UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

EDUARDO SAKAI

GESTÃO DO CONHECIMENTO NO DESENVOLVIMENTO ÁGIL DE SOFTWARE:
TÉCNICAS E FATORES QUE PROMOVEM A CRIAÇÃO DE CONHECIMENTO

KNOWLEDGE MANAGEMENT IN AGILE SOFTWARE DEVELOPMENT:
TECHNIQUES AND FACTORS THAT PROMOTE KNOWLEDGE CREATION

CAMPINAS
2016

EDUARDO SAKAI

GESTÃO DO CONHECIMENTO NO DESENVOLVIMENTO ÁGIL DE SOFTWARE: TÉCNICAS E FATORES QUE PROMOVEM A CRIAÇÃO DE CONHECIMENTO

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica, na área de Engenharia da Computação.

Dissertation presented to the School of Electrical and Computer Engineering of the University of Campinas in partial fulfillment of the requirements for the degree of Master in the area of Electrical Engineering, in the area of Computer Engineering.

Supervisor/Orientador: Prof. Dr. Ivan Luiz Marques Ricarte

ESTE EXEMPLAR CORRESPONDE À VERSÃO FINAL DA DISSERTAÇÃO DEFENDIDA PELO ALUNO EDUARDO SAKAI E ORIENTADA PELO PROF. DR. IVAN LUIZ MARQUES RICARTE

CAMPINAS
2016

Agência(s) de fomento e nº(s) de processo(s): CAPES

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Luciana Pietrosanto Milla - CRB 8/8129

Sa29g

Sakai, Eduardo, 1981-
Gestão do conhecimento no desenvolvimento ágil de software : técnicas e fatores que promovem a criação de conhecimento / Eduardo Sakai. – Campinas, SP : [s.n.], 2016.

Orientador: Ivan Luiz Marques Ricarte.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Gestão do conhecimento. 2. Aprendizagem organizacional. 3. Engenharia de software. 4. Desenvolvimento ágil de software. 5. Teoria fundamentada em dados. I. Ricarte, Ivan Luiz Marques,1962-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Knowledge management in agile software development : techniques and factors that promote knowledge creation
Palavras-chave em inglês:
Knowledge management
Organizational learning
Software engineering
Agile software development
Grounded theory
Área de concentração: Engenharia de Computação
Titulação: Mestre em Engenharia Elétrica
Banca examinadora:
Ivan Luiz Marques Ricarte [Orientador]
Regina Lúcia de Oliveira Moraes
Mario Jino
Data de defesa: 23-05-2016
Programa de Pós-Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - DISSERTAÇÃO DE MESTRADO

Candidato: Eduardo Sakai RA: 043120

Data da Defesa: 23 de maio de 2016

Título da Tese: "Gestão do Conhecimento no Desenvolvimento Ágil de Software: *Técnicas e Fatores que Promovem a Criação de Conhecimento*"

Prof. Dr. Ivan Luiz Marques Ricarte (Presidente, FEEC/UNICAMP)

Prof. Dra. Regina Lúcia de Oliveira Moraes (FT/UNICAMP)

Prof. Dr. Mario Jino (FEEC/UNICAMP)

A ata de defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no processo de vida acadêmica do aluno.

À minha mãe e ao meu pai.

AGRADECIMENTOS

Agradeço especialmente ao meu mestre e amigo Ivan Ricarte.

À minha mãe, Maria Izabel, ao meu pai, Kenji, e a minha irmã, Karen, por todo o suporte.

À minha companheira Sheila, por todo amor, pela paciência e pelos conselhos.

Ao meu amigo Andrei, pelas longas discussões e trocas de ideias.

Ao meu amigo Leandro, ao José, ao Valmir, ao Caio e ao Claudinei, pela colaboração.

À Sensedia e à Clickideia, por tornarem esta pesquisa possível.

À Dona Vilma, do Colégio Porto Seguro, e ao Prof. Souza, do Cursinho Objetivo, pelas oportunidades.

À CAPES, pela ajuda financeira.

Aos meus amigos taboenses.

Aos meus amigos paulistanos.

Aos meus amigos campineiros.

Aos meus amigos cosmopolenses.

Aos meus amigos da Universidade Estadual de Campinas.

Aos meus amigos da Universidade de São Paulo.

Aos meus amigos da Universidade Estadual Paulista.

Aos meus amigos do DCA.

Aos meus amigos do grupo de pesquisa.

Aos meus amigos da FEEC.

À Universidade Estadual de Campinas

À cidade de Campinas.

“Aprender é a única coisa de que a mente nunca se cansa, nunca tem medo e nunca se arrepende.”

(Leonardo da Vinci)

RESUMO

SAKAI, Eduardo. **Gestão do conhecimento no desenvolvimento ágil de software:** *Técnicas e fatores que promovem a criação de conhecimento.* Faculdade de Engenharia Elétrica e Computação, Universidade Estadual de Campinas, Campinas, 2016. (Dissertação de mestrado apresentada à Faculdade de Engenharia Elétrica e Computação).

O processo de desenvolvimento de software pode ser considerado um processo intensivo de criação de conhecimento. Os membros de equipes ágeis de desenvolvimento de software são trabalhadores que compartilham intensivamente seus conhecimentos tácitos e explícitos para a construção de software inovadores. Esse conhecimento, inicialmente distribuído nas mentes dos programadores, deve ser gerenciado por meio de técnicas e ferramentas que apoiem a aprendizagem organizacional, a criação, a construção e a avaliação de software. Nesse sentido, este trabalho objetivou identificar e compreender as técnicas e os fatores capacitadores que auxiliem a criação de conhecimento em equipes ágeis de desenvolvimento de software.

Para atingir aos propósitos desta pesquisa, primeiro, foi feita uma revisão de literatura, que seguiu os princípios de revisões sistemáticas, para compreender o Estado da Arte e identificar técnicas das metodologias ágeis e fatores que promovem a criação de conhecimento no desenvolvimento ágil de software. Segundo, para o percurso metodológico dois estudos de caso foram conduzidos por meio de entrevistas e analisados qualitativamente por meio da Teoria Fundamentada em Dados.

Como resultados, dois fenômenos centrais emergiram dos dados analisados. O Fenômeno 1 mostra que as equipes ágeis estabelecem técnicas que apoiam a aprendizagem organizacional, a criação e a construção de software. O Fenômeno 2 indica que as equipes ágeis promovem fatores capacitadores que alinham a cultura organizacional e as características pessoais dos colaboradores e essas equipes são gerenciadas mediante uma inteligência coletiva.

Palavras-chave: gestão do conhecimento, criação de conhecimento, aprendizagem organizacional, engenharia de software, desenvolvimento de software, metodologias ágeis, teoria fundamentada em dados.

ABSTRACT

SAKAI, Eduardo. ***Knowledge management in agile software development:*** *Techniques and factors that promote the knowledge creation. School of Electrical and Computer Engineering, University of Campinas, Campinas, 2016 (Master's thesis presented to the School of Electrical and Computer Engineering).*

The software development process can be considered as an intensive process of knowledge creation. The members of agile software development teams are workers that intensively share their tacit and explicit knowledge to build innovative software. This knowledge, initially distributed in programmer's minds, should be managed by means of techniques and tools that support the organizational learning, the creation, the construction and the evaluation of software. Therefore, this study aimed to identify and understand the techniques and the enabling factors that support the creation of knowledge in agile software development.

To achieve the purposes, first, a literature review was made, which followed the principles of systematic reviews, not only to understand the state of the art but also to identify techniques and factors of the agile methodologies that promote the creation of knowledge. Second, two case studies were conducted as a methodology research through interviews whose data was qualitatively analyzed using the Grounded Theory.

As results, two main phenomena emerged from the data analyzed. The first phenomenon shows that agile teams establish techniques that support the organizational learning, the creation and the construction of software. The second phenomenon indicates that teams promote enabling factors that align organizational culture and the employees' personal characteristics and the team is managed by a collective intelligence.

Keywords*: knowledge management, knowledge creation, organizational learning, software engineering, software development, agile methodologies, grounded theory.*

RESUMEN

SAKAI, Eduardo. ***Gestión del conocimiento en el desarrollo ágil de software:*** *Las técnicas y los factores que promueven la creación de conocimiento. Escuela de Ingeniería Eléctrica y Computación de la Universidad Estadual de Campinas, Campinas, 2016 (tesis de maestría presentada a la Facultad de Ingeniería Eléctrica y Computación).*

El proceso de desarrollo de software puede considerarse un intenso proceso de creación del conocimiento. Los miembros de equipos ágiles de desarrollo de software trabajan intensamente compartiendo sus conocimientos tácitos y explícitos para construir software innovadores. Este conocimiento, distribuido inicialmente en la mente de los programadores, debe ser administrado por medio de técnicas y herramientas para apoyar el aprendizaje organizacional, la creación, la construcción y la evaluación de software. Por lo tanto, este estudio tuvo como objetivo identificar y comprender las técnicas y los factores facilitadores que ayuden a la creación del conocimiento en equipos ágiles de desarrollo de software.

Para lograr los propósitos de esta investigación, en primer lugar, se hizo una revisión de literatura, que siguió los principios de revisiones sistemáticas, para comprender el estado del arte e identificar las técnicas de las metodologías ágiles y los factores que promueven la creación del conocimiento en el desarrollo ágil de software. En segundo lugar, se llevaron a cabo dos estudios de caso como metodología de investigación a través de entrevistas cuyos datos se analizaron cualitativamente utilizando el Muestreo Teórico.

Como resultados, dos fenómenos principales surgieron de los datos fueron analizados. El Fenómeno 1 muestra que los equipos ágiles establecen técnicas que apoyan el aprendizaje organizacional, la creación y la construcción de software. El Fenómeno 2 indica que los equipos ágiles promueven factores que alinean la cultura organizacional y las características personales de los empleados y que estos equipos son gestionados por una inteligencia colectiva.

Palabras clave: *gestión del conocimiento, creación del conocimiento, aprendizaje organizacional, ingeniería de software, desarrollo de software, metodologías ágiles, muestreo teórico.*

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

AM	*Agile Modeling*
ACM	*Association for Computing Machinery*
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior
CEP	Comitê de Ética em Pesquisa
CMMI	*Capability Maturity Model Integration*
CRC	*Class – Responsibility – Collaborator*
DAS	Desenvolvimento Adaptativo de Software
DCA	Departamento de Engenharia de Computação e Automação Industrial
DNA	Ácido Desoxirribonucleico
DSDM	*Dynamic System Development Method*
FDD	*Feature Driven Development*
FEEC	Faculdade de Engenharia Elétrica e de Computação
IEEE	*Institute of Electrical and Electronics Engineers*
KIS	*Keep It Simple*
MER	Modelo Entidade Relacionamento
MVC	*Model-View-Controller*
POC	*Proof Of Concept*
PRISMA	*Preferred Reporting Items for Systematic Reviews and Meta-Analysis*
PU	Processo Unificado
RAD	*Rapid Application Development*
SEI	*Software Engineering Institute*
SWEBOK	*The Guide to the Software Engineering Body of Knowledge*
TCLE	Termo de Consentimento Livre e Esclarecido
TDD	*Test Driven Development*
TFD	Teoria Fundamentada em Dados
TIC	Tecnologia da Informação e Comunicação
UNICAMP	Universidade Estadual de Campinas
XP	*eXtreme Programming*
WGO	*What's Going On*

SUMÁRIO

1 INTRODUÇÃO

Com o advento da revolução industrial, fez-se necessária a criação de novas ideias e pensamentos econômicos. Era necessário romper com as ideias dos antigos filósofos e criar uma nova linha de pensamento. O escocês Adam Smith, considerado um dos fundadores da ciência econômica, foi uns dos pioneiros numa tentativa de responder às perguntas que surgiam com as novidades da revolução. Publicou seu famoso livro "Uma Investigação sobre a Natureza e as Causas da Riqueza das Nações", em 1976, mais conhecido como "A Riqueza das Nações".

Patrick Jake O'Rourke, em seu trabalho sobre "A Riqueza das Nações de Adam Smith" explica que Smith discute três princípios básicos: a busca do interesse próprio, a divisão do trabalho e a liberdade de comércio (O'ROURKE, 2007, p. 10).

Sem dúvidas, houve uma grande inflexão no rumo da história depois do trabalho de Adam Smith, não só no que se refere à economia, mas também na administração e no modo de produção. A divisão do trabalho é tão antiga quanto o próprio trabalho, mas Smith, possivelmente, foi o primeiro a compreender as múltiplas implicações da divisão do trabalho (O'Rourke, 2007, p. 11).

Produção em Massa

Frederick Winslow Taylor, em 1903, publicou seu livro "*Shop Management*", no qual tratava sobre as técnicas de racionalização do trabalho do operário, por meio do Estudo de Tempos e Movimentos (CHIAVENATO, 2003, p. 54). Em 1911, Taylor publicou "*The Principles of Scientific Management*". Nesse livro, Taylor idealizou a administração como ciência que visava reduzir a vadiagem sistemática dos operários, o desconhecimento da gerência das rotinas de trabalho, o tempo para sua execução e a falta de uniformidade das técnicas e dos métodos de trabalho (CHIAVENATO, 2003, p. 54-57).

Taylor procurou encarar sistematicamente o estudo da organização. Ele foi o primeiro a fazer uma análise completa do trabalho, incluindo tempos e movimentos, padrões de execução, treinar operários, especializar o pessoal, instalar salas de planejamento. Em resumo, foi o

pioneiro em assumir uma atitude metódica ao analisar e organizar a unidade fundamental de trabalho, seguindo esse critério até o topo da organização (CHIAVENATO, 2003, p. 56).

Chiavenato (2003) explica que Taylor fundamentou a Organização Racional do Trabalho nos seguintes aspectos:

> 1. Análise do trabalho e do estudo dos tempos e movimentos.
> 2. Estudo da fadiga humana.
> 3. Divisão do trabalho e especialização do operário.
> 4. Desenho de cargos e de tarefas.
> 5. Incentivos salariais e prêmios de produção.
> 6. Conceito de *homo economicus*.
> 7. Condições ambientais de trabalho, como iluminação, conforto etc.
> 8. Padronização de métodos e de máquinas.
> 9. Supervisão funcional.
>
> (CHIAVENATO, 2003, p. 56)

O principal objetivo da administração proposta por Taylor era de assegurar o máximo de prosperidade ao patrão e, ao mesmo tempo, ao empregado. Dessa maneira, deve haver uma convergência de interesses entre empregados e empregadores (CHIAVENATO, 2003, p. 56).

Taylor teve diversos seguidores, entre eles podemos destacar Henry Ford. Segundo Chiavenato (2003, p. 65), Ford tinha como ideia “popularizar um produto antes artesanal e destinado a milionários, ou seja, vender carros a preços populares, com assistência técnica garantida, revolucionando a estratégia comercial da época”. Dessa forma e seguindo as ideias de Taylor, Ford desenvolveu a produção em massa.

Racionalizar a produção possibilitou à linha de montagem produzir produtos em série de grandes lotes. Na produção em série ou em massa, o produto, o maquinário, o material, a mão-de-obra e o projeto do produto são padronizados. Isso se deu devido à condição precedente da capacidade de consumo em massa, seja real ou potencial (CHIAVENATO, 2003, p. 65).

De maneira sucinta, podemos dizer que o sistema de produção em massa procurava a redução de custos unitários dos produtos, a especialização e divisão do trabalho, caracterizava-se pelo estoque excessivo e não havia uma grande preocupação com a qualidade.

Chiavenato (2003) aponta algumas das principais críticas ao modelo de Administração Científica, entre elas: o mecanicismo, a superespecialização do operário, a visão microscópica do homem, a ausência de comprovação científica, a abordagem incompleta da organização, a limitação do campo de aplicação, a abordagem prescritiva e normativa e, a abordagem de sistema fechado, sem a preocupação da criação de valor ao cliente.

Duck (1998) ressalta que o legado de Taylor e da administração científica impede a percepção que a gestão da mudança envolve principalmente administrar a dinâmica, e não as partes. Assim o dirigente continua quebrando a mudança em pequenas partes, e daí administrando somente as partes. A maioria dos problemas administrativos está em entender a complexidade de dinâmica, e não a complexidade de detalhes. Quando a mesma ação produz efeitos completamente diferentes a curto e longo prazo é garantia de que existe complexidade dinâmica (SENGE, 1990).

O desafio do projeto organizacional reside justamente em adquirir visão de totalidade e compreender essa complexidade dinâmica da organização. Na década de 1970, o aumento de competitividade e o excesso de oferta já não deixavam alternativas, obrigando as organizações adquirirem a capacidade de se adequarem ao ambiente, ou seja, se tornarem flexíveis.

A globalização da economia e a rapidez das alterações no contexto social e político afetam igualmente a sobrevivência imediata e a viabilidade futura das empresas. A intensidade da competição, a vulnerabilidade de mercados, a versatilidade da clientela e a variação tecnológica fazem da mudança a essência da gerência (MOTTA, 2001). Nesse sentido, inicia-se uma tendência crescente de substituir o modelo estático tradicional por um que enxerga a organização como um sistema dinâmico.

Produção Enxuta

O Sistema Toyota de Produção teve como idealizadores Sakichi Toyoda, Kichiro Toyoda, Eiji Toyoda, Taiichi Ohno e Shigeo Shingo. Em “O Sistema Toyota de Produção: Além da Produção em Larga Escala”, Taiichi Ohno explica que a base do Sistema Toyota de Produção é a eliminação total do desperdício. Para tanto é necessária que a sincronização da produção seja praticada com rigidez o que faz com que a flutuação seja nivelada ou suavizada. O tamanho dos lotes é diminuído e é evitado o fluxo contínuo de um item em grande quantidade (OHNO, 1997, p. 107).

Segundo Ohno (1997, p. 39), o passo preliminar para a aplicação do Sistema Toyota de Produção é identificar e eliminar completamente os sete desperdícios:

1. Superprodução;
2. Transporte;
3. Tempo disponível (tempo de espera);
4. Processamento em si;
5. Estoque disponível;
6. Movimento; e
7. Defeitos, produzir produtos defeituosos;

Para Ohno (1997, p. 25-26) dois pilares sustentam o Sistema Toyota de Produção: o *just-in-time* e a *autonomação* (*jidoka*). Esses pilares visam sempre a absoluta eliminação do desperdício.

Conforme explica Ohno (1997), *just-in-time* refere-se ao fato de as partes corretas e necessárias à montagem, em um processo de fluxo, alcancem a linha de montagem apenas no momento em que são necessárias e somente em quantidade necessária. Uma empresa que consiga estabelecer esse fluxo integralmente pode chegar ao estoque zero (OHNO, 1997, p. 26).

Segundo Ohno (1997) o conceito de autonomação se difere da automação tradicional pela autocapacidade de distinguir entre condições normais e anormais de funcionamento da máquina pela inserção de um dispositivo para tal finalidade. Este dispositivo, chamado *Poka-Yoke*, inserido na máquina vai além de simplesmente parar o funcionamento da máquina em condições anormais. “Parar a máquina quando ocorre um problema força todos a tomar conhecimento do fato. Quando o problema é claramente compreendido, a melhoria é possível.” (OHNO, 1997, p. 28).

Taiichi Ohno (1997) explica a principal diferença entre o Sistema Ford e o Sistema Toyota de produção:

> O Sistema Ford advoga os grandes lotes, lida com vastas quantidades, e produz muito inventário. Por contraste, o Sistema Toyota trabalha com a premissa de eliminar totalmente a superprodução gerada pelo inventário e custos relacionados a operários, propriedade e instalações necessárias à gestão do inventário. Para atingir isto, praticamos o sistema *kanban*, segundo o qual um processo posterior vai

> até um processo anterior para retirar peças necessárias apenas-a-tempo (*just-in-time*).
>
> [...]
>
> Em resumo, onde o Sistema Ford tem fixa a ideia de produzir em uma só vez uma boa quantidade do mesmo item, o Sistema Toyota sincroniza a produção de cada unidade. A ideia por trás desta abordagem é a de que no mercado cada consumidor adquire um carro diferente, e assim na fabricação os carros devem ser feitos um por vez. Mesmo no estágio de produção de peças, cada peça é produzida uma de cada vez.
>
> (OHNO, 1997, p. 107-108)

Outro ponto em que a produção enxuta difere da produção em massa é que a comunicação entre o fornecedor e o cliente não é uma via de mão única. Só agrega valor ao produto aquilo que agrega valor ao cliente. Ohno (1997) explica essa visão:

> Os valores sociais mudaram. Agora, não podemos vender nossos produtos a não ser que nos coloquemos dentro dos corações de nossos consumidores, cada um dos quais tem conceitos e gostos diferentes. Hoje, o mundo industrial foi forçado a dominar de verdade o sistema de produção múltiplo, em pequenas quantidades (OHNO, 1997, p. X).

Sociedade do conhecimento

Nonaka e Takeuchi (1997, p. 5) explicam que vários autores proeminentes como Peter Drucker, Alvin Toffler, James Brian Quinn e Robert Reich, cada qual a seu modo, todos anunciam a chegada de uma nova economia ou sociedade baseada no conhecimento. Peter Drucker a chama de “sociedade do conhecimento”, segundo Nonaka e Takeuchi (1997):

> Drucker (1993) argumenta [...] que, na nova economia, o conhecimento não é apenas mais um recurso, ao lado dos tradicionais fatores de produção – trabalho, capital e terra – mas sim o único recurso significativo atualmente. Ele afirma que o fato de o conhecimento ter se tornado *o* recurso, muito mais do que apenas *um* recurso, é o que torna singular a nova sociedade (NONAKA; TAKEUCHI, 1997, p. 5, grifo do autor).

Chiavenato (2003) também aponta para uma administração que visa a economia do conhecimento e a sociedade do conhecimento. Para o autor, o progresso científico tem influenciado

profundamente a teoria administrativa com a teoria da complexidade e a teoria do caos. As organizações devem focar na gestão do conhecimento para gerirem seu mais importante patrimônio: os ativos intangíveis. O que faz necessária uma abordagem voltada para a aprendizagem organizacional (CHIAVENATO, 2003, p. 625).

Gestão do conhecimento no desenvolvimento de software

A produção de software certamente foi influenciada pela indústria tradicional de bens tangíveis. Os primeiros métodos de desenvolvimento de software, conhecidos como metodologias tradicionais, pesadas, prescritivas ou tayloristas compartilham das principais ideias de Taylor e aquilo que foi aplicado por Ford, guardadas as respectivas diferenças. As prescrições completas das metodologias exaustivamente produzidas seguindo os passos do planejamento e documentações, a especialização dos projetistas de requisitos, dos desenvolvedores, dos testadores e todos os envolvidos no desenvolvimento, carregaram o desenvolvimento de software sob estigmas extremamente rígidos. A mão de via única com o cliente, de dentro para fora da organização, onde o cliente deve se moldar ao software e não o contrário.

Já as metodologias ágeis, apreenderam e incorporaram as melhores técnicas da produção enxuta. A resposta à mudança, equipes multifuncionais e autônomas, a ênfase no conhecimento tácito para evitar desperdícios com documentação, a inspeção da qualidade em cada entrega de incremento, a comunicação direta com o cliente para atender suas expectativas, certamente sofreram influências da indústria japonesa de bens tangíveis.

Na era da informação, o software é a tecnologia única mais importante no palco mundial pois ao mesmo tempo ele é o produto e o veículo de entrega do produto e, além disso, o software entrega o mais importante produto de nossa época: a informação (Pressman, 2006).

Durante a última década os estudos da gestão do conhecimento no desenvolvimento de software vem sendo aprofundado. Autores como Armour (2000) acreditam que o software é a mais nova forma de se armazenar conhecimento. De maneira geral os estudiosos do assunto tratam o desenvolvimento de software como uma atividade intensiva do conhecimento.

Embora as metodologias de desenvolvimento de software tenham incorporado alguns conceitos da gestão do conhecimento, como a ênfase no conhecimento tácito, suas maiores

preocupações residem em oferecer agilidade para enfrentar as mudanças exigidas pelo ambiente. Porém, visões como aprendizagem organizacional e estímulo à criação de conhecimento, seja no âmbito técnico ou comercial, não são muito trabalhadas. Após a revisão de literatura verificou-se que poucos estudiosos têm se dedicado a compreender a dinâmica inter-relacional que existe entre as técnicas e fatores utilizados no desenvolvimento de software com a gestão do conhecimento voltada à aprendizagem organizacional para inovação. Nesse sentido, a gestão do conhecimento tem muito a contribuir com o desenvolvimento ágil de software inserindo visões de aprendizado organizacional e inovação e trazendo para as metodologias ágeis técnicas, ferramentas e fatores fundamentados na teoria de gestão do conhecimento, o que motivou este estudo.

Objetivos

Dessa forma, os objetivos desta dissertação são identificar e compreender as técnicas e os fatores utilizados por equipes ágeis que privilegiam a criação de conhecimento e compreender como se dá essa dinâmica na prática em equipes ágeis de desenvolvimento de software criadoras de conhecimento.

A interdisciplinaridade inerente a este trabalho levou a compreender os principais conceitos de gestão do conhecimento e abordagens metodológicas de desenvolvimento de software, os quais são apresentados no Capítulo 2 – Fundamentação Teórica. Em seguida foi feita uma revisão de literatura, seguindo os princípios de revisões sistemáticas, utilizando-se do *framework* PRISMA para compreender o Estado da Arte da criação de conhecimento no desenvolvimento de software e foi possível classificar as principais técnicas das metodologias ágeis e fatores capacitadores que privilegiam a gestão do conhecimento, relatada no Capítulo 3 – Revisão de Literatura.

Após a revisão de literatura foi elaborado o desenho de pesquisa e optou-se por uma abordagem qualitativa por meio de estudos de caso. Foi preparado um roteiro semiestruturado para entrevistas, que foi pré-validado com um grupo de alunos de pós-graduação da UNICAMP e adquiridas todas as aprovações junto ao Comitê de Ética em Pesquisa. Em seguida, foram feitas entrevistas em duas empresas de software de Campinas, sendo com três colaboradores da Empresa A e dois colaboradores da Empresa B. Para análise qualitativa dos dados coletados, este trabalho se baseou na Teoria Fundamentada de Dados (*Grounded Theory*) que procurou classifi-

car os principais conceitos (códigos) em grupos de conceitos-chaves e analisar suas inter-relações. Todo esse percurso metodológico é explicado no Capítulo 4 – Metologia de Pesquisa.

O Capítulo 5 – Resultados e Discussões mostra que foram encontrados dois fenômenos centrais que compuseram a teoria substantiva que emergiu dos dados coletados nos dois estudos de caso. O Fenômeno 1 indica que “a equipe ágil estabelece ***técnicas*** que apoiam a **aprendizagem organizacional**, que ajudam na **criação** e na **construção** de software inovadores e que permitem uma eficiente **comunicação** entre todas as partes envolvidas fazendo uso de **ferramentas** adequadas”. Já o Fenômeno 2 sugere que “a equipe ágil seja gerenciada mediante uma eficiente **comunicação** que privilegie a **inteligência coletiva** e promova **fatores capacitadores** que alinham a **cultura organizacional** e as **características pessoais** dos colaboradores apoiados por **ferramentas** adequadas”.

Por fim, as conclusões e os trabalhos futuros são apresentadas no Capítulo 6 – Conclusões, seguido do referencial bibliográfico e anexos.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção são apresentados os principais conceitos envolvidos nesta pesquisa. De caráter interdisciplinar esta pesquisa entrelaça os temas de gestão do conhecimento com o desenvolvimento de software.

2.1 Gestão do Conhecimento

A teoria de gestão do conhecimento envolve as áreas de epistemologia, psicologia, sociologia e administração. São apresentados os principais conceitos de conhecimento e a teoria de gestão do conhecimento com foco na criação de conhecimento organizacional.

2.1.1 Conceitos de Conhecimento

Os estudos sobre conhecimento são tão antigos quanto a própria história da filosofia ocidental. O ser humano utiliza a razão e capacidade de criar conhecimentos para seu benefício próprio. Desse estudo se derivou o ramo da filosofia conhecido como epistemologia. Segundo os professores japoneses Nonaka e Takeuchi (1997, p. 24) “os filósofos ocidentais concordam que conhecimento é a ‘crença verdadeira e justificada’, um conceito introduzido inicialmente por Platão em Ménon, Pédon e Teeteto”. Na visão de Platão, para alguém possuir conhecimento de algo, esse algo deve ser verdadeiro, esse alguém deve acreditar ou ter convicção de que esse algo é verdadeiro e ainda essa crença deve ter uma devida explicação, um motivo ou alguma razão. Ainda assim, por mais que se possa ter certeza sobre algo, existe ainda uma pequena possibilidade de se estar errado. Em termos formais, segundo as Notas do livro a Criação de Conhecimento na Empresa de Nonaka e Takeuchi (1997):

> Para que o indivíduo A tenha conhecimento de algo (ou seja, uma proposição P de agora em diante), é preciso que o que se segue sejam condições necessárias e suficientes do conhecimento de A sobre P:
> (a) P é verdadeiro (condição de verdade);
> (b) A precisa acreditar que P é verdadeiro (condição de crença); e
> (c) A crença de A de que P é verdadeiro precisa ser justificada (condição de justificação)
>
> (NONAKA; TAKEUCHI, 1997, p. 289)

Ainda segundo Nonaka e Takeuchi (1997, p. 25) foi Platão quem desenvolveu uma elaborada estrutura de pensamento sobre o conhecimento de uma perspectiva racionalista, no qual acreditava na “forma” proveniente do mundo das ideias, eterna e imutável, pela abstração do olho mental puro.

Aristóteles viu o conhecimento de maneira mais empírica contrariando seu mestre. Para o discípulo de Platão, existem seis formas ou graus de conhecimento: sensação, percepção, imaginação, memória, raciocínio e intuição. Toda a teoria aristotélica do conhecimento segue uma explicação de como o sujeito pode partir de dados sensíveis que lhe mostram sempre o concreto para chegar finalmente a formulações científicas, na medida que são necessárias e universais (PESSANHA, 2004, p. 21).

A origem do conhecimento

A partir dessas duas visões mencionadas previamente, grandes filósofos basearam suas teorias quanto à origem ou fundamentação do conhecimento. Duas correntes são comumente tratadas: o racionalismo e o empirismo.

O filósofo francês René Descartes procurou entender a realidade de maneira puramente racional. Segundo Nonaka e Takeuchi (1997, p. 26-27), o filósofo francês acreditava que as ideias eram inatas à mente humana. Descartes formulou sua famosa teoria na qual separa o “eu pensante” do corpo e da matéria. Por meio do “método da dúvida”, em busca de responder a questão: “O que posso manter como verdade acima de qualquer dúvida?” Ele descobriu que era possível questionar todas as crenças, exceto a existência do questionador, o que foi expresso pela famosa frase “penso, logo existo”, sua filosofia ficou conhecida como a divisão cartesiana.

Já o filósofo britânico John Locke, em seu livro “Ensaio Acerca do Entendimento Humano” em resposta a Descartes defendeu a visão empirista. Para o filósofo britânico as ideias não são inatas como acreditava Descartes. Locke (1999) afirma que não há proposições aceitas universalmente uma vez que não há proposições inatas. O filósofo britânico afirma, por exemplo, que ninguém apreende o gosto do abacaxi até que o experimente (LOCKE, 1999, p. 276). Segundo Nonaka e Takeuchi (1997, p. 27-28) para Locke, a mente humana pode ser comparada a uma “tábula rasa” ou uma “folha de papel em branco”. Ela seria preenchida de maneira com que nossos sentidos fossem tendo contato com as coisas do mundo, por meio de experiência. Na medida em que vamos tendo contato por meio dos sentidos, nossa mente vai se

equipando de conhecimento. Locke (1999) acredita que todo o nosso conhecimento está baseado na experiência e dela se deriva, empregada tanto nos objetos sensíveis externos como nas operações internas de nossas mentes, percebidas e refletidas por nós mesmo. Dessas duas fontes de conhecimento jorram todas as nossas ideias, ou as que possivelmente teremos (LOCKE, 1999, p. 57).

Baseado na dialética, o filósofo alemão Georg Wilhelm Friedrich Hegel propôs o seu método dialético para explicar o conhecimento. A dialética hegeliana apresenta ideias semelhantes às de Heráclito, filósofo pré-socrático, que partia do princípio da identidade de opostos e a constância da mudança, o *devir*. Para o filósofo alemão, "o conhecimento começa com a percepção sensorial, que se torna mais subjetiva e mais racional pela purificação dialética dos sentidos e, por fim, chega ao estágio do autoconhecimento do 'espírito Absoluto'" (RUSSEL, 1961, p. 704 apud NONAKA; TAKEUCHI, 1997 p. 28). Para Hegel o *Espírito Absoluto* é a totalidade única e trata-se do "conhecimento mútuo". Em suma, seu método dialético é caracterizado em *tese*, *antítese* e *síntese*. Para termos a faculdade do conhecimento, primeiro, nossa mente separa tudo aquilo que pode ser (tese) e exclui tudo aquilo que não pode ser (antítese). O resultado disso é apresentado em forma de síntese. A síntese torna-se uma nova tese, essa por sua vez terá em contraposição uma outra antítese para formar uma nova síntese e assim por diante, num processo infinito. O método dialético de Hegel refere-se à epistemologia mas pode ser ampliado e aplicado em vários domínios. Karl Marx utiliza-se do método dialético para explicar as relações sociais.

Tipos de conhecimento

Mais recentemente, durante a década de 1960, o húngaro-britânico Michael Polanyi, ainda no domínio da filosofia, introduziu a distinção entre conhecimento tácito e conhecimento explícito. Polanyi acreditava que o conhecimento tem uma componente pessoal indispensável.

> Polanyi observa que os seres humanos adquirem conhecimentos criando e organizando ativamente suas próprias experiências. Assim, o conhecimento que pode ser expresso em palavras e números representa apenas a ponta do *iceberg* do conjunto de conhecimentos como um todo. Como diz Polanyi (1966, p. 4), "Podemos saber mais do que podemos dizer".
>
> (NONAKA; TAKEUCHI, 1997, p. 65, grifo do autor)

De maneira geral podemos dizer que o conhecimento humano se apresenta em dois tipos: explícito e tácito. O conhecimento explícito refere-se a todo tipo de conhecimento que pode ser formulado e transmitido por livros, manuais ou mesmo por linguagem formal. Já o conhecimento tácito é todo o conhecimento inarticulado, difícil de expressar e tem um caráter pessoal extremamente relevante (NONAKA; TAKEUCHI, 1997). De certa forma, para compreender um certo conhecimento explícito é necessário estar equipado de algum conhecimento tácito sobre o assunto. Assim, retomando a definição de conhecimento de Platão, em que o conhecimento é "crença verdadeira e justificada", o conhecimento tácito está intimamente ligado ao caráter pessoal da crença justificada.

A distinção entre o conhecimento tácito e o conhecimento explícito é o ponto central da teoria de Nonaka e Takeuchi (1997) e compõem a dimensão epistemológica do modelo de criação de conhecimento. Segundo os autores, o conhecimento tácito inclui elementos cognitivos e técnicos. Os elementos cognitivos centram-se nos "modelos mentais", em que os seres humanos criam modelos do mundo estabelecendo e manipulando analogias em suas mentes; tais modelos como esquemas, paradigmas, perspectivas, crenças e pontos de vista, ajudam os indivíduos a perceberem e definirem seu próprio mundo. Já os elementos técnicos incluem o *know-how* concreto, técnicas e habilidades (NONAKA; TAKEUCHI, 1997, p. 66).

2.1.2 Gestão do Conhecimento na Empresa

Depois de analisar de forma crítica as principais teorias econômicas, administrativas e organizacionais, Nonaka e Takeuchi (1997) sentiram a necessidade de criar uma nova teoria para a criação de conhecimento na empresa. Os autores explicam: "Por criação de conhecimento organizacional queremos dizer a capacidade que uma empresa tem de criar conhecimento, disseminá-lo na organização e incorporá-lo a produtos, serviços e sistemas." (NONAKA; TAKEUCHI, 1997, p. XII).

Conhecimento vs. Informação

Embora seja comum alguns autores não fazerem distinção entre conhecimento e informação, para a teoria da criação de conhecimento ela é necessária. Nonaka e Takeuchi (1997) ressaltam três observações:

1. O conhecimento, ao contrário da informação, diz respeito às crenças e aos compromissos. Ele é uma função de uma atitude, perspectiva ou intenção específica.
2. O conhecimento, ao contrário da informação, relaciona-se com a ação. É sempre o conhecimento “com algum fim”.
3. O conhecimento, assim como a informação, refere-se ao significado. É específico ao contexto e relacional.

Ainda segundo Nonaka e Takeuchi (1997, p. 63-64), a informação pode ser vista por duas perspectivas: “sintática” e “semântica”. A informação sintática refere-se ao fluxo de informação medido sem levar em consideração o significado. Já a informação semântica concentra-se em seu significado e é mais importante para a teoria da criação do conhecimento. Portanto “a informação é um fluxo de mensagens, enquanto o conhecimento é criado por esse próprio fluxo de informação, ancorado nas crenças e compromissos do seu detentor.” (NONAKA; TAKEUCHI, 1997, p. 64).

As duas dimensões da criação do conhecimento

A teoria da criação de conhecimento na empresa de Nonaka e Takeuchi (1997), baseia-se em duas dimensões: epistemológica e ontológica.

A dimensão epistemológica, como dito anteriormente, está fundamentada na distinção entre o conhecimento tácito e o conhecimento explícito; retomando, segundo Nonaka e Takeuchi (1997) o conhecimento explícito é facilmente comunicado e compartilhado sob a forma de palavras e números, dados brutos, fórmulas científicas, procedimentos ou princípios universais. Esse tipo de conhecimento é visto como sinônimo de um código de computador ou um conjunto de regras gerais. O conhecimento tácito é altamente pessoal e difícil de formalizar, o que dificulta sua transmissão e compartilhamento com os outros. Ele está profundamente enraizado nas ações, experiências, emoções, valores e ideais de um indivíduo. Nessa categoria de conhecimento estão as conclusões, *insights* e palpites subjetivos. Para os autores japoneses, os dois tipos de conhecimento, tácito e explícito, não são entidades totalmente separadas, e sim mutuamente complementares. Eles interagem um com o outro e realizam trocas nas atividades criativas dos seres humanos (NONAKA; TAKEUCHI, 1997, p. 67).

Já a dimensão ontológica refere-se justamente a interação dos indivíduos nos diferentes níveis: individual, grupal e organizacional. Como afirmam Nonaka e Takeuchi (1997) uma orga-

nização não pode criar conhecimento sem indivíduos. Ela deve apoiar os indivíduos criativos e lhes proporcionar contextos favoráveis a criação de conhecimento. A criação de conhecimento organizacional deve ampliar o conhecimento criado pelos indivíduos de forma a cristalizá-lo como parte da rede de conhecimento da organização (NONAKA; TAKEUCHI, 1997, p. 65).

2.1.3 A Espiral do Conhecimento

O modelo dinâmico de criação de conhecimento proposto por Nonaka e Takeuchi (1997) "está ancorado no pressuposto crítico de que o conhecimento humano é criado e expandido pelas interações sociais entre o conhecimento tácito e o conhecimento explícito. Chamamos essa interação de "conversão do conhecimento" (NONAKA; TAKEUCHI, 1997, p. 67).

A partir deste pressuposto, Nonaka e Takeuchi (1997) postularam quatro modos de conversão de conhecimento:

(1) **Socialização**: conversão de conhecimento tácito em conhecimento tácito.
(2) **Externalização**: conversão de conhecimento tácito em conhecimento explícito.
(3) **Combinação**: conversão de conhecimento explícito em conhecimento explícito.
(4) **Internalização**: conversão de conhecimento explícito em conhecimento tácito.

A criação de conhecimento na empresa se dá por meio da espiral do conhecimento. A espiral do conhecimento é justamente os quatro tipos de conversão do conhecimento, além da transferência entre os indivíduos do grupo e da organização, chamada de "conversão social". A conversão social viabiliza o processo de expansão do conhecimento tácito e do conhecimento explícito, tanto em termos de qualidade quanto de quantidade por toda a organização (NONAKA; TAKEUCHI, 1997).

Socialização

A socialização do conhecimento se dá por meio de um processo de compartilhamento de experiências e, a partir daí, da criação do conhecimento tácito, como modelos mentais ou habilidades técnicas compartilhadas. Devido ao caráter inarticulado do conhecimento tácito e sua dificuldade de ser transmitido, o segredo para sua transferência é a experiência e é possível de se transmitir esse tipo de conhecimento por meio da observação, imitação e prática. Sem alguma

forma de experiência compartilhada, é extremamente difícil para uma pessoa projetar-se no processo de raciocínio do outro indivíduo (NONAKA; TAKEUCHI, 1997, p. 69).

