CONSIDERAÇÕES SOBRE O DESENVOLVIMENTO DE JOGOS POR MEIO DO FRAMEWORK SCRUM

Olívia Hamada[1]

Artigo recebido em setembro de 2015

RESUMO

Os jogos (*games*) se configuram como a contemporânea maneira das pessoas se relacionarem com as soluções tecnológicas e com os aplicativos. As expansões nas vendas dos dispositivos, dos aplicativos e os números de usuários, ano a ano, aumentam em proporções interessantes em termos de negócios e de consolidação de tendências nos hábitos. Desta forma, o desenvolvimento rápido, eficaz e de qualidade para os jogos se torna essencial para as empresas e para os parceiros que participam das suas elaborações, comercialização, suporte e disseminação. Utilizam-se metodologias diversas para esse desenvolvimento, tendo as metodologias ágeis as maiores possibilidades de sucesso. Este artigo trata do desenvolvimento dos jogos, as metodologias aplicadas e, em particular, do *framework* Scrum. Conclui-se que o Scrum é efetivamente uma boa opção para a construção de jogos, em função de suas características próprias.

Palavras-chave: Scrum. Desenvolvimento. *Software*. Metodologia. Ágil.

ABSTRACT

The games is regarded as the contemporary way people relate to technology solutions and applications. Expansions in sales of devices, applications and the number of users, year on year, increase in interesting proportions in terms of business and trends of consolidation in habits. Thus, the rapid, efficient development and quality for games becomes essential for businesses and the partners involved of its elaborations, marketing, support and dissemination. Use different methodologies for this development with agile methodologies with greatest chances of success. This article deals with the game development, methodologies and, in particular, the Scrum framework. Conclude that Scrum is actually a good choice for building games, due to its own characteristics.

Keywords: Scrum. Development. Software. Methodology. Agile.

[1] Pós-graduanda em Gestão de Projetos no Centro Paula Souza. E-mail: oliviahamada@gmail.com

1 INTRODUÇÃO

O mercado de games no mundo movimenta cifras significativas. Segundo a pesquisa realizada pela PwC (*PricewaterhouseCoopers*), empresa prestadora de serviços que trabalha de forma integrada na prestação de serviços de Assessoria Tributária e Empresarial e de Auditoria, cujos resultados estão presentes no Relatório Final do Mapeamento da Indústria Brasileira e Global de Jogos Digitais, elaborado pelo BNDES, em 2014.

> O mercado mundial de jogos digitais (*games*) movimentou US$57 bilhões em 2010, enquanto o de cinema, US$ 31,8 bilhões. Em 2011 o setor movimentou US$ 74 bilhões, e as previsões indicam que deverá ultrapassar US$82 bilhões em 2015. Em 2013, apenas o lançamento do jogo Grand Theft Auto V, que teve o custo de US$225 milhões, faturou US$800 milhões em 24 horas, recorde na história de produtos de entretenimento. O jogo Angry Birds já foi instalado em 500 milhões de celulares. No Brasil, estima-se que o mercado já esteja perto de US$3 bilhões (BNDES, 2014, p.6).

Os números impressionam. Indicam estarem ligados diretamente a uma indústria sólida e milionária. No Brasil, essa indústria se encontra em franco desenvolvimento. Contudo, o mercado ainda não está devidamente consolidado.

Mesmo sendo indústria em franco desenvolvimento, as técnicas usadas para se desenvolver *games*, apresentam-se análogas às técnicas de desenvolvimento de software, tendo em vista que um jogo, antes de tudo, é *software*, mesmo que esse seja destinado ao entretenimento.

A geração Z, indivíduos que nasceram em meados da década de 2000, é uma geração que surgiu ao mesmo tempo dos avanços dessas novas tecnologias. A pesquisa realizada pelo Ibope Mídia apresenta que a tecnologia é componente incorporado pela geração Z, algo que faz parte do seu cotidiano. Como por exemplo: enquanto estudam, assistem televisão, acessam redes sociais, ou ouvem músicas, tudo em paralelo (IBOPE, 2011).

Ainda a mesma pesquisa, mostra que a geração Z busca entretenimento, sendo a melhor e principal fonte disto a internet, seja para acessar, se atualizar, se conectar as redes sociais e/ou para jogar (IBOPE, 2011).

A geração Z (dos nascidos depois dos anos 2000) nasceu em meio à expansão da evolução tecnológica, em seus diferentes extratos sociais, o que leva à exploração dos ambientes digitais como forma de dialogar, trocar conteúdos e experiências do cotidiano, tendo como base tecnologias que expandem as capacidades humanas, tornando-se assim, uma espécie de extensão do corpo humano.

> Este deslocamento ocorre com a extensão do corpo em novas invenções e tecnologias sociais. Uma nova extensão estabelece um novo equilíbrio entre todos os sentidos e faculdades, de modo a conduzir a uma "nova visão" – novas atitudes e preferências em muitas áreas (MCLUHAN, 1996, p.147).

