empowerment" **OR** "empowering" **OR** "autonomous" **OR** "autonomy" **OR** "semi-autonomous" **OR** "semi-autonomy" **OR** "self-managing" **OR** "self-managed" **OR** "self-management" **OR** "self-organizing" **OR** "self-organized" **OR** "self-organization" **OR** "self-manageable" **OR** "self-guided" **OR** "self-supervised" **OR** "self-coached" **OR** "self-coordinated" **OR** "self-determining" **OR** "self-determined" **OR** "self-designing" **OR** "self-

designed" **OR** "self-sufficient" **OR** "self managing" **OR** "self managed" **OR** "self management" **OR** "self organizing" **OR** "self organized" OR "self organization" **OR** "self determining" **OR** "self determined" **OR** "self designing" **OR** "self designed" **OR** "self sufficient" **OR** "self manageable" **OR** "self guided" **OR** "self supervised" **OR** "self coached" **OR** "self coordinated")

Fonte: Elaboração Própria

A4.2 Busca Manual

Este processo de busca compreenderá identificar os estudos referenciados nos estudos incluídos nessa revisão (do inglês, *Snowball*), além de analisar nos sites dos *journals* cada um dos anos de publicação, volumes e *issues*. Portanto, a busca manual não faz uso da String de Busca anteriormente apresentada.

Como observado no Capitulo 2 do trabalho, com o conhecimento prévio obtido para a pesquisa, foi observado que alguns autores referiam-se ao método ágil como favorecedor do auto-gerenciamento em equipes de software (BEGEL e NAGAPPAN, 2007). A partir de então, o processo de busca manual deverá analisar os estudos publicados a partir do ano 2001, ano de publicação do Manifesto Ágil.

A5. Engenhos de pesquisa

No intuito de atingir uma ampla cobertura dos estudos existentes sobre o tema de pesquisa, serão realizadas buscas automáticas e buscas manuais. Esta seção apresenta detalhadamente o procedimento para realização de cada uma dessas buscas, bem como os Engenhos de Busca utilizados.

A5.1 Buscas Automáticas

As buscas automáticas são feitas por meio de Engenhos de Busca, com o uso de uma String de Busca, como apresentado anteriormente na seção 3.3.1.1. Para que um engenho de busca fosse selecionado, foram utilizados os seguintes critérios:

- Deve ser acessado para consulta dos estudos através da Internet;
- Deve ser considerado importante e relevante para a área de investigação;

A partir desses critérios, chegamos aos cinco engenhos de busca utilizados na pesquisa, como mostra a Tabela A3.

Tabela A3 – Fontes da Busca Automática

	Engenhos	Sites
1	ACM Digital library	http://portal.acm.org/
2	IEEE Xplore Digital Library	http://ieeexplore.ieee.org/
3	ScienceDirect	http://www.sciencedirect.com/
4	Scopus	http://www.scopus.com/
5	Jstor	http://www.jstor.org/

Fonte: Elaboração Própria

A5.2 Busca Manual

Este processo de busca consiste em identificar os estudos referenciados nos estudos incluídos nessa revisão (Snowball) e analisar nos sites dos *journals* cada um dos anos de publicação, volumes e *issues*, lendo os títulos, resumos e palavras-chaves. Serão descartados (sem necessidade de documentação) os estudos julgados como sendo claramente irrelevantes para a pesquisa. Aqueles que forem incluídos devem ser documentados.

A busca manual em *journals* será feita nas seguintes fontes[7]:

Tabela A4 – Fontes de Busca utilizadas para a condução da Busca Manual

Fonte	Tipo	Referência
1. Information and Software Technology (IST)	Journal	http://www.informatik.uni-trier.de/~ley/db/journals/infsof/
2. IEEE Transactions on Software Engineering (TSE)	Transactions	http://www.computer.org/portal/web/csdl/transactions/tse#1
3. Software Practice and Experience (SPE)	Journal	http://onlinelibrary.wiley.com/journal/10.1002/%28ISSN%291097-024X
4. Journal of System and Software (JSS)	Journal	
5. International Symposium on Empirical Software Engineering and Measurement (ESEM)	Proceedings	http://portal.acm.org/event.cfm?id=RE228&tab=pubs&CFID=15631611&CFTOKEN=81304600 http://ieeexplore.ieee.org/xpl/conhome.jsp?punumber=1001929
6. Evaluation and Assessment of Software Engineering (EASE)	Proceedings	http://www.scm.keele.ac.uk/ease/archive.html

[7] As fontes utilizadas para a busca manual foram indicadas por um especialista na área de Engenharia de Software.

7. Human and Social Factors of Software Engineering (ICSE - HSSE)	Proceedings	http://ieeexplore.ieee.org/xpl/conhome.jsp?punumber=1000691 HSSE '05:Proceedings of the 2005 workshop on Human and social factors of software engineering
8. ACM Transactions on Software Engineering Methodology (TOSEM)	Proceedings	
9. Agile Conference	Proceedings	http://www.agileconference.org/
10. Snowball	--	Realizado a partir da busca nos estudos considerados relevantes para esta pesquisa
11. Pesquisa Aleatória	--	scholar.google.com/

A6. Avaliação de Qualidade

Apesar de Arksey e O'Malley (2005) indicarem que em um Estudo de Mapeamento Sistemático não é realizada a avaliação da qualidade dos estudos encontrados, será realizada a avaliação da qualidade de cada um dos estudos incluídos na pesquisa. Além de essa avaliação ser considerada importante por Kitchenhan (2004), também visa apoiar a resposta à segunda questão central da pesquisa, que questiona qual é a força das evidências que apóiam as conclusões obtidas a partir da questão central Q1. Isso se deve ao fato de avaliações da qualidade serem consideradas como um processo que determina a força das inferências (KHAN et al., 2001)

Portanto, cada um dos estudos considerados relevantes para este trabalho terá sua qualidade avaliada independentemente por dois pesquisadores. Após a finalização da avaliação individual, ambos os pesquisadores discutirão cada uma das divergências e chegarão a um acordo comum sobre elas, gerando uma única lista contendo a pontuação final de cada estudo e posterior normalização, permitindo a comparação.

O processo de Avaliação da Qualidade é realizado com o uso de formulários pré-estabelecidos. Para esta pesquisa, serão utilizados formulários diferentes, de acordo com o tipo do estudo avaliado. Aqui, foram considerados estudos experimentais, teóricos e secundários. Estudos experimentais, ou empíricos, são baseados em evidências diretas ou experimentos. Os estudos teóricos são conceituais e baseados em um entendimento de uma área, referenciando outros trabalhos relacionados. Já os estudos secundários reexaminam outros trabalhos ou fazem mapeamentos sistemáticos e revisões da literatura sistemáticas ou tradicionais.

Abaixo, serão apresentados os formulários com os critérios de qualidade utilizados para avaliar cada tipo de estudo, sendo a seção 6.1 apresentando o procedimento para estudos empíricos, 6.2 para estudos teóricos e a 6.3 para estudos secundários.

A6.1 Estudos Experimentais

A avaliação dos estudos experimentais, ou empíricos, ocorrerá por meio de uma lista de critérios e suas pontuações. Os estudos empíricos serão avaliados por meio de 10 critérios, baseados no formulário extraído de Dyba *et al.* (2008).Os critérios de avaliação da qualidade, bem como a pontuação que deve ser atribuída a cada um deles são apresentados no Formulário A.

Formulário A

Formulário de Avaliação da Qualidade de Estudos Empíricos		
Avaliador:	**ID:**	**Data:**
Critérios de Avaliação da Qualidade	**Pontuação**	**Opções para pontuação**
1. Há uma declaração clara dos objetivos da pesquisa?		Sim = 1 / Não = 0
2. Existe uma descrição adequada do contexto no qual a pesquisa foi realizada?		Sim = 1 / Não = 0
3. O projeto de pesquisa foi apropriado para abordar os objetivos da pesquisa?		Sim = 1 / Não = 0
4. A estratégia de recrutamento foi adequado aos objetivos da pesquisa?		Sim = 1 / Não = 0
5. Existia um grupo de controle com o qual comparar tratamentos?		Sim = 1 / Não = 0
6. Os dados foram coletados de forma que tenha abordado a questão de pesquisa?		Sim = 1 / Não = 0
7. A análise dos dados foi suficientemente rigorosa?		Sim = 1 / Não = 0
8. A relação entre pesquisador e participante foi considerada em um grau adequado?		Sim = 1 / Não = 0
9. Há uma declaração clara dos resultados?		Sim = 1 / Não = 0
10. O estudo tem valor para pesquisa ou prática?		Sim = 1 / Não = 0
Pontuação Total		**%**

Fonte: Adaptado de Dyba et al. (2008).