Os clientes e os funcionários de linha de frente também são peças importantes no processo de socialização já que eles possuem mais conhecimento de domínio. As interações sociais com os clientes antes do desenvolvimento do produto e após seu lançamento formam um processo infinito de compartilhamento do conhecimento tácito e criação de ideias para melhoria e aperfeiçoamento (NONAKA; TAKEUCHI, 1997, p. 71).

Externalização

O processo de conversão do conhecimento tácito em conhecimento explícito é o ponto-chave da teoria dos professores japoneses. É nesse tipo de conversão de conhecimento que se cria conceitos novos e explícitos a partir do conhecimento tácito (NONAKA; TAKEUCHI, 1997, p. 73). Trata-se de um processo de criação do conhecimento perfeito pois o conhecimento tácito se torna explícito, expresso na forma de metáforas, analogias, conceitos, hipóteses ou modelos (NONAKA; TAKEUCHI, 1997, p. 71). Dessa forma, externalizar o conhecimento pode ser entendido como a criação de conceitos, estimulados pelo diálogo ou pela reflexão coletiva.

A linguagem é uma das formas de se converter conhecimento tácito em conhecimento explícito (EMIG, 1983 apud NONAKA; TAKEUCHI, 1997). Por isso se faz necessário o estímulo ao diálogo. Os métodos analíticos de dedução e indução servem para criar conceitos. Os conceitos podem ser deduzidos por meio de *slogans* bem formulados como norteadores das equipes de desenvolvimento e induzidos a partir de “viagens conceituais” que se constituem em experiências de direção realizadas pelos membros da equipe de desenvolvimento (NONAKA; TAKEUCHI, 1997, p. 71).

As metáforas e analogias ganham destaque na externalização pois são ferramentas que fazem o elo entre conhecimento tácito e o conhecimento explícito. O uso de uma metáfora ou analogia é muito eficaz pois estimula diretamente o processo criativo (NONAKA; TAKEUCHI, 1997, p. 72).

Por meio do uso de metáforas as pessoas reúnem seus conhecimentos de novas formas e começam a expressar aquilo que ainda não são capazes de dizer (NONAKA; TAKEUCHI,

1997, p. 13). Portanto, a metáfora é uma maneira de entender intuitivamente uma coisa imaginando simbolicamente outra coisa (NONAKA; TAKEUCHI, 1997, p. 74). Dessa forma é possível relacionar conceitos completamente distintos e distantes em nossa mente, relacionando até mesmo conceitos abstratos a fim de criar conceitos concretos (NONAKA; TAKEUCHI, 1997, p. 75).

Já a analogia apresenta uma melhor estrutura com relação à distinção entre duas ideias ou objetos. Ela é realizada pelo pensamento racional e se concentra em destacar o caráter "comum", nas semelhanças estruturais/funcionais entre duas coisas distintas. Portanto, a analogia permite entender o desconhecido por meio do conhecido, eliminando a lacuna entre a imagem e o modelo lógico (NONAKA; TAKEUCHI, 1997, p. 75).

Combinação

A combinação é a forma clássica de se transmitir conhecimento realizada por meio da educação e do treinamento formal. Trata-se de um processo de sistematização de conceitos. Nesse modo de conversão de conhecimento está envolvido a combinação de conjuntos diferentes de conhecimento explícito como documentos, reuniões, conversas ao telefone ou ferramentas de comunicação baseadas em tecnologia da informação. Pode-se reconfigurar as informações por meio da classificação, do acréscimo, da combinação e da categorização do conhecimento explícito (como o realizado em banco de dados de computadores) podendo levar a novos conhecimentos (NONAKA; TAKEUCHI, 1997, p. 75-76).

Para que ocorra esse tipo de conversão de conhecimento, Nonaka e Takeuchi (1997) ressaltam a importância dos gerentes de nível médio. Os gerentes de nível médio desmembram e operacionalizam visões empresariais, conceitos de negócios ou conceitos de produtos. Esses gerentes desempenham um papel crítico na criação de novos conceitos através da rede de informações e conhecimentos codificados.

Internalização

A conversão do conhecimento explícito novamente em conhecimento tácito é a última etapa do ciclo de conversão de conhecimento. Segundo Nonaka e Takeuchi (1997) a internalização é o processo de incorporação do conhecimento explícito em conhecimento tácito sob a forma de modelos mentais ou *know-how* técnico compartilhado e está intimamente relaciona-

do ao “aprender fazendo”. Portanto, “para viabilizar a criação de conhecimento organizacional, o conhecimento tácito acumulado precisa ser socializado com outros membros da organização, iniciando assim uma nova espiral de criação do conhecimento.” (NONAKA; TAKEUCHI, 1997, p. 77).

A internalização ocorre o tempo todo pois à medida que um indivíduo entra em contato com novos conhecimentos esse conhecimento explícito é internalizado e combinado com o prévio conhecimento tácito pessoal do indivíduo atualizando assim suas crenças, sentimentos e modelos mentais.

A verbalização e diagramação do conhecimento sob a forma de documentos ou histórias orais ajudam a internalização do conhecimento explícito em conhecimento tácito, permitindo vivenciar indiretamente as experiências dos outros. Além disso, ler ou ouvir uma história de sucesso permite aos membros da organização sentirem o realismo da história e transformá-lo em um modelo mental tácito. O compartilhamento de tal modelo mental pelo maior número possível de membros da organização, permite que esse conhecimento tácito passe a fazer parte da cultura organizacional (NONAKA; TAKEUCHI, 1997, p. 78).

Modelo

Nonaka e Takeuchi (1997, p. 79-80) explicam que esses quatro modos de conversão do conhecimento, quando trabalhados isoladamente, não são propícios à criação de conhecimento. Porém, quando trabalhados em conjunto permitem e favorecem a criação de novos conceitos não só no nível individual, mas grupal e organizacional. A Figura 1 ilustra a espiral do conhecimento ou modelo SECI.

Figura 1 – Espiral do conhecimento.
Fonte: Nonaka e Takeuchi (1997, p. 80)

Segundo a teoria de Nonaka e Takeuchi (1997, p. 82), o conhecimento tácito dos indivíduos constitui a base da criação do conhecimento organizacional. A organização deve mobilizar o conhecimento tácito criado e acumulado no nível individual. O conhecimento tácito mobilizado é ampliado “organizacionalmente” por meio dos quatro modos de conversão do conhecimento e cristalizado em níveis ontológicos superiores. Nonaka e Takeuchi (1997) chamam esse processo de “espiral do conhecimento”, na qual a interação entre conhecimento tácito e conhecimento explícito terá uma escala cada vez maior na medida em que subirem os níveis ontológicos. Assim, a criação do conhecimento organizacional é um processo em espiral, que começa no nível individual e vai subindo, ampliando comunidades de interação que cruzam fronteiras entre seções, departamentos, divisões e até organizações. A Figura 2 ilustra esse processo.

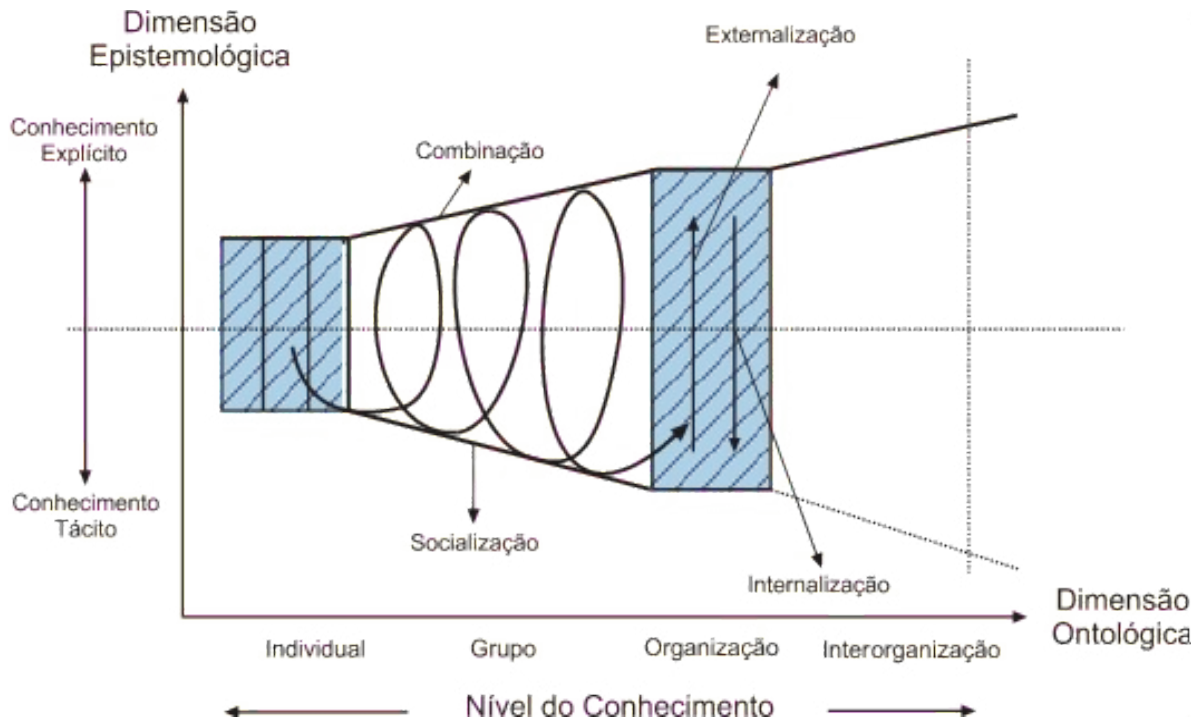

Figura 2 – Espiral de criação do conhecimento organizacional

Fonte: Nonaka e Takeuchi (1997, p. 82)

2.1.4 As Cinco Condições Capacitadoras para a Criação de Conhecimento

Nonaka e Takeuchi (1997) afirmam que existem cinco condições capacitadoras para a criação de conhecimento: intenção, autonomia, flutuação e caos criativo, redundância e variedade de requisitos.

Intenção

A intenção organizacional direciona a espiral do conhecimento pois define a aspiração de uma organização às suas metas. Ela fornece o critério mais importante para julgar a veracidade de um determinado conhecimento. Frequentemente é expressa pela visão ou padrões organizacionais que podem servir para avaliar e justificar o conhecimento criado. A intenção é necessariamente carregada de valor (NONAKA; TAKEUCHI, 1997, p. 84).

As organizações devem formular uma intenção organizacional e propor essa intenção a seus funcionários. Essa atividade é mais organizacional do que individual e por meio do compromisso coletivo a organização orienta e promove seus funcionários. A intenção é uma peça chave nesse processo (NONAKA; TAKEUCHI, 1997, p. 84).

Autonomia

A autonomia é fundamental na teoria da criação de conhecimento na empresa, pois propicia a automotivação e amplia a chance de se introduzir oportunidades inesperadas. Ela aumenta a possibilidade dos indivíduos se auto motivarem. Todos os membros de uma organização devem agir de forma autônoma conforme as circunstâncias.

Nonaka e Takeuchi explicam que a autonomia é importante para as equipes auto-organizadas. Esse tipo de equipe é uma poderosa ferramenta para criação de circunstâncias nas quais os indivíduos agem de forma autônoma. Equipes auto-organizadas devem ser interfuncionais, envolvendo membros de diferentes atividades organizacionais (NONAKA; TAKEUCHI, 1997, p. 86).

Flutuação e caos criativo

A flutuação e o caos criativo estimulam a interação entre a organização e o ambiente externo (NONAKA; TAKEUCHI, 1997, p. 88).

A flutuação é caracterizada pela “ordem sem recursividade” e quando é introduzida, seus membros enfrentam um “colapso” de rotinas, hábitos ou estruturas cognitivas. O colapso permite a reconsideração de pensamento e de perspectivas fundamentais. Quando essa flutuação gera um colapso dentro da organização é possível criar novos conhecimentos (NONAKA; TAKEUCHI, 1997, p. 89).

Já o caos criativo pode ser gerado naturalmente quando a empresa enfrenta uma crise real ou pode ser induzido intencionalmente pelos líderes da organização quando tentam simular um “sentido de crise” entre os membros da organização, propondo metas desafiadoras (NONAKA; TAKEUCHI, 1997, p. 90). Porém é necessário que os membros da organização possuam a habilidade de refletir sobre suas ações para que os benefícios do caos criativo sejam alcançados. Caso contrário, a flutuação pode levar ao caos “destrutivo” (NONAKA; TAKEUCHI, 1997, p. 90).

Redundância

A redundância de informações em uma organização favorece o compartilhamento do conhecimento tácito, apresentando grande importância no estágio de desenvolvimento de con-

ceitos, uma vez que permite aos indivíduos sentir o que os outros estão tentando expressar, externalizando seus conhecimentos. Diferentes pontos de vistas são apresentados como uma superposição intencional de informações sobre as atividades da empresa, o que permite que os indivíduos forneçam novas informações de diferentes perspectivas (NONAKA; TAKEUCHI, 1997, p. 92).

Existem diversas formas de se alcançar a redundância em uma organização criadora de conhecimento, dentre elas estão a divisão da equipe de desenvolvimento em grupos concorrentes que apresentam suas propostas e discutem as vantagens e desvantagens de cada uma. Outra forma de criar redundância se dá por meio de um “rodízio estratégico” de pessoal de diferentes funções, o que leva os indivíduos a compreenderem diversas perspectivas (NONAKA; TAKEUCHI, 1997, p. 93).

Variedade de requisitos

Por fim, a quinta condição capacitadora para promover a espiral do conhecimento é a variedade de requisitos. A variedade de requisitos diz respeito ao ambiente, ou seja, quanto maior a variedade de requisitos maior a chance de atender a variedade do ambiente.

Nesse sentido, conforme explicam Nonaka e Takeuchi (1997, p. 94), a variedade de requisitos pode ser desenvolvida e melhorada com o auxílio da combinação de informações de maneira rápida e flexível e com acesso facilitado às informações em todos os níveis da organização. Ainda segundo os autores, os sistemas de informações desempenham um papel fundamental nessa condição capacitadora pois permitem o intercâmbio de informações e opiniões entre diversas unidades da organização com diferentes pontos de vista. Uma maneira de reagir rapidamente as variações inesperadas do ambiente e apresentar uma diversidade interna é alterar frequentemente a estrutura da organização. Portanto, uma maneira de lidar com a complexidade do ambiente é desenvolver uma estrutura organizacional horizontal e flexível na qual diferentes unidades são interligadas por uma rede de informações (NONAKA; TAKEUCHI, 1997, p. 95).

2.1.5 Modelo de Cinco Fases de Criação de Conhecimento Organizacional

O modelo de criação de conhecimento proposto por Nonaka e Takeuchi (1997) constitui um processo ideal de cinco fases: (1) compartilhamento do conhecimento tácito; (2) criação

de conceitos; (3) justificação dos conceitos; (4) construção de um arquétipo; e (5) difusão interativa do conhecimento. O modelo é ilustrado na Figura 3.

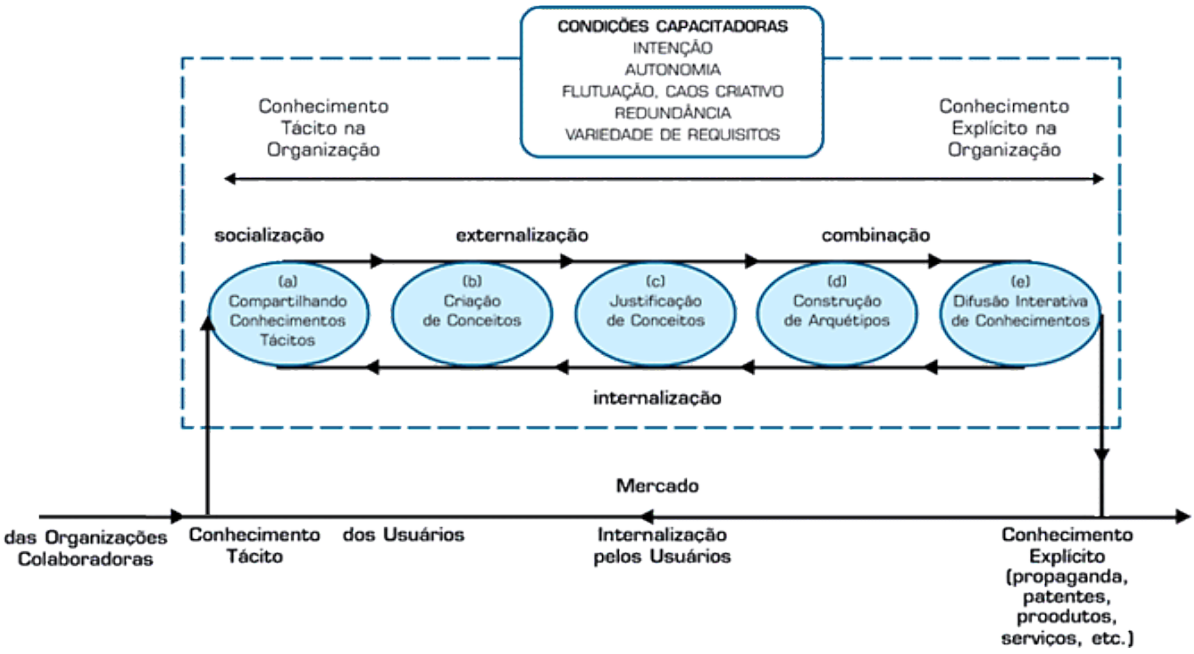

Figura 3 – Modelo de cinco fases do processo de criação de conhecimento.
Fonte: Nonaka e Takeuchi (1997, p. 96)

Primeira fase: Compartilhamento do conhecimento tácito

Conforme explicam Nonaka e Takeuchi (1997), uma organização não pode criar conhecimento sozinha. Ela depende do conhecimento tácito dos indivíduos que constitui a base da criação de conhecimento organizacional. Assim, é natural que o processo de criação de conhecimento comece pelo compartilhamento do conhecimento tácito dos indivíduos, o que corresponde aproximadamente à socialização, pois o conhecimento rico e inexplorado dos indivíduos, que apresentam diferentes experiências e históricos pessoais, perspectivas e motivações, emoções, sentimentos e modelos mentais, deve ser compartilhado e amplificado dentro da organização.

Para atingir esse compartilhamento é necessário um ambiente de interação por meio de diálogos pessoais, pois os membros compartilharão experiências e sincronizarão seus ritmos corporais e mentais. O ambiente de interação típico é a equipe auto-organizada e diversificada, na qual a autonomia dos indivíduos e a variedade de departamentos funcionais que traba-

lham juntos para alcançar uma meta comum favorece essa troca de experiências, inclusive com o ambiente externo (NONAKA; TAKEUCHI, 1997, p. 96-98).

Segunda fase: Criação de conceitos

O compartilhamento tácito compartilhado por meio do diálogo contínuo da primeira fase é expresso sob a forma de reflexão coletiva. Esses modelos mentais formados são verbalizados em palavras, fórmulas, esquemas e diagramas cristalizando em conceitos explícitos, o que corresponde a externalização (NONAKA; TAKEUCHI, 1997, p. 96-98).

O uso de múltiplos métodos de raciocínio, como a dedução, indução e abdução ganha grande importância. A abdução ganha destaque nessa etapa pois emprega a linguagem figurativa como metáforas e analogias. A qualidade do diálogo pode ser aprimorada com o uso da dialética pois utiliza ferramentas como contradições e paradoxos para sintetizar o novo conhecimento (NONAKA; TAKEUCHI, 1997, p. 98).

Terceira fase: Justificação de conceitos

Os novos conceitos criados em algum momento devem ser justificados, na tentativa de descobrir se os conceitos criados na segunda fase valem realmente a pena para a organização e sociedade. Essa etapa de filtragem permite a organização conduzir essa justificação de forma explícita, garantindo a direção proveniente da intenção organizacional e ter certeza que os conceitos que estão sendo criados geram valor e atendem às necessidades da sociedade de forma mais ampla (NONAKA; TAKEUCHI, 1997, p. 96-100).

Os critérios de justificação podem ser tanto qualitativos quanto quantitativos. Os critérios mais comuns incluem custo, margem de lucro e grau de contribuição do produto para o crescimento da empresa. Critérios mais abstratos podem incluir premissas de valor como aventura, romantismo e estética (NONAKA; TAKEUCHI, 1997, p. 96-100).

Quarta fase: Construção do arquétipo

Nesta fase, um conceito justificado é transformado em algo tangível ou concreto. Um arquétipo pode ser um protótipo ou mesmo um novo valor da empresa, um novo modelo de negócio, um sistema gerencial inovador ou uma nova estrutura organizacional. Como os concei-

tos justificados (explícitos) são combinados em forma de um arquétipo (também explícito) essa fase se assemelha à combinação (NONAKA; TAKEUCHI, 1997, p. 96-101).

Quinta fase: Difusão interativa do conhecimento

A quinta fase amplia o conhecimento criado para toda organização ou mesmo à componentes externos, como clientes, parceiros e fornecedores. Uma empresa criadora de conhecimento não opera em um sistema fechado, mas sim em um ambiente aberto que permite um intercambio constante de conhecimento com o ambiente externo (NONAKA; TAKEUCHI, 1997, p. 96-102).

2.2 Metodologias de Desenvolvimento de Software

Esta seção apresenta os principais conceitos e as bases teóricas referentes a engenharia de software: desenvolvimento de software tradicional e ágil.

2.2.1 Desenvolvimento de Software

Existem diversas definições para software, mas de maneira geral os autores concordam que o software não é apenas o código em execução sobre algum hardware, mas também todos os dados de documentação e configuração associados, necessários para que o programa opere corretamente (SOMMERVILLE, 2007, p. 4). Ou seja, software de computador é o produto que os profissionais de software constroem e mantém ao logo do tempo e abrange programas que executam em qualquer tipo de computador (PRESSMAN, 2006, p. 1).

Além disso, Pressman (2006) diferencia sete amplas categorias de software de computadores, são elas: software de sistemas, software de aplicação, software científico e de engenharia, software embutido, software para linhas de produtos, software para inteligência artificial e aplicações da web.

Os avanços significativos no hardware e no software nos últimos 50 anos mudaram os rumos das indústrias e do mercado. Segundo Pressman (2006), hoje, o software se apresenta como a tecnologia única mais importante no palco mundial, pois é ao mesmo tempo o produto e atua como o veículo de entrega do mais importante produto de nossa época: a informação. O software está embutido em todos os tipos de sistemas, sejam sistemas de transporte, de teleco-

municações, médicos, militar, industrial, entretenimento, máquina de escritório e uma infinidade de outros sistemas (PRESSMAN, 2006).

Segundo Sommerville (2007, p. 5), “a engenharia de software é uma disciplina de engenharia relacionada a todos os aspectos da produção de software, desde os estágios iniciais de especificação do sistema até sua manutenção, depois que este entrar em operação.”

Os estudiosos também destacam a importância da engenharia de software para que os software pudessem alcançar o nível de complexidade de hoje. Metodologias, técnicas e ferramentas também evoluíram quase que na mesma velocidade dos software desenvolvidos e os responsáveis por essa evolução são os engenheiros de software.

De maneira genérica, um processo de software pode ser definido como “um conjunto de atividades e resultados associados que produz um produto de software” (SOMMERVILLE, 2007, p. 6). Durante os últimos 45 anos vários métodos foram propostos para orientar os processos de software de maneira sistemática. “Um método de engenharia de software é uma abordagem estruturada para desenvolvimento de software, cujo objetivo é facilitar a produção de software de alta qualidade dentro de custos adequados.” (SOMMERVILLE, 2007, p. 8).

O grande desafio da engenharia de software é criar maneiras sistemáticas para integrar o conhecimento de todos os atores envolvidos ao software.

> Desde que o software, como todo capital, é conhecimento incorporado, e como esse conhecimento está inicialmente disperso, tácito, latente e incompleto na sua totalidade, o desenvolvimento de software é um processo de aprendizado social. O processo é um diálogo no qual o conhecimento, que deve se transformar em software, é reunido e incorporado ao software. O processo fornece interações entre usuários e projetistas, entre usuários e ferramentas em desenvolvimento e entre projetistas e ferramentas em desenvolvimento [tecnologia]. É um processo interativo no qual a própria ferramenta serve como meio de comunicação, com cada nova rodada de diálogo explicitando mais conhecimento útil do pessoal envolvido.
>
> (BAETJER, 1998 apud PRESSMAN, 2006, p. 16)

Durante a década de 1980, o SEI (*Software Engineering Institute*) desenvolveu o CMMI (*Capability Maturity Model Integration*), “um metamodelo de processo baseado em um con-

junto de capacidades de engenharia de software que devem estar presentes à medida que as empresas alcançam diferentes níveis de capacidade e maturidade de processo.” (PRESSMAN, 2006, p. 21). Nesse metamodelo são classificados seis níveis de capacitação de acordo com o nível de gerenciamento, são eles: (Nível 0) incompleto, (Nível 1) realizado, (Nível 2) gerido, (Nível 3) definido, (Nível 4) quantitativamente gerido e (Nível 5) otimizado.

2.2.2 Metodologia de Desenvolvimento Tradicional

As metodologias tradicionais apresentam uma estrutura genérica de processo independente do modelo. Pressman (2006) chama essa estrutura de arcabouço de processos que inclui: comunicação, planejamento, modelagem, construção e implantação.

Da abordagem tradicional, baseada em repositórios, destacam-se modelos de processo de desenvolvimento de software como modelo cascata, modelos incrementais, modelos evolucionários e modelos especializados de processos.

Pressman (2006) chama as metodologias tradicionais de modelos prescritivos. O autor explica a preferência por essa nomenclatura devido ao fato deles prescreverem um conjunto de elementos de processos que são atividades de arcabouço, ações de engenharia de software, mecanismos de garantia de qualidade e controle de versões de projeto. Cada modelo de processo prescreve um fluxo de trabalho (PRESSMAN, 2006, p. 38).

Modelos prescritivos de processo definem um conjunto distinto de atividades, ações, tarefas, marcos e produtos de trabalho que são necessários para fazer engenharia de software com alta qualidade. Esses modelos de processo não são perfeitos, mas efetivamente fornecem um roteiro útil para o trabalho de engenharia de software (PRESSMAN, 2006, p. 37).

De maneira geral podemos destacar a divisão, ou atividades de arcabouço, muito bem definida das etapas das metodologias tradicionais, seja no modelo cascata, espiral ou qualquer outra metodologia prescritiva. Essa divisão pode levar a especialização do trabalho em equipes que não forem multifuncionais, daí a justificação de alguns autores a chamarem as metodologias tradicionais de metodologias tayloristas.

Outro ponto que podemos destacar é a necessidade de rigorosa documentação, seja na dependência de uma análise de requisito extremamente elaborada ou ao final de cada etapa

quando são gerados documentos atualizados ou mesmo uma nova versão do sistema. Desta maneira a documentação é extremamente exaustiva e requer muito tempo e esforço para mantê-la atualizada.

2.2.3 Metodologia de Desenvolvimento Ágil

No ambiente global atual, os negócios estão sujeitos a rápidas mudanças e desta forma é praticamente impossível elaborar um conjunto completo de requisitos no início de projetos de software. Muitas vezes, nem os próprios clientes que apresentam o conhecimento de domínio sabem exatamente o que precisam para seus negócios. A engenharia de software foi obrigada e rever alguns conceitos para oferecer a agilidade necessária à resposta às mudanças no decorrer do projeto. Baseada nos sucessos dos processos enxutos (*lean*) da indústria de bens tangíveis durante a década de 1990 foram criados alguns métodos que quebravam a linha vigente das metodologias tradicionais e enfatizavam a resposta às mudanças e ao fluxo de valor.

Segundo Pressman (2006, p. 58), em 2001, 17 notáveis desenvolvedores, produtores e consultores de software criaram a Aliança Ágil (*Agile Alliance*), e assinaram o “Manifesto para o Desenvolvimento Ágil de Software”, eles declararam:

> Estamos descobrindo melhores modos de desenvolvimento de software fazendo-o e ajudando outros a fazê-lo. Por meio desse trabalho passamos a valorizar:
>
> *Indivíduos e interações* em vez de processos e ferramentas
> *Software funcionando* em vez de documentação abrangente
> *Colaboração do cliente* em vez de negociações de contratos
> *Resposta a modificações* em vez de seguir um plano
>
> Isto é, ainda que haja valor nos itens à direita, valorizamos mais os itens à esquerda. (AGILE ALLIANCE, 2003 apud PRESSMAN, 2006, p. 58)

A dinâmica dos negócios atuais, com mudanças contínuas no ambiente e necessidades de resposta às mudanças cada vez mais rápidas fez com que as metodologias tradicionais não suportassem mais muitos projetos de software. O desenvolvimento ágil pode fornecer importantes benefícios, porém não pode ser aplicado a todos os projetos, produtos, pessoas e situações (PRESSMAN, 2006, p. 58-59).

Cockburn (2007, p. 222) afirma que ser ágil implica ser eficaz e flexível e um processo ágil é leve e suficiente. A leveza é o meio de ser flexível e a suficiência é o meio de permanecer no jogo. A necessidade de resposta à mudança privilegiou a agilidade na engenharia de software. Larman (2004) corrobora as palavras de Cockburn afirmando que as metodologias ágeis têm um lema ele seria abraçar a mudança. Se as metodologias ágeis têm um ponto estratégico, ele seria a capacidade de manobra (flexibilidade).

Pressman (2006, p. 59-60) aponta que o principal guia para a agilidade é o poder de modificação. Os engenheiros de software devem reagir rapidamente para acomodar as mudanças. Porém a agilidade é mais do que isso. Ela engloba toda a filosofia do manifesto ágil, ou seja, ela enfatiza a rápida entrega de software operacional, adota clientes como parte da equipe de desenvolvimento e procura eliminar a distância “nós e eles” que continua a permear muitos projetos de software. Ela reconhece que o planejamento em um mundo incerto tem seus limites e que um plano de projeto deve ser flexível.

A Aliança Ágil também define 12 princípios para se alcançar a agilidade:

1. Nossa maior prioridade é satisfazer ao cliente desde o início por meio de entrega contínua de software valioso.
2. Modificações de requisitos são bem-vindas, mesmo que tardias no desenvolvimento. Os processos ágeis aproveitam as modificações como vantagem para a competitividade do cliente.
3. Entrega de software funcionando frequentemente, a cada duas semanas até dois meses, de preferência no menor espaço de tempo.
4. O pessoal de negócio e os desenvolvedores devem trabalhar em conjunto diariamente durante todo o projeto.
5. Construção de projetos em torno de indivíduos motivados. Forneça-lhes o ambiente e apoio que precisam e confie que eles farão o trabalho.
6. O método mais eficiente e efetivo para levar informação para dentro de uma equipe de desenvolvimento é a conversa face a face.
7. Software funcionando é a principal medida de progresso.
8. Processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante, indefinidamente.
9. Atenção contínua à excelência técnica e ao bom projeto facilitam a agilidade

10. Simplicidade – a arte de maximizar a quantidade de trabalho não efetuado – é essencial.
11. As melhores arquiteturas, requisitos e projetos surgem de equipes auto-organizadas.
12. Em intervalos regulares, a equipe reflete sobre como se tornar mais efetiva, então sintoniza e ajusta adequadamente seu comportamento.

(AGILE ALLIANCE, 2003 apud Agile Alliance, 2003)

Qualquer processo ágil de software é caracterizado de modo que atenda a três suposições chaves:

1. É difícil prever antecipadamente quais requisitos de software vão persistir e quais serão modificados. É igualmente difícil prever como as prioridades do cliente serão modificadas à medida que o projeto prossegue.
2. Para muitos tipos de software, o projeto e a construção são intercalados, isto é, as duas atividades devem ser realizadas juntas de modo que os modelos de projeto sejam comprovados à medida que são criados. É difícil prever o quanto de projeto é necessário antes que a construção seja usada para comprovar o projeto.
3. Análise, projeto, construção e testes não são tão previsíveis (do ponto de vista do planejamento) como gostaríamos.

(FOWLER, 2002 apud PRESSMAN, 2006, p. 60-61)

O processo deve se moldar às necessidades das pessoas e da equipe, e não o contrário (PRESSMAN, 2006, p. 62). Assim, Pressman (2006) enfatiza o "fator humano" e enumera sete características do processo que devem existir entre as pessoas de uma equipe ágil e na equipe em si, são eles: competência, foco comum, colaboração, capacidade de tomada de decisão, habilidade de resolver problemas vagos, respeito e confiança mútua e auto-organização.

Existem diversas metodologias ágeis de desenvolvimento de software, como a Programação Extrema *(eXtreme Programming* – XP), o Scrum, o Desenvolvimento Adaptativo de Software (DAS), o Método de Desenvolvimento Dinâmico de Sistemas (*Dynamic System Development Method* – DSDM), o Desenvolvimento Guiado por Características (*Feature Driven Development* - *FDD*), o Crystal, e a Modelagem Ágil (*Agile Modeling* – AM). "Embora esses métodos ágeis sejam todos baseados na noção de desenvolvimento e entregas incrementais,

eles propõem processos diferentes para se conseguir isso. Contudo compartilham um conjunto de princípios e, portanto, têm muito em comum.” (SOMMERVILLE, 2007, p. 262). Neste trabalho veremos as metodologias XP e Scrum em maiores detalhes.

2.2.3.1 Scrum

O Scrum é um modelo ágil de processo que foi desenvolvido por Jeff Sutherland e por sua equipe em 1993. Sutherland (2012) afirma que se baseou no artigo de Takeuchi e Nonaka (1986) “*The New New Product Development Game*”, publicado na Harvard Business Review, no qual, os professores japoneses apontam as melhores técnicas, descrevendo as melhores equipes que viram em todo o mundo como a auto-organização de equipes por meio de uma analogia com o *rúgbi*. Isto é, os membros da equipe são totalmente responsáveis, colocam o time em primeiro lugar e todo mundo aprende a fazer os trabalhos de todos. O líder da equipe tem um estilo facilitador, ajuda a equipe a traduzir os objetivos da alta gerência em implementação prática, e se torna um catalisador para a inovação e transformação organizacional.

O Scrum está baseado nas teorias empíricas de controle de processo. O empirismo afirma que o conhecimento vem da experiência e deve-se tomar decisões baseadas no que se conhece. Três pilares sustentam a implementação de um controle de processo empírico: transparência, inspeção e adaptação (SCHWABER; SUTHERLAND, 2011, p. 3). O Scrum aplica uma abordagem iterativa e incremental para aperfeiçoar a previsibilidade e o controle de riscos.

Segundo Advanced Development Methods, (1996 apud Pressman 2006) os princípios Scrum são consistentes com o manifesto ágil:

- Pequenas equipes de trabalho são organizadas de modo a “maximizar a comunicação, minimizar a supervisão e maximizar o compartilhamento de conhecimento tácito informal”.
- O processo precisa ser adaptável tanto as modificações técnicas quanto de negócios “para garantir que o melhor produto possível seja produzido”.
- O processo produz frequentes incrementos de software “que podem ser inspecionados, ajustados, testados, documentados e expandidos”.
- O trabalho de desenvolvimento e o pessoal que o realiza é dividido “em participações claras, de baixo acoplamento ou em pacotes”.

- Testes e documentação constantes são realizados à medida que o produto é construído.
- O processo Scrum fornece a "habilidade de declarar o produto 'pronto' sempre que necessário (porque a concorrência acabou de entregar, porque a empresa precisa de dinheiro, porque o usuário/cliente precisa das funções, porque foi pra essa data que foi prometido...)".

(ADVANCED DEVELOPMENT METHODS, 1996 apud PRESSMAN, 2006, p. 69)

O Scrum é um "*framework* que consiste nas equipes do Scrum, papéis, eventos, artefatos e regras, associados. Cada componente do *framework* serve a um propósito específico e é essencial para o sucesso e uso do Scrum." (SCHWABER; SUTHERLAND, 2011, p. 3, grifo meu).

Equipe do Scrum

Segundo Schwaber e Sutherland (2011, p. 4-5), a Equipe Scrum é formada pelo Dono do Produto (*Product Owner*), pela Equipe de Desenvolvimento e pelo *Scrum Master*. São equipes auto-organizadas que escolhem a melhor forma de realizar seu trabalho. São também equipes transfuncionais e multifuncionais que apresentam todas as competências necessárias para realizar o trabalho sem a dependência de outros que não fazem parte da equipe. Dessa forma a Equipe Scrum é formada para aperfeiçoar a flexibilidade, criatividade e produtividade. A produção é iterativa e incremental de forma a maximizar as oportunidades para realimentação. Incrementos de produtos 'Prontos' são entregues ao cliente como uma versão potencialmente útil do produto (SCHWABER; SUTHERLAND, 2011, p. 5).

O Dono do Produto é o "representante" do cliente dentro da equipe de desenvolvimento, ele deve direcionar a equipe com as prioridades daquilo que se deve fazer, sempre com uma finalidade em mente: agregar valor ao cliente. O Dono do Produto é responsável por maximizar o valor do produto (SCHWABER; SUTHERLAND, 2011, p. 5). Dessa maneira, uma das atribuições do Dono do Produto é gerenciar o Backlog do Produto (*Product Backlog*). O Backlog do Produto é uma espécie de lista dos requisitos, que nunca está completa. O Dono do Produto, ao elaborar o Backlog do Produto deve procurar:

- Expressar com clareza os itens do Backlog do Produto;
- Ordenar os itens do Backlog do Produto para melhor alcançar os objetivos e missões;
- Garantir o valor do trabalho desempenhado pela equipe de desenvolvimento;

- Garantir que o Backlog do Produto seja visível, transparente e claro para todos, e mostre no que a equipe do Scrum trabalhará em seguida;
- Garantir que a equipe de desenvolvimento entenda os itens do Backlog do Produto no nível necessário.

O Dono do produto pode fazer o trabalho acima, ou deixar que a equipe de projeto o faça. Entretanto, o Dono do produto é o responsável por ele.

(SCHWABER; SUTHERLAND, 2011, p. 5)

Segundo Schwaber e Sutherland (2011) a equipe de desenvolvimento do Scrum é responsável por entregar uma parte do incremento utilizável ao final de cada Sprint. Essas equipes devem ser auto-organizadas, ou seja, ninguém diz como transformar o Backlog do Produto em incrementos de funcionalidades potencialmente utilizáveis; devem estar cientes que pertencem a um grupo, respondem à equipe de desenvolvimento como um todo, como quando os jogadores de rúgbi se armam para fazer uma jogada Scrum, cada um pode ter sua especialidade, mas devem se completarem formando uma equipe sólida. Por fim, as equipes de desenvolvimento não possuem subequipes dedicadas a domínios particulares do conhecimento, como teste ou análise de negócio.

O *Scrum Master* é responsável por assegurar que o Scrum seja entendido e aplicado de forma a garantir que as equipes Scrum obedeçam à teoria, às práticas e às regras do *framework*. O *Scrum Master* é um líder facilitador para a equipe que ajuda aqueles de fora da equipe do Scrum entender quais são as interações com a equipe do Scrum que são benéficas e quais não são. O *Scrum Master* ajuda todos a mudar suas interações para maximizar o valor criado pela equipe do Scrum (SCHWABER; SUTHERLAND, 2011, p. 6).

O *Scrum Master* tem um papel fundamental na comunicação entre o Dono do produto e a Equipe de Desenvolvimento. Ele deve entender a visão a longo prazo do produto em um ambiente empírico e comunicar claramente a visão, objetivos e os itens do Backlog do Produto para a Equipe de Desenvolvimento (SCHWABER; SUTHERLAND, 2011, p. 6).

Eventos do Scrum

Eventos prescritos criam uma rotina para minimizar a necessidade de encontros não definidos no Scrum. O Scrum usa eventos *time-boxed*, no qual cada evento tem uma duração

máxima, o que garante uma quantidade apropriada de tempo gasto em planejamento sem permitir perdas no processo de planejamento (SCHWABER; SUTHERLAND, 2011, p. 7).

O Sprint é um processo que dura no máximo um mês para entregar uma versão do incremento. Schwaber e Sutherland (2011) explicam que o coração do Scrum é o Sprint, um *time-boxed* de um mês ou menos durante o qual uma versão potencialmente utilizável de um incremento do produto é criada. Sprints têm da conclusão do Sprint anterior. Sprints consistem em reunião de planejamento do Sprint, Scrums Diários, trabalho de desenvolvimento, reunião de Revisão do Sprint e Retrospectiva do Sprint (SCHWABER; SUTHERLAND, 2011, p. 7).

O trabalho a ser executado no Sprint é planejado na Reunião de Planejamento do Sprint. Este planejamento é feito por toda a equipe do Scrum de maneira colaborativa. Um objetivo também é criado com a finalidade de oferecer uma certa flexibilidade à equipe de desenvolvimento sobre as funcionalidades que serão implementadas no Sprint e manter o objetivo da equipe sempre em mente (SCHWABER; SUTHERLAND, 2011, p. 8-10).