Nesse contexto em que as novas gerações estão expostas às tecnologias que alteram seus "sentidos e faculdades" e analisando as características presentes nos ambientes digitais, como imersão, agência e transformação, identifica-se a possibilidade de cogitar diversos tipos de experiências por meio dos jogos, seja pela capacidade de agir, produzir modificações e combinações das informações (MURRAY, 2003).

Surge a necessidade de buscar entender as novas tecnologias que surgem o tempo todo e adaptá-las aos interesses dessa nova geração. Porém, a tecnologia está em constante transformação, então é preciso analisar a demanda dessa transformação e desenvolver projetos, não só de jogos, que atendam todos os quesitos num tempo curto de prazo.

Uma forma de atender essa demanda é utilizar metodologias no desenvolvimento que consigam atender as necessidades específicas. Com a evolução da indústria dos jogos, os projetos se tornaram cada vez mais complexos e mais caros. O risco de se construir um jogo ruim, não aceito, não comercial, causaria importantes prejuízos. A fim de minimizar esses impactos, surgem necessidades de se utilizar metodologias para o desenvolvimento dos projetos, essas metodologias, por força das adaptações, também devem sofrer alterações para se ajustarem às demandas e às tecnologias presentes.

2 REFERENCIAL TEÓRICO

Antes da contextualização do processo de desenvolvimento e metodologias para projetos de *games*, é necessário esclarecer a evolução da indústria, perceber qual o seu

funcionamento e quais as evoluções ocorridas com o passar do tempo. A partir deste estudo, é possível identificar as melhores práticas para a construção de jogos, as metodologias mais adequadas às necessidades dos jogadores e à realidade/cenário dos projetos.

A indústria de *games* nasceu no século XX, por volta da década de 1950, durante a Guerra Fria. De acordo com Discovery (2007), os Estados Unidos e a União Soviética investiram em jogos que utilizavam a tecnologia da simulação como uma forma de simular o mundo real e prever resultados, tais como: o uso de novas armas e estratégias de batalha. Tais tecnologias eram utilizadas como ferramentas de apoio à estratégia e jogos de guerra.

Em 1958, o físico Higinbotham realizou a primeira experiência interativa de entretenimento em computador utilizando as tecnologias baseadas nos jogos da guerra Fria. O jogo *Tennis For Two* (tênis para dois) era exibido em um osciloscópio (figura 1). O jogo consistia em duas linhas e um ponto piscando, os jogadores controlavam os movimentos do ponto (bola), por meio dos botões do osciloscópio, passando-o por cima da linha que representava a rede.

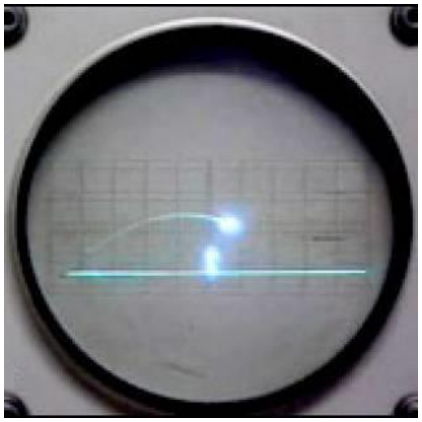

Figura 1 – Jogo tênis para dois
Fonte: Stony Brook University, 2013

Na década de 1980, o ex-*designer* da NamcoToruIwatani criou o Pac Man (figura 2). A ideia inicial era criar algo que não transmitisse violência porque em sua maioria os jogos de *fliperama* apresentavam essa característica. Além de colorido o desenvolvedor queria transmitir a impressão de personagens carismáticos. O público visado se compunha basicamente de garotas, em sua maioria de meninas que tinham o costume de comer sobremesa após as refeições, então o criador resolveu que o jogo deveria se relacionar à comida.

O Pac Man foi o primeiro jogo a possuir um personagem principal, o que incentivou os demais desenvolvedores e *designers* de jogos a apostar em jogos com um protagonista.

Figura 2- Tela Pac Man

Fonte: UOL Jogos, 2015

Com o fim da Guerra Fria, no final da década de 1980 e início da década de 1990, os primeiros jogadores de Pong cresceram, queriam novos jogos e novas experiências, somando-se que os adultos também queriam. Assim, a empresa Sega lançou o jogo Sonic, em 1991, utilizando tecnologias melhores comparadas as da Nintendo. Outra empresa, a Sony, antes parceira da Nintendo, desenvolveu novas e refinadas tecnologias para que se pudessem sustentar cenários mais realistas e a inovadora computação gráfica, tudo isso no console Sony Playstation, em 1994.

Figura 3 - Imagem do jogo Final Fantasy VII

Fonte: Comunidade Steam, 2015

Os jogos eram produzidos em CDs, permitindo maiores complexidades computacionais e com destacadas qualidades gráficas e sonoras, devido a capacidade de armazenamento. Um dos jogos lançados para o Playstation 1 foi o Final Fantasy VII, em 1997 (figura 3). O jogo tratava de lutas, traições e de outros assuntos que pareciam mais maduros, para o público de faixa etária maior. Com isso, muitas mais pessoas se sentiram atraídas pelo jogo e passaram a jogar o videogame.

