UNIVERSIDADE FEDERAL DO CEARÁ
CAMPUS QUIXADÁ
BACHARELADO EM ENGENHARIA DE SOFTWARE

CARLOS EDUARDO FREITAS

INTEGRAÇÃO DO FRAMEWORK SCRUM COM ETAPAS ARTÍSTICAS E PRODUÇÃO DE UMA FERRAMENTA PARA O DESENVOLVIMENTO DE JOGOS

QUIXADÁ
2018

CARLOS EDUARDO FREITAS

INTEGRAÇÃO DO FRAMEWORK SCRUM COM ETAPAS ARTÍSTICAS E PRODUÇÃO DE UMA FERRAMENTA PARA O DESENVOLVIMENTO DE JOGOS

Monografia apresentada ao curso de Engenharia de Software da Universidade Federal do Ceará, como requisito parcial à obtenção do título de Bacharel/Tecnólogo em Engenharia de Software.
Área de concentração: Computação.

Orientadora: Profª. Dra. Paulyne Matthews Jucá.

QUIXADÁ
2018

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

F936i Freitas, Carlos Eduardo.
Integração do framework scrum com etapas artísticas e produção de uma ferramenta para o desenvolvimento de jogos / Carlos Eduardo Freitas. – 2018.
48 f. : il. color.

Trabalho de Conclusão de Curso (graduação) – Universidade Federal do Ceará, Campus de Quixadá, Curso de Engenharia de Software, Quixadá, 2018.
Orientação: Prof. Dr. Paulyne Matthews Jucá.

1. Processo. 2. Desenvolvimento . 3. Jogos de computador. I. Título.

CDD 005.1

CARLOS EDUARDO FREITAS

INTEGRAÇÃO DO FRAMEWORK SCRUM COM ETAPAS ARTÍSTICAS E PRODUÇÃO DE UMA FERRAMENTA PARA O DESENVOLVIMENTO DE JOGOS

Monografia apresentada ao curso de Engenharia de Software da Universidade Federal do Ceará, como requisito parcial à obtenção do título de Bacharel/Tecnólogo em Engenharia de Software.
Área de concentração: Computação.

Aprovada em: ___/ ___ / ___.

BANCA EXAMINADORA

_____
Profª. Dra. Paulyne Matthews Jucá (Orientadora)
Universidade Federal do Ceará (UFC)

_____
Profª. Ma. Antonia Diana Braga Nogueira
Universidade Federal do Ceará (UFC)

_____
Prof. Ma. Jeferson Kenedy Morais Vieira
Universidade Federal do Ceará (UFC)

AGRADECIMENTOS

A minha mãe e minha vó que sempre me apoiaram em minhas decisões.

A minha namorada Emilly que sempre me incentivou sendo uma pessoa incrível na minha jornada.

A minha Prof. Dr. Paulyne Matthews Jucá, pela excelente orientação tanto acadêmica como de vida, sendo mais uma mãe para mim.

Aos colegas da turma, pelos momentos de conhecimento compartilhado, aos meus amigos que viraram irmãos Robson e Roger pelos momentos de descontração e perseverança durante a graduação.

RESUMO

Os jogos mesmo sendo um software, sempre possuíram em sua essência de produção necessidades diferentes das encontradas em um software comum, isso se dá pelas multitarefas que fazem parte do desenvolvimento de um jogo como a produção artística e musical. Essa sempre foi uma das dificuldades em se utilizar processos na produção de jogos, a incompatibilidade. Este trabalho visa construir um processo flexível que se adeque as necessidades da produção de um jogo, incluindo novas etapas e remodelando um processo já utilizado para o desenvolvimento de software convencional. Além disso ele demonstra os resultados obtidos pelo processo ao ser executado na produção de um jogo em um contexto real. Com base nos resultados obtidos da execução do processo e análise das necessidades de melhoria no gerenciamento do projeto, notou-se que poderíamos melhorar nossos resultados, para isso desenvolvemos o Game Control Lab (GCL) que autuou como uma ferramenta de apoio ao processo. Ao final do trabalho são apresentados os resultados sobre a aplicação do processo e uma avaliação da ferramenta em funcionamento.

Palavras-chave: jogos. desenvolvimento. processo.

