Universidade Federal do Estado do Rio de Janeiro
Centro de Ciências Exatas e Tecnologia
Escola de Informática Aplicada

Um Diagnóstico de Maturidade de Equipes Ágeis Utilizando a Bússola Ágil

Renard Sebastian Pessoa Ferreira
Thiago Albuquerque de Lima

Orientador
Prof. Gleison dos Santos Souza, D.Sc.

Rio de Janeiro, RJ - Brasil
Dezembro de 2018

Catalogação informatizada pelo(a) autor(a)

F383 Ferreira, Renard Sebastian Pessoa
Um Diagnóstico de Maturidade de Equipes Ágeis Utilizando a Bússola Ágil / Renard Sebastian Pessoa Ferreira. -- Rio de Janeiro, 2018.
65

Orientador: Gleison dos Santos Souza.
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal do Estado do Rio de Janeiro, Graduação em Sistemas de Informação, 2018.

1. Metodologias ágeis. 2. Maturidade ágil. 3. Bússola ágil. 4. Grupo Focal. I. Souza, Gleison dos Santos, orient. II. Título.

L345 Lima, Thiago Albuquerque de
Um Diagnóstico de Maturidade de Equipes Ágeis Utilizando a Bússola Ágil / Thiago Albuquerque de Lima. -- Rio de Janeiro, 2018.
65

Orientador: Gleison dos Santos Souza.
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal do Estado do Rio de Janeiro, Graduação em Sistemas de Informação, 2018.

1. Metodologias ágeis. 2. Maturidade ágil. 3. Bússola ágil. 4. Grupo Focal. I. Souza, Gleison dos Santos, orient. II. Título.

Um Diagnóstico de Maturidade de Equipes Ágeis Utilizando a Bússola Ágil

Renard Sebastian Pessoa Ferreira
Thiago Albuquerque de Lima

Projeto de Graduação apresentado à Escola de Informática Aplicada da Universidade Federal do Estado do Rio de Janeiro (UNIRIO) para obtenção do título de Bacharel em Sistemas de Informação.

Aprovado por:

_____

Prof. Gleison dos Santos Souza, D.Sc. (UNIRIO)

_____

Prof. Pedro Nuno de Souza Moura, D.Sc. (UNIRIO)

_____

Prof. Rodrigo Pereira dos Santos, D.Sc. (UNIRIO)

Rio de Janeiro, RJ - Brasil
Dezembro de 2018

AGRADECIMENTOS - RENARD

Gostaria de agradecer, primeiramente, à minha mãe, principal incentivadora e quem mais lutou para que eu chegasse até aqui. Quero agradecer tanto a ela quanto a minha avó, por todo o amor, e o esforço em moldar o meu caráter através de bons exemplos. E por toda a minha família, sou muito grato pelo carinho, pelos abraços, nos momentos bons e ruins e por todas as lições aprendidas. Sem eles, faltariam muitas das coisas que tenho de melhor.

Agradeço a todos os amigos, que da escola à universidade viveram os mesmos desafios, e estiveram comigo nos piores e nos melhores momentos. Em especial, ao Jorge Henrique e ao José Carlos, que já estão ao meu lado há mais de 15 anos e hoje são uma extensão da minha família. Além deles, devo agradecer ao João Victor e à Ana Luisa, pelos anos de apoio, pelo cuidado e por tudo que contribuíram com a minha formação.

Ao Pedro Barros, por ter me adotado como irmão, e me proporcionado tanto aprendizado. Aos demais amigos de EJCM, que me inspiraram e me mostraram que era possível construir um grupo de trabalho unido a ponto de desenvolver relações profundas e onde as pessoas tivessem uma preocupação genuína com a felicidade umas das outras.

Ao corpo docente da Unirio, pela excelência no ensino, preocupação e dedicação aos alunos. Agradeço aos professores Asterio Tanaka, Bruno Simões, Marcio Barros, Morganna Diniz e Pedro Nuno pelos conselhos e ensinamentos. Gratidão ainda maior ao Gleison, pela confiança em mim, pelo exemplo de dedicação e postura, pela paciência incrível e por todo o cuidado antes e durante a orientação deste trabalho. Agradeço, também, ao professor Rodrigo Santos e novamente ao Pedro Nuno, por aceitarem o convite para formação da banca.

Aos meus amigos de UFRJ e Unirio, por terem feito parte da minha trajetória acadêmica e que continuarão fazendo parte da minha vida: Afonso Carvalho, Amanda Rodrigues, Daniel Andrade, Daniel Lima, Daniel Villaça, Daniela Negreiros, Felipe Barros, Matheus Lessa, Pedro Braga, Raul Barbosa, Victor Springer e Stefan Teixeira. Ao Thiago Albuquerque, que além de minha dupla neste trabalho, é um dos meus amigos mais próximos, deixo um agradecimento especial por toda a parceria, incentivo e motivação nesses últimos anos. Por fim, agradeço às pessoas incríveis com quem trabalho ou já trabalhei, por todos os ensinamentos diários: Agapito Neto, Alexandre Sander, Arthur Vieiras, Caio Silva, Debora Algamis, Erik Veloso, Fernanda Dutra, Gabriel Guzowski, Gustavo Antunes, Igor Lima, Renata Ferreira, Ricardo Filardi e Romulo Brito.

AGRADECIMENTOS - THIAGO

À minha irmã Beatriz, meu orgulho, meu xodó. Apesar da distância física, ter a sua companhia virtual no meu dia a dia é uma das minhas maiores fortalezas. "Amore", dedico esse trabalho a você. Obrigado por carnavalizar ainda mais a minha vida.

Aos meus pais, Rosangela e Lima. Mãe, a sua luta e suas conquistas estão entre os meus maiores exemplos. Tenho orgulho de ter herdado alguns dos seus valores que eu mais admiro: honestidade e perseverança. Pai, obrigado por ter tornado possível o meu desejo de iniciar a vida acadêmica na minha querida cidade natal. Aos dois, obrigado pela educação, minha maior herança.

Ao meu amigo, irmão, parceiro de TCC e confidente para todas as horas, Renard. Há poucos anos eu ganhei um irmão mais velho, e você tem noção disso. Obrigado pelo suporte, obrigado por estar presente em todos os momentos e ter dividido tanta coisa. Minha inteira confiança, minha demasiada gratidão.

Aos meus padrinhos, Rosi e William, e aos meus primos, Mylena e Gabriel. Vocês foram incríveis. Minha família de sangue no Rio de Janeiro que sempre esteve disposta a me dar suporte e apoio em tudo que eu precisasse.

Aos meus amigos de curso, Allan Magalhães, Amanda Rodrigues, Bruno Franco, Daniel Villaça, Jonatham Petzold, Leonardo Menezes, Victor Fernandes e Victor Springer, vocês foram incríveis companhias ao longo dos quatro anos de muitas experiências adquiridas. Pilares incondicionais da minha formação, essenciais nessa jornada.

Aos meus amigos, em especial a: Arnaldo Júnior, Felipe Brito, Gabriel Bustamante, Gabriel Pinheiro, Geovane Júnior, Letícia Ramos, Michelle Bittencourt, Nicholas Alvares, Stefan Teixeira, e outros que foram fortalezas nesses últimos anos. A amizade de vocês me faz mais forte.

Ao meu amigo, educador, Pedro Nuno, como já manifestei em outras situações, você com certeza foi uma das pessoas que me ajudaram a ser quem sou hoje, como pessoa, e não apenas em relação a minha vida acadêmica. Você é magnânimo, dono de um caráter íntegro. Obrigado por ter aceitado o convite de fazer parte dessa banca.

Aos meus amigos de Salvador, Recife e Olinda, tantos e tão queridos, ainda que distantes, vocês também fizeram parte disso.

Aos inúmeros professores que estiveram comigo nessa longa jornada. Em especial os meus mais sinceros agradecimentos ao nosso professor Gleison Santos, pela orientação e dedicação a esse trabalho.

Aos também professores, Alexandre Andreatta, Claudia Cappelli, Geiza Hamazaki, Leonardo Azevedo, Márcio Barros, Mariano Pimentel, Sidney Lucena, Vânia Félix, entre outros, a minha mais profunda gratidão a todos os ensinamentos transmitidos, e ao Rodrigo Pereira dos Santos, que também se disponibilizou para a formação da banca.

Ao Zuzú Goró. Charlene, Maicon Douglas, Thiago. Foram anos incríveis, ao lado de excelentes companhias, nunca esquecerei as tantas noites de conversas e experiências trocadas.

À Rafaela Fontana, idealizadora da Bússola Ágil, inspiração para essa pesquisa.

A todos os funcionários da Universidade Federal do Estado do Rio de Janeiro pelo apoio competente em suas atividades, vocês são parte da realização da minha graduação e diretamente ajudaram a tornar esse sonho possível. Em especial ao Douglas Brito e a diretora Morganna, em todos os momentos dispostos e solícitos.

A todos que contribuíram direta ou indiretamente para minha formação, que sempre acreditaram no meu potencial e fomentaram a eterna busca pelo conhecimento a qual estou inserido.

A todos os santos e arcanjos que me protegem e iluminam o tempo inteiro. A fé foi substancial e imprescindível nesse percurso.

À Deus por todas as oportunidades, saúde e força para ter concluído esta caminhada.

RESUMO

Após mais de duas décadas da publicação do manifesto ágil, o pós-agilismo ganhou forma, aonde as equipes ágeis adotam e adaptam diferentes práticas aos seus próprios contextos. O conceito de maturidade ágil surge como uma resposta ao questionamento de times de desenvolvimento a respeito de quão ágeis eles estão sendo, na prática, em seus processos de software. Os níveis de maturidade podem ser analisados, fomentando-se uma discussão na qual é verificado se de fato os times de desenvolvimento alcançaram a maturidade ágil em seus processos de software.

Esse trabalho pretende verificar o nível de maturidade ágil em dois times de desenvolvimento de software, apresentando a experiência e o diagnóstico da aplicação da Bússola Ágil, relatando as discussões e propondo soluções de melhorias para os processos das equipes participantes do diagnóstico.

Para atingir o objetivo, foi utilizada a metodologia Grupo Focal, pois oferece uma abordagem rápida e eficiente para coletar experiências e feedbacks de um grupo de pessoas sobre um tema em comum. Os dados obtidos foram suficientes para que fosse possível gerar o desenho da Bússola, propiciando assim a identificação de oportunidades de melhoria nos processos das equipes.

Palavras-chaves: *Metodologias ágeis, Maturidade ágil, Bússola ágil, Grupo Focal.*

ABSTRACT

After more than two decades of the publication of the agile manifesto, post agilism has gained shape, agile teams adopt and adapt various practices to their own contexts. The concept of agile maturity emerges as a response to the questioning of development teams as to how agile they are in practice in their software processes. The levels of maturity may be analyzed, fomenting a discussion where it is meant to be verified if development teams have in fact reached maturity in their software processes.

This work intends to verify the level of agile maturity in software development teams, through the application and diagnosis of the Agile Compass, reporting discussions and proposing improvement solutions for the participants teams processes evaluated in the experiment.

In order to achieve this goal, the Focal Group methodology was applied, as it offers a fast and efficient approach to collect experiences and feedback from a group of people on a common theme. The collected data was sufficient to generate the teams compass designs, and also allowed identification of opportunities for improvement on their processes.

Keywords: *Agile Methodologies, Agile Maturity, Agile Compass, Focal Group.*

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

LISTA DE SIGLAS E ABREVIATURAS

CMMI	Capability Maturity Model Integration
FDD	Feature Driven Development
MVP	Minimum Viable Product
MPS	Melhoria de Processo de Software
RAD	Rapid Application Development
RUP	Rational Unified Process
TI	Tecnologia da Informação
UML	Unified Modeling Language
XP	Extreme Programming

1. INTRODUÇÃO

1.1 Contexto

O conceito de agilismo surgiu como uma resposta aos problemas de longa data da indústria e revolucionou a forma como os processos de software são executados. Com a popularização dos métodos ágeis, a partir da publicação do Manifesto Ágil, a adoção pela indústria ao longo dos anos foi significativa e hoje seus princípios e metodologias são amplamente utilizados por times e organizações. Se no início da década passada, os métodos ágeis ainda eram vistos com desconfiança e adotados apenas por organizações e times mais jovens, com o passar do tempo, tornaram-se consenso na indústria [Melo et al., 2012]. O número de artigos e pesquisas relacionadas ao tema tem sido crescente, segundo levantamento de dados de números de publicação sobre desenvolvimento ágil de software [Dingsøyr et al., 2012].

Hoje, como consequência dessa discussão, muitos enxergam os métodos ágeis como regras de ouro, onde frameworks inteiros são não só aceitos, como esperados em diversas organizações. Por isso, muitos times podem adotar um método ágil, fazendo uso das semânticas e dos léxicos das práticas ágeis, sem de fato estar abraçando seus princípios, nem avaliando e criticando os resultados obtidos. Acabam por utilizar uma espécie de “camuflagem ágil”.

1.2 Problema

Superficialmente, esses times parecem estar usando métodos ágeis e, muitas vezes, realmente acreditam que o estão fazendo. Porém, um time que usa camuflagem ágil, além de poder estar na verdade utilizando um modelo tradicional, não estará colhendo os reais benefícios trazidos pela metodologia e nem alcançará a maturidade ágil. Um exemplo dessa camuflagem ágil pode ser visto em equipes que organizam suas entregas em sprints, mas convivem com uma figura da gerência que constantemente exige que atividades sejam adicionadas e priorizadas durante o sprint. Independente de usar um método ou adotar uma prática específica, é importante identificar e verificar o quão ágil é uma equipe e o seu nível de maturidade.

Assim, o objetivo deste trabalho é realizar um diagnóstico de dois times ágeis atuantes em organizações sediadas no Rio de Janeiro utilizando a Bússola Ágil [Fontana, 2015]. A partir desse diagnóstico, a maturidade ágil será determinada e soluções de melhoria para os processos serão elencados.

1.3 Estrutura

O presente trabalho está organizado em capítulos e, além desta introdução, será desenvolvido da seguinte forma:

- *Capítulo 2: Fundamentação Teórica*: Neste capítulo é abordado o conceito de agilidade, a origem dos métodos ágeis e seus valores originais. Será comentado também sobre o conflito de opiniões acerca da utilização das metodologias ágeis por parte dos criadores e críticos do Movimento Ágil. Além disso, serão trazidas discussões a respeito do conceito de Maturidade Ágil e as definições e diretrizes acerca da Bússola Ágil proposta por [Fontana, 2015].
- *Capítulo 3: Diagnóstico de Maturidade:* Neste capítulo é apresentada a experiência de aplicação da Bússola Ágil, definida em [Fontana, 2015], nos times ágeis escolhidos, por meio da execução de um Grupo Focal [Romain, 2015]. Discutem-se, também, os resultados obtidos e as possíveis melhorias identificadas nos processos de desenvolvimento em uso pelas referidas equipes.
- *Capítulo 4: Conclusão*: Neste capítulo serão apresentadas as conclusões finais decorrentes da análise dos dados e resultados obtidos. Também serão apresentadas as limitações do trabalho e possíveis trabalhos futuros.
- *Anexo 1:* Bússola Ágil
- *Apêndice 1:* Execução e Análise da Sessão Piloto

2. FUNDAMENTAÇÃO TEÓRICA

Esse capítulo além de apresentar informações sobre o conceito de agilidade, a origem dos métodos ágeis e suas principais metodologias, trará as motivações que incentivaram o surgimento dessas metodologias e como elas foram exploradas na indústria. Também serão apresentados fatores que impactaram a transformação ágil e discussões sobre o conceito de Maturidade Ágil, além das definições e diretrizes acerca da Bússola Ágil proposta por [Fontana, 2015].

2.1 Metodologias Tradicionais

Antes de nos aprofundarmos nas metodologias ágeis, começaremos com uma contextualização sobre modelos tradicionais, por possuírem uma grande importância na história da Engenharia de Software e por serem amplamente utilizados pela indústria de software. A Figura 1 apresenta em ordem cronológica o histórico de surgimento dos principais métodos de desenvolvimento de software usando como critério o ano de sua publicação, diferenciando as metodologias ágeis das tradicionais por cores e definindo o marco do Manifesto Ágil.