Em 2001, o lançamento do GTA III (*Grand Theft Auto*) para Playstation 2 foi relevante sucesso. O jogador é inserido num cenário onde deve sobreviver e alcançar seus objetivos, podendo ou não se tornar um personagem pretendido e retomar a vidas desejadas. É como na vida real, o jogador pode fazer as suas escolhas (durante o jogo) da mesma maneira como poderia fazer na vida real, as cenas são mais curtas e o jogador pode explorar o ambiente.

Imagem do jogo GTA III
Fonte: Rockstar Games, 2015

Os videogames deixaram apresentar características amadoras, as histórias são mais profundas, mais complexas, os gráficos mais realistas, e as imagens produzidas em alta definição. Para que os personagens dos jogos, por exemplo, baseados em esportes, fizessem os mesmos movimentos dos atletas, buscando ainda melhor precisão, a EA Sports (empresa desenvolvedora e distribuidora de *games*) utilizou a captura de movimentos reais para tornar os jogos mais perfeitos. Capturou os dados dos movimentos de atletas humanos e inseriu-os nos personagens (figura 4).

Um fato merecedor de destaque: o mercado de jogos teve uma clara oportunidade devido ao infeliz evento ocorrido no dia 11 de setembro de 2001, porque fomentou o

desenvolvimento e elevou a produção de jogos militares e de simuladores, baseados em cenas e movimentos visando situações de segurança.

Figura 4 - Imagens da captura de movimento e do jogo Fifa 16
Fonte: EA Sports, 2015

Depois disso, a indústria de jogos evoluiu e continua evoluindo ainda mais, com o passar do tempo mais jogos e novas tecnologias aparecem permitindo que novos produtos sejam lançados, o que torna esse mercado tendente à lucratividade, apresentando contínuo crescimento nas vendas, no mercado mundial de jogos (tabela 1).

Segmento	2012	2013	2014	2015
Videogame console	$ 37.400	$ 44.288	$ 49.375	$ 55.049
Videogames portáteis	$ 17.756	$ 18.064	$ 15.079	$ 12.399
Games para celulares	$ 9.280	$ 13.208	$ 17.146	$ 22.009
PC Games	$14.437	$ 17.722	$ 20.015	$ 21.601
Total de videogames no mercado	$ 78.873	$ 93.282	$ 101.615	$ 111.058

Tabela 1 - Projeção do mercado mundial de videogames por segmento em milhões de dólares
Fonte: GARTNER, 2013

Os *games* atuais exigem muito da parte gráfica, consequentemente, de equipamentos que ofereçam suporte tecnológico ao software desenvolvido. Existe sim o risco de se criar um *game* que não faça sucesso, ainda mais em se tratando de jogos mais complexos, que envolvam várias equipes desenvolvedoras, constituídas de programadores, *designers*, músicos, artistas e estilistas de jogos.

Surge então, a necessidade de se adotar metodologias para o desenvolvimento de *games* e adaptá-las da melhor forma possível à redução dos riscos. Muitas metodologias já existem, dentre essas, o Modelo GWP (*Game WaterfallProcess*), Modelo XGD (*eXtreme Game Development*) e Modelo GUP (*Game UnifiedProcess*).

Este artigo apresenta, por conveniência metodológica e de forma simplificada, o modelo mais utilizado para o desenvolvimento de *games*: o Modelo GWP.

O GWP é uma adaptação do tradicional modelo cascata. Para Keith (2010), o modelo cascata ou *WaterfallModel*, é uma metodologia empregada para o desenvolvimento de projetos de *softwares* de grande porte dividido por uma série de fases. Cada fase é conduzida para uma fase subsequente mais valiosa que a anterior. Consiste basicamente em fazer o planejamento do *software* a ser construído, desenvolvê-lo, e por último, integrar todos os componentes e testá-lo.

Segundo Rucker (2003), o processo de desenvolvimento de *software* mais tradicional é baseado no modelo em cascata, as etapas de desenvolvimento realizam-se sequencialmente, isto é, a segunda etapa se inicia com o término da primeira, a terceira etapa principia com o termino da segunda, e assim sucessivamente (figuras 5 e 6).