A6.2 Estudos Teóricos

Os estudos Teóricos serão avaliados a partir de 2 critérios, extraídos e adaptados de Beecham *et al.* (2006). Os critérios de avaliação da qualidade, bem como a pontuação que deve ser atribuída a cada um deles são apresentados no Formulário B.

Formulário B

Formulário de Avaliação da Qualidade de Estudos Teóricos		
Avaliador:	**ID:**	**Data:**
Critérios de Avaliação da Qualidade	**Pontuação**	**Opções para pontuação**
1. O relatório do estudo faz claras e inequívocas descobertas baseadas em evidências e argumentos?		Sim = 1 / Moderadamente = 0,5 / Não = 0
2. O estudo é bem/adequadamente referenciado?		Sim = 1 / Moderadamente = 0,5 / Não = 0
Pontuação Total		**%**

Fonte: Adaptado de Beecham et al. (2006).

A6.3 Estudos Secundários

A qualidade de estudos considerados secundários, como Revisões Sistemáticas da Literatura e Estudos de Mapeamento Sistemáticos, será avaliada por meio de 4 critérios, obtidos em (*Centre for Reviews and Dissemination, 2007*), onde em cada critério o pesquisador utilizará uma nota entre “1”, “0.5” e “0” (Formulário C). Aquele estudo que obtiver pontuação final inferior a 1 não deve ser considerado relevante para a pesquisa, por ter sido considerado de baixa qualidade.

Formulário C

Formulário de Avaliação da Qualidade de Estudos Secundários		
Avaliador:	**ID:**	**Data:**
Critérios	**Pontuação**	**Opções para pontuação**
1. São a inclusão da revisão e critérios de exclusão descritos e adequada?		Sim = 1 / Moderadamente = 0,5 / Não = 0
2. É a pesquisa bibliográfica probabilidade de ter coberto todos os estudos relevantes?		Sim = 1 / Moderadamente = 0,5 / Não = 0
3. Será que os comentários de avaliar a qualidade / validade dos estudos incluídos?		Sim = 1 / Moderadamente = 0,5 / Não = 0
4. Foram os dados básicos / estudos adequadamente descrita?		Sim = 1 / Moderadamente = 0,5 / Não = 0
Pontuação Total		**%**

Fonte: Adaptado de Centre for Reviews and Dissemination (2007)

A7. Estratégia para Extração dos Dados

A Extração dos Dados será realizada pelos pesquisadores, individualmente, com o preenchimento de um formulário, desenvolvido no Microsoft Excel ™. O Formulário D contém informações detalhadas sobre a identificação de cada um dos estudos incluídos na pesquisa. No Formulário E serão documentadas informações sobre os estudos excluídos na segunda etapa de seleção da pesquisa, como: título, local de publicação e critério usado para exclusão. No Formulário F serão preenchidos os dados relevantes para apoiar as respostas das questões desta pesquisa.

Formulário D

Formulário de Trabalhos Incluídos								
ID	**Ano**	**Autores**	**País**	**Tipo de Estudo**	**Método de Pesquisa**	**Sujeito investigado**	**Avaliação da Qualidade**	**Foco do estudo**

Fonte: Elaboração Própria

Formulário E

Formulário de Trabalhos Excluídos		
ID	**País**	**Critérios utilizados para exclusão**

Fonte: Elaboração Própria

Formulário F

Formulário de Coleta de Dados **Mapeamento Sistemático sobre o Auto-gerenciamento em Equipes de Software**					
ID:	*Revisor:*		*Data de Avaliação:*		
Título do Trabalho:					
Autores do Trabalho:					
Tipo do Estudo:			*Fonte:*		
Data de Publicação:			*Local de Publicação:*		
Critérios de Inclusão utilizados:					
Método de Gerenciamento abordado	*Fatores Técnicos Influentes (Q1.3)*	*Fatores Humanos Influentes (Q1.3)*	*Fatores Organizacionais Influentes (Q1.3)*	*Características (Q1.1)*	*Efeitos (Q1.2)*

Fonte: Elaboração Própria

A8. Síntese dos Dados

Após a extração dos dados, será feita a análise e a síntese desses dados, de modo que permita o mapeamento das informações, representando o conhecimento gerado por esta pesquisa. Portanto, o processo de síntese dos dados apresentará a freqüência com que esses dados são tratados nos estudos já publicados, para, com isso, auxiliar nas respostas às questões de pesquisa.

Para proceder a síntese dos dados obtidos a partir da extração, como mencionado no item 3.1 deste trabalho, será realizada uma Meta-Etnografia, proposta originalmente por Noblit et al. (1988), em que é usado o Método de Comparações Constantes (*Constant Comparison Method*),

proposto por Glaser e Strauss (1967). De acordo com Noblit et al. (1988), além da determinação das questões de pesquisa apresentadas em A2, da compreensão do que é relevante para a pesquisa apresentado em A3 e da leitura dos estudos incluídos, o procedimento da meta-etnografia consiste em:

- Determinar a relação existente entre os estudos, identificando palavras, frases, temas ou conceitos-chave e documentando a partir dos termos originais, na tentativa de não causar ruídos;
- Interpretar e comparar os estudos para traduzir os conceitos abordados;
- Sintetizar as traduções para desenvolver novas interpretações e conceitos;
- Integrar a síntese e apresentar o resultado;

A partir dos dados aqui obtidos, serão gerados gráficos e tabelas dinâmicas, focados em freqüência e percentual, que, de acordo com Kitchenham (2007) constituem um mecanismo de comunicação eficaz. Esses mecanismos foram apresentados na Seção 4.2 desse trabalho.

A9. Apresentação do Mapeamento

Este estudo foi planejado para apoiar as respostas a questões de uma pesquisa que será utilizada para conclusão do curso de Mestrado em Ciência da Computação na área de Engenharia de Software, realizado no Centro de Informática da Universidade Federal de Pernambuco. Para isso, será elaborado um relatório detalhado contendo todos os aspectos da pesquisa.

Além disso, planejamos publicar os resultados da realização do Estudo de Mapeamento Sistemático sobre o auto-gerenciamento em equipes de software em uma revista ou como um artigo de conferência na área de engenharia de software.

A10. Alterações no Protocolo

No decorrer da aplicação dos procedimentos, é provável que ocorram mudanças no protocolo. Algumas mudanças serão feitas por necessidade e outras para melhorar a qualidade do processo. Todas as mudanças deverão ser registradas e o protocolo atualizado.

Referências (do Protocolo do Mapeamento)

AGILE MANIFESTO, 2001. Disponível em: </http://www.agilemanifesto.org/S>

ARKSEY, H.; O'MALLEY, L. Scoping studies: towards a methodological framework. International Journal of Social Research Methodology, v. 8, n. 1, p. 19-32, 2005. Disponível em: <http://www.informaworld.com/openurl?genre=article&doi=10.1080/1364557032000119616&magic=crossref||D404A21C5BB053405B1A640AFFD44AE3>.

BEECHAM, S., BADDOO, N., HALL, T., & ROBINSON, H. Protocol for a Systematic Literature Review of Motivation in Software Engineering Systematic Review – Cover Sheet. Computer, 2006.

BEGEL, A; NAGAPPAN, N. Usage and perceptions of agile software development in an industrial context: An exploratory study. *ESEM '07:* 255–264, IEEE Computer Society, Washington, 2007.

CENTRE FOR REVIEWS AND DISSEMINATION. What are the criteria for the inclusion of reviews on DARE?, 2007 Disponível em: <http://www.york.ac.uk/inst/crd/faq4.htm>, (acessado em 17/09/2010).

COHEN, S. G. Designing Effective Self-Managing Work Teams. Advances, 1, 1993.

DEVARO, J.; EAST, I. H.; The Effect of Self-Managed and Closely-Managed Teams on Labor Productivity and Product Quality : An Empirical Analysis of a Cross Section of Establishments The Effects of Self-Managed and Closely-Managed Teams on Labor Productivity and Product Quality : A. Employee Relations, 2007.

DYBA, T.; DINGSØYR, T.; Empirical studies of agile software development : A systematic review. Information and Software Technology. doi: 10.1016/j.infsof.2008.01.006, 2008.

GLASER B.; STRAUSS, A. L.; *The Discovery of Grounded Theory*. Aldine, Chicago, 1967.

GOODMAN, P. S.; DEVADAS, R.; HUGHSON, T. L.; Groups and productivity: Analyzing the effectiveness of self-managing teams. In J.P. Campbell, R.J. Campbell, & Associates (Eds.), Productivity in organizations, pp. 295-325. San Francisco: Jossey-Bass & Associates, 1988.