Schwaber e Sutherland (2011, p. 10) explicam que o Scrum Diário, as reuniões diárias do Scrum, são reuniões curtas de no máximo 15 minutos para a Equipe de Desenvolvimento sincronizar as atividades e criar um plano para as próximas 24 horas. Segundo Pressman (2006), durante a reunião, cada membro da Equipe de Desenvolvimento esclarece:

> - O que você fez desde a última reunião da equipe?
> - Que obstáculos você está encontrando?
> - O que você planeja realizar até a próxima reunião de equipe?
>
> (PRESSMAN, 2006, p. 70)

A Reunião de Revisão do Sprint é executada ao final do Sprint para inspecionar o incremento e adaptar ao Backlog do Produto se necessário. Já a Retrospectiva do Sprint é uma oportunidade para a Equipe do Scrum inspecionar-se a si mesma e criar um plano de melhorias que deve valer durante o próximo Sprint (SCHWABER; SUTHERLAND, 2011, p. 8-10).

Artefatos do Scrum

O Backlog do Produto é uma lista ordenada de tudo o que pode ser necessário no produto e é a fonte única dos requisitos para qualquer mudança a ser feita no produto. Um Backlog do Produto nunca está completo. Ele lista todas as funcionalidades, funções, requisitos, melhorias

e consertos que representam as mudanças a serem feitas no produto para as próximas versões. Seus itens têm os atributos de descrição, ordem e estimativa de esforço. É, geralmente, ordenado pelo valor que agrega para o negócio, pelo risco, por prioridade e por necessidade. Os itens mais altos do Backlog do Produto determinam atividades de desenvolvimento mais imediatas (SCHWABER; SUTHERLAND, 2011, p. 12).

O Backlog do Sprint é um conjunto de itens do Backlog do Produto, selecionados para um Sprint além de um plano para obter o incremento do produto e atingir o objetivo do Sprint. O Backlog do Sprint define o trabalho que a Equipe de Desenvolvimento desempenhará para transformar os itens do Backlog do Produto em Incrementos “Prontos” (SCHWABER; SUTHERLAND, 2011, p. 13). No final de um Sprint, o novo incremento deve estar “Pronto”, o que significa que ele está em uma condição utilizável e atende à definição da Equipe do Scrum do “Pronto”.

Técnicas como o *burndown* são utilizadas com bastante sucesso para estimar os tempos de conclusão de cada Sprint e mesmo do projeto inteiro. Entretanto, não substituem a importância do empirismo e da experiência. Em ambientes complexos, é impossível saber tudo o que vai acontecer. Somente o que aconteceu pode ser usado para uma tomada de decisão do que está por vir (SCHWABER; SUTHERLAND, 2011, p. 13).

A Equipe de Desenvolvimento entrega um incremento da funcionalidade do produto a cada Sprint. Este incremento é usável, assim o Dono do Produto pode decidir liberá-lo imediatamente. Cada incremento é adicionável a todos os incrementos anteriores e por meio de testes, garantem que todos os incrementos trabalhem em conjunto (SCHWABER; SUTHERLAND, 2011, p. 14).

Embora não apareça nas descrições de Schwaber e Sutherland (2011) é comum utilizar o Scrum em conjunto com o *Kamban* (cartão de sinalização) ou com o *Task Board* (quadro visual) para o controle do fluxo de produção do software. Este controle visual é especialmente útil na transparência do que cada um está fazendo e em que situação se encontra o processo de desenvolvimento do software.

A Figura 4, mostra o *framework* scrum, seus membros, eventos e artefatos.

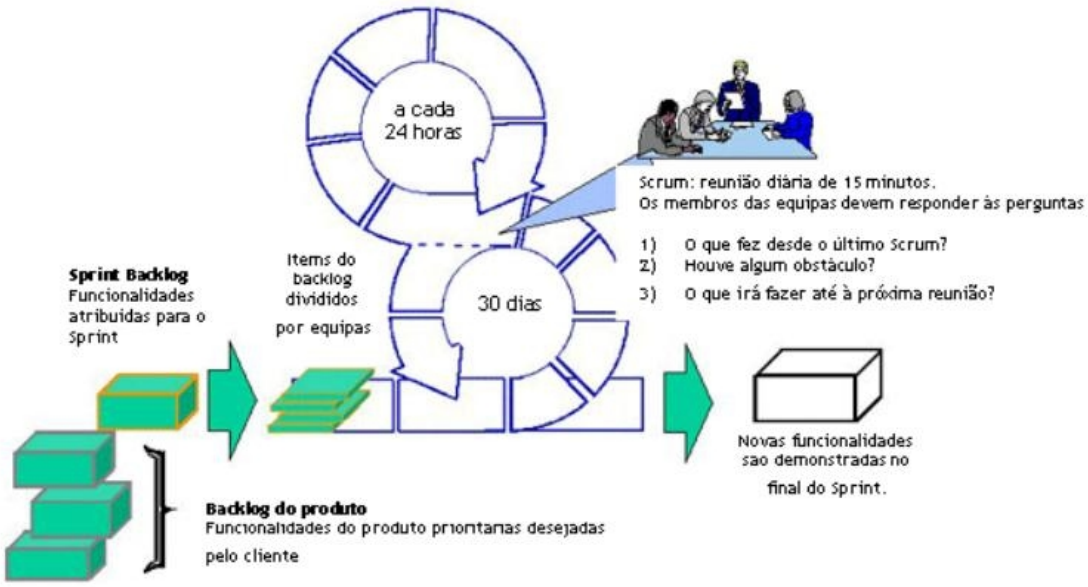

Figura 4 – O *framework* Scrum
Fonte: Pressman (2006, p. 70)

2.2.3.2 eXtreme Programming (XP)

A metodologia XP utiliza a abordagem orientada a objetos. Ela contempla um conjunto de regras e prática em quatro atividades de arcabouço: planejamento, projeto, codificação e teste (PRESSMAN, 2006, p. 63).

Pressman (2006, p. 63) explica que a atividade de planejamento consiste na criação de um conjunto de histórias de usuários que descrevem um conjunto de características e funcionalidades requeridas para o software a ser construído. O cliente atribui um valor ou prioridade para cada história. Os membros da equipe XP atribuem um custo para cada história. Cada história deve ter um custo de no máximo três semanas, caso o custo seja maior, pede-se ao cliente para dividir as histórias de usuários em histórias menores e a atribuição de valor e custo ocorrem novamente. Os clientes e desenvolvedores trabalham juntos para decidir como agrupar histórias para o incremento seguinte. Depois que o primeiro incremento é finalizado, a equipe XP calcula a velocidade de projeto que é a quantidade de histórias implementadas na primeira versão. Com isso é possível estimar o prazo de entrega e cronogramas das versões seguintes. À medida que o projeto anda, o cliente pode adicionar novas histórias.

Vale ressaltar o uso de metáforas utilizada na XP que tem como objetivo ajudar a comunicação entre o cliente e o grupo de desenvolvimento. Por meio das metáforas é possível diminuir a lacuna entre o cliente que detém o conhecimento de domínio e os desenvolvedores que detêm o conhecimento do sistema. Alinhar as duas visões é de extrema importância.

Como explica Pressman (2006), um projeto simples é sempre preferível do que um projeto complexo, desta maneira a metodologia XP segue rigorosamente o princípio KIS (*keep it simple* – mantenha a simplicidade). O projeto fornece diretrizes de implementação para uma história como ela está escrita, nada mais, nada menos. A XP encoraja o uso de cartões CRC (*Class – Responsibility – Collaborator*) que identificam e organizam as classes orientadas a objetos que são relevantes para o incremento de software atual. Os cartões CRC são o único produto de trabalho do projeto que é realizado como parte do processo XP. Em caso de projetos difíceis, utiliza-se a criação de um protótipo operacional, denominado solução de ponta, que é implementado e avaliado. A XP também encoraja a refabricação que consiste num aperfeiçoamento do código depois que ele foi escrito (PRESSMAN, 2006, p. 64).

Um conceito-chave durante a atividade de codificação na XP é a programação em pares. A XP recomenda que duas pessoas trabalhem juntas em uma estação de trabalho de computador para criar o código correspondente a uma história. Isso fornece um mecanismo de solução de problemas em tempo real (duas cabeças são frequentemente melhor do que uma) e garante qualidade em tempo real. Também mantém os desenvolvedores focados no problema em mãos. Na prática cada pessoa assume um papel ligeiramente diferente. Por exemplo, uma pessoa poderia pensar nos detalhes de código de uma parte específica do projeto, enquanto a outra garantiria que as normas de codificação (parte necessária no XP) estejam sendo seguidas e que o código gerado vai se “encaixar” no projeto mais amplo da história (PRESSMAN, 2006, p. 65).

Cabe ressaltar a ideia de propriedade coletiva do código fonte. O código fonte não tem um dono exclusivo e ninguém precisa de uma permissão formal para modificá-lo. Embora necessite de ferramentas de sincronização para que não haja perda de trabalho, essa prática da XP permite que toda a equipe conheça todas as partes do sistema.

Outra prática que merece ser destacada é a integração contínua. Cada interação ou funcionalidade criada para o software deve ser integrada imediatamente à versão atual do sistema. Isso

minimiza os problemas de conflito com as versões anteriores e permite saber o andamento real do desenvolvimento.

Pressman (2006) afirma que a metodologia XP recomenda criar uma série de testes unitários, antes da codificação, que exercitarão cada uma das histórias que devem ser incluídas na versão atual, pois uma vez criados os testes o desenvolvedor sabe o que precisa ser feito para passar no teste unitário. Os testes unitários são criados de maneira que possam ser automatizados e fáceis de executar, pois serão executados repetidamente. Isso encoraja uma estratégia de teste de regressão sempre que o código é modificado. O teste de integração e validação pode ocorrer diariamente, fornecendo uma indicação contínua de progresso e também podem ligar sinais de alerta se as coisas estiverem se deteriorando. Por fim, os testes de aceitação ou testes do cliente são testes feitos pelo cliente com base nas histórias de clientes e focalizam as características e funcionalidades do sistema global que são visíveis e passível de revisão do cliente (PRESSMAN, 2006, p. 65-66). A Figura 5 ilustra as etapas do XP.

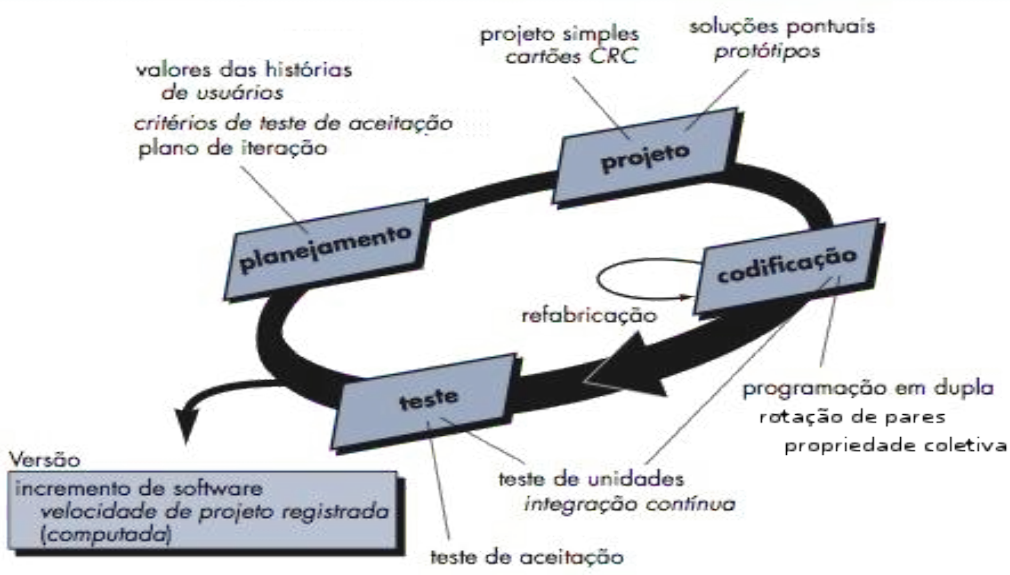

Figura 5 – O processo eXtreme Programming (XP)

Fonte: Adaptado de Pressman (2006)

3 REVISÃO DE LITERATURA

Uma das formas de analisar e discutir os artigos publicados em uma área específica do conhecimento é a revisão de literatura. Ela pode ter como objetivo a verificação de textos e artigos relacionados ao assunto estudado e que já foram publicados em congressos, simpósios, revistas e foram cadastrados em alguma base de dados ou conhecer a forma como esse assunto foi abordado e analisado em estudos anteriores.

Segundo Macedo (1994, p. 3), a revisão de literatura consiste em uma varredura e seleção de documentos do que foi publicado sobre o assunto, a fim de que os pesquisadores não reinvente a roda.

3.1 PRISMA

Moher et al. (2009) afirmam que a revisão sistemática é uma revisão de uma pergunta claramente formulada que usa métodos sistemáticos e explícitos para identificar, selecionar e avaliar criticamente pesquisas relevantes, coletar e analisar dados dos estudos incluídos na revisão. Métodos estatísticos (meta-análise) podem ou não serem utilizados para analisar e resumir os resultados dos estudos incluídos.

De maneira semelhante Petticrew e Roberts (2006, p. 10-11) definem as revisões sistemáticas como revisões de literatura que comportam um conjunto de métodos científicos que visam limitar o erro sistemático, principalmente pela tentativa de identificar, avaliar e sintetizar todos os estudos relevantes, a fim de responder a uma questão particular (ou um conjunto de perguntas). As revisões sistemáticas são particularmente valiosas como um meio de analisar todas as evidências sobre uma questão particular, se há alguma incerteza sobre a resposta. Se não está claro se uma determinada intervenção é eficaz, então, uma revisão sistemática da evidência disponível pode ajudar a resolver o problema.

Foi utilizado um “*framework*" de revisão sistemática de literatura, conhecido como PRISMA (*Preferred Reporting Items for Systematic Reviews and Meta-Analyses*), que auxiliou encontrar o tema buscado de forma abrangente. Moher et al. (2009) afirmam que o PRISMA tem o objetivo de ajudar os autores a melhorar a comunicação de revisões sistemáticas e meta-análises e consiste em uma lista de 27 itens e um diagrama de fluxo de quatro fases.

Para esta revisão de literatura, a pesquisa bibliográfica teve como tema ***"gestão do conhecimento no desenvolvimento de software"***, com foco temático relacionado a metodologias ágeis e procurou responder a seguinte questão: **"Quais são as principais técnicas e os principais fatores das metodologias de desenvolvimento de software que contribuem para a criação de conhecimento organizacional?"**.

Os textos foram procurados nas bases bibliográficas IEEExplore, ACM Digital Library e ScienceDirect. Os principais conceitos, *conhecimento* e *desenvolvimento de software*, foram combinados por meio de operadores booleanos *E* (*AND*). Para refinar a busca, foi predeterminado que o conceito "conhecimento" deveria estar explícito no título; o conceito "desenvolvimento de software" e suas variantes relevantes para esta pesquisa (metodologias ágeis, desenvolvimento ágil de software, melhoria de software, scrum, programação extrema, XP) foram combinados com o operador booleano *OU* (*OR*) e poderiam constar em qualquer parte dos textos. Por exemplo, a expressão exata utilizada para a base IEEExplore foi:

> *(("Document Title":"knowledge") AND ("software development" OR "agile methodology" OR "scrum" OR "agile software development" OR "extreme programming" OR "XP" OR "software environment"))*

Após aplicar as estratégias de busca nas três bases, foram encontrados 1765 artigos. Foram limitados aos artigos de janeiro de 2002 até junho de 2016, dessa maneira, restaram 1159 estudos. Na etapa de eliminação por título foi dada prioridade aos artigos que discutem sobre a gestão do conhecimento e as metodologias de desenvolvimento de software. Após a eliminação por título, restaram 452 publicações. Na etapa de eliminação por resumo se deu prioridade àqueles artigos que discutem o conhecimento nas duas formas: tácito e explícito. Em seguida foram eliminados 260 artigos após a leitura dos resumos, resultando em 192 artigos. Durante a eliminação por foco temático foram preteridos aqueles artigos que discutiam apenas sobre documentação (conhecimento explícito), aqueles que focavam exclusivamente em sistemas de informação para equipes distribuídas e aqueles que abordavam como tema principal o CMMI (*Capability Maturity Model – Integration*) ou o SWEBOK (*The Guide to the Software Engineering Body of Knowledge*) restando 41 artigos. Após essas eliminações foram encontrados três artigos duplicados. Finalmente, após aplicar as fases do diagrama de fluxo do PRISMA, a busca resultou em 38 artigos. A Figura 6 exibe os resultados obtidos e as principais fases do processo por meio de um diagrama PRISMA.

Após a seleção dos artigos foi feita uma tentativa de agrupar os conceitos semelhantes. Assim, foram agrupados de acordo com os focos abordados e pertinentes à dimensão epistemológica e à dimensão ontológica conforme o trabalho de Nonaka e Takeuchi (1997) e as abordagens de desenvolvimento de software (tradicionais e ágeis), além do espaço de trabalho e cultura organizacional. Nessa classificação foi feita uma análise qualitativa dos resultados das pequisas. Além dos resultados e conclusões, também foram considerados as premissas e os embasamentos dos artigos selecionados.

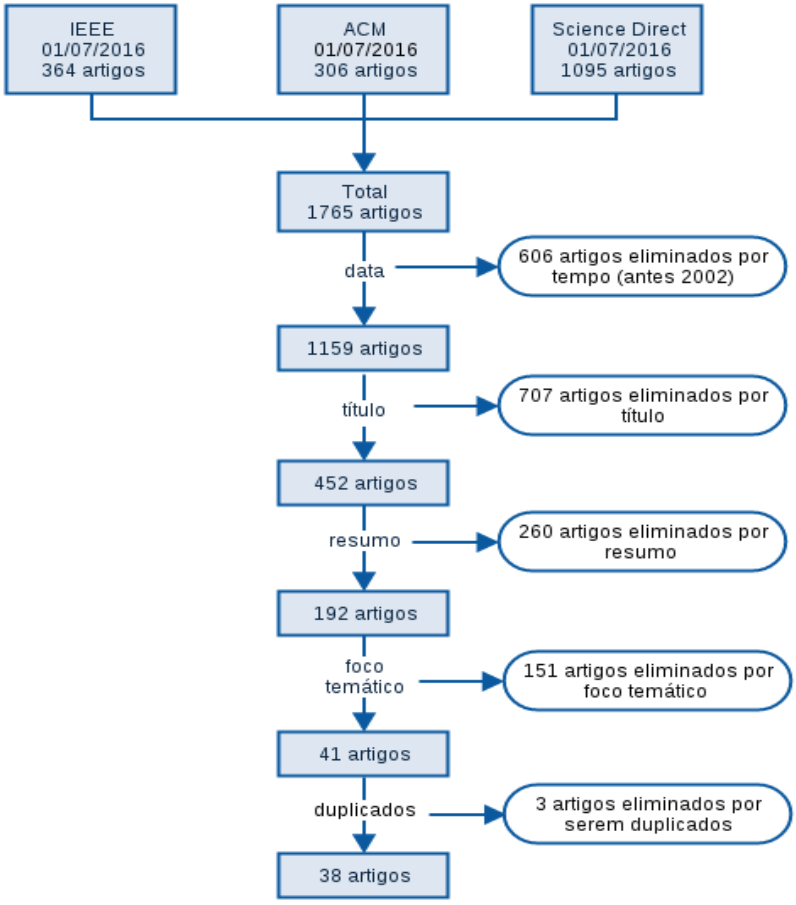

Figura 6 – Diagrama PRISMA.

Embora a revisão sistemática de literatura seja feita na prática por mais de uma pessoa para minimizar a subjetividade, o fato de ter sido feito neste trabalho de maneira solo não invalida o método pois, além de oferecer um bom senso de orientação de como procurar e encontrar os artigos, ela garante a não-tendenciosidade da busca em relação à pergunta formulada.

O Quadro 1 mostra a classificação dos conceitos dos artigos selecionados para a revisão de literatura, sendo evidenciado o método de pesquisa, a dimensão epistemológica e dimensão ontológica, abordagem de desenvolvimento de software, o espaço de trabalho e a cultura organizacional. A Figura 7 mostra os resultados da tabela em um mapa conceitual simplificado.

Quadro 1 - Classificação simplificada dos conceitos-chaves

Autor	Ano	Método de Pesquisa		Gestão de conhecimento no desenvolvimento de software										
				Dimensão Epistemol.		Dimensão Ontológica			Abordagem		Espaço		Cultura	
		Quantitativo	Qualitativo	Explícito	Tácito	Individual	Grupo de desenv.	Organizacional	Taylorista	Ágil	Local	Distribuído	Facilitador	Barreiras
Rus e Lindvall	2002		X	X	X	X	X	X					X	X
Ramesh	2002		X	X	X		X							
Chau, Maurer e Melnik	2003		X	X	X				X	X				
Melnik e Maurer	2004		X		X		X			X	X			
Ye, Yamamoto e Kishida	2004			X	X		X		X	X		X		
Ye	2005		X	X	X		X					X		
Bahli	2005		X	X	X		X		X	X				
Feng	2006			X				X		X		X	X	
Ye	2006		X	X	X			X				X		
Kokkoniemi	2008			X	X		X			X				
Yanyan e Renzuo	2008		X		X	X								
Boden e Avram	2009		X		X			X		X		X	X	
Chen, Shie e Liang	2009	X			X		X						X	
Dakhli e Chouikha	2009			X	X			X	X	X				
Dingsøyr, Bjørnson e Shull	2009		X	X	X				X	X				
Fægri	2009		X		X		X			X	X			
Levy e Hazzan	2009		X		X		X			X	X			X
Biao-Wen	2010			X	X			X						X
Faegri, Dyba e Dingsøyr	2010		X		X		X			X			X	
Huang, Shih e Hsu	2010	X	X		X		X						X	
Kavitha e Ahmed	2011			X	X		X			X	X	X		
Abdullah e Talib	2011		X	X			X		X					
Corrigan	2012				X	X								
Dorairaj, Noble e Malik	2012		X	X	X		X			X		X		
Eloranta e Koskimies	2012			X	X		X			X	X			
Nidhra, Yanamadala, Afzal e Torkar	2013		X	X	X		X					X	X	
Ryan e O'Connor	2013	X			X		X			X	X			
Santos, Goldman e Hernesto Filho	2014	X		X	X		X	X		X				
Zieris e Prechelt	2014		X	X	X	X	X			X	X			
Flores, Aguiar e Sereno	2014		X	X	X	X	X			X	X		X	
Plonka, Sharp, Linden e Dittrich	2014		X	X	X	X	X			X	X		X	
Menolli, Cunha, Reinehr e Malucelli	2014	X		X			X			X	X	X		
MacLeod, Storey e Bergen	2015		X	X	X		X						X	
Rabelo, Oliveira e Viana	2015		X	X	X		X			X				
Vasanthapriyan, Tian e Xiang	2015		X	X			X		X	X			X	X
Viana, Conte e Marczak	2015		X	X	X		X	X		X				
Athukorala, Perera e Meedeniya	2016	X		X	X	X	X						X	
Zieris e Prechelt	2016		X	X	X	X	X			X	X			

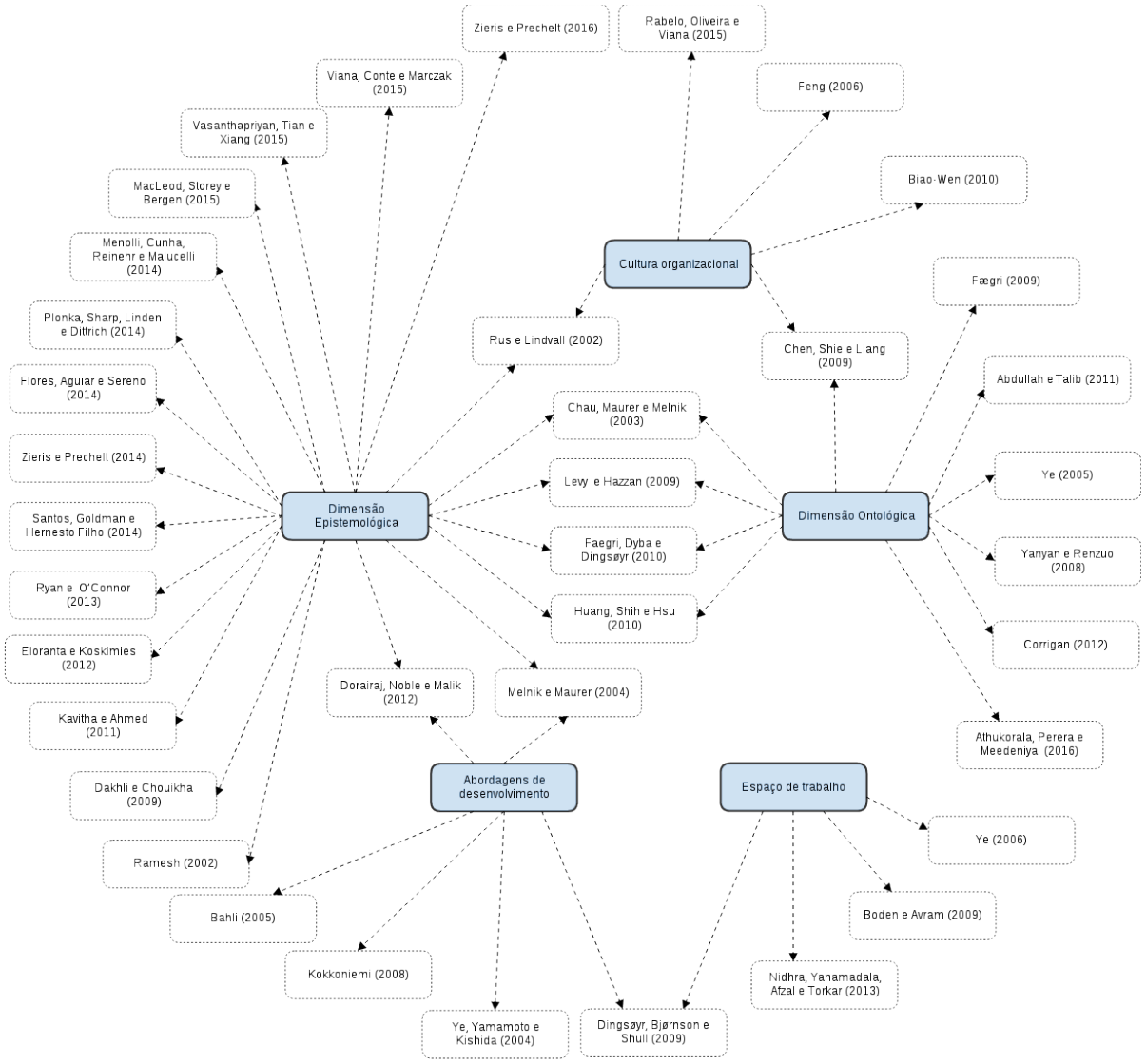

Figura 7 – Mapa conceitual simplificado

3.2 Estado da Arte

Esta seção apresenta o Estado da Arte, com os artigos classificados de acordo com a dimensão epistemológica, dimensão ontológica, abordagens de desenvolvimento de software, espaço de trabalho e cultura organizacional.

3.2.1 Dimensão Epistemológica

Parece ser consenso que o desenvolvimento de software é um desenvolvimento intensivo de conhecimento e que aqueles que trabalham com seu desenvolvimento são trabalhadores do conhecimento. Para Rus e Lindvall (2002), o principal ativo de uma organização de software é seu capital intelectual e o capital intelectual de uma organização consiste em ativos tangíveis e intangíveis. Os ativos tangíveis correspondem ao conhecimento explícito e os ativos intangíveis ao conhecimento tácito.

Mingxi (apud Biao-Wen, 2010) tem raciocínio parecido, porém enfatiza o conhecimento tácito e ressalta que o maior patrimônio das empresas de software são os talentosos funcionários de desenvolvimento de software e o conhecimento tácito em sua mente é o mais importante. Existem inúmeras formas de gerir o conhecimento tácito e o conhecimento explícito nas empresas de software. Para Kavitha e Ahmed (2011) o conhecimento de software está armazenado também em documentos, processos, práticas, tecnologias e artefatos do projeto. O conhecimento explícito é fácil de classificar e armazenar pela utilização de dados e da tecnologia, mas o conhecimento tácito é difícil de ser entendido pelos outros. As empresas de software têm uma proporção maior de conhecimento tácito do que as outras. Assim, a gestão desse conhecimento tácito é obviamente mais importante e urgente. Segundo Kavitha e Ahmed (2011), verifica-se que cerca de 40% do conhecimento organizacional é armazenado nas cabeças dos funcionários.

A espiral do conhecimento inicia-se por meio do conhecimento tácito dos indivíduos e é compartilhado durante a socialização. Segundo Ryan e O'Connor (2013), o conhecimento tácito é adquirido e compartilhado diretamente pela boa qualidade de interações sociais. Fahy e Prusak (1998 apud Levy e Hazzan, 2009) afirmam que o conhecimento tácito é composto de perspectivas, percepções, crenças e valores, e tem um papel central na captação e assimilação do conhecimento explícito.

Dingsøyr, Bjørnson e Shull (2009) afirmam que as empresas devem buscar soluções que enfatizam tanto o conhecimento tácito quanto o conhecimento explícito ao planejarem iniciativas de gestão do conhecimento para apoiar a engenharia de software, mas os gestores tendem a valorizar o conhecimento explícito mais do que o conhecimento tácito. Os autores Levy e Hazzan (2009) acreditam que o principal desafio da gestão do conhecimento é a transferência de conhecimento tácito em conhecimento explícito, bem como a transferência de conheci-

mento explícito dos indivíduos aos grupos dentro da organização e ressaltam a importância de se criar uma rotina sistemática para a captura de conhecimento tácito pois sem isso uma empresa não pode se beneficiar de seu conhecimento tácito acumulado. Kavitha e Ahmed (2011) propõem um *framework* para captura do conhecimento tácito em equipes ágeis de desenvolvimento de software.

Ramesh (2002) afirma que a disponibilidade de conhecimento não é suficiente para a transferência de conhecimento. Outros fatores, como a capacidade de absorção do receptor e o valor percebido do conhecimento são fundamentais para o sucesso da transferência de conhecimento.

Dakhli e Chouikha (2009) explicam a crise do software em termos de lacuna do conhecimento e enfatizam as relações entre o conhecimento tácito de propriedade dos atores organizacionais, processos organizacionais e arquitetura de sistema de informação. Para os autores a lacuna entre o conhecimento de domínio dos usuários dos sistemas (clientes) e dos desenvolvedores é uma das principais razões da crise do software.

Uma maneira eficaz de compartilhar o conhecimento tácito no desenvolvimento ágil de software é a programação em par, bem como a rotação dos pares. A programação em par é uma técnica proveniente da metodologia XP que consiste em dois programadores atuarem junto no mesmo problema ou atividade, permitindo o compartilhamento do conhecimento tácito dos indivíduos que compõem o par. Por meio da rotação de pares o conhecimento tácito é difundido e compartilhado por toda a equipe de desenvolvimento. Para Kavitha e Ahmed (2011) esta técnica permite que o conhecimento tácito possa propagar-se de forma mais eficaz pela comunicação face a face do que pela documentação, bancos de dados ou outros meios. Benedicenti e Paranjape (2001 apud CHAU; MAURER; MELNIK, 2003), de maneira similar, afirmam que essa técnica promove uma cultura de partilha de conhecimentos e facilita o compartilhamento de conhecimento tácito, como por exemplo o conhecimento do sistema, convenções de práticas de codificação e *design* e truques de uso de ferramentas. Desenvolvedores tendem a não documentar esse conhecimento e normalmente não é ensinado por meio de treinamento formal.

Além disso a programação em par e a rotação de pares introduzem a redundância de informação e conhecimento para o problema em questão. Segundo Fægri, Dyba e Dingsøyr (2010) a rotação de trabalho pode contribuir para a melhoria da redundância de conhecimento e seus

benefícios incluem a inovação decorrente da integração de diferentes áreas do conhecimento e melhor apreciação das questões organizacionais. Huang, Shih e Hsu (2010) concordam que a redundância de informações e concorrência interna podem acelerar a atualização tecnológica e difusão do conhecimento.

Plonka et al. (2014) procuraram entender, por meio de uma análise qualitativa, como se dá a transferência de conhecimento do condutor para o navegador durante a programação em par e quais desafios encontram quando o conhecimento da dupla não é equalizado. Eles identificaram um conjunto de estratégias de ensino e comportamentos que estão relacionados com as funções do condutor e navegador e a influência do ensino e aprendizagem, associado aos desafios e benefícios para ambos os parceiros. Para os autores a transferência de conhecimento acontece de forma fluída e natural na programação em par.

Zieris e Prechelt (2014) analisaram a transferência de conhecimento durante sessões de programação em par. por meio de uma análise qualitativa dos dados. Segundo os autores alguns pares são muito mais eficientes na transferência de conhecimento do que outros, pelo fato de utilizarem três principais mecanismos: (1) não tentarem explicar várias coisas ao mesmo tempo, (2) não perderem de vista um tópico que ainda não foi totalmente resolvido, e (3) esclarecerem os pontos difíceis.

Zieris e Prechelt (2016) definem dois tipos de conhecimento em seu estudo: conhecimento de longa duração (conhecimento permanente como linguagem de programação, por exemplo) e conhecimento de curta duração (referente ao domínio do projeto). Para alcançar a difusão do conhecimento pela programação do par, o conhecimento de longa duração deve ser transferido, e para uma sessão de programação em par ser verdadeiramente colaborativa, o conhecimento adicional de curta duração deve ser transferido entre a dupla.

Outra forma eficaz de compartilhamento do conhecimento tácito é a interação face a face, comumente utilizadas nas reuniões diárias do Scrum e do XP. Tsuchiya (1993 apud RYAN; O'CONNOR, 2013) afirma que a conversa face a face é mais adequado para transmitir conhecimento tácito, porque ele pode usar uma variedade muito maior de metáforas que conversa por meio da tecnologia da informação. Melnik e Maurer (2004) acreditam que canais de interação face a face oferecem a perspectiva de uma comunicação mais rica por causa da capacidade de transmitir múltiplos sinais, por exemplo, presença física, inflexão de voz e linguagem

corporal. Ainda segundo Melnik e Maurer (2004), a prática do Scrum de reuniões diárias e reuniões pós-*sprints* garantem um trabalho colaborativo em equipe e compartilhamento no decorrer do projeto.

Eloranta e Koskimies (2012) explicam que no Scrum, depois de cada sprint, há uma reunião de avaliação, onde os resultados são mostrados como uma *demo* para o Dono do Produto ou cliente. Além disso, uma sessão retrospectiva é realizada, onde as lições aprendidas com o Sprint são discutidos e melhorias de processo são planejadas. Após a reunião de avaliação, a equipe tem uma reunião de planejamento para o próximo sprint, no qual novamente são divididas as tarefas listadas no *backlog*. Essa prática permite, principalmente, a socialização do conhecimento.

Viana, Conte e Marczak (2015) procuram entender como o conhecimento é criado e perdido nas organizações de software. Eles entendem que as organizações de software praticam um conjunto de atividades que apoiam essa criação de conhecimento, porém a falta de documentação impede a retenção de um conhecimento preciso durante o desenvolvimento de software.

Para Rus e Lindvall (2002) um projeto de desenvolvimento de software envolve uma variedade de documentos orientados a processos e atividades. O trabalho centra-se na frequência de criação, revisão, edição e uso desses documentos, que se tornam ativos da organização na captura de conhecimento explícito. Portanto, a gestão de documentos é uma atividade básica para apoiar a implementação de uma organização de um sistema de gestão do conhecimento. Dorairaj, Noble e Malik (2012) apontam que a codificação do conhecimento traduz o conhecimento tácito em conhecimento explícito que são armazenados como artigos da Wiki, documentos de projeto e materiais de apresentação em sistemas de gestão do conhecimento. MacLeod, Storey e Bergen (2015) analisaram o compartilhamento de conhecimento por meio de vídeos e concluíram que a utilização de vídeos pode ser uma alternativa eficiente e útil para as documentações.

Santos et al. (2014) analisaram as características de estratégias organizacionais e fluxo de comunicação e canais de comunicação na eficácia do compartilhamento entre equipes. Eles puderam observar que empresas ágeis promovem a interação entre as equipes por meio de

conversas face a face, reuniões coletivas e uso de ferramentas comuns como Wikis, listas de e-mails e intranet.

Menolli et al. (2014) identificaram as principais ferramentas e tecnologias utilizadas por empresas de desenvolvimento de software no Brasil para gerenciar o conhecimento externalizado. Os autores destacam o uso de repositórios de conhecimento compartilhado, Wikis e ferramentas de propriedades.

A Wiki tem recebido uma atenção especial dos estudiosos pois além de codificar o conhecimento de maneira explícita permite a colaboração colaborativa. Chau, Maurer e Melnik (2003) afirmam que ferramentas como a Wiki permitem o compartilhamento fácil e eficaz de conhecimento explícito. Esta ferramenta se apresenta como um modelo para a gestão do conhecimento de alto impacto, pois oferecem oportunidades para a criação colaborativa de conhecimento na equipe e facilita a aprendizagem individual. Na mesma linha, Kavitha e Ahmed (2011) explicam que a tecnologia Wiki permite que qualquer usuário possa acessar, criar, organizar e atualizar as páginas da web em tempo real usando apenas um navegador web. Esta tecnologia suporta a colaboração assíncrona e permite aos usuários capturar informações em um formato estruturado como texto e gráficos.

Kavitha e Ahmed (2011) ressaltam que o armazenamento de informações não garante que outras pessoas possam encontrá-las. Os membros devem ser capazes de recuperar com êxito as informações de um repositório que combina dados estruturados e não estruturados. Mecanismos eficientes de consulta podem ser úteis para facilitar a recuperação de dados necessários.

Flores, Aguiar e Sereno (2014) apresentam uma visão compartilhada e um conjunto de requisitos dos quais as ferramentas devem cobrir de forma que o conceito de “Ba” possa ser mais bem apoiada no desenvolvimento de software e, portanto, para ajudar a evolução do conhecimento ao longo do ciclo de vida do software. Os autores concluem que o conceito de “Ba” aplicado ao desenvolvimento de software pode ajudar a detectar pontos de ineficiência em termos de evolução do conhecimento de software e que ainda há espaço para melhorar o seu apoio por meio de ferramentas adequadas.

Vasanthapriyan, Tian e Xiang (2015) fizeram uma revisão de literatura no tema de gestão do conhecimento no desenvolvimento de software e constataram que as ferramentas, técnicas

e metodologias atualmente empregadas no desenvolvimento de software são inadequadas para uma gestão do conhecimento eficaz.

3.2.2 Dimensão Ontológica

O fator humano no desenvolvimento de software dever ser levado em conta. Para Yanyan e Renzuo (2008) se um projeto de desenvolvimento de software é bem-sucedido ou não depende inteiramente do fator humano; assim, eles analisam a influência de fatores humanos no desenvolvimento de software, numa análise do individuo. Para os autores fatores humanos não podem ficar de fora de qualquer análise que englobe o ser humano.

Corrigan (2012) acredita que algumas técnicas contemplativas podem melhorar o fator humano no desenvolvimento de software sob as perspectivas da flexibilidade, atenção, criatividade e a confiança em si mesmo e entre os membros da equipe. O autor ressalta que além dos estudos sobre o processo e os fatores ambientais devemos também dar a devida atenção às técnicas que promovem o aumento da capacidade intelectual dos engenheiros de software e trabalhadores do conhecimento por meio de técnicas contemplativas como a “atenção plena” (*mindfulness*) e outras práticas do Yoga Nidra (CORRIGAN, 2012). Isso permite um aumento na capacidade de prestar atenção e se concentrar nas atividades, um aumento da flexibilidade e abertura às mudanças, um aumento na criatividade e na confiança no seu próprio trabalho e na confiança dos membros da equipe entre si. Ainda segundo Corrigan (2012), o aumento dessas qualidades entre os membros de uma equipe reduz o estresse e os conflitos que frequentemente podem surgir em ambientes de alta pressão como encontrado na área de engenharia de software e essa redução pode, naturalmente, promover o crescimento da confiança entre os membros da equipe.

Athukorala, Perera e Meedeniya (2016) procuraram a relação entre dois estilos de liderança (transformacional e transacional) com a criação de conhecimento e descobriram que a liderança do tipo transformacional tem uma correlação positiva com a criação de conhecimento, não havendo um efeito moderador da cultura organizacional na relação analisada.