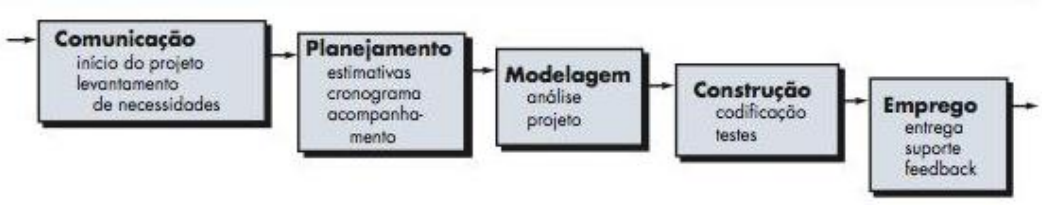

Figura 5 - Sequenciamento

Modelo cascata tradicional, Presman, 2011

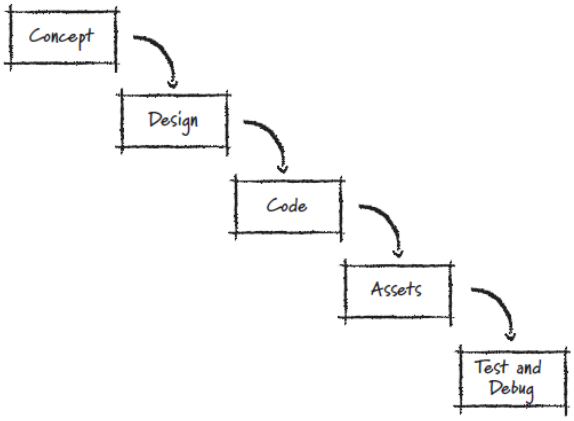

Figura 6 - Modelo cascata adaptado para projetos de games

Fonte: Keith, 2010

Ambos os modelos apresentados apresentam o ciclo de vida do projeto. Para os projetos de desenvolvimento de *games*, o modelo passa por uma alteração de nomenclatura, porém a lógica segue a mesma sequência.

> O modelo cascata é o paradigma mais antigo da engenharia de software. Entretanto, ao longo das últimas três décadas, as críticas a este modelo de processo fez com que até mesmo seus mais ardentes defensores questionassem sua eficácia. (PRESSMAN, 2011, p.60).

Por se tratar de uma abordagem de natureza linear, os requisitos devem ser bem compreendidos desde o início, e com uma mínima probabilidade de mudanças ao decorrer do desenvolvimento do software.

Hirama (2012) destaca que o modelo cascata é restrito, as atividades que envolvem o processo de desenvolvimento, como especificar, codificar e testar, se iniciam após o término da anterior, e o cliente final só participa no início e no fim do projeto. Entretanto, no cenário atual, é fundamental a presença do cliente no decorrer do desenvolvimento, pois se o software não atender as suas necessidades, pode correr o risco de ser rejeitado, assim, o projeto corre o risco de ser um fracasso.

Outro problema apresentado, por Sommerville (2007), é que no final de cada fase, é necessário um ou vários documentos aprovando-a, e assim seguindo para a próxima.

> Na prática, esses estágios se sobrepõem e trocam informações entre si. Durante o projeto, são identificados problemas com requisitos; durante a codificação são encontrados problemas de projeto e assim por diante. (SOMMERVILLE, 2007, p.44-45).

Pressman (2011) também apresenta problemas relacionados com o modelo cascata:

a) O fato de ser linear não impede que os projetos possam ter iterações, porém os projetos não seguem uma sequência, o que pode causar desorganização à equipe quando o projeto necessitar de mudanças;
b) O cliente nem sempre consegue especificar de forma clara quais suas necessidades, algumas vezes, elas vão surgindo com o decorrer do projeto, o

modelo necessita que essas necessidades sejam bem compreendidas e especificadas no início do projeto;

c) Se um erro grave, só for detectado na versão operacional (que é disponibilizada próximo ao final do projeto), o projeto pode ser desastroso;

d) Conduz a equipe a um estado de bloqueio, na qual uma equipe deve aguardar o término das atividades dependentes realizadas por outras equipes, a fim de prosseguir. Com isto, pode-se concluir que o tempo de espera poderia ser utilizado para um fim produtivo dentro do projeto.

Nota-se que apesar do modelo cascata ser a modelo tradicional mais utilizado, apresenta muitos problemas envolvendo a produtividade da equipe e respostas para mudanças, sendo que, alguns dessas anomalias poderiam ou deveriam ser resolvidas no início do projeto, assim evitando desdobramentos mais graves, não apenas pelo escopo mal definido, mas também pelos problemas de tempo, prazo e custo.

2.1 Metodologia de desenvolvimento ágil

As iniciativas inovadoras e o surgimento de tecnologias emergentes suportando produtos e também os *softwares* agregaram constantes mudanças nos desenvolvimentos. Surgiu então o "Manifesto para o Desenvolvimento Ágil de Software", que valoriza, segundo (PRESSMAN, 2011, p.81):

a) Indivíduos e interações acima de processos e ferramentas;

b) *Software* operacional acima de documentação completa;

c) Colaboração dos clientes acima de negociação contratual;

d) Respostas a mudanças acima de um plano.

Isso indica que, apesar de os componentes à direita possuírem seus valores, os componentes à esquerda são os mais valorosos.

A respeito disso, Amaral et al. (2011) destacam alguns dos princípios contidos no manifesto:

a) Prioridade pela satisfação do consumidor, por meio de entregas contínuas, de valor e o mais brevemente possível;

b) Mudanças de requisitos são bem-vindas mesmo em estágios avançados do desenvolvimento. Processos ágeis aproveitam mudanças em benefício da vantagem competitiva do cliente;
c) Entregar o produto funcionando em curto período;
d) Desenvolvedores e gestores devem trabalhar diariamente em conjunto;
e) Criar projetos com as pessoas motivadas. Confie nelas e dê suporte e ambiente para que o trabalho seja feito;
f) O método mais eficiente e eficaz de transmitir informações em um projeto é pela conversa "cara a cara";
g) Produto funcionando é a principal medida de progresso;
h) Processos ágeis promovem o desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente;
i) Atenção contínua a excelência técnica e ao design melhoram a agilidade;
j) Simplicidade. A arte de deixar de fazer trabalhos desnecessários é essencial;
k) Os melhores requisitos, arquiteturas e design surgem de equipes que praticam a autogestão;
l) Em intervalos regulares a equipe deve refletir sobre como se tornar mais eficaz. Após a reflexão deve-se reajustar-se de acordo com as necessidades percebidas.

Algumas características da abordagem ágil levam a facilidades em reduzir os custos de mudanças (respostas às mudanças; incentiva o "espírito em equipe"; torna a comunicação mais fácil entre todos os envolvidos; assume o cliente como parte da equipe de desenvolvimento; trabalha para eliminar a atitude de 'nós e eles'; facilita a entrega rápida do software operacional; e, diminui a importância dos artefatos intermediários (PRESSMAN, 2011 p.83).

3 METODOLOLGIA

A metodologia adotada neste artigo se baseia na revisão bibliográfica, descrição de metodologias de desenvolvimento de *software*, análises de casos e análises qualitativas.

4 DISCUSSÃO E RESULTADOS

Como todo processo de desenvolvimento de *software*, é difícil prever antecipadamente quais dos requisitos serão permanentes e quais serão alterados; ou ainda, qual o tempo estimado para o projeto, levando em consideração que para muitos projetos de *software* as atividades se realizam sequencialmente; e todas as etapas dos processos do projeto (desde a análise de requisitos até o desenvolvimento) são etapas imprevisíveis, ou seja, é possível que haja a necessidade de alteração no seu planejamento (PRESSMAN, 2011).

A partir dessas afirmações Pressman (2011) define que o processo ágil é adaptável, pois não há estabilidade, até mesmo pela prioridade do cliente, e pela constante análise da viabilidade dos requisitos iniciais, o que leva o projeto a sofrer ajustes, dinamicamente, em seus processos.

4.1 Scrum

Segundo Schwaber e Sutherland (2013), o Scrum é um *framework* utilizado para o desenvolvimento de produtos complexos e adaptativos, que está sendo aplicado desde 1990, e é formado por um conjunto de boas práticas de gerenciamento de projetos, tornando-os eficazes.

Esse *framework* emprega abordagem iterativa (repetições) e é baseado na teoria de controle de processos empíricos (SCHWABER; SUTHERLAND, 2013). O empirismo acredita que o conhecimento é adquirido por meio da experiência e da tomada de decisão, ou seja, não é necessário indicar como deve ser feito, apenas o que se quer como resultado. Diante disso, a equipe aprende coletivamente durante o processo e evolui com essa aprendizagem, assim conseguindo alcançar o objetivo.

O Scrum é um método de desenvolvimento ágil de *software* e seus princípios, os mesmo princípios citados por Amaral et al. (2011), vistos anteriormente, são utilizados para orientar todas as atividades que envolvem o desenvolvimento de *software*: requisitos, análise, projeto, evolução (construção e testes) e entrega (PRESSMAN, 2011).

Segundo Laubisch e Clua, o Scrum é direcionado à:

> [...] adaptabilidade a quaisquer mudanças em um projeto de software. Ao invés de voltar-se para o detalhamento em busca da ordem, o Scrum aceita que não há como controlar o caos do ambiente e foca em usá-lo ao seu favor.Seu foco principal está na comunicação – não só

> entre a equipe completa em si como também com o cliente, que tem uma participação ativa durante todo o desenvolvimento do processo Laubisch e Clua (2010, p.179).

Por se tratar de processos adaptáveis e flexíveis, o Scrum é utilizado em projetos complexos onde não é possível ou é minimamente possível prever mudanças (SCHWABER; SUTHERLAND, 2013). O conjunto de práticas oferecido por esse *framework* torna todas as atividades visíveis, permitindo a equipe envolvida ter visão mais ampla e exata no decorrer do projeto.

De acordo com Schwaber e Sutherland (2013), o Scrum possui três pontos bases (pilares) que apoiam a implementação de controle de processo empírico:

a) Transparência: Os processos devem ser visíveis para os responsáveis pelos resultados. Todos devem compartilhar um mesmo entendimento sobre o que está observando, não apenas para aqueles que realizam o trabalho, mas também para aqueles que aceitam o resultado;
b) Inspeção: Como o nome indica, é necessário realizar frequentes inspeções (desde que não atrapalhe a execução das atividades) nos artefatos do Scrum e nos processos para identificar o quanto antes as possíveis variações;
c) Adaptação: Se um processo está fora dos limites da especificação, determinado pela inspeção, é necessário ajustar e adaptar o processo para que o mesmo não ocorra novamente, assim, minimizando mais desvios.

4.1.1 Equipe

A equipe do Scrum, segundo Schwaber e Sutherland (2013) é formada por:

a) *Product Owner* (dono do produto): é a pessoa responsável pelo produto e também por fazer com que seu valor e o trabalho da equipe de desenvolvimento sejam maximizados. É o único responsável a gerenciar os itens do *Sprint Backlog* (artefato) e conhecer as necessidades do cliente final;
b) Equipe de desenvolvimento: é composta por profissionais que são responsáveis pela execução dos itens do *ProductBacklog* e realizar as entregas de uma versão usável ao final de cada Sprint;
c) *Scrum Master*: é responsável por garantir que os processos do *Scrum* sejam entendidos e aplicados. É o líder para a equipe de desenvolvimento, e deve garantir que a equipe esteja totalmente funcional e produtiva. Ajuda os que

estão fora da equipe a entender quais das suas interações são uteis ou não para a equipe de desenvolvimento.

4.1.2 Artefatos

Os mesmos autores definem os três principais artefatos que fornecem a transparência e as oportunidades para inspeção e adaptação:

a) *ProductBacklog*: é uma lista ordenada com todas as características, melhorias e os itens necessários para o desenvolvimento do produto de acordo com suas prioridades. Os itens descritos na lista contem a descrição, ordem, estimativa e valor para o desenvolvimento. Todas as mudanças no desenvolvimento do produto deverão partir dos requisitos descritos nesse documento, alteração de prioridade, adição ou remoção de requisitos;

b) *Sprint Backlog*: é uma lista de tarefas contendo um conjunto de itens presentes no *ProductBacklog* que serão desenvolvidas durante a *Sprint* (iteração). O objetivo do *Sprint Backlog* tem como objetivo tornar visível todo o trabalho que a equipe de desenvolvimento deve realizar;

c) *ProductIncrement*: ou incremento do produto, é o conjunto de todos os itens do *Sprint Backlog* que já foram desenvolvidos durante a *Sprint*. Cada incremento deve estar "pronto" em condições utilizáveis (software funcional e testado), independe da decisão do *ProductOwner* de liberá-lo ou não.

4.1.3 Eventos

Os eventos do Scrum, também conhecidos como processos, têm por finalidade proporcionar maior e melhor controle sobre todo o processo adquirindo rotinas de trabalho, aumentando a transparência ao decorrer do projeto e minimizando as reuniões não definidas pelo Scrum. Tais eventos são definidos como:

a) *Sprint*: é considerada o coração do *Scrum*. Esse processo é o ciclo de desenvolvimento que dura cerca de 15 dias a um mês (também conhecido como *time-boxed*). Todo projeto que utiliza *Scrum* funciona por meio de iterações (*Sprints*), a cada Sprint é realizado um ciclo de reuniões, que ao final desse processo é gerado um incremento como forma de retorno ao cliente;

b) *Sprint Planning Meeting*: todo trabalho de desenvolvimento a ser realizado na *Sprint* é definido na reunião de planejamento da *Sprint*, que tem um *time-boxed* de

no máximo oito horas (varia de acordo com o tempo de duração de cada *Sprint*, se ela tem duração menor o tempo do planejamento consequentemente é menor). Seu objetivo é planejar quais serão suas entregas (itens da *Sprint Backlog*), ou seja, funcionalidades para o próximo incremento e o trabalho que será realizado durante a iteração;

c) *Daily ScrumMeeting*: as reuniões diárias tem um *time-boxed* de 15 minutos em média, são realizadas para inspecionar o trabalho desde a última reunião. O intuito das reuniões diárias é para que a equipe de desenvolvimento possa ajustar o tempo das atividades e assim criar um plano de ação para as próximas 24 horas.

d) *Sprint ReviewMeeting*: após a conclusão da *Sprint* é realizado uma reunião de revisão da *Sprint*, onde a equipe de desenvolvimento apresenta ao *ProductOwner* e aos demais interessados no projeto as funcionalidades implementadas. A reunião de revisão da *Sprint* tem um *time-boxed* de 4 horas (varia de acordo com o *time-boxed* da *Sprint*). A reunião ocorre de forma informal e colaborativa, visando inspecionar o resultado da *Sprint* (incremento) e, se caso necessário, adaptar o *ProductBacklog* para a próxima *Sprint*, assim, "[...] a apresentação do incremento destina-se a motivar e obter comentários e promover a colaboração." (SCHWABER; SUTHERLAND, 2013, p.12).

e) *Sprint RetrospectiveMeeting*: a retrospectiva da *Sprint* é uma reunião com o *time-boxed* de três horas (varia de acordo com o *time-boxed* da *Sprint*). Essa reunião ocorre após a reunião de revisão e antes da reunião de planejamento da próxima *Sprint*. Para o Fabichak (2009), a reunião de retrospectiva propõe melhorias para a próxima reunião de planejamento da *Sprint*, tais melhorias implicam na interação entre os membros da equipe e na busca de soluções para possíveis erros presentes nos processos da mesma.

4.1.4 Processo Scrum

Sintetizando o processo Scrum, entende-se, de forma geral, que os seus eventos (*Sprints*) acontecem a cada 30 dias. Em cada ciclo é gerado uma lista de atividades, realizadas dentro do mesmo, possui ainda, subciclos quando se realizam reuniões diárias a fim de se manter atualizados sobre o andamento da *Sprint*, podendo assim tomar medidas o mais rápido possível, caso necessário. Ao final de cada *Sprint* é gerando uma nova funcionalidade (incremento) que será acrescentada com as funcionalidades já finalizadas (figura 7).