KHAN, K.S.; TER RIET, G.; GLANVILLE, J.; SOWDEN, A.J.; KLEIJNEN, J.;. Undertaking Systematic Review of Research on Effectiveness. CRD Report Number 4 (Second Edition), NHS Centre for Reviews and Dissemination, University of York, UK, 2001.

KITCHENHAM, B. A.; Procedures for Performing Systematic Reviews. joint technical report, Software Engineering Group, Keele Univ., and Empirical Software Eng., Nat'l ICT Australia, 2004.

KITCHENHAM, B. Guidelines for performing Systematic Literature Reviews in Software Engineering. Vol 2.3 EBSE Technical Report, EBSE-2007-01, 2007.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering, Technical Report EBSE-2007-01, School of Computer Science and Mathematics, Keele University, 2007.

MANZ, C. C.; SIMS, H. P., Jr.; Business Without Bosses, John Wiley & Sons, New York, 1993.

NOBLIT, George W.; HARE, R. Dwight; Meta-Ethnography: Synthesizing Qualitative Studies, Newbury Park, CA: Sage Publications, 9-37, 1988.

OATES, J. B.; CAPPER G. Using systematic reviews and evidence-based software engineering with masters students. International Conference on Evaluation & Assessment in Software Engineering, EASE, 2009.

PASMORE, W.; FRANCIS, C.; HALDEMAN, J.; SHANI, A.; Sociotechnical systems: A North American reflection on empirical studies of the seventies. Human Relations, 35, 1179-1204, 1982.

PETERSEN, K.; FELDT, R.; MUJTABA, S.; MATTSSON, M. Systematic Mapping Studies in Software Engineering. , p. 1-10, 2007

THOMS, P.; MOORE, K. S.; SCOTT, K. S.; The relationship between self-efficacy for participating in self-managed work groups and the big five personality dimensions. Journal of Organizational Behavior, 17(4), 349-362. doi: 10.1002/(SICI)1099-1379(199607)17:4<349::AID-JOB756>3.0.CO;2-3, 1996.

WALL, T. D.; KEMP, N. J.; JACKSON, P. R.,; CLEGG, C. W. Outcomes of autonomous workgroups: A long-term field experiment. Academy of Management Journal, 29(2), 280-304, 1986.

APÊNDICE B. Estudos Incluídos na Pesquisa

Abaixo, a Tabela B1 apresenta o detalhamento de todos os 43 estudos incluídos no Mapeamento Sistemático, incluindo seu ano de publicação, engenho de busca onde foi encontrado e sua referência.

Tabela B1 – Estudos incluídos no Mapeamento

ID	Ano	Engenho de Busca	Referência
SM016	2007	ACM	QUMER, A; HENDERSON-SELLERS, B. A framework to support the evaluation , adoption and improvement of agile methods in practice. **Journal of Systems and Software**, v. 81, p. 1899-1919, 2008.
SM023	2010	IEEE	SAMPAIO, S.; BARROS, E.; AQUINO, G.; SILVA, M.; MEIRA, S. A Review of Productivity Factors and Strategies on Software Development. **International Conference on Software Engineering Advances**, 2010.
SM034	2010	ACM	MOE, NILS BREDE; DINGSØYR, T.; DYBÅ, T. A teamwork model for understanding an agile team : A case study of a Scrum project. **Information and Software Technology**, v. 52, n. 5, p. 480-491, 2010. Elsevier B.V. Disponível em: <http://dx.doi.org/10.1016/j.infsof.2009.11.004>.
SM040	2010	ACM	DUBINSKY, Y.; HAZZAN, O. Ad-Hoc Leadership in Agile Software Development Environments. **ACM Transactions on Software Engineering and Methodology**, p. 32-38, 2010.
SM043	2009	Science Direct	KETTUNEN, P. Adopting key lessons from agile manufacturing to agile software product development — A comparative study. **Technovation**, v. 29, p. 408-422, 2009.
SM060	2010	ACM	WOODWARD, E. V.; BOWERS, R.; THIO, V. S. et al.. Agile methods for software practice transformation. **IBM Journal of Research and Development**, v. 54, n. 2, p. 1-12, 2010. Disponível em: <http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5438943>. Acesso em: 15/10/2010.
SM103	2004	Scopus	SHEN, Y.; GALLIVAN, M. An Empirical Test of the Job Demand / Control Model Among IT Users. **SIGMIS'04**, p. 39-47, 2004.
SM105	2004	EASE	SHARP, H.; ROBINSON, H. An Ethnographic Study of XP Practice. **Empirical Software Engineering**, p. 353-375, 2004.
SM119	2004	Software Practice	ACUÑA, S. T.; JURISTO, N. Assigning people to roles in

		and Experience	software projects. **Software - Practice and Experience**, v. 696, n. October 2003, p. 675-696, 2004.
SM121	2006	Scopus	HOEGL, M.; PARBOTEEAH, K. P. Autonomy and teamwork in innovative projects. **Human Resourse Management**, v. 45, n. 1, p. 67-79, 2006.
SM124	2010	ACM	HODA, R.; NOBLE, J.; MARSHALL, S. Balancing Acts : Walking the Agile Tightrope. **CHASE'10**, p. 5-12, 2010.
SM125	2009	Scopus	BARNEY, H. T.; MOE, NILS B; DYBÅ, T.; AURUM, A.; WINATA, M. Balancing Individual and Collaborative Work in Agile Teams. **Springer**, p. 53-62, 2009.
SM129	2010	IEEE	KARHATSU, H.; IKONEN, M.; KETTUNEN, P.; FAGERHOLM, F.; ABRAHAMSSON, P. Building Blocks for Self-Organizing Software Development Teams - A Framework Model and Empirical Pilot Study. **2010 2nd International Conference on Software Technology and Engineering(ICSTE)**, p. 297-304, 2010.
SM140	2010	Agile Conference	KAJKO-MATTSSON, M.; AZIZYAN, G.; MAGARIAN, M. K. Classes of Distributed Agile Development Problems. **Agile Conference 2010**, p. 51-58, 2010.
SM142	2009	Scopus	VIDGEN, R.; WANG, X. Coevolving Systems and the Organization of Agile Software Development. **Information Systems Research**, v. 20, n. 3, p. 355-376, 2009.
SM152	2007	IEEE	QUMER, ASIF; HENDERSON-SELLERS, B. Construction of an Agile Software Product-Enhancement Process by Using an Agile Software Solution Framework (ASSF) and Situational Method Engineering. **31st Annual International Computer Software and Applications Conference(COMPSAC 2007)**, 2007.
SM185	2007	IEEE	BEECHAM, S.; SHARP, H.; BADDOO, N.; HALL, T.; ROBINSON, H. Does the XP environment meet the motivational needs of the software developer ? An empirical study. **IEEE Computer Society**, 2007.
SM197	1984	SLR MOTIVATION	CHENEY, P. H. Effects of Individual Characteristics, Organizational Factors and Task Characteristics on Computer Programmer Productivity and Job Satisfaction. **Elsevier Science Publishers B.V.**, v. 7, 1984.
SM206	2008	Science Direct	DINGSØYR, T.; DYBÅ, T. Empirical studies of agile software development : A systematic review. **Information and Software Technology**, 2008.
SM260	2005	Agile Conference	SUTHERLAND, J. Future of Scrum : Parallel Pipelining of Sprints in Complex Projects. **Agile 2005 Conference**, 2005.
SM262	2009	IEEE	TALBY, D.; DUBINSKY, Y. Governance of an Agile Software