ABSTRACT

The games even being a softwares have always had in their essence of production different needs from the ones found in a common software, this is due to the multitasking that are part of the development of a game like artistic and musical production. This has always been one of the difficulties in using processes in game production, the incompatibility. This work aims to build a flexible process that adjusts the needs of the production of a game, including new steps and remodeling a process already used for the development of conventional software. In addition, it demonstrates the results obtained by the process when executed in the production of a game in a real context. Based on the results obtained from the execution of the process and analysis of the needs for improvement in the project management, it was noticed that we could improve our results, for this we developed the Game Control Lab (GCL) that autou as a tool to support the process. At the end of the work are presented the results on the application of the process and an evaluation of the tool in operation.

Keywords: games. development. procces.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

Por se tratar de uma área multidisciplinar que envolve produção artística, produção de software e requisitos imprecisos, o desenvolvimento de jogos é um ramo, muitas vezes, difícil de se trabalhar usando um processo de desenvolvimento de software previamente definido. Isso faz com que muitas empresas trabalhem com procedimentos "artesanais" criados a partir de necessidades vistas ao longo do tempo, ou até mesmo sem padrões que garanta a qualidade do produto (SANTOS, GÓES e ALMEIDA, 2012).

Muitas empresas de desenvolvimento de jogos adaptam processos de desenvolvimento de software conhecidos para as suas necessidades. Apesar de existirem alguns processos propostos na literatura como (SANTOS, GÓES e ALMEIDA, 2012), quando a Marmota Factory (MF) da UFC Quixadá resolveu desenvolver um jogo, foi necessário propor um processo específico para essa necessidade. Apesar do processo proposto não ser exclusivo para a MF, essa demanda serviu como motivação e fonte de inspiração para a proposta. Isso aconteceu, pois as equipes são mais reduzidas com dedicação de tempo parcial para o projeto e com a participação de voluntários sem alocação específica para a parte artística. Essas características são semelhantes às encontradas em empresas independentes de desenvolvimento de jogos (Indies), nas quais os profissionais utilizam o tempo livre para desenvolver seus projetos.

Foi escolhido como framework base o Scrum (SCHWABER, 2018) definido pelos autores originais. Por se tratar de uma metodologia ágil que possui maior flexibilidade em comparação a metodologias tradicionais (SILVA, SOUZA e CAMARGO, 2013). O Scrum tem em sua metodologia alguns pontos como a rapidez na produção, flexibilidade, integração da equipe e pouca documentação, para o cenário de jogos isso é ideal já que estamos tratando de processos que envolvem a criatividade. Possuir uma baixa rigidez quanto a documentação é importante já que em grande parte dos momentos de produção artística estão envolvidos esboços, desenhos em quadro que geralmente ficam registrados por meio de fotos. Outro fator é a integração, equipes de jogos são multidisciplinares como já mencionado, logo é necessário um acompanhamento de perto entre as áreas para manter a qualidade. A participação do cliente ajuda no desenvolvimento, onde ele pode ir validando versões previas do jogo. Entretanto existem etapas como definição de arte 2D, criação de modelos 3D, definição do Game Designer, e várias outras, que não são contempladas pelo processo (HAMADA, 2016) que o torna incompleto para o contexto de desenvolvimento de jogos.

Neste sentido este trabalho propõe uma adaptação do processo Scrum para que seja possível o desenvolvimento de um jogo estilo plataforma baseado na teoria do Big Bang. Esta adaptação vai ocorrer com base em estudos não sistemáticos sobre processos e suas aplicações no desenvolvimento de jogos. Para atender as necessidades encontradas dentro do âmbito do desenvolvimento de jogos, este processo adaptado deverá apresentar a inclusão de etapas artísticas logo na concepção da ideia, bem como em etapas oportunas. Desta forma, este processo pode ser amplamente usado pela comunidade, podendo ser aplicado e adaptado em outros projetos que necessitem de etapas artísticas agregadas ao desenvolvimento de jogos.

2 TRABALHOS RELACIONADOS

Em (SANTOS, GÓES e ALMEIDA, 2012) foi proposta uma metodologia para o desenvolvimento de jogos, baseada em metodologias ágeis e prototipação denominada Origame. O Origame divide o processo geral em três fases: Design e Projeto, Produção e Implementação. Na fase de Design, é feito o levantamento de requisitos do jogo e definidas as características que o jogo deve possuir. A fase de Produção fica responsável por criar os recursos necessários, sejam eles *sprites*, modelos 3D, áudios, vídeos que foram definidos e serão utilizados no jogo. Por fim, na fase de Implementação, será feita a união dos elementos estéticos à codificação, resultando em um protótipo digital, que virá a ser uma versão Alpha após os testes.