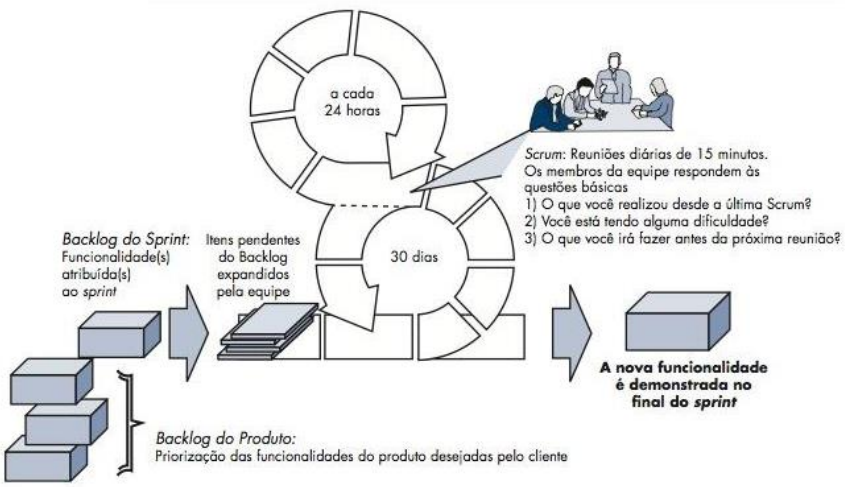

Figura 7 - Fluxo do processo *Scrum*

Fonte: Pressman, 2011

4.2 Scrum aplicado a games

Em pesquisa realizada pelo BNDES (2014), tendo 149 empresas respondentes, sobre a utilização metodologias no desenvolvimento de games, a tabela 2 apresenta os resultados.

Metodologia	Cascata (Waterfall)	PMBOK	Scrum	Nenhuma	Outro
Empresas	6	15	81	34	13
%	4,5%	11,3%	60,9%	26,6%	9,8%

Tabela 2 - Metodologia de desenvolvimento de software

Fonte: BNDES, 2014

Conforme observado, nota-se que significativa quantidade de empresas de desenvolvimento de *games* utilizam o Scrum, e 15,8% utilizam métodos tradicionais (modelo cascata e PMBOK), sendo que desses, apenas 4,5% utilizam o modelo cascata, demonstrando não ser uma modelo adequado para o desenvolvimento de projetos de *games*. O mais preocupante é que 26,6% das empresas de *games* não utilizam qualquer modelo para o desenvolvimento dos jogos, o que demonstra o grau de baixa maturidade dessas empresas e a tendência à falta de profissionalismo na indústria de games.

A utilização de metodologias voltadas à rápida adaptação às mudanças possui grande importância no desenvolvimento de *software*, por sua flexibilidade em tratar de novas situações e imprevistos ao longo do projeto (LAUBISCH; CLUA, 2010).

Como apresentado na tabela 2, o Scrum tem maior aceitação dentro desse conceito de metodologias aplicado ao desenvolvimento de *games*. Isso ocorre porque prioriza a comunicação entre todos os envolvidos (LAUBISCH; CLUA, 2010).

Segundo Cohn e Miller (2007; 2008 apud Laubisch e Clua, 2010), não é viável a utilização de Scrum no desenvolvimento de *games* sem que passe por adaptações. Alguns aspectos presentes, tais como: perfil da equipe, ciclo de vida do produto e perfil do consumidor/cliente são elementos, que a priori, obrigam a adaptação do Scrum, visto que, os projetos possuem características diferentes de *softwares* tradicionais em diversos aspectos, e vice-versa.

Em quantidade, existem poucas bibliografias referentes à utilização do Scrum adaptado aos *games*. Pode-se notar que esse tema é recente, muitas empresas já o utilizam em seus processos, como visto na tabela, porém não existe documentação conceitual sobre sua aplicabilidade, a maioria da bibliografia encontrada parte de análises do *framework* de forma pura e estudo de casos, e o adaptam de acordo com as necessidades do projeto.

CONCLUSÕES

Os princípios da tecnologia agrupada de determinada maneira, que hoje se tornou *games*, surgiram em meio a Guerra Fria, onde o terror e o medo estavam instalados na mente da população. Essa tecnologia era utilizada como forma de auxiliar militares durante a guerra, simulando possíveis ataques. Naquele cenário, essa tecnologia também surgiu como um meio de distração para as pessoas, assim surgindo o conceito de *games*.

Diante de vários acontecimentos, a ideia do *game* foi evoluindo juntamente com os *hardwares*, que suportaram melhor a qualidade de computação gráfica, assim propiciando um cenário onde os games pudessem se tornar mais realistas e com boas condições de jogabilidade.

Em se tratando de desenvolvimento de *software* é possível notar sua alta complexidade, esse envolve várias áreas como: análise, negócio, codificações (desenvolvimento), entre outros, porém quando a questão é a produção de *games*, a complexidade é maximizada. Além de envolver as áreas presentes no desenvolvimento de *software*, envolvem outras, como design, arte visual e auditiva, textos, enredos, cenários, estudo de comportamento (quando envolvem personagens, como pessoas e/ou animais) etc.

Um *game* quando projetado para produzir experiência ou satisfação emocional por meio da diversão se torna mais complexo, visto que é difícil planejar um jogo que seja assertivo em relação aos seus efeitos no usuário (jogador). Portanto, o aspecto humano se torna fundamental para o sucesso do projeto.

Diante dessas complexidades surge a necessidade de aplicar metodologias que pudessem gerenciar e controlar os processos de desenvolvimento de *games*. A metodologia GWP (adaptação da metodologia tradicional *WaterfallModel* para games) não atende as necessidades do projeto, que atualmente exigem respostas rápidas e eficientes para as mudanças, pois a metodologia tradicional segue um fluxo linear em que todo o esforço é planejado antes do projeto ser desenvolvido, assim prevendo que os requisitos sejam compreendidos desde o início, e com uma mínima probabilidade de mudanças.

Com o propósito de atender de forma rápida as demandas do mercado de *games*, algumas metodologias foram criadas, implementadas e testadas, visando supri-las. Umas, no entanto, se tornaram obsoletas para a área de negócio da empresa que a adotou, e outras como o Scrum, que por possuir flexibilidade para adaptações, se propagou e conquistou espaços entre as empresas.

As características fundamentais do Scrum, tais como eventos, artefatos e os papeis da equipe, possibilitam conceituá-lo como um *framework*, isto é, não segue um conjunto de processos de forma sistemática e linear, mas sim, se apoia no foco das suas iterações e os resultados a serem alcançados.

REFERÊNCIAS

AMARAL, D. C. et al. Gerenciamento Ágil de Projetos: Aplicação em Produtos Inovadores. 1. Ed. São Paulo: Editora Saraiva, 2011.

BNDES. Relatório Final: Mapeamento da Indústria Brasileira e global de jogos. 2014. Disponível em: <http://www.bndes.gov.br/SiteBNDES/bndes/bndes_pt/Galerias/Arquivos/conhecimento/seminario/seminario_mapeamento_industria_games042014_Relatorio_Final.pdf>. Acesso em: 05 nov. 2014.

COMUNIDADE STEAM. Final Fantasy VII. Disponível em: <https://steamcommunity.com/app/39140/screenshots/>. Acesso em: 18 jul. 2015.

DISCOVERY. A Era do Videogame. 2007. Documentário. Acesso em: 28 jun. 2015.

EA SPORTS. FIFA 16: Momentos Mágicos. 2015. Disponível em: <https://www.easports.com/br/fifa/noticias/2015/fifa-16-attacking-moments-of-magic>. Acesso em: 18 jul. 2015.

FABICHAK, M. O uso de método Scrum em empresas de desenvolvimento de software de jogos. 2009. Disponível em: <http://docplayer.com.br/131462-O-uso-de-metodo-scrum-em-empresas-de-desenvolvimento-de-software-de-jogos.html>. Acesso em: 18 jul. 2015.

HIRAMA, K. Engenharia de Software: Qualidade e Produtividade com Tecnologia. Rio de Janeiro: Elsevier Editora Ltda., 2012.

IBOPE. Gerações Y e Z: Juventude Digital. 2011. Disponível em: < http://www.ibope.com.br/pt-br/noticias/Documents/geracoes%20_y_e_z_divulgacao.pdf>. Acessoem: 07 nov. 2014.

KEITH, C. Agile Game Development with Scrum. 1. Ed. Boston: Addison-Wesley, 2010.

LAUBISCH, A.; CLUA, E. Scrum4Games: Uma aplicação do Scrum para projetos de games focada em game design. SBGames, 178-187, 2010. Disponível em: < http://www.sbgames.org/papers/sbgames10/artanddesign/Full_A&D_21.pdf>. Acesso em: 18 jul. 2015.

MCLUHAN, M. Os meios de comunicação como extensão do homem. 8. Ed. São Paulo: Cultrix, 1996.

MURRAY, J. H. (tradução de ElissaKhouryDaher e Marcelo Fernandez Couzziol). Hamlet no Holodeck: O Futuro da Narrativa no Ciberespaço. São Paulo: Editora Unesp, 2003.

PRESSMAN, R. S. (tradução de Ariovaldo Griesi e Mario Moro Fecchio). Engenharia de Software: Uma Abordagem Profissional. 7. Ed. Porto Alegre: AMGH, 2011.

ROCKSTAR GAMES. Grand Theft Auto 3. Disponível em: <http://www.rockstargames.com/grandtheftauto3/flash/infoScreens/index.html/>. Acesso em: 18 jul. 2015.

SCHWABER, K.; SUTHERLAND, J. Guia do Scrum. 2013. Disponível em: <http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf>. Acesso em: 13 maio 2015.

SOMMERVILLE, I. Engenharia de Software. 8. Ed. São Paulo: Pearson Education do Brasil, 2007.

STONY BROOK UNIVERSITY.Game Studies Collection. Tennis For Two. Nova Yorque.2013. Disponível em: <http://www.stonybrook.edu/libspecial/videogames/tennis.html>. Acesso em: 18 jun. 2015.