			Project. **ICSE'09**, p. 40-45, 2009.
SM280	2009	Scopus	ACUÑA, S. T.; GÓMEZ, M.; JURISTO, N. How do personality , team processes and task characteristics relate to job satisfaction and software quality ? **Information and Software Technology**, v. 51, n. 3, p. 627-639, 2009. Elsevier B.V. Disponível em: <http://dx.doi.org/10.1016/j.infsof.2008.08.006>.
SM329	2007	Scopus	TESSEM, B.; MAURER, F. Job Satisfaction and Motivation in a Large Agile Team. **XP'07 Proceedings**, , n. 5020, 2007.
SM346	1992	Jstor	HENDERSON, J. C.; LEE, S. Managing I / S Design Teams : A Control Theories Perspective. **Management Science**, v. 38, n. 6, p. 757-777, 2011.
SM369	2008	IEEE	ZHANG, S.; TREMAINE, M.; EGAN, R. et al.. Occurrence and Effects of Leader Delegation in Virtual Teams. **Proceedings of the 41st Hawaii International Conference on System Sciences**, p. 1-10, 2008.
SM378	2010	Scopus	HODA, R.; NOBLE, J.; MARSHALL, S. Organizing Self-Organizing Teams. **ICSE'10**, p. 285-294, 2010b.
SM382	2009	IEEE	MOE, NILS BREDE; DINGSØYR, T.; DYBÅ, T. Overcoming Barriers to Self-Management in Software Teams. **IEEE Computer Society**, 2009.
SM392	1995	SLR MOTIVATION	SANTANA, M.; ROBEY, D. Perceptions of Control During Systems Development : Effects on Job Satisfaction of Systems Professionals. **Computer Personnel**, , n. January, 1995.
SM400	2009	Scopus	MOE, NILS BREDE; DINGSØYR, T.; RØYRVIK, E. A. Putting Agile Teamwork to the Test – An Preliminary Instrument for Empirically Assessing and Improving Agile Software Development. **XP 2009 Conference**, p. 114-123, 2009.
SM426	2008	BUSCA ALEATÓRIA	MOE, NILS BREDE; DINGSØYR, T. Scrum and Team Effectiveness : Theory and Practice. **XP 2008 Conference**, , n. 7465, p. 11-20, 2008.
SM428	2008	IEEE	MARCHENKO, A.; ABRAHAMSSON, P. Scrum in a Multiproject Environment : An Ethnographically-Inspired Case Study on the Adoption Challenges. **Agile 2008 Conference**, p. 15-26, 2008.
SM430	2010	IEEE	RONG, G.; SHAO, D.; ZHANG, H. SCRUM-PSP : Embracing Process Agility and Discipline. **2010 Asia Pacific Software Engineering Conference**, 2010.
SM493	2007	Scopus	MCAVOY, J.; BUTLER, T. The impact of the Abilene Paradox on double-loop learning in an agile team. **Information and Software Technology**, v. 49, p. 552-563, 2007.

SM506	2009	ACM	NAWAZ, A. I.; ZUALKERNAN, I. A. The Role of Agile Practices in Disaster Management and Recovery : A Case Study. **CASCON '09 Proceedings of the 2009 Conference of the Center for Advanced Studies on Collaborative Research**, p. 164-173, 2009.
SM511	2009	SLR COHESION	MCAVOY, J.; BUTLE, T. The Role Of Project Management In Ineffective Decision Making Within Agile Software. **European Journal of Information Systems**, 2009.
SM516	2007	Scopus	WHITWORTH, E.; BIDDLE, R. The Social Nature of Agile Teams. **Agile 2007 Conference**, 2007.
SM528	2000	SLR MOTIVATION	GAMBILL, S. E.; CLARK, W. J.; WILKES, R. B. Toward a holistic model of task design for IS professionals. **Information & Management**, v. 37, 2000.
SM534	2010	Scopus	LI, J.; MOE, NILS B; DYBÅ, T. Transition from a Plan-Driven Process to Scrum – A Longitudinal Case Study on Software Quality. **ESEM'10**, 2010.
SM541	2008	IEEE	MOE, NILS BREDE; AURUM, A. Understanding Decision-Making in Agile Software Development : a Case-study. **34th Euromicro Conference Software Engineering and Advanced Applications**, p. 216-223, 2008.
SM543	2008	Scopus	MOE, NILS BREDE; DINGSØYR, T.; DYBÅ, T. Understanding Self-organizing Teams in Agile Software Development. **19th Australian Conference on Software Engineering (aswec 2008)**, , n. 3, p. 76-85, 2008.
SM544	2009	IEEE	MOE, NILS BREDE; DINGSØYR, T.; KVANGARDSNES, Ø. Understanding Shared Leadership in Agile Development : A Case Study. **Proceedings of the 42nd Hawaii International Conference on System Sciences**, , n. 7465, p. 1-10, 2009.
SM549	2007	IEEE	JUDY, K. H.; KRUMINS-BEENS, I. Using Agile Practices to Spark Innovation in a Small to Medium Sized Business. **Proceedings of the 40th Hawaii International Conference on System Sciences**, p. 1-10, 2007.
SM573	2003	Agile Conference	ROBINSON, H.; SHARP, H. XP Culture : Why the twelve practices both are and are not the most significant thing. **Proceedings of the Agile Development Conference (ADC'03)**, 2003.

APÊNDICE C. Estudos Excluídos da Pesquisa

Abaixo, a Tabela C1 apresenta as referências de todos os 533 estudos excluídos do Mapeamento Sistemático, bem como os critérios de exclusão utilizados.

Tabela C1 – Estudos excluídos do Mapeamento

ID	Titulo	Critérios Utilizados
SM001	"Come Together, Right Now" - How the Songs of The Beatles Helped our Product Owners and Teams Live in Harmony	3; 7
SM002	"It's Not the Pants, it's the People in the Pants" Learnings from the Gap Agile Transformation What Worked, How We Did it, and What Still Puzzles Us	3
SM003	25 percent Ahead of Schedule and just at Step 2 of the SAMI	7; 3; 5
SM004	25 percent Ahead of Schedule and just at “Step 2” of the SAMI	7
SM005	5 Users Every Friday: A Case Study in Applied Research	7
SM006	5 Users Every Friday: A Case Study in Applied Research	7; 5
SM007	7 Years of Agile Management	3
SM008	A brief report on working smarter with Agile software development	7
SM009	A case study on the impact of customer communication on defects in agile software development	7
SM010	A case study on the impact of scrum on overtime and customer satisfaction	7
SM011	A comparison of issues and advantages in agile and incremental development between state of the art and an industrial case	7
SM012	A Contingency Approach to Software Project Coordination	8
SM013	A control theory perspective on agile methodology use and changing user requirements	7
SM014	A Dynamic Coordination Policy for Software System Construction	7
SM015	A Framework to Support Non-fragile Agile Agent-Oriented Software Development	8
SM017	A framework to support the evaluation, adoption and improvement of agile methods in practice	5
SM018	A Model-Driven Scrum Process for Service-Oriented Software Reengineering: mScrum4SOSR	7
SM019	A network perspective on open source software development: team formation and community participation	7
SM020	A Preliminary Roadmap for Empirical Research on Agile Software Development	7
SM021	A Product Manager's Guide to Surviving the Big Bang Approach to Agile Transitions	3

SM022	A report on the XP workshop on agile product line engineering	7
SM024	A Self Funding Agile Transformation	7
SM025	A software methodology for applied research eXtreme Researching	7
SM026	A software-based trust framework for distributed	7
SM027	A Study Case: Evolution of Co-Location and Planning Strategy	7
SM028	A study into the effects of personality type and methodology on cohesion in software engineering teams	7
SM029	**A survey of self-management in dynamic software architecture specifications**	**6**
SM030	A survey study of critical success factors in agile software projects	7
SM031	A tale of two writing teams	3; 7
SM032	A teamwork model for understanding an agile team A case study of a Scrum project	5
SM033	A teamwork model for understanding an agile team: A case study of a Scrum project	5
SM035	A teamwork model for understanding an agile team: A case study of a Scrum project	5
SM036	A Technical Story	3
SM037	Acceptance of agile methodologies: A critical review and conceptual framework	7
SM038	Accidental Adoption: The Story of Scrum at Amazon.com	7; 5
SM039	Accidental adoption: The story of scrum at amazon.com	7
SM041	Adopting an Agile Culture	3
SM042	Adopting an Agile Culture A User Experience Team's Journey	3
SM044	Adventures in Agile Contracting: Evolving from Time and Materials to Fixed Price, Fixed Scope Contracts	3
SM045	Agile at Yahoo! From the Trenches	3
SM046	Agile development : Issues and avenues requiring a substantial enhancement of the business perspective in large projects	3; 5
SM047	Agile development and remote teams: learning to love the phone	3; 7
SM048	Agile development in the old economy	3
SM049	Agile Development Iterations and UI Design	7
SM050	Agile development: issues and avenues requiring a substantial enhancement of the business perspective in large projects	3
SM051	Agile development: overcoming a short-term focus in implementing best practices	3
SM052	Agile EDI Framework for B2B Applications	7
SM053	Agile ERP: "You don't know what you've got 'till it's gone!"	7
SM054	Agile ERP: "You don't know what you've got 'till it's gone!"	7; 5
SM055	Agile in India: challenges and lessons learned	3
SM056	Agile management - an oxymoron?: who needs managers anyway?	3