Huang, Shih e Hsu (2010) afirmam que para acelerar a eficiência de aprendizagem em grupo é muito mais complicado do que a prática individual. Segundo Prusak e Lesser (1999

apud CHAU; MAURER; MELNIK, 2003), a aprendizagem ou a internalização do conhecimento explícito é um processo social. Não se aprende sozinho, mas aprende principalmente por meio do conhecimento tácito adquirido a partir de interações com os outros indivíduos. Além disso, como o desenvolvimento de software é um processo totalmente social, é importante para desenvolver a confiança organizacional e individual nas equipes e também entre as equipes e os clientes.

Conradi et al. (2002 apud ABDULLAH; TALIB, 2012) afirmam que o trabalho de software, como outros trabalhos de *design*, não é como a produção mecanizada ou disciplinada. Ele tem uma forte componente criativa que envolve a interação humana e social que não pode ser totalmente pré-planejada em um modelo padronizado e detalhado de processo.

O ambiente de trabalho e interação social também têm grande importância. Levy e Hazzan (2009) salientam que, a partir da perspectiva da teoria dos jogos, a principal limitação é que as pessoas tendem a não colaborar em condições de incerteza, quando o comportamento colaborativo não é garantido, compartilhar o conhecimento é tempo e esforço consumido.

Alguns autores também apontam que, devido à complexidade dos software, os conhecimentos exigidos para o seu desenvolvimento deve ser feito de forma colaborativa. Segundo Ye (2005), a construção colaborativa de conhecimento não é uma simples soma do conhecimento de cada participante, mas um resultado mutuamente estimulante de troca, inserido na interação e na prática social. Para o autor, a colaboração do conhecimento é um esforço intelectual conjunto em que um trabalhador do conhecimento interage com ferramentas cognitivas e com os colegas, tornando-se assim uma atividade essencial na maioria do desenvolvimento de software de hoje, e que define a capacidade de saber e competências dos desenvolvedores de software.

Para Chau, Maurer e Melnik (2003) as equipes de desenvolvimento ágil reúnem indivíduos que executam todas as funções definidas. Rotações de funções e troca de papéis são comuns. Também é possível ter membros altamente especializados (por exemplo, os analistas de segurança e engenheiros de usabilidade) compartilhados entre várias equipes em uma organização. A rotação de trabalho é uma técnica importante também no sentido de difundir o conhecimento elevando o nível da dimensão ontológica. Fægri (2009) acredita que a rotação de trabalho é outra prática conhecida para construir conhecimento geral além do nível da equipe.

Segundo Fægri, Dyba e Dingsøyr (2010), a rotação de trabalho pode contribuir para a melhoria da redundância conhecimento. Benefícios da redundância de conhecimento incluem a inovação decorrente da integração de diferentes áreas do conhecimento e melhor apreciação das questões organizacionais. Chen, Shie e Liang (2009) analisaram a relação entre a diversidade de conhecimento e desempenho do grupo. Os resultados sugerem que quanto maior a diversidade de um grupo de conhecimento, melhor é capacidade de decisão.

Segundo Levy e Hazzan (2009), a "prática de equipe inteira" significa que a equipe de desenvolvimento (incluindo todos os detentores de função e o cliente) se comunica em uma interação face a face, tanto quanto possível. A prática de equipe inteira é aplicada de diversas formas. Primeiro, a equipe de desenvolvimento está co-localizada em um espaço de trabalho colaborativo, um espaço que apoia e facilita a comunicação. Em segundo lugar, todos os membros da equipe participam de todas as apresentações do produto para o cliente, ouvirem as necessidades dos clientes e são ativos no processo de planejamento real. Terceiro, os detentores de papéis, que tradicionalmente pertencem a equipes separadas (por exemplo, testadores e designers), são integrados na equipe de desenvolvimento.

3.2.3 Abordagens

Devido à complexidade dos projetos de software são necessários o compartilhamento e a colaboração dos conhecimentos, explícito e tácito. Podemos separar em duas abordagens as metodologias de desenvolvimento de software: (1) a abordagem tradicional, baseada em repositórios ou abordagem taylorista que enfatiza o conhecimento explícito e (2) a abordagem baseada em comunidade ou abordagens ágeis que enfatiza o conhecimento tácito. Segundo Rus e Lindvall (2002) e Ye, Yamamoto e Kishida (2004), a primeira procura manter todos os requisitos, ferramentas, ciclos de vidas e tarefas devidamente documentados. Nessa abordagem os usuários buscam encontrar o conhecimento aplicável aos seus problemas. Já a segunda parte do pressuposto que é impossível saber tudo que vai acontecer no desenvolvimento de um software e se baseia no empirismo e em respostas ágeis. Essa útima abordagem enfatiza o compartilhamento de conhecimento tácito e passa a ser necessário, além de *saber o quê*, saber *quem sabe o quê*.

Dingsøyr, Bjørnson e Shull (2009) estudaram as abordagens de outra perspectiva, mas explicam que as escolas tecnocratas estão intimamente relacionadas ao desenvolvimento de

software tradicional (taylorista), enquanto as escolas comportamentais estão mais relacionadas com a abordagem ágil.

Segundo Chau, Maurer e Melnik (2003), nos métodos tradicionais, basicamente, é utilizada a documentação para a captura de conhecimento adquirido nas atividades de um ciclo de vida de projetos de software. Eles garantem a conformidade de produtos e processos de planos anteriores, apoiando iniciativas de melhoria de qualidade e satisfazendo as normas legais. Nessa abordagem, como o modelo cascata e suas variantes, na maioria, se não todos, o conhecimento é externalizado e combinado em diversos documentos para assegurar todos os requisitos possíveis. O que caracteriza uma abordagem centrada em documentações. Os autores afirmam que a abordagem taylorista leva uma vantagem na externalização do conhecimento e assim reduz a probabilidade de perda de conhecimento no caso de saída de algum talento, por exemplo.

As metodologias tradicionais apresentam a característica de especialização dos indivíduos envolvidos no desenvolvimento de software. Para Melnik e Maurer (2004) a divisão do trabalho é muitas vezes rigorosamente aplicada e leva a especializações; dessa maneira, pessoas são consideradas facilmente substituíveis. Métodos tradicionais resultam em cadeias longas de transferência de conhecimento; além disso, a comunicação direta entre o cliente e o desenvolvedor não é estimulada.

Para Chau, Maurer e Melnik (2003) os métodos tradicionais diferem dos métodos ágeis, principalmente, no fato de que todos os requisitos são capturados antes de iniciar desenvolvimento. Segundo Ye (2006) o grande problema com a abordagem baseada em repositório é que não se pode capturar o conhecimento tácito e contextual. Para Bahli e Zeid (2005) os métodos ágeis de desenvolvimento surgiram para superar alguns processos e problemas relacionados com os modelos tradicionais.

Nas abordagens ágeis a produção de software se aproxima das técnicas de produção puxada (*lean)*. Para Lindvall et al. (2002 apud KOKKONIEMI, 2008), os princípios ágeis promovem a entrega de software em pequenos incrementos e iterações, o que permite verificações de progresso frequentes e oferece a oportunidade de refinar os objetivos do projeto, de acordo com os desejos do cliente. A autoadaptação também é promovida tanto no projeto quanto no processo. Os métodos ágeis de desenvolvimento são particularmente adequados para projetos

em que os requisitos não são claros e, provavelmente, haverá alterações. Uma definição de agilidade, segundo Highsmith (2004 apud KOKKONIEMI, 2008) é indicado como a habilidade de criar e responder a mudanças, a fim de lucrar em um ambiente de negócios turbulento. Essa é, provavelmente, a principal vantagem das abordagens ágeis sobre as abordagens tradicionais, de planos orientados.

Segundo Cockburn e Highsmith (2002 apud CHAU; MAURER; MELNIK, 2003) os métodos ágeis, em contraste com os métodos tayloristas, sugerem que a maioria das documentações possam ser substituídas por comunicações formais e informais entre membros da equipe interna e entre a equipe e os clientes com uma maior ênfase no conhecimento tácito, em vez de conhecimento explícito. Holz e Maurer (2002 apud NEVES et al. 2011) afirmam que um dos objetivos do desenvolvimento ágil de software é converter o conhecimento tácito em explícito, permitindo o compartilhamento do conhecimento entre os membros das equipes de desenvolvimento de software e dentro da organização.

Além disso, as equipes ágeis, segundo Dorairaj, Noble e Malik (2012), são equipes multifuncionais que promovem o compartilhamento de conhecimento específico do projeto por meio da frequente interação face a face, da comunicação eficaz e da colaboração do cliente. De maneira semelhante, Kavitha e Ahmed (2011) afirmam que as abordagens ágeis dependem da comunicação face a face para a transferência de conhecimento. Para Rus e Lindivall (2002) é atraente se basear em conhecimento tácito em vez de explícito porque relaxa a exigência de documentar exaustivamente o conhecimento. No entanto, embora essa abordagem use o conhecimento, ainda não resolve o problema de uma organização ser dependente de seus empregados e seus conhecimentos tácitos.

Chau, Maurer e Melnik (2003) apontam que, no que se refere ao treinamento e aprendizagem, os métodos tayloristas utilizam-se de treinamentos formais padronizados e ampla documentação. Já os métodos ágeis utilizam-se de práticas informais de programação, como por exemplo a programação em par, rotação de pares e as interações face a face das reuniões diárias e retrospectivas. Quanto à gestão de competências, embora os métodos tradicionais não exijam qualquer prática específica para lidar com esse problema, uma prática comum é a identificação de especialistas com base na autoria do documento. No caso da abordagem ágil, a gestão de competências se dá por meio das reuniões diárias. Quanto à composição das equipes,

na abordagem taylorista, os diferentes papéis são agrupados em funções bases, nas quais contêm os membros do mesmo papel. Em contraste, as equipes ágeis são equipes multifuncionais.

Apropriando-se do método dialético, se a abordagem baseada em repositório é a tese e a abordagem baseada em comunidade é a antítese, era de se esperar que fossem propostas algumas sínteses. Segundo Ye, Yamamoto e Kishida (2004) o conceito da comunidade dinâmica é uma tentativa de unir as duas abordagens de forma integrada. A comunidade dinâmica é um subgrupo de trabalhadores do conhecimento que forma uma rede de apoio a um determinado usuário e uma determinada tarefa. Para afirmar suas ideias, os autores explicam que três tipos de relações existem em um espaço de trabalho de conhecimento: a relação entre as pessoas, a relação entre pessoas e informações e, por fim, a relação entre informações. A principal diferença é que os repositórios de comunidades dinâmicas de conhecimento também guardam a relação entre informação e pessoas, bem como a relação social entre as pessoas.

3.2.4 Espaço de Trabalho

O estudo do espaço de trabalho e da cultura organizacional devem ser explorados em todas as organizações, o que inclui as organizações desenvolvedoras de software. Rus e Lindvall (2002) ressaltam que colaborar e compartilhar conhecimento deve ser independente do tempo e do espaço. O espaço de desenvolvimento de software, normalmente, é tratado como local ou distribuído.

Segundo Dingsøyr, Bjørnson e Shull (2009) a escola espacial se concentra na concepção de espaço de escritório para promover o compartilhamento de conhecimento. Quanto às equipes de desenvolvimento co-localizadas, Levy e Hazzan (2009) afirmam que as paredes do espaço de trabalho de desenvolvimento servem como um meio de comunicação, constituindo um espaço informativo e colaborativo. A informação afixada nas paredes inclui, entre informações adicionais relevantes, o status das tarefas pessoais que pertencem à iteração atual e as medidas tomadas.

Dingsøyr, Bjørnson e Shull (2009) afirmam que as equipes ágeis têm, frequentemente, implementadas boas práticas para gestão do conhecimento por meio de técnicas como retrospectivas, reuniões frequentes em equipes co-alocadas num mesmo espaço de trabalho, mas os métodos ágeis oferecem pouco suporte para gestão do conhecimento além do nível de equipe.

Em organizações de desenvolvimento de software, o cliente pode ter um papel fundamental se tiver no mesmo espaço de trabalho devido ao seu conhecimento de domínio do sistema, como ressaltam Beyer et al. (2004 apud DORAIRAJ; NOBLE; MALIK, 2012), clientes no local podem impulsionar o desenvolvimento de software, fornecendo continuamente compreensão correta e completa de suas necessidades e contribuindo efetivamente para o crescimento e utilização do domínio do conhecimento em causa. Para Dorairaj, Noble e Malik (2012), quando os clientes estão trabalhando no local com a equipe, a colaboração pode ser melhorada por meio da participação efetiva no planejamento de lançamento, reuniões diárias, reuniões de avaliação e reuniões retrospectivas.

Segundo Chau, Maurer e Melnik (2003) existem vários estudos procurando adaptar as metodologias ágeis para uso em ambientes distribuídos. Segundo os autores, pela externalização do conhecimento em forma explícita, os métodos tradicionais, que são centrados em documentos, auxiliam as equipes de software distribuídos para colaborar independente do tempo e espaço. Isso pode ser uma lição valiosa para as equipes ágeis.

Ye (2006) aponta que o desenvolvimento de software não está mais confinado a um desenvolvedor individual, mas deve-se confiar em uma cognição distribuída por estarmos em uma era de rede de informações e colaboração mediada por computador. Com a tendência atual de globalização, empresas de software estão cada vez mais em zonas distribuídas ao longo dos tempos, lugares e culturas diferentes. Ainda para Ye (2006, p. 16), numa perspectiva da atividade cognitiva:

1. Processos Cognitivos podem ser distribuídos entre os membros de um grupo social;
2. Processos Cognitivos podem ser distribuídos entre estrutura interna e externa (materiais ou ambientais); e
3. Processos Cognitivos podem ser distribuídos ao longo do tempo de tal modo que os produtos de eventos anteriores possam transformar a natureza dos eventos posteriores.

Para Boden e Avram (2009), em ambientes distribuídos, pode ser muito difícil lidar com as barreiras espaciais e temporais, as relações legais organizacionais, nacionais e a cultura. Os autores estudaram o papel das “pontes” para facilitar a colaboração e coordenação entre locais distribuídos e afirmam que essas pontes são facilitadas por pessoas que agem naturalmente para gerenciar e mediar a comunicação e preencher as lacunas estruturais nas redes sociais.

Nidhra et al. (2013) concluíram que a gestão eficaz de projetos e fatores pessoais, facilitadas por fatores tecnológicos, são cruciais para uma transferência bem-sucedida do conhecimento em projetos distribuídos globalmente. Os autores destacam as barreiras linguísticas e culturais que podem dificultar a transferência de conhecimento em equipes distribuídas. Outro ponto apontado pelos autores é a confiança. A confiança é vista como um capacitador essencial para transferência de conhecimento. Quando a fonte de conhecimento não é entendida como confiável, o conhecimento transferido é reduzido.

Dorairaj, Noble e Malik (2012) afirmam que o compartilhamento de conhecimentos é difícil para equipes ágeis distribuídas devido a barreiras espaciais, temporais e culturais, que afetam negativamente a interação face a face, comunicação e colaboração. A transferência de conhecimento nesse caso se dá por meio do uso eficaz da comunicação mediada pela tecnologia, tais como ferramentas de gestão do conhecimento, a Wiki e a videoconferência. Assim, a transferência de conhecimentos entre os membros da equipe distribuídos ocorre usando ferramentas adequadas, durante a reunião diária, oficinas de iniciação, visitas, programação em pares, rotação e sessões de discussão. Porém ainda para os autores a falta de sinais não-verbais, tais como expressão facial ou gestos na comunicação mediada pela tecnologia reduz a riqueza das informações trocadas entre os membros da equipe distribuídos. Portanto, membros de equipes distribuídas devem visitar outros membros em diferentes locais, quando necessário, para obter uma melhor compreensão de situações críticas por meio de interações face a face que oferecem rica comunicação e compartilhamento de conhecimentos eficazes.

3.2.5 Cultura Organizacional

A cultura pode ser uma barreira ou um facilitador da gestão do conhecimento. Rus e Lindvall (2002) afirmam que nas organizações que não promovem uma cultura de compartilhamento, os funcionários podem se sentir possessivos com seu conhecimento e não estarem dispostos a compartilhar. Os funcionários sabem que a organização os valoriza por causa de seus conhecimentos, temem que serão considerados redundantes e descartáveis, logo que o empregador tenha capturado o seu conhecimento. Os empregados podem não estar dispostos a compartilhar experiências negativas e as lições aprendidas com base em falhas por causa da sua conotação negativa. Assim, embora o propósito da gestão do conhecimento seja evitar erros semelhantes, os funcionários podem temer que tais informações possam ser utilizadas contra

eles. Outro obstáculo salientado pelos autores é a síndrome do “não foi inventado aqui” que os engenheiros de software relutam em reutilizar soluções de outras pessoas. Embora a mudança seja difícil, tais crenças devem ser revistas e substituídas por uma atitude positiva, que recompensem o compartilhamento do conhecimento a fim de gerar valor.

Chen, Shie e Liang (2009) afirmam que a diversidade de conhecimento ajudaria a equipe a lidar com a variação ambiental e ressaltam a importância de um ambiente com redundância de comunicação pois é importante para a difusão do conhecimento e o conhecimento convergente na organização de atividades de conhecimento. O ambiente com redundância pode não só manter a vantagem do conhecimento organizacional, mas também aumentar a velocidade de criação de conhecimento. Numa perspectiva de que a cultura pode ser um facilitador da gestão do conhecimento, Feng (2006) afirma que a cultura é o conhecimento, incluindo a hipótese, o conceito de valor, crenças e sistema de representação, que pode ser compartilhado dentro da organização entre os membros.

Segundo Biao-Wen (2010), a gestão do conhecimento mudou a estrutura organizacional das empresas de software, processos de negócios, métodos de gestão e, inevitavelmente, toca sobre os direitos e interesses de alguns gerentes e funcionários. Vendo a cultura como uma possível barreira, Biao-Wen (2010) ressalta que a gestão do conhecimento exige que os funcionários aprendam novos conhecimentos e novas habilidades, convertam posições já conhecidas e aumentem a intensidade do trabalho, o que faz com que os funcionários sintam seus interesses ameaçados. Segundo o autor, a gestão do conhecimento requer o estabelecimento de uma nova cultura correspondente das empresas de software, o que choca a formação de ideias ao longo do tempo e valores realizados pela equipe e faz com que eles se sintam perdidos. A formação da nova cultura é instável, não é completa e não é, muitas vezes, uma cultura de “resposta” a esse fenômeno.

Boehm et al. (2000 apud CHEN; SHIE; LIANG, 2009) acreditam que a cultura e a estabilidade pessoal afetam a eficiência do trabalho cooperativo porque a cooperação, o compartilhamento de informações e acumulação de experiências afetam a produtividade. Para Levy e Hazzan (2009), tanto a gestão do conhecimento quanto o desenvolvimento ágil de software são dois processos organizacionais que enfrentam barreiras comuns quando introduzidos e aplicados. Ambas as disciplinas lidam com a cultura organizacional e gestão da mudança e é melhor analisada e implementada quando feita de maneira simultânea.

Rabelo, Oliveira e Viana (2015) compararam o modelo SECI e a cultura organizacional, por meio do *Competing Values Framework* que procura identificar as mudanças na cultura organizacional. Segundo os autores a cultura organizacional é essencial para estimular as interações entre indivíduos e para facilitar o fluxo de conhecimento dentro da organização. Entretanto, os autores não encontraram evidências de uma relação direta entre o modelo SECI e o *framework* analisado.

3.3 Conclusões da Revisão de Literatura

Nesta seção são apresentadas as conclusões da revisão de literatura organizadas em: socialização, externalização, combinação, internalização, condições capacitadoras, abordagens de desenvolvimento de software, espaço de trabalho e cultura organizacional.

Socialização

Como visto, de acordo com a teoria de Nonaka e Takeuchi (1997), a criação de conhecimento inicia-se pela socialização, ou seja, a conversão de conhecimento tácito em conhecimento tácito. Esse modo de conversão de conhecimento foi, de certa forma, negligenciado pelas abordagens tayloristas e praticamente não encontramos elementos, técnicas e ferramentas nessas abordagens. Já nas abordagens ágeis, a programação em par se revelou extremamente útil e eficaz na socialização. Com esta técnica os programadores trocam o conhecimento tácito de cada indivíduo por meio da observação, imitação e prática, além de permitir a restruturação dos modelos mentais de cada indivíduo do par. A programação em par também introduz a redundância de informação e conhecimento trazendo diferentes perspectivas para o problema abordado o que pode acelerar o compartilhamento do conhecimento tácito. Com a rotação de pares e rotações de trabalho, nos quais os programadores se revezam com o suporte ao cliente, a espiral do conhecimento também transita nos diferentes níveis da dimensão ontológica.

Outras técnicas das metodologias ágeis que privilegiam a socialização do conhecimento são as reuniões diárias, as reuniões *pós-sprints* ou mesmo as reuniões *post-mortem*. As reuniões face a face permitem a transmissão de múltiplos sinais corporais o que propicia a conversão de conhecimento tácito em conhecimento tácito. Nessas reuniões, o compartilhamento do conhecimento tácito é favorecido principalmente na formação de modelos mentais dos indivíduos participantes.

De certa forma, as equipes distribuídas têm uma maior dificuldade para socializar o conhecimento tácito e praticamente não encontramos elementos dessa conversão nas equipes distribuídas. Em situações complexas, as equipes distribuídas no espaço devem se encontrar com o intuito de permitir a transmissão do conhecimento tácito em reuniões face a face que permitem a transmissão de múltiplos sinais e expressões corporais.

Externalização

A conversão do conhecimento tácito em conhecimento explícito, externalização, é o grande desafio da gestão do conhecimento. A preocupação com a documentação nas abordagens tayloristas incentiva a externalização, portanto o conhecimento tácito presente inicialmente na cabeça dos indivíduos devem ser externalizados para que se possa elaborar, por exemplo, os requisitos do sistema e dessa forma os conceitos, hipóteses ou modelos são criados.

Nas metodologias ágeis, as mesmas técnicas citadas para a socialização, programação em par, rotação de pares, rotação de trabalho e as reuniões também contribuem para a externalização. Quando o conhecimento tácito de alguma forma passa a ser articulado, a conversão de conhecimento tácito em conhecimento explícito está presente, e assim pode ser transmitida por meio de metáforas, analogias, conceitos, hipóteses ou modelos.

Como o conhecimento passa a ser articulado, essa conversão de conhecimento também é encontrada nas equipes distribuídas, o auxílio das tecnologias como videoconferência é de vital importância nessas equipes.

Combinação

A conversão de conhecimento explícito em conhecimento explícito, de certa forma é mais comum nas metodologias tayloristas. Após a articulação do conhecimento tácito, é possível combinar os elementos do conhecimento explícito em novos conhecimentos explícitos. Novamente a ênfase na documentação nas metodologias tradicionais permite a combinação de forma eficiente. Ao final de cada fase dos métodos prescritivos são geradas novas documentações combinadas dos conhecimentos adquiridos por meio de reuniões, conversas ao telefone ou redes de comunicação computadorizadas nesta etapa, combinados com os antigos documentos de versões anteriores.

Nas abordagens ágeis a combinação é, de certo modo, pouco utilizada, embora ela esteja presente. A ênfase no conhecimento tácito faz com que o conhecimento explícito seja combinado apenas por meio de reuniões, e-mails e conversas por telefone e as poucas documentações existentes.

A combinação certamente não é um problema para as equipes distribuídas, pois o conhecimento já está em sua forma mais bem definida, ou seja, explicitado. Nessa conversão de conhecimento as tecnologias disponíveis como e-mails, videoconferências, sistemas de informação, repositório Wikis e documentações podem ser compartilhadas e combinadas sem maiores problemas.

Internalização

Por fim, a conversão de conhecimento explícito em conhecimento tácito, permite que os indivíduos internalizem o conhecimento. Nos métodos tradicionais é comum o “aprender fazendo” seguindo manuais e técnicas estabelecidas pela organização. Embora, de certa forma, a internalização seja um processo natural, pois todos os indivíduos ao terem contato com novos conhecimentos explícitos acabam internalizando esse conhecimento e criando, modificando ou atualizando seus modelos mentais. Quando esses novos conhecimentos são aplicados na prática possibilitam modificações nos *know-how* técnico de cada um.

Nos métodos ágeis, novamente a técnica de programação em par se revela eficaz devido ao “aprender fazendo”. É comum nessa técnica combinar programadores *seniors* com programadores iniciantes, o que permite, principalmente aos programadores iniciantes internalizarem mais rapidamente o conhecimento explícito, tanto do outro programador, quanto de manuais e repositórios Wikis em conhecimento tácito.

Condições capacitadoras

Das cinco condições capacitadoras descritas por Nonaka e Takeuchi, apenas a redundância e a variedade de requisitos foram estudadas de forma aprofundada nos artigos selecionados para a revisão de literatura e, indiretamente a autonomia e a variedade de requisitos, no que se refere às equipes ágeis multifuncionais. Os estudiosos, de certa maneira, não se atentaram a essas condições, provavelmente porque estão mais preocupados em ver como a gestão do co-

nhecimento pode ser gerida nas empresas de desenvolvimento de software do que como essas condições podem ser utilizadas para a criação de conhecimento.

Abordagens

Quanto à capacidade de resposta às mudanças as equipes ágeis levam vantagem sobre as equipes tradicionais. Em sua essência, as metodologias ágeis privilegiam as respostas à mudança e partem do pressuposto de que é impossível saber todos os requisitos de antemão, além de apostarem em equipes auto-organizadas e multifuncionais. As equipes tradicionais, com sua ênfase em documentação e especialização do trabalho, sofrem com as mudanças que ocorrem no meio do desenvolvimento. Por outro lado, perder um funcionário de uma equipe taylorista não é tão sentido como em uma equipe ágil, justamente pelas documentações abrangentes.

Fica evidente que as duas abordagens têm suas relevâncias para a teoria da criação de conhecimento e podem mutualmente se completarem. Enquanto as abordagens tayloristas reduzem a probabilidade de perda de conhecimento e incentivam a externalização e combinação as abordagens ágeis incentivam a interação face a face e privilegiam a socialização e a internalização. Em uma síntese das abordagens os desenvolvedores têm uma maior probabilidade de achar as informações necessárias pois além de saber o que procurar e onde procurar, os desenvolvedores terão auxílio a quem procurar, bem como propiciam a espiral do conhecimento de maneira mais eficaz. Como afirma Pressman (2006), “há muito a ser ganho considerando o melhor de ambas as escolas, e quase nada a ser ganho denegrindo qualquer uma dessas abordagens”.

Espaço de trabalho

As equipes distribuídas têm recebido atenção especial no mundo globalizado de hoje, devido ao fato de poder se aproveitar de talentos muitas vezes distantes e também pelos custos. Entretanto, pode-se concluir que mesmo com o auxílio das tecnologias disponíveis como a videoconferência o desenvolvimento ágil de software é mais eficiente em equipes co-alocadas no mesmo espaço de trabalho. Isso se dá pelo poder de interação social face a face dos indivíduos num mesmo espaço de trabalho, além de poder utilizar até as paredes como ferramenta de trabalho. Já nas equipes tradicionais a distribuição da equipe não é um grande empecilho.

Cultura organizacional

Quanto à cultura organizacional, vale ressaltar a grande dificuldade de se conseguir implementar técnicas de gestão do conhecimento e de técnicas do desenvolvimento ágil pois em grandes mudanças de paradigmas as pessoas tendem a querer continuar na situação anterior, mais cômoda, seja pelo medo de aprender novas tecnologias e ferramentas seja pelo acréscimo de novos desafios. Em equipes auto-organizadas, os seus membros devem ser pró-ativos, ou seja, fazem sem precisar de uma ordem de um líder ou de um procedimento e privilegiam sempre a equipe.

Técnicas e fatores que promovem a criação de conhecimento

Após a revisão de literatura pôde-se perceber a necessidade de uma classificação das técnicas de desenvolvimento de software e dos fatores capacitadores de acordo com a teoria de criação de conhecimento.

Por meio da revisão de literatura e do referencial teórico foi possível identificar algumas técnicas provenientes principalmente das metodologias ágeis e alguns fatores capacitadores para a criação de conhecimento organizacional no desenvolvimento de software. De acordo com os materiais analisados foram feitas diferentes classificações que abordaram a dimensão epistemológica, a dimensão ontológica e o modelo SECI.

Com essa classificação foi possível identificar quais técnicas das metodologias de desenvolvimento de software desempenham um papel central e quais desempenham um papel secundário na criação de conhecimento.

As Reuniões (incluindo as reuniões diárias, as reuniões de *sprint*, o *Planning Poker* ou o Jogo do Planejamento) e a Programação em Par (incluindo a rotação de pares) desempenham um papel extremamente relevante em três modos de conversão de conhecimento e propiciam a disseminação de diversos fatores capacitadores, incluindo os cinco fatores propostos por Nonaka e Takeuchi (1997): intenção, autonomia, redundância, flutuação e caos criativo e variedade de requisitos.

A Figura 8 mostra as técnicas e os artefatos classificados de acordo com a dimensão epistemológica.

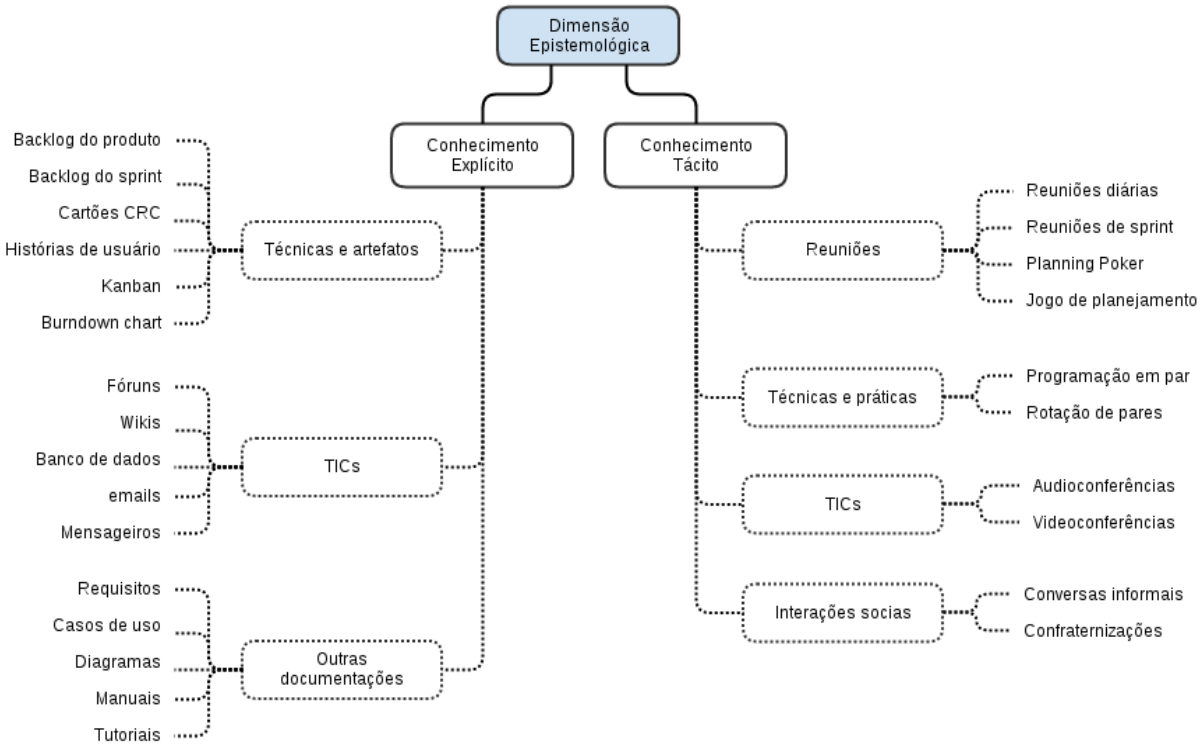

Figura 8 - Classificação das técnicas e dos artefatos do desenvolvimento ágil na dimensão epistemológica.

Já a Figura 9 apresenta os fatores capacitadores para a criação de conhecimento organizacional classificados de acordo com a dimensão ontológica.

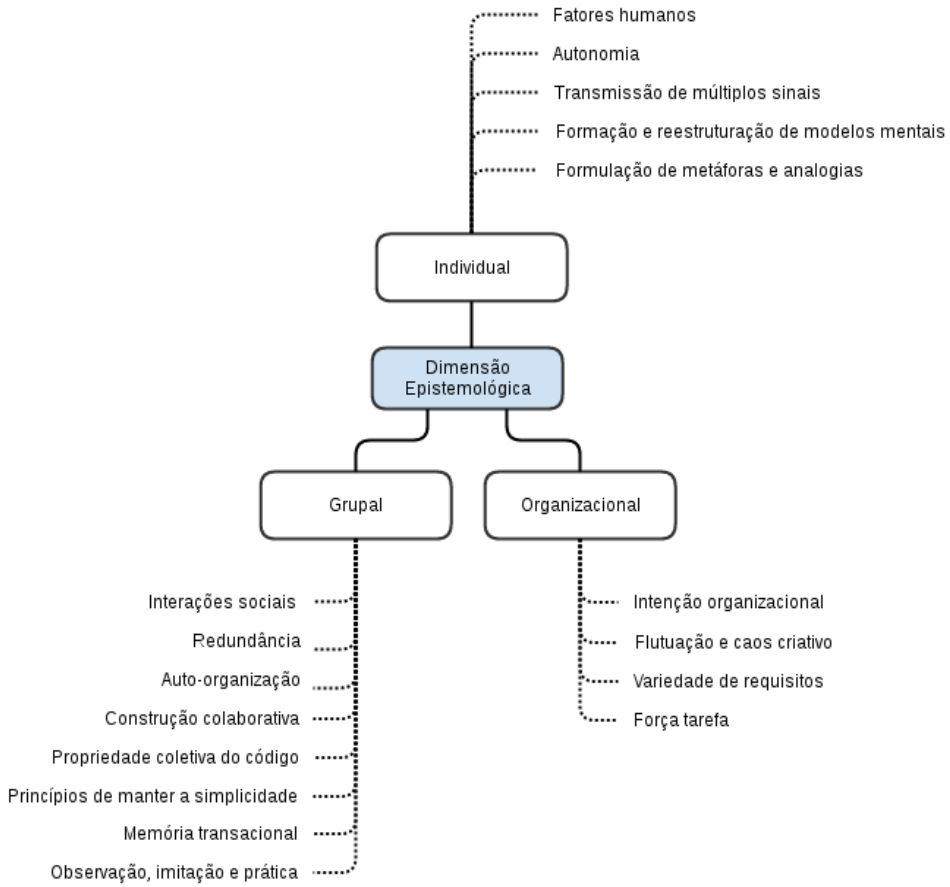

Figura 9 - Classificação dos fatores capacitadores na dimensão ontológica.

E, por fim, a Figura 10 sintetiza as principais técnicas, artefatos e fatores capacitadores classificados de acordo com o modelo SECI. Algumas técnicas favorecem mais de uma forma de conversão de conhecimento.

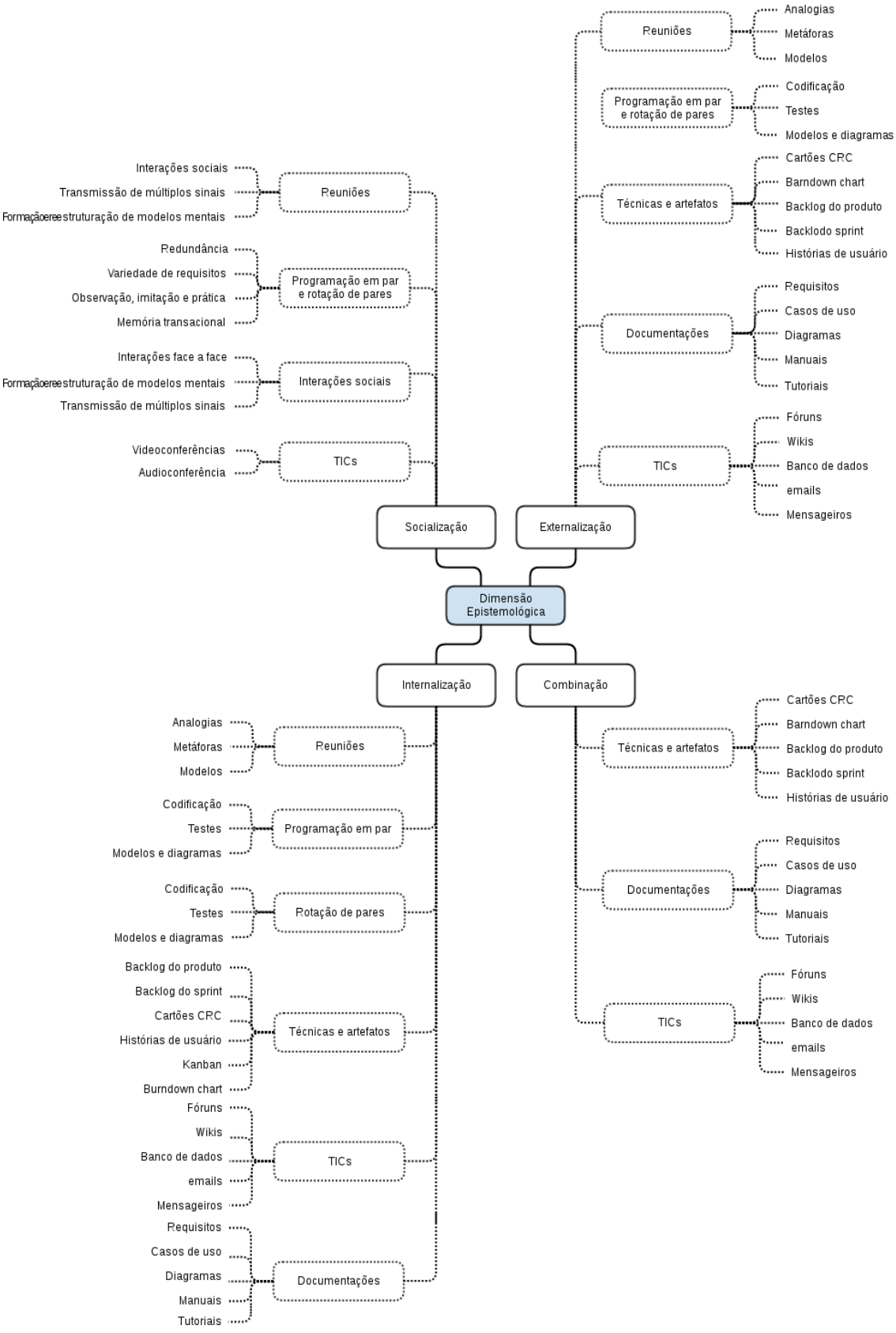

Figura 10 - Classificação das técnicas e dos artefatos do desenvolvimento ágil e dos fatores capacitadores no modelo SECI.

4 MÉTODO DE PESQUISA

A fim de conduzir e direcionar coerentemente o processo de pesquisa são adotados procedimentos metodológicos para coleta e análise dos dados. Assim, nesta seção são apresentados os parâmetros que guiaram a metodologia de pesquisa.

4.1 Caracterização da Pesquisa

A pesquisa teve caráter exploratório-descritivo, de concepção filosófica socioconstrutivista, por meio de uma abordagem de análise qualitativa. A adoção da análise qualitativa deve-se aos propósitos desta pesquisa, que procura compreender fenômenos de natureza social durante o processo de criação de conhecimento no desenvolvimento ágil de software. Esta pesquisa adotou uma investigação empírica realizada no local de ocorrência do fenômeno. Dessa forma teve como método de procedimento o estudo de casos. O Quadro 2 sintetiza o desenho de pesquisa.

Quadro 2 - Resumo do desenho de pesquisa

Desenho de pesquisa	
Caráter	Exploratório/descritivo
Concepção filosófica	Socioconstrutivista
Abordagem	Qualitativa
Método de procedimento	Estudo de caso
Método de coleta	Entrevistas com roteiro estruturado
Método de análise	Teoria Fundamentada em Dados

A pesquisa foi desenvolvida em duas organizações de base tecnológica que atuam no desenvolvimento de software e estão localizadas na cidade de Campinas, SP. Para a seleção das organizações considerou-se como critério sua adequação aos propósitos desta pesquisa, isto é, serem empresas de desenvolvimento de software que se utilizam de metodologias ágeis em seu desenvolvimento e que houvessem produzido produtos inovadores. Os sujeitos da pesquisa foram as equipes ágeis de desenvolvimento de software (*Scrum Master*, *Product Owner* e programadores) responsáveis pelo processo de desenvolvimento de novos software. A fim de preservar a identidade das empresas escolhidas, elas foram denominadas ao longo do trabalho de “Empresa A” e “Empresa B”.