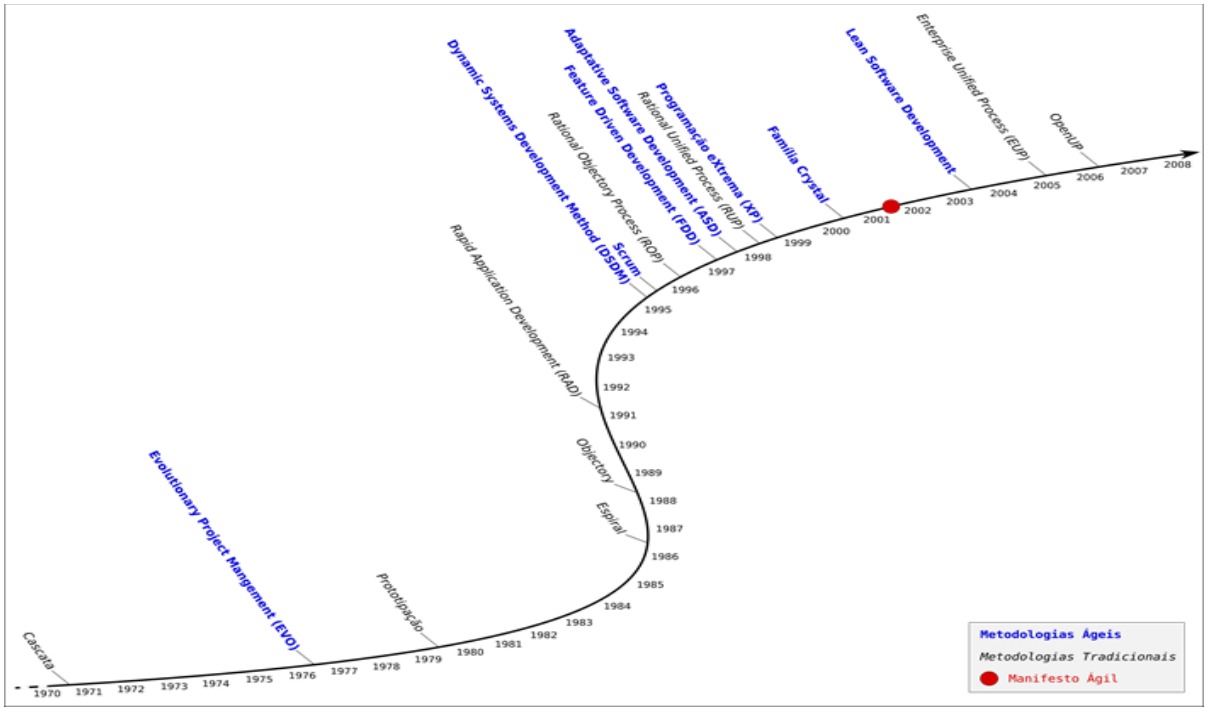

Figura 1: Histórico de publicações de metodologias de desenvolvimento de software. Fonte: [Bassi, 2008]

Desde o final da década de 60, técnicas de engenharia passaram a ser usadas para o desenvolvimento de sistemas de software com o objetivo de definir processos que

pudessem ser descritos e depois replicados. Essas abordagens segmentam a criação do software em fases apoiadas por processos, pelo uso de ferramentas e por uma documentação detalhada de tudo que é produzido [Bassi, 2008].

Como parte do processo, a equipe precisa produzir diversos tipos de itens intermediários até chegar no produto final. Alguns exemplos são: diagramas UML, glossários, casos de uso, especificações, modelagens, solicitações, planos de trabalho, formulários, componentes, testes, relatórios, análises, manuais, dados de carga, entre outros. Estes elementos são chamados genericamente de artefatos. Para cada fase da produção do software, alguns artefatos são criados ou atualizados e um conjunto de ferramentas deve ser usado para armazenar dados que podem ser empregados em análises ou servir como uma parte da documentação. Os documentos criados em cada etapa descrevem os detalhes do trabalho realizado até aquele momento, de forma que a próxima fase possa usá-los como base para continuar a produção do software. Esta, por sua vez, atualiza e produz novos documentos que serão utilizados pela etapa seguinte [Bassi, 2008].

A documentação atua como elo entre as fases do desenvolvimento do software e, entre os seus participantes, exercendo basicamente duas funções específicas: servem como registros históricos das decisões do projeto e como manuais que explicam o código e o funcionamento do sistema.

Alguns modelos de maturidade tradicionais são descritos a seguir.

2.1.1 *Cascata*

Neste modelo, o software final é obtido através da execução de etapas sistematicamente definidas. O processo segue linearmente as etapas de: engenharia do sistema, análise de requisitos, projeto, geração de código, testes e manutenção [Pressman, 2004]. Em cada uma dessas fases, um conjunto pré-estabelecido de atividades é executado de forma que os artefatos produzidos em cada etapa sirvam de entrada para a etapa seguinte.

2.1.2 Prototipação

Neste modelo, após uma fase inicial de análise e coleta de requisitos, a equipe de desenvolvimento parte para a construção de um protótipo do projeto principal com o objetivo de tornar as discussões em torno do suposto produto mais palpáveis. Com base no protótipo, o cliente pode refinar suas ideias e sugerir mudanças, até possuir uma visão consistente o suficiente para delinear as características do produto final [Bassi, 2008].

2.1.3 Espiral

O modelo Espiral, proposto por [Boehm, 1986], apresenta duas grandes inovações em relação aos modelos anteriores: a introdução do modelo iterativo, no qual as etapas do desenvolvimento são realizadas várias vezes no formato de ciclos e a inclusão de uma etapa de análise de riscos, onde os envolvidos no projeto podem avaliar e tomar decisões conforme as dificuldades que surgem durante o desenvolvimento.

As abordagens cascata e de prototipação estão presentes como uma das etapas do ciclo de desenvolvimento do modelo espiral. Cada iteração é formada por quatro etapas: Planejamento, Análise de Riscos, Engenharia e Avaliação do Cliente. A cada iteração, o produto evolui e se aproxima do que será a versão que entrará em produção.

2.1.4 Rapid Application Development - RAD

James Martin criou o RAD [Martin, 1991] no início da década de 90 com a proposta de produzir aplicações com um único ciclo de desenvolvimento, de no máximo três meses. Apesar do nome e de usar um ciclo de desenvolvimento relativamente curto, RAD não é um método ágil. Esse modelo foi especialmente recomendado para aplicações que possam ser modularizadas, com escopo claro e bem definido [Martin, 1991]. Sua abordagem se baseia em quebrar o software em módulos para que várias equipes trabalhem ao mesmo tempo. Cada equipe trabalha em um dos módulos, usando o modelo cascata com as seguintes etapas: modelagem de negócios, modelagem de dados, modelagem do processo, geração da aplicação, testes e correções [Kerr and Hunter, 1993]. No final, os módulos são integrados.

As duas principais vantagens do RAD são: a proposta de concluir o projeto em um período preestabelecido e relativamente curto (90 dias) e o estímulo à produção de componentes modularizados e reutilizáveis. Por outro lado, há alguns pontos negativos: requer mais pessoal e equipamentos para equipes que trabalham simultaneamente, não comporta mudanças de requisitos e não é recomendado quando os riscos técnicos são altos.

2.2 Origem das Metodologias Ágeis

Os Métodos Ágeis surgiram como uma resposta aos métodos tradicionais, muitas vezes, considerados lentos e burocráticos. Um número crescente de organizações de TI está em um movimento de transformação ágil para não somente incorporar as práticas e metodologias ágeis em toda a organização, como também seus valores e princípios [Dikert et al., 2016].

A exponencial evolução da computação e o crescimento da utilização de suas tecnologias, tanto na indústria e nas organizações, quanto para uso pessoal, fez com que a

demanda por sistemas aumentasse em grande escala [Bassi Filho, 2008]. As metodologias tradicionais são orientadas a planejamentos, o contato com o cliente e seu *feedback* não são frequentes e seu sucesso depende de uma série de requisitos e fatores imutáveis [Leal, 2014]. Isso não condiz com a realidade de grande parte das empresas de desenvolvimento, que não podem arcar com os custos de um processo pesado e demorado [Soares, 2004].

Entregar um produto funcional com a maior qualidade, em menores tempo e custo possíveis continua sendo uma meta das empresas de desenvolvimento. Nesse contexto, em meados da década de 90 começaram a surgir processos alternativos de desenvolvimento de software, em resposta aos métodos tradicionais, considerados lentos e burocráticos. Esses novos processos foram apelidados de “leves” (*lightweight*) e tinham como características um desenvolvimento iterativo, flexível a mudanças, alinhado aos objetivos da empresa ou do cliente, com foco em entregas rápidas e de alta qualidade [Prikladnicki et al., 2014].

Essas metodologias só realmente passaram a ser chamadas de Métodos Ágeis em fevereiro de 2001, quando 17 especialistas em desenvolvimento de software se reuniram representando diversas metodologias já existentes (Scrum, XP, Crystal e outras) e estabeleceram princípios e valores comuns a todas elas. Essa reunião desencadeou o que conhecemos hoje como Manifesto Ágil, que é composto pela declaração de 4 grandes valores e 12 princípios. A seguir, são apresentados os valores ágeis [Beck et al., 2001]:

- Indivíduos e interações mais que processos e ferramentas;
- Software em funcionamento mais que documentação abrangente;
- Colaboração com o cliente mais que negociação de contratos;
- Responder a mudanças mais que seguir um plano.

O Manifesto Ágil deixa claro que apesar dos itens da esquerda serem mais prioritários (indivíduos, software em funcionamento, colaboração com o cliente, responder a mudanças), também existe valor nos itens à direita (processos, ferramentas, documentação, contratos e planos).

2.3 Princípios Ágeis

Os 4 valores do Manifesto Ágil se desdobram em 12 princípios [Beck et al., 2001], que são:

- Satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado;
- Mudanças nos requisitos são bem-vindas, em qualquer fase do desenvolvimento. Processos ágeis tiram vantagem das mudanças visando à vantagem competitiva para o cliente;
- Entregar frequentemente software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo;
- Pessoas de negócio e desenvolvedores devem trabalhar diariamente em conjunto por todo o projeto;
- Construa projetos em torno de indivíduos motivados. Dê a eles o ambiente e o suporte necessário e confie neles para fazer o trabalho;
- O método mais eficiente e eficaz de transmitir informações para e entre uma equipe de desenvolvimento é através de conversa face a face;
- Software funcionando é a medida primária de progresso;
- Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente;
- Contínua atenção à excelência técnica e bom design aumenta a agilidade;
- Simplicidade, a arte de maximizar a quantidade de trabalho não realizado, é essencial;
- As melhores arquiteturas, requisitos e designs emergem de equipes auto-organizáveis;
- Em intervalos regulares, a equipe reflete sobre como se tornar mais eficaz e então refina e ajusta seu comportamento de acordo.

Dentre as metodologias ágeis disponíveis, as mais conhecidas são: Extreme Programming [Beck, 1999], Lean Software Development [Charette, 2002], Feature- Driven Development [Coad e Palmer, 2002], Scrum [Schwaber e Beedle, 2002] e Adaptive Software Development [Highsmith, 2002].

Cada metodologia abrange diversas práticas, sendo algumas únicas de um método específico ou compartilhadas entre diferentes métodos. Para uma maior compreensão de sua distribuição, a Tabela 1 agrupa por similaridade as mais utilizadas de acordo com o estudo feito por Silva (2013) nos artigos de Abrantes e Travassos (2011), Williams (2010), Kurapati

et al., (2012) e Jalali e Wohlin (2010). Algumas práticas de características semelhantes receberam nomenclatura distintas de acordo com cada autor, por esse motivo foram mantidos os termos em inglês.

ABRANTES E TRAVASSOS (2011)	WILLIAMS (2010)	KURAPATI et al (2012)	JALALI E WOHLIN (2010)
Coding standards	Code and Tests Nightly Build Ten-Minute Build	Coding standards	Code standards
Continuous integration	Continuous Integration	Continuous integration	Continuous integration
Pair programming	Pair Programming	Pair programming	Pair programming
Planning Game	Planning Poker Wideband Delphi Estimation	Planning game	Planning game
Project visibility	Informative Workspace	Tracking progress	Burndown charts Virtual scrum wall
Refactoring	Code and Tests Executable Documentation	Refactoring	Refactoring Code reviews
Small releases	Short Iterations Short Releases Sprint Incremental Design	Short / small releases Sprint / iteration	Short iteration
Stand-up meetings	Stand-Up Meeting	Stand-ups	Stand-up meetings
Test Driven Development	Acceptance Test-Driven Development Unit Test-Driven Development	Test driven development	TDD
Collective Code Ownership	Collective Code Ownership Code Ownership	Collective Ownership	
Metaphor		Metaphors	System metaphor
On-site customer	Sit Together Negotiated Scope		Close collaboration Proxy customer
Product Backlog	Release and Iteration Backlog		Backlog
Sustainable Pace (40 hour week)	Sustainable Pace Energized Work	40 hour week	
Whole team (multi-skill teams)	Whole Team	Team	
	Retrospective	Retrospective	Retrospectives
	Scrum Meeting	Sprint planning meeting Sprint review meeting Informative workshops	Planning meeting Sprint review
	Features Stories	Stories / Features	User stories
	Automation-Driven Root Cause Analysis of Failures Iteration Demonstration Inspections	Testing	Automated testing Acceptance tests Unit testing
Open workspace		Office structure that supports agile development	
Simple design		Simple design	
		Communication	Instant messages
	Done Criteria		
		Configuration and Change management	
		Documentation	
			Feature driven development
			Scrum of scrums
			Sprint demo

Tabela 1: Relação das práticas utilizadas em seus respectivos relatos. Fonte: [Silva, 2013]

2.4 Métodos ágeis mais utilizados

Para diferentes times e cenários, existem diferentes metodologias, das quais as mais conhecidas são apresentadas a seguir.

2.4.1 Adaptive Software Development

Processo que cresceu do RAD (Rapid Application Development), utiliza os princípios de adaptação contínua do processo ao trabalho. É dividido em 4 fases: comunicação e planejamento, análise, design e desenvolvimento, testes e implementação [Highsmith, 2002].

2.4.2 Agile Unified Process

Uma versão simplificada do RUP (*Rational Unified Process*). Descreve uma abordagem simples de desenvolvimento de software para área de negócios usando os conceitos ágeis [Ambler, 2014].

2.4.3 Crystal Methods

Família de métodos escolhida de acordo com características da equipe e do projeto. Cada método possui sua própria cor, de acordo com o tamanho (número de pessoas) do projeto e letras que caracterizam o nível de criticidade do mesmo [Highsmith, 2002].

2.4.4 Dynamic Systems Development Method

Uma formulação do RAD voltada para a entrega de projetos da indústria. Possui 9 princípios que se relacionam diretamente com o Manifesto Ágil, com ênfase na necessidade e valor de negócio, empoderamento dos times, entregas frequentes, testes integrados e colaboração dos clientes [Highsmith, 2002] [Mclaughlin, 2017].

Na Figura 2, extraída de [Stapleton, 2003], as três fases iterativas do processo são representadas por círculos: a iteração do modelo funcional, a iteração de design e a implementação. As setas indicam que além da iteratividade dentro de cada uma das fases, as três são iterativas também entre si. Com experiência, esse modelo permite a criação de planos altamente flexíveis para o projeto [Bassi, 2008].

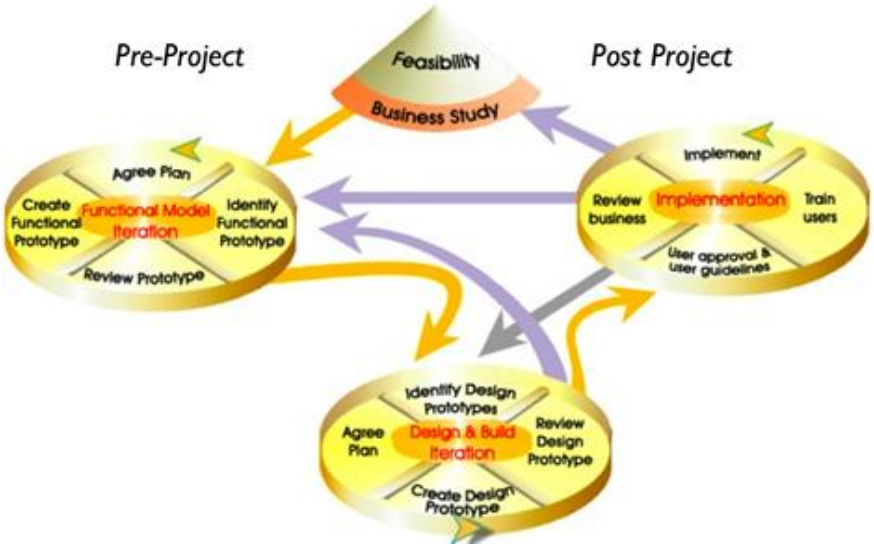

Figura 2: Fases do processo do DSDM [Fonte: Stapleton, 2003]

2.4.5 Extreme Programming (XP)

Um dos métodos ágeis mais conhecidos, tendo como foco a entrega contínua de software de alta qualidade através do trabalho próximo ao cliente. Contribuiu para alterar a visão acerca do custo de mudanças no projeto e prega 4 valores: simplicidade, comunicação, feedback e coragem. Suas 12 práticas são: planning game, entregas curtas e frequentes (*small releases*), metáfora, design simples, desenvolvimento voltado a testes (*test-driven development*), refatoramento, programação em duplas (*pair programming*), autoria coletiva, integração contínua, semana de 40 horas, cliente no local e código padronizado [Highsmith, 2002] [Mclaughlin, 2017].