SM057	Agile methods applied to embedded firmware development	3
SM058	Agile methods for large organizations - building communities of practice	7
SM059	Agile methods for software practice transformation	5
SM061	Agile monitoring using the line of balance	7
SM062	Agile offshore techniques - a case study	7
SM063	Agile Orientation and Psychological Needs, Self-Efficacy, and Perceived Support: A Two Job-Level Comparison	7
SM064	Agile phase I - the pragmatic case study of Schneider National	7
SM065	Agile Practices in Global Software Engineering - A Systematic Map	7
SM066	Agile Principles and Ethical Conduct	7
SM067	Agile Principles as a Leadership Value System: How Agile Memes Survive and Thrive in a Corporate IT Culture	3
SM068	Agile product line planning A collaborative approach and a case study	7
SM069	Agile Program Management: Lessons Learned from the VeriSign Managed Security Services Team	3
SM070	Agile Project Experiences The Story of Three Little Pigs	3
SM071	Agile Project Leadership My Top 10 Value Driven Principles	3
SM072	Agile project management: steering from the edges	7
SM073	Agile Software Assurance: An Empirical Study	7
SM074	Agile Software Development Methodologies and Practices	7
SM075	Agile Software Development, The Business of Innovation	7
SM076	Agile software development, the people factor	3
SM077	Agile Software Development: It's about Feedback and Change	7
SM078	Agile software development: the business of innovation	7 ; 5
SM079	Agile Supports Improved Culture and Quality for Healthwise	7
SM080	Agile Systems Development and Stakeholder Satisfaction : A South African Empirical Study	7
SM081	Agile Teams Require Agile QA: How to Make it Work, An Experience Report	3
SM082	Agile user centered design: enter the design studio - a case study	7
SM083	Agile with Fragile Large Legacy Applications	3
SM084	Agile Won't Work: Implementing Agility in Non-standard Teams	7
SM085	Agile Won't Work: Implementing Agility in Non-standard Teams	7; 5
SM086	Agile: Adopting a New Methodology at Harvard Business School	3
SM087	Agility and the Inconceivably Large	3
SM088	Agility in Context	7
SM089	Agility in Context	7; 5
SM090	Agility versus Discipline: Is Reconciliation Possible?	7

SM091	Aligning strategic planning with agile development: extending agile thinking to business improvement	3
SM092	An adaptive software development process model	7
SM093	An Agile Approach to Proposition Assessment	3; 7
SM094	An Agile Classroom Experience: Teaching TDD and Refactoring	3
SM095	An Agile Development Team's Quest for CMMI® Maturity Level 5	3
SM096	An Analytical Survey of "On-Site Customer" Practice in Extreme Programming	7
SM097	An empirical study of system design instability metric and design evolution in an agile software process	7
SM098	An empirical study of using planning poker for user story estimation	7
SM099	An empirical study on software engineers motivational factors	7
SM100	An Empirical Study on Software Engineers Motivational Factors	7; 5
SM101	An empirical study on the relationship between the use of agile practices and the success of Scrum projects	7
SM102	An Empirical Study on the Relationship between the Use of Agile Practices and the Success of Scrum Projects	5
SM104	An environment for collaborative iteration planning	4
SM106	An evaluation of the degree of agility in six agile methods and its applicability for method engineering	7
SM107	An evaluation of the degree of agility in six agile methods and its applicability for method engineering	7; 5
SM108	An Industrial Case Study on Introducing XP in a Complex Software Development Environment	8
SM109	An institutional analysis of software teams	3
SM110	An Iterative Approach for Development of Safety-Critical Software and Safety Arguments	3; 7
SM111	Analyse Changing Risk of Organizational Factors in Agile Project Management	7
SM112	Analyzing the modes of communication in agile practices	7
SM113	Antecedents and consequences of team potency in software development projects	7
SM114	Architects as Service Providers	7
SM115	Are Developers Complying with the Process: An XP Study	7
SM116	Are we ready to be unleashed? A comparative analysis between agile software development and war fighting	3
SM117	Are we ready to be unleashed? A comparative analysis between agile software development and war fighting	3; 5
SM118	Are You Sure? Really? A Contextual Approach to Agile User Research	3; 7
SM120	Autonomy and Teamwork in Innovative Projects	5
SM122	Back to Basics: The Role of Agile Principles in Success with an Distributed Scrum Team	3

SM123	Balancing acts: Walking the Agile tightrope	5
SM126	Balancing the human and the engineering factors in software development	7
SM127	Behind the rules: XP experiences	3
SM128	Bootstrapping Scrum and XP under Crisis A Story from the Trenches	3
SM130	Building blocks for self-organizing software development teams: A framework model and empirical pilot study	5
SM131	Building reliable systems based on self-organizing multi-agent systems	7
SM132	Building the linkage between project managers' personality and success of software projects	7
SM133	Capitalizing on Empirical Evidence during Agile Adoption	3
SM134	Career Orientation and the Global IT Workforce: Research in Progress	7
SM135	Certifying for CMM level 2 and ISO9001 with XP@Scrum	7
SM136	Challenges of global software development	7
SM137	Challenges of Migrating to Agile Methodologies	3
SM138	Challenging the status quo: What motivates proactive behaviour?	7
SM139	Change your organization (for Peons)	3
SM141	Coding Dojo: An Environment for Learning and Sharing Agile Practices	3
SM143	Coevolving Systems and the Organization of Agile Software Development	5
SM144	Collaboration and co-ordination in mature eXtreme programming teams	7
SM145	Collaborative and social aspects of software development	7
SM146	Colossal, Scattered, and Chaotic (Planning with a Large Distributed Team)	3
SM147	Combining Agile Methods with Stage-Gate Project Management	7
SM148	Communication and co-ordination practices in software engineering projects	7
SM149	Comparative analysis of job satisfaction in agile and non-agile software development teams	7
SM150	Complementing Measurements and Real Options Concepts to Support Inter-iteration Decision-Making in Agile Projects	7
SM151	Consciously Evolving an Agile Team	3
SM153	Contextualizing Knowledge Management Readiness to Support Change Management Strategies	7
SM154	Coordinating Mechanisms for Agile Global Software Development	9
SM155	Coordination implications of software architecture in a global	7

	software development project	
SM156	Coordination in innovative design and engineering: observations from a lunar robotics project	7
SM157	Costs of compliance: agile in an inelastic organization	7
SM158	Costs of compliance: agile in an inelastic organization	7; 5
SM159	Covert Agile: Development at the Speed of… Government?	3
SM160	Cowboys and Indians: Impacts of cultural diversity on agile teams	7; 5
SM161	Cowboys and Indians: Impacts of Cultural Diversity on Agile Teams	5
SM162	Cowboys and Indians: Impacts of cultural diversity on agile teams	7
SM163	Crawl, Walk, Run: 4 Years of Agile Adoption at BabyCenter.com	3; 7
SM164	Creating a dual-agility method: The value of method engineering	7
SM165	Creating Agile Streams for Business & Technical Value	3
SM166	Creativity in innovative projects: How teamwork matters	7
SM167	Cross-continent development using scrum and XP	7
SM168	Customer relationships and Extreme Programming	7
SM169	Daily iterations: approaching code freeze and half the team is not agile	3
SM170	DecisionSpace infrastructure: agile development in a large, distributed team	3
SM171	Dependency Management in a Large Agile Environment	3
SM172	Descending from the Architect's Ivory Tower	7
SM173	Descending from the Architect's Ivory Tower	7; 5
SM174	Design of self-managing dependable systems with UML and Fault Tolerance Patterns	7
SM175	Developer autonomy in the FreeBSD open source project	7
SM176	Developing a Theory of the Strategic Core of Teams: A Role Composition Model of Team Performance	7
SM177	Development of a team measure for tacit knowledge in software development teams	7
SM178	Direct verbal communication as a catalyst of agile knowledge sharing	7
SM179	Disciplined and free-spirited 'Time-out behaviour' at the Agile conference	7
SM180	Distributed Agile: Growing a Practice Together	7
SM181	Distributed agile: project management in a global environment	7
SM182	Distributed Multilevel Data Fusion for Networked Embedded Systems	7
SM183	Distributed Scrum : Agile Project Management with Outsourced Development Teams	7
SM184	Do Agile Methods Marginalize Problem Solvers?	7
SM186	Does the XP environment meet the motivational needs of the	5

	software developer? An empirical study	
SM187	Does the XP environment meet the motivational needs of the software developer? An empirical study	5
SM188	Drifting Toward Invisibility: The Transition to the Electronic Task Board	3
SM189	Driving Process Improvement via Comparative Agility Assessment	7
SM190	Driving Process Improvement via Comparative Agility Assessment	7; 5
SM191	Dynamic team structures for supporting software design episodes	7
SM192	Early community building: a critical success factor for XP projects	3
SM193	Effects of agile practices on social factors	7
SM194	Effects of agile practices on social factors	5
SM195	Effects of Agile Practices on Social Factors	3; 5
SM196	Effects of agile practices on social factors.	5
SM198	Effects of participative management on the performance of software development teams	7
SM199	Eleven Guidelines for Implementing Pair Programming in the Classroom	3
SM200	Embedded agile project by the numbers with newbies	3; 7
SM201	Embedded agile project by the numbers with newbies	7; 3; 5
SM202	Emergence of new project teams from open source software developer networks: Impact of prior collaboration ties	7
SM203	Empirical studies of agile software development : A systematic review	5
SM204	Empirical studies of agile software development: A systematic review	5
SM205	Empirical studies of agile software development: A systematic review	5
SM207	Empirical study of how personality, team processes and task characteristics relate to satisfaction and software quality	7
SM208	Empirical Validation of Three Software Metrics Suites to Predict Fault-Proneness of Object-Oriented Classes Developed Using Highly Iterative or Agile Software Development Processes	7
SM209	Enabling Agile in a Large Organization Our Journey Down the Yellow Brick Road	3
SM210	Engineering the Software for Understanding Climate Change	7
SM211	Engineering-based processes and agile methodologies for software development a comparative case study	7
SM212	Enhanced agile software development — hybrid paradigm with LEAN practice	7
SM213	Enterprise Agile Transformation: The Two-Year Wall	3
SM214	Establishing the agile PMO: Managing variability across projects and	7

	portfolios	
SM215	Establishing the Agile PMO: Managing variability across Projects and Portfolios	7; 5
SM216	Evolution of the Tools and Practices of a Large Distributed Agile Team	3
SM217	Evolving into Embedded Develop	3
SM218	Examining team cohesion as an effect of software engineering methodology	7
SM219	Examining the Foundations of Agile Usability with eXtreme Scenario-Based Design	7
SM220	Executing Agile in a Structured Organization: Government	3
SM221	Experience Report 'Offshore XP for PDA development'	3
SM222	Experience Report: Design to Delivery in 7 Weeks	3
SM223	Experience Report: Extremely Short Iterations as a Catalyst for Effective Prioritization of Work	3
SM224	Experience Report: The Social Nature of Agile Teams	5
SM225	Experience report: The social nature of agile teams	7
SM226	Experience with Training a Remotely Located Performance Test Team in a Quasi-Agile Global Environment	7
SM227	Experiences Applying Agile Practices to Large Systems	3
SM228	Experimental analysis of self-organizing team’s behaviors	7
SM229	Exploring extreme programming in context: an industrial case study	7
SM230	Exploring Motivational Differences between Software Developers and Project Managers	7
SM231	Exploring Synergistic Impact through Adventures in Group Pairing	8
SM232	Exploring the impact of team mental models on information utilization and project performance in system development	7
SM233	Extreme Product Line Engineering: Managing Variability and Traceability via Executable Specifications	7
SM234	Fast & Predictable A Lightweight Release Framework Promotes Agility through Rhythm and Flow	3
SM235	Feature Teams Collaboratively Building Products from READY to DONE	3
SM236	Finding a place for discount usability engineering in agile development: throwing down the gauntlet	6; 7
SM237	FirmNet: The scope of firms and the allocation of task in a knowledge-based economy	7
SM238	Follow the sun: distributed extreme programming development	3
SM239	For the Special issue on Qualitative Software Engineering Research	7
SM240	Formalizing agility, part 2: how an agile organization embraced the CMMI	3
SM241	Formalizing agility: an agile organization's journey toward CMMI	3

	accreditation	
SM242	Forming effective worker teams with multi-functional skill requirements	7
SM243	Forming effective worker teams with multi-functional skill requirements	7; 5
SM244	Forming successful extreme programming teams	7
SM245	Forming to Performing: The Evolution of an Agile Team	3
SM246	Forming to Performing: The Evolution of an Agile Team	3; 5
SM247	Forming to Performing: Transitioning Large-Scale Project Into Agile	3
SM248	Forming to Performing: Transitioning Large-Scale Project Into Agile	3; 5
SM249	Fostering creativity thinking in agile software development	7
SM250	From Anarchy to Sustainable Development: Scrum in Less than Ideal Conditions	3
SM251	From Cards to Code: How ExtremeProgramming Re-Embodies Programming as aCollective Practice	7
SM252	From CMMI and Isolation to Scrum, Agile, Lean and Collaboration	3
SM253	From Cradle to Sprint: Creating a Full-Lifecycle Request Pipeline at Nationwide Insurance	4
SM254	From Waterfall to Agile - How does a QA Team Transition?	3; 7
SM255	Fully Distributed Scrum : The Secret Sauce for Hyperproductive Offshored Development Teams	7
SM256	Fully Distributed Scrum: Linear Scalability of Production between San Francisco and India	7
SM257	Fully Distributed Scrum: The Secret Sauce for Hyperproductive Offshored Development Teams	7
SM258	Functional group roles in software engineering teams	7
SM259	Future Implementation and Integration of Agile Methods in Software Development and Testing	7
SM261	GET YOUR AGILE FREAK ON! Agile Adoption at Yahoo! Music	3
SM263	Governance of an agile software project	5
SM264	Group spontaneity	6
SM265	Groupware extension for a software-project management system	7
SM266	Growing a Build Management System from Seed	3
SM267	Growing an Agile Culture from Value Seeds	3
SM268	Growing an Agile Culture from Value Seeds	3; 5
SM269	Growing and Sustaining an Offshore Scrum Engagement	3
SM270	Growing PMI® Using Agile	7
SM271	Growing PMI® Using Agile	7; 5
SM272	Hidden gap in agile methods adoption	8
SM273	Hook, line and sinker: the role of line management in relation to Agile teams	7

SM274	Hook, Line and Sinker: The Role of Line Management in Relation to Agile Teams	7; 5
SM275	How agile are industrial software development practices	7
SM276	How Agile COTS Selection Methods are (and can be)?	7
SM277	How Being Agile Changed Our Human Resources Policies	3; 7
SM278	How Did We Adapt Agile Processes to Our Distributed Development?	7
SM279	How do personality, team processes and task characteristics relate to job satisfaction and software quality?	5
SM281	How do personality, team processes and task characteristics relate to job satisfaction and software quality?	5
SM282	How do personality, team processes and task characteristics relate to job satisfaction and software quality?	5
SM283	How does readiness for agile development relate to team climate and individual personality attributes?	7
SM284	How technological support can enable advantages of agile software development in a GSE setting	9
SM285	How We Made Onsite Customer Work - An Extreme Success Story	3
SM286	Identification in organizations: The role of self-concept orientations and identification motives	7
SM287	Identifying some important success factors in adopting agile software development practices	5
SM288	Identifying some important success factors in adopting agile software development practices	7
SM289	Identifying some important success factors in adopting agile software development practices	7; 5
SM290	Implementing a Professional Services Organization Using Type C Scrum	3
SM291	Implementing a Professional Services Organization Using Type C Scrum	3; 5
SM292	Implementing Scrum in a Distributed Software Development Organization	7
SM293	Implementing Scrum in a Distributed Software Development Organization	7; 5
SM294	Improving agile software development by the application of method engineering practices	8
SM295	Improving agile software development by the application of method engineering practices	8; 5
SM296	Improving agile team learning by improving team reflections [agile software development]	3
SM297	Improving General Knowledge in Agile Software Organizations: Experiences with Job Rotation in Customer Support	7; 5

SM298	Improving general knowledge in agile software organizations: Experiences with job rotation in customer support	7
SM299	Improving Obama Campaign Software: Learning from Users	3
SM300	Improving the interface between business and product development using agile practices and the cycles of control framework	3; 7
SM301	Incorporating social software into distributed agile development environments	7
SM302	Incremental Adoption of Scrum for Successful Delivery of an IT Project in a Remote Setup	3
SM303	Influence of Large-Scale Organization Structures on Leadership Behaviors	3
SM304	Insights into an Agile Adventure with Offshore Partners	3
SM305	Integrated theoretical model for building effective teams	6
SM306	Integrating agile software development and software process improvement: a longitudinal case study	7
SM307	Integrating agile software development into stage-gate managed product development	7
SM308	Integrating Scrum with the Process Framework at Yahoo! Europe	3
SM309	Internet collaboration and service composition as a loose form of teamwork	7
SM310	Introducing agile development (XP) into a corporate Webmaster environment - an experience report	7
SM311	Introducing agile development (XP) into a corporate Webmaster environment - an experience report	7; 5
SM312	Introducing agile development into bioinformatics: an experience report	3; 6; 7
SM313	Introducing agile into a software development Capstone project	3
SM314	Introducing an Agile Method for Enterprise Mash-Up Component Development	7
SM315	Introducing lean principles with agile practices at a Fortune 500 company	3
SM316	Introduction to Agile Methods	7
SM317	Investigating Knowledge Management practices in software development organisations – An Australian experience	7
SM318	Is the Most Effective Team Leadership Shared? The Impact of Shared Leadership, Age Diversity, and Coordination on Team Performance	6
SM319	Is the Most Effective Team Leadership Shared? The Impact of Shared Leadership, Age Diversity, and Coordination on Team Performance	6; 5
SM320	IT knowledge integration capability and team performance: The role of team climate	7
SM321	IT self-employed workers between constraint and flexibility	7
SM322	Iterating a Team in Flux	3