4.2 Método de Coleta de Dados

Foram realizadas visitas técnicas às equipes de desenvolvimento e seus gestores e foi utilizado um roteiro (em forma de questionário) semiestruturado, para conduzir as entrevistas. Uma entrevista pode ser entendia como uma conversa guiada (LOFLAND; LOFLAND, 1984, 1995, apud, CHAMARZ, 2006). As questões foram elaboradas de acordo com o referencial teórico e a revisão de literatura, e os entrevistados puderam discursar livremente, com a menor interferência possível do entrevistador.

O roteiro em forma de questionário foi composto por questões abertas dirigidas aos membros das equipes de desenvolvimento de software nas duas empresas. Inicialmente foi feito um contato com as empresas por e-mail e após a sinalização positiva do aceite da participação no projeto de pesquisa foi marcado uma reunião preliminar na qual foi informado aos possíveis participantes o contexto da pesquisa, além de apresentar o documento referente ao Termo de Consentimento Livre e Esclarecido (TCLE). Após o entendimento e a aceitação do TCLE, foi agendado um horário, dentro do local e durante o horário de trabalho, com os participantes.

Optou-se por um roteiro semiestruturado para que tanto fosse possível pular algumas questões que não fossem pertinentes àquele grupo participante quanto buscar algum detalhe a mais nas respostas às perguntas mais pertinentes para aquela determinada equipe. A escolha dessa opção se deu com o intuito de agregar valor às respostas, visto que o método escolhido para a análise dos dados não parte de uma hipótese para depois tentar refutá-la ou confirmá-la. O método se baseia nos dados, na sua sistemática aquisição, para em seguida teorizar.

Vale mencionar que antes da aplicação das entrevistas para realização da coleta de dados, a fim de verificar a clareza das questões, foi realizado um teste com seis alunos de pós-graduação do Departamento de Engenharia de Computação e Automação (DCA) da Faculdade de Engenharia Elétrica e Computação (FEEC) da Universidade Estadual de Campinas (UNICAMP). Todos os alunos tiveram suas formações ligadas à carreira de informática o que possibilitou avaliar o entendimento das questões e sua adequação ao tema. O teste teve apenas o intuito de verificar a clareza das questões e a estrutura das perguntas e portanto não houve a preocupação com o conteúdo das respostas. O *feedback* dos pós-graduandos foi no sentido de simplificar as questões para que fossem de mais fácil compreensão. Exemplificando, uma questão na fase de teste:

> 16. A programação é feita em par? Qual o papel de cada um na programação? Eles trabalham 100% do tempo juntos?

Dessa maneira optou-se por separá-las em três questões diferentes, como segue:

> 24. A programação é feita em par? Se sim, responda 25 e 26.
> 25. Qual o papel de cada um na programação?
> 26. Eles trabalham 100% do tempo juntos?

As entrevistas com os membros da equipe de desenvolvimento da Empresa A foram feitas no dia 26 de junho e no dia 20 de agosto de 2015, no local e horário de trabalho. Já as entrevistas com os membros da equipe da Empresa B foram feitas no dia 24 de julho de 2015. É compreensível que a agenda de equipes multifuncionais sejam bem apertadas e foi um grande problema conseguir um horário na agenda dos participantes, mas, sem dúvida, o maior gargalo desse processo foi a burocracia dos trâmites com o comitê de ética.

A fim de uma análise mais refinada dos dados, todas as entrevistas foram gravadas em áudio, em formato digital m4a. Os áudios foram ouvidos e analisados apenas pelos pesquisadores (orientado e orientador). Seus conteúdos foram transcritos e salvos sem a identificação dos participantes, mantendo o compromisso assumido com o comitê de ética em pesquisa, aprovado sob o número CAAE 43519215.9.0000.5404.

O roteiro semiestruturado para entrevista contou com 50 questões divididas em quatro categorias: perfil do ambiente de estudo; condições capacitadoras para a criação do conhecimento; técnicas do desenvolvimento ágil que favorecem a criação de conhecimento; e processo de criação do conhecimento organizacional.

4.3 Método de Análise de Dados

O processo de desenvolvimento de software é uma atividade intensiva de conhecimento. Seu complexo processo visa integrar os conhecimentos das pessoas envolvidas com o desenvolvimento e estimular suas interações sociais. Nessa perspectiva, a metodologia foi delineada a fim de atender as necessidades, que surge da própria concepção do estudo, do conhecimento, seja ele individual ou coletivo, suas relações e conversões.

Dessa maneira, a investigação qualitativa pode contribuir já que é neste domínio que a pesquisa foi aplicada e desenvolvida. Assim, da perspectiva do interacionismo simbólico, para análise de dados foi utilizado a Teoria Fundamentada em Dados (TFD), inicialmente desenvolvida por Barney Glaser e Anselm Strauss, durante a década de 1960. Trata-se de um método sistemático de investigação, de natureza indutiva, que enfatiza a criação de novas teorias, derivada de uma análise sistemática e rigorosa dos dados (GLASER; STRAUSS, 1967). A TFD é adequada para áreas subexploradas ou nas quais uma nova perspectiva pode ser benéfica (SCHREIBER; STERN, 2001). Além disso, a TFD permite que pesquisadores estudem os comportamentos pessoais e as interações sociais (GLASER; STRAUSS, 1967).

O modo como os seres humanos interpretam e atribuem sentido às suas experiências na sua realidade subjetiva é o ponto central dessa perspectiva de investigação (HOLLOWAY, 1997), que busca deliberadamente descobrir novos conceitos a partir dos dados (REISSMAN, 1994). Por meio das formas discursivas, esse processo se torna possível pois resgata vivências contextuais que não só respondam, mas também desobstruam os fenômenos analisados.

As abordagens qualitativas permitem que as etapas de coleta e análise dos dados ocorram de forma simultânea. O pesquisador pode conduzir a pesquisa de acordo com suas próprias descobertas, pois a teoria evolui durante a investigação e isso ocorre por meio da relação dinâmica e contínua entre análise e coleta dos dados (STRAUSS; CORBIN, 1994). Além disso, a consistência de uma teoria construída a partir da TFD decorre da utilização de dados relevantes, que podem compreender anotações de campo, entrevistas, informações de gravações e relatórios (CHARMAZ, 2006). Para este trabalho, como mencionado na seção anterior, foram utilizadas entrevistas abertas conduzidas por meio de um roteiro semiestruturado que foram gravadas em áudio e cuidadosamente transcritas.

Após a coleta de dados, a TFD sugere o processo de codificação, no qual os dados são transformados em informações categorizadas, por meio de segmentos de dados, com um rótulo (uma denominação sucinta) capaz de sintetizar e representar cada parte dos dados (CHARMAZ, 2006). Na TFD a codificação é o elo crucial entre os dados coletados e o desenvolvimento de uma teoria emergente para explicar esses dados, pois é a partir dessa codificação que se conduz a interpretação analítica dos dados coletados (CHARMAZ, 2006).

A etapa de codificação pode compreender três níveis de codificação: codificação inicial (também chamada de codificação aberta ou codificação substantiva), codificação seletiva (também chamada codificação focada ou codificação de categorias; nesta etapa pode conter a codificação axial) e a codificação teórica (ou codificação de variáveis).

Segundo Charmaz (2006), a codificação inicial deve se ater estritamente aos dados, visualizando as ações em cada segmento de dados em vez de aplicar categorias preexistentes. Trata-se de um processo que demanda a denominação de palavra por palavra, linha por linha ou segmento por segmento. Este passo inicial na codificação conduz na direção de decisões posteriores sobre a definição das categorias conceituais fundamentais (CHARMAZ, 2006). Neste trabalho, optou-se por fazer a codificação inicial de segmento por segmento.

Codificação seletiva (ou focada) utiliza os códigos da fase inicial mais significativos e/ou frequentes por filtragem de grandes quantidades de dados. Codificação seletiva requer decisões sobre quais códigos iniciais fazem mais sentido analítico para categorizar seus dados completamente e de maneira incisiva (CHARMAZ, 2006). Já a codificação axial relaciona categorias com subcategorias, especifica propriedades e dimensões de uma categoria e reagrupa os dados segmentados durante a codificação inicial para dar coerência à análise emergente (CHARMAZ, 2006).

Por fim, como explica Charmaz (2006), a codificação teórica compreende o nível mais sofisticado de codificação. Nela se especificam as possíveis relações entre categorias desenvolvidas na fase de codificação seletiva, ou seja, apresenta um caráter integrativo. Eles dão formas aos códigos substantivos resultantes da etapa anterior e esses códigos podem ajudar a contar uma história analítica coerente. Assim, esses códigos não só conceituam como os códigos substantivos estão relacionados, mas também movem a história analítica em uma direção teórica (CHARMAZ, 2006).

Em seguida são feitas redações preliminares a partir das codificações visando o relacionamento dos códigos definidos entre os diferentes dados coletados, conhecida como redação de memorandos (*memo-writing*). A redação de memorandos constitui um método crucial na TFD já que leva a analisar os dados e códigos no início do processo de investigação (CHARMAZ, 2006). Por meio da redação de memorandos é possível construir notas analíticas para explicar e formar categorias. É nessa etapa que se faz comparações entre os dados, entre dados e códi-

gos, entre códigos e outros códigos, entre códigos e categorias, entre categorias e entre categorias e conceitos para a articulação de conjecturas sobre essas comparações ajudando a pensar sobre os dados e descobrir ideias sobre eles (CHARMAZ, 2006).

Na etapa seguinte, a etapa de integração, os memorandos são classificados, o que permite gerar a teoria que explica o fenômeno estudado. Charmaz (2006) chama de amostragem teórica e afirma que nesta etapa se buscam dados pertinentes para o desenvolvimento da teoria emergente. O principal objetivo da amostragem teórica é elaborar e refinar as categorias que constituem a teoria emergente. Deve-se conduzir amostragens teóricas para desenvolver as propriedades das categorias até que não haja novas propriedades emergentes (CHARMAZ, 2006).

Assim, o pesquisador consolida todos os esforços na escrita da teoria substantiva. Uma vez que as categorias emergem da codificação, deve-se juntar os modelos teóricos em torno de uma categoria central e refinar o modelo, iterativamente (CHARMAZ, 2006).

4.4 Análise de dados com a TFD

Conforme explicado anteriormente, a TFD sugere as seguintes etapas: (1) codificação substantiva; (2) codificação seletiva; (3) codificação axial; (4) codificação teórica; (5) redação de memorandos; (6) teoria substantiva. Esta seção ilustra os passos da TFD aplicada neste trabalho.

1. Codificação substantiva

   A primeira etapa de codificação consiste em dividir os dados em segmento por segmento e atribuir um código ou rótulo, para cada segmento. Neste trabalho os rótulos referem-se a técnicas e fatores (objetos de estudo deste trabalho). A resposta da questão 15, respondida pelo entrevistado 4, é utilizado para ilustrar o processo:

   > A decisão nunca é individual. Por mais que eu tenho um projeto que está na mão de uma pessoa e outro que está na mão de outro, eu chamo a equipe de TI inteira para discutir sobre aquele novo projeto de uma pessoa só. Todo mundo participa. Fazemos *brainstorming* sempre antes dos projetos. A autonomia está nisso, no próprio *brainstorming*, pós-*brainstorming* ou no refino *brainstorming*. As pessoas falam: isso está certo, isso está errado. No *brainstorming* todo mundo participa.

(Entrevista 4, questão 15)

Dividindo a resposta em segmentos:

> [A decisão nunca é individual.] [Por mais que eu tenho um projeto que está na mão de uma pessoa e outro que está na mão de outro, eu chamo a equipe de TI inteira para discutir sobre aquele novo projeto de uma pessoa só.] [Todo mundo participa.] [Fazemos *brainstorming* sempre antes dos projetos.] [A autonomia está nisso, no próprio *brainstorming*, pós-*brainstorming* ou no refino *brainstorming*.] [As pessoas falam: isso está certo, isso está errado.] [No *brainstorming* todo mundo participa.]
>
> (Entrevista 4, questão 15)

Os sete segmentos do trecho da entrevista foram separados e analisados em seu contexto para se atribuir, ao menos, um rótulo a cada segmento. O Quadro 3 ilustra essa atribuição dos códigos a cada segmento.

Quadro 3 - Exemplo de codificação substantiva.

1	A decisão nunca é individual.	➡	**Decisão coletiva**
2	Por mais que eu tenho um projeto que está na mão de uma pessoa e outro que está na mão de outro, eu chamo a equipe de TI inteira para discutir sobre aquele novo projeto de uma pessoa só.	➡ ➡	**Participação de todos** **Interação face a face**
3	Todo mundo participa.	➡	**Participação de todos**
4	Fazemos *brainstorming* sempre antes dos projetos.	➡	***Brainstorming***
5	A autonomia está nisso, no próprio *brainstorming*, pós-*brainstorming* ou no refino *brainstorming*.	➡ ➡	**Autonomia** ***Brainstorming***
6	As pessoas falam: isso está certo, isso está errado	➡	**Opinião**
7	No *brainstorming* todo mundo participa.	➡ ➡	***Brainstorming*** **Participação coletiva**

2. Codificação seletiva

A segunda etapa da metodologia adotada é a codificação seletiva. Nessa codificação são selecionados os códigos mais significativos por meio de filtragem. A filtragem procura descartar os rótulos que não são relevantes para o contexto. O Quadro 4 ilustra esse processo de seleção.

Quadro 4 - Exemplo de filtragem dos códigos.

1	A decisão nunca é individual.	➡	Decisão coletiva
2	Por mais que eu tenho um projeto que está na mão de uma pessoa e outro que está na mão de outro, eu chamo a equipe de TI inteira para discutir sobre aquele novo projeto de uma pessoa só.	➡ ➡	Participação de todos ~~Interação face a face~~
3	Todo mundo participa.	➡	Participação de todos
4	Fazemos *brainstorming* sempre antes dos projetos.	➡	*Brainstorming*
5	A autonomia está nisso, no próprio *brainstorming*, pós-*brainstorming* ou no refino *brainstorming*.	➡ ➡	Autonomia *Brainstorming*
~~6~~	~~As pessoas falam: isso está certo, isso está errado~~	➡	~~Opinião~~
7	No *brainstorming* todo mundo participa.	➡ ➡	*Brainstorming* Participação coletiva

O Quadro 5 resume os códigos filtrados para o exemplo ilustrativo.

Quadro 5 - Exemplo de codificação seletiva.

1	A decisão nunca é individual.	➡	Decisão coletiva
2	Por mais que eu tenho um projeto que está na mão de uma pessoa e outro que está na mão de outro, eu chamo a equipe de TI inteira para discutir sobre aquele novo projeto de uma pessoa só.	➡	Participação de todos
3	Todo mundo participa.	➡	Participação de todos
4	Fazemos *brainstorming* sempre antes dos projetos.	➡	*Brainstorming*
5	A autonomia está nisso, no próprio *brainstorming*, pós-*brainstorming* ou no refino *brainstorming*.	➡ ➡	Autonomia *Brainstorming*
7	No *brainstorming* todo mundo participa.	➡ ➡	*Brainstorming* Participação coletiva

3. Codificação axial

A codificação axial consiste em agrupar os códigos em categorias. O Quadro 6 mostra esse processo de classificação para o exemplo.

Quadro 6 - Exemplo de codificação axial.

1	A decisão nunca é individual.	➡	Decisão coletiva	➡	**Decisão coletiva**
2	Por mais que eu tenho um projeto que está na mão de uma pessoa e outro que está na mão de outro, eu chamo a equipe de TI inteira para discutir sobre aquele novo projeto de uma pessoa só.	➡	Participação de todos	➡	**Planejamento coletivo**
3	Todo mundo participa.	➡	Participação de todos	➡	**Planejamento coletivo**
4	Fazemos *brainstorming* sempre antes dos projetos.	➡	*Brainstorming*	➡	***Criatividade***
5	A autonomia está nisso, no próprio *brainstorming*, pós-*brainstorming* ou no refino *brainstorming*.	➡ ➡	Autonomia *Brainstorming*	➡ ➡	**Parte da equipe** **Criatividade**
7	No *brainstorming* todo mundo participa.	➡ ➡	*Brainstorming* Participação de todos	➡ ➡	**Criatividade** **Planejamento coletivo**

Nessa etapa, todos os códigos são agrupados em categorias e subcategorias. A Figura 11 ilustra essa classificação.

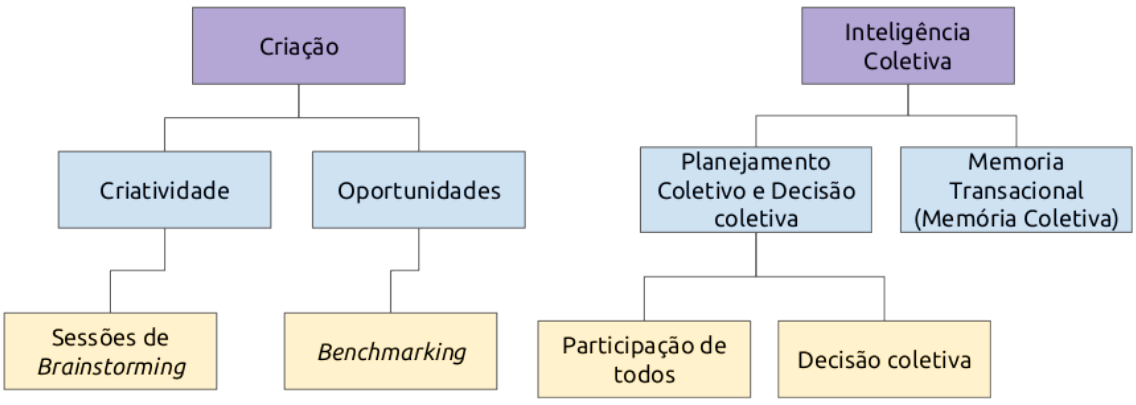

Figura 11 - Exemplo de classificação dos códigos em categorias e subcategorias.

4. Codificação teórica

Após a classificação de todos os códigos, incia-se a etapa de codificação teórica. Nela as atenções são voltadas a explorar a relação entre as categorias e subcategorias. A Figura 12 mostra a relação entre as categorias encontradas para esse estudo.

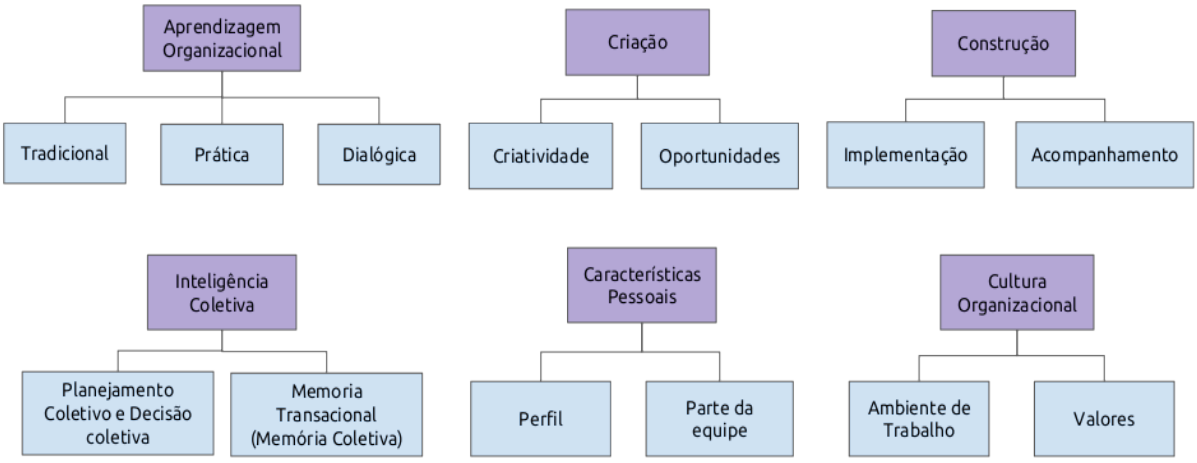

Figura 12 - Relação entre categorias e subcategorias

5. Redação de memorandos

A etapa de redação de memorandos descreve a relação entre as categorias, entre os rótulos e entre rótulos e categorias. O trecho a seguir ilustra o processo de redação de memorandos para o exemplo.

> Sessões de *brainstorming* são utilizadas para estimular a criatividade em reuniões de planejamento e também é utilizado para soluções específicas, em conjunto com esboços e desenhos.

Outro exemplo de redação de memorando:

> A programação em par atua tanto na implementação (construção) quanto na aprendizagem prática. Como duas pessoas diferentes não detêm exatamente o mesmo conhecimento, ao trabalhar em par há um processo de aprendizagem que busca equalizar o conhecimento da dupla.

6. Teoria substantiva

   A inter-relação de diversos memorandos compõem os fenômenos emergentes dos dados. Por fim, os fenômenos compõem a teoria substantiva emergida dos dados. Vale ressaltar que o processo é iterativo, incremental e exaustivo no qual se explora as relações de diversas formas e combinações na direção de construir uma teoria consistente com os dados encontrados.

4.5 Limitações da Pesquisa

Os dados da pesquisa foram obtidos por meio do estudo de caso de duas empresas de desenvolvimento de software na região de Campinas, SP. Seus grupos de desenvolvimento de software utilizam metodologias ágeis e gerenciam seus conhecimentos ativamente. Pelo fato da pesquisa inferir resultados referentes a apenas duas organizações de contextos específicos os dados não podem ser generalizados para todas equipes de desenvolvimento ágil de software.

Além disso, o método de análise de dados adotado, a TFD, visa gerar uma teoria substantiva. Conforme explicam Strauss e Corbin (1994) a teoria substantiva é uma teoria de escopo específico para um determinado fenômeno, grupo ou situação e não objetiva a generalização estatística. Ela busca uma generalização analítica e procura explicar o contexto dos sujeitos analisados em vez de uma verdade absoluta (GLASER; STRAUSS, 1967; STRAUSS; CORBIN, 1994).

Outro ponto a se destacar, diz respeito ao fato de que esta pesquisa apresenta um escopo amplo e interdisciplinar. Explora-se temas que compõem diversas áreas do saber tais como epistemologia, administração, ciências da informação, ciência da computação, ciências sociais, psicologia, engenharia de produção e engenharia de software. As informações abordadas não esgotam a totalidade dos respectivos conteúdos que envolvem cada uma dessas disciplinas. Conforme Nakayama (2009 apud PACHECO; TOSTA; FREIRE, 2010), e alinhado com a psicologia *Gestalt*[1]., esse espaço jamais poderá ser constituído pela simples adição de todas as especialidades nem tampouco por uma síntese de ordem filosófica dos saberes especializados.

[1] Dentre as diversas prerrogativas da psicologia *Gestalt*, neste contexto refere-se a "(…) "A+B" não é simplesmente "(A+B)", mas sim um terceiro elemento "C", que possui características próprias" (GUILLAUME, 1960).

O fundamento do espaço interdisciplinar deverá ser procurado na negação e na superação das fronteiras disciplinares.

A interdisciplinaridade se caracteriza pela intensidade das trocas entre os especialistas e pelo grau de integração real das disciplinas (PACHECO; TOSTA; FREIRE, 2010). Existe um consenso de que a interdisciplinaridade contribui significativamente para o avanço da ciência, principalmente no estudo de temas complexos, impossíveis de estudo disciplinar (PACHECO; TOSTA; FREIRE, 2010). Abordagens interdisciplinares procuram respeitar os paradigmas de diferentes disciplinas que esse intercâmbio de especialistas, advindos de diferentes disciplinas, constrói novas visões de mundo que retroalimentam paradigmas unitários, de maneira a fortalecer não somente o espaço em que elas conversam, mas principalmente fazendo evoluir suas próprias certezas paradigmáticas (PACHECO; TOSTA; FREIRE, 2010).

Gestão do conhecimento, desenvolvimento de software, aprendizagem e inovação são caracterizados por uma grande diversidade conceitual, relacionam-se a diversos atores sociais organizacionais e podem sofrer influências de variáveis estruturais, contextuais, sociais e individuais (ISIDRO FILHO; GUIMARÃES, 2010). Essas áreas apresentam similaridades em termos ontológicos e epistemológico, porém com grandes diferenças teórico-conceituais. Diversas disciplinas como psicologia, sociologia, ciência administrativa, engenharia de produção, engenharia de software, economia, antropologia cultural, entre outras, têm contribuído ativamente para a construção de um conceito interdisciplinar para a área (STEIL, 2002).

Segundo Isidro Filho e Guimarães (2010) essas abordagens não são excludentes, mas enfatizam aspectos diferentes desses fenômenos em contextos organizacionais. A abordagem cognitiva prioriza variáveis individuais como, por exemplo, tomada de decisão, solução de problemas, criatividade, informações, valores, entre outras, enquanto que as abordagens social e econômica ressaltam esses fenômenos como processos sociais de construção, transformação e aplicação de algo considerado relevante para o contexto ou mercado.

5 RESULTADOS E DISCUSSÕES

Na pesquisa exploratória da TFD, como dito anteriormente, a coleta e análise de dados são feitas de maneira simultânea. Dessa maneira os resultados são apresentados de forma organizada conforme a análise.

Os resultados foram agrupados de acordo com as informações que emergiram dos dados analisados e foram reunidos em seis grupos principais: aprendizagem, criação, construção, características pessoais, cultura organizacional e inteligência coletiva. Mais dois grupos secundários foram encontrados e foram distribuídos nos grupos principais: comunicação e ferramentas. As análises das interligações desses grupos possibilitaram a organização de diversos agrupamentos os quais resultaram em dois fenômenos:

Fenômeno 1

Quadro 7 - Fenômeno 1.

A equipe ágil estabelece ***técnicas*** que apoiam a **aprendizagem organizacional**, que ajudam na **criação** e na **construção** de software inovadores e que permitem uma eficiente **comunicação** entre todas as partes envolvidas fazendo uso de **ferramentas** adequadas.

Fenômeno 2

Quadro 8 - Fenômeno 2.

A equipe ágil é gerenciada mediante uma eficiente **comunicação** que privilegia a **inteligência coletiva** e promove ***fatores capacitadores*** que alinham a **cultura organizacional** e as **características pessoais** dos colaboradores apoiados por **ferramentas** adequadas.

O Fenômeno 1, enunciado no Quadro 7, mostra que a equipe ágil valoriza o conhecimento e procura aprender em todas etapas do processo de desenvolvimento de software. Também utiliza-se de técnicas que promovem a criatividade de seus membros e acompanhem o processo de construção do software. Valoriza-se uma comunicação direta e eficiente entre os membros do grupo, entre as diferentes áreas funcionais e está em diálogo contínuo com clientes e

parceiros. Por fim, apoia-se em ferramentas visuais e tecnológicas que permitem ao processo fluir com maior facilidade.

O Fenômeno 2, expresso no Quadro 8, sugere a importância de uma gestão que busque a decisão coletiva, utilize e valorize a inteligência de todo o grupo como equipe em busca de unir esse conhecimento, inicialmente distribuído na mente de cada um dos colaboradores durante esse processo de criação de novos software. Para tanto, é necessário o alinhamento de fatores pessoais com fatores culturais da organização e que resulte em um ambiente propício à aprendizagem, que seja criativo e produtivo.

5.1 Caracterização dos Participantes da Pesquisa

Por se tratar de dois estudos de caso que tiveram como objetivo a geração de uma teoria substantiva que correspondesse à realidade das equipes estudadas, não se pode generalizar a teoria para todas as equipes ágeis. Dessa forma, faz-se necessária a caracterização dos participantes entrevistados.

A Empresa A tem entre 50 e 60 funcionários e pode ser caracterizada como uma organização orientada a produtos e serviços. Apresenta uma subdivisão em áreas, sendo a equipe ágil de desenvolvimento de software pertencente à área de Pesquisa e Desenvolvimento.

A equipe ágil da Empresa A utiliza a metodologia kanban (*scrumban*) e é composta pelo *Product Owner*, o *Scrum Master* e os desenvolvedores. O *Product Owner* tem o cargo de gerente de tecnologia. O *Scrum Master* é o líder de desenvolvimento e os desenvolvedores são programadores.

Na Empresa B trabalham cerca de 40 funcionários e pode ser caracterizada como uma organização orientada a serviços. A equipe ágil estudada pertence a área de TI, mais especificamente, ao departamento de Pesquisa e Desenvolvimento.

A metologia utilizada pela equipe ágil da Empresa B é o scrum (*scrumban*) e fazem parte da equipe o *Scrum Master* e os desenvolvedores. Nesta equipe, o supervisor de TI que exerce a função de *Scrum Master* acumula a função de *Product Owner* em conjunto com outros departamentos da empresa.

5.2 Descrevendo o Fenômeno 1

O Fenômeno 1 observado tem como foco central as técnicas utilizadas pelos colaboradores envolvidos no desenvolvimento ágil de software. As técnicas foram separadas em três categorias centrais:

1. Técnicas que permitam a aprendizagem organizacional.
2. Técnicas que promovam a criação de software inovadores.
3. Técnicas que auxiliem na construção do produto de software.

A categoria Aprendizagem Organizacional apresenta três subcategorias: Aprendizagem Tradicional, Aprendizagem Prática e Aprendizagem Dialógica. A categoria central Criação agrupa duas subcategorias: Criatividade e Oportunidades. Já a categoria Construção reúne as subcategorias: Implementação e Acompanhamento. O Quadro 9 mostra a relação do fenômeno 1, com as categorias centrais e suas subcategorias.

Quadro 9 - Categorias e subcategorias do Fenômeno 1.

Fenômeno 1: A equipe ágil estabelece ***técnicas*** que apoiam a **aprendizagem organizacional**, que ajudam na **criação** e na **construção** de software inovadores e que permitem uma eficiente **comunicação** entre todas as partes envolvidas fazendo uso de **ferramentas** adequadas.	
Categorias Centrais	Subcategorias
Aprendizagem Organizacional	Aprendizagem tradicional
	Aprendizagem prática
	Aprendizagem dialógica
Criação	Criatividade
	Oportunidades
Construção	Implementação
	Acompanhamento

5.2.1 Categoria Aprendizagem Organizacional

Na categoria central Aprendizagem Organizacional foram reunidos os códigos referentes a internalização do conhecimento por meio de participações em eventos (internos e externos à

organização), apresentações e palestras agrupados sob a subcategoria Aprendizagem Tradicional, os códigos mentoria, programação em par, rotação de pares e rotação de trabalho referentes a subcategoria Aprendizagem Prática e os códigos referentes à importação de conhecimento do especialista de domínio por meio do diálogo contínuo com clientes, parceiros, consultores e diversas áreas funcionais que permitam a incorporação desse conhecimento foram agrupados na subcategoria Aprendizagem Dialógica.

5.2.1.1 Subcategoria Aprendizagem Tradicional

Embora as equipes ágeis valorizem mais o conhecimento tácito, a aprendizagem tradicional não foi deixada de lado pelas equipes entrevistadas. Neste trabalho a aprendizagem tradicional refere-se ao modelo clássico de ensino no qual o conhecimento já está na forma de conhecimento explícito e esse conhecimento é transmitido de maneira previamente estruturada como em treinamentos, aulas, palestras e eventos. Assim, a subcategoria aprendizagem tradicional reúne os códigos referentes a eventos externos dos quais as organizações participam e eventos internos que elas organizam.

Apesar de não fornecerem treinamento formal para seus funcionários (Entrevista 5, questão 34; Entrevista 2, questão 34), a participação em eventos de mercado são incentivadas em ambas as empresas que destacam a importância de se replicar esse conhecimento adquirido pelos participantes com os demais colaboradores da empresa por meio de eventos internos (Entrevista 1, questão 34; Entrevista 3, questão 34; Entrevista 4, questão 38).

Esses eventos internos não se limitam a replicar os eventos externos, mas também procuram fornecer produção de conteúdo por algum colaborador que detenha conhecimento de determinado assunto. Esse é o caso do evento chamado “*I Know How*” utilizado pela empresa A (Entrevista 3, questão 34).

Outro evento que visa a produção de conteúdo pela Empresa A é chamado de clube de autores, no qual um tema é definido pela equipe de marketing e um grupo de colaboradores se comprometem a produzir conteúdo para ser entregue ao mercado (Entrevista 1, questão 16).

A equipe ágil da Empresa B compartilha o conhecimento e incentiva a aprendizagem tradicional por meio do evento interno chamado de espaço nerd. O Espaço nerd vai além de sim-

plesmente compartilhar conhecimento, ele fomenta a autonomia dos funcionários (Entrevista 4, questão 16) e é valorizado pelos colaboradores (Entrevista 5, questão 33).

A aprendizagem tradicional está baseada no uso do conhecimento explícito, como ele é manipulado em livros, artigos ou em uma aula ou palestra (NONAKA; TAKEUCHI, 1997). Esse conhecimento já explicitado forma o conhecimento perfeito, maduro e bem formado, que pode ser facilmente articulado e transmitido (NONAKA; TAKEUCHI, 1997). Ele permite uma fácil internalização do conhecimento e aumenta a possibilidade de combinações do conhecimento explícito.

5.2.1.2 Subcategoria Aprendizagem Prática

Segundo Nonaka e Takeuchi (1997) a internalização do conhecimento está intimamente ligada ao “aprender fazendo”. Esse processo de imitação e prática que possibilita e capacita o “aprender fazendo” constitui uma forma de aprendizagem indispensável dentro da organização.

Neste trabalho a aprendizagem prática refere-se, principalmente, à aprendizagem que está intimamente ligada ao contato com suas funções profissionais. Assim, nessa subcategoria se encontram os códigos referentes à mentoria, programação em par e rotação de trabalhos.

Como uma espécie de mentoria, nas duas empresas estudadas, o líder (*Scrum Master*) ou aqueles que detenham um maior conhecimento de um determinado assunto acompanham os menos experientes que procuram extrair o máximo de conhecimento possível (Entrevista 2, questão 22; Entrevista 2, questão 34; Entrevista 5, questão 22; Entrevista 5, questão 34, Entrevista 4, questão 34).

Deve-se tomar cuidado para não confundir programação em par com mentoria. A mentoria visa somente o aprendizado e a programação em par está fundamentada no princípio da variedade de requisitos para a resolução de problemas. Porém, como duas pessoas diferentes não detêm exatamente o mesmo conhecimento, intrinsecamente ao trabalho em par, há um processo de aprendizagem que busca equalizar o conhecimento da dupla. A programação em par é utilizada pelas duas equipes em casos especiais, nos quais a complexidade das funcionalidades se sobressaem, mas também é utilizada como uma ferramenta de aprendizagem pelas equipes entrevistadas (Entrevista 3, questão 24; Entrevista 4, questão 24; Entrevista 5, questão 25).

A rotação de trabalho é uma técnica utilizada pela equipe ágil da Empresa B que além de promover a aprendizagem prática ela pode ter uma componente motivacional importante pois mexe com a zona de conforto dos colaboradores e dissemina o conhecimento por toda a organização (Entrevista 4, questão 24, Entrevista 4, questão 34).

A aprendizagem prática está intimamente ligada ao aprender fazendo. Esse tipo de aprendizagem privilegia o uso do conhecimento tácito, seja na sua dimensão técnica ou cognitiva e pode ser transmitido e aprendido por imitação e prática ou a reformulação dos modelos mentais, crenças e percepções (NONAKA; TAKEUCHI, 1997). Dessa forma esse tipo de aprendizagem é normalmente encontrado no processo de socialização e internalização do conhecimento.

5.2.1.3 Subcategoria Aprendizagem Dialógica

Aprendizagem dialógica na educação refere-se ao conhecimento que é co-construído entre os envolvidos em atividades realizadas em parceria, por meio do debate e do discurso lógico (DOTTA, 2009). O conceito de Comunidades de Aprendizagem sustenta-se na aprendizagem dialógica e refere-se a uma aprendizagem colaborativa, baseada no diálogo e no inter-relacionamento. Neste trabalho, a aprendizagem dialógica refere-se ao processo de aprendizagem por meio do diálogo e interação social da equipe ágil com as diferentes áreas funcionais ou com o ambiente (parceiros, clientes, concorrentes e sociedade).

A subcategoria aprendizagem dialógica reúne os códigos interação com o ambiente e interação com diferentes áreas funcionais que buscam a importação de conhecimento para ser incorporado ao produto de software.

A interação com o cliente é uma forma de interação com o ambiente e aparece, basicamente, em duas formas nas equipes entrevistadas. A primeira trata-se da interação para conhecer as necessidades dos clientes e assim propor soluções. A segunda refere-se a interação para se obter um *feedback*, uma resposta de algo entregue ao cliente e que necessita ser avaliado.

Da primeira, em busca de compreender as necessidades do cliente, a equipe da Empresa A, se necessário, interage com o cliente e outras áreas funcionais em busca de validar os principais conceitos (Entrevista 3, questão 43; Entrevista 3, questão 50). Embora toda a equipe ágil possa

se envolver com o cliente em busca de entender suas necessidades, ela é, em última instância, responsabilidade do *Product Owner* (Entrevista 1, questão 23, Entrevista 3, questão 23).

Com o intuito de se obter *feedback* do cliente, a equipe ágil da Empresa A, tanto pode fazer uma pré-validação das funcionalidades (Entrevista 3, questão 50), quanto validações definitivas pelo usuário final (Entrevista 1, questão 41; Entrevista 1, questão 50; Entrevista 3, questão 36), ou ainda as solicitações dos clientes podem ser feitas em tempo de implementação (Entrevista 2, questão 43).

A equipe ágil da empresa B, não tem contato diretamente com o cliente para buscar entender seus problemas e suas necessidades e enxerga a falta do contato direto com o cliente como uma fraqueza (Entrevista 4, questão 41). Porém recebe o *feedback* com relação às funcionalidades apresentadas aos clientes (Entrevista 5, questão 30).

O diálogo constante com consultores ou outra área funcional que mantêm contato contínuo com o cliente também é considerado valioso pelas equipes ágeis (Entrevista 3, questão 48; Entrevista 1, questão 13; Entrevista 1, questão 38; Entrevista 3, questão 46; Entrevista 5, questão 43).

A comunicação direta entre as equipes ágeis e as outras áreas funcionais também gera uma aprendizagem que pode ser importante para a organização. São utilizadas técnicas como reuniões programadas para relatar o *status* (Entrevista 4, questão 23; Entrevista 5, questão 47) ou a participação de representantes de outras áreas dentro da equipe de desenvolvimento (Entrevista 3, questão 41)

A aprendizagem dialógica utiliza-se da síntese do conhecimento tácito e do conhecimento explícito. Este tipo de aprendizagem desempenha um papel extremamente importante na gestão do conhecimento que é a articulação do conhecimento inicialmente tácito em conhecimento explícito (externalização do conhecimento) ou, no sentido contrário, o conhecimento explícito é internalizado em conhecimento tácito, reformulando os modelos mentais dos indivíduos participantes.

5.2.2 Categoria Criação

Neste trabalho, criação apresenta uma concepção de criar para inovar, com enfoque no ser humano criativo. Criar é poder dar forma a algo novo (OSTROWER, 1977). Dessa forma, a categoria central Criação foi dividida em duas subcategorias: criatividade e oportunidades. Na subcategoria criatividade foram reunidos os códigos referentes a sessões de *brainstorming*, ao ambiente de criação e interações sociais que permitam a criatividade. Já na subcategoria oportunidades reuniram-se as reuniões, análises de mercado, *benchmarkings* e novas tecnologias.

5.2.2.1 Subcategoria Criatividade

Ostrower (1977) não trata a criatividade como propriedade exclusiva de algumas pessoas afortunadas, mas como uma condição inerente ao ser humano. A criatividade pode ser trabalhada e, até mesmo, por meio de técnicas contemplativas podem ser estimuladas e melhoradas (CORRIGAN, 2012).

As duas equipes ágeis estudadas utilizam-se de sessões de *brainstroming* para estimular a criatividade em reuniões de planejamento (Entrevista 1, questão 32; Entrevista 4, questão 15; Entrevista 1, questão 36). O *brainstorming* também é feito para soluções específicas, em conjunto com esboços e desenhos (Entrevista 3, questão 17) e além disso, pode ser visto como um capacitor de autonomia (Entrevista 4, questão 15, Entrevista 4, questão 16).

O local de trabalho pode representar um ambiente propício para a criatividade no dia a dia. Mesas próximas em um ambiente descontraído permitem a troca de ideias e discussões no instante em que as ideias brotam (Entrevista 4, questão 36; Entrevista 5, questão 36).

Os momentos intensivos de socialização também propiciam a criação de ideias. Para Nonaka e Takeuchi (1997) o compartilhamento do conhecimento tácito é a primeira etapa da formação de conceitos para inovação. Segundo De Masi (2000) o trabalhador criativo não sabe distinguir o trabalho do lazer, pois a ideia criativa não avisa em que hora vai aparecer. As equipes ágeis também identificam as reuniões e as confraternizações como um espaço e momento para a criatividade (Entrevista 2, questão 36).

A criatividade pode ser considerada a principal qualidade para a criação de conhecimento e as sessões de *brainstorming* aparecem como a principal técnica. Seu uso se dá principalmen-

te para a socialização do conhecimento e são particularmente eficazes no sentido de compartilhar o conhecimento tácito e criar novas perspectivas (NONAKA; TAKEUCHI, 1997).

5.2.2.2 Subcategoria Oportunidades

A subcategoria Oportunidades agrupa os códigos referentes a situações que possibilitem a inovação. Análise de mercado, análise de concorrentes, *Benchmarking* que também formam um conjunto de técnicas que permitem a aprendizagem organizacional em busca de novas oportunidades.

O monitoramento contínuo do mercado possibilita observar oportunidades de negócios. Na Empresa A, os consultores de linha de frente que ficam dentro de clientes desempenham um papel de destaque no entendimento do que o mercado está demandando (Entrevista 1, questão 13; Entrevista 1, questão 38). Porém a responsabilidade de entender essa demanda é, em última instância, do *Product Owner* (Entrevista 1, questão 38) que busca o meio termo entre a pesquisa de mercado por meio de análise de concorrentes e a análise de clientes para estabelecer o *full backlog* (uma espécie de *backlog* da *release*) (Entrevista 1, questão 23).

O cliente tem grande relevância na composição da lista do *backlog* do produto (Entrevista 1, questão 50). As vezes a oportunidade aparece diretamente por solicitação ou análise do cliente e aproveitar a oportunidade está em reconhecer o padrão de maneira que possa ser replicado a outros clientes (Entrevista 3, questão 19).

Outra forma de buscar oportunidades no mercado é participando de eventos externos para sentir o mercado (Entrevista 4, questão 36). Ou mesmo analisar as consultorias de mercado como Gartner, Forrester que indicam tendência de mercado (Entrevista 1, questão 38; Entrevista 2, questão 38).

A análise de concorrentes também é utilizada nas empresas analisadas como uma forma de melhorar suas próprias concepções para observar tendências e encontrar oportunidades no mercado (Entrevista 3, questão 36; Entrevista 1, questão 50; Entrevista 4, questão 38).

A busca por oportunidades é responsabilidade de todos na empresa. Embora a decisão final esteja no nível grupal ou individual, todos os níveis ontológicos se envolvem em encontrar novas oportunidades para possibilitar a criação de conhecimento na organização.

5.2.3 Categoria Construção

A construção é o principal foco das metodologias ágeis ou, no caso das equipes analisadas, do *scrumban*, que seria uma mescla do scrum com o kanban. Suas técnicas foram pensadas principalmente para evitar desperdícios no processo de construção do software e oferecer uma maior flexibilidade referente aos *sprints*.

A categoria central Construção apresenta duas subcategorias: Implementação e Acompanhamento. Na subcategoria Implementação foram agrupados os códigos *backlogs*, modelagens, protótipos e testes. Já na subcategoria Acompanhamento foram reunidos os códigos kanban, reuniões diárias e de retrospectivas e prazos.

5.2.3.1 Subcategoria Implementação

Kanban, em japonês, significa “cartão” pois seu conceito principal gira em torno dos cartões que ficam visíveis para todos acompanharem a execução das tarefas. Os membros da equipe ágil da Empresa A explicam, resumidamente, que o funcionamento do Kanban é parecido com o scrum pela estrutura, mas adicionado dos *cards* que apresentam um ciclo de vida que se iniciam no *full backlog*, passando pelos estágios *This Week*, *Doing*, *Blocked*, *Waiting Test*, *Building*, *Staging* e finalizam na *Release* (Entrevista 1, questão 18).

A opção pela utilização do kanban (ou no caso o scrumban) se deve pela sua maior flexibilidade que o scrum. O scrum apresenta o conceito de sprints fechados, ou seja, o que está na lista do *backlog* do *sprint* dever ser priorizado, já o kanban não vê problema em priorizar e despriorizar os cards oferecendo maior agilidade às respostas impostas pelo ambiente (Entrevista 3, questão 18; Entrevista 1, questão 18).

A técnica de modelagem é utilizada por ambas as equipes analisadas, em maior ou menor grau de formalismo. Enquanto a Empresa B utiliza-se de padrões de arquitetura de software como o *model-view-controller* (MVC) ou o modelo entidade relacionamento (MER) para apoiar a construção dos software (Entrevista 4, questão 28) a empresa A guarda os esboços das modelagens em arquivos digitais para consultas futuras (Entrevista 3, questão 33).

A utilização de protótipos no desenvolvimento de software é incentivado pelo método XP pois permite um melhor entendimento do produto de software por todos os envolvidos. Du-

rante o desenvolvimento, a empresa B, além do protótipo também utiliza *wareframe* para uma prévia discussão da interface, respeitando etapas de um processo (Entrevista 4, questão 21). A utilização dessas técnicas permite uma rápida construção de um Produto Mínimo Viável, como o proposto por Ries (2012) para uma melhor validação do produto em busca de produtos inovadores.

Embora as equipes estudadas não se utilizem de metodologias orientada a testes, elas compreendem a importância e acreditam que os testes fazem parte do processo de construção do produto de software. Na empresa B, o teste faz parte do sprint (Entrevista 4, questão 30) e na empresa A, toda funcionalidade implementada pressupõe um teste para ela (Entrevista 1, questão 30) e a última está em um processo de migração para o *Test Driven Development* (TDD) (Entrevista 3, questão 31).

A equipe ágil da Empresa A utiliza diversas ferramentas TICs para apoiar a implementação dos programas de software que são desenvolvidos. O Slack é usado como repositório de conhecimento e comunicação, o Dropbox como compartilhamento de arquivos, o Bitbucket como controle de versão, a integração contínua é feita com o Jenkins, além de e-mails e mensageiros instantâneos para a comunicação direta (Entrevista 3, questão 37; Entrevista 1, questão 37).

As principais técnicas do scrumban apoiam a etapa construção do software. Do ponto de vista da agilidade, é importante encurtar a distância entre a criação e a construção do software e as metodologias ágeis oferecem a oportunidade de enxergar esse processo como um processo único, visto que as eventuais alterações que ocorram no meio do percurso sejam facilmente incorporadas para a construção do software. A possibilidade de experimentar com rapidez, utilizando protótipos ou um Produto Mínimo Viável oferece o suporte necessário para validar os conceitos criados e possibilita a aprendizagem (RIES, 2012).

5.2.3.2 Subcategoria Acompanhamento

O uso dos cartões no kanban constitui uma das técnicas de acompanhamento mais eficientes em diversas situações. Seu acompanhamento visual permite ter noção, em tempo real, da quantidade de trabalho em cada etapa do processo (Entrevista 1, questão 30; Entrevista 4, questão 18).

A equipe ágil da Empresa A utiliza o Trello, uma ferramenta gratuita, flexível e visual de organizar os *cards* para apoiar a implementação e, principalmente, acompanhamento das atividades que estão sendo desenvolvidas, as que já foram feitas e as que ainda serão implementadas (Entrevista 2, questão 11; Entrevista 2, questão 37; Entrevista 3, questão 18).

Embora a equipe da Empresa A não faça estimativa para prever o prazo de entregas, ela acompanha e registra todo o histórico de quantos cards são implementados por dia, média de cards bloqueados, se gasta mais tempo implementando uma funcionalidade ou em um *bug* (Entrevista 1, questão 18; Entrevista 3, questão 31).

O principal acompanhamento se dá nas reuniões diárias e no dia a dia devido à proximidade das mesas de trabalho. Nas reuniões diárias é discutido tudo que foi feito desde última reunião, os impedimentos que a equipe tem encontrado e o que será feito até a próxima reunião (Entrevista 1, questão 18; Entrevista 3, questão 21). A equipe ágil da Empresa B somente utiliza as reuniões em situações nas quais é necessário um acompanhamento mais rigoroso devido ao prazo de entrega, caso contrário preferem que, principalmente, os impedimentos sejam ditos na hora em que eles estão sendo ocasionados, de forma a tentar minimizar os desperdícios (Entrevista 4, questão 20).

As reuniões de retrospectiva também funcionam como acompanhamento das atividades que foram feitas. Essa retrospectiva repassa o que foi feito durante a semana e pode influenciar no que será feito na próxima semana (Entrevista 3, questão 21). Na Empresa B a equipe também relata o *status* das atividades para as outras áreas da organização (Entrevista 4, questão 23).

Acompanhar e medir o processo de desenvolvimento também merece destaque. Tanto no controle do processo produtivo quanto na validação com clientes ou usuários, acompanhar e medir possibilitam o aprendizado e o autoconhecimento da eficiência do grupo no projeto. Esse aprendizado permite ajustar o direcionamento do projeto e ajuda na validação dos conceitos criados.

5.3 Descrevendo o Fenômeno 2

O Fenômeno 2 observado tem como foco central uma gestão colaborativa por meio de inteligência coletiva, que disseminam fatores capacitadores e que promovem a inovação e o ali-

nhamento entre a cultura da organização e as características pessoais dos colaboradores. Assim, o Fenômeno 2 apresenta três categorias centrais:

1. Gerenciamento por meio de inteligência coletiva;
2. Fatores capacitadores relativos à cultura organizacional; e
3. Fatores capacitadores relativos às características pessoais.

A Inteligência Coletiva, neste trabalho, refere-se ao planejamento e à tomada de decisão de forma coletiva, por toda a equipe ou por toda a empresa, além do conhecimento sobre o conhecimento da equipe, chamado neste trabalho de memória transacional, memória coletiva ou de grupo. A categoria Cultura Organizacional refere-se ao ambiente, à atmosfera de trabalho e aos valores da empresa. Já as Características Pessoais referem-se ao perfil dos colaboradores e como esses colaboradores se organizam como parte da equipe. O Quadro 10 sintetiza a relação do Fenômeno 2, com as categorias centrais e suas subcategorias.

Quadro 10 - Categorias e subcategorias do Fenômeno 2.

Fenômeno 2: A equipe ágil é gerenciada mediante uma eficiente **comunicação** que privilegie a **inteligência coletiva** e promove fatores capacitadores que alinham a **cultura organizacional** e as **características pessoais** dos colaboradores apoiada por **ferramentas** adequadas.	
Categorias Centrais	Subcategorias
Inteligência Coletiva	Planejamento coletivo e decisão coletiva
	Memória transacional (Memória Coletiva)
Cultura Organizacional	Ambiente de trabalho
	Valores
Características Pessoais	Perfil
	Parte da equipe

5.3.1 Categoria Inteligência Coletiva

O conceito de Inteligência Coletiva refere-se a um tipo de inteligência compartilhada a partir da colaboração de diversos indivíduos. Para Lèvy (1999), trata-se de uma inteligência na massa distribuída, um coletivo de indivíduos que pensam juntos, num contexto social. Emprestando a ideia de inteligência coletiva ao contexto de uma organização de desenvolvimento

de software, a equipe ágil procura utilizar-se do conhecimento distribuído de cada indivíduo para formar um produto comum, o software.

Dessa forma, neste trabalho, a inteligência coletiva pode ser entendida como o planejamento, a decisão e a execução de forma coletiva já que ninguém sabe tudo, porém todos sabem alguma coisa (LÈVY, 1999). Embora, em última instância, o líder ou o gerente tomem a decisão final.

A categoria central Inteligência Coletiva apresenta duas subcategorias: Planejamento e Decisão Coletiva e Memória Transacional. Na subcategoria Planejamento e Decisão Coletiva foram agrupados os códigos participação de todos, decisão coletiva e redundância. Já na subcategoria Memória Transacional foram reunidos os códigos confiança, variedade de requisitos, Wiki e multifuncionalidade.

5.3.1.1 Subcategoria Planejamento Coletivo e Decisão Coletiva

Diversos fatores influenciam o planejamento das equipes ágeis. Esse planejamento procura levar em consideração a análise de mercado por meio de consultorias externas, avaliações de concorrentes e consultores de linha de frente ou consultores especializados (Entrevista 1, questão 13; Entrevista 1, questão 50; Entrevista 3, questão 46, Entrevista 1, questão 47; Entrevista 3, questão 38).

As duas equipes ágeis estudadas fazem uma reunião de planejamento semanal (Entrevista 1, questão 30; Entrevista 4 questão 20). Durante a reunião de planejamento são feitas sessões de *brainstorming* e procuram decidir o que deve ser feito durante a semana (Entrevista 1, questão 32). Todas as ideias são ouvidas e discutidas em busca de um melhor plano para o desenvolvimento das atividades (Entrevista 2, questão 39; Entrevista 2, questão 17; Entrevista 4, questão 15)

Em situações nas quais o prazo de entrega não é problema, a autonomia técnica se estende à gerencial, ao *o quê fazer*, cabendo aos colaboradores se auto-organizarem na atribuição das tarefas (Entrevista 2, questão 22). Envolver a equipe nas atividades de planejamento favorece a autoestima e aumenta o engajamento dos colaboradores com os objetivos da organização (Entrevista 2, questão 50). A decisão sobre o planejamento procura ouvir a opinião de todos

porém a decisão final é sempre do líder ou gerente (Entrevista 2, questão 15; Entrevista 2, questão 50).

Em situações de discussão para resolução de problemas as propostas de soluções são ouvidas e discutidas por todos da equipe e a decisão coletiva também pode ser observada. A equipe ágil da Empresa A utiliza-se de desenhos para visualizar as propostas de soluções e por meio da opinião de cada um dos colaboradores procuram chegar a um consenso para a melhor solução (Entrevista 3, questão 17). Já a equipe ágil da Empresa B, normalmente utilizam-se de protótipos e *wareframes* para enriquecer a discussão (Entrevista 4, questão 21, Entrevista 4, questão 14).

Envolver todos os colaboradores no planejamento e na tomada de decisão conduz a uma maior e melhor variedade de requisitos além de promover a redundância de informação. Para maximizar a variedade de requisitos todos os membros da organização devem ter livre acesso à informação para auxiliar o planejamento e a tomada de decisão (NONAKA; TAKEUCHI, 1997). Já a redundância traz a superposição intencional de informações o que permite um melhor planejamento e uma tomada de decisão mais fundamentada. Ela é importante, sobretudo, na etapa de formação de conceito e possibilita o “aprendizado por intrusão” (NONAKA; TAKEUCHI, 1997).

5.3.1.2 Subcategoria Memória Transacional (Memória Coletiva)

Em atividades intensivas de conhecimento como o desenvolvimento de software não é apenas importante ter conhecimento técnico, mas também saber quem sabe o quê. Corrigan (2012) emprestou o conceito de memória transacional (*transactive memory*), desenvolvido por Daniel Wegner (1985) no campo da psicologia, na qual refere-se ao conceito de um mecanismo de memória de grupo que, coletivamente, codifica, armazena e recupera conhecimento.

Apesar de as equipes estudadas não terem uma lista do perfil e do conhecimento técnico de seus colaboradores as reuniões são os melhores momentos para saber quem sabe o quê (Entrevista 2, questão 35). As reuniões aumentam o entrosamento da equipe e ao expor um impedimento ou alguma limitação, as pessoas que detêm aquele conhecimento necessário para transpor esse impedimento prontamente se apresentam (Entrevista 3, questão 35, Entrevista 5, questão 35).

A autonomia e a proatividade são características pessoais que estão intimamente ligados à memória transacional. Autonomia permite o discernimento dos colaboradores com relação ao conhecimento pessoal e ao conhecimento do grupo (Entrevista 4, questão 15; Entrevista 4, questão 16). Já a proatividade faz com que os colaboradores busquem saber fazer e isso passa por saber utilizar-se dos conhecimentos dos outros membros do grupo (Entrevista 4, questão 35).

A programação em par também utiliza-se desse conhecimento distribuído para a resolução de um mesmo problema. Dois programadores apresentam visões de mundo diferentes, trazem redundância de informação e permitem a construção da solução colaborativamente (Entrevista 5, questão 25).

Uma importante ferramenta que auxilia o uso eficiente da memória transacional é a Wiki. A equipe ágil da Empresa B utiliza a Wiki como um repositório de conhecimento e seus mecanismos de controle de versão permitem saber quem são as pessoas que podem ajudar em um determinado assunto por ter preenchido e colaborado com o tópico específico de conhecimento, além de permitir o ensino e aprendizado (Entrevista 4, questão 34).

5.3.2 Categoria Cultura Organizacional

Para Shein (1992), a cultura organizacional pode ser entendida como um padrão de pressuposições básicas compartilhadas por um grupo na resolução de problemas de adaptação externa e integração interna com resultados satisfatórios para serem consideradas como válidas e possam se propagar aos novos membros como a maneira correta de perceber, pensar e sentir.

De maneira similar, Pettigrew (1979) acredita que a cultura organizacional pode ser entendida como um sistema de significados aceitos por um determinado grupo em um certo período de tempo. Já Hofstede et al. (1990) acreditam que a cultura organizacional está relacionada com a história e tradição da organização e é por natureza coletiva, compartilhada e ideacional.

A categoria central Cultura Organizacional também apresenta duas subcategorias: Ambiente de Trabalho e Valores. Na subcategoria Ambiente de Trabalho foram agrupados os códigos desafios, interação face a face, ambiente descontraído e confraternizações. Já na subcategoria Valores foram reunidos os códigos transparência, intenção e recompensa.

5.3.2.1 Subcategoria Ambiente de Trabalho

Os esforços da gestão do conhecimento concentram-se em criar um ambiente de trabalho propício à criação de conhecimento. Nonaka e Konno (1998) definem esse ambiente como "*ba*", que pode ser entendido como "lugar" ou "ambiente" e refere-se ao espaço dinâmico compartilhado propício a criação, compartilhamento e utilização do conhecimento. Este espaço pode ser físico, virtual, mental ou combinações dos mesmos. Neste trabalho, ambiente de trabalho pode ser entendido como a atmosfera encontrada dentro da organização, a relação entre os colaboradores e seus grupos de trabalho propícios a criação de conhecimento.

Um ambiente descontraído, que privilegie a troca de experiência e diálogo contínuo possibilita aflorar a criatividade e aumentar a produtividade, pois aumenta a predisposição dos colaboradores (Entrevista 2, questão 49).

O espaço físico impacta diretamente nesse ambiente de trabalho. A proximidade das mesas possibilita um diálogo contínuo mesmo durante o desenvolvimento de software. Essa configuração do lugar fornece ao time um ambiente propício ao surgimento de ideias e pode eliminar desperdícios em caso de situações nas quais há algum impedimento à execução de uma determinada tarefa (Entrevista 4, questão 36; Entrevista 5, questão 47). Além disso, as paredes do local de trabalho funcionam como um painel de comunicação eficiente quando utilizado o quadro do kanban (Entrevista 1; questão 31; Entrevista 4, questão 18). A utilização de ferramentas TICs auxilia nessa comunicação e permite aos desenvolvedores uma eficiente comunicação independente do tempo e do espaço (Entrevista 2, questão 33).

Confraternizações podem ser consideradas uma extensão do ambiente de trabalho. Segundo De Masi (2000) trabalhadores do conhecimento atuam durante todo o tempo e não somente durante o expediente de trabalho. As confraternizações oferecem um espaço rico e diversificado com um alto grau de liberdade para os colaboradores desenvolverem suas criatividades. As empresas estudadas incentivam a confraternização, não apenas financeiramente, mas como algo institucionalizado. A Empresa B oferece comida e bebida após o expediente para seus colaboradores confraternizarem e buscarem a troca de conhecimento com o Espaço *Nerd* (Entrevista 4, questão 16). Já a Empresa A conta com diversas confraternizações: comemoração de entrega de *releases*, festas juninas e de fim de ano, além de confraternizações mensais e confraternizações não programadas (Entrevista 1, questão 39; Entrevista 2, questão 22; Entrevista 2, questão 32).

5.3.2.2 Subcategoria Valores

Como explicado no Capítulo 2, as metodologias de desenvolvimento ágil de software foram baseadas nos processos de produção enxuta que primam pela qualidade e procuram evitar desperdícios. A busca permanente da qualidade de seus produtos e serviços e a intenção constante de evolução da empresa incentivam os colaboradores a fazerem cada vez melhor e estimulam a vontade de crescer pessoalmente (Entrevista 2, questão 13; Entrevista 5, questão 49).

Outro valor que pode influenciar no processo criativo e produtivo é a transparência da empresa. Uma empresa que não oculta a situação financeira e envolve os seus colaboradores nas tomadas de decisões faz com que esses colaboradores se sintam parte ativa da empresa (Entrevista 2, questão 10; Entrevista 2, questão 38).

Uma cultura organizacional que valoriza e recompensa seus talentos é essencial para uma boa gestão do conhecimento. Ambas as empresas estudadas não têm um programa de recompensa institucionalizado, porém sempre procuram premiar seus funcionários de destaque. A Empresa A, busca premiar os funcionários que se destacaram seja mensal ou trimestralmente. Dessa maneira, ela incentiva seus colaboradores a trazerem boas ideias e se esforçarem para fazer um trabalho acima da média (Entrevista 3, questão 16; Entrevista 2, questão 16). A equipe ágil da Empresa B também premia e se diverte com metas criadas por eles mesmos, mostrando auto-organização (Entrevista 4, questão 16). Para as equipes ágeis estudadas, o reconhecimento é uma grande recompensa (Entrevista 2, questão 40) e eventos que promovam a aprendizagem pode ser considerado uma recompensa pois o grande prêmio é o conhecimento (Entrevista 5, questão 33).

5.3.3 Categoria Características Pessoais

O professor estado-unidense Douglas McGregor (1960) propõe a teoria X e a teoria Y, no seu livro "*The Human Side of Enterprise*" que são teorias de motivação e gestão de recursos humanos. A teoria X, também conhecida como "hipótese da mediocridade das massas" considera que os trabalhadores não gostam do seu trabalho e tratam-se de trabalhadores pessimistas, estáticos e rígidos que sempre procurarão evitar o trabalho. Já na teoria Y os trabalhadores são considerados proativos, competentes e responsáveis. Eles aceitam o trabalho de forma natural, gostam de trabalhar e fazem por diversão desde que as condições sejam favoráveis.

Como é sabido, a equipe ágil é auto-organizável e deve ser capaz de conduzir seus compromissos proativamente mediante uma autonomia concedida. Dessa maneira o perfil dos membros de uma equipe ágil naturalmente se enquadra na teoria Y, ou seja, colaboradores motivados, proativos, comprometidos com o trabalho e que naturalmente compartilham conhecimento para crescer como equipe.

A categoria central Características Pessoais apresenta duas subcategorias: Perfil e Parte da Equipe. Na subcategoria Perfil foram agrupados os códigos proatividade, liderança e adaptabilidade. Já na subcategoria Parte da Equipe foram reunidos os códigos multifuncionalidade, autonomia, auto-organização, confiança e liberdade de expressão.

5.3.3.1 Subcategoria Perfil

O perfil dos colaboradores e suas capacidades, quando alinhados com a cultura organizacional, completam esse ambiente de sucesso. A proatividade em conjunto com a autonomia possibilitam à equipe ser criativa e atingir uma alta produtividade (Entrevista 1, questão 49; Entrevista 3, questão 49). Além disso, essas duas características podem ser direcionadas pelos padrões estabelecidos pela organização (Entrevista 2, questão 14).

Os líderes de desenvolvimento nas empresas estudadas desempenham o papel de *Scrum Master* dentro da equipe. Uma espécie de gerente de nível médio, que segundo Nonaka e Takeuchi (1997), desempenha um papel central na gestão do conhecimento. Nas duas equipes estudadas foi possível perceber uma certa admiração pelo líder seja pela sua capacidade técnica, gerencial ou em momentos de aprendizagem (Entrevista 2, questão 22; Entrevista 5, questão 22). Além disso, os líderes das equipes estudadas apresentam uma característica pessoal fundamental: ter a sensibilidade para atribuir desafio aos colaboradores e extrair deles os melhores resultados (Entrevista 2, questão 14).

5.3.3.2 Subcategoria Parte da Equipe

Nas duas equipes estudadas a autonomia é concedida aos colaboradores diretamente proporcional ao seu conhecimento (Entrevista 3, questão 14; Entrevista 4, questão 16). A equipe ágil da Empresa B busca atribuir desafios e treinamentos visando aumentar a autonomia de seus colaboradores (Entrevista 4, questão 16). Para o *Scrum Master* da equipe ágil da Empre-

sa A, conceder autonomia é importante pois estimula a pessoa a buscar a melhor solução e força a aprender cada vez mais (Entrevista 3, questão 14). Já o *Product Owner* desta mesma equipe acredita que a autonomia aumenta a agilidade e faz com que os membros da equipe se sintam donos daquilo que estão fazendo (Entrevista 1, questão 15).

Ao contrário dos times tradicionais nos quais o conhecimento é tido como um produto estratégico e que não dever ser compartilhado sob perda da posição privilegiada que o conhecimento lhe concede, nas equipes ágeis valoriza-se o compartilhamento de conhecimento. Para o líder da equipe ágil da Empresa B é de extrema importância criar uma cultura de compartilhamento de conhecimento e dessa forma são elaborados metas de externalização do conhecimento na Wiki da equipe (Entrevista 4, questão 16). Na Empresa A, colaboradores que produzem mais conteúdo de conhecimento para ser replicado internamente normalmente são premiados, o que incentiva o compartilhamento de conhecimento (Entrevista 3, questão 34).

Por fim, a liberdade de expressão também é incentivada nas duas equipes estudadas. Um indivíduo que é reprimido em suas angústias não se sente livre para se abrir e, consequentemente, guardam suas ideias para si próprio. Nas equipes estudadas, os membros são incentivados a compartilharem seus sentimentos, suas emoções e angústias, principalmente durante as reuniões diárias, o que permite um maior entrosamento da equipe (Entrevista 1, questão 32; Entrevista 4, questão 32).

5.4 *Framework*: Técnicas e Fatores que Promovem a Criação de Conhecimento

Após aplicar todas as etapas da TFD e analisar os fenômenos encontrados, sentiu-se a necessidade de reunir os resultados na forma de um modelo para a teoria substantiva encontrada. O modelo proposto não tem a finalidade de ser generalizado a todos os domínios e sim elucidar a teoria emergida dos estudos de caso. A análise dos resultados dos fenômenos 1 e 2 são apresentados em forma de quadro (*framework*) para ilustrar a teoria substantiva que emergiu dos dados, conforme mostra a Figura 13.

Embora as equipes ágeis estudadas estabeleçam técnicas de aprendizagem organizacional, criação e construção, utilizem-se de fatores capacitadores em todos os níveis ontológicos e privilegiem um planejamento e decisão coletiva, elas não estabelecem um procedimento sequencial que integre todas as etapas como um fluxo de trabalho ou um processo de negócio.

A comunicação privilegia a socialização por meio de interações face a face, seja em reuniões, confraternizações ou mesmo no local de trabalho, além de comunicação mediante uma intermediação tecnológica. As ferramentas de trabalho são, em sua maioria, baseadas em TICs e auxiliam as equipes ágeis em todas as etapas do desenvolvimento de software.

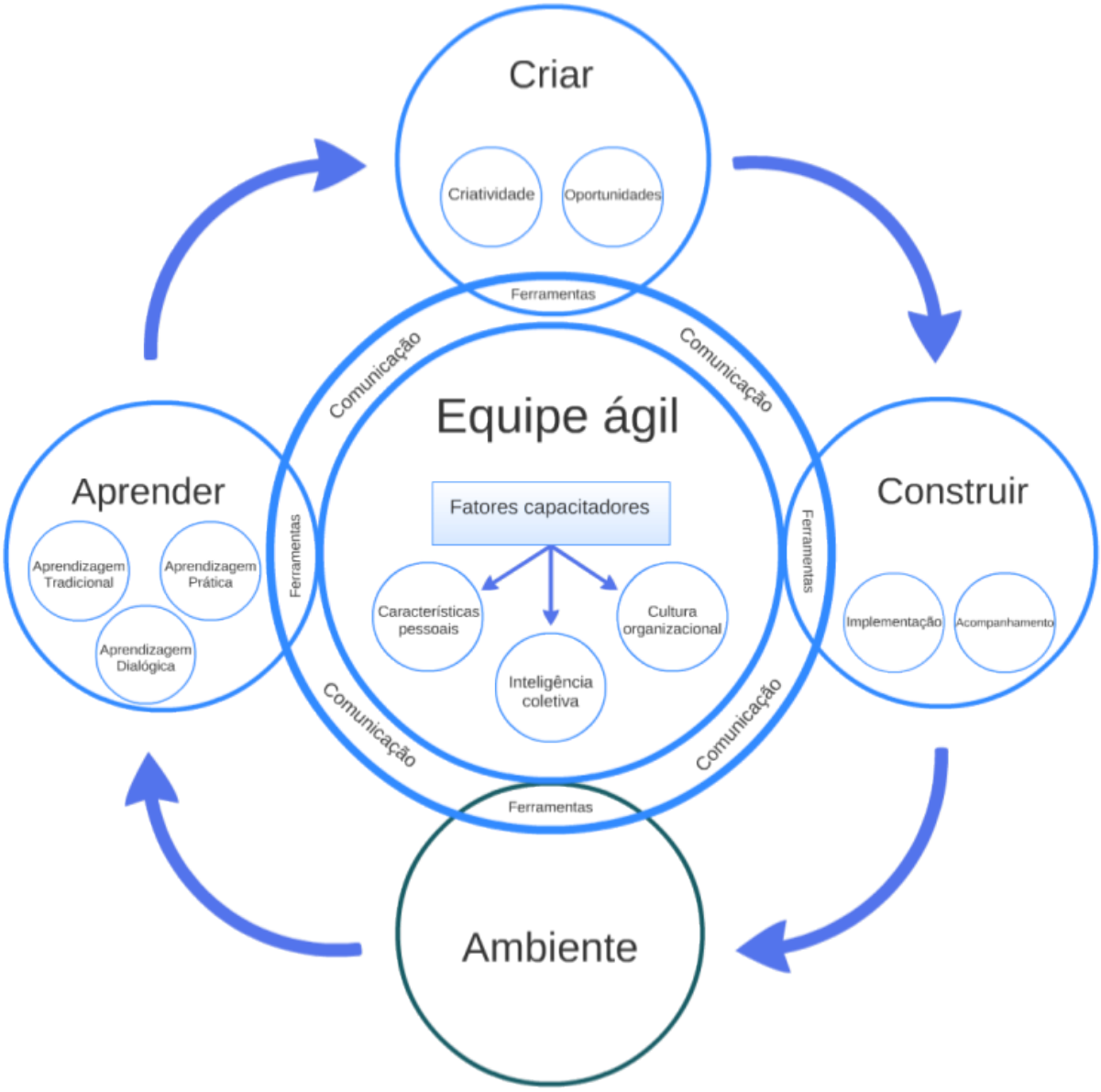

Figura 13 –*Framework*: Teoria substantiva emergida dos dados.

6 CONCLUSÕES

Neste trabalho foram abordados os principais conceitos de criação de conhecimento na empresa, analisados as abordagens metodológicas de desenvolvimento de software, com foco nas metodologias ágeis, em seguida foi feita uma revisão sistemática de literatura, utilizando-se do framework PRISMA para tentar compreender o Estado da Arte da criação de conhecimento no desenvolvimento de software e identificar técnicas e fatores que influenciam na criação de conhecimento. Após a revisão de literatura foi elaborado o desenho de pesquisa, preparado um roteiro semiestruturado para entrevistas, que foi pré-validado com um grupo de alunos de pós-graduação da UNICAMP e adquiridas todas as aprovações junto ao Comitê de Ética em Pesquisa. Em seguida, foram feitas entrevistas em duas empresas de software de Campinas, sendo com três colaboradores da Empresa A e dois colaboradores da Empresa B. Para análise qualitativa dos dados coletados, este trabalho se baseou na Teoria Fundamentada de Dados (*Grounded Theory*) que procurou classificar os principais conceitos (códigos) em grupos de conceitos-chaves e analisar suas inter-relações.

Como resultados foram encontrados dois fenômenos centrais que compuseram a teoria substantiva que emergiu dos dados coletados e foi apresentado um framework que sintetiza a teoria emergida. O Fenômeno 1 indica que: “A equipe ágil estabelece ***técnicas*** que apoiam a **aprendizagem organizacional**, que ajudam na **criação** e na **construção** de software inovadores e que permitem uma eficiente **comunicação** entre todas as partes envolvidas fazendo uso de **ferramentas** adequadas.”. Já o Fenômeno 2 sugere que: “A equipe ágil seja gerenciada mediante uma eficiente **comunicação** que privilegie a **inteligência coletiva** e promova ***fatores capacitadores*** que alinham a **cultura organizacional** e as **características pessoais** dos colaboradores apoiados por **ferramentas** adequadas”.

Do Fenômeno 1, pode-se concluir que, estabelecer técnicas que propiciam a aprendizagem organizacional é tão importante quanto praticar técnicas para criação e construção dos programas de software.

A aprendizagem organizacional é uma das características centrais das empresas que gerenciam seu conhecimento. Ela deve ser vista como um processo de interação (aquisição, transferência e criação) entre os níveis individual, grupal, organizacional e inter-organizacional, com troca

de conhecimentos que resultam em mudança e adaptação organizacional. Vendo a Aprendizagem Organizacional como um processo de natureza comportamental que se estabelece no subsistema técnico e social (GARVIN et al. 1998), podemos remeter, grosso modo, o subsistema técnico ao conhecimento explícito e o subsistema social ao conhecimento tácito. Segundo Nonaka e Takeuchi (1997), uma organização cria, utiliza e difunde o conhecimento convertendo o conhecimento tácito em conhecimento tácito (socialização), conhecimento tácito em conhecimento explícito (externalização), conhecimento explícito em conhecimento explícito (combinação) e conhecimento explícito em conhecimento tácito (internalização). Portanto, a aprendizagem nas organizações é resultado de um processo de ampliação do conhecimento individual, para o conhecimento grupal e transformando-o em saber organizacional de forma cristalizada, que influência nos comportamentos dos indivíduos. As empresas estudadas estabelecem técnicas de aprendizagem tradicional, aprendizagem prática e aprendizagem dialógica.

Sendo a criação de conhecimento a finalidade da aprendizagem organizacional, a inovação pode ser entendida como o resultado dos processos de aprendizagem organizacional nos quais conhecimentos são combinados e restruturados em forma de novas soluções que agreguem valor. Dessa maneira, as equipes ágeis utilizam-se de técnicas, não só para a construção dos produtos de software, mas também utilizam-se de técnicas que possibilitam a criação, a inovação. Para a criação, as empresas estudadas apoiam-se em técnicas que estimulem a criatividade, com destaque para as sessões de *brainstormings*, e ficam atentas às oportunidades que o mercado pode oferecer.

As principais técnicas das metodologias de desenvolvimento de software concentram-se nos processos de construção: implementação e acompanhamento. Algumas técnicas utilizadas pelas empresas estudadas mesclam técnicas das metodologias tradicionais com as de metodologias ágeis, como o caso de algumas documentações e as reuniões diárias. Este fato evidencia a importância da dualidade das abordagens tradicionais e ágeis, do ponto de vista da gestão do conhecimento.

O processo de construção, na gestão do conhecimento, não pode ser totalmente separado da etapa de aprendizagem organizacional e de criação de conceitos, por isso, as empresas deveriam encarar a aprendizagem organizacional, criação e construção com o caráter interdisciplinar que os temas exigem.

Com relação ao Fenômeno 2, é possível concluir que uma gestão colaborativa, além de incentivar a participação e o envolvimento de todos no processo de desenvolvimento, planejamento e tomada de decisão, permite uma maior variedade de requisitos e traz a redundância de informação.

A inteligência coletiva refere-se a uma inteligência de grupo, no qual ninguém sabe tudo, porém todos sabem alguma coisa (LÈVY, 1999). Pôde-se observar nas duas equipes ágeis de desenvolvimento analisadas que o planejamento dos sprints ou o planejamento da *release* são discutidos por todos e a decisão procura ser coletiva. A programação em par é outra forma de se utilizar a inteligência coletiva, na qual dois programadores trazem visões de mundo diferentes para a resolução de um mesmo problema. As equipes estudadas procuram planejar e decidir coletivamente, embora, em última instância, a decisão final fique a cargo do líder ou gerente. As equipes também utilizam-se de memória coletiva ou transacional que são estimuladas, principalmente, durante as reuniões.

A cultura organizacional e as características pessoais dos colaboradores devem estar alinhadas. Quando os colaboradores se identificam com a intenção e valores da organização e encontram um ambiente descontraído e propício à criação de ideias, em que são valorizados por seus palpites estabelece-se uma relação de confiança mútua entre a organização e seus colaboradores. Essa confiança mútua está diretamente relacionada com a autonomia que a empresa concede e à proatividade do colaborador. A relação desses dois fatores pode contribuir para que a organização seja criadora de conhecimento.

A liderança é fundamental para aumentar a autoconfiança em si e na equipe. O líder deve ter a sensibilidade para atribuir autonomia e desafios que estimulem a resolução de problema e a aprendizagem por parte dos colaboradores, mexendo com suas zonas de conforto. Conforme explicam, Nonaka e Takeuchi (1997), atribuir esses desafios significa um ambiente que permita a flutuação e caos criativo, atribuindo um colapso de rotinas o que permite reconsiderar suas perspectivas fundamentais.

Por fim, deve ser estimulado a liberdade de expressão e uma cultura de compartilhamento de conhecimento. A liberdade de expressão é uma condição chave para o compartilhamento do conhecimento e de sentimentos. Numa sociedade baseada no conhecimento, é fundamental

o seu compartilhamento, o que permite um crescimento exponencial do conhecimento organizacional.

As técnicas de reuniões (planejamento, diárias, retrospectivas) e a programação em par (e rotação de trabalho), que tiveram destaques na revisão de literatura, também se destacaram nas equipes ágeis estudadas. As reuniões e a programação em par desempenham um papel extremamente relevante em três modos de conversão de conhecimento e propiciam a disseminação de diversos fatores capacitadores, incluindo os cinco fatores propostos por Nonaka e Takeuchi (1997): intenção, autonomia, redundância, flutuação e caos criativo e variedade de requisitos.

As reuniões diárias, as reuniões de planejamento *sprint* e reuniões de retrospectiva do *sprint* têm sua principal relevância na socialização pois essas interações sociais face a face possibilitam a transmissão de múltiplos sinais melhorando a capacidade de reestruturação dos modelos mentais. Também se destacam na conversão do conhecimento tácito em conhecimento explícito quando são incentivados os usos de analogias e metáforas, o que também possibilita externalização desses modelos mentais. As reuniões são utilizadas como técnicas para aprendizagem organizacional, criação e construção dos programas de software, além de serem o palco principal para a inteligência coletiva.

Quando duas pessoas atuam juntas em um mesmo problema elas trazem diferentes visões de mundo, o que aumenta a variedade de requisitos da equipe em suprir a exigência do ambiente por meio da redundância de conhecimento. A rotação de pares (ou rotação de trabalho), além dos mesmos benefícios da programação em par, ainda propicia a elevação do conhecimento do nível individual até o nível organizacional da dimensão ontológica. A programação em par pode ser utilizada como técnica para o aprendizado organizacional, embora seu foco principal seja na criação e, sobretudo, na construção de programas de software. Ela também aparece como uma importante técnica para apoiar a utilização da inteligência coletiva.

Essas duas técnicas (reuniões e programação em par) podem fomentar a criação de conhecimento nas organizações de desenvolvimento de software, porém vale ressaltar que nem todas as pessoas ou grupos de desenvolvimento apresentam características pessoais e culturais capazes de se beneficiarem dessas técnicas. A gestão do conhecimento requer o estabeleci-

mento de uma nova cultura nas empresas de software o que pode exigir uma reformulação de ideais e valores que podem fazer com que as pessoas se sintam perdidas.

Os resultados também sugerem que tanto as dualidades de conhecimento (explícito e tácito) quanto de desenvolvimento de software (tradicional e ágil) ou de pensar e agir são muito importantes para a gestão do conhecimento.

O principal contributo deste estudo está na classificação das técnicas e fatores de acordo com a teoria da criação de conhecimento e na definição das variáveis envolvidas na intersecção da gestão do conhecimento e do desenvolvimento de software. A classificação das técnicas e fatores de acordo com a teoria da criação de conhecimento organizacional permite identificar quais técnicas e fatores têm maior relevância para a criação de conhecimento, além de possibilitar o questionamento da utilidade de cada técnica ou fator. Os estudos de caso conduzidos possibilitaram conhecer as principais variáveis e compreender a dinâmica dessas variáveis nas equipes estudadas.

O caráter interdisciplinar do tema de pesquisa implica uma análise mais criteriosa e elaborada que deve procurar coexistir e respeitar os diferentes pontos de vistas advindas de diferentes disciplinas que contribuem para o tema. Foi ao mesmo tempo um desafio e estímulo a realização deste trabalho e não procurou esgotar os assuntos em cada tema, mas sim compreender justamente a dinâmica de todas as áreas envolvidas.

Esta pesquisa possibilitou, até o final do ano de 2015, a publicação de dois artigos científicos, sendo um para um congresso de gestão do conhecimento e inovação e outro para um periódico de informática, especificados a seguir:

1. SAKAI, E., RICARTE, I. L. M. Técnicas e Fatores que Promovem a Criação do Conhecimento no Desenvolvimento de Software In: V Congresso Internacional de Conhecimento e Inovação - ciKi, 2015, Joinville. Anais do Congresso Internacional de Conhecimento e Inovação. 2015.

2. SAKAI, E., RICARTE, I. L. M. Gestão do Conhecimento em Empresas de Desenvolvimento Ágil de Software: Conceitos e Práticas Essenciais. *Journal on Advances in Theoretical and Applied Informatics*., v.1, p. 13 - 21, 2015.

Por fim, com este trabalho foi possível verificar a existência de algumas lacunas na gestão do conhecimento no desenvolvimento de software, além de identificar algumas variáveis envolvidas na gestão do conhecimento no desenvolvimento de software. Assim, aproveitando-se de algumas dessas lacunas, questões que poderão ser abordadas em trabalhos futuros incluem:

- Qual o melhor processo gerencial para a criação de conhecimento no desenvolvimento de software?
- Como as principais técnicas de desenvolvimento de software possibilitam a criação de conhecimento na prática?
- É possível se basear em um *workflow* para a criação de conhecimento no desenvolvimento de software que contemple: aprendizagem, criação, construção?
- Os resultados encontrados nas equipes estudadas podem ser generalizados para todas as equipes ágeis de desenvolvimento de software?
- As equipes ágeis de desenvolvimento de software podem ser comparadas com as equipes ágeis de desenvolvimento de produtos tradicionais?
- Quais as fragilidades dos processos ágeis de acordo com a gestão do conhecimento?

REFERÊNCIAS

ABDULLAH, R.; TALIB, A. M. Knowledge Management System Model in Enhancing Knowledge Facilitation of Software Process Improvement for Software House Organization. 2012 International Conference on Information Retrieval and Knowledge Management, CAMP'12 (2012). **Anais...** . p. 60–63, 2012.

ARMOUR, P. G. The five orders of ignorance. **Communications of the ACM**, v. 43, n. 10, p. 10–17, 2000.

BAHLI, B.; ZEID, E. S. A. The role of knowledge creation in adopting extreme programming model: An empirical study. 2005 International Conference on Information and Communication Technology. **Anais...** . v. 2005, p. 75–87, 2005.

BIAO-WEN, L. The analysis of obstacles and solutions for software enterprises to implement knowledge management. **ICIME 2010 - 2010 2nd IEEE International Conference on Information Management and Engineering**, v. 6, p. 211–214, 2010.

BODEN, A.; AVRAM, G. Bridging knowledge distribution - the role of knowledge brokers in distributed software development teams. **Proceedings of the 2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering, CHASE 2009**, CHASE'09., p. 8–11, 2009.

CHARMAZ, K. Constructing grounded theory: A practical guide through qualitative analysis. London: Sage Publications. 2006.

CHAU, T.; MAURER, F.; MELNIK, G. Knowledge Sharing : Agile Methods vs . Tayloristic Methods. Twelfth IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (2003). **Anais...** . p. 302–307, 2003.

CHEN, D. N.; SHIE, Y. J.; LIANG, T. P. The impact of knowledge diversity on software project team's performance. **Proceedings of the 11th International Conference on Electronic Commerce ICEC 09**, p. 222, 2009.

CHIAVENATO, I. **Introdução à Teoria Geral da Administração.** Rio de Janeiro: Campus, 2003.

COCKBURN, A. **Agile Software Development:** The Cooperative Game. Boston: Addison-Wesley, 2007.

CORRIGAN, J. Augmented intelligence-the new AI-unleashing human capabilities in knowledge work. Proceedings of the 2012 International Conference on Software Engineering. **Anais...** . p.1285–1288, 2012.

DE MASI, D. O ócio criativo. Rio de Janeiro: Sextante, 2000.

DAKHLI, S. B. D.; CHOUIKHA, M. B. The knowledge-gap reduction in software engineering. Proceedings of the 2009 3rd International Conference on Research Challenges in Information Science, RCIS 2009. **Anais...** . p. 287–294, 2009.

DINGSØYR, T.; BJØRNSON, F. O.; SHULL, F. What Do We Know about Knowledge Management? **Journal of Advertising Research**, v. 53, n. 3, p. 238–239, 2013.

DORAIRAJ, S.; NOBLE, J.; MALIK, P. Knowledge management in distributed agile software development. Proceedings - 2012 Agile Conference, Agile 2012. **Anais...** . p.64–73, 2012.

DOTTA, S. *Aprendizagem dialógica em serviços de tutoria pela internet: Estudo de caso de uma tutora em formação em uma disciplina a distância.* 2009. 213 f. Tese (Doutorado em Educação) - Faculdade de Educação, Universidade de São Paulo, São Paulo.

DUCK, J. D. Managing change: the art of balancing. In: **Harvard Business Review on Change**. Boston: Harvard Business School Press, 1998.

ELORANTA, V.; KOSKIMIES, K. Aligning architecture knowledge management with Scrum. **Proceedings of the WICSA/ECSA 2012 …**, p. 112–115, 2012.

FÆGRI, T. E. Improving general knowledge in agile software organizations: Experiences with job rotation in customer support. **Proceedings - 2009 Agile Conference, AGILE 2009**, p. 49–56, 2009.

FÆGRI, T. E.; DYBÅ, T.; DINGSØYR, T. Introducing knowledge redundancy practice in software development: Experiences with job rotation in support work. **Information and Software Technology**, v. 52, n. 10, p. 1118–1132, 2010.

FENG, J. A Knowledge Management Maturity Model and Application. 2006 Technology Management for the Global Future - PICMET 2006 Conference. **Anais...** . v. 3, p. 1251–1255, 2006

FLORES, N.; AGUIAR, A.; SERENO, H. The concept of “ba” applied to software knowledge. **Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering - CHASE 2014**, p. 95–98, 2014.

GARVIN, D. A., NAYAK, P. R., MAIRA, A N., BRAGAR, J L. Aprender a Aprender. In: **HSM Management.** n. 9. São Paulo: julho-agosto 1998. p. 58-64.

GLASER, B. G.; STRAUSS, A. L. The discovery of grounded theory: Strategies for qualitative research. Aldine Transaction. New Brunswick and London. 1967.

GUILLAUME, P. Psicologia da forma. São Paulo: Companhia Editora Nacional. 1960.

HOFSTEDE, G.; NEUIJEN, B.; OHAYV, D. D.; SANDERS, G. Measuring organizational cultures: A qualitative and quantitative study across twenty cases. **Administrative Science Quarterly,** v. 35 n.2, 286-316. 1990.

HOLLOWAY, I. Basic Concepts for Qualitative Research. In: Basic Concepts for Qualitative Research. Oxford: Blackwell Science. 1997.

HUANG, H. C.; SHIH, H. Y.; HSU, S. C. Team structure to accelerate knowledge diffusion: A case study in computer software developer. **5th IEEE International Conference on Management of Innovation and Technology, ICMIT2010**, p. 928–933, 2010.

ISIDRO FILHO, A.; GUIMARÃES, T. D. A. Conhecimento, Aprendizagem E Inovação Em Organizações: Uma Proposta De Articulação Conceitual. **Review of Administration and Innovation**, v. 7, n. 2, p. 127–149, 2010.

KAVITHA, R. K.; IRFAN AHMED, M. S. A Knowledge Management Framework for Agile Software Development Teams. **2011 International Conference on Process Automation, Control and Computing (PACC 2011)**, p. 1-5, 2011.

KOKKONIEMI, J. K. Gathering experience knowledge from iterative software development processes. Proceedings of the Annual Hawaii International Conference on System Sciences. **Anais...** , 41st Annual Hawaii International Conference on System Sciences 2008, HICSS. p. 333, 2008.

LARMAN, C, Agile & Iterative Development: A manager's guide. Boston: Pearson Education, 2004.

LEVY, M.; HAZZAN, O. Knowledge management in practice: The case of agile software development. **2009 ICSE Workshop on Cooperative and Human Aspects on Software Engineering**, p. 60-65, 2009.

LÈVY, P. A Inteligência Coletiva: por uma Antropologia do Ciberespaço. São Paulo: Loyola, 1999.

LOCKE, J. Ensaio acerca do entendimento humano. In: **Os Pensadores.** São Paulo: Nova Cultura, 1999.

MACEDO, N. D. **Iniciação à pesquisa bibliográfica:** guia do estudante para a fundamentação do trabalho de pesquisa. São Paulo: Edições Loyola, 1994

MACLEOD, L.; STOREY, M. A.; BERGEN, A. Code, Camera, Action: How Software Developers Document and Share Program Knowledge Using YouTube. **IEEE International Conference on Program Comprehension**, v. 2015-August, p. 104–114, 2015.

MCGREGOR, D. The Human Side of Enterprise, New York: McGrawHill. 1960.

MELNIK, G.; MAURER, F. Direct verbal communication as a catalyst of agile knowledge sharing. Agile Development Conference, 2004. **Anais...** , Proceedings of the Agile Development Conference. ADC 2004. p.21–32, 2004.

MENOLLI, A.; CUNHA, M. A.; REINEHR, S.; MALUCELLI, A. “Old” theories, “new” technologies: Understanding knowledge sharing and learning in Brazilian software

development companies. **Information and Software Technology**, v. 58, p. 289–303, 2015.

MOHER, D.; LIBERATI, A.; TETZLAFF, J.; ALTMAN, D. G. Preferred Reporting Items for Systematic Reviews and Meta-Analyses: The PRISMA Statement. **Physical Therapy**, v. 89, n. 9, p. 873–880, 2009.

MOTTA, P. R. Transformação organizacional: a teoria e a prática de inovar**.** 5°ed., Rio de Janeiro: Ed. Qualitymark, 2001.

NEVES, F. T.; ROSA, V. N.; CORREIA, A. M. R.; NETO, M. D. C. Knowledge creation and sharing in software development teams using agile methodologies: Key insights affecting their adoption. 6th Iberian Conference on Information Systems and Technologies (CISTI 2011). **Anais...** . p. 1–6, 2011.

NIDHRA, S.; YANAMADALA, M.; AFZAL, W.; TORKAR, R. Knowledge transfer challenges and mitigation strategies in global software development-A systematic literature review and industrial validation. **International Journal of Information Management**, v. 33, n. 2, p. 333–355, 2012.

NONAKA, I.; KONNO, N. The concept of “Ba”: Building foundation for Knowledge Creation. **California Management Review**, v. 40, n. 3, 1998.

NONAKA, I.; TAKEUCHI, H. Criação de conhecimento na empresa: como as empresas japonesas geram a dinâmica da inovação. Rio de Janeiro: Campus, 1997.

OHNO, T. O sistema Toyota de Produção: além da produção em larga escala. Porto Alegre: Bookman, 1997.

O’ROURKE, P. J. A Riqueza das Nações de Adam Smith. Rio de Janeiro: Jorge Zahar Ed., 2008.

OSTROWER, F. Criatividade e Processos de Criação. Rio de Janeiro: Editora Vozes. 1977.

PACHECO, R. C. D. S.; TOSTA, K. C. B. T.; FREIRE, P. D. S. Interdisciplinaridade vista como um processo complexo de construção do conhecimento: uma análise do Programa de Pós-Graduação EGC / UFSC. **R.B.P.** G, v. 7, n. 12, p. 136–159, 2010.

PESSANHA, J. A. M. Vida e Obra do Livro Aristóteles Volume I, Livro IV. Os pensadores. 2004.

PETTICREW, M.; ROBERTS, H. Systematic reviews in the social sciences: A pratical guide. Oxford, United Kingdom: Blackwell Publishing, 2006.

PETTIGREW, A. M. On studying organizacional cultures. Administrative Science Quarterly, v. 24, n.4, p. 570-581, 1979.

PLONKA, L.; SHARP, H.; LINDEN, J. VAN DER; DITTRICH, Y. Knowledge transfer in pair programming: An in-depth analysis. **International Journal of Human-Computer Studies**, v. 73, p. 66–78, 2015.

PRESSMAN, R. S. Engenharia de Software. São Paulo:McGraw-Hill, 2006.

RABELO, J. DE H.; OLIVEIRA, E. C. C. DE; SANTOS, D. V. DOS; et al. Knowledge Management and Organizational Culture in a Software Organization - A Case Study. **2015 IEEE/ACM 8th International Workshop on Cooperative and Human Aspects of Software Engineering**, p. 89–92, 2015.

RAMESH, B. Process Knowledge Management with Traceability. **Software, IEEE**, v. 19, n. 3, p. 50–52, 2002.

REISSMAN, C. Qualitative Studies in Social Work Research. Thousand Oaks: Sage. 1994.

RIES, E. A startup enxuta: como os empreendedores atuais utilizam a inovação contínua para criar empresas extremamente bem-sucedidas. São Paulo: Lua de Papel, 2012.

RUS, I.; LINDVALL, M. Knowledge management in software engineering. **IEEE Software**, v. 19, n. 3, p. 26–38, 2002.

RYAN, S.; O'CONNOR, R. V. Acquiring and Sharing tacit knowledge in software development teams: An empirical study. **Information and Software Technology**, v. 55, n. 9, p. 1614–1624, 2013.

SANTOS, V.; GOLDMAN, A.; FILHO, H.; MARTINS, D.; CORT??S, M. The influence of organizational factors on inter-team knowledge sharing effectiveness in agile environments. **Proceedings of the Annual Hawaii International Conference on System Sciences**, p. 4729–4738, 2014.

SCHREIBER, R. S.; STERN, p. N. Using Grounded Theory in Nursing. Springer Publishing, Broadway, New York, 2001.

SCHWABER, K.; SUTHERLAND, J. Guia do Scrum Um guia definitivo para o Scrum: As regras do jogo. Scrum.org, 2011 Disponível em: <http://www.scrum.org/Portals/0/Documents/Scrum%20Guides/Scrum%20Guide%20-%20Portuguese%20BR.pdf>. 25 de fevereiro 2013

SENGE, P. M. The fifth discipline: the art and practice of the learning organization. New York: Doubleday, 1990.

SOMMERVILLE, I. Engenharia de Sofware. São Paulo: Pearson Addison-Wesley, 2007.

STEIL, A.V. Um Modelo de Aprendizagem Organizacional Baseado na Ampliação de Competências Desenvolvidas em Programas de Capacitação. 2002. 216 f. Tese (Doutorado em Engenharia de Produção) - Universidade Federal de Santa Catarina, Florianópolis.

STRAUSS, A.; CORBIN, J. Grounded theory methodology: An overview. In N. K. Denzin & Y. S. Lincoln, Handbook of qualitative research. Newbury Park, CA: Sage. p. 273–285. 1994.

SUTHERLAND, J. The Scrum Papers: Nut, Bolts, and Origins of an Agile Framework. Scrum, Inc. Cambridge, 2012. Disponível em: <http://jeffsutherland.com/ScrumPapers.pdf>. 04 de junho 2013

TAKEUCHI, H.; NONAKA, I. The new new product development game. **Journal of Product Innovation Management**, v. 3, n. 3, p. 205–206, 1986.

TAKEUCHI, H.; NONAKA, I. Gestão do Conhecimento. Porto Alegre: Bookman, 2004.

VASANTHAPRIYAN, S.; TIAN, J.; XIANG, J. A Survey on Knowledge Management in Software Engineering. **2015 IEEE International Conference on Software Quality, Reliability and Security - Companion (QRS-C)**, p. 237–244, 2015.

VIANA, D.; CONTE, T.; MARCZAK, S.; FERREIRA, R.; SOUZA, C. DE. Knowledge creation and loss within a software organization: An exploratory case study. **Proceedings of the Annual Hawaii International Conference on System Sciences**, v. 2015-March, p. 3980–3989, 2015.

WEGNER, D. M.; GIULIANO, T., HERTEL, P. T. Cognitive interdependence in close relationships. En W. J. Ickes. **Compatible and incompatible relationships.** New York: Springer-Verlag. p. 253-276. 1985.

YANYAN, Z. Y. Z.; RENZUO, X. R. X. The Basic Research of Human Factor Analysis Based on Knowledge in Software Engineering. **2008 International Conference on Computer Science and Software Engineering**, CSSE ’08., v. 5, p. 1302–1305, 2008.

YE, Y. Dimensions and forms of knowledge collaboration in software development. Proceedings - Asia-Pacific Software Engineering Conference, APSEC. **Anais...** . v. 2005, p.805–812, 2005.

YE, Y.; YAMAMOTO, Y.; KISHIDA, K. Dynamic Community: A New Conceptual Framework for Supporting Knowledge Collaboration in Software Development. 11th Asia-Pacific Software Engineering Conference (2004). **Anais...** . p.472–481, 2004. IEEE Computer Society. 2004.

YE, Y. Supporting Software Development as Knowledge-Intensive and Collaborative Activity. 2006 International Workshop on Workshop on Interdisciplinary Software Engineering Research WISER 06. **Anais...** , WISER ’06. p.15–21, 2006.

ZIERIS, F.; PRECHELT, L. On Knowledge Transfer Skill in Pair Programming. **Proceedings of the 8th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement**, p. 11:1–11:10, 2014.

ZIERIS, F.; PRECHELT, L. Observations on Knowledge Transfer of Professional Software Developers during Pair Programming. 2016 IEEE/ACM 38th IEEE International Conference on Software Engineering Companion. **Anais...** ,p. 242-250. 2016.

ANEXOS

ANEXO I – Parecer consubstanciado do CEP

COMITÊ DE ÉTICA EM PESQUISA DA UNICAMP - CAMPUS CAMPINAS

PARECER CONSUBSTANCIADO DO CEP

DADOS DO PROJETO DE PESQUISA

Título da Pesquisa: Gestão do Conhecimento no Desenvolvimento Ágil de Software: Técnicas e Fatores que Promovem a Criação de Conhecimento

Pesquisador: Eduardo Sakai

Área Temática:

Versão: 2

CAAE: 43519215.9.0000.5404

Instituição Proponente: Faculdade de Engenharia Elétrica e de Computação

Patrocinador Principal: Financiamento Próprio

DADOS DO PARECER

Número do Parecer: 1.072.879

Data da Relatoria: 18/05/2015

Apresentação do Projeto:

"Esta pesquisa tem como tema a “gestão do conhecimento no desenvolvimento ágil de software” e busca responder a seguinte questão de pesquisa: “Quais são as técnicas e os fatores capacitadores e como eles influenciam no processo de criação de conhecimento no desenvolvimento de novos softwares, considerando o uso de técnicas das metodologias ágeis?” Foi feita uma revisão de literatura utilizando-se do framework PRISMA, para tentar compreender o estado da arte da gestão do conhecimento no desenvolvimento de software. Como resultados dessa revisão de literatura foram definidos os principais conceitos e práticas que os estudiosos têm encontrado, reunidos sobre a classificação da dimensão epistemológica (conhecimento tácito e conhecimento explícito) e da dimensão ontológica (individual, grupo de desenvolvimento e organizacional), além das abordagens de desenvolvimento de software (tradicionais e ágeis) e do espaço e da cultura. A partir da revisão de literatura, as técnicas utilizadas nas metodologias ágeis foram identificadas e classificadas de acordo com as dimensões epistemológica e ontológica e também de acordo com o modelo SECI (Socialização, Externalização, Combinação e Internalização). Após os resultados obtidos com a revisão de literatura, evidenciou-se a necessidade de uma nova teoria que englobe técnicas e fatores que auxiliam a criação de conhecimento no desenvolvimento de software. A metodologia adotada para a realização desta pesquisa será o estudo de caso a fim de definir quais

fatores capacitam e compreender como eles influenciam no processo de criação de conhecimento no desenvolvimento de software. E então será proposto uma nova teoria para a gestão do conhecimento no desenvolvimento ágil de softwares."

Objetivo da Pesquisa:

“O objetivo geral deste estudo de caso será tentar descobrir quais técnicas e fatores capacitam e como eles influenciam no processo de criação de conhecimento no desenvolvimento ágil de software.

A presente pesquisa tem os seguintes objetivos específicos: Entender como se dá a gestão do conhecimento no desenvolvimento ágil de software; Identificar e classificar as técnicas das metodologias ágeis que contribuem para a criação de conhecimento no desenvolvimento de novos sistemas de software; Identificar e classificar os fatores capacitadores que contribuem para a criação de conhecimento no desenvolvimento de novos sistemas de software; Propor uma nova teoria sobre a criação de conhecimento no desenvolvimento de software.”

Avaliação dos Riscos e Benefícios:

“Riscos: Algumas informações que serão coletadas através das entrevistas serão sobre experiências pessoais dos participantes, assim o participante poderá escolher não responder a quaisquer perguntas que possam fazer-los sentirem incomodados. Nenhum dos procedimentos que serão utilizados ofereceram riscos a dignidade do participante.

Benefícios: As entrevistas ajudarão a entender e melhorar os processos de desenvolvimento ágil de software, mas não será, necessariamente, para benefício direto dos participantes. Entretanto, beneficiará todas as equipes de desenvolvimento de software que privilegiam a gestão do conhecimento.”

Comentários e Considerações sobre a Pesquisa:

Estudo qualitativo, com amostra prevista de 09 indivíduos integrantes ou gestores de grupos de desenvolvimento ágil de software cujas metodologias utilizadas sejam o Scrum e/ou Extreme Programming e que tenham desenvolvido softwares inovadores nos últimos anos.

O pesquisador pretende utilizar entrevista estruturada como técnica de coleta de dados, sendo estas gravadas em áudio perante a autorização dos indivíduos de pesquisa.

Não há riscos previsíveis, e não há benefícios diretos aos participantes da pesquisa. Com benefício global, o pesquisador descreve que a pesquisa poderá ajudar a entender e melhorar os processos de desenvolvimento ágil de software.

O pesquisador não esclarece os critérios de inclusão e exclusão da pesquisa, não está claro, por

exemplo, se menores de 18 anos (grupo vulnerável) participarão da pesquisa. O pesquisador também não esclarece qual será o destino dos áudios após o término da pesquisa.

Considerações sobre os Termos de apresentação obrigatória:

1.Protocolo de Pesquisa gerado pela Plataforma Brasil, com ausência de preenchimento dos itens critérios de inclusão ou exclusão. O item orçamento indica financiamento pelo próprio pesquisador com custo estimado em R$0,00 e coleta de dados com início em 02/02/2015, por tanto, anterior a esta relatoria; 2. Folha de Rosto preenchida e assinada pelo pesquisador responsável e pelo responsável legal pela instituição;

3.Termo de Consentimento Livre e Esclarecido (TCLE) anexado à Plataforma Brasil; 4.Projeto de Pesquisa, com finalidade de pesquisa de mestrado, anexado à Plataforma Brasil;

Recomendações:

Conclusões ou Pendências e Lista de Inadequações:

1. De acordo a Resolução CNS no466 de 2012, item XI.2.a. cabe ao pesquisador apresentar o protocolo devidamente instruído ao CEP ou à CONEP, aguardando a decisão de aprovação ética, antes de iniciar a pesquisa. Nas informações fornecidas pelo pesquisador no documento intitulado "PB_INFORMAÇÕES_BÁSICAS_DO_PROJETO_446097.pdf", anexado na Plataforma Brasil em 11/03/2015 às 18h15, o inicio da coleta de dados está previsto para 02/02/2015, por tanto, anterior a esta relatoria. Solicitamos ao pesquisador esclarecer se a coleta de dados já foi iniciada. Caso contrário, solicitamos ao pesquisador adequar o cronograma proposto;

RESPOSTA: Em carta anexada à Plataforma Brasil, o pesquisador esclarece que não iniciou a coleta de dados, que a mesma acontecerá somente após a aprovação do CEP.

ANÁLISE: Pendência atendida.

2.Nas informações fornecidas pelo pesquisador no documento intitulado "PB_INFORMAÇÕES_BÁSICAS_DO_PROJETO_446097.pdf", anexado na Plataforma Brasil em 11/03/2015 às 18h15, o pesquisador descreve que a pesquisa terá custo R$0,00. Solicitamos ao pesquisador rever este item, incluindo despesas previstas com material gráfico, despesas com deslocamentos para coleta de dados, etc.

RESPOSTA: O pesquisador forneceu detalhes de seu orçamento financeiro, estimando um total de R$ 118,70.

ANÁLISE: Pendência atendida.

COMITÊ DE ÉTICA EM PESQUISA DA UNICAMP - CAMPUS CAMPINAS

3.Sobre as informações fornecidas pelo pesquisador no documento intitulado "PB_INFORMAÇÕES_BÁSICAS_DO_PROJETO_446097.pdf", anexado na Plataforma Brasil em 11/03/2015 às 18h15, solicitamos ao pesquisador incluir critérios de inclusão (Por exemplo: indivíduos de ambos os sexos, maiores de 18 anos, integrante de equipe de desenvolvimento de software que utilize Scrum e/ou Extreme Programmimg, etc). Lembramos que a assinatura do Termo de Consentimento Livre e Esclarecido (TCLE) não se caracteriza como critério de inclusão;
RESPOSTA: O pesquisador esclareceu os critérios de inclusão: "Indivíduos de ambos os sexos, maiores de 18 anos, integrante de equipe de desenvolvimento de software que utilizem metodologias ágeis Scrum e/ou Extreme Programmimg ou alguma de suas variações/sínteses".
ANÁLISE: Pendência atendida.

4. Sobre as informações fornecidas pelo pesquisador no documento intitulado "PB_INFORMAÇÕES_BÁSICAS_DO_PROJETO_446097.pdf", anexado na Plataforma Brasil em 11/03/2015 às 18h15, solicitamos ao pesquisador incluir critérios de exclusão. Os critérios de exclusão são condições que impeçam os voluntários de participarem da pesquisa, ainda que preencham todos os critérios de inclusão. Não se aplica aos sujeitos que não queiram participar da pesquisa ou que não satisfaçam os critérios de inclusão;
RESPOSTA: O pesquisador esclareceu os critérios de exclusão: "Indivíduos que estão em fase de treinamentos."
ANÁLISE: Pendência atendida.

5. Solicitamos ao pesquisador esclarecer os seguintes tópicos através de uma carta anexada à Plataforma Brasil:
5.a. Sobre o destino dos áudios gravados durante as sessões de entrevistas: Serão armazenados? Se sim, por quanto tempo? Como serão descartados? Pretende-se usar em projetos futuros ou será utilizado exclusivamente para este projeto?
RESPOSTA: O pesquisador esclareceu sobre o uso de audios em sua pesquisa através de carta anexada à Plataforma Brasil.
ANÁLISE: Pendência atendida.

5.b. Sobre as sessões de entrevista: as entrevistas acontecerão em qual local? Acontecerão em dia/horário de trabalho dos voluntários de pesquisa? Como será feita a autorização oficial da empresa participante para que a pesquisa seja realizada?

COMITÊ DE ÉTICA EM PESQUISA DA UNICAMP - CAMPUS CAMPINAS

RESPOSTA: O pesquisador esclareceu sobre as sessões de entrevistas, através de carta anexada à Plataforma Brasil.
ANÁLISE: Pendência atendida.

6. Sobre o Termo de Consentimento Livre e Esclarecido (TCLE), referente ao arquivo “esakai_tcle.pdf” postado na Plataforma Brasil em 09/01/15 às 01h33:
6.a. De acordo com o item IV.3.a. da Resolução CNS 466/2012, o TCLE deve conter obrigatória a justificativa para a realização da pesquisa. Solicitamos ao pesquisador adequar este item;
6.b. No item “procedimentos”, solicitamos ao pesquisador esclarecer o local da entrevista e se a mesma ocorrerá durante o expediente de trabalho do voluntário;
6.c. No item “Desconfortos e riscos”: a Resolução CNS 196/96 foi substituída pela Resolução CNS 446/2012. Sugerimos ao pesquisador, substituir a frase “Os procedimentos adotados nesta pesquisa obedecem aos Critérios da Ética em Pesquisa com Seres Humanos conforme Resolução no. 196/96 do Conselho Nacional de Saúde. Nenhum dos procedimentos usados oferece riscos à sua dignidade” por “Não existem riscos previsíveis. Os procedimentos adotados nesta pesquisa obedecem à Resolução CNS 446/2012”;
6.d. Sugerimos ao pesquisador adequar o conteúdo do item “benefícios”, deixando claro que não haverá benefícios diretos aos participantes da pesquisa: “Sua entrevista PODERÁ ajudar a entender e melhorar os processos de desenvolvimento ágil de software, todavia não haverá benefícios diretos à você. Participando deste estudo você fornecerá (...)”;
6.e. De acordo com o item IV.3.g. da Resolução 466/2012, solicitamos ao pesquisador esclarecer, em linguagem clara, se haverá ressarcimento de despesas (por exemplo, transporte, alimentação, etc), qual o valor e de que modo será feito. Se não estiver previsto ressarcimento, sugerimos ao pesquisador, incluir no item “Ressarcimento” do TCLE a seguinte frase: “não está previsto qualquer tipo de ressarcimento pela sua participação na pesquisa”.
RESPOSTA: O pesquisador adequou seu TCLE, de acordo com o solicitado no parecer consubstanciado 1.039.832, emitido em 28/04/2015.
ANÁLISE: Pendências atendidas.

Situação do Parecer:
Aprovado

Necessita Apreciação da CONEP:
Não

Considerações Finais a critério do CEP:

- O sujeito de pesquisa deve receber uma via do Termo de Consentimento Livre e Esclarecido, na íntegra, por ele assinado.

- O sujeito da pesquisa tem a liberdade de recusar-se a participar ou de retirar seu consentimento em qualquer fase da pesquisa, sem penalização alguma e sem prejuízo ao seu cuidado.

- O pesquisador deve desenvolver a pesquisa conforme delineada no protocolo aprovado. Se o pesquisador considerar a descontinuação do estudo, esta deve ser justificada e somente ser realizada após análise das razões da descontinuidade pelo CEP que o aprovou. O pesquisador deve aguardar o parecer do CEP quanto à descontinuação, exceto quando perceber risco ou dano não previsto ao sujeito participante ou quando constatar a superioridade de uma estratégia diagnóstica ou terapêutica oferecida a um dos grupos da pesquisa, isto é, somente em caso de necessidade de ação imediata com intuito de proteger os participantes.

- O CEP deve ser informado de todos os efeitos adversos ou fatos relevantes que alterem o curso normal do estudo. É papel do pesquisador assegurar medidas imediatas adequadas frente a evento adverso grave ocorrido (mesmo que tenha sido em outro centro) e enviar notificação ao CEP e à Agência Nacional de Vigilância Sanitária – ANVISA – junto com seu posicionamento.

- Eventuais modificações ou emendas ao protocolo devem ser apresentadas ao CEP de forma clara e sucinta, identificando a parte do protocolo a ser modificada e suas justificativas. Em caso de projetos do Grupo I ou II apresentados anteriormente à ANVISA, o pesquisador ou patrocinador deve enviá-las também à mesma, junto com o parecer aprovatório do CEP, para serem juntadas ao protocolo inicial.

- Relatórios parciais e final devem ser apresentados ao CEP, inicialmente seis meses após a data deste parecer de aprovação e ao término do estudo.

COMITÊ DE ÉTICA EM PESQUISA DA UNICAMP - CAMPUS CAMPINAS

Continuação do Parecer: 1.072.879

CAMPINAS, 21 de Maio de 2015

Assinado por:
Renata Maria dos Santos Celeghini
(Coordenador)

ANEXO II – Termo de Consentimento Livre e Esclarecido

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

GESTÃO DO CONHECIMENTO NO DESENVOLVIMENTO ÁGIL DE SOFTWARE:

TÉCNICAS E FATORES QUE PROMOVEM A CRIAÇÃO DE CONHECIMENTO

Eduardo Sakai – Orientador Prof. Dr. Ivan Ricarte

Número do CAAE: 43519215.9.0000.5404

Você está sendo convidado a participar como voluntário de um estudo. Este documento, chamado Termo de Consentimento Livre e Esclarecido, visa assegurar seus direitos e deveres como participante e é elaborado em duas vias, uma que deverá ficar com você e outra com o pesquisador.

Por favor, leia com atenção e calma, aproveitando para esclarecer suas dúvidas. Se houver perguntas antes ou mesmo depois de assiná-lo, você poderá esclarecê-las com o pesquisador. Se preferir, pode levar para casa e consultar seus familiares ou outras pessoas antes de decidir participar. Se você não quiser participar ou retirar sua autorização, a qualquer momento, não haverá nenhum tipo de penalização ou prejuízo.

Justificativa:

Em uma economia na qual a única certeza é a incerteza, apenas o conhecimento se apresenta com uma fonte segura de vantagem competitiva (NONAKA, 1991). Nesse contexto, organizações focam na gestão do conhecimento e do capital intelectual como seu mais importante patrimônio; ativos intangíveis passam a ter uma abordagem voltada para a aprendizagem organizacional (CHIAVENATO, 2003).

As empresas de software apresentam grande utilização do capital intelectual dos indivíduos envolvidos na criação de novos software, evidenciando a necessidade e a importância de gerenciar o conhecimento dos profissionais envolvidos nessa criação.

Objetivos:

Este projeto tem por objetivo definir quais técnicas e fatores capacitam e compreender como técnicas e fatores influenciam no processo de criação de conhecimento no desenvolvimento ágil de software.

Procedimentos:

Se você decidir integrar este estudo, você participará de uma entrevista em grupo e/ou de uma entrevista individual que durará aproximadamente 1 hora e 30 minutos. As entrevistas ocorreram no seu horário e local de trabalho.

Todas as entrevistas serão gravadas em áudio, em formato digital m4a. Os áudios serão ouvidos por mim e por meu orientador. Os arquivos serão utilizados somente para coleta de dados. Se você não quiser ser gravado em áudio, você não poderá participar deste estudo.

Desconfortos e riscos:

Você pode achar que determinadas perguntas incomodam a você, porque algumas das informações que coletamos são sobre suas experiências pessoais. Assim você pode escolher não responder quaisquer perguntas que o faça se sentir incomodado.

Não existem riscos previsíveis. Os procedimentos adotados nesta pesquisa obedecem à Resolução CNS 446/2012.

Benefícios:

Sua entrevista poderá ajudar a entender e melhorar os processos de desenvolvimento ágil de software, todavia não haverá benefícios diretos à você. Participando deste estudo você fornecerá mais informações sobre o lugar e relevância desses escritos para própria instituição em questão.

Sigilo e privacidade:

Você tem a garantia de que sua identidade será mantida em sigilo e nenhuma informação será dada a outras pessoas que não façam parte da equipe de pesquisadores. Na divulgação dos resultados desse estudo, seu nome não será citado.

Ressarcimento:

Não está previsto qualquer tipo de ressarcimento pela sua participação na pesquisa.

A sua participação não acarretará nenhum tipo de despesa, bem como nada será pago a sua participação.

Contato:

Em caso de dúvidas sobre o estudo, você poderá entrar em contato com **Eduardo Sakai** do Departamento de Engenharia de Computação e Automação Industrial (DCA) da Faculdade de Engenharia Elétrica e de Computação (FEEC) da Universidade Estadual de Campinas (UNICAMP), localizado na Av. Albert Einstein, 400 - Cidade Universitária Zeferino Vaz - Distrito Barão Geraldo – Campinas - SP Brasil ou pelo email: dusakai@gmail.com.

Em caso de denúncias ou reclamações sobre sua participação no estudo, você pode entrar em contato com a secretaria do Comitê de Ética em Pesquisa (CEP): Rua: Tessália Vieira de Camargo, 126; CEP 13083-887 Campinas – SP; telefone (19) 3521-8936; fax (19) 3521-7187; e-mail: cep@fcm.unicamp. br

Consentimento livre e esclarecido:

Após ter sido esclarecimento sobre a natureza da pesquisa, seus objetivos, métodos, benefícios previstos, potenciais riscos e o incômodo que esta possa acarretar, aceito participar:

Nome do(a) participante: _____

_____ Data: ____/_____/_____.

(Assinatura do participante)

Responsabilidade do Pesquisador:

Asseguro ter cumprido as exigências da resolução 466/2012 CNS/MS e complementares na elaboração do protocolo e na obtenção deste Termo de Consentimento Livre e Esclarecido. Asseguro, também, ter explicado e fornecido uma cópia deste documento ao participante. Informo que o estudo foi aprovado pelo CEP perante o qual o projeto foi apresentado. Comprometo-me a utilizar o material e os dados obtidos nesta pesquisa exclusivamente para as finalidades previstas neste documento ou conforme o consentimento dado pelo participante.

_____ Data: ____/_____/_____.

(Assinatura do pesquisador)

ANEXO III – Roteiro semiestruturado para entrevistas

Bloco I – Perfil do ambiente de estudo

1. Qual sua escolaridade?
2. Qual seu cargo ou função na organização?
3. Há quanto tempo você trabalha na organização?
4. Qual é a configuração estrutural da organização?
5. Qual a principal orientação da organização?
6. A organização tem quantos funcionários?
7. A organização está dividida em departamentos, divisões ou grupos? Se sim, responda 8 e 9.
8. Quais departamentos, divisões ou grupos compõem a organização?
9. Como são as relações de interdependência entre eles?

Bloco II – Condições capacitadoras para a criação do conhecimento

10. Uma importante técnica para estabelecer a intenção organizacional é a divulgação da missão, visão e valores. A empresa divulga satisfatoriamente sua missão, visão e valores para seus funcionários? Se sim, responda as questões 11, 12 e 13.
11. Qual o meio em que é divulgado?
12. Você saberia dizer qual é a missão, visão e valores?
13. Os projetos de desenvolvimento de novos sistemas de software incorporam a missão, a visão e os valores da empresa estabelecido pela alta gerência?
14. O que se pode afirmar dos membros da equipe de desenvolvimento quanto a autonomia individual e auto-organização da equipe?
15. A equipe de desenvolvimento tem autonomia (em nível de equipe) durante o processo de criação de novos sistemas de software?
16. Quais condições ou desafios são atribuídos ao grupo de desenvolvimento, visando aumentar a criatividade durante o processo de criação de novos sistemas de software?
17. Em determinadas situações, para uma melhor compreensão de problema ou desafio os membros da equipe utilizam-se de quais técnicas de linguagens ou ferramentas?

Bloco III – Técnicas do desenvolvimento ágil que favorecem a criação de conhecimento

18. Qual metodologia ágil é utilizada pelo grupo de desenvolvimento?

As questões 19 a 23 são para equipes que utilizam *Scrum*.

19. O *backlog* do produto tem uma visão do produto compatível com a visão da empresa? A sua autonomia permanece apenas no âmbito técnico?
20. Quais reuniões do s*crum* são feitas pelo grupo de desenvolvimento?
21. Como são as reuniões do *scrum*? A do planejamento e do review?
22. Qual o papel do *Scrum Master* dentro do grupo de desenvolvimento?
23. E quanto ao papel exercido pelo *Scrum Master* com relação a intermediação entre o grupo de desenvolvimento e a alta gerência?

As questões 24 a 28 são para equipes que utilizam Extreme Programming (XP).

24. A programação é feita em par? Se sim, responda 25 e 26.
25. Qual o papel de cada um na programação?
26. Eles trabalham 100% do tempo juntos?
27. O XP incentiva o uso de metáfora para descrever situações. É comumente utilizado a metáfora como ferramenta de comunicação entre o grupo de desenvolvimento?
28. Como se dá a dinâmica de utilização dos cartões CRC?
29. Algumas técnicas não provenientes do Scrum ou do XP normalmente são utilizadas para complementar a metodologia, como o *Planning Poker*, o Kanban (cartões) ou o *Test Driven Development* (TDD). O seu grupo faz uso de alguma técnica complementar? Se sim responda a questão 30.
30. Como se dá na prática o uso dessa(s) técnica(s)?
31. Quais técnicas do desenvolvimento ágil se mostraram favoráveis e tornaram-se essenciais no desenvolvimento de novos sistemas de software?

Bloco IV – Processo de criação do conhecimento organizacional

32. Em que situações os membros do grupo de desenvolvimento compartilham suas emoções, sentimentos e modelos mentais?
33. Como os membros das equipes de desenvolvimento de software trocam e compartilham seus conhecimentos entre si?

34. De que maneira os membros do grupo de desenvolvimento absorvem, internalizam ou incorporam conhecimento?
35. Como os membros do grupo ficam sabendo dos conhecimentos prévios uns dos outros para saber quem sabe o quê?
36. Durante o desenvolvimento de novos software, em quais situações surgem novas ideias?
37. Quais ferramentas de TI são usadas para a colaboração/compartilhamento dos conhecimentos dos funcionários?
38. Existem diversas formas que a empresa pode utilizar para descobrir novas oportunidades e transformar essas oportunidades em novos conceitos. Como a organização procura identificar essas novas oportunidades para o desenvolvimento de novos software?
39. A empresa incentiva ou fornece meios para o diálogo contínuo e a reflexão coletiva?
40. A organização incentiva ou fornece meios para valorizar os *insights* dos funcionários?
41. A organização estimula a participação de especialistas nos projetos de desenvolvimento?
42. Existe uma equipe exclusiva para o desenvolvimento de novos software (departamento de P&D)? Se sim, como ela é organizada?
43. A organização interage com agentes do meio externo (clientes, fornecedores, parceiros ou concorrentes) com o intuito de importar conhecimento? Se sim, responder a questão 43.
44. Como se dá este processo?
45. Para incentivar a criatividade na criação de novos software, o que faz a empresa?
46. Durante o processo de desenvolvimento de novos software existe uma colaboração coletiva de troca de experiências, criação de ideias e resolução de problemas de diferentes áreas funcionais? Se sim, responder a questão 47.
47. Como se dá esse processo?
48. A organização procura estimular o diálogo e a participação de todos (mesmo fora do grupo de desenvolvimento) a fim de compartilharem suas experiências, ideias e sugestões que possam servir para a criação de novos conceitos?
49. Quais são os principais fatores relacionados ao sucesso da organização?
50. Como a empresa procura filtrar, escolher e validar os novos conceitos criados?

ANEXO IV – Respostas das Entrevistas

ENTREVISTA 1

(Entrevista 1, questão 13)

Os projetos são direcionados pelo que o mercado está demandando e isso vem de diversas fontes: nossos próprios consultores que trabalham na linha de frente, avaliação de concorrentes, avaliação de mercado, esse tipo de coisa que direciona o que vai ser colocado no mercado.

(Entrevista 1, questão 15)

Dentro das atividades que eles vão desenvolver eles têm autonomia para tomar as melhores decisões do ponto de vista deles. Eles não precisam validar todas as soluções que eles querem implementar com o gestor ou gerente. Isso é importante porque oferece maior agilidade e um senso maior de *ownership* para os membros da equipe. Eles se sentem mais donos daquilo que eles estão fazendo se foi ele que tomou aquela decisão.

(Entrevista 1, questão 16)

O clube de autores tem um conjunto de temas que é definido pela equipe de marketing que são temas nos quais produzimos conteúdo e entregamos o conteúdo para o mercado. Dentro daquele conjunto de temas esse grupo de autores se comprometem a escrever sobre um determinado tema. [...] os autores que produzirem os melhores conteúdos no mês são premiados com um brinde ao final do mês.

(Entrevista 1, questão 18)

O Kanban é muito parecido com o scrum. Ele tem conceitos de *Cards* e status dos *cards*. Os cards representam as atividades que precisam ser feitas. Esses c*ards* têm um ciclo de vida: nascem no *Full Backlog* e vão transitando entre diversos status. Do *Full Backlog* eles vão para o "*This Week*", que são o conjunto de c*ards* que precisam ser feitos naquela semana, esse conjunto de c*ards* são discutidos no início da semana como eles serão feitos, características técni-

cas, o que precisa ser feito em cada um. Depois eles vão passando para *Doing* para, *Blocked* se houver algum problema, *Waiting Test* para passar para parte de QA e, por fim, ele fica aguardando a geração de *Building* e vai para *Staging*, que é a homologação. Depois ele é fechado na *Release*.

Temos *Daily Meeting*, assim como scrum, todos os dias fazemos reuniões rápidas de 15 a 30 minutos, para falar: o que está fazendo? quais as dificuldades? e, o que vai fazer em seguida?

Usamos um gráfico para acompanhar o *Cumulative Flow Diagram* que mostra a evolução dos *cards* durante o período de entrega de uma *release*. Nossas entregas são de três releases por ano, esse *Cumulative Flow Diagram* armazena dados de quantos *cards* estão em cada estado por dia durante quatro meses até uma *release* ser entregue. Com eles sabemos quantos *cards* são implementados por dia, tempo que demora para a implementação dos *cards*, quanto c*ards* ficam bloqueados e média de *cards* bloqueados por dia. Tem uma série de informações que conseguimos extrair desse gráfico, defeitos pro exemplo.

O kanban permite que atividades entre em *Doing* para fazer naquela semana e essa lista não é fechada, ela é simplesmente uma priorização do Full Backlog para aquele conjunto de Cards, mas nada impede que, por exemplo, um defeito que surgiu naquela semana entre para ser feito naquela semana, então ele não tem um conjunto fechado, diferente do scrum que dentro daquelas atividades programadas pro sprint fechado e não poderia entrar outra atividade para aquela semana. Esse foi o principal motivo para gente migrar do scrum para o kanban.

(Entrevista 1, questão 23)

O *Product Owner* realiza um meio-termo entre a pesquisa de mercado através de análise de concorrentes e análise de clientes para estabelecer o que vai ser listado no *full backlog*.

(Entrevista 1, questão 30)

É feito teste unitário durante o desenvolvimento, todos serviços implementados pressupõe um teste unitário para ele. Existe o teste unitário na entrega daquela atividade. A gente usa uma

ferramenta chamada sonar para medir a cobertura do teste unitário e acompanhar o percentual de sucesso dos testes unitários.

Utilizamos o quadro para arrastar os *cards* que ficam disponíveis para todos verem e acompanharem o processo em tempo real.

Fazemos uma reunião de planejamento na segunda-feira das atividades da semana.

(Entrevista 1, questão 31)

Utilizamos o quadro, para arrastar os Cards que fica disponível para todos verem e acompanharem o processo em tempo real.

(Entrevista 1, questão 32)

Temos os momentos fixos de compartilhamento inclusive de sentimos, de angustias que são as *daily meetings*, onde as pessoas expressam seus impedimentos e limitações, são momentos onde a pessoa se abre mais.

Existe um momento de *brainstorming*, de compartilhamento de ideias que é o momento de planning que é no início da semana onde a gente define as atividades da semana.

(Entrevista 1, questão 34)

Participamos de uma série de eventos de mercado, como o JavaOne, QCon, DevCamp, todos esses eventos técnicos a gente participa e incentivamos que as pessoas submetam propostas de apresentações. […] e desses eventos técnicos, eles fazem um resumo do que aprenderam, das coisas mais legais para replicar internamente. Chamamos esse evento de "*I Know How*", que é um evento interno e toda a empresa é convidada para compartilhamento das coisas mais legais que ele viu no evento.

(Entrevista 1, questão 35)

Através do entrosamento da equipe. Não utilizamos nenhuma técnica ou ferramenta para saber o conhecimento prévio dos novos funcionários.

(Entrevista 1, questão 36)

O principal momento de surgimento de ideias é durante as sessões de *brainstorming*. Tanto no *brainstorming* do planejamento da semana, que acontece segunda-feira, quanto no *brainstorming* de priorização das atividades que vão começar na segunda-feira.

(Entrevista 1, questão 37)

As ferramentas do time são o Slack que arquiva tudo. O Dropbox para compartilhamento de arquivos e o Trello que é a ferramenta de acompanhamento dos Cards.

(Entrevista 1, questão 38)

A equipe de serviço, os consultores que ficam dentro de clientes têm um papel mais importante e a análise de mercado que é feita por várias pessoas da empresa. Qualquer um consegue compartilhar o conhecimento de mercado de concorrência de análise de consultoria tipo Gartner, Forrester mas a responsabilidade em última instância a responsabilidade é do *Product Owner*. Ele deve entender o que o mercado está demandando.

(Entrevista 1, questão 39)

Celebrações de entrega de *releases* e toda equipe comemora, com almoço ou *hapy hour* para integração. Também tem as confraternizações da empresa, festa junina, recorrentemente a confraternização mensal de status geral da empresa, que é a WGO (W*hat's Going On*) e sempre depois dessa apresentação tem um *hapy hour*. Também temos o Nijas no Boteco, toda última quinta-feira do mês, fazemos em cada uma das bases e fazemos aqui mesmo. Temos um

barril de chopp e compramos salgados na padaria e nas outras bases cada um decide como vai fazer.

(Entrevista 1, questão 41)

O *Product Owner* conversa com o cliente para pegar o *feedback*.

(Entrevista 1, questão 47)

Por exemplo, se temos um problema de *big data*, vamos chamar o consultor que tem um maior conhecimento nesse assunto para conversar e trocar uma ideia para chegar na melhor solução.

(Entrevista 1, questão 49)

A gente associa isso a uma equipe muito especializada. Temos poucas pessoas inexperientes na equipe. Contamos com pessoas que tem bastante capacidade e iniciativa. Não são muitas pessoas mas são pessoas bem qualificadas.

(Entrevista 1, questão 50)

Muitas vezes o cliente tem uma maior relevância na definição da funcionalidade que vai entrar. Se for uma demanda de cliente eu preciso atender o quanto antes. Preciso deixar o cliente feliz no final das contas. Dessa lista de coisas que entraram, de clientes e mercado, é feita uma discussão interna entre os gestores da empresa pegando o *feedback* de todos para dentre aquilo priorizar onde vai ser feito e investimento.

Outra coisa que auxilia nessa escolha é a análise de concorrentes. Com a análise de concorrentes identificamos onde estão nossos *gaps* e nossas forças com relação aos concorrentes. Onde foi identificado *gaps* devemos fazer o *sketch-up*, alcançá-los e onde foram identificados as forças a gente precisa investir para que isso se torne um diferencial.

ENTREVISTA 2

(Entrevista 2, questão 10)

A empresa é muito transparente. Todo mês tem uma reunião chamada WGO (*What's Going On*). Que eles explicam o que está acontecendo, assim: quanto faturou, quais são as metas e se elas foram atingidas naquele mês, se foram atingidas quanto de faturamento, se não foram eles falam quanto faltam para atingir. Tudo isso é muito bem explicado e de forma muito transparente. Não tem nenhum dado que eles ocultam, muito pelo contrário. Eles colocam nossa meta, agente atingiu essa aqui, essa aqui a gente não bateu ainda, faturamos tanto, esperamos faturar tanto, até agora não chegamos aqui, esse mês passou nossa meta, então é bem transparente.

(Entrevista 2, questão 11)

Tem um *board* do Trello, um aplicativo na web, e lá tem um *roadmap* que você consegue saber quais serão as próximas coisas a serem feitas ou o que já foi implementado.

(Entrevista 2, questão 13)

Acho que é a primeira cia que trabalho que se importa com isso, em trabalhar direito, não fazer qualquer coisa, não. Eu vejo evolução constante e mesmo que o trabalho seja maior ele é feito para poder entregar uma coisa de qualidade. É bem legal porque não apaga fogo apenas.

(Entrevista 2, questão 14)

Então na verdade você nem precisa chegar alguém para te dizer como você vai fazer, você já tem um padrão a ser seguido e isso leva todo mundo num mesmo sentido. Você tem autonomia para dar sugestões para sugerir modificações, mas tudo dentro desse padrão.

O líder é uma pessoa muito inteligente e dá muita liberdade para gente. As vezes ele precisa que aquilo seja entregue mais rápido, uma funcionalidade específica e dependendo como está ele pode pegar e entregar para uma pessoa específica: “você vai pegar isso aqui para ajudar a gente nisso”, ou então tem semanas que estão mais tranquilas que fala assim: “as tarefas da semana são essas, podem pegar os cards que cada um quer fazer”. Eu que estou há pouco tempo ele fala pega esse aqui que é mais tranquilo, ou ele fala pega esse aqui que vai exigir um pouco mais de você.

(Entrevista 2, questão 15)

A equipe decide, eu acho que é 90%. 10% dependendo como for deve ser seguido o *roadmap.* Se isso não impacta no *roadmap* eu acho que tudo bem. Por exemplo a gente está fazendo testes com ferramentas de monitoramento. Eles estão testando vários tipos de ferramenta e vendo qual melhor se adéqua. Qual melhor se comporta, e tudo mais não tem muita regra. O que a gente não pode fazer, por exemplo, é vamos mudar do java que é nossa linguagem nativa desse sistema e vamos para outra. Isso não vai acontecer, a menos que isso venha da diretoria. Mas para outras coisas sim, a gente tem liberdade total. Lógico, isso tem que ser comunicado ao gerente, ele tem que aprovar e tudo mais.

(Entrevista 2, questão 16)

Na verdade, temos o “*I wana do*”, essa plataforma ele te obriga a dizer o que você faz e você pode ganhar pontos por aquilo que você fez, desde ah essa semana eu trouxe um café diferente para o pessoal ou essa semana eu ajudei a resolver um problemão num cliente que já estava há duas semanas e lá no “*I wana do*” você tem diferentes áreas para essas funcionalidades e cada coisa que você faz, você escreve o que você fez e manda o pacotinho la para uma área de aprovação, daí seu gerente vai e fala: “beleza isso realmente ele fez”, “isso aqui é valido vou aprovar ele vai ganhar tantos pontos por causa disso”. E ai tem uma área lá que tem uma casinha e sua casinha vai crescendo, você vai ficando maior e tudo mais. Todo funcionário tem isso daí.

(Entrevista 2, questão 17)

Dificilmente uma pessoa toma uma decisão sozinha aqui. Sempre: “Tenho tal problema e agente precisa resolver”. Quando é um negócio mais chato e mais complexo geralmente vão dois ou três mais experientes que já conhecem muito mais a ferramenta para fazer. Senão, a equipe costuma ser envolvida.

(Entrevista 2, questão 22)

Ele (o líder) é quem, atualmente, mais conhece da plataforma, do produto ali. Ele é quem, junto com o Gerente, consegue saber e dizer para o time: “têm essas funcionalidades que a gente precisa fazer para entregar tal *release*, ele conhecendo cada funcionário, consegue saber quem vai desenvolver qual tarefa. Embora muitas vezes as pessoas falam: “essa tarefa eu vou assumir”, “vou pegar que eu já sei como vou fazer” e sem problema nenhum. Ele ajuda muito as pessoas que tem algum problema na hora de fazer alguma coisa: “não, isso ai eu sei como é...”. Senta do seu lado, te ajuda, te dá o caminho. Isso sempre. É uma pessoa que o papel dele tem sido: cuidar dos problemas com o cliente, ajudar o time e dar direção nas *releases* que a gente vai entregar. Ele programa também e programa muito. Ele é muito bom e é mais novo do que quase todo mundo do time.

(Entrevista 2, questão 32)

A gente tem os almoços das *releases* para comemorar as entregas. Todo final do mês a gente tem o ninjas no boteco, que a gente teve ontem no caso. Tem um barril de chopp, eles trazem salgadinhos, trazem um monte de coisas. Confraternizações não faltam.

(Entrevista 2, questão 33).

A gente está no mesmo ambiente, a gente tem mesas próximas, mas as vezes é até melhor deixar registrado no Slack. Muitas vezes só tem você a noite, ou você e outra pessoa, você deixa registrado: “Estou procurando tal coisa”. Daí você deixa la e vai embora. De manhã o pessoal

vê, ou mesmo a noite o pessoal vê e responde. Acho que o Slack é o ponto que consegue reunir todo mundo.

(Entrevista 2, questão 34)

Eu não tive treinamento em si. [...] Além disso, no dia a dia eles te mostram a ferramenta, te dizem com o que você vai trabalhar que no caso é o eclipse e o java. Ai eles te passam os padrões de limitação, eles tem lá um arquivo que tem as configurações toda que quando você faz ele já coloca seu código bonitinho. A medida que você vai tendo dúvida o líder ou alguém mais experiente vem e te ajudar, fala assim: “Isso aqui é para fazer tal coisa, você vai fazer desse jeito aqui”. Então não tem um treinamento específico. Acho que tem que ser no dia a dia mesmo, porque cada funcionalidade é uma coisa nova. Tem coisa que você já fez e que você tirou de letra, e tem coisa nunca viu na vida e o cara: “não isso aqui é tal coisa” e ele vem e te ajuda. Então, acho que não tem como treinar a pessoa para isso.

(Entrevista 2, questão 35)

Ninguém tem uma lista de *skills* do outro. Acho que em reuniões e em conversas mesmo você acaba sabendo.

(Entrevista 2, questão 36)

Acho que as *dailys*, quando a gente está falando de um problema específico ou de alguma coisa, por exemplo, a gente estava falando de criptografia outro dia e daí surgiram algumas ideias: “acho melhor fazer assim...” ou “eu acho melhor fazer assado...” nessas horas, geralmente que acontece esse tipo de coisa. As vezes acontece nas confraternizações mas não é planejado, mas a gente não vê problema em falar em confraternizações esse tipo de coisa. Fala-se muito sobre tecnologia aqui.

(Entrevista 2, questão 37)

Agora o Trello e o Slack são ferramentas de trabalho, essas a gente tem usado para trabalhar mesmo. Se o trelo fica offline a gente fica sem norte.

(Entrevista 2, questão 38)

Nos *What's Going On (WGOs)* eles falam muito sobre isso. para a gente está muito claro como eles têm trabalhado. A empresa ataca em diferentes segmentos, desde uma propaganda ou uma melhor indexação no Google, por exemplo, até eventos que ela faz parte para angariar *leads*, para tentar transformar em oportunidades, até em vendas mesmo, consultores de venda que vão até o cliente e tentam fazer a venda. E eles têm feito estudo de mercado, Gartner e tudo mais, eles se preocupam muito com esse tipo de coisa. Inclusive eu fiquei sabendo que mudaram um pouco o foco de um dos produtos em função de pesquisas que o Gartner publicou. Eles estão bem por dentro disso aí. Eles são bem preocupados com esse tipo de coisa.

(Entrevista 2, questão 39)

Com certeza. Aqui todo mundo ouve todo mundo. Aqui é bem legal.

(Entrevista 2, questão 40)

Recompensado por uma ideia, dificilmente há uma recompensa física. O que tem é: “Foi tal pessoa que fez, você viu que legal?” Sempre tem esse reconhecimento do líder e dos gerentes. Isso eu vejo direto aqui, são coisas bem legais.

(Entrevista 2, questão 43)

[...] as vezes na entrega de um projeto, que tem uma customização o cara vai lá no cliente implementar e o cliente pede algumas coisas e ai vai fazendo em tempo de implantação.

(Entrevista 2, questão 49)

Mas uma coisa que eu posso te dizer é o seguinte: esse ambiente que você está vendo aqui, essa salinha com puff, televisão, etc. Isso não fica só aqui. Isso se estende lá para dentro. Isso faz todo mundo ter vontade de levantar para vir trabalhar, acho que isso é o principal.

(Entrevista 2, questão 50)

[…] reuniões para definir, por exemplo, a gente esta fazendo uma interface nova para a versão nova da API *switch*, fazemos reuniões, a gente olha interface, mostra e o diretor fala: “isso aqui não está legal, isso aqui tem que melhorar, o que vocês estão achando?” É legal porque a diretoria envolveu também os desenvolvedores para dar opinião sobre aquilo que estava acontecendo. Nessa decisão eles escutaram nossas opiniões e depois fizeram uma reunião mais fechada entre eles para bater o martelo.

ENTREVISTA 3

(Entrevista 3, questão 14)

Sim, eles têm bastante autonomia, a equipe em si tem bastante autonomia. Depende um pouco do nível de experiência de cada pessoa. Os mais júniores não tem tanta autonomia, eles ficam dependendo um pouco mais de pedir uma orientação para alguma coisa ou não. É mais uma questão de experiência mesmo, mas a gente tenta dar uma autonomia para todo mundo. Todo mundo vê ali um melhor caminho a ser seguido para uma funcionalidade, para desenvolver, para resolver um problema a gente tenta dar uma autonomia geral para todo mundo. Eu acho que para a pessoa que está ali no trabalho dela, ela ter a autonomia para ver o melhor caminho, ver a melhor solução, ver como que ela pode e gostaria de fazer, é importante poque faz com que ela fique instigada a ver realmente qual a solução correta pro problema em si, ou para implementar em si, isso força ela a querer realmente aprender mais, buscar mais conhecimento e ficar mais engajada no trabalho do dia a dia.

(Entrevista 3, questão 16)

Algo definido [para premiação], a empresa não tem. Mas a empresa sempre está premiando os funcionários realmente por se destacarem. Não tem uma cultura: se você fizer isso vai ganhar isso, mas a empresa sempre busca destacar os funcionários, seja mensalmente ou a cada trimestre, os funcionários de destaque, que foram atrás que fizeram alguma coisa de diferente, que fizeram um trabalho acima da média.

(Entrevista 3, questão 17)

A gente sempre faz um *brainstorming* e depois vai pro desenho. Vai para uma lousa e coloca tudo que a gente pensou, desenha para analisar bem e tentar chegar na melhor solução. Nós pegamos opiniões de várias pessoas diferentes e tenta desenhar. Depois disso fazemos uma documentação, que pode ser um *word*, alguma coisa, para gente tentar mapear qual era o problema e a solução que a gente está atacando.

(Entrevista 3, questão 18)

Antigamente a gente usava o Scrum que tinha conceitos de *sprints* fechados e tudo mais. A gente passou pro kanban exatamente porque nos do time de desenvolvimento, como a gente além de desenvolver produtos a gente cuida dos ambientes de produção, corrige problemas, inclui pequenas funcionalidades que o cliente precisa, então a gente não tinha muito controle do *roadmap*, para saber se num período de duas semanas a gente ia trabalhar só aquilo que estava definido. Normalmente não acontecia isso, então a gente mudou pro kanban para ter um escopo um pouco mais aberto. A gente tem um conjunto grande de *backlog*, de atividades a serem feitas e a forma como a gente trabalha é que a gente não define assim: "Olha a gente um sprint, vai fazer só isso e não vai mudar". Não! A gente tem um conjunto de tarefas prioritárias, a gente tem um conjunto de 15 tarefas prioritárias para serem feitas e na quarta-feira aconteceu um problema e é um problema grave, vamos despriorizar alguma atividade e vamos colocar isso aí e vamos fazer. Essa é a forma que a gente trabalha.

A gente usa o Trello hoje para controlar as nossas atividades. Então nas listas do Trello a gente tem todas as atividades a serem feitas. A gente tem um controle de atividades que cada um está fazendo no momento e tem o controle de tudo que já foi feito também.

(Entrevista 3, questão 19)

[…] eu vejo uma solicitação de um cliente que eu julgo importante para os demais clientes então eu incluo ali no meio, e a gente já faz junto com as demais atividades, mesmo não estando no nosso *roadmap.*

(Entrevista 3, questão 21)

Toda segunda-feira de manhã gente faz uma reunião um pouco mais extensa porque é onde a gente vê como foi as atividades da última semana, como se fosse uma retrospectiva, mas bem curta, no máximo 20 minutos, só repassando as atividades da última semana e então priorizamos as atividades que a gente julga que devem ser feitos durante a semana corrente. Então, a

gente já vê as principais atividades do *backlog* e coloca para serem feitas durante essa semana.

Depois no restante da semana, diariamente a gente tem as reuniões de *daily* mesmo. Para saber como está indo as atividades de cada um, ou se tem algum problema, algum *block*, o quê que tem? Então diariamente a gente tem uma reunião normalmente de 15 minutos.

(Entrevista 3, questão 23)

O *Product Owner*, no nosso caso é o próprio gerente. Ele é o responsável por me ajudar a priorizar as atividades, principalmente. Não é ele que fala tudo que precisa ser feito, bastantes solicitações são feitas diretamente pelos clientes, pois temos bastante contato com o cliente. Os *inputs* vem direto dos clientes, mas ele me ajuda a priorizar os itens do *backlog* e também a validar algumas coisas, nem tudo ele valida por causa dessa múltipla função, mas ele que é o responsável por dar o OK da maior parte das funcionalidades do produto.

(Entrevista 3, questão 24)

Para alguma funcionalidade grande a gente costuma sentar juntos e, pelo menos no início, fazer uma programação em par com uma pessoa mais experiente e uma outra mais júnior, para depois caminhar sozinho. Mas é bem pouco, muito menos do que a gente gostaria.

(Entrevista 3, questão 31)

[...] Antes, alguns faziam outros não. Eu estou tentando mudar isso. A gente está trabalhando no esquema de TDD atualmente. Nem tudo a gente consegue testar por falta de tempo. Mas nosso foco está em TDD. Conseguir fazer todos os testes necessários. A gente trabalha com java aqui, a gente faz tanto das classes java quanto também do *front-end*. A gente testa, a gente faz testes do java-script. Então a gente esta tentando cobrir a maior parte possível do produto com testes automatizados.

[...]

O que a gente faz hoje é que temos uma aplicação que roda junto com o Trello, puxando os dados do Trello ele plota várias informações para a gente. Como a gente trabalha bastante com *labels* no Trello para cada tipo de atividade, criticidade e tudo mais. Assim, essa aplicação plota os gráficos com base nos *labels* e nos *cards*. Então a gente consegue saber, por exemplo, se estamos gastando mais tempo numa *feature* ou em um *bug*. Esse tipo de informação. Mas isso tudo medido depois, não durante o desenvolvimento e nem antes, apenas depois.

(Entrevista 3, questão 32)

A gente tem a área do café, nossa copa. Ela serve bastante para isso mesmo. Para a gente: "vamos dar uma pausa", "vamos tomar um café", a gente conversa, põe tudo para fora realmente. Temos confraternizações mensais, que acontece aqui também. Confraternizações de aniversariantes ou confraternizações que a gente combina por fora, algum *happy hour*, alguma coisa. Normalmente acontece no dia a dia mesmo.

(Entrevista 3, questão 33)

A gente faz um documento quando estamos modelando as funcionalidades, a gente faz um *Word*, basicamente aquilo que comentei com você, escrevendo como vai ser a funcionalidade, o que tem que ser implementado, as vezes as classes, os métodos, mas outros documentos não.

(Entrevista 3, questão 34)

No sentido de treinamento e aprendizado no dia a dia a gente tem o que a gente chama aqui de "*I Know How*" que são palestras sobre assuntos específicos que alguém aqui mesmo da empresa dá. [...] é uma hora de conteúdo que o cara prepara e apresenta para todo mundo. Isso é uma forma que a gente tem de compartilhar conhecimento.

(Entrevista 3, questão 35)

O que ajuda bastante nisso são as nossas *dailys*. No memento da *daily* a pessoa fala estou com problema nisso e quem sabe já se apresenta e fala acabando a *daily* e já te ajudo nisso. Esse é o momento que a gente mais compartilha isso. É justamente nesse momento que as pessoas ficam sabendo qual é o ponto forte de cada um.

(Entrevista 3, questão 36)

A gente trabalha bastante com análise de concorrentes. A gente tem poucos concorrentes mas concorrentes bem fortes. Pega o produto do concorrente, vamos usar, vamos ver, sentir como é, para gente saber os problemas e não fazer no nosso.

(Entrevista 3, questão 37)

Trello, Slack, Bitbucket (que é um servidor de git). De metodologia da parte mais técnica a gente usa Jenkins, nosso servidor de *build*. E-mails, Hangouts, intranet Yammer.

(Entrevista 3, questão 38)

Pra gente é bem importante essa conversa com eles. Eles estão em contato direto com os clientes. O que eles trazem para gente são sugestões dos próprios clientes, que chegam até a gente através deles. É bastante útil o que eles trazem para gente.

(Entrevista 3, questão 41)

O Arquiteto é responsável pela arquitetura do projeto e o *tester* é exclusivo para teste. Um recurso compartilhado com o marketing, quando a gente precisa de algo além disso a gente consegue pegar alguém de fora e passar algum tempo no desenvolvimento. Ou dentro da própria empresa, mas de outra área ou através de nossos parceiros.

(Entrevista 3, questão 43)

Quando tem que ir no cliente, sou eu que vou até o cliente para uma pré-venda, para fazer uma POC (*Proof of Concept*), montar o desenho de uma POC e dai eu vou junto com a área comercial para entender qual é a necessidade, mas depois disso a gente não tem mais esse contato.

(Entrevista 3, questão 46)

Os consultores também trabalham com esse produto que a gente vende, eles têm bastantes *insights* também e bastantes sugestões, então eles trazem esses *inputs* para a gente e a gente acaba utilizando esse conhecimento no desenvolvimento.

(Entrevista 3, questão 48)

Para a gente é bem importante essa conversa com eles [consultores de linha de frente]. Eles estão em contato direto com os clientes. O que eles trazem para gente são sugestões dos próprios clientes, que chegam até a gente através deles. É bastante útil o que eles trazem para gente.

(Entrevista 3, questão 49)

Acho que uma das principais coisas são as pessoas envolvidas aqui da empresa. Os funcionários são bastante capacitados, que correram atrás e se especializaram realmente nesse assunto de APIs, que é o foco da empresa hoje. Então acho que um dos grandes fatores, não totalmente, mas um dos grandes fatores são os funcionários, as pessoas que trabalham aqui. E acho que a visão dos próprios diretores, a forma que eles gerenciam a empresa, a cultura que foi criada aqui dentro, acho que isso tudo facilitou bastante o sucesso da empresa.

(Entrevista 3, questão 50)

Todas as nossas funcionalidades são pensadas em como o cliente vai utilizar, a melhor forma dele fazer, do cliente resolver o problema que ele tem através do nosso produto. Então o foco está sempre no nosso cliente mesmo.

Dependendo da funcionalidade, se for algo um pouco mais complexo, é feito uma pré-validação diretamente com o cliente. Antes de implementar de fato, conversamos com o cliente: “a gente esta pensando nessa resolução, em fazer dessa forma. Você acha que para você resolveria isso?”. Daí o cliente pode dar o palpite antes da gente implementar de fato.

ENTREVISTA 4

(Entrevista 4, questão 14)

Com 6 meses de organização, treinamento e tudo mais, hoje em dia eles têm 100% de autonomia. Jogo na mão deles e eles se viram. Eles falam para mim como vão fazer, a gente discute, faz o cronograma. Eles dão o prazo e eu dou outro. A gente chega em um meio-termo do prazo, fazendo uma continha lá. Mas a maior parte eles que decidem.

(Entrevista 4, questão 15)

A decisão nunca é individual. Por mais que eu tenho um projeto que está na mão de uma pessoa e outro que está na mão de outro. Eu chamo a equipe de TI inteira para discutir sobre aquele novo projeto de uma pessoa só. Todo mundo participa.

Fazemos *brainstorming* sempre antes dos projetos. A autonomia está nisso, no próprio *brainstorming* (pós *brainstorming* ou o refino *brainstorming*) as pessoas falam: isso está certo, isso está errado. No *brainstorming* todo mundo participa.

(Entrevista 4, questão 16)

O que mais surtiu efeito foi dar autonomia para eles. O outro é que a gente tem um espaço que a gente chama de espaço *nerd*, que foi durante a época de treinamento. E depois do trabalho o dono pagava pizza para gente e eu trazia temas de computação, alguma coisa nova de programação, algoritmo diferente e a gente discutia isso. E eles propunham temas e a partir disso a gente construía o conhecimento da equipe para poder dar autonomia para eles. Então isso foi o que mais surtiu efeito. Para eles darem ideias novas. Isso surtiu efeito mesmo.

As metas são cumprir prazo. […] Outra meta que é subentendida é a geração de conhecimento para dentro da equipe. É feito pela Wiki, jogar o conhecimento lá na Wiki. Não [existe recompensa], é mais na zoação mesmo! Um zoando o outro! Por exemplo na operação a gente tem

os *tickets*. Então toda semana a gente divulgava quem matou mais *tickets*, dai “ah, esse matador ganhou um pedaço em pizza!”. Essas coisas.

(Entrevista 4, questão 18)

A gente usa um pouco do Kanban também, mas do Kanban a gente utiliza só o quadro mesmo.

(Entrevista 4, questão 20)

A gente só não faz a *daily*. A *daily* é muito informal. Como a gente está fisicamente muito próximo um do outro, a *daily* é feita durante o dia. Então de vez em quando, uma semana ou outra, que está num caminho crítico do projeto, tá em uma parte que não pode atrasar, então eu fico mais no pé! “E aí você fez até o que hoje?” Mas não é bem uma *daily* scrum, é mais para saber onde está e o que andou.

Fazemos o planejamento do *sprint*, o *sprint review*, a gente tem o *waremap* toda semana, que já faz o sprint ali. A gente faz o *sprint review* e inicia um novo, numa mesma reunião a gente faz isso, toda segunda-feira.

(Entrevista 4, questão 21)

A reunião do planejamento a gente faz um *brainstorming*, faz uma *wareframe*. “Todo mundo entendeu?” Então a gente faz o protótipo. Feito o protótipo, ao mesmo tempo a gente separa as funcionalidades, sequencia para ver quais têm dependência em relação a outra e começa com as de menor dependência, as que têm dependência zero. [...] Falo para os funcionários o que tem que ter, a gente já vê, já escreve tudo nesse quadro e já começa no mesmo dia, bem rápido mesmo. Assim, no máximo a gente dá uma semana para revisão. Eu mando o cronograma com o *backlog* inicial e todo munda fala “*ok”* e a gente começa, ou a gente faz a revisão.

(Entrevista 4, questão 23)

Temos o *status report* toda semana com outras áreas. Eu tenho o *sprint review* na segunda e na terça eu passo as coisas para eles. Passo o *link* para eles navegarem e tudo.

(Entrevista 4, questão 24)

No começo eu pegava 2 programadores, e falava: “vocês vão fazer esse projeto juntos” e passavam a mesma atividades para eles fazerem, então ou eles se auto dividiam ou eles faziam juntos. Aí deixava a cargo da afinidade de cada um.

Eu faço rotação também para mexer com as pessoas. Traz visões diferentes e também o fato do programador ou os caras do suporte se acomodam nas suas atividades. Quando começa a se acomodar daí a gente troca, daí dá uma agita. É motivação, né?

(Entrevista 4, questão 28)

A gente usa o MVC, *Model-View-Control*. O view é a interface, controller é o meio-termo entre interface e o model. O *model* é a interface entre o banco de dados e o sistema de arquivo. O *model* é baseado na estrutura de banco de dados. É uma estrutura, então cada tabela vai ter uma classe. Quando eu desenho o banco de dados e entrego o MER eles já sabem como vai ser a estrutura de classes.

(Entrevista 4, questão 30)

Não é orientado a testes. Acabou o sprint ele testa. O teste da pessoa faz parte do sprint. Daí ela manda um ou dois dias antes e a gente tem um formulário que o pessoal preenche no *docs*, tipo: “crie tal coisa, faça tal coisa e a gente passa por aí”, Daí eles preenchem. Assim, tem uma metodologia de teste, que é própria também.

(Entrevista 4, questão 32)

A gente ia almoçar junto, mas não falava muito não. Mas essa equipe que está aqui eles tem a característica de falar tudo na reunião mesmo! Ele chega na reunião mesmo e fala: “Isso aqui não está legal, não estou gostando disso” Ele fala mesmo, aberto mesmo. Então como tudo mundo pegou essa cultura, ficou bem tranquilo, ninguém esconde nada.

(Entrevista 4, questão 34)

Geralmente quando é algum conhecimento que é necessário de imediato, que o grupo não detém, eu passo mais atividades que requerem aquele conhecimento, para forçar mesmo! O cara que é desenvolvimento e ele não sabe fazer uma ligação entre dois sistemas, e ele não sabe fazer esse *swap* aí no meio, vou jogar ele na operação e vou começar a passar os atendimentos que tenham a ver com a troca de mensagem de servidores, fazer um rodízio mesmo.

O que mais acontecia era sentar do lado e ensinar, mas isso tem acontecido cada vez menos. Quando começou a equipe eles vinham e ficavam conversando comigo e como alguns absorveram mais rápido eles começaram a ser o centro.

Quando começou a equipe eles vinham e ficavam conversando comigo e como alguns absorveram mais rápido eles começaram a ser o centro. O legal da Wiki é que como o pessoal era nova, tava todo mundo a fim de preencher, todo mundo a fim de ensinar então a galera foi preenchendo sozinha mesmo.

(Entrevista 4, questão 35)

É mais na experiência mesmo, no entrosamento, proatividade. Você joga um assunto você vê quem fala primeiro. Geralmente é uma coisa meio manipulável! Se você quer que ele seja referência no assunto você começa a elogiar.

(Entrevista 4, questão 36)

As ideias mais interessantes surgem durante o desenvolvimento do sistema, ali na baia mesmo, no dia a dia. Quando o cara tá fazendo ele tem uma ideia e fala: “E se a gente fizer assim? Assim?”. Por isso que o scrum é legal, ele permite fazer uma modificação no meio do caminho sem ter que refazer todo o projeto.

(Entrevista 4, questão 38)

Geralmente a gente participa de congressos, feiras, etc. Daí a gente tenta sentir como está o mercado, ou o próprio mercado que vem procurar o que está precisando. Geralmente o pedagógico que vem com as ideias.

Na hora que a gente está escrevendo o sistema a gente olha os concorrentes, principalmente a interface.

(Entrevista 4, questão 41)

O cliente não fala diretamente, é muito raro, 99% vem da equipe de formação, mas geralmente é uma opinião contaminada, é uma opinião que não vem crua dos usuários. Uma das mudanças do sistema que implementamos agora é deixar o cliente colocar a opinião dele diretamente. Quanto maior a cadeia é pior.

ENTREVISTA 5

(Entrevista 5, questão 22)

Ele [o líder] gerencia esse processo do scrum e direciona. Caso a gente tenha alguma dificuldade ele costuma auxiliar-nos nessas tarefas. Ele senta do lado e programa junto.

(Entrevista 5, questão 25)

Cada um faz uma coisa. Caso algum tenha dificuldade em algum determinado ponto, um auxilia o outro. A gente meio que subdivide essa tarefa em 2, um pouco para cada. Cada um vai fazendo um e juntando tudo. Caso exista algum problema, como já falei, um auxilia o outro.

(Entrevista 5, questão 30)

Depois no final do projeto a gente submete para a equipe testar primeiro, corrige se tiver algum *bug* e depois submete para o cliente final. O cliente tem um período para teste e só quando esta 100% que a gente libera para o uso.

(Entrevista 5, questão 31)

O *brainstorming* é fundamental com outros departamentos. As reuniões, a Wiki, o processo todo, o acompanhamento do projeto, a programação em par.

(Entrevista 5, questão 33)

A gente tem uma reunião semanal, que não é essa de acompanhamento de projeto. A gente chama de espaço nerd, onde a gente discute algumas tecnologias, alguns métodos, algumas coisas que um tem mais afinidade e o outro não. Se tem alguma coisa que eu acho interessante eu vou e apresento, se outro tem ele vem e apresenta. A gente vai levando assim. É bem legal.

Não tem recompensa para a participação no espaço nerd. A recompensa é o conhecimento mesmo.

(Entrevista 5, questão 34)

O mais experiente ensina o menos experiente. Cursos a gente não têm, mas tem a Wiki que dá para ir atrás e aprender coisas.

(Entrevista 5, questão 35)

Como a equipe é pequena, acho que todos se conhecem bem. É o entrosamento mesmo. É conversando mesmo.

(Entrevista 5, questão 36)

Os momentos que mais surgem ideias são durante as sessões *de brainstorming*, nas reuniões e durante o desenvolvimento, no dia a dia.

(Entrevista 5, questão 43)

[…] uma equipe daqui da área de formação vai ao cliente, explica como funciona, treina as pessoas a utilizar essa ferramenta e traz bastante retorno de sugestão, dúvidas e críticas desses clientes. Também têm pessoas de desenvolvimento que vão fazer esse tipo trabalho, mas a maior parte são da formação. São professores, educadores.

(Entrevista 5, questão 47)

Se a gente tem um projeto novo que envolva outras áreas, temos reuniões que eles vão participar. Têm as reuniões semanais que eles também acompanham. Toda informação e integração

que essas áreas estão envolvidas são nessas reuniões. Como é tudo próximo aqui é fácil chegar e conversar com as pessoas envolvidas.

(Entrevista 5, questão 49)

Eu acho que é a qualidade do que a gente desenvolve. As ferramentas que o sistema possui. A gente tenta sempre estar um passo a frente da nossa concorrência. Com metodologia de ensino, porque seria como se fosse uma escola online, então a gente tenta ter algumas ferramentas a mais do que os nossos concorrentes têm. Então acho que é qualidade e inovação.

(Entrevista 5, questão 50)

São todos. Marketing, negócios, vendas. A gente trabalha muito com concorrência por licitação, edital, então se o mercado exige alguma coisa nova a gente tenta desenvolver para conseguir aquele edital. A gente tenta oferecer um produto que eles estão pedindo.