2.4.6 Feature Driven Development

Método que busca o desenvolvimento por funcionalidade através de um processo minimalista de 5 passos: desenvolvimento de um modelo abrangente, criação de uma lista de funcionalidades, planejamento das funcionalidades, detalhamento das funcionalidades e construção de acordo com as funcionalidades [Highsmith, 2002].

2.4.7 Lean Software Development

Seus princípios derivam da produção Lean, que adota o conceito de estabilidade dinâmica, que é a habilidade de se adaptar rapidamente e de forma efetiva à demanda dos clientes, mantendo a estabilidade dos processos internos da organização.

Tem como princípios: a eliminação de desperdícios, o aprendizado amplificado, decidir o mais tarde possível, entregar o mais rápido possível, times empoderados, integridade na construção do software e a visão do todo [Highsmith, 2002].

2.4.8 Scrum

Um framework ágil de gerenciamento de projetos amplamente adotado devido a sua simplicidade, produtividade comprovada em diversos contextos e a sua flexibilidade quando comparado aos outros métodos ágeis [Abi-Saber, 2016]. O trabalho é realizado em iterações ou ciclos de até um mês de calendário chamado de Sprints. No Scrum, o trabalho realizado em cada Sprint (Figura 3, Ciclo dos Sprints) deve criar algo de valor tangível para o cliente ou usuário. Sprints são *time-boxed* (duração fixa) para que tenham sempre uma data fixa de início e fim, e, geralmente, todos eles devem estar com a mesma duração.

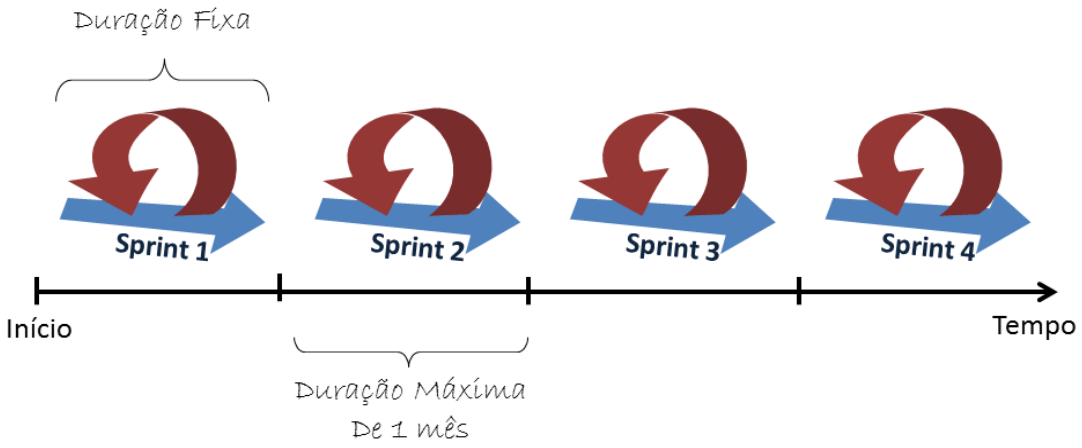

Figura 3: Ciclo dos Sprints [Fonte: Vieira, 2014]

O dono do produto (Product Owner) prioriza as estórias de usuário e, juntamente com o time de desenvolvimento, define um backlog de atividades para o próximo Sprint. Suas principais práticas são: gráfico burndown, planning poker, scrum of scrums, entregas curtas

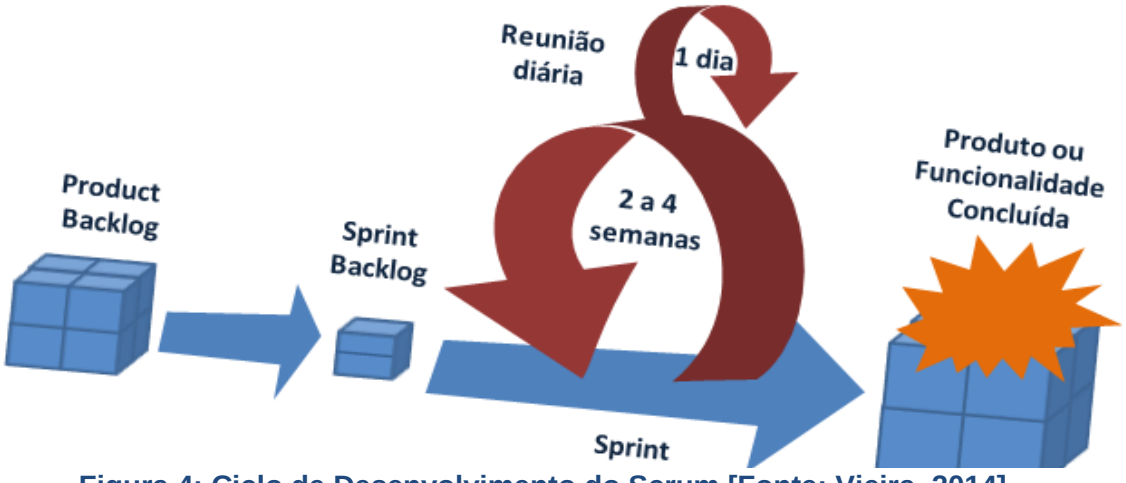

Figura 4: Ciclo de Desenvolvimento do Scrum [Fonte: Vieira, 2014]

e frequentes, reuniões diárias, reuniões de planejamento e reuniões de retrospectiva [Schwaber e Sutherland, 2017].

Na Figura 4 é apresentado o ciclo de desenvolvimento do Scrum. Na primeira reunião de planejamento do sprint, todos se reúnem para definir a meta do sprint e criar o backlog (Figura 4, Sprint Backlog) selecionado de acordo com as prioridades de negócio estabelecidas pelo dono do produto. Em seguida, a equipe se reúne para quebrar os itens do backlog selecionado em tarefas de implementação de menor granularidade.

Durante o sprint, o Scrum Master assegura que os itens do backlog selecionado não mudarão e elimina os itens do backlog de impedimentos para garantir que a equipe mantenha-se focada em seu objetivo. O dono do produto acompanha o desenvolvimento e esclarece eventuais dúvidas, sem poder mudar o escopo, porém, caso realmente precise mudar os itens do backlog selecionado, ele tem a opção de cancelar o sprint [Schwaber e Sutherland, 2017].

No final do Sprint, existem duas atividades adicionais que são fundamentais. Uma delas é chamada Sprint Review (revisão do Sprint). O objetivo desta atividade é verificar e adaptar o produto que está sendo construído. Esta é uma reunião informal, e a apresentação do incremento destina-se a motivar e obter comentários, além de promover a colaboração.

Enquanto o objetivo do Sprint Review é verificar necessidades de adaptações no produto, a Sprint Retrospective (retrospectiva do Sprint) tem como objetivo verificar necessidades de adaptações no processo de trabalho. A retrospectiva ocorre depois da Sprint Review e antes da reunião de planejamento da próxima Sprint. Esta é uma reunião time-boxed de três horas para uma Sprint de um mês.

2.4.9 Kanban

Através de um quadro (um exemplo de quadro kanban pode ser visto na Figura 5), as organizações gerenciam a criação de seus produtos de forma contínua e sem sobrecarregar o time de desenvolvimento. É baseado em 3 princípios: visualizar o que precisa ser feito, limitar a quantidade de trabalho em progresso e melhorar o fluxo de trabalho ao criar um backlog de atividades priorizadas [Mclaughlin, 2017].

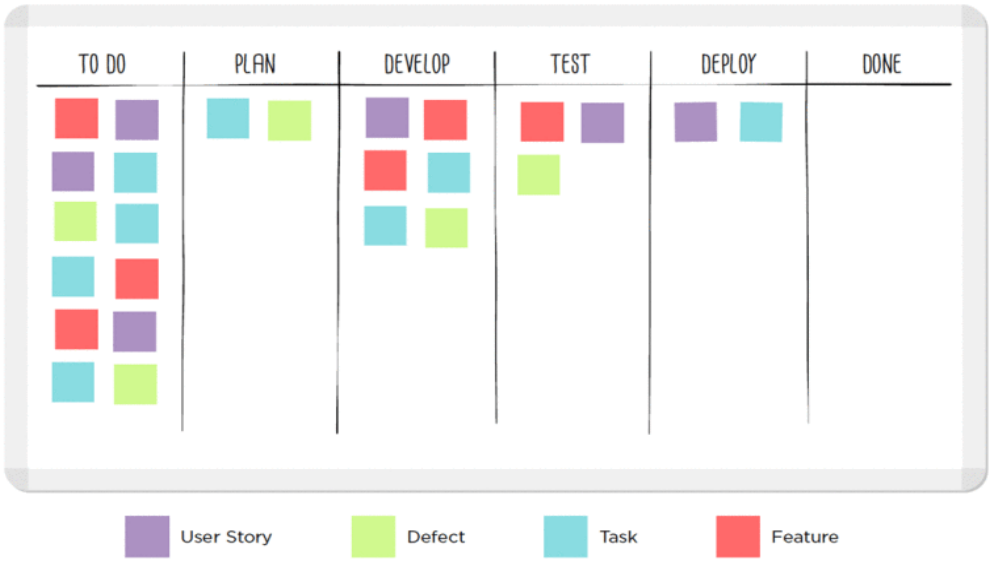

Figura 5: Quadro kanban [Fonte: Wikimedia Foundation, 2015]

2.5 O Pós-agilismo

Em meados de 2006, o termo Pós-Agilismo, atribuído simultaneamente a Jonathan Kohl e Jason Gorman [Skorkin, 2008], foi começando a emergir em discussões entre membros da comunidade de informática. Pós-Agilismo é uma nova era onde desenvolvedores e profissionais de TI são livres para usar as práticas que funcionam e descartar as que não funcionam, sem a obrigação de seguir metodicamente todas as regras de uma metodologia [Kohl, 2007].

A inspiração para a criação do termo remete à arquitetura e ao meio artístico, vindo através do movimento modernista e pós-modernista. O “Movimento Ágil” se equipara ao modernismo pela sua visão progressiva, porém com muitas regras em torno de seus valores e ferramentas [Leal, 2014], enquanto que no pós-modernismo (equiparado ao Pós-Agilismo), a percepção de falhas do movimento anterior ocasionou na junção ou alteração de seus processos, possibilitando uma criação mais livre [Kohl, 2007].

A análise dos relatos descritos anteriormente é um indício de que o Pós-Agilismo já é uma realidade entre as equipes de desenvolvimento. Os artigos de Block (2011) e Marchenko e Abrahamsson (2008) destacam a escolha das equipes em adaptar ou desconsiderar uma ou outra prática específica, ou combinar práticas oriundas de diferentes metodologias.

Alguns agilistas questionam a definição de Pós-Agilismo do ponto de vista da adaptação, argumentando que esta adaptação já se refere a um dos principais valores da metodologia ágil, portanto não se pode afirmar que essa característica é uma novidade do método. Como contra-argumento, os Pós-Agilistas distinguem entre ser “ágil” e ser “Ágil”, tendo no primeiro a capacidade de se adaptar a situações sociais ou comerciais, e no

segundo a adoção de uma metodologia como um dogma, seguindo as práticas fielmente sem reflexão de seu benefício [Myer, 2008].

Realizar uma transformação ágil em toda uma organização é algo desafiador e complexo, principalmente para aquelas que possuem estrutura e normas organizacionais voltadas à criação de um único plano (métodos tradicionais) para desenvolvimento de seus projetos [Rose, 2015]. Uma organização precisa saber quando efetivamente se tornou ágil. A literatura não define o ponto exato em que uma organização é considerada ágil: métodos ágeis com pouca orientação de como chegar a esse estado são apresentados, afirmando que a agilidade está mais relacionada com a forma com que se responde a diferentes cenários [Abi-Saber, 2016]. Mas a agilidade no escopo organizacional precisa de algo que possa ser desenvolvido, entregue e medido [Laanti, 2012], pois organizações são sistemas de adaptação complexos. Isso significa que indivíduos independentes e descentralizados interagem de maneira auto organizada guiados por um conjunto de regras simples e generalistas, tendo como objetivo gerar inovação e resultados [Highsmith e Cockburn, 2001].

O desenvolvimento ágil muitas vezes é rejeitado devido a razões culturais. A engenharia de sistemas, por exemplo, lida com design e construções de grandes sistemas que com frequência envolvem componentes de software, hardware e mecânicos. Engenheiros de sistemas afirmam duas grandes dificuldades em aplicar os princípios ágeis em projetos [Cockburn, 2007]:

- Por conter componentes mecânicos e de hardware, o tempo de feedback do design a implementação e avaliação é mais longo quando comparado a software;
- Os projetos são tipicamente grandes e possuem múltiplos subcontratados, tornando difícil a apresentação dos sistemas ao usuário final e cliente.

A adoção de práticas ágeis requer uma mudança cultural nas empresas, especialmente por parte dos diretores e gerentes. Muitos dos mecanismos de controle tradicionais são substituídos pela entrega do produto. Sendo assim, é importante a organização ter confiança nas habilidades e competências do time [Abrahamsson et al., 2002].

2.6 Maturidade Ágil

Modelos de maturidade são usados em engenharia de software com o objetivo de conduzir melhoria de processos de desenvolvimento. Esta melhoria significa criar produtos com mais qualidade, num processo de trabalho definido. Atualmente, a família de normas

ISO/IEC 330xx [ISO/IEC 33001:2015], Modelo Integrado de Maturidade em Capacitação (CMMI-DEV) [CMMI Product Team, 2010] e o Modelo de Referência MPS-BR para Software [SOFTEX, 2016] são referências para orientar melhorias em processos de software.

Quando uma equipe de desenvolvimento de software usa métodos ágeis, o processo de desenvolvimento tende a valorizar - como afirma o Manifesto Ágil [Beck et al., 2001] - indivíduos e interações mais que processos e ferramentas; software de trabalho mais do que documentação abrangente; colaboração do cliente mais do que negociação de contrato e resposta para mudanças mais do que seguir um plano. Neste contexto, a melhoria do software pode seguir os modelos tradicionais de referência, mas com algumas implicações. Como modelos tradicionais de referência usam definições e controle de processos para atingir a maturidade no desenvolvimento de software, isso geralmente implica que a agilidade pode ser prejudicada em níveis mais altos de maturidade [Paulk, 2001].

Alguns pesquisadores vêm propondo modelos de maturidade ágil [Schweigert et al., 2012]. Esses modelos adaptam as diretrizes de melhoria aos valores e princípios estabelecidos pelos métodos ágeis. Embora vários modelos tenham sido propostos com diferentes estruturas, apenas algumas delas são avaliadas empiricamente [Leppänen, 2013] e não se sabe se são aplicáveis em times de fato. Alguns modelos de avaliação da utilização de práticas ágeis propostos na indústria têm mostrado não serem eficazes em ajudar os adotantes das práticas a melhorarem sua agilidade [Adal, 2016].

As avaliações empíricas, no entanto, são essenciais para os pesquisadores entenderem a aplicabilidade das soluções que propõem, e a criarem um modelo de maturidade ágil que efetivamente ajude a melhorar a agilidade. Estes modelos devem estar alinhados com a finalidade de realmente serem usados na indústria [Adal, 2016].

A Bússola Ágil [Fontana, 2015] é uma ferramenta que permite avaliar a maturidade em equipes ágeis de desenvolvimento de software sem a necessidade de extensa definição e controle de processos. Define a maturidade ágil como um conjunto de resultados que são perseguidos pelas equipes e a avaliação é realizada com uma lista de verificação e reuniões para discussão.

A Bússola é um modelo criado com dados empíricos de nove diferentes equipes ágeis, como descrito em [Fontana, 2015]. O modelo define a maturidade ágil como busca de resultados. As equipes podem usar diferentes práticas, técnicas e métodos para perseguir esses resultados. E a avaliação é baseada no atingimento dos resultados pela equipe. A avaliação é feita com base em 7 categorias diferentes. Para a avaliação, cada um dos resultados é descrito por uma declaração e por dois ou três itens da lista de verificação. Os autores [Fontana, 2015] propõem que os membros da equipe avaliem juntos esta lista de verificação e discutam quais resultados foram parcial ou totalmente realizados. Essa discussão também pode levar a um plano de ação para melhoria da agilidade.

As 7 categorias e seus itens de verificação são apresentadas a seguir.