Em (LAUBISCH e CLUA, 2010), foi feita uma adaptação na metodologia Scrum para ser aplicada em projetos de jogos. Essa adaptação ocorreu a partir da análise de problemas encontrados no decorrer da aplicação do Scrum em uma empresa de desenvolvimento de jogos que faz parte de uma incubadora de empresas.

Embora o processo Origame seja bem estruturado, a etapa de produção não define de forma satisfatória como os recursos artísticos são produzidos para o jogo, como ocorre dentro desta etapa, ou se há algum tipo de técnica seguida. A adaptação do Scrum feita por (LAUBISCH e CLUA, 2010) demonstra a capacidade do processo em se adequar as mais variadas áreas de atuação, mas por se tratar de uma área diversificada que exige a necessidade de muitos elementos fora da produção do software em si, esta adaptação se mostra falha em não possuir dentro do processo uma cobertura geral dos passos necessários para a construção de um jogo, como por exemplo uma atividade voltada ao estudo da história do jogo, análise artística, produção de arte.

Diante da necessidade de ter um maior suporte a construção artística, tendo em vista processos já existentes para o desenvolvimento de jogos, mas que, por muitas vezes são informais, empresas com processos internos que não divulgam para a comunidade, além do pouco material voltado a própria produção artística, a maioria acontece sem seguir um processo, apenas segue pela intuição e criatividade do designer (DUALIBI e SIMONSEN, 2000). Neste sentido, se faz necessária a inclusão de etapas voltadas a produção de elementos artísticos dentro de um processo de desenvolvimento com foco na definição de uma estrutura que dê suporte a essa produção artística. Esse processo precisa ser de fácil adaptação para que possa aceitar a inclusão de novas etapas sem muitos problemas. Para isto, escolhemos o framework Scrum por ter como característica sua adaptabilidade e por apresentar semelhança com o processo adotado atualmente pelo MF, o que facilitaria a sua implantação.

3 OBJETIVOS

3.1 Objetivo Geral

Propor um processo de desenvolvimento de jogos através da integração do framework Scrum com elementos de produção artística, executar este processo e analisar os resultados para melhora do mesmo, por fim construir uma ferramenta de apoio para o processo.

3.2 Objetivos específicos

a) Escolher as técnicas de produção artísticas que o processo vai utilizar;
b) Integrar o Scrum com os elementos artísticos resultando em um novo processo;
c) Desenvolver um jogo utilizando esse processo;
d) Avaliar a aplicação do processo na construção do jogo;
e) Desenvolver uma ferramenta de apoio a aplicação do processo proposto.

4 FUNDAMENTAÇÃO TEÓRICA

Nesta seção, serão descritos os principais conceitos que fundamentam este trabalho. Iniciando com a explicação do framework Scrum e como este pode dar suporte ao desenvolvimento dos objetivos deste presente trabalho, outro conceito descrito durante esta seção será o processo criativo com um detalhamento de etapas deste processo.

4.1 Framework SCRUM

Há cerca de dez anos, o Scrum foi formalizado como um processo ágil. A sua aceitação e utilização por parte da comunidade deu-se rapidamente, sendo hoje uma metodologia padrão em várias companhias (KNIBERG, 2007).

O ciclo do Scrum acontece no decorrer de várias iterações bem definidas, chamadas de Sprint, cada uma destas iterações tem duração de 2 a 4 semanas. Antes de cada Sprint, realiza-se uma Reunião de Planejamento (*Sprint Planning Meeting*) onde o time (equipe) de desenvolvedores se reúne com o cliente (*Product Owner*) para definir e priorizar o que precisa ser feito, escolher e estimar as tarefas que o time deve realizar dentro de uma Sprint (SCHWABER, 2018).

Em seguida, acontece a execução das Sprints. Durante a execução, o time gerencia o andamento do desenvolvimento do projeto por meio de Reuniões Diárias Rápidas (*Daily Meeting*), não duram mais que quinze minutos, e observando o progresso no gráfico chamado *Sprint Burndown* (SCHWABER, 2018).

No final de cada Sprint, é feita uma revisão do produto gerado para verificar se tudo que foi definido na *Sprint Backlog* se apresentou devidamente implementado. Esta revisão acontece na Reunião de Revisão (*Sprint Review*), na qual o time demonstra o produto gerado na Sprint e valida se o objetivo foi atingido. Logo em seguida, realiza-se a Reunião de Retrospectiva (*Sprint Retrospective*), uma reunião de lições aprendidas com o objetivo de melhorar o processo/time e/ou produto para a próxima Sprint (SCHWABER, 2018).

Estão inclusos no framework Scrum alguns papéis bem definidos com tarefas e finalidades diferentes durante o processo, sendo estes o *Scrum Master, Product Owner, Scrum Team*, o cliente, o usuário e o gerente. A seguir uma descrição desses papéis de acordo com as definições de Schwaber Beedle (2007):

- ***Scrum Master*** - É responsável por garantir que o projeto seja realizado, gerindo a equipe de acordo com as práticas, valores e regras do Scrum e pelo avanço da equipe no projeto conforme planejado.
- ***Product Owner*** - É responsável por definir os itens que irão compor o *Product Backlog* e por priorizar estes itens nas *Sprint Planning Meetings*. O *Product Owner* pode representar os desejos de um comitê no *Product Backlog*, mas aqueles que desejam alterar a prioridade do item *Backlog* do produto devem dirigir ao *Product Owner*.

- ***Scrum Team*** - É o time de desenvolvimento do projeto. As equipes de desenvolvimento têm as seguintes características:
  - Eles são auto organizados. Ninguém (nem mesmo o *Scrum Master*) diz à equipe de desenvolvimento como transformar o *Product Backlog* em incrementos funcionais;
  - As equipes de desenvolvimento são multifuncionais, com todas as habilidades que uma equipe precisa para criar um produto;
  - O Scrum não reconhece títulos para os membros do *Scrum Team* que não seja o de desenvolvedor;
  - O Scrum não reconhece subequipes no *Scrum Team*;
  - Os membros podem individualmente ter habilidades especializadas e áreas de foco, mas a responsabilidade pertence ao *Scrum Team* como um todo.

Durante a Sprint o time deve controlar como as tarefas devem ser executadas, buscando sempre manter a organização. Durante a Sprint não deve existir interferência externa, esse é um dos principais papéis do *Scrum Master*, evitar que o time tenha qualquer desvio do objetivo traçado (SCHWABER, 2018).

O Scrum não foi feito para atender apenas as necessidades encontradas no desenvolvimento de software, mas criado no intuito de atender a qualquer projeto complexo, de qualquer área, principalmente os que propõem inovações. Neste sentido, é comum ver sua grande utilização na área do desenvolvimento de jogos, em sua maioria, são projetos complexos e multidisciplinares (KEITH, 2010).

Segundo (KEITH, 2010), algumas atividades, práticas e princípios propostos pelo Scrum se mostram interessantes quando aplicadas no desenvolvimento de jogos, são elas:

- **Planejamento ágil** - o planejamento deve ocorrer ao longo de todo o projeto e não concentrado em uma única etapa (KEITH, 2010). A prioridade fica para as *features* mais importantes, que serão inicialmente planejadas com mais detalhes; logo o que não é considerado como importante pode ser planejado mais tarde. Deve-se evitar no planejamento incrementos que não farão parte do jogo (AMBLER, 2009).
- **Priorização** - o time de desenvolvimento deve trabalhar sempre no que é priorizado, sendo trabalhado o mais importante primeiro. A fim de focar no desenvolvimento daquilo que apresenta ser mais divertido (KEITH, 2010).

- **Iterações rápidas** - Sprints de curta duração entre três a quatro semanas, garantem a entrega constante de funcionalidade permitindo um fácil acompanhamento da construção do jogo mediante seu progresso. Ao final de cada Sprint, é indicado que as novas funcionalidades sejam integradas para produzir uma nova versão do jogo (KEITH, 2010).
- **Teste** - o *feedback* vindo dos testes que ocorrem em vários momentos por pessoas diferentes é sem dúvidas de grande valia para tomadas de decisões futuras (GIBSON, 2007).
- **Conhecimento, experiência e adaptabilidade** – o conhecimento e experiência adquiridos durante o desenvolvimento do jogo são base para adaptações necessárias ao que se tem como prioridade, que é a diversão (KEITH, 2010).
- **Contato direto** - o Scrum incentiva sempre que possível e necessário a comunicação frente a frente (SCHWABER, 2018). No desenvolvimento de jogos é fundamental que haja discussões periódicas a respeito do conceito do jogo, da diversão e dos resultados dos testes.

4.2 Processo Artístico

A criatividade pode ser vista como a capacidade de criar coisas novas. Apesar da criatividade ser normalmente entendida como uma atividade livre e avessa as regras, existem muitas técnicas de criatividade e processos criativos.