SM323	Iteration advocate/iteration transition meeting: small sampling of new agile techniques used at a major telecommunications firm	3
SM324	It's more than just toys and food: leading agile development in an enterprise-class start-up	3
SM325	It's more than just toys and food: leading agile development in an enterprise-class start-up	5; 3; 7
SM326	Job Characteristics And Quality Of Working Life In The It Workforce: The Role Of Gender	7
SM327	Job satisfaction among information system (IS) personnel	7
SM328	Job Satisfaction and Motivation in a Large Agile Team	5
SM330	Job satisfaction and motivation in a large agile team	5
SM331	Large Build Teams: Help or Hindrance?	3
SM332	Large Scale Agile Transformation in an On-Demand World	3
SM333	Large Scale Agile Transformation in an On-Demand World	3; 5
SM334	Leadership and trust: Their effect on knowledge sharing and team performance	6
SM335	Leadership of information systems development projects	7
SM336	Leadership style and team processes in self-managed teams	7
SM337	Leading Manual Test Efforts with Agile Methods	7
SM338	Lean Software Development: An Agile Toolkit	2
SM339	Learning from Experience: The Analysis of an Extreme Programming Process	3
SM340	Learning teamwork skills in university programming courses	7
SM341	Lessons learned from an extremely distributed project	3
SM342	Lightweight management - Taming the RoboCup development process	7
SM343	Management Challenges to Implementing Agile Processes in Traditional Development Organizations	7
SM344	Managing a Large "Agile" Software Engineering Organization	3
SM345	MANAGING HIGH ACHIEVING INFORMATION SYSTEMS PROFESSIONALS	7
SM347	Managing radical software engineers: between order and chaos	7
SM348	Managing viable virtual communities: an exploratory case study and explanatory model	7
SM349	Marriott's Agile Turnaround	3
SM350	Mature Agile with a Twist of CMMI	3
SM351	Method and developer characteristics for effective agile method tailoring: A study of XP expert opinion	7
SM352	Metrics functions for kanban guards	7
SM353	Metrics Functions for Kanban Guards	7; 5

SM354	Metrics Functions for Kanban Guards	7
SM355	Metrics Functions for Kanban Guards	7; 5
SM356	Migrating from SharePoint to a Better Scrum Tool	3
SM357	Minding the gaps: Understanding technology interdependence and coordination in knowledge work	7
SM358	Modified agile practices for outsourced software projects	7
SM359	Modified agile practices for outsourced software projects	7; 5
SM360	Motivated humans for reliable software products	7
SM361	Motivating IT Professionals - The Hong Kong Challenge	7
SM362	Motivating job design as a factor in open source governance	7
SM363	Motivation and cohesion in agile teams	7
SM364	Moving from Waterfall to Agile	3
SM365	Multi-agent system: A guiding metaphor for the organization of software development projects	7
SM366	Multiple simultaneous projects with one extreme programming team	3
SM367	MyBoeingFleet and Agile Software Development	3
SM368	Observations on balancing discipline and agility	7
SM370	Octopus : Agile software development facing our imperfect world	3
SM371	Offshore Agile Maintenance	7
SM372	On the Impact of a Collaborative Pedagogy on African American Millennial Students in Software Engineering	7
SM373	One XP experience: introducing agile (XP) software development into a culture that is willing but not ready	7
SM374	Ongoing quality improvement, or: how we all learned to trust XP	3
SM375	Optimizing major release frequency using agile practices	3
SM376	Organically Growing Internal Coaches	3
SM377	Organisational culture and XP: three case studies	7
SM379	Organizing self-organizing teams	5
SM380	Other Software Development Approaches	8
SM381	Over the Waterfall in a Barrel - MSIT Adventures in Scrum	3
SM383	Overcoming barriers to self-management in software teams	5
SM384	Overcoming Brooks' Law	3
SM385	Overcoming Fear with Trust	3
SM386	Overcoming the Challenges of Building a Distributed Agile Organization	3
SM387	Overhauling a Failed Project Using Out of the Box Scrum	3
SM388	Participatory design activities and agile software development	7
SM389	Patterns for agile development practice part 3	7
SM390	Perceived productivity threats in large agile development projects	7

SM391	Perceived Productivity Threats in Large Agile Development Projects	5
SM393	Personality characteristics in an XP team: a repertory grid study	7
SM394	Planning and Facilitating Release Retrospectives	3
SM395	Practical Considerations for Distributed Agile Projects	3
SM396	Practices of an Agile Team	3
SM397	Predicting teleworker success: An exploration of personality, motivational, situational, and job characteristics	7
SM398	Programming Task Demands	3
SM399	Promiscuous pairing and beginner's mind: embrace inexperience [agile programming]	3
SM401	Quality of Working Life and Turnover Intention in Information Technology Work	7
SM402	Rapid Software Development through Team Collocation	7
SM403	Reactive Variability Management in Agile Software Development	7
SM404	Refactoring the development process: experiences with the incremental adoption of agile practices	3
SM405	Reflections on reflection in agile software development	7
SM406	Remote Mentoring a Distributed Agile Team	3
SM407	Ript: Innovation and Collective Product Ownership	3
SM408	Ript: Innovation and Collective Product Ownership	3; 5
SM409	Risk identification and mitigation processes for using scrum in global software development: A conceptual framework	7
SM410	Roadmap Transformation: From Obstacle to Catalyst	3
SM411	Rolling out Agile in a Large Enterprise	3
SM412	Rubber ducks, nightmares, and unsaturated predicates: proto-scientific schemata are good for agile	7
SM413	Scaling agile methods	7
SM414	Scaling agile software development through lean governance	7
SM415	Scaling agile software development through lean governance	7; 5
SM416	Scaling Agile: Finding your Agile Tribe	3
SM417	Scaling Agile: Finding your Agile Tribe	3; 5
SM418	Scaling Product Ownership	3
SM419	Scaling Up Pushing Scrum out of its Comfort Zone	3
SM420	Scrum 911! Using Scrum to Overhaul a Support Organization	7
SM421	Scrum 911! Using Scrum to Overhaul a Support Organization	7; 5
SM422	Scrum and CMMI Going from Good to Great	3
SM423	Scrum and CMMI Level 5 : The Magic Potion for Code Warriors	7
SM424	Scrum and CMMI Level 5: The Magic Potion for Code Warriors	3; 7
SM425	SCRUM and Productivity in Software Projects: A Systematic Literature Review	7

SM427	Scrum at a Fortune 500 Manufacturing Company	7
SM429	Scrum in a Multiproject Environment: An Ethnographically-Inspired Case Study on the Adoption Challenges	5
SM431	Seamless long term learning in agile teams for sustainable leadership	7
SM432	Self-determination theory and work motivation	7
SM433	Self-organization of teams for free/libre open source software development	7; 5
SM434	Self-organization of teams for free/libre open source software development	7
SM435	Self-organization of teams for freelibre open source software development	7
SM436	Shock Therapy: A Bootstrap for Hyper-Productive Scrum	7
SM437	Shock Therapy: A Bootstrap for Hyper-Productive Scrum	7; 5
SM438	Small is beautiful: A study of packaged software development teams	7
SM439	Social behaviors on XP and non-XP teams: a comparative study	7
SM440	Social behaviors on XP and non-XP teams: a comparative study	7; 5
SM441	Software developer perceptions about software project failure: a case study	7
SM442	Software developer perceptions about software project failure: a case study	7; 5
SM443	Software Development Methodology for Fast Changing Environment	7
SM444	Software Process Improvement as organizational change A metaphorical analysis of the literature	7
SM445	Software project managers and project success An exploratory study	7
SM446	Software release management for component-based software	7
SM447	SOX Compliant Agile Processes	3
SM448	SPI - A Role for Method Engineering	7
SM449	Stakeholder Identification in Agile Software Product Development Organizations: A Model for Understanding Who and What Really Counts	7
SM450	State of the Art' in Using Agile Methods for Embedded Systems Development	7
SM451	Staying agile in government software projects	3
SM452	Stories, Sketches, and Lists: Developers and Interaction Designers Interacting Through Artefacts	7
SM453	Successful Integration of Agile Development Techniques within DISA	7
SM454	Successful Integration of Agile Development Techniques within DISA	7; 5
SM455	Supporting Program Comprehension in Agile with Links to User Stories	7
SM456	Take No Prisoners How a Venture Capital Group Does Scrum	7
SM457	Teaching Agile Software Development: A Case Study	7

SM458	Teaching Agile Software Development: A Case Study	7; 5
SM459	Teaching disciplined software development	3
SM460	Team leader strategies for enabling collaboration technology adaptation: Team technology knowledge to improve globally distributed systems development work	7
SM461	**Team Self-Managing Behaviors and Team Effectiveness: The Moderating Effect of Task Routineness**	**6**
SM462	Team SPICE: A SPICE-based teamwork assessment model	7
SM463	Team structure and team performance in IS development: a social network perspective	7
SM464	Team-based work and work system balance in the context of agile manufacturing	7
SM465	Team-RUP: Agent-based simulation of team behaviour in software development organisations	7
SM466	Teams' reactions to conflict and teams' task and social outcomes: The moderating role of transformational and emotional leadership	6
SM467	Teamwork and associated psychological factors: A review	7
SM468	Technical lessons Learned Turning the Agile Dials to Eleven!	3
SM469	The A,B,C's of Scrum, Course at Scrum Gathering	4
SM470	The Accidental Agilists: One Team's Journey from Waterfall to Agile	7
SM471	The agile marathon	3
SM472	The Agile Maturity Map A Goal Oriented Approach to Agile Improvement	3
SM473	The agile requirements refinery Applying SCRUM principles to software product management	7
SM474	The Amazing Team Race A Team Based Agile Adoption	7
SM475	The Amazing Team Race A Team Based Agile Adoption	7; 5
SM476	The Big Projects Always Fail: Taking an Enterprise Agile	3
SM477	The Bold, New Extreme Programming Experiment Now in its Ninth Year	7
SM478	The Bold, New Extreme Programming Experiment Now in its Ninth Year	7; 5
SM479	The cost of code quality	3; 7
SM480	The Covert Agilist	3
SM481	The Design Studio: Interface Design for Agile Teams	3
SM482	The effect of intrinsic and extrinsic rewards for developers on information systems project success	7
SM483	The Failure of the Off-shore Experiment: A Case for Collocated Agile Teams	3
SM484	The FrAgile Organisation	3
SM485	The Good and Bad of Agile Offshore Development	3
SM486	The Growth of an Agile Coach Community at a Fortune 200	7

	Company	
SM487	The Growth of an Agile Coach Community at a Fortune 200 Company	7; 5
SM488	The Impact of Agile Methods on Software Project Management	3
SM489	The impact of agile practices on communication in software development	7
SM490	The impact of an agile methodology on the well being of development teams	7
SM491	The impact of an agile methodology on the well being of development teams	7; 5
SM492	The Impact of Organization, Project and Governance Variables on Software Quality and Project Success	7
SM494	The impact of the Abilene Paradox on double-loop learning in an agile team	5
SM495	The impact of the Abilene Paradox on double-loop learning in an agile team	5
SM496	The impact of the Abilene Paradox on double-loop learning in an agile team	5
SM497	The Importance of Identity and Vision to User Experience Designers on Agile Projects	6; 7
SM498	The influence of organizational culture on the adoption of extreme programming	7
SM499	The influence of organizational culture on the adoption of extreme programming	7
SM500	The Inkubook Experience: A Tale of Five Processes	3
SM501	The knowledge management efficacy of matching information systems development methodologies with application characteristics—an experimental study	7
SM502	THE LEARNING CIRCLE: An Evolution of Agile for Learning Environments	3
SM503	The Matrix of Control: Combining Process and Structure Approaches to Managing Software Development	8
SM504	The Price of Agile Is Eternal Vigilance	3; 7
SM505	The Relationship between Customer Collaboration and Software Project Overruns	7
SM507	The Role of Face-to-Face Meetings in Technology-Supported Self-Organizing Distributed Teams	3
SM508	The Role of Incremental Change in Agile Software Processes	7
SM509	The role of physical artefacts in agile software development: Two complementary perspectives	7
SM510	The role of physical artefacts in agile software development: Two complementary perspectives	7; 5

SM512	The role of team behavioral integration and cohesion in shaping individual improvisation	7
SM513	The Rollercoaster of Required Agile Transition	3
SM514	The social nature of agile teams	5
SM515	The Social Nature of Agile Teams	5
SM517	The social nature of agile teams	5
SM518	The Toolbox of a Successful Software Craftsman	7
SM519	The XP Customer Team: A Grounded Theory	7
SM520	Theoretical reflections on agile development methodologies	7
SM521	There has to be a better way! [software development]	7
SM522	Thoughts on weak links and Alexandrian life in Scrum	7
SM523	Three-way cultural change: introducing agile within two non-agile companies and a non-agile methodology	3
SM524	Tidal Wave: The Games Transformation	3
SM525	Time Management Behavior as a Moderator for the Job Demand—Control Interaction	7
SM526	Toward a conceptual framework of agile methods: a study of agility in different disciplines	7
SM527	Toward a Framework for Evaluating Extreme Programming	7
SM529	Toward a Systems Theory of Motivated Behavior in Work Teams	6
SM530	Toward agile: An integrated analysis of quantitative and qualitative field data on software development agility	7
SM531	Towards a Framework for Agile Management Based on Chaos and Complex System Theories	7
SM532	Towards Extreme(ly) Usable Software: Exploring Tensions Between Usability and Agile Software Development	7
SM533	Training students to be agile information systems developers: a pedagogical approach	7
SM535	Transition from a Plan-Driven Process to Scrum – A Longitudinal Case Study on Software Quality	5
SM536	Transitioning a Team to Agile Test Methods	3
SM537	Transitioning from Agile Development to Enterprise Product Management Agility	3
SM538	Trouble in Paradise: the Open Source Project PyPy, EU-Funding and Agile Practices	7
SM539	Two Case Studies of User Experience Design and Agile Development	6
SM540	Un-Assessments Reflections by the Team, for the Team	7
SM542	Understanding self-organizing teams in agile software development	5
SM545	Unfixing the fixed scope project: using agile methodologies to create flexibility in project scope	3; 7
SM546	Usage and Perceptions of Agile Software Development in an Industrial Context: An Exploratory Study	7

SM547	U-SCRUM: An Agile Methodology for Promoting Usability	3
SM548	Use of Agile Methods in Software Engineering Education	7
SM550	Using agile software process with offshore development	3
SM551	Using competition to build a stronger team	3
SM552	Using Factor Analysis to Generate Clusters of Agile Practices (A Guide for Agile Process Improvement)	7
SM553	Using open spaces to resolve cross team issue [software development]	3
SM554	Using open spaces to resolve cross team issue [software development]	3; 5
SM555	Using Opinion Polls to Help Measure Business Impact in Agile Development	7
SM556	Using Persona with XP at LANDesk Software, an Avocent Company	3
SM557	Using risk to balance agile and plan- driven methods	7
SM558	Using Scrum in Global Software Development: A Systematic Literature Review	7
SM559	Using Scrum to guide the execution of software process improvement in small organizations	7
SM560	Using Scrum to guide the execution of software process improvement in small organizations	5
SM561	Utilizing Digital Tabletops in Collocated Agile Planning Meetings	7
SM562	Virtual team effectiveness using dyadic teams	7
SM563	We Are Naked Volunteers: How an Agile Users Group Rediscovered Itself	3
SM564	We Didn't Quite Get It	3
SM565	Weaponized Scrum	3; 7
SM566	What Are We Arguing About? A Framework for Defining Agile in our Organization	7
SM567	What are we arguing about? A framework for defining agile in our organization	7; 5
SM568	What do software practitioners really think about project success: an exploratory study	7
SM569	What lessons can the agile community learn from a maverick fighter pilot?	7
SM570	Which reduces IT turnover intention the most: Workplace characteristics or job characteristics?	7
SM571	Which reduces IT turnover intention the most: Workplace characteristics or job characteristics?	7; 5
SM572	XP and Junior Developers: 7 Mistakes (and how to avoid them)	3
SM574	XP Customer Practices: A Grounded Theory	7
SM575	Yahoo! Distributed Agile: Notes from the World Over	7
SM576	Yahoo! Distributed Agile: Notes from the World Over	7; 5