- A categoria “Aprendizado das Práticas” compreende os resultados que as equipes buscam quando elas decidem mudar a maneira de trabalhar. Os itens associados a essa categoria são: Tentando ser ágil, Aprendendo a ser ágil, Atuando nos processos de trabalho, Compreendendo a situação.
- A categoria de “Conduta da Equipe” representa como a equipe evolui o comportamento no uso de métodos ágeis. Os itens associados a essa categoria são: Equipe responsiva, Equipe confiante, Equipe assertiva, Equipe brilhante.
- A categoria “Ritmo de Entregas” mostra como as entregas evoluem à medida que a equipe amadurece. Os itens associados a essa categoria são: Terminando código, Gerando entregáveis, Entregando quase no prazo, Entregando no prazo.
- A “Descoberta das Features” compreende a definição e elicitação dos requisitos. Os itens associados a essa categoria são: Coletando requisitos, Descobrindo requisitos, Refinando requisitos.
- A categoria “Software” representa os resultados que as equipes realizam quando tentam melhorar o produto de software. Os itens associados a essa categoria são: Código fonte de alto-nível, Conhecimento das falhas, Software entregue de alto-nível, Codificação eficiente.
- O “Relacionamento com o Cliente” compreende os resultados perseguidos pelas equipes quando implementam práticas para melhorar o relacionamento com o cliente. Os itens associados a essa categoria são: Equipe conhecendo o cliente, Cliente conhecendo a equipe, Cliente confiante, Cliente parceiro.
- “Suporte da Organização”, representa a posição da organização no que diz respeito a adoção ágil. Os itens associados a essa categoria são: Movimentação ágil, Comprometimento ágil, Prioridade ágil, Negócio ágil.

A Figura 6 mostra um exemplo de preenchimento do template da Bússola, mostrando os resultados alcançados por uma equipe madura. Os retângulos podem ser pintados em cores mais claras ou escuras, possibilitando a visão dos resultados atingidos parcial ou totalmente pela equipe.

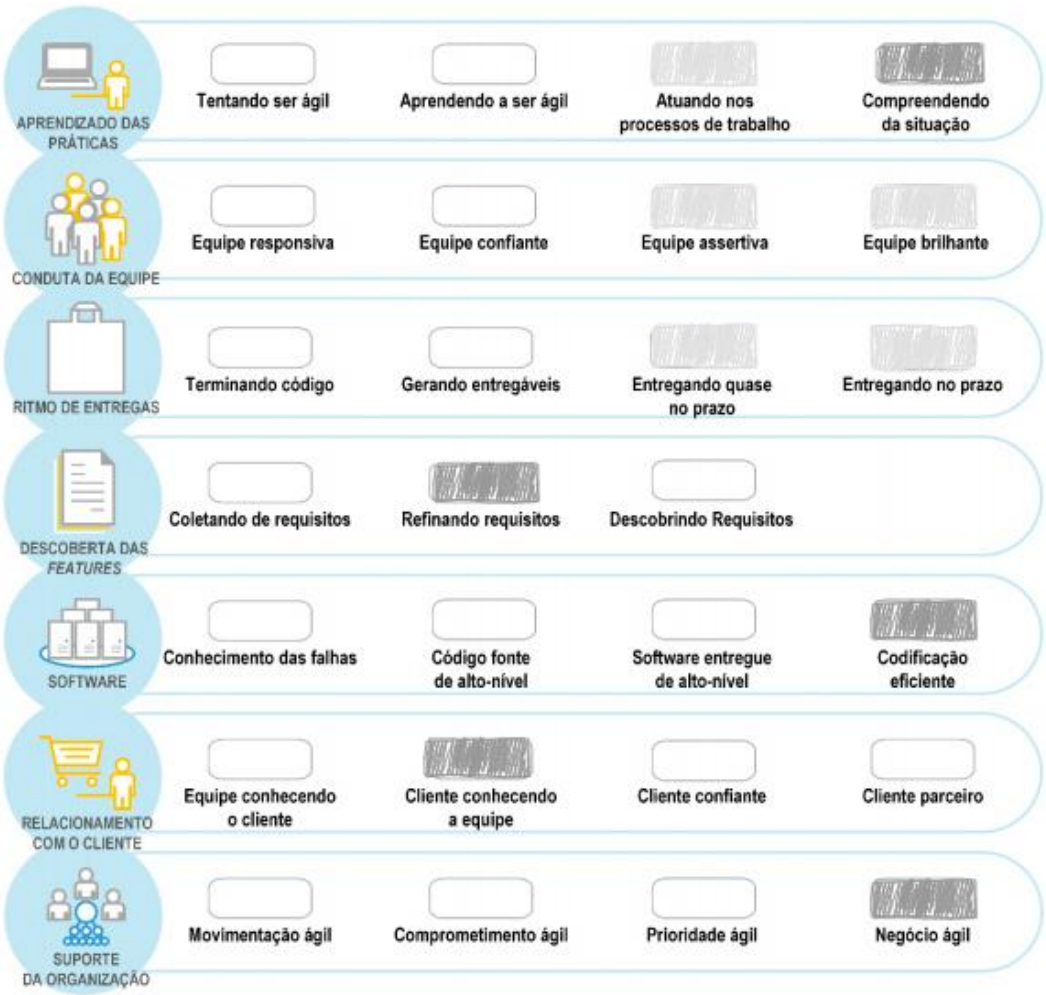

Figura 6: Resultados e categorias da Bússola Ágil [Fontana, 2015]

2.7 Considerações Finais

Os métodos ágeis nasceram como respostas às burocracias que ignoram as particularidades do desenvolvimento de software e pressionam as equipes para levar adiante projetos com compromissos irracionais assumidos de forma indevida. Ao definir metas curtas e priorizar a participação do cliente nas decisões de negócio, além de colocar um freio concreto às mudanças arbitrárias, os métodos ágeis resgatam o valor da engenharia de software frente aos embates burocráticos de certos níveis nas corporações. Entre outras virtudes, a decisão pela equipe deve essencialmente ser visível para todos. Em consequência, a transparência dos projetos ágeis é um de seus atributos mais valiosos e mais revolucionários [Boria e Rubinstein, 2013].

Os agilistas se consideram um movimento pró-métodos. Os que acreditam que a agilidade é contrária aos processos e às ferramentas, à documentação ou aos planos, estão

equivocados. O que procuram é que estas entidades estejam a serviço das atividades da equipe e não ao contrário. É equivocado pensar que ser ágil é não planejar, não seguir processos por mais ligeiros que sejam, não ter mais ferramentas que o ambiente de desenvolvimento interativo e algumas linguagens e não documentar as decisões, por exemplo. Os agilistas pensam que o planejamento detalhado não pode levar mais que umas poucas horas e deve envolver a equipe, que os processos são flexíveis, mas devem ter o acordo de todos aqueles que os colocam em prática, que a documentação deve ser fácil de manter e responder a uma necessidade orgânica do projeto e deve ser feita quando essa necessidade se manifesta e não como uma condição contratual depois que o produto estiver concluído [Boria e Rubinstein, 2013].

Pensando em analisar a utilização desses métodos, esse trabalho pretende verificar o nível de maturidade ágil em equipes de desenvolvimento, apresentando a experiência e o diagnóstico da aplicação da Bússola Ágil [Fontana, 2015], relatando as discussões e propondo soluções de melhorias para os processos das equipes de desenvolvimento participantes dessa experiência. Para alcançar o objetivo, utilizamos a metodologia de Grupo Focal [Romain, 2015], pois oferece uma abordagem rápida e eficiente para coletar experiências e *feedbacks* de um grupo de pessoas sobre um tema em comum, nesse contexto, o time de desenvolvimento ágil.

3. Diagnóstico de Maturidade

A abordagem dessa pesquisa foi utilizar a Bússola Ágil [Fontana., 2015] como uma ferramenta de avaliação de maturidade para equipes ágeis de desenvolvimento de software. A partir da avaliação da maturidade ágil, foram coletados *insights* para apoiar as sugestões de melhorias visando a tornar seus processos ágeis mais maduros. A metodologia de pesquisa escolhida foi o Grupo Focal [Romain, 2015], pela possibilidade de se obter dados e *insights* mais ricos com as restrições apresentadas [Kontio et al., 2004].

3.1 Explicando o método Grupo Focal

Para atingir esse objetivo, utilizou-se a metodologia de Grupo Focal para investigar a maturidade ágil de equipes de desenvolvimento de software.

O Grupo Focal oferece uma abordagem rápida e eficiente para coletar impressões, experiência e feedbacks de um grupo de pessoas sobre um tema em comum. O método tem um baixo custo de execução e pode prover bons resultados qualitativos, revelando insights e detalhes que seriam difíceis ou custosos de se capturar por outros meios [Kontio et al., 2004]. O Grupo Focal consiste em uma reunião de participantes selecionados com base no objetivo do estudo, onde os tópicos e discussões são planejadas e guiadas pelos moderadores. A reunião costuma ser presencial e o fator social contribui para que os participantes se sintam mais engajados e construam ideias a partir das contribuições de outra pessoa. Possibilitando, assim, que as contribuições geradas pelo grupo sejam mais ricas e profundas do que o somatório das suas entrevistas individuais [Johnson, 1994].

Após a definição do problema, os principais passos para a execução de uma pesquisa que utiliza o Grupo Focal são:

1) O planejamento da sessão, que define cuidadosamente os tópicos discutidos, os objetivos e o tempo dedicado a cada discussão;
2) A execução de uma sessão piloto para validar a dinâmica da sessão, seguida da revisão do planejamento inicial;
3) A seleção dos participantes, que deve buscar conteúdo rico para a discussão proposta;
4) A execução das sessões com os grupos definidos, onde os moderadores têm o papel de otimizar as discussões no tempo máximo definido;
5) A análise dos resultados, após a compilação da informação coletada - gravações de áudio e vídeo e anotações de moderadores e participantes.

A figura do moderador é central nas reuniões de grupo focal e pode ser ainda mais desafiador no domínio de engenharia de software, devendo possuir experiência nos assuntos discutidos e saber utilizar perguntas não intrusivas, de facilitação, para estimular e direcionar as discussões [Kontio et al., 2004].

3.2 Planejamento do diagnóstico

O planejamento das sessões foi focado em garantir que o tempo limite combinado com os times fosse respeitado, sem deixar de coletar as impressões de cada um sobre as dimensões de maturidade abordadas. As sessões consistiram da introdução, da leitura e discussão sobre as dimensões e os itens que melhor representam a equipe, além das considerações finais, que forneceram um espaço para feedback dos participantes sobre a reunião e a aplicabilidade da Bússola. Os áudios e anotações foram coletadas, com o consentimento prévio dos participantes e foram transcritos para facilitar a análise dos pontos discutidos. Porém, não incluem reações e comunicações não relacionadas à reunião. Após a análise das discussões, foram gerados o desenho da Bússola e as sugestões de melhorias para cada equipe participante.

Para atender a proposta, e com o objetivo de maximizar a contribuição para a experiência, as equipes participantes foram escolhidas observando-se os critérios considerados como desejáveis para a proposta deste trabalho:

- As equipes deveriam ser focadas em desenvolvimento de software, onde os entrevistados cumprissem diversos papéis do ciclo de vida do desenvolvimento como os de analista, programador, designer, gerente, etc.;
- As equipes selecionadas deveriam se denominar equipes ágeis, e citar o uso de métodos ou práticas ágeis no seu desenvolvimento;
- As equipes deveriam ser pequenas (de 3 a 6 pessoas), de tal forma que fosse viável reunir toda a equipe em uma única sessão de grupo focal, com a limitação de tempo estabelecida para as sessões;
- Não sendo possível realizar a sessão com toda a equipe, entre os seus representantes deveriam estar os mais experientes e com mais conhecimento sobre os processos de trabalho do time.

A disponibilidade de agenda com as equipes mostrou-se um desafio, e foi necessário ajustar o escopo e o cronograma da experiência para se adequar aos cancelamentos e remarcações solicitadas pelos grupos. Como consequência da dificuldade em conciliar-se as agendas, foram realizadas reuniões com 4 das 6 equipes de desenvolvimento de software planejadas inicialmente. Por conta da disponibilidade dos membros de cada equipe, os representantes dos times das organizações A e B foram selecionados apenas para a sessão

piloto (que foi encarada como uma reunião teste), já que somente os líderes dos seus times puderam comparecer em tempo hábil. As organizações C, D e E participaram da segunda e terceira sessões que foram realizadas após a compilação das lições aprendidas com o piloto - e revisão da dinâmica proposta para as reuniões.

Para melhor entendimento do contexto em que as equipes avaliadas trabalham, segue uma breve descrição das organizações, seus negócios, e desafios encontrados:

- A organização A é uma empresa global de tecnologia de recrutamento online com sede no Vale do Silício. A equipe avaliada faz parte do time de desenvolvimento interno, responsável pelo desenvolvimento e melhorias no principal produto da empresa.
- A organização B é uma empresa de comércio digital que está entre os grandes *players* do varejo digital brasileiro. A equipe avaliada faz parte da área de pesquisa e desenvolvimento, entregando soluções para melhorar uma série de produtos, podendo ir da experiência do usuário até a revisão dos principais algoritmos presentes nos sistemas.
- A organização C é uma consultoria multinacional, com diversos projetos de tecnologia em grandes empresas brasileiras. E a organização D é uma grande empresa do ramo da indústria alimentícia brasileira, cliente da consultoria C. A equipe avaliada no diagnóstico é um time misto composto por membros das empresas C e D. Essa equipe forma uma célula de desenvolvimento de ferramentas para apoiar a automação e melhorias de processos da empresa D.
- A organização E é uma *startup* de educação, com foco em tecnologia. Possui uma grande base de usuários, e formas inovadoras de apresentar o seu conteúdo didático, sempre apoiando-se em soluções tecnológicas. A equipe avaliada é o time de desenvolvimento e suporte dos produtos dessa empresa.

Antes de seguir com a execução das reuniões, uma sessão de teste foi planejada para que os moderadores pudessem avaliar o tempo e a dinâmica proposta, além de servir de prática para a condução da reunião. Após essa sessão piloto, foram revisadas a alocação do tempo para as discussões, e a abordagem dos moderadores para otimizar a coleta dos dados e o direcionamento das discussões.

3.3 Execução

A descrição da execução das reuniões com os times de desenvolvimento é descrita nas seções a seguir.

3.3.1 Sessão Piloto

A *sessão piloto* foi realizada com os líderes dos times das organizações A e B. Ambos possuem mais de cinco anos de experiência trabalhando em equipes ágeis, e por mais que fossem os únicos representantes de cada time, foram capazes de fornecer conteúdo rico durante a reunião. A reunião aconteceu presencialmente, e seguiu a sequência de tópicos planejada. Durante a introdução, foram apresentados os conceitos de maturidade ágil, a proposta do trabalho e a visão geral da Bússola ágil, entregue impressa para facilitar o acompanhamento por parte dos participantes. Após a discussão, cada dimensão, e seus resultados foram lidos em voz alta e discutidos pelos participantes, que apontaram os resultados atingidos total ou parcialmente por suas equipes, justificando suas escolhas aos moderadores. Quando a justificativa não foi clara para algum dos moderadores ou para o outro participante, foram realizados os questionamentos, revisando o entendimento do grupo sobre o que é proposto em cada item, e pedindo explicações mais detalhadas a quem estava expondo seu processo. O detalhamento das discussões documentadas no piloto, assim como os resultados da análise, e o desenho da Bússola para os dois times representados, estão descritos no Apêndice 1. O tempo total despendido na reunião foi de 35 minutos, sendo realizado abaixo dos 45 minutos planejados inicialmente. A finalização antes do prazo estabelecido, mesmo com as discussões que se mostraram profundas em alguns itens, indicou que a leitura consumiria menos tempo do que o estimado, abrindo a possibilidade para mais questionamentos e aprofundamento nos temas do questionário. Além disso, durante a reunião, o papel de ler o questionário em voz alta passou organicamente dos moderadores para os participantes, que se mostraram engajados com as reflexões propostas. Sendo assim, o planejamento para as reuniões seguintes foi revisado, com o objetivo de utilizar esse tempo extra, questionando ainda mais os participantes.

3.3.2 Segunda Sessão

A *segunda sessão* foi realizada com a célula de desenvolvimento de automatizações, composta por três consultores da empresa C e um analista da empresa D. Um dos consultores da empresa C participou da reunião remotamente, utilizando uma ferramenta de reuniões de conferência para otimizar a comunicação entre ele e o grupo presencial. Não foram percebidas nem relatadas dificuldades de comunicação significativas por conta dessa organização. O questionário foi entregue impresso aos participantes presentes e enviado em formato digital para o participante remoto enquanto as preparações para a chamada de voz e as gravações eram realizadas. Em seguida, o moderador deu início à introdução da reunião. Na segunda sessão, a introdução foi otimizada para focar apenas nos conceitos mais relevantes à aplicação da dinâmica proposta, explicando rapidamente o conceito de

maturidade ágil; a visão geral da Bússola ágil e das suas dimensões; e o objetivo de fornecer tanto uma reflexão quanto uma análise dos processos do time, baseando-se no modelo de maturidade da Bússola. Foi possível concluir a introdução, levando em consideração as dúvidas dos participantes, em apenas 5 minutos. Então, deu-se início à leitura e discussão dos itens de cada dimensão do questionário. Após a leitura, os participantes foram estimulados a chegar em um consenso sobre quais dos resultados foram alcançados total e parcialmente. O questionário é apresentado, na íntegra, no Anexo 1.

Os áudios foram transcritos e tanto comentários quanto às reações de concordância ou discordância a respeito das características descritas na dinâmica foram levadas em consideração para a análise dos resultados. As sugestões de alteração nos processos dos times para atingir níveis mais altos de maturidade ágil são apresentadas após o desenho das Bússolas preenchidas por cada equipe. O resumo das discussões de cada dimensão é apresentado a seguir:

- *Aprendizado das Práticas:* Assim como no piloto, a discussão da primeira dimensão foi a que gerou mais ruído, com bastante conteúdo sem valor para o diagnóstico, já que os participantes fugiram um pouco do assunto e ficaram confusos com a possibilidade de marcação parcial de vários resultados. Os participantes discutiram sobre algumas das dificuldades para se colocar em prática um processo de desenvolvimento ágil, na empresa tradicional, e como nenhum método foi aplicado à risca, desde o início do projeto. Então, discutiram sobre o que é esperado em cada nível, e o entendimento dos resultados, pelas suas descrições. Refletiram sobre a diferença de conhecimento técnico de métodos ágeis entre os membros da equipe, que é uma característica forte do segundo resultado “aprendendo a ser ágil”. A equipe ficou em dúvida sobre a marcação dos três primeiros resultados e chegou ao consenso dos itens 2 “Aprendendo a ser ágil” e 3 “Atuando nos processos de trabalho”, com a justificativa de que já superaram o nível em que ainda está se entendendo o que são métodos ágeis e como podem entregar valor ao cliente a partir deles, já que conseguem avaliar e aplicar práticas ágeis para grande parte do processo. Possuem boa autonomia para definir seu processo, mas ainda não utilizam métricas para medir o seu desempenho ou tomar decisões com base nesses números para ajustar o processo de acordo.

- *Conduta da Equipe:* Logo após a leitura do primeiro resultado “equipe responsiva”, os participantes concordaram em eliminá-lo das opções por estar muito distante da sua realidade. No segundo resultado, “equipe confiante”, os participantes o excluíram pela citação da necessidade no momento presente, de um Scrum Master próximo para

serem produtivos. Em seguida, foram discutidas as mudanças no processo de desenvolvimento ao longo dos 10 meses em que essa equipe o acompanhou, e como a equipe seria bem representada pelo nível “Equipe confiante” no início do ano, devido à forte presença dos gestores, limitando a autonomia da equipe tanto para a tomada de decisões de projeto quanto para as decisões do processo de desenvolvimento. A partir da leitura e discussão dos níveis seguintes, compararam as características descritas em “equipe assertiva” e “equipe brilhante”, eliminando a última após a descrição de foco em excelência técnica. Chegaram ao consenso de que a opção que melhor os representa hoje, é o terceiro resultado: “equipe assertiva”.

- *Ritmo de Entregas:* Após a leitura do nível “Terminando código”, a equipe discutiu sobre o tamanho das suas histórias e os entregáveis que muitas vezes são trabalhados em vários *timeboxes*. Concordaram que entendem bem o que possui e não possui valor para o cliente, devido à proximidade com o cliente e com o negócio. Concordaram também com a parte de testes geralmente serem manuais, mas voltaram a discordar sobre o atraso nas entregas ao final de cada sprint. Foi um consenso entre o time que a necessidade de possuir uma infraestrutura para *build* automatizado não se aplica ao contexto do desenvolvimento das automatizações. E que a equipe costuma entregar no prazo que se propõe, mesmo que o entregável seja atômico e maior do que o tamanho de um *Sprint*. Ao final da discussão, e da revisão dos outros níveis, a equipe ficou dividida entre estar caracterizada na terceira e quarta opções: “entregando quase no prazo” e “entregando no prazo”, respectivamente.

- *Descoberta das Features:* Assim que o texto do nível mais baixo “Coletando requisitos” foi lido, todo o time concordou em pular para a próxima opção, porque a equipe acredita se sentir bem confortável e flexível às mudanças de requisitos durante o projeto. Além de o projeto se basear na prospecção e repriorização de demandas semanalmente, sendo o oposto da descrição apresentada no primeiro nível. Sobre o nível “Descobrindo requisitos”, o time acredita que os requisitos são definidos em um nível de detalhe razoável. Mas admite que se utilizam pouco de provas de conceito para validar hipóteses e abordagens inovadoras. E concordaram com o restante da descrição, que explica a definição de requisitos a cada Sprint, e a identificação de MVPs (produtos mínimos entregáveis) nas estórias de usuário. O próximo nível (“Refinando requisitos”) requisitava do time possuir iniciativas sistemáticas para a garantia da qualidade dos requisitos e sua conformidade com a necessidade dos clientes. A equipe não possui nada parecido, que possa ser visto como uma prática sistemática, então recusaram o terceiro nível, marcando apenas o segundo.

- *Software:* Durante a introdução, onde se apresentou o conceito de “código fonte de alto nível”, a equipe discutiu brevemente sobre o tema, da preocupação com o código e as iniciativas que possuem para garantia de qualidade do código. Já no primeiro nível “conhecimento das falhas”, a equipe concordou com alguns de seus pontos principais: gastar tempo em *sprints* corrigindo falhas de *sprints* anteriores e gastar tempo das *sprints* resolvendo problemas de *build* e integração de código. O nível “Conhecimento das falhas” foi aceito após levantar-se a questão do que seria *build* e integração automatizados no contexto da célula de automatização. Chegou-se à conclusão de que existe o tempo gasto com a configuração do ambiente para o *deploy*, que muitas vezes esbarra em problemas ou imprevistos. Após a leitura dos níveis acima (“Software entregue de alto-nível” e “Codificação eficiente”), a equipe chegou à conclusão de que o primeiro representa melhor o estado atual do processo de trabalho.

- *Relacionamento com o Cliente:* Foi discutida a diferença do relacionamento com cliente em um projeto como da célula com o relacionamento de uma fábrica de software, e como projetos como o da equipe são privilegiados pelo contato próximo com o cliente e as áreas de negócio. Novamente, foi discutida a mudança nos processos ao longo dos meses. Onde, foi identificada uma diferença grande entre o relacionamento com o cliente no início e no final do ano. Os clientes mudaram, e a postura e proximidade, também. O relacionamento mais atual, agora não é mais considerado como “cliente parceiro”. Mesmo assim, atinge totalmente, o nível “Cliente confiante“ devido à percepção de confiança do cliente na equipe, flexibilidade do cliente com as entregas, transparência da equipe e sua participação na definição e priorização das demandas.
- *Suporte da Organização*: Após a leitura de cada nível, a equipe discutiu sobre qual seria a organização a ser considerada nessa dimensão. Tomando apenas a organização C como referência, o consenso do time foi de que esta organização cumpriria todos os critérios descritos no nível de negócio ágil. A justificativa em optar por avaliar a organização C em vez da D foi que a organização C participou muito mais da definição, e do apoio às práticas ágeis adotadas, do que a organização D, que apenas aceitou, passivamente, que a célula se organizasse como desejasse. Uma citação que reforça a escolha da equipe foi: “*a empresa está sempre focada em maximizar o valor entregue reduzindo o desperdício em esforços que não gerem valor pro cliente, seja o projeto que for*”.

3.3.3 Terceira Sessão

A *terceira sessão* foi realizada com a equipe de desenvolvimento da organização E, responsável, principalmente, pela manutenção e melhoria nos produtos *web* e *mobile* utilizados para fornecer conteúdo aos clientes da *startup*. Os quatro membros que compõem a equipe participaram presencialmente da terceira sessão, que seguiu a mesma estrutura da segunda. O questionário impresso foi entregue aos participantes para que acompanhassem a leitura e o moderador iniciou a introdução, focando na motivação para o diagnóstico, e relacionando as dimensões do questionário com os princípios ágeis correspondentes. Durante a introdução, os participantes já forneceram um *feedback* positivo quanto a experiência de refletir sobre seus processos e adiantaram alguns dos problemas que seriam discutidos mais profundamente ao longo do questionário. Foi iniciada então, a leitura das descrições e dos níveis das dimensões.

- *Aprendizado das Práticas*:

  Durante a leitura surgiram dúvidas quanto aos critérios, quanto à necessidade de se atingir os níveis progressivamente e quanto ao que seria o nível de maturidade em relação a um modelo tradicional, que não era o foco da discussão proposta. Após a leitura, a equipe discutiu sobre quais critérios melhor refletiam o seu estado atual. Os participantes expuseram que constantemente revisam e adaptam seus processos de trabalho, sempre o encarando como algo não finalizado e que precisa desses ajustes para buscar eficiência. O principal questionamento que se fizeram foi se conseguiam ou não explicar os acontecimentos a partir dos indicadores que são coletados nos *sprints.* Chegaram à conclusão que atualmente não é possível chegar a conclusões com os dados que são coletados. Chegaram, inicialmente, ao consenso que o nível “atuando nos processos de trabalho” é a opção que melhor representa o estado atual da equipe. Antes de passar para a próxima dimensão, o moderador também questionou aos participantes sobre o conhecimento que cada um considera possuir para tomar decisões sobre as mudanças de processo. As respostas do time tornaram evidente a dependência do líder técnico do time para as tomadas de decisão, e fizeram o time refletir sobre quanto o nível “Estamos aprendendo fazendo” descreve o seu dia-a-dia. Pela falta de conhecimento técnico e a característica muito presente de testar e aprender empiricamente, também poderiam escolher essa opção. Adiantaram discussões sobre os desafios de otimizar o valor em um cenário de startup com pouquíssimo recurso. Falando um pouco também sobre outros aspectos do processo como as mudanças na etapa de previsão e pontuação das histórias; a

qualidade do código e o *trade-off* para manter o ritmo de entregas, apesar dos bugs e correções. São assuntos abordados em mais detalhes nas dimensões seguintes.

- *Conduta da Equipe*:

  Após a leitura e descrição dos níveis, a equipe debateu sobre o papel do líder da equipe, que desempenha o papel de *Scrum Master*, faz a gestão da equipe e é, também, o líder técnico da empresa. A grande dúvida foi se deveriam ou não considerar esse líder como parte da equipe, ou como a figura da gestão na hora de medir a autonomia do time. O entendimento de todos foi de que esse líder técnico é a principal referência da gestão, mas ele faz parte da equipe, realizando inclusive tarefas operacionais com a mesma. Os participantes optaram pela marcação parcial dos níveis "equipe assertiva" e "equipe brilhante", tendo em vista que cumprem várias das categorias apresentadas, inclusive as relacionadas à autonomia e o empoderamento. Esse foi o sentimento da equipe mesmo com uma figura de gestão forte, tão presente no seu dia-a-dia. Alguns pontos não são cumpridos totalmente pela equipe, como o fato de não terem políticas definidas para proteger o seu trabalho. Indicaram iniciativas para buscar essa proteção, como a mudança recente no tamanho das *sprints,* mas nenhuma política definida e combinada com as áreas que são clientes internas. Discutiram, também, sobre como a evolução dos seus processos não deu valor a alguns dos pontos levantados, como a excelência técnica e conhecimento das práticas.

- *Ritmo de Entregas*:

  A partir dessa dimensão, um dos participantes passou a conduzir a leitura, como sugerido pelo moderador. O participante responsável pela leitura e todo o restante do grupo passaram a se expor mais do que nos quesitos anteriores, reagindo e comentando ao longo da leitura. Os pontos "ter código finalizado ao fim da *sprint* e não entregue" e "os testes serem geralmente manuais" fizeram alguns participantes darem risadas, incitando brincadeiras relacionadas às falhas no dia-a-dia. Discutiram rapidamente sobre o valor ganho com o tamanho pequeno e homogêneo das histórias, e como podem melhorar sua previsibilidade com entregáveis pequenos. O grupo mostrou indecisão entre os níveis "*gerando entregáveis"* e "*entregando quase no prazo".* A fala de um dos participantes foi decisiva para a escolha do grupo:

  "Eu acho que é mais a dois (gerando entregáveis) porque… essa frase aqui do: 'ambiente tecnológico não dar o suporte necessário para integração de código e build ágil' eu acho bem forte porque.. é o que acontece. E 'o teste geralmente é

manual' - geralmente, não, sempre! O nosso teste é sempre manual. Então… essas frases aqui são muito verdade".

- Desenvolvedor da organização E.

Em seguida discutiram sobre quanto consideram importante ter os requisitos não só entregues, mas rodando ao final da *sprint*. Expuseram que não planejam a *sprint* para ter código em produção e consideram o entregue como o que chegou ao status revisado, por conta da organização atual dos processos de DevOps[1]. O conhecimento do que entrega valor ao cliente foi um ponto que o time alegou possuir.

- *Descoberta das Features*:

  Após a leitura, os participantes discutiram sobre o que seriam e se possuiriam iniciativas sistemáticas para garantir a qualidade dos requisitos como descrito no último nível "*Refinando requisitos*". O moderador explicou como a prática de *grooming*[2] auxilia nesse sentido e recuperou a discussão sobre o uso do *planning poker*[3], citada durante a introdução, questionando se o grupo considerava que a prática contribuía para deixar mais claros os requisitos para todos. A adoção recente do *planning poker* foi elogiada pelos participantes já que, como citado, obrigou a equipe a "trocar entre si o que cada um pensa sobre aquele problema, o que gera uma previsibilidade muito maior do que o *feeling*", e sobre como a experiência foi tão positiva que gostariam de utilizar a dinâmica de *planning poker*, mesmo se não fossem aproveitar os pontos para as métricas, apenas pelo ganho observado no melhor entendimento dos requisitos e da solução. Ainda sobre o ponto de "qualidade dos requisitos", foi discutido sobre como essa melhoria no entendimento pela equipe afeta a sua qualidade, e quais outros aspectos da qualidade do requisito precisam ser observadas. Um dos participantes questionou sobre quanto seus requisitos garantem que o que se propõe a ser entregue está em conformidade com o que o cliente precisa. Então, foram citadas e explicadas as práticas de *costumer development* e as reuniões semanais de *roadmap* de produto realizada por parte da equipe para questionar se as principais necessidades do cliente estão sendo contempladas. A partir dessa reunião,

[1] O termo DevOps deriva da junção das palavras "desenvolvimento" e "operações", sendo uma prática de engenharia de software que possui o intuito de unificar o desenvolvimento de software e a operação de software.

[2] Grooming é uma reunião para refinamento do Backlog. A palavra Grooming em inglês significa cuidar da aparência, manter limpo e arrumado. Uma reunião de Backlog Grooming deve ser realizada próximo ao final da iteração, garantindo assim, que o Product Backlog esteja sempre pronto para a próxima.

[3] Planning Poker é uma técnica baseada no consenso para estimar o esforço a ser realizado em uma tarefa.

a equipe define *features* de *mockups* e protótipos para que ideias e soluções propostas sejam testadas e validadas. E só após essa validação, as histórias são escritas e priorizadas nos *sprints*, que normalmente preveem a refatoração das *features* desenvolvidas como protótipos, e que foram construídas rapidamente, sem preocupação com os padrões e boas práticas da equipe. Os participantes também resumiram a discussão recente sobre o detalhamento dos requisitos. Nessa discussão - realizada alguns meses antes desse diagnóstico - chegaram a conclusão de que preferiam correr o risco de ter requisitos mal especificados, mas que fossem simples e entregues rapidamente, já que contam com uma comunicação eficiente entre as áreas de tecnologia e design para solucionar rapidamente as dúvidas durante o desenvolvimento. O grupo, enfim, optou por marcar parcialmente os níveis “descobrindo requisitos” e “refinando requisitos”.

- *Software*:

  A descrição dos níveis foi lida, e dessa vez fez-se um momento mais silencioso, onde os participantes examinaram alguns pontos individualmente. De forma divertida, um dos participantes sugeriu ao restante do grupo que nem lessem os dois níveis mais altos “Software entregue de alto-nível ” e “Codificação eficiente”. O que produziu risadas e contou com a aprovação dos demais. Comentaram sobre o segundo nível “Conhecimento das falhas” dar a impressão de ser um estado pior do que o descrito em “Código fonte de alto-nível”. O moderador pediu ao grupo para desconsiderar a ordem apresentada, explorando apenas as características descritas. Um dos participantes comentou sobre a sua preocupação com o código, e ressaltou que mesmo existindo essa preocupação na equipe, quando realizam a revisão ou quando uma história pede melhorias de algum trecho do código, dificilmente realizam qualquer tipo de refatoração. A preocupação é com o código imediatamente entregue e não com o que já está em produção. O restante do grupo concordou com essa visão e optaram por marcar a primeira opção “Conhecimento das falhas”.

- *Relacionamento com o Cliente*:

  Antes de iniciar a leitura da dimensão, o moderador pediu para que o grupo se atentasse a quem seria o cliente referenciado no texto. Assim que a descrição do primeiro nível, “equipe conhecendo o cliente”, foi lida, os participantes discutiram sobre quem seriam os clientes da equipe de desenvolvimento dentro da organização, já que inicialmente só enxergavam o cliente do produto (usuários) como seu cliente. Chegaram à conclusão de que qualquer *stakeholder* que participa das decisões de

produto, definindo novas *features* e melhorias poderiam ser vistos como clientes, como as áreas de marketing, financeiro e conteúdo.

As características apresentadas em cada nível foram discutidas baseando-se na relação da equipe com essas áreas. Revisando as características apresentadas, os participantes consideraram que possuem conhecimento e abertura para contribuir com o negócio, e que existe uma relação de parceria entre a equipe e as áreas que são clientes, com a percepção de responsabilidade compartilhada com elas. O nível selecionado foi o último: “cliente parceiro”.

- *Suporte da Organização*:

  As descrições dos níveis foram lidas e o grupo respondeu de bate-pronto: “negócio ágil”. Explicaram que consideram que a startup apresenta todas as características descritas nesse nível. Vale ressaltar a citação de um dos participantes:

  “A gente já nasceu ágil. A empresa toda, não só a parte de TI”.

  - Desenvolvedor da Organização E.

3.4 Análise

Como proposto no trabalho, os pesquisadores elaboraram um diagnóstico complementar à Bússola ágil, baseado nos principais pontos positivos e negativos expostos pelas equipes nas reuniões.

3.4.1 Segunda Sessão - Diagnóstico

Para a *segunda sessão*, envolvendo a equipe das organizações C e D, a Bússola preenchida durante as discussões é apresentada abaixo.

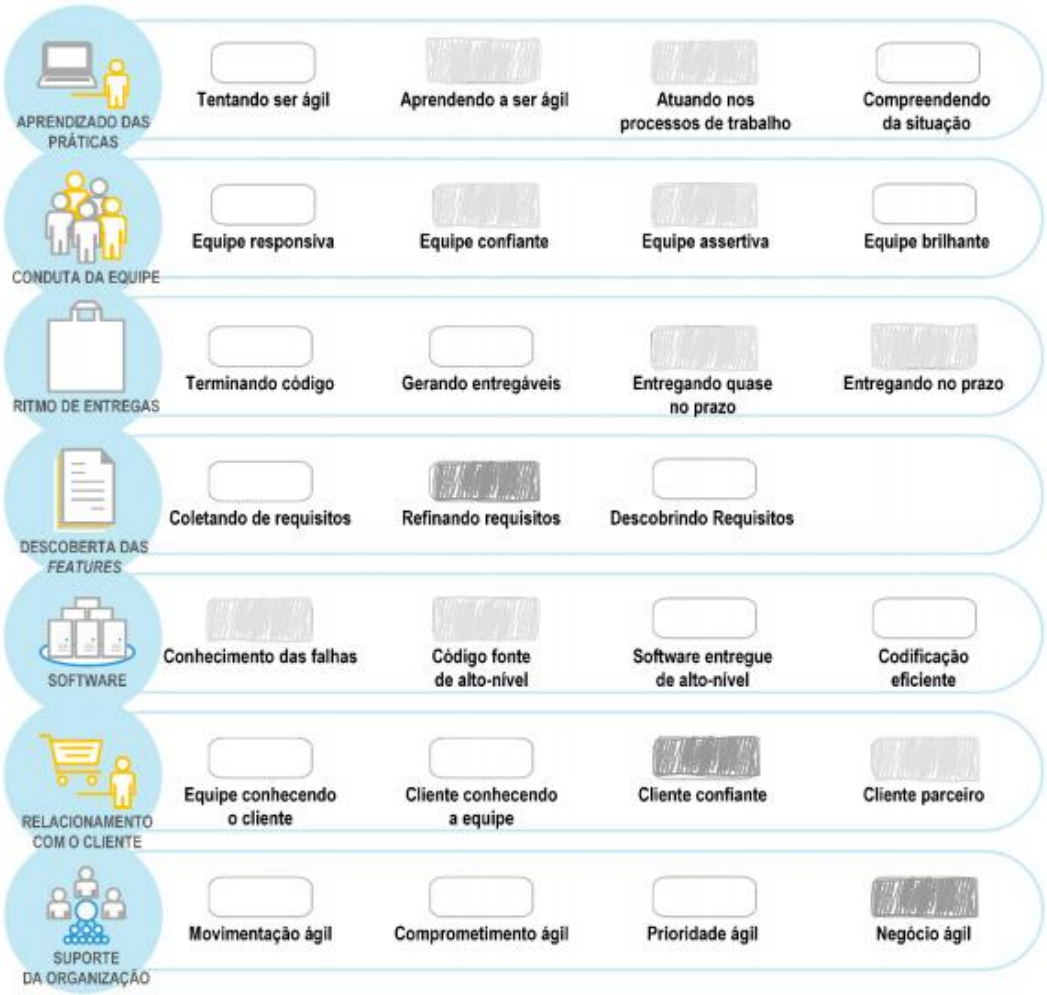

Figura 7: Desenho da Bússola para a equipe da segunda sessão

Os principais destaques da equipe em relação ao nível de maturidade encontrado foram: o ritmo de entregas, o relacionamento com o cliente e o suporte da organização. No quesito relacionamento com o cliente, a equipe se beneficia do contexto onde está inserida, que implica no trabalho muito próximo ao cliente, o que facilita a construção de uma relação de confiança entre ele e a equipe.

Devido à mudança dos clientes e dos gestores da organização D, o relacionamento de parceria evidenciado pela responsabilidade compartilhada entre equipe e cliente não é mais observado. Ainda assim, a equipe demonstrou ter confiança do cliente na equipe, que fica evidente na participação da equipe na definição dos requisitos. Se o time não prezasse pela transparência do seu processo de trabalho, e do *feedback* constante da evolução do andamento das entregas, seria muito difícil atingir esse nível de maturidade.

A organização C foi dita como negócio ágil por ser uma empresa focada em maximizar o valor entregue reduzindo o desperdício em esforços que não gerem valor para o cliente, em qualquer um de seus projetos. Possui processos enxutos e sua estratégia é muito focada nas

pessoas. Existe o estímulo para que todos os seus projetos sejam ágeis, por mais que o nível de maturidade ágil seja difícil de atingir pelas restrições apresentadas nos processos das organizações onde os projetos são executados, como apontado na discussão desse quesito. O ritmo de entregas também foi ressaltado, positivamente. Um dos fatores mais positivos ressaltados pela equipe foi o entendimento do valor gerado para o cliente. Isso acontece porque existe uma proximidade tanto do cliente quanto do negócio, e o time é estimulado pela organização C a buscar oportunidades, sempre pesando o valor agregado ao negócio do cliente. O time também deixou claro que consegue entregar ao final da sprint, mas muitas vezes abre mão da qualidade do que é entregue para cumprir o prazo. As citações "*o foco sempre foi em colocar pra rodar*" e "*a gente não espera o negócio ficar perfeito pra entregar*" evidenciam essa postura da equipe, que preza mais pela entrega de algo imperfeito, porém entregue rapidamente, do que entregar uma solução tecnicamente melhor (mais rápida, estável e consistente).

Como não existe uma cultura forte de testes para garantir a qualidade dos requisitos, a postura de entregar algo funcional, para o problema mais evidente, mas sem uma preocupação mais incisiva com a excelência da entrega, acaba impactando o ritmo das entregas em sprints futuros. E essa necessidade recorrente de corrigir bugs e problemas de configuração de requisitos já entregues impede que a equipe alcance o nível mais alto de maturidade nesse quesito. E impacta ainda mais a performance da equipe na dimensão de Software, que foi a pior avaliada.

Além da ausência de testes para garantir a qualidade do que é entregue, outros pontos de melhoria se fossem adotados forneceriam um nível de maturidade mais elevado nas outras dimensões. Em relação ao aprendizado das práticas, a equipe possui algum conhecimento e realiza uma adaptação das práticas ágeis ao contexto do seu projeto, mas ainda concentra o conhecimento técnico sobre métodos ágeis em poucos membros da equipe e não utiliza métricas para medir e acompanhar o desempenho ao longo dos sprints. Essa análise quantitativa também serviria para evidenciar problemas e servir de fundamento para mudanças em seu processo de trabalho. Por mais que a equipe possua autonomia, e se sinta empoderada para tomar decisões, não atingiu o último nível por não possuir a característica de aprendizado constante e foco em excelência técnica.

E por último, o nível mais alto de descoberta das *features* também não foi alcançado. Faltaram ao time iniciativas sistemáticas para melhorar seus requisitos e garantir se aquilo que quer se desenvolver atende o que é esperado pelo cliente. Práticas como a de *grooming* ou até a confirmação do entendimento, fornecendo uma descrição simples do requisito, e a aprovação por parte da área de negócio serviriam para obter-se um resultado melhor neste critério.

Em resumo, as principais iniciativas sugeridas para que o time atinja um maior nível de maturidade ágil são:

- Implantação de processos de testes como forma de garantir a qualidade do que é entregue;
- Dedicar maior tempo ao estudo de práticas, com o foco na excelência técnica e no aprendizado de métodos ágeis, que embasariam a equipe para tomar melhores decisões sobre seu processo de trabalho;
- A utilização de métricas para acompanhar, analisar e fornecer transparência da velocidade da equipe, evidenciando possíveis melhorias no seu processo de desenvolvimento;
- A implantação de processos enxutos para garantir a conformidade dos seus requisitos junto ao cliente e ao negócio.

3.4.1 Terceira Sessão - Diagnóstico

É apresentado abaixo o desenho da Bússola preenchida a partir das discussões da terceira reunião, que contou com a equipe da organização E.

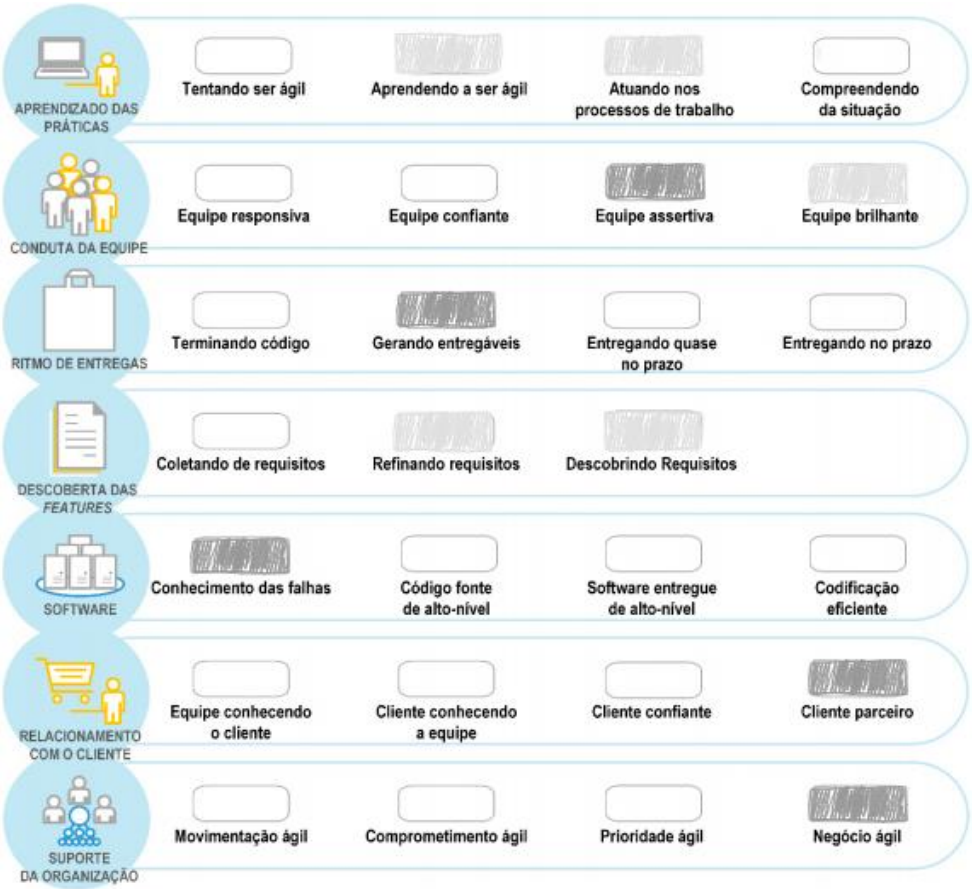

Figura 8: Desenho da Bússola para a equipe da terceira sessão

As dimensões “Relacionamento com o Cliente”, “Suporte da Organização” e “Conduta da Equipe” foram os destaques positivos da equipe em relação aos critérios apresentados. O perfil da empresa, que é uma *startup* construída com inspiração em conceitos de agilidade, serviu de motivação para que a equipe a classificasse como um “negócio ágil”. E justamente por contar com processos enxutos, transparência e empoderamento dos membros, contribuiu para que a equipe de desenvolvimento se sentisse capaz de opinar no negócio, mesmo fora do domínio da tecnologia. Dado que a empresa fornece esse ambiente para os membros, também é esperado para o quesito “Conduta da Equipe” que o time de desenvolvimento sinta autonomia para propor e testar modificações em seu processo de trabalho, o que foi refletido na visão do time nesse quesito. Para atingir completamente o nível máximo (“Equipe Brilhante”) nessa dimensão, a equipe precisaria ter o foco em excelência técnica, como descrito nas características desse nível. A escolha da equipe despriorizando a excelência técnica no seu processo de software traz impactos no seu dia a dia, como ficou evidente nas discussões “Software”, “Ritmo de Entregas” e “Aprendizado das Práticas”.

Sobre a discussão realizada na dimensão de “Aprendizado das Práticas”, ficou evidente para os pesquisadores que a equipe abre mão da capacitação constante dos seus membros tanto em tecnologias para o desenvolvimento quanto em métodos ágeis, e acabam dependendo da figura de um líder técnico para guiar as decisões sobre a mudança em seus processos. A falta de conhecimento e a dependência do líder da equipe faz com que muitas práticas sejam experimentadas sem fundamento teórico, como citado nas discussões, são escolhidas apenas no *feeling*. É importante que a equipe como um todo busque mais conhecimento sobre métodos ágeis, para que possam embasar melhor suas decisões no futuro, atingindo assim, um maior nível de maturidade nessa dimensão.

Em relação à “Software” e “Ritmo de Entregas”, as duas dimensões discutidas mostraram um nível baixo de maturidade ágil, muito impactadas pela ausência de testes automatizados. Até mesmo os testes que são realizados manualmente costumam ser poucos e não são encarados como uma prioridade para a equipe. Em decorrência disso, várias características exigidas nos níveis mais elevados não são atendidas. Entre as principais, podemos ressaltar a não existência de infraestrutura para *deploy* e *build* automatizado, a falta de garantia de integridade dos repositórios, e o grande impacto de *bugs* tomando tempo da equipe em *sprints* posteriores ao desenvolvimento das *features*.

Outro ponto que precisa ser melhor atendido pela equipe é a utilização da refatoração como ferramenta para minimizar os *bugs* e garantir maior qualidade das features já entregues.

A equipe demonstrou um bom nível de maturidade ágil na dimensão de “Descoberta das Features”, onde são realizadas práticas para garantir a qualidade dos requisitos. O uso do *planning poker* e a comunicação constante com as áreas clientes da equipe foram pontos positivos para melhorar o entendimento do time sobre as histórias e tarefas planejadas. E as

reuniões de *customer development* e planejamento de produto servem para garantir a relevância daquilo que é entregue ao cliente. Mas a utilização de práticas como *Grooming* e *Spikes*[4] contribuiriam para uma definição mais precisa dos requisitos, diminuindo a necessidade de consulta dos clientes da equipe durante o desenvolvimento. Além de garantir uma melhor estimativa do trabalho durante o *sprint*.

Em resumo, as principais iniciativas sugeridas para que o time atinja um maior nível de maturidade ágil são:

- A implantação de processos de testes como forma de garantir a qualidade do que é entregue;
- Dedicação de maior tempo ao estudo de práticas, com o foco na excelência técnica e no aprendizado de métodos ágeis, que embasariam a equipe para tomar melhores decisões sobre seu processo de trabalho;
- A utilização de refatoração como uma forma de garantia de qualidade do produto entregue;
- A implantação de uma arquitetura para *build* e *deploy* ágil, automatizando tarefas relacionadas.

3.5 Considerações Finais

Esse capítulo apresentou o planejamento e a execução da avaliação da maturidade em dois times ágeis aplicando a Bússola Ágil. Também foram discutidas as oportunidades de melhoria identificadas para os times participantes da experiência. O capítulo seguinte apresenta as considerações finais, limitações e sugestões de trabalhos futuros.

[4] Spike é uma estória de usuário com pouca ou nenhuma definição.

4. Conclusão

4.1 Considerações Finais

O objetivo inicial deste trabalho foi avaliar equipes ágeis de desenvolvimento de software sob o ponto de vista de um modelo de maturidade ágil e sugerir iniciativas para que essas equipes consigam atingir um maior nível de maturidade ágil.

O estudo foi de execução rápida, e com uma dinâmica replicável para qualquer equipe de desenvolvimento que atenda aos requisitos descritos na seção 3.2. Para atingir o objetivo proposto, foi utilizada a metodologia de Grupo Focal, pois a mesma oferece uma abordagem rápida e eficiente para coletar experiências e feedbacks de um grupo de pessoas sobre um tema de interesse mútuo.

Os dados coletados foram suficientes para que fosse gerado o desenho da Bússola, e por mais que ofereçam um diagnóstico superficial, podem propiciar que sejam observadas diversas oportunidades de melhoria nos processos do time.

4.2 Limitações

Durante a execução do trabalho, percebemos que não seria possível acompanhar a implantação das melhorias propostas para os times participantes do estudo, dessa forma, não conseguiremos ver se de fato as equipes aumentaram o seu nível de maturidade ágil com a implantação das melhorias sugeridas. As sessões foram planejadas para serem executadas em um tempo relativamente curto, já que os participantes envolvidos tinham limitações de tempo para participar das sessões de Grupo Focal, e esse fato pode ter contribuído para limitar as discussões ou descobertas de melhorias.

Do ponto de vista de ameaças de construção [Shull et al., 2008], questionamos se a Bússola Ágil de fato mede a maturidade ágil de uma equipe, ou se deveria ter sido utilizado algum outro mecanismo. Optamos pela Bússola ágil para determinar a maturidade das equipes porque é um método definido com esse propósito e já avaliado experimentalmente [Fontana, 2015]. No entanto, outros métodos poderiam chegar a respostas diferentes. Utilizar um questionário simplesmente baseado na presença ou ausência de práticas ágeis na equipe/organização poderia levar a resultados divergentes porque a equipe poderia dizer que usava alguma prática específica sem usar na realidade ou usar uma prática bem adaptada ou de forma reduzida de tal forma que a descaracterizaria.

Em relação às ameaças internas [Shull et al., 2008], questionamos se os resultados de fato derivam dos dados. A participação de um dos pesquisadores como moderador na experiência com a Equipe D, sendo o mesmo, membro desse time, poderia influenciar nos

resultados obtidos. Entretanto, o moderador seguiu o questionário proposto na Bússola e garantiu que todas as análises fossem baseadas nos áudios e transcrições coletados. A escolha dos participantes também pode impactar, por isso, foram escolhidos todos os participantes de cada equipe. Inicialmente foi realizado um piloto para que os moderadores se familiarizassem com o instrumento e também que os participantes fossem treinados nos procedimentos do Grupo Focal e obtivessem acesso ao texto da Bússola para leitura durante a experiência. Algo que pode ter impactado é que na equipe D havia um participante a distância. Por mais que não tenha sido observada dificuldades de comunicação.

Em relação às ameaças externas [Shull et al., 2008], no caso das experiências relatadas, elas são características das equipes alvo, logo não podem ser generalizadas para outros contextos.

Ameaças de confiabilidade [Shull et al., 2008] dizem respeito a se os mesmos resultados poderiam ser obtidos caso se rodasse a experiência novamente. Para tratar essa ameaça, foi definido um protocolo para fazer os grupos focais baseados nos propostos por [Fontana, 2015] para utilização da Bússola Ágil.

E, por fim, tanto o protocolo quanto as análises da execução foram avaliadas por outro pesquisador.

4.3 Trabalhos futuros

Durante a execução desse trabalho foi possível detectar algumas propostas para trabalhos futuros na área, como:

A aplicação da dinâmica em outros cenários ou domínios diferentes, como por exemplo, em empresas juniores - como a Alpha Consultoria, empresa júnior da UNIRIO - que poderiam se beneficiar da análise para amadurecer seus processos de desenvolvimento de software, podendo ser uma boa oportunidade de contribuição social.

Uma outra possibilidade da utilização da dinâmica em domínios diferentes, envolveria equipes que utilizam outros métodos ágeis.

Acompanhar as equipes após a experiência, visando de fato observar se as melhorias indicadas foram realmente implementadas e a equipe conseguiu alcançar um maior nível de maturidade ágil em seus processos.

REFERÊNCIAS

ABI-SABER, Gian Biolchini. Um Estudo Sobre a Percepção do Uso de Metodologias Ágeis por Equipes de Infraestrutura /Gian Biolchini Abi-Saber. Rio de Janeiro, 2017.

ABRANTES, J. F.; TRAVASSOS, G. H. Common agile practices in software processes. In: Empirical Software Engineering and Measurement (ESEM), 2011 International Symposium on. IEEE, 2011. p. 355-358.

ABRAHAMSSON, P.; SALO, O.; RONKAINEN, J.; and WARSTA, J. Agile software development methods: Review and Analysis. Espoo, Finlândia: Technical Research Centre of Finland, VTT Publications 478. 2002.

ADAL O.E., OZCAN-TOP, O., DEMIR, O. Evaluation of Agility Assessment Tools: A Multiple Case Study. In: Clarke P., O'Connor R., Rout T., Dorling A. (eds) Software Process Improvement and Capability Determination. SPICE 2016.
Communications in Computer and Information Science, vol 609. (2016)

BASSI FILHO, D. L. Experiências com desenvolvimento ágil. Tese de Doutorado. Universidade de São Paulo. 2008.

BECK, K.; BEEDLE, M.; VAN BENEKUM, A.; COCKBURN, A.; CUNNINGHAM, W.; FOWLER, M.; GREENING, J.; HIGHSMITH, J.; HUNT, A.; JEFFRIES, R.; KERN, J.; MARICK, B.; MARTIN, R. C.; MELLOR, S.; SCHWABBER, K.; SUTHERLAND, J. & THOMAS, D. “Manifesto for Agile Software Development”. 2001.

BOEHM, Barry. A spiral model of software development and enhancement. ACM SIGSOFT Software Engineering Notes, 11(4):14–24, 1986.

BORIA, Jorge Luis; RUBINSTEIN, Viviana Leonor; RUBINSTEIN, Andrés. A História da Tahini-Tahini: Melhoria de Processos de Software com Métodos Ágeis e Modelo MPS.1. ed.PBQP, 2013.

BLOCK, M. Evolving to Agile: A story of agile adoption at a small SaaS company. Agile Conference 2011.

CHARETTE, R. N. Foundations of Lean Development: The Lean Development Manager's Guide. Vol. 2, The Foundations Series on Risk Management (CD). Spotsylvania, Va.: ITABHI Corporation. 2002.

COAD, P.; PALMER, S. Feature-Driven Development. Prentice Hall, Englewood Cliffs, NJ. 2002.

COCKBURN, A. Agile Software Development: The Cooperative Game. 2nd Edition. Pearson Education, Boston, USA, ISBN 978-0321482754 (2007)

CMMI Product Team. CMMI for Development, Version 1.3 (CMU/SEI-2010- TR-033). Software Engineering Institute, Carnegie Mellon University (2010). http://www.sei.cmu.edu/library/abstracts/reports/10tr033.cfm

DIKERT, K.; PAASIVAARA, M.; LASSENIUS, C. Challenges and success factors for large-scale agile transformations: A systematic literature review. Journal of Systems and Software, Volume 119, pp. 87-108 (2016). DOI: 10.1016/j.jss.2016.06.013

DINGSØYR, T.; NERUR, S.; BALIJEPALLY, V.; NILS, B. M. A decade of agile methodologies: Towards explaining agile software development. Journal of Systems and Software, Volume 85, Issue. 6, Junho de 2012. p 1213-1221

FONTANA, Rafaela Mantovani. Maturity in agile software development / Rafaela Mantovani Fontana; 2015, supervisor: Sheila Reinehr ; co-supervisor: Andreia Malucelli. Pontifícia Universidade Católica do Paraná, Curitiba, 2015

HIGHSMITH, J.; COCKBURN, A. Agile software development: the business of innovation. IEEE Computer, Vol. 24, n. 9, IEEE, pp. 120-122 (2001).
DOI: 10.1109/2.947100

HIGHSMITH, J., 2002. Agile software development ecosystems, Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

ISO/IEC International Organization for Standardization/ International Electrical Committee. ISO/IEC 33001:2015. Information technology - Process assessment - Concepts and terminology (2015).

JALALI, S.; WOHLIN, C. Agile practices in global software engineering-A systematic map. In: Global Software Engineering (ICGSE), 2010 5th IEEE International Conference on. IEEE, 2010. p. 45-54.

KERR, James; HUNTER, Richard. Inside RAD: How to Build Fully Functional Computer Systems in 90 Days or Less. McGraw-Hill, 1993.

KOHL, J. Post-Agilism: Process Skepticism. 9 de Junho de 2006
http://www.kohl.ca/2006/post-agilism-process-skepticism/
Acessado em 2 de Dezembro de 2018.

KOHL, J. Post-Agilism Frequently Asked Questions. 28 de Abril de 2007.
http://www.kohl.ca/2007/post-agilism-frequently-asked-questions
Acessado em 2 de Dezembro de 2018.

KROLL, Per; MACISAAC, Bruce. Agility and Discipline Made Easy: Practices from OpenUP and RUP. Addison-Wesley Professional, 2006.

KURAPATI, N; MANYAN, V. S. C; PETERSEN, K. Agile Software Development Practice Adoption Survey. In: Agile Processes in Software Engineering and Extreme Programming. Springer Berlin Heidelberg, 2012. p. 16-30.

LAANTI, M. Agile methods in large-scale software development organizations: Applicability and model for adoption. PhD Dissertation, University of Oulu, Department of Information Processing Science, Oulu, Finlândia (2012)

LEAL, Tainá. Pós-agilismo – Um estudo sobre o legado das metodologias ágeis para os processos de software. Rio de Janeiro, 2014.

LEPPÄNEN, M. A Comparative Analysis of Agile Maturity Models. In: R. Pooley et al. (eds.), Information Systems Development: Reflections, Challenges and New Directions. Springer Science+Business Media. New York (2013).

MARCHENKO, A; ABRAHAMSSON, P., Scrum in a Multiproject Environment: An Ethnographically-Inspired Case Study on the Adoption Challenges. Agile, 2008. AGILE '08. Conference , vol., no., pp.15,26, 4-8 Aug. 2008

MARTIN, James. Rapid Application Development. Macmillan Publishing Co., 1991.

MCLAUGHLIN, M. What Is Agile Methodology.
Disponível em <https://www.versionone.com/agile-101/agile-methodologies/>.
Acesso em: 2 de dezembro de 2018.

MELO, C., SANTOS, V., CORBUCCI, H., KATAYAMA, E., GOLDMAN, A., KON, F. Métodos ágeis no Brasil: estado da prática em times e organizações. Relatório Técnico RT-MAC-2012-03. Departamento de Ciência da Computação. IME-USP. Maio, 2012.

MEYER, B. Agile! The Good, the Hype and the Ugly. ETH Zurich, Zurich, Switzerland. Springer International Publishing Switzerland 2014

MYER, T. Professional CodeIgniter. John Wiley & Sons. 2008.

PAULK, M., Extreme Programming from a CMM Perspective. IEEE Software, Vol. 18, No. 6, pp. 19-26 (2001). PRESSMAN, Roger S. Software Engineering: a Practitioner's Approach. McGraw-Hill, 6th edition, 2004.

PRIKLADNICKI, R. et al. Métodos Ágeis Para Desenvolvimento de Software. Porto Alegre: Bookman, 2014.

ROMAIN, G. Characterizing the presence of agility in large-scale agile software development. Masters Thesis presented in the Faculty of Computer Science of the Pontifical Catholic University of Rio Grande do Sul (PUCRS) (2015).

ROSE, D. Symbolic Innovation in Agile Transformations. Agile Conference, Washington, USA (2015). DOI: 10.1109/Agile.2015.17

SCHWEIGERT, T.; NEVALAINEN, R.; VOHWINKEL, D.; KORSAA, M.; BIRO, M., Agile Maturity Model: Oxymoron or the Next Level of Understanding. A. Mas. Et al. (Eds). : SPICE 2012, May 29-31, pp. 289-294 (2012).

SCHWABER, K.; BEEDLE, A. 2002. Agile Software Development with SCRUM. Prentice-Hall, Upper Saddle River, NJ.

SHULL, F.; SINGER, J.; SJØBERG, D. Guide to Advanced Empirical Software Engineering. Springer-Verlag London Limited 2008

SILVA, C. E. A. C. Um Estudo de Caso Sobre Adoção de Práticas Ágeis Em Um Ambiente Tradicional. Monografia de Conclusão de Curso (Bacharelado em Sistemas de Informação) – Universidade Federal do Estado do Rio de Janeiro, Rio de Janeiro. 2013

SKORKIN, A. Are you actually a Post-Agilist? 14 de Outubro de 2008.
Disponível em <http://www.skorks.com/2008/10/are-you-actually-a-post-agilist/>
Acessado em 2 de dezembro de 2018

SOARES, M. S. Comparação entre metodologias Ágeis e tradicionais para o desenvolvimento de software. INFOCOMP Journal of Computer Science 3.2. 2004. p. 8-13.

SOFTEX, 2016, ASSOCIAÇÃO PARA PROMOÇÃO DA EXCELÊNCIA DO SOFTWARE BRASILEIRO – SOFTEX., MPS.BR – Guia de Implementação:
Implementação e Avaliação do MR-MPS-SW:2016 em Conjunto com o CMMI-DEV Disponível em: http://www.softex.br.

STAPLETON, Consortium; STAPLETON, Jennifer. DSDM: A Framework for Business-Centered Development. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003.

VENABLE, J.; PRIES-HEJE, J. P.; BASKERVILLE, R. A Comprehensive Framework for Evaluation in Design Science Research. In: Peers, K.; Rothenberger, M.; Kuechler,B. (Eds.) DESRIST 2012, LNCS 7286, pp. 423-438. (2012)

VIEIRA, D., Visão Geral da Metodologia Ágil Scrum. 16 de Junho de 2014.
http://www.mindmaster.com.br/scrum/
Acessado em 9 de Dezembro de 2018

WIKIMEDIA Foundation. Wikipedia. Kanban Boards Web. 21 Dec. 2015.
Acessado em 9 de Dezembro de 2018

WILLIAMS, L. Agile software development methodologies and practices. Advances in Computers, v. 80, p. 1-44, 2010.

ANEXO I - Bússola Ágil

A Bússola Ágil [Fontana, 2015] é uma ferramenta que permite avaliar a maturidade ágil em equipes de desenvolvimento de software. As dimensões avaliadas e seus resultados esperados estão descritas na íntegra, abaixo:

APRENDIZADO DAS PRÁTICAS

• Tentando ser ágil

A equipe está empiricamente tentando utilizar um método ágil. Na maioria das vezes, nem todas as práticas estão implementadas. Os benefícios ainda não são percebidos completamente.

•Escolhemos aprender fazendo
•Ainda não percebemos benefícios em se tornar ágil
•Ainda estamos aprendendo a entregar valor para o cliente

• Aprendendo a ser ágil

A equipe está aprendendo a ser ágil. Os métodos são usados passo-a-passo, como descrito nos livros e tutoriais. Normalmente a equipe está em treinamento ou coaching. Todas as práticas são utilizadas e é importante que a equipe aprenda o valor de cada uma delas para poder adaptá-las ou até abandoná-las mais tarde.

▪ Estamos seguindo um método ágil conforme descrito em tutoriais e livros
▪ Estamos passando por treinamento ou coaching
▪ Estamos trabalhando em projetos piloto

• Atuando nos processos de trabalho

A equipe entende as práticas ágeis e o valor de cada uma. Assim, a equipe decide como adaptar tais práticas ao seu contexto. Os membros da equipe se sentem confiantes com processos de trabalho flexíveis, sem abandonar a agilidade.

▪ Aprendemos um método ágil e entendemos o valor de cada prática

▪ Estamos atualmente adaptando as práticas

▪ Sentimos que os processos de trabalho são mais flexíveis e de acordo com as características de cada projeto

• Compreendendo a situação

A equipe tem informação sobre seus processos de trabalho. A informação é simples e normalmente baseada em dados visíveis a todos ou métricas simples. A equipe usa essa informação para tomar decisões e implementar melhorias nos processos. Essas melhorias são, geralmente, pequenas e incrementais.

▪ Somos capazes de explicar o que acontece nos nossos processos de trabalho

▪ Temos algumas métricas

▪ Nós usamos nosso entendimento sobre o processo para realizar melhorias

Conduta da Equipe

• Equipe responsiva

A equipe precisa de uma liderança próxima para que as atividades sejam realizadas. Os membros da equipe não têm (ou não lhes foi dada) autonomia para tomar decisões. A equipe não tem confiança para proteger suas prioridades de trabalho e isso gera, normalmente, trabalhos extras por demandas não planejadas.

▪ Temos autonomia parcial para tomar decisões (sobre o design do software, sobre tecnologia, sobre o projeto etc)

▪ Sentimos necessidade de um Scrum Master conosco a maior parte do tempo

▪ Temos políticas fracas para proteger o trabalho da equipe. Isso significa que normalmente nossas prioridades de trabalho mudam.

• Equipe confiante

Esta equipe precisa do Scrum Master por perto, mas se sente confiante para tomar pequenas decisões sobre o projeto. Sua comunicação é fluente. Mudanças nos processos de trabalho sugeridas pela equipe precisam de aprovação da gestão.

▪ Somos fluentes nas práticas e sentimos autonomia para tomar decisões no projeto
▪ Nós nos conhecemos e nos sentimos à vontade para se comunicar
▪ Sugerimos melhorias no processo, mas elas normalmente precisam ser aprovadas

• Equipe assertiva

Essa equipe se sente responsável pelo projeto. Seus membros possuem autonomia para mudar os processos de trabalho. Eles tomam, juntos, decisões e informam seus gestores.

▪ Temos autonomia para melhorar nossos próprios processos de trabalho
▪ Temos capacidade de direcionar nosso trabalho durante o projeto
▪ Temos políticas para proteger nosso trabalho, de forma que dificilmente interrupções prejudicam nossa produtividade

• Equipe brilhante

Esta equipe está focada em excelência técnica. Eles criam seus próprios processos de trabalho, com o suporte da gestão. Eles precisam de um ambiente que dê suporte ao aprendizado contínuo para, com isso, manterem-se motivados. É possível sentir, nos membros dessa equipe, o foco em aprender sempre, o tempo todo.

▪ Definimos nossos processos de trabalho e os mudamos quando sentimos necessidade
▪ Sentimo-nos motivados a continuar sempre aprendendo
▪ Sentimos que nossa gestão nos apoia na implementação de nossas ideias

Ritmo de Entregas

• Terminando código

As Sprints são planejadas de forma que, ao final, algum código seja finalizado e integrado ao repositório. Na maioria das vezes, a funcionalidade não foi testada ainda. A equipe está aprendendo como entregar valor no fim da Sprint.

▪ No fim da Sprint, temos código finalizado, mas não entregamos ainda
▪ Nosso tamanho de Sprint varia
▪ Não temos certeza do que adiciona valor ao nosso cliente

• Gerando entregáveis

Esta equipe identifica o valor a ser entregue para o cliente, mas não consegue entregar no fim da Sprint. O ambiente tecnológico não dá o suporte necessário para integração de código e build ágil. O teste geralmente é manual.

▪ Somos capazes de ter funcionalidades prontas para a entrega ao fim da Sprint, mas não entregamos
▪ Conseguimos identificar um "*produto mínimo entregável*" (minimum releasable product)

• Entregando quase no prazo

As entregas são planejadas e realizadas, mas normalmente atrasadas. Às vezes, demandas extras aparecem durante a Sprint e a equipe precisa atender. O ambiente tecnológico dá suporte razoável para integração e build automatizado.

▪ Planejamos nossas entregas e as realizamos, geralmente com atraso
▪ Os tamanhos das nossas estórias de usuário variam

• Entregando no prazo

As entregas são planejadas e realizadas no prazo. Às vezes, até antes do prazo. O tamanho das estórias de usuário é, geralmente, pequeno. O ambiente tecnológico dá suporte completo para integração e build automatizado.

▪ Planejamos nossas entregas e as realizamos, às vezes antes do prazo
▪ Os tamanhos das nossas estórias de usuário são similares, geralmente pequenos

Descoberta das Features

• Coletando requisitos

Os requisitos são definidos no início do projeto, normalmente com texto ou diagramas. A equipe não se sente confortável com mudanças ou requisitos emergentes durante o projeto.

▪ Definimos a maioria dos requisitos no início do projeto
▪ Não sabemos exatamente como lidar com mudanças nos requisitos durante o projeto

• Descobrindo requisitos

Os requisitos são vagamente definidos no início do projeto e detalhados assim que as Sprints são iniciadas. Mudanças e requisitos emergentes são bem-vindos. Spykes (ou provas de conceitos) podem ser usadas para elicitação de requisitos.

▪ Definimos, em detalhes, os requisitos para a próxima Sprint
▪ Sentimo-nos confortáveis com mudanças ou requisitos emergentes
▪ Conseguimos identificar um produto mínimo entregável (*minimum releasable product*) nas nossas estórias

• Refinando requisitos

Existem iniciativas sistemáticas para garantir a qualidade dos requisitos, para que eles estejam de acordo com as necessidades dos clientes.

▪ Estamos preocupados se o que entregamos ao cliente é exatamente o que ele precisa;
▪ Temos iniciativas para melhorar a qualidade dos requisitos

Software

• Conhecimento das falhas

Existem falhas nas entregas que precisam ser tratadas o mais rapidamente possível. Tais falhas acontecem devido a problemas no processo de desenvolvimento.

▪ Gastamos tempo nas Sprints corrigindo bugs de Sprints anteriores

▪ Gastamos tempo nas Sprints resolvendo problemas de build e integração de código.

• Código fonte de alto-nível

A codificação é considerada uma atividade importante e existem iniciativas para garantir a clareza e a robustez do código, utilizando as melhores práticas disponíveis.

▪ Preocupamo-nos com nosso código

▪ Realizamos iniciativas para garantir que o código está bom, como revisões, refatorações, programação aos pares ou outras

• Software entregue de alto-nível

Existe uma preocupação com a qualidade da feature que está sendo entregue. Por esta razão, teste é prioridade – automatizado ou não. Também há uma preocupação com a integridade do código no repositório.

▪ Vemos teste como uma prioridade

▪ O código é integrado o mais rapidamente possível

▪ O código do nosso repositório está sempre íntegro

• Codificação eficiente

Codificação, integração e testes são realizados em uma infraestrutura tecnológica que permite agilidade. A equipe está preocupada em remover esperas/atrasos desnecessários nos processos de trabalho. Devops é uma infraestrutura comum para permitir codificação eficiente.

▪ Temos testes automatizados, sejam unitários ou funcionais
▪ Nosso build e integração são automatizados
▪ Estamos focados em remover esperas/atrasos nos processos de trabalho automatizando tarefas manuais

Relacionamento com o Cliente

• Equipe conhecendo o cliente

A equipe está aprendendo como funciona o negócio do cliente e qual é a dinâmica das demandas. Esta é a razão pela qual a contribuição para o negócio do cliente ainda é incipiente.

▪ Estamos conhecendo o cliente e suas demandas
▪ Conseguimos ajudar pouco nosso cliente a definir seus requisitos

• Cliente conhecendo a equipe

É o processo de aprendizado do cliente sobre a equipe. O cliente está conhecendo os processos de trabalho da equipe, mas não está completamente acostumado. Às vezes parece que há falta de confiança na equipe.

▪ Nosso cliente está conhecendo nossa forma de trabalho
▪ O cliente sabe o que vamos entregar
▪ O cliente não se sente confortável em repriorizar requisitos

• Cliente confiante

O cliente está confiante com a forma de trabalho da equipe. Ele acredita que a equipe está preocupada em entregar o que ele precisa. O cliente tem uma postura mais flexível com a entregas e se sente mais confortável com a repriorização de requisitos. Existe transparência entre a equipe e o cliente.

▪ Ajudamos nosso cliente na definição de requisitos

▪ O cliente conhece seu papel no processo de trabalho
▪ Sentimos que o cliente confia no nosso trabalho

• Cliente parceiro

O cliente reconhece a parceria com a equipe: enquanto ele sente que a equipe está comprometida com o seu negócio, a equipe sente que o cliente é parte da equipe.

▪ Identificamos e sugerimos melhorias no negócio do nosso cliente
▪ Nosso cliente se sente parte da equipe, com responsabilidade compartilhada

Suporte da Organização

• Movimentação ágil

Algumas equipes isoladas estão iniciando projetos utilizando métodos ágeis. A organização sabe disso, mas não demonstra interesse nos novos processos ou nos seus resultados.

▪ Vemos iniciativas isoladas de agilidade em nossa empresa
▪ Sentimos pouco reconhecimento de iniciativas ágeis por parte da alta gestão

• Comprometimento ágil

A alta gestão apoia a implementação de métodos ágeis na empresa e as iniciativas são oficiais. Existem investimentos em treinamentos, coaching, comunicação e infraestrutura tecnológica para a transformação ágil.

▪ Existem projetos piloto oficiais acontecendo na empresa
▪ Sentimos suporte organizacional (treinamento, coaching, infraestrutura) para a transformação ágil

• Prioridade ágil

Existe suporte completo da alta gestão para a transformação ágil. Departamentos,

papéis e equipes mudam para dar suporte à agilidade. Quando é uma iniciativa top-down, resistência ainda surge em algumas equipes.

▪ Vemos que a estrutura organizacional (física e departamental) mudou para dar suporte à transformação ágil

▪ Sentimos resistência à mudança em algumas equipes

• Negócio ágil

Empresas de desenvolvimento de software criadas com base em métodos ágeis normalmente são “negócios ágeis”. As estratégias de gestão da organização são focadas em pessoas e processos enxutos. Isso não somente no desenvolvimento de software, mas para toda a empresa.

▪ Nossa empresa é reconhecida por sua agilidade

▪ As equipes da nossa organização adotam valores ágeis, mas adaptam seus processos de acordo com as características dos projetos

APÊNDICE I - Execução e análise da sessão piloto

As organizações A e B são negócios ágeis e suas equipes diagnosticadas possuem um perfil bem parecido de foco em excelência técnica e qualidade através de testes. Essa preocupação explica o alto nível de maturidade identificado nas dimensões de software e ritmo de entregas. Seus processos são maduros e as empresas contam com especialistas em agilismo para definir padrões para o desenvolvimento e buscar maior eficiência na adoção de práticas ágeis por suas equipes. As equipes possuem uma certa autonomia para realizar alterações em seus processos, mas precisam alinhar essas mudanças com a gestão.

O principal ponto negativo identificado nas equipes é o relacionamento com o cliente. Os participantes não consideram que exista uma relação de confiança entre os clientes e a equipe de desenvolvimento, nem consideram seus clientes como parceiros. Para chegar em um nível mais alto de maturidade nessa dimensão, as equipes precisariam conhecer melhor o negócio do cliente, e buscar mais participação na decisão de *features* a serem desenvolvidas.

Outro ponto negativo relacionado é a falta de visibilidade do impacto no negócio dos artefatos desenvolvidos por eles

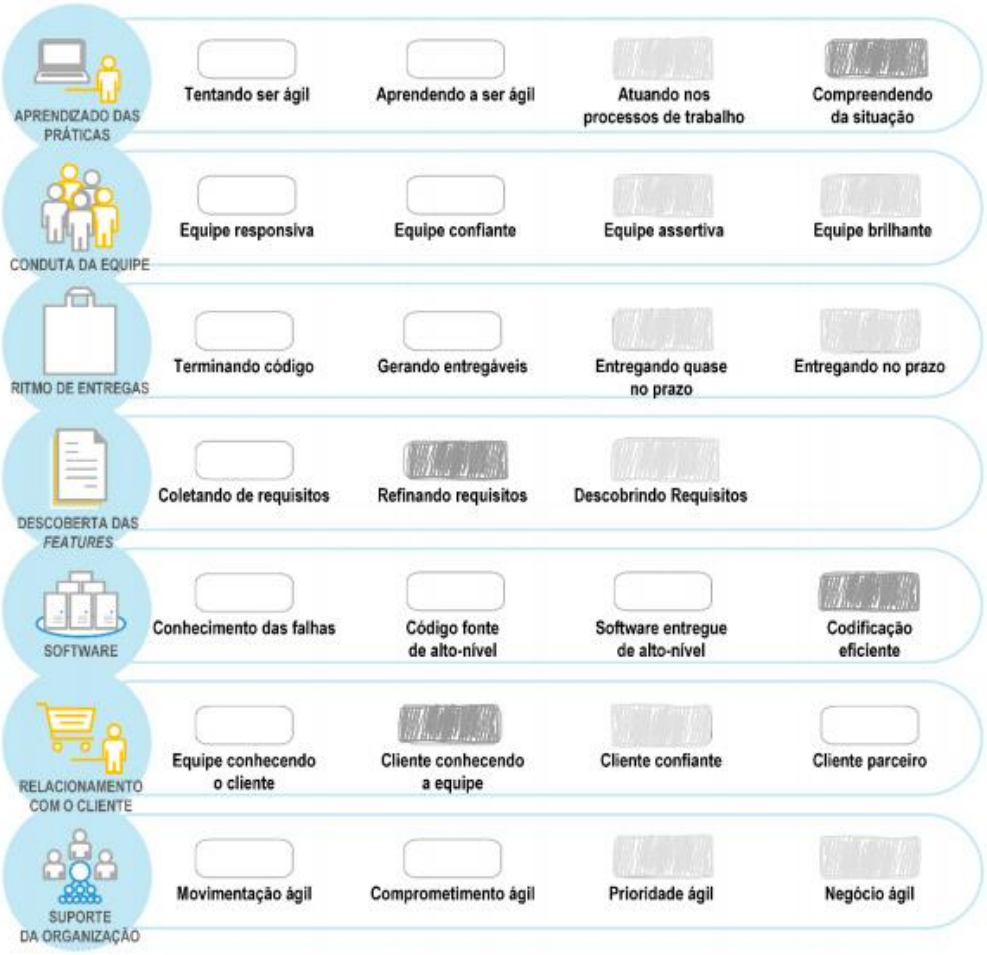

Figura 9: Desenho da Bússola preenchida no piloto referente ao time do participante A

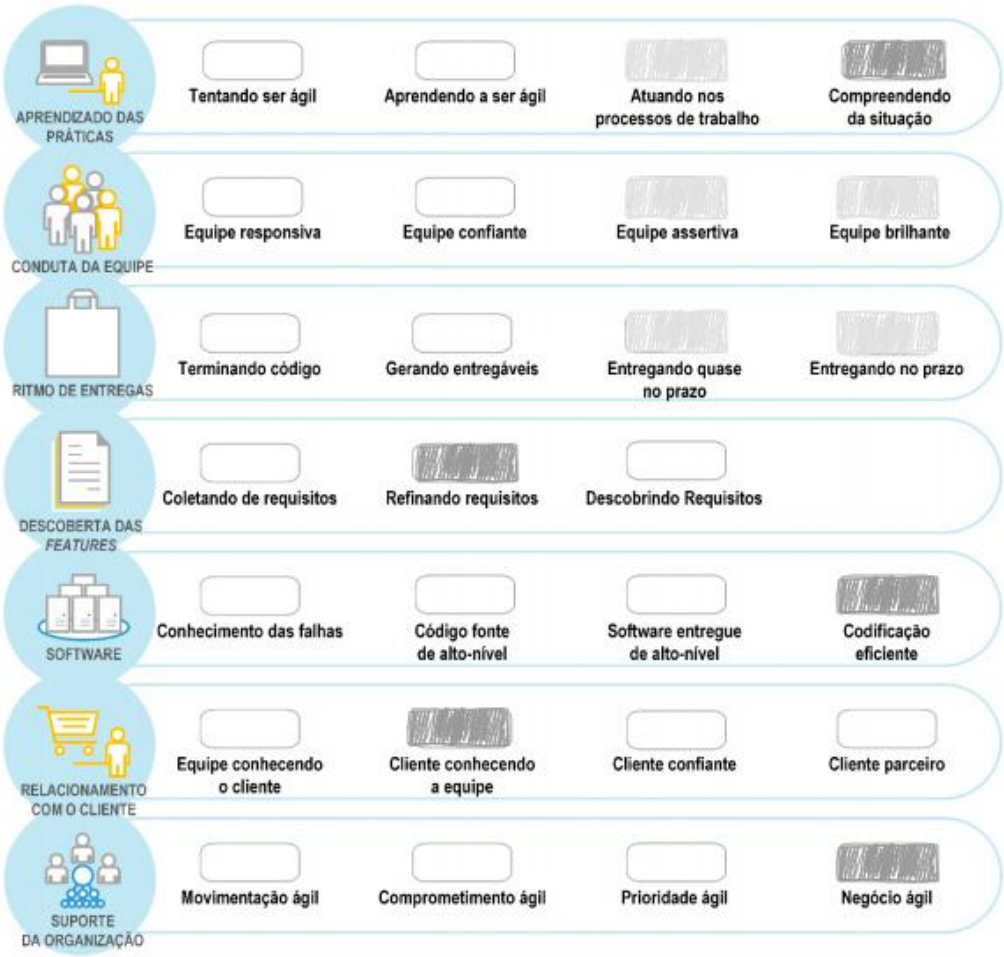

Figura 10: Desenho da Bússola preenchida no piloto referente ao time do participante B