AGILE UX: Projetando a *User Experience* no Mundo Ágil.
AGILE UX: Designing The User Experience In An Agile World

RIKER, Diogo Seffair.[1]
TEXEIRA, Narle Silva.[2]

RESUMO

O presente artigo tem como tema central o *Agile UX*, demonstrando como é aplicado dentro de um time de desenvolvimento de *scrum* da FPF Tech, um instituto de pesquisa e desenvolvimento localizado em Manaus/AM. O método procura valorizar mais a experiência do usuário dentro da cultura ágil, utilizando os fundamentos, princípios e técnicas do *Lean UX* e *Design Thinking* a fim de agregar mais valor nos incrementos entregues para o cliente e, consequentemente, para os usuários. A investigação teve como base metodológica a pesquisa bibliográfica sobre os temas envolvidos e sua aplicação. Os resultados obtidos foram bastante promissores para todos os envolvidos (desde a equipe, até o cliente) com algumas ressalvas, tais como: a ausência de um padrão para sua aplicação, dando liberdade para ser moldado de acordo com o ritmo do time e a necessidade do perfil multidisciplinar de seus integrantes, aumentando as chances de sucesso do projeto. Observou-se também a possibilidade de continuar esse estudo com temas relacionados, como perfil ideal de um time scrum para utilizar-se do *Agile UX*, por exemplo.

Palavras-chave: *Agile UX. Scrum. Lean UX*.

ABSTRACT

The present article has, as its central theme, the Agile UX, demonstrating how it is applied in a scrum development team at FPF Tech, an Institute of research and development, established in Manaus/AM. The method tries to valorize more the user experience, within the agile culture, using the principles, techniques, and fundamentals of the Lean UX and Design Thinking, in order to aggregate more value in the increments delivered to the client, and consequently, to the users. The investigation used, as methodological base, the literature research on the themes involved, and its application. The results obtained were very promising, to all involved (for the staff and the client), with a few reservations, such as: the lack of a standard for its application, giving freedom to be molded in accordance with the team's pace, and the multidisciplinary profile of its members, increasing the chances of success of the project. It was observed, also, the possibility of continuing this study with related themes, as ideal profile of a scrum team to use the Agile UX, for example.

Keywords: *Agile UX. Scrum. Lean UX*.

[1] Aluno do curso de Pós-graduação em Design de Interação da Faculdade Fucapi (Instituto de Ensino Superior Fucapi). E-mail: diogo.riker@gmail.com
[2] Designer, Mestre em Educação, professora da Faculdade Fucapi (Instituto de Ensino Superior Fucapi). E-mail: narle.texeira@fucapi.br

1. INTRODUÇÃO

A tecnologia evolui continuamente, facilitando o cotidiano de todos em diversos aspectos, tendo as pessoas e suas necessidades como variáveis importantes para que tal evolução seja possível. Como consequência, a área de desenvolvimento de *software* tem acompanhado essa evolução a fim de suprir os problemas das pessoas com soluções que façam a diferença em seu dia-a-dia. Entretanto, para que isso seja possível é preciso um trabalho interdisciplinar para analisar os usuários sob diversas esferas.

Ao analisar o processo de desenvolvimento de *software* na década de 80 e 90, procurava-se fazer analogias de processos utilizados em outros contextos. Sabbagh (2013, p. 19) afirma que "desde quase o começo da história do desenvolvimento de *software,* a comparação com a construção civil foi largamente utilizada para descrever esse tipo de projeto", dando origem a cargos como engenheiro de software, por exemplo, e também aos métodos tradicionais. Entretanto as áreas possuem naturezas bem distintas.

A internet foi um grande diferencial que ajudou a impulsionar a tecnologia, permitindo o seu avanço com mais rapidez, quebrando paradigmas da época e criando novos, como a velocidade do trabalho e a distribuição dos *softwares*. Gothelf (2013, p. 4, tradução nossa) afirma que "a internet tem mudado a distribuição de *software* radicalmente. Maioria dos *softwares* são distribuídos online. Nós não precisamos mais nos limitarmos ao processo de fabricação físico e temos liberdade para trabalhar em ciclos mais curtos", ficando claro o surgimento de uma cultura ágil (como o manifesto ágil e, consequentemente, o *scrum*) por trás dessa área.

Sabbagh (2013, p.18) afirma que o *scrum "*é um *framework* Ágil, simples e leve, utilizado para gestão do desenvolvimento de produtos complexos imersos em ambientes complexos", ou seja, facilitar a implementação de algo num período curto de modo incremental, agregando mais valor de negócio para o cliente e buscando sempre a evolução continua do time de desenvolvimento.

Atualmente ao analisar a área de desenvolvimento de *software,* é muito comum encontrar empresas em transição do modelo tradicional (*Waterfall*) para o mundo ágil, sendo esse um grande choque cultural. Outro aspecto que tem acontecido também, é a procura de uma solução para pensar mais na experiência do usuário nesse contexto ágil, dando origem a alguns métodos como o *Agile UX*. Na Fundação Paulo Feitosa (FPF Tech) é possível encontrar dois ambientes, ou seja, *designers* separados em seu respectivo setor (o que dificulta a comunicação com o resto do time, seguindo os

preceitos do modelo tradicional) e *designers* junto com a equipe de desenvolvimento (preservando os princípios do manifesto ágil). Portanto, o presente artigo teve como questão norteadora saber como é possível aplicar o *Agile UX* em times de desenvolvimento dentro do *scrum* e como isso ocorre na FPF Tech?

Assim, o objetivo geral da investigação consiste em demonstrar como é aplicado o *Agile UX* dentro de um time de desenvolvimento de *Scrum* da FPF Tech. Os objetivos específicos foram elaborados como: apresentar os valores e princípios do manifesto ágil; descrever o conceito do *framework "Scrum"* e discutir o conceito de *Agile UX* (considerando os princípios do *Design Thinking* e do *Lean UX*).

O método empregado para a realização do artigo foi a pesquisa bibliográfica, tendo como fonte livros, artigos e sites especializados a fim de aprimorar o conhecimento nos seguintes temas: *Manifesto Ágil, Scrum, Lean UX, Kanban, Design Thinking* e *Agile UX*. Após essa fase, ocorreu um procedimento de observação e descrição do procedimento que o time de *Scrum* da FPF Tech utiliza em seu cotidiano.

O referencial teórico utilizado transitou pelos seguintes temas: *Agile UX* (Desireé Sy, Jeff Gothelf, Lydia Waldmann), *Design Thinking* (Maurício Vianna, Ysmar Vianna, Isabel K. Adler, Brenda Lucena, Beatriz Russo), *Lean UX* (Jeff Gothelf e Jesper Boeg), Manifesto Ágil (André Faria Gomes, Rafael Sabbagh e Wiston Royce), *Scrum* (Rafael Sabbagh, César Brod, Jeff Sutherland e Ken Schwaber).

O artigo está estruturado em quatro tópicos. No primeiro, procedeu-se em apresentar os princípios e valores do manifesto ágil, mostrando o conceito do modelo tradicional e suas principais diferenças em relação à cultura ágil. No segundo, foi feito uma explanação sobre o *scrum*, demonstrando seus artefatos, cerimônias, valores, pilares e o time de *scrum.* Já no terceiro tópico, abordou-se sobre *Agile UX*, demonstrando o conceito, valores e princípios do *Lean UX*, um entendimento sobre *Design Thinking* e seu método e a interação entre *Lean UX e Design Thinking* dentro do *Agile UX* . Também foi mostrado formas de como o modelo já foi aplicado. Por fim, no ultimo tópico, foi descrito como o *Agile UX* é aplicado na FPF Tech.

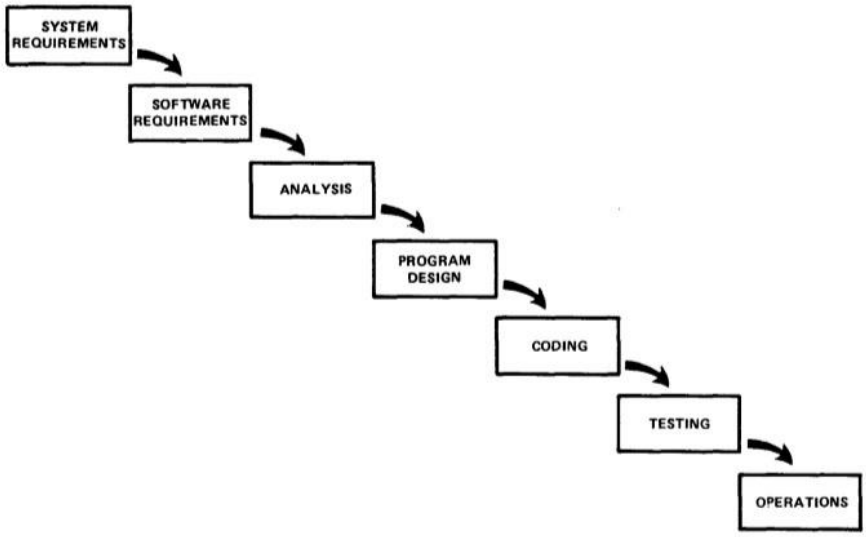
2. FUNDAMENTAÇÃO TEÓRICA

2.1 Manifesto Ágil

A área de desenvolvimento de *software* é uma área que cresce ano após ano, tendo passado por constantes evoluções. Inicialmente, tem-se o método de trabalho em

cascata (ou *Waterfall)*, criado na década de 70 por Royce (figura 01) e caracteriza-se pelo trabalho linear, percorrendo uma sequência de fases, onde a próxima só inicia-se após o termino da anterior. Nesse modelo é exigido uma série de procedimentos onde o usuário só participa no inicio do processo, quando ocorre o levantamento de requisitos das funcionalidades que devem constar no *software*.

Figura 1: Método tradicional Waterfall



Fonte: ROYCE, W. (1970, p. 2)

Entretanto, de acordo com Royce (1970, p. 329, tradução própria), esse método "apresenta riscos e possíveis falhas". Sabbagh (2013, p. 19) deixa isso bem claro, ao demonstrar o pensamento utilizado na época que segundo seu ponto de vista "acreditava-se que seria possível tratar o desenvolvimento de software como um processo previsível". Porém, quando alguma fase falha, aumenta cada vez mais a complexidade do seu uso.

Em contra partida a essa visão, um grupo de profissionais da área, no ano de 2001, reuniram-se para trocar ideias e experiências de como eles estavam conseguindo agregar valor na entrega de *softwares* para seus clientes e, consequentemente, rompendo com o modelo tradicional. Ao final desse encontro, apesar dos diferentes métodos apresentados, foram identificados quatro valores comuns entre eles, passando a ser conhecido como base do manifesto ágil (*Agile Manifesto*, 2001). São eles:

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

Ou seja, mesmo havendo valor nos itens da direita, é valorizado os itens da esquerda.

Além dos quatros valores, foram identificados doze princípios que os complementam. São eles:

- Nossa maior prioridade é satisfazer o cliente através da entrega contínua e adiantada de *software* com valor agregado.
- Mudanças nos requisitos são bem-vindas, mesmo tardiamente no desenvolvimento. Processos ágeis tiram vantagem das mudanças visando 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 projeto 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.

Portanto, percebe-se que esse encontro revolucionou o mundo do desenvolvimento de *software*, deixando claro que pessoas são muito mais importante que qualquer processo e quanto menos engessado for o processo, mas ágil ele será. De acordo com Gomes (2013, p. 4) “O manifesto ágil é o embasamento filosófico de todos os métodos ágeis e diversos métodos de desenvolvimento de *software* que estão alinhados a ele”, deixando claro a possibilidade de modelar qualquer *framework* ágil de acordo com a necessidade do time, buscando sempre sua melhoria continua.

2.2 Scrum

O scrum, de acordo com Schwaber e Sutherland (2013, p. 3), "é um *framework* dentro do qual pessoas podem tratar e resolver problemas complexos e adaptativos, enquanto produtiva e criativamente entregam produtos com o mais alto valor possível", ou seja, um *framework* que busca, continuamente, entregar softwares com valores (retorno de investimento) para seus clientes em períodos curtos, onde a repriorização é constantemente ativa.

De acordo com a relatório anual de 2013 da *Scrum Alliance*, onde foram entrevistados mais de 500 participantes de 70 países do mundo, o número de adoção do *scrum* nas empresas em geral é de 61%, como podemos ver na figura 02.

Figura 2: *How would you describe the state of scrum in your organization?*

Fonte: Scrum Alliance (2013).

Pode-se atribuir esse número justamente pelo fato de que o *scrum* é um *framework* e não uma metodologia. De acordo com Sabbagh (2013, p. 29) "Um *framework* ou arcabouço é uma estrutura básica que pretende servir de suporte e guia para a construção, a partir da expansão dessa própria estrutura, de algo com uso prático", ou seja, a flexibilidade que tem-se com o *scrum* é muito alta, permitindo moldá-la de acordo com a necessidade do time ou da organização (seja utilizando apenas o *scrum* ou mais de um método ágil).

Por enfatizar o trabalho em equipe, a *Scrum Alliance* defende a existência dos valores do *scrum*, são eles: **foco** (aumentando a produtividade do time, evitando a multitarefa e trabalhar em mais de um projeto simultaneamente), **coragem** (respeitando os valores ágeis, ter coragem para aceitar a mudança como parte natural do processo), **franqueza** (fundamental para o bom andamento do *framework*, pois permite a aplicação dos pilares do *scrum*), **compromissos** (o time tem a liberdade de determinar como vai trabalhar, responsabilizando-se pelos resultados) e **respeito** (o time deve respeitar as opiniões e ponto de vistas dos outros membros do time).

Além de seus valores, Schwaber e Sutherland (2013) enfatizam que o *scrum* é fundamentado nas teorias empíricas de controle de processo, havendo a necessidade de três pilares para sustentá-lo: **transparência** (visibilidade do processo aos envolvidos pelo os resultados), **inspeção** (constantes inspeções do processo a fim de melhora-lo) e **adaptação** (caso o processo interfira de modo negativo no resultado, o time precisa adaptar o processo de acordo com suas necessidades).

No *scrum*, existem três elementos que são fundamentais para todo processo. São eles:

- ***Scrum Master (SM):*** Age como o facilitador para o time de desenvolvimento e reuniões do *scrum*. Responsável por remover impedimentos que gerem riscos de não atingir o objetivo da *sprint*, facilitar a comunicação entre o time e o *product owner* e disseminar a cultura do *scrum* para a organização que trabalha.
- ***Product Owner (PO)*****:** é considerado o dono do produto. Está em constante comunicação com o cliente e seus stakeholders, incluindo os usuários. É responsável por definir, manter e comunicar a visão do produto ao longo do projeto, é o único que tem o poder em cancelar uma *sprint*, caso mude a meta do que foi planejado para aquele ciclo de desenvolvimento e é apenas ele que decide se o que foi realizado vai ser entregue para o cliente ou não.
- **Time de desenvolvimento:** são os responsáveis em realizar o desenvolvimento do produto. Segundo Sabbagh (2013, p. 40) “Ele é multidisciplinar, o que significa que possui, em seus membros, todo o conhecimento necessário para realizar esse trabalho”, ficando claro a necessidade de um time com *know-how* em diferentes áreas, desde desenvolvimento até *design*. Outro fator importante é que esse time é auto gerenciável, acordando as metas a serem entregues naquela *sprint* com o PO e definindo como irão se organizar para atingirem o objetivo da meta, respeitando sempre o *timebox* do ciclo.

O *scrum* é divido em ciclos de desenvolvimentos, sendo chamados de *Sprints* e seu período (ou *timebox)* varia entre 1 a 4 semanas. De acordo com Schwaber e Sutherland (2013, p. 8) "O coração do *scrum* é a *Sprint,* um *time-boxed* de um mês ou menos, durante o qual um "Pronto", versão incremental potencialmente utilizável do produto, é criado", ou seja, é por meio desses ciclos que o *framework* se faz incremental e iterativo, buscando sempre entregar uma versão ou funcionalidade do *software* com valor para o cliente já poder utilizá-lo.

No decorrer da *sprint* (figura 03), é preciso acontecer algumas cerimonias.

Figura 3: Esquematização de uma *sprint*

Fonte: https://www.cprime.com/wp-content/static/images/resources/Scrum_cycle.JPG

- A reunião de planejamento (*Sprint Planning),* onde são definidos o objetivo e a meta do ciclo de desenvolvimento, formando assim o *Sprint Backlog*. Envolvem-se o *scrum master*, time de desenvolvimento e o *product owner* (PO).
- Trabalho de desenvolvimento do produto, onde é feito toda a parte funcional do que foi planejado. Deve ocorrer num período entre 1 a 4 semanas. Envolvem-se o time de desenvolvimento e o *scrum master*.
- Reunião diária *(Daily Meeting)*, ou seja, momento em que o time e o *scrum master* se reúnem para responder três perguntas (O que eu fiz ontem?, O que estou fazendo agora? e Quais impedimentos tive?), dando visibilidade do progresso das atividades para todos os envolvidos. Ressalta-se que esta deve durar, no máximo, 15 minutos.
- Reunião de *Sprint Review*, ocorre no final de cada ciclo, onde é apresentado parte do produto para o PO e o cliente, cujos fornecem seus *feedbacks* com

relação a entrega. Envolvem-se o time de desenvolvimento, *scrum master*, *product owner* e cliente.

- Reunião de *Sprint Retrospective*, momento em que todos avaliam o que funcionou bem e o que precisa ser melhorado, buscando sempre a melhoria contínua do time. Envolvem-se o time de desenvolvimento, *scrum master e product owner.*

O *scrum* começa a entrar em prática quando inicia-se um projeto, ou seja o PO procura traduzir o problema, a necessidade do cliente e o objetivo a ser almejado na visão do produto. Após essa etapa, a organização define quem será o time de desenvolvimento e o *scrum master*, e em seguida o time *Scrum* e o PO reúnem-se para deixar claro a todos a visão do produto para pensarem e discutirem as possíveis soluções que atendem as necessidades do cliente e, consequentemente, resolver seu problema. Esse momento é conhecido como *sprint zero* e, de acordo com Sabbagh (2013, p. 40) "essa fase possui uma duração que depende do que e de quanto é necessário se definir e preparar".

Apesar de não fazer parte do *framework*, esse momento é muito importante para todos os envolvidos. Sabbagh (2013, p. 41) deixa isso bem claro quando faz a seguinte afirmação:

> Ainda nessa fase de pré-jogo, pode ser necessário especificar uma arquitetura básica de produto, de forma a reduzir os riscos de decisões tardias que invalidariam o que já foi produzido. Pode ser também necessário criar ou adaptar uma infraestrutura que servirá de suporte para o desenvolvimento do produto. A ideia é gerar apenas o mínimo necessário e suficiente para reduzir os riscos, mas sem engessar o desenvolvimento do produto nem gerar grandes desperdícios.

Ou seja, é nessa fase onde começa-se a identificar os principais impedimentos do time de desenvolvimento e também é o momento no qual todos tem acesso ao *Product Backlog*, uma lista ordenada, incompleta e dinâmica de itens e/ou funcionalidades que representam o que o PO acredita que será desenvolvido ao longo do projeto. Após esse período, o time *Scrum* começa a trabalhar efetivamente nas funcionalidades do projeto, *sprint* por *sprint*, até o término do projeto.

Um aspecto muito importante que deve ficar claro para todos é que apenas o PO tem o poder para mexer no *Product Backlog*, onde ele vai identificar os itens que tem mais retorno de investimento (ROI) para o cliente e priorizá-los conforme sua necessidade. A ideia aqui é sempre gerar o mínimo necessário para o time de desenvolvimento começar a trabalhar. Outro fator importante para levar em

consideração é a afirmação de Brod (2013, p. 13) “o scrum, por sua simplicidade e suas práticas rápidas e eficazes, é o caminho para o paraíso e uma tentação para o inferno”, pois é necessário o mínimo de documentação e também que todas as cerimônias sejam respeitadas.

Portanto, percebe-se que o *scrum* representa uma ruptura grande com o modelo tradicional *Waterfall*, respeitando todos os valores ágeis e que funciona muito bem com outros métodos ágeis, buscando sempre potencializar a produtividade do time. Outro ponto interessante, é que esse *framework* funciona também em outras áreas e não apenas no âmbito profissional. Brod (2013, p. 145) deixa isso bem claro ao afirmar: “quando você menos percebe, tudo na sua vida vai se tornando um projeto a ser realizado através de uma sequência de *sprints*”, ou seja, os valores ágeis passam a constituir a pessoa de uma tal forma que todos os planos se tornam uma funcionalidade a ser desenvolvida no decorrer da vida.

2.3 *Agile UX*

Atualmente, o mundo ágil tem evoluído bastante dentro e fora da área de desenvolvimento de *software*. Ao pensar no time de desenvolvimento do *scrum* (geralmente composto por desenvolvedores, testadores e *designers*), foi visto que a necessidade do mesmo em ser multidisciplinar se faz presente. Porém, para que isso ocorra com mais facilidade, é preciso que as diferentes áreas envolvidas comuniquem-se entre si. Sabbagh (2013, p. 13) deixa muito claro essa ideia quando afirma que “trabalhar em equipe é considerado essencial para o sucesso do projeto”, ou seja, a interação, a comunicação e o alinhamento entre os integrantes do time é de suma importância.

Com a difusão da cultura ágil, tem-se percebido que o trabalho em conjunto de outras áreas tem enriquecido profundamente os incrementos (partes menores entregáveis de um *software)* buscando agregar mais valor para o cliente, surgindo ou adaptando métodos para que aumentem o envolvimento do time e, consequentemente, o êxito do projeto. Baseado nesse pensamento, um método que tem ganhado destaque é o *Agile UX*. Entretanto, para entendê-lo melhor, primeiramente, precisa-se conhecer alguns conceitos que o compõem, como o *design thinking* e o *lean ux*.

2.3.1 Design Thinking

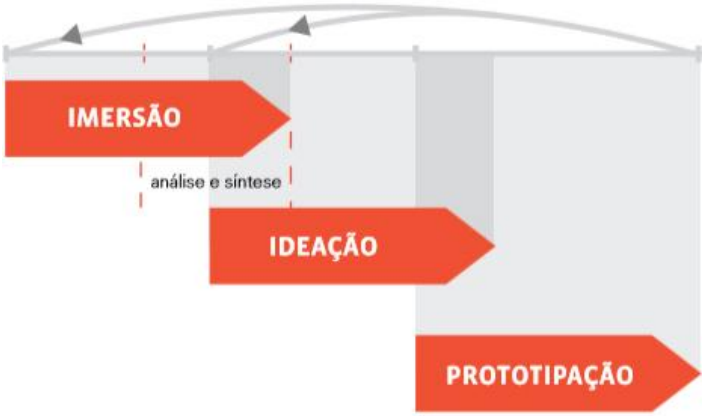
Ao pensar em *design thinking*, deve-se entender que este não é bem um processo e sim uma filosofia (ou uma forma de enxergar o design) que busca soluções procurando analisar, principalmente, o usuário e seu contexto sob diversos ângulos e perspectivas. Segundo Vianna *et al.* (2012, p. 12)

> Foi buscando novos caminhos para inovação que se criou hoje o que é conhecido como *'Design Thinking'*: uma abordagem focada no ser humano que vê na multidisciplinaridade, colaboração e tangibilização de pensamentos e processos, caminhos que levam a soluções inovadoras para negócios.

Ou seja, o principal objetivo dessa abordagem é procurar entender o usuário de uma forma completa, como suas culturas, suas experiências, suas emoções, seus comportamentos, entre outros aspectos, trazendo uma riqueza de informações a fim de inspirar ideias novas para projetos e negócios.

Essa abordagem é divida em três aspectos fundamentais, são elas: imersão, ideação e prototipação (figura 04).

Figura 4: Esquema representativo das etapas do *Design Thinking*



Fonte: VIANNA et al. (2012, p. 18)

A primeira fase é a imersão, onde toda a equipe do projeto (e não apenas *designers)* procuram entender o problema a ser resolvido e todo seu contexto, considerando algumas variáveis como o cotidiano do usuário, seu comportamento, entre outros. Vianna *et al.* (2012) divide essa fase em dois momentos: preliminar (tem como objetivo o entendimento do problema inicial. Ideal para definir o escopo de um projeto e suas fronteiras) e em profundidade (identificar as necessidades e oportunidades que irão direcionar as soluções a serem pensadas na próxima etapa).

Após os dados levantados no processo de imersão, é preciso analisá-los e sintetizá-los. Esse momento encontra-se entre as fases de imersão e ideação, pois permite a criação de *insights* (identificações de uma oportunidade), nos quais vão levar à geração de alguns artefatos (personas, diagrama de afinidades, mapas conceituais, etc) que auxiliem a compreensão do problema.

Em seguida tem a fase de ideação. Vianna *et al*. (2012, p. 99) define que "essa fase tem o intuito de gerar ideias inovadoras para o tema do projeto", ou seja, a equipe se utiliza de ferramentas e técnicas, considerando sempre as informações obtidas na fase de imersão (análise e síntese dos dados). Recomenda-se, nesse momento, a participação de pessoas de todas as áreas disponíveis dentro de uma organização e fora dela (usuários que irão se servir da solução, por exemplo).

A última fase é a de prototipação, cuja equipe tem a função de validar as soluções geradas, no momento anterior, com o usuário. Esta, tem um valor fundamental a nível de projeto, pois reduz a possibilidade de retrabalho, após o produto ou serviço ser entregue para o cliente e, consequentemente, reduzindo os custos do projeto.

Por não ser uma abordagem linear, após o término da ultima fase, há duas possibilidades. Se o protótipo for bem sucedido, o processo continua em frente. Caso o contrário, volta-se a fase de imersão ou ideação com as melhorias levantadas pelos usuários, gerando novos protótipos para validá-los novamente. Nota-se, portanto, que é muito mais valioso para o projeto identificar o erro logo no inicio do processo do que no final.

2.3.2 *Lean UX*

Com o mercado cada vez mais exigente, o *lean thinking* também vem ganhando muito destaque, pois está sempre verificando o processo, procurando eliminar os desperdícios e, consequentemente, aumentando o ritmo de produção de uma organização. Este, é um conceito que vem sido utilizado há, pelo menos, 50 anos e teve seu início nas linhas de produção da Toyota (também conhecido como *Kanban*). Boeg (2012, p. 4) afirma que "a palavra *kanban* vem do japônes e significa 'cartão visual' (...) a palavra também é utilizada para descrever o sistema que vem sendo utilizado há décadas pela Toyota para visualmente controlar e equilibrar a linha de produção".

Apesar de ser um conceito bem difundido nas fábricas e montadoras, de acordo com Boeg (2012), essa ideia foi trazida para a área de desenvolvimento de *software* (manutenção e operação também) por David J. Anderson e Don Reinerstsen,

expandindo o conhecimento do *Lean* e o uso do *Kanban* para visualizar e otimizar o fluxo de trabalho.

Outro fator que deve-se levar em consideração, é que mesmo sendo aplicado em outro contexto, alguns princípios se mantiveram. Waldmann (2014, tradução nossa) destaca os três mais importantes que são utilizados na área de desenvolvimento. São eles:

- O princípio fundamental do *Lean* para o desenvolvimento de *software* é a "eliminação do desperdício", onde todo desperdício é um processo extra, defeitos, funcionalidades extras, etc.
- De acordo com Taiichi Ohno, um dos fundadores do *Lean Manufacturing*, desperdício é definido como qualquer coisa que não produz valor para o consumidor.
- *Designs* e protótipos não são úteis para o consumidor: eles apenas têm valor quando um produto novo é entregue.

Esses fundamentos ficam claros, quando Gothelf (2013, p. 13, tradução nossa) resume os princípios do *Lean UX* da seguinte forma

> Os princípios *Lean* são subjacentes do *Lean Startup*, aplicados no *Lean UX* de três maneiras: Em primeiro lugar, eles ajudam-nos a remover desperdícios do nosso processo de UX *design*. Em segundo lugar, eles nos guiam para a harmonizar nosso "sistema" de *designers*, desenvolvedores, gerentes de produto, engenheiros de qualidade, profissionais de marketing e outros em uma colaboração transparente multifuncional que trazem não *designers* para dentro do processo de design. Por último, e talvez o mais importante, é a mudança da mentalidade que ganhamos com a adoção de um modelo baseado em experimentação.

Ou seja, nota-se que esse método foi construído a fim de unificar os conceitos do *design thinking*, aperfeiçoá-lo com os princípios do manifesto ágil e os fundamentos do *Lean Startup (Learning Loops: build - mensaure - learn)*, buscando sempre a entrega do incremento com mais valor para o cliente, alinhando a comunicação do time e a importância da participação de todos os membros da equipe no processo de criação e desenvolvimento de ideias.

Baseado nisso, Gothelf (2013) determina 15 princípios que servem como base para o método. São eles:

- **Times multi-funcionais:** equipe com áreas de formações diferentes, tais como engenheiro de *softwares*, *marketing*, *designers*, sociólogos, antropólogos, entre outros. O *Lean UX* demanda um nível alto de colaboração entre todas as disciplinas, tendo elas um envolvimento continuo e diário. Recomenda-se essa

diversificação, pois os *insights* e as ideias tem muito mais valor para todos os envolvidos.

- **Pequeno, dedicado e posicionado:** o ideal é que o time não seja muito grande, devendo conter, no máximo, 10 membros. Todos eles devem estar participando do mesmo projeto e, de preferência, na mesma localidade (por exemplo: todos no mesmo laboratório), pois facilita a comunicação, o foco mantem-se no que está sendo desenvolvido e melhora o entrosamento entre eles.
- **Progresso = Resultados e Não Saídas:** O progresso é medido nos resultados e não nas saídas do fluxo de desenvolvimento. Segundo Gothelf (2013, p. 8, tradução nossa) "funcionalidades e serviços são saídas. O objetivo do negócio que querem alcançar, são considerados resultados", ou seja, medir o progresso do time pelas entregas que possuem mais retorno de investimento para o cliente.
- **Time focado no problema:** O envolvimento de todos para a resolução dos problemas do projeto mostra a confiança que a equipe tem em si, permitindo que eles utilizem a sua própria solução, aumentando o engajamento dos membros com relação ao produto em si.
- **Remoção de desperdícios:** Geralmente, os recursos da equipe são limitados, quanto mais desperdícios em seu processo são eliminados, mais rápido eles evoluirão.
- **Desenvolver em pequenos pacotes:** Projetar apenas o que for necessário para fazer o time prosseguir, evitando um grande *backlog* de ideias sem potenciais e com baixo valor para o cliente. Desenvolver em grandes pacotes deixa o time menos efetivo, pois os forçam esperar uma entrega de demanda alta por parte dos *designers.*
- **Descoberta continua:** Envolver sempre o usuário final no processo do time, procurando entender o que eles estão fazendo com o produto entregue e porque o estão fazendo. Essa prática é uma excelente oportunidade para validar novas ideias.
- **GOOB**: Acrônimo criado pelo professor de *stanford* Steve Blank que significa "*getting out of the building*". Ou seja, ir para campo com mais frequência para entender realmente os problemas e as necessidades de seu usuário final.
- **Compartilhar os entendimentos e *Anti-Pattern: Rockstars, Gurus, and Ninjas*:** procurar sempre coletivizar o conhecimento do time sobre aquilo que eles estão trabalhando, tirando a responsabilidade e dependência de apenas um ou poucos membros da equipe em terem tal conhecimento.

- **Externalizar seu trabalho:** procurar deixar visível as atividades que estão sendo desenvolvidas. Em geral costuma-se usar o quadro *Kanban* para que todos do time possam trabalhar em harmonia.
- **Fazer ao invés de analisar:** O desenvolvimento de uma primeira versão da ideia tem muito mais valor do que gastar metade de um dia discutindo seus méritos em uma sala de reunião.
- **Conhecer antes de escalonar:** Certificar que as ideias e soluções estão clara para todos antes de desenvolvê-las e escaloná-las. Gothelf (2013, p. 11, tradução nossa) afirma que "o *lean ux* é a favor de focar em primeiro aprender e depois escalonar". Ou seja, escalonar uma ideia que possui muitas dúvidas, faz com que o risco aumente consideravelmente.
- **Permissão para falhar:** Buscar sempre validar uma solução antes de desenvolvê-la, pois se for para errar, é melhor que isso aconteça no início do projeto e não no final, diminuindo os riscos de retrabalho e aumentando a maturidade da equipe.
- **Evitar documentações extensas:** como principio do manifesto ágil, documentações do produto muito grande, não agregam valor para o cliente. Gothelf (2013) deixa claro que as documentações não resolvem os problemas do usuário e sim um bom produto.

Ou seja, percebe-se que o Lean UX é um método que possui um potencial muito grande, sendo possível notar que alguns princípios estão totalmente alinhados com o *design thinking* e o *agile software development*. Essa mescla de princípios e conceitos enfatizam a riqueza do trabalho em equipe, no qual Sabbagh (2013) defende como uma das variáveis mais importantes para o sucesso do projeto.

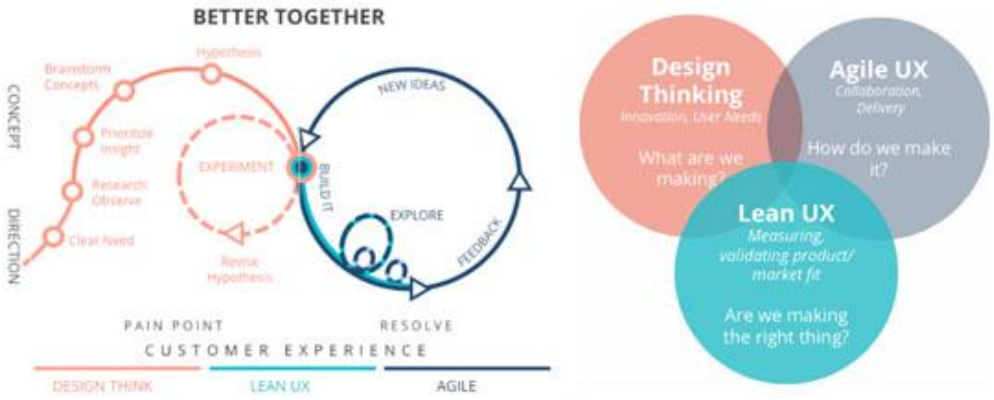
2.3.3 *Mas o que é Agile UX?*

Como consequência da popularização da cultura ágil e dos métodos de *User Experience*, pode-se dizer que o *Agile UX* é uma atualização dos métodos ágeis buscando uma maior aproximação do usuário final com processos e *frameworks* ágeis de desenvolvimento de *software*, como o *scrum* por exemplo, fazendo com que a participação dos *designers* dentro da equipe seja muito mais efetiva e agregue mais valor na entrega do incremento para o cliente.

Porém, para adequar os métodos de UX dentro do *scrum*, é preciso levar em consideração alguns aspectos característicos do *framework* e como as distintas áreas

irão comunicar-se. De acordo com Waldmann (2014), utiliza-se a imersão do *design thinking* para entender o usuário e o problema a ser resolvido. Em seguida, os princípios do *Lean UX* para a geração e validação de ideias e criação do MVP *(Minimum Viable Possible)*, servindo de base para ser implementado de modo colaborativo, iterativo e incremental, como mostra a figura 05.

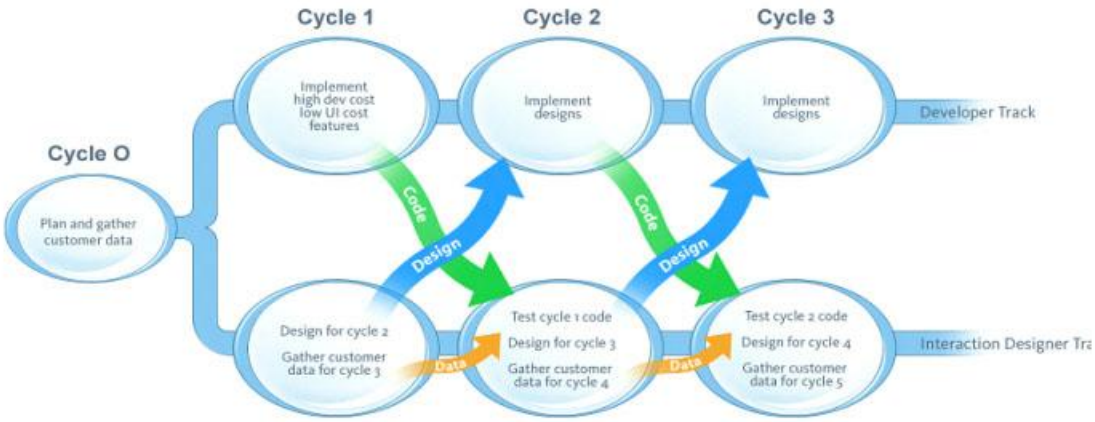
Figura 5: Esquematização dos conceitos que envolvem o *Agile UX*



Fonte: WALDMANN, Lydia. (2014).

Desirée Sy, uma das pioneiras na área de *agile UX*, apresentou uma solução de como conseguiu aplicar o método. Segundo a autora (2007, p. 117, tradução nossa) é preciso "encontrar um jeito de separar as iterações de *design* e as iterações da implementação", sendo necessário dividir o time em duas trilhas (*design* e desenvolvimento). Ou seja, a autora defende a ideia de que o time de *User Experience* esteja sempre a um ciclo a frente da *sprint* do time de desenvolvimento, responsabilizando-se em gerar protótipos validados pelos testes de usabilidades, deixando-os prontos para implementá-los. A equipe também assume o compromisso de realizar investigações contextuais para os fluxos que irão compor o sistema a cada dois ciclos de antecedência e também realizar testes de usabilidades para a versão atual desenvolvida, conforme a figura 06.

Figura 6: Uma das primeiras formas de *Agile UX* aplicada por Desireé Sy

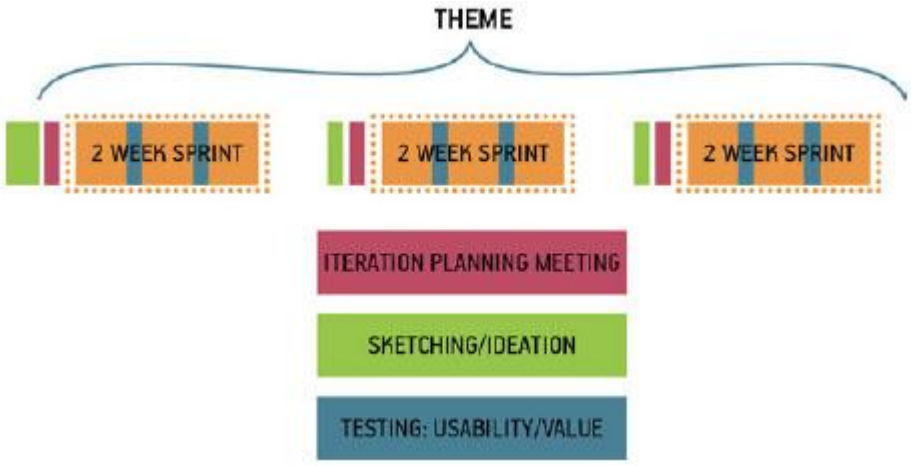


Fonte: SY, Desireé. (2007, p. 6)

Entretanto, Gothelf (2013) afirma que o modelo apresentado por Desireé Sy, é interessante para times que estão em um período de transição, pois apresenta alguns pontos que não vão de encontro com o pensamento do *Lean UX*, como a falta de foco no que está sendo desenvolvido, deixando de lado os benefícios da colaboração participativa e o desperdício desnecessário criando documentações que expliquem o que deve acontecer em cada *sprint* de *design*. O autor (2013, p. 98) deixa claro esses pontos ao fazer o seguinte questionamento: "Podem eles realmente construir os projetos específicos nas próximas duas semanas? Se não, o trabalho que começou a ser projetado, se torna desperdício".

Sendo assim, Gothelf (2013) demonstra como ele aplicou o *Agile UX* em seus projetos. No início de cada *sprint*, o time todo participa da ideação, utilizando *brainstorms*, esboçando as ideias e possíveis soluções. Em seguida, reúne-se todos os artefatos gerados e, juntamente com o time, os transformam em *users stories*, procurando sempre evoluí-las até conseguirem realizar a priorização das mesmas. Após essa fase, recomenda-se que seja planejado toda semana, sessões com usuários para a validação da ideia, usando como base os artefatos gerados na fase inicial da *sprint*, buscando o valor real que a funcionalidade (ou serviço) tem para o usuário final, redefinindo-se as prioridades de acordo com os resultados obtidos, como pode-se ver na figura 07.

Figura 7: *Lean UX* aplicado no *Scrum* por Jeff Gothelf



Fonte: GOTHELF, Jeff. (2013, p. 100)

Conclui-se, portanto, que o *Agile UX* é bastante recente e encontra-se em constante fase de aplicação e aprendizagem, não havendo um padrão para usá-lo. Outro fator que precisa ser levar em consideração é a importância de incluir o usuário (e também o cliente) em várias fases da *sprint* e não apenas no final de cada ciclo, procurando sempre fazer entregas significativas tanto para o negócio, como para o usuário.

2.4 *Agile UX* na *FPF Tech*

A FPF Tech é uma empresa sem fins lucrativos e, de acordo com o gerente de projetos, também um instituto de pesquisa, no qual atua em diversas áreas, tais como sistemas eletrônicos, design digital e usabilidade, medicina do trabalho e saúde educacional, capacitação tecnológica, tecnologia assistivas, produção cientifica, entre outras. Sua missão é a realização de projetos de pesquisa e desenvolvimento nas áreas de software, hardware, biotecnologia, capacitação e treinamento, transformando conhecimento em inovação.

A empresa trabalha com o *scrum* há, pelo menos, 8 anos e procura espalhar a cultura do *framework* dentro e fora da empresa, realizando diversos treinamentos em todos os níveis e setores. Atualmente, existem mais de 30 times (conhecidos como Liga da Justiça, Tartarugas Ninjas, Papaléguas, Yodas, etc) que utilizam os métodos ágeis em vários projetos, adaptando conforme suas necessidades e sempre respeitando todas as suas cerimônias.

Apesar de todos os times utilizarem o *scrum*, cada equipe procura criar sua maneira de trabalhar, servindo-se de outros métodos ágeis com o intuito de potencializar

a produtividade de todos seus integrantes, tais como *Kanban*, *Extreme Programming (XP)*, *Feature Driven Development (FDD)*, entre outros.

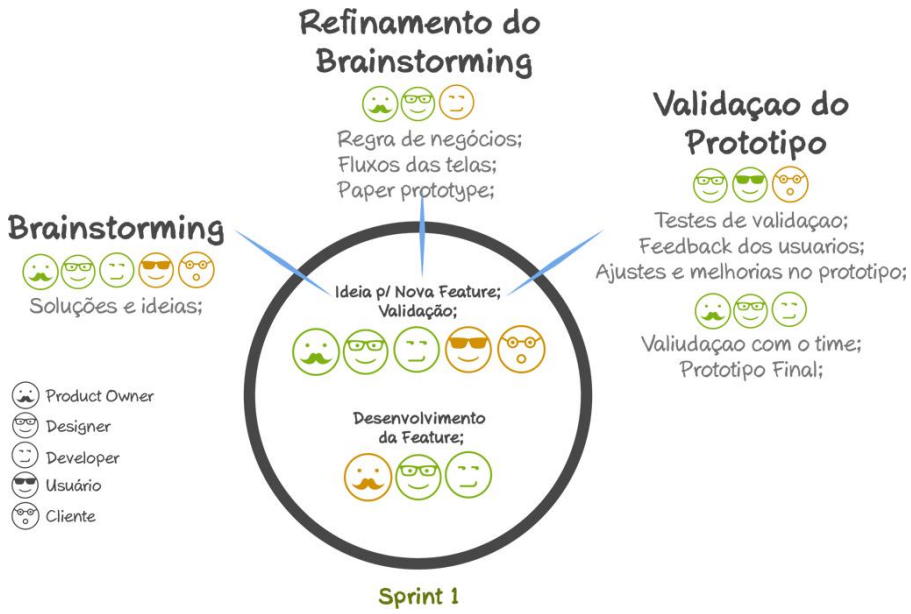
Um desses times é o *Wex.Terminators*, composto por dois desenvolvedores, um *product owner* e um designer/*scrum master*. Este time utiliza o *Scrum* e o *Kanban* e, de 37 *sprints*, tem entregue incrementos com valores em, pelo menos, 28 *sprints*. A particularidade do time é que estão conseguindo aplicar o *Agile UX*, conforme a descrição a seguir.

No início de cada *sprint*, o time reúne-se para planejar as funcionalidades a serem entregues e as atividades da *feature* que tem mais prioridade, criando assim o *sprint backlog*. Destaca-se que o time define as atividades das funcionalidades apenas no momento em que começam a desenvolve-las, seguindo um dos princípios do *lean startup* e eliminando o desperdício de uma reunião longa e com muitas chances de replanejamento.

Em seguida, o *product owner* fala sobre a ideia geral da próxima *feature* priorizada no *product backlog* para todo o time, começando uma sessão de *brainstorming* e gerando diversos rabiscos com possíveis ideias e soluções. Ao término dessa reunião, o *designer*, o *product owner* e um desenvolvedor (quando necessário) trabalham nas regras de negócios e no fluxo de telas da *feature* em questão, tendo como resultando alguns *paper prototypes* (em baixa resolução) e os principais requisitos da funcionalidade. Após esse processo, o time possui uma ideia pronta para ser validada.

Depois dessa fase, o *designer* reúne-se com os usuários finais procurando realizar alguns testes rápidos de validação com os protótipos gerados no momento anterior, resultando em *feedbacks* importantes. Com esse tipo de informação, aplica-se as sugestões e melhorias levantadas pelos usuários e, junto com o *product owner*, é feito outra validação com o time. O resultado é um protótipo final para aquela *feature*. Após esse momento, o *designer* junta-se com os desenvolvedores para focarem na funcionalidade a ser entregue na *sprint* recorrente, desenvolvendo e adaptando as mudanças que forem necessárias do *layout*, no qual teve seu protótipo validado no ciclo anterior (figura 08).

Figura 8: Processo de Validação de Ideias

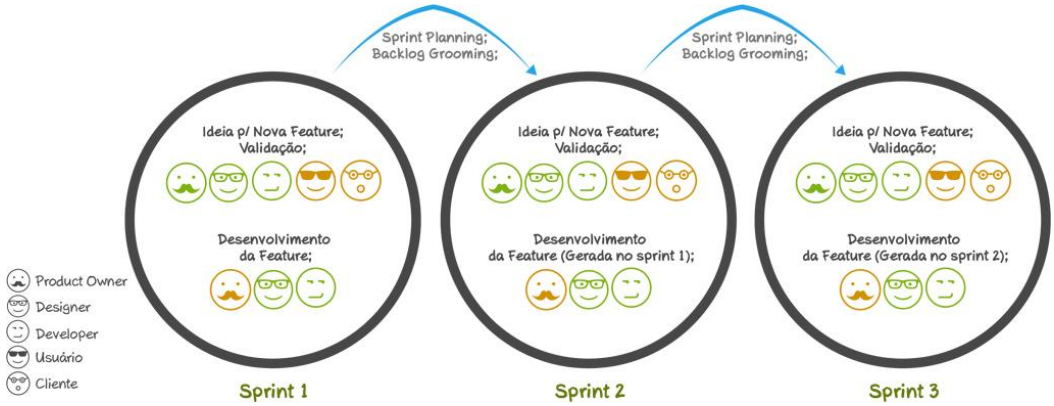


Fonte: Próprio autor

Observa-se que essa ideia de estar trabalhando novas funcionalidades sempre uma *sprint* a frente é totalmente focada para a criação de ideias e sua validação, gerando um protótipo. Nesse momento não é trabalhado a *User Interface (UI)*, sendo esta desenvolvida apenas quando a funcionalidade em si estiver prestes a ser codificada, evitando assim desperdícios de tempo caso a mesma seja repriorizada.

Outro ponto bastante importante, é que essa fase de validação entre o time e os usuários facilita bastante a questão do *Backlog Grooming* ou Refinamento do *backlog* (figura 09). De acordo com Sabbagh (2013, p. 252) "o *Product Backlog* é progressivamente atualizado e detalhado ao longo de todo o projeto", garantindo que ele seja ordenado, planejável, emergente e gradualmente detalhado. Com o *Product Owner* participando ativamente dos processos de geração e validação de ideias e soluções, notou-se que as reuniões de refinamento do *backlog* passaram a consumir menos tempo.

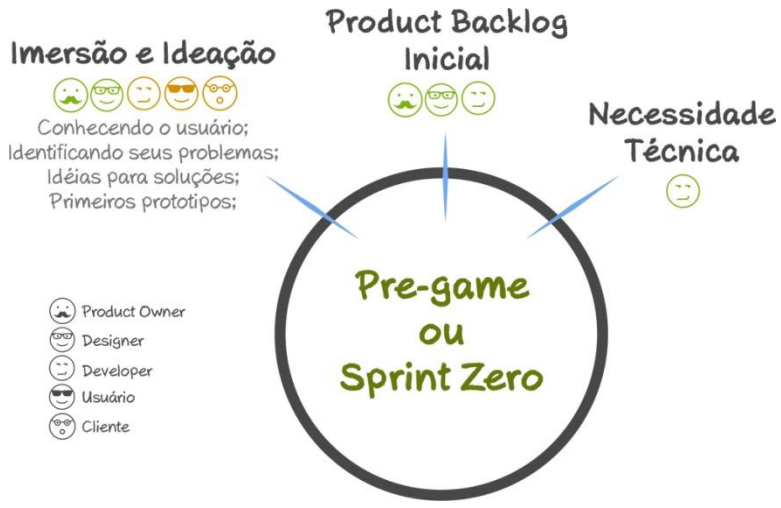
Figura 9: Visão geral do Agile UX aplicado.



Fonte: Próprio autor

Entretanto ao considerar novos projetos, tem-se a *sprint* zero ou pré-*game* (figura 10) onde ocorre a formação de uma equipe multidisciplinar (*product owner*, *designer* e, pelo menos, um desenvolvedor) para realizar a fase de imersão do *design thinking*, procurando conhecer os perfis dos usuários finais, quais os principais problemas que os cercam. Próxima fase é a ideação, buscando originar ideias e protótipos iniciais. Logo em seguida, todos os envolvidos no projeto se reúnem para gerar a visão do produto e um *product backlog* inicial (de modo macro), no qual vai sendo planejado melhor nas sessões de *brainstorming* recorrentes do *Backlog Grooming* no decorrer dos ciclos.

Figura 10: Pre game ou sprint zero.



Fonte: Próprio autor

Portanto, percebe-se que o modo que o time aplica o *Agile UX* segue a ideia de Desireé Sy de ter o *designer* trabalhando sempre um ciclo a frente com relação as novas

features, porém utiliza-se da ideia de Jeff Gothelf em relação aos princípios do Lean UX, como o time focado no problema, por exemplo. Ressalta-se também que, inicialmente, esse método enfrentou muitas barreiras, pois o time não estava em sintonia e a cultura ágil não era tão familiar para todos os seus membros, dificultando a comunicação e as dinâmicas utilizadas para gerar ideias de modo colaborativo, mas conforme o grupo foi evoluindo e a adoção aos princípios ágeis tornou-se efetiva, e o método passou a ser bem visto por todos, inclusive pelo cliente.

3. CONCLUSÃO

Ao término do artigo, concluiu-se que a cultura da agilidade tem ganhado muito destaque na área de desenvolvimento de *software* e os princípios do manifesto ágil são fundamentais para sua disseminação, buscando diminuir a burocracia e o "engessamento" que os métodos tradicionais impõem, ou seja, é dado maior valor para a comunicação e o dinamismo entre as pessoas do que suas relações com os processos de trabalho.

Ao pensar nos métodos ágeis, foi possível identificar diversos modelos. Entretanto, o que mais tem ganhado força é o *scrum*, pois é um *framework* que possibilita moldá-lo e adaptá-lo de acordo com a necessidade da equipe, tendo como fator essencial suas cerimônias e artefatos que são necessários para o aprimoramento da ferramenta, sua potencialização e a produtividade das pessoas que a utilizam. Outro ponto importante que se destaca no *scrum* é a sua simplicidade e a facilidade no aprendizado, possibilitando a sua utilização em outras áreas, seja ela profissional ou pessoal.

Com a abrangência do *scrum*, notou-se que o trabalho em equipe tem sido essencial para o bom andamento de um projeto. Para isso acontecer, é necessário que a comunicação, a confiança e o "entrosamento" entre seus membros sejam notórios. Porém, foi percebido que o envolvimento de outras áreas na atuação do projeto é bastante enriquecedor e agrega muito mais valor para o usuário final e, consequentemente, para o cliente.

Procurando expandir esse cenário, tem-se buscado fortalecer mais a participação do usuário final dentro dos moldes do *scrum* e também de profissionais de outras áreas, utilizando metodologias e técnicas bastantes difundidas na área de *design* cujo intuito é conhecer melhor o usuário sob diversos ângulos (seu contexto, sua cultura, suas experiências, por exemplo).

Seguindo esse pensamento, o *Agile UX* entra em cena. Um método que tem como base os princípios do ***Design Thinking*** (processo de imersão, análise e síntese de dados, com objetivo de conhecer melhor o usuário, e ideação, com o intuito em gerar *insights* e ideias de soluções baseados nos dados colhidos) e do ***Lean UX*** (criação do protótipo baseado no *MVP* e validação da ideia).

Outro ponto que ficou claro foi que, apesar do *scrum* e o *lean ux* possuírem valores alinhados com a cultura ágil, a diferença entre eles é perceptível. O *scrum* tem como norteador os princípios e valores do manifesto ágil (indivíduos e interações, software em funcionamento, colaboração com o cliente e responder a mudança) e garante a entrega das funcionalidades planejadas apenas no final do ciclo. Já o *lean ux* tem seus valores e princípios baseados no *lean startup (learning loop, learn - measure - build)* e também no *lean thinking* (remoção de desperdícios e entregas de valor para o cliente continuamente), buscando alguns pontos do *design thinking* para aperfeiçoá-lo. Entretanto, apesar das diferenças, é possível conciliar ambos os métodos.

Ainda sobre *Agile UX*, percebeu-se que não há um padrão em sua aplicação. Durante a pesquisa foram encontrados, pelo menos, dois modos diferentes. Uma delas feita por Desireé Sy (uma das primeiras em apresentar um estudo experimental sobre o tema) e por Jeff Gothelf (especialista em *Lean UX*), no qual analisa a forma que foi aplicado pela autora e apresenta uma outra visão.

Baseado nessa ideia, foi demonstrado como o time *Wex.Terminators* aplica o *agile UX* na FPF Tech, procurando contemplar a ideia de ambos autores. Ou seja, o time multidisciplinar gera ideias e possíveis soluções da *feature* priorizada no *backlog* no ciclo anterior ao seu desenvolvimento, validando com o usuário final. Ao terminar essa validação, a equipe foca em desenvolver a funcionalidade planejada para aquela *sprint* atual, possuindo algumas particularidades nesse processo, como o inicio de um projeto, por exemplo.

Inicialmente, esse método gerou bastante desconforto para o time, pois a afinidade entre seus membros era frágil e a cultura ágil não fazia parte do cotidiano de todos os integrantes. Entretanto, quando esses fatores começaram a se alinhar, a equipe aperfeiçoou-se de tal forma que as entregas passaram a ter mais retorno de investimento (ROI) para o cliente, pois o processo de geração e validação da solução apresentada trouxe mais segurança na utilização do *Agile UX* no decorrer do projeto, trazendo a responsabilidade para todos os membros do time, pois todos participaram.

Sendo assim, fica claro como a aplicação do método foi possível, deixando perceptível que a sua flexibilidade permite sua constante evolução, o amadurecimento do time e um cliente e usuário mais satisfeitos.

Por fim, sugere-se como pesquisas futuras um estudo tendo como ênfase na formação de times, pois foi uma das dificuldades encontradas na aplicação do método. Outra indicação é abordar outras formas de aplicação do modelo apresentado, enfatizando-o se obteve sucesso ou não e o porquê, permitindo a evolução continua do método tanto no meio profissional quanto acadêmico.

REFERÊNCIAS

ALLIANCE, Scrum. "***The State of Scrum: Benchmarks and Guidelines***." Disponível em: < https://www.scrumalliance.org/scrum/media/ScrumAllianceMedia/Files%20and%20PDFs/State%20of%20Scrum/2013-State-of-Scrum-Report_062713_final.pdf> acesso em: 15 de Fevereiro de 2015.

BECK, Kent. BEEDLE, Mike. BENNEKUM, Arie Van [Et. Al]. **"*Manifesto for Agile Software Development*"**. Disponível em: < http://agilemanifesto.org/iso/ptbr/> acesso em: 23 de Fevereiro de 2015.

BOEG, Jesper. **"*Kanban em 10 passos: Otimizando Fluxo de Trabalho em Sistemas de Entrega de Software*"** . Disponível em: < http://www.infoq.com/br/minibooks/priming-kanban-jesper-boeg> acesso em: 25 de Fevereiro de 2015.

BROD, Cesar. Scrum: **"Guia Prático Para Projetos Ágeis"**. São Paulo: Novatec, 2013.

GOMES. André Faria. **"*Agile*: Desenvolvimento de software com entregas frequentes e foco no valor de negócio"**. Editora: Casa do Código, São Paulo: 2013.

GOTHELF, Jeff. *"**Lean UX: Applying Lean Principle To Improve the User Experience**"*. Editora: O'Reily, California, Estados Unidos: 2013.

ROYCE, W. **"Managing the Development of Large Software Systems: Concepts and Techniques"**. In: Proceedings of IEEEWESCON. Piscataway, NJ, EstadosUnidos: IEEE Press, ago. 1970, p. 1-9.

SABBAGH. Rafael **"Scrum: Gestão ágil para projetos de sucesso"**. Editora: Casa do Código, São Paulo: 2013.

SCHWABER, Ken. SUTHERLAND, Jeff. **"Guia do Scrum – Um guia definitivo para o Scrum: As Regras do Jogo"**. Disponível em: <https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/2013/Scrum-Guide-Portuguese-BR.pdf> acesso em: 17 de Fevereiro de 2014.

SY, Desireé. **"*Adapting Usability Investigations for Agile User-centered Design"***. In: Jus - Journal of Usability Studies. Vol 2. Issue 3. p.112-132. Canadá: 2007;

VIANNA, Maurício. [Et al.]. **"Design Thinking: Inovação em negócios"**. Editora: MJV Press, Rio de Janeiro: 2012.

WALDMANN, Lydia. ***"Part 1 - Agile UX: Understanding the Agile World form a UX Perspective"***. Disponível em: < http://experience.sap.com/skillup/part-1-agile-ux-understanding-the-agile-world-from-a-ux-perspective/> acesso em: 27 de Fevereiro de 2015.