Algumas técnicas de criatividade muito utilizadas incluem: brainstorms, os seis chapéus do pensamento, mapas conceituais, mapas mentais, caixas morfológicas, inversão de hipóteses e associação de ideias. Está fora do escopo deste trabalho apresentar as técnicas de criatividade. As técnicas podem ser utilizadas em diferentes atividades do processo criativo. Partes das etapas do processo criativo são usadas na definição do processo proposto nesse trabalho.

Um modelo pioneiro sobre processos criativos foi proposto por Graham Wallas, este lançou uma teoria chamada The Art of Thought (WALLAS, 1926). Nesta teoria Wallas sugere quatro estágios para o processo criativo: preparação, incubação, iluminação e verificação. Entretanto, essa teoria não detalha o modo como surgem as ideias durante o processo. O trabalho de Wallas serviu como base para outros processos mais recentes, um deles foi propostos por (DUALIBI e SIMONSEN, 2000), este se mostrou relevante para nosso

cenário por sugerir que não se pode seguir uma sequência, pois o processo criativo não é linear. É possível avançar ou retroceder, repetir ou realizar várias etapas do processo criativo ao mesmo tempo, propondo um ciclo com sequências dinâmicas

Ele foi originalmente criado para o marketing, mas é genérico o suficiente para ser adaptado para este trabalho. Os autores definem um processo criativo composto por 7 etapas: identificação do problema, preparação, incubação, aquecimento, iluminação, elaboração e verificação.

A identificação do problema é a etapa onde se pretende responder à pergunta "qual é o problema?". Essa etapa pretende avaliar as necessidades da tarefa e envolve o uso de muitas das técnicas de criatividade.

A preparação é a etapa de coleta de informações necessárias para entender o problema. Utiliza-se de perguntas como "o quê?", "quem?", "quando?", "onde?", "como?" e "por quê?".

A incubação acontece depois que todos os dados da preparação foram coletados. As pessoas conhecem suficientemente o problema a ser resolvido e agora precisam de tempo para refletir sobre ele. Os autores comentam que a incubação acontece com a mente em descanso.

O aquecimento é quando existe uma volta do trabalho consciente no desenvolvimento do problema. O uso de brainstorms pode acontecer nessa fase. Ela gera a iluminação que é quando a solução é criada pela primeira vez e corresponde ao momento "eureka".

A elaboração é onde a solução é construída de fato. Na verificação, a solução proposta é validada.

Para esse trabalho, o processo criativo descrito anteriormente foi simplificado e gerou 3 novas etapas no processo proposto: concepção da ideia, criação artística e desenvolvimento do design. A concepção da ideia é composta da execução completa do processo criativo, utilizando diferentes técnicas de criatividade para onde o problema a ser resolvido é "qual jogo será desenvolvido?". O artefato produzido ao final da fase de concepção da ideia é o GDD e dele é derivado o backlog do produto (ver a definição do processo Scrum).

Para a inserção destas etapas artísticas no processo foi definido a criação de um subprocesso Ad-hoc que conteria todas estas fases artísticas, segundo a notação do *Business Process Model and Notation* (MODEL, 2011) as atividades contidas dentro de um subprocesso Ad-hoc não possuem uma ordem de execução definida, logo, não é preciso executar todas as atividades. Este subprocesso foi pensado para ser executado pela equipe de design em paralelo

ao desenvolvimento lógico do jogo, neste sentido elas ocorrem após a definição do *Sprint Backlog* do jogo, no qual já possui um GDD contendo a ideia de como o jogo será, uma descrição dos elementos artísticos necessários para o jogo, e com base no *Backlog da Sprint* vai ser produzido no subprocesso as artes conceituais do jogo.

4.3 Jogos Digitais

Segundo (MARCELO e PESCUITE, 2009), os jogos podem ser divididos em duas formas: jogos reais e eletrônicos. Para jogos reais temos os jogos de tabuleiro ou de mesa, que possuem toda sua mecânica e jogabilidade por meio de uma visualização física de seus elementos, tendo como exemplo os jogos de cartas e RPGs de mesa. Já como representantes de jogos eletrônicos o autor define como participantes os jogos de vídeo game e computadores, bem como os de redes, onde estes expõem seus elementos de maneira virtual para os usuários.

Para uma definição mais literária e filosófica do que vem a ser um jogo podemos dizer que:

> Um game é uma atividade lúdica composta por uma série de ações e decisões, limitada por regras e pelo universo do game, que resultam em uma condição final. As regras do universo do game são apresentadas por meios eletrônicos controlados por um programa digital. As regras e o universo do game existem para proporcionar uma estrutura e um contexto para as ações de um jogador. As regras também existem para criar situações interessantes como objetivo de desafiar e se contrapor ao jogador. As ações do jogador, suas decisões, escolhas e oportunidades, na verdade, sua jornada, tudo isso compões a "alma do game". A riqueza do contexto, o desafio, a emoção e a diversão da jornada de um jogador, e não simplesmente a obtenção da condição final, é que determinam o sucesso do game (SCHUYTEMA, 2008)

5 PROCEDIMENTOS METODOLÓGICOS

Neste tópico iremos abordar os procedimentos que foram seguidos para a realização deste trabalho.

5.1 Identificar as técnicas de produção artísticas que o processo vai utilizar

Esta etapa teve como objetivo identificar técnicas artísticas que pudessem ser melhor utilizadas na criação de personagens e cenários de um jogo. Para isso, foi realizado um

estudo não sistemático da literatura para encontrar as técnicas artísticas que se enquadram melhor as necessidades apresentadas na criação de elementos artísticos dentro de um jogo. Para este trabalho, foi escolhido o Processo Artístico de (DUALIBI e SIMONSEN, 2000) pela sua flexibilidade, podendo ser usado na criação de diversos elementos necessários como: cenário, personagens, itens, objetos e outros.

5.2 Adaptar o processo Scrum

Esta etapa visa elaborar um processo que seja baseado no framework Scrum, incorporando técnicas de produção artística selecionadas na etapa anterior. Desta forma, os envolvidos que utilizarem o processo podem construir um jogo completo com todos os elementos necessários, sejam eles componentes visuais como modelos 3D/2D, artes conceituais dos personagens e cenário, Game Designer Detalhado contendo as especificidades do jogo, código implementado da mecânica e funcionalidade do jogo.

Os passos para realização desta etapa são: Estudo da produção artística durante o desenvolvimento jogos, definir os momentos em que acontecerá a incorporação das atividades artísticas no processo de desenvolvimento Scrum e finalmente Modelagem do processo.

- Estudo da produção artística durante o desenvolvimento de jogos - Primeiramente, é preciso analisar quando a produção artística se faz necessária durante a produção de jogos, para isso será feito um estudo de como ocorre atualmente essa produção, embora esta atividade como já dito anteriormente na seção 1, aconteça de maneira muitas vezes aleatória em meio a necessidade, sem seguir um fluxo, o intuito deste estudo é identificar os melhores momentos onde essa atividade acontece, dentro da construção de um jogo.
- Definir os momentos em que acontecerá a incorporação das atividades artísticas no processo de desenvolvimento Scrum - Será feita uma análise do Scrum, a fim de identificar em quais etapas do processo deverá ser incluída a atividade de produção artística, com base no estudo feito anteriormente.
- Modelar o processo - O processo será definido usando a ferramenta Bizagi (BIZAGI, 2016) a partir das adaptações feitas, estas adaptações levam em consideração todas as inclusões levantadas no passo anterior.

5.3 Desenvolver um jogo utilizando esse processo

Nesta etapa, foi produzido um jogo desde seu início, seguindo o processo proposto. Para garantir que isso tenha acontecido, houve uma supervisão por parte do Scrum Master dentro da equipe. O Scrum Master nesse contexto foi o responsável por garantir que a equipe entendesse e executasse o processo definido, explicando a equipe as práticas e regras a serem adotadas.

5.4 Avaliar a aplicação do processo na construção do jogo

Em (VOSS, TSIKRIKTSIS e FROHLICH, 2002) eles enfatizam que pesquisas científicas devem conter métodos e regras que devem ser utilizadas no processo de coleta dos dados, informar com quem ou onde as informações poderão ser coletadas. Neste sentido, o foco das diretrizes aqui estabelecidas para a pesquisa é avaliar o nível de satisfação e sucesso na construção do jogo ao utilizar o processo.

As avaliações do processo foram realizadas com a aplicação de um estudo de caso, que levou em consideração a construção de um jogo por uma equipe de pessoas utilizando o processo proposto. Neste trabalho usamos observação direta como técnica de estudo de caso para o levantamento dos dados da pesquisa.

Ao final foi feita uma análise dos resultados, mediante reuniões feitas pela equipe e anotações da observação, evidenciando os pontos positivos e negativos, as necessidades não atendidas e pontos para melhoria.

5.5 Desenvolver uma Ferramenta

Por fim foi construída uma ferramenta, que foi desenvolvida com base no processo proposto e sua modelagem visa sanar os pontos fracos encontrados na etapa anterior. Esta ferramenta foi construída com base também na metodologia de cards encontradas no Trello, já que esta se utiliza da api do Trello.

6 DEFININDO O PROCESSO

Na primeira etapa, onde foi feita a identificação das técnicas de produção artística, o resultado obtido foi a seleção das técnicas propostas por (DUALIBI e SIMONSEN, 2000)(ver

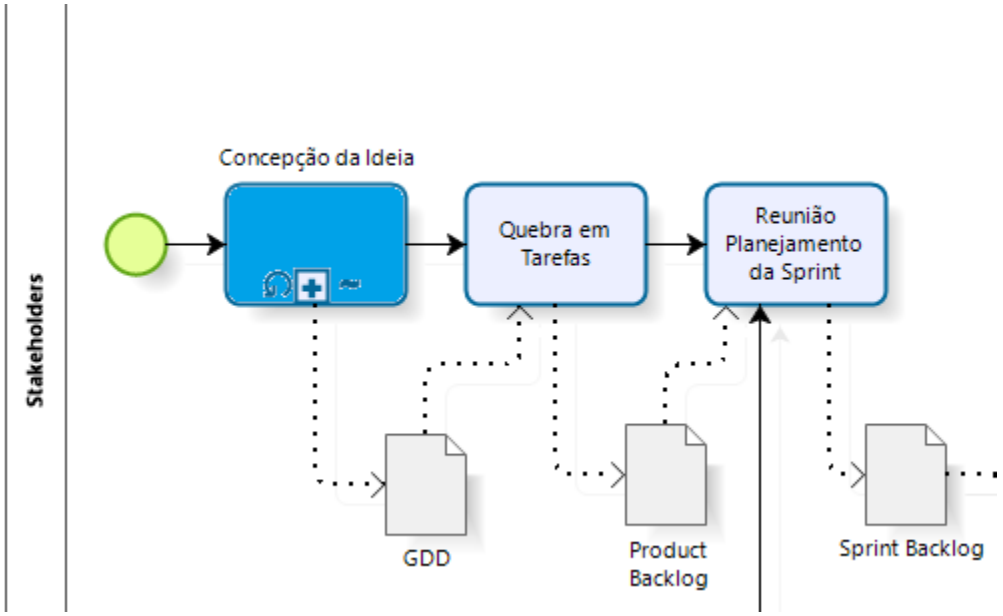
na seção 4.2), por conter etapas bem definidas e se adaptar bem a criação de elementos distintos no âmbito artístico.

Em seguida, foi realizado um estudo da produção artística durante o desenvolvimento de jogos. Em (DIAS, 2018) ele afirma que o designer de jogos envolve tanto a concepção, criação como coordenação do jogo, em seguida ele indica que um profissional dessa área atuaria no planejamento da interface, interatividade, enredo e mecânicas do jogo. Outra fonte sobre a presença da produção artística dentro da construção de um jogo afirmou que após definir o conceito do jogo e sua mecânica básica, o game designer continua realizando modificações, ajustes e alterações no decorrer da produção do jogo (SATO e DE GAMES, 2010). Mediante tal estudo foi possível entender os momentos em que a produção artística se fazia necessária dentro da construção de um jogo, foram eles:

- **Etapa de concepção:** Sem dúvidas essa é uma etapa importante, pois, apesar de não existir ainda o desenvolvimento dos artefatos finais, ela vai gerar uma visão abrangente do que virá a ser o jogo. Nessa etapa são feitos vários esboços iniciais da arte do jogo sobre como poderia ser o herói ou o cenário. Nesta etapa inicial o fator criativo é muito importante, pois essa etapa vai ser o responsável por gerar várias artes, mecânicas e estilos de jogo que podem ser utilizadas no jogo a ser desenvolvido.
- **Etapa de produção:** Esta etapa se refere diretamente a fase de desenvolvimento, onde o jogo já foi definido e seus componentes estão sendo listados para que possam ser produzidos e logo após integrados com suas respectivas mecânicas ou posições dentro do jogo.

A escolha dessas etapas também teve influência da metodologia de *Game Design* proposta por (PEREIRA, 2006), que serviu como fundamento para a extração dos momentos de criação artística e base para alocação das mesmas dentro do processo. Pereira (2006) define que no início da produção existe a ideia inicial seguida do conceito do jogo para após aprovação ser desenvolvido e então testado. Ao estudar a metodologia proposta por ele foi possível notar que ela poderia ser espelhada no *Scrum*. *D*esta forma foi possível incluir a Etapa de Concepção no início do projeto antes do *Product Backlog* como pode ser visto na imagem abaixo que mostra essa etapa em destaque na cor azul.

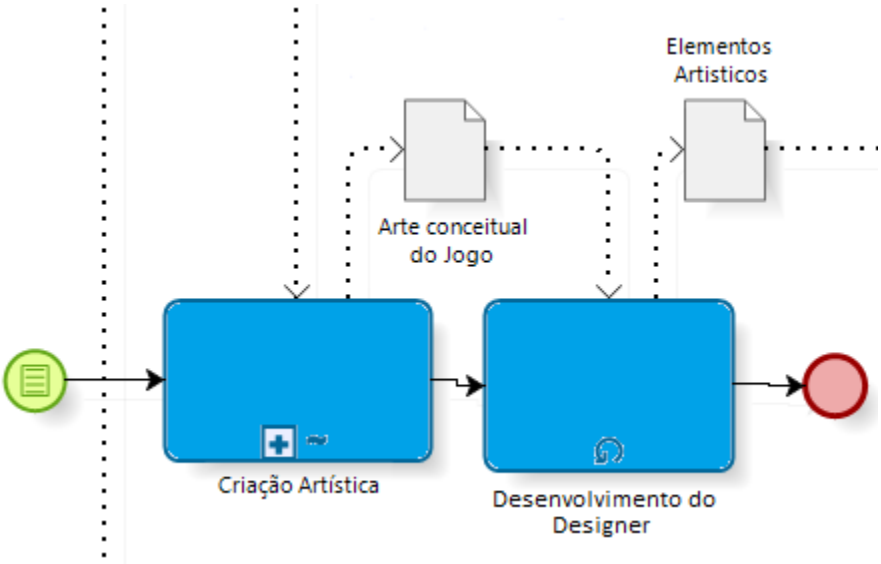
Figura 1 – Subprocesso Concepção da Ideia



Fonte – Print da tela do processo no sistema Bizagi.

Em seguida remodelamos os ciclos para que tivessem a Etapa de Produção, para isso dividimos ela em duas tarefas, uma voltada a gerar novos conceitos e modelos que serão produzidos (Criação Artística) e outra responsável por produzir o produto de arte em si (Desenvolvimento do Designer). Estas etapas podem ser vistas em destaque em azul na imagem abaixo.

Figura 2 – Etapa de Produção



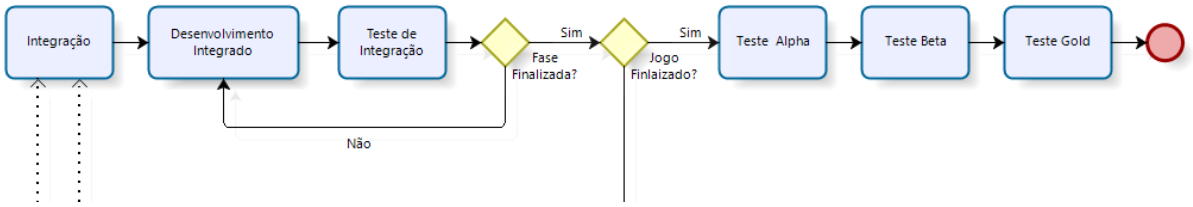
Fonte – Print da tela do processo no sistema Bizagi.

Para a inserção destas etapas artísticas no processo foi definido a criação de um subprocesso Ad-hoc que conteria todas estas fases artísticas, segundo a notação do *Business Process Model and Notation* (MODEL, 2011) as atividades contidas dentro de um subprocesso Ad-hoc não possuem uma ordem de execução definida, logo, não é preciso executar todas as atividades. Este subprocesso foi pensado para ser executado pela equipe de design em paralelo ao desenvolvimento lógico do jogo, neste sentido elas ocorrem após a definição do *Sprint Backlog* do jogo, no qual já possui um GDD contendo a ideia de como o jogo será, uma descrição dos elementos artísticos necessários para o jogo, e com base no *Backlog da Sprint* vai ser produzido no subprocesso as artes conceituais do jogo.

Ao fazer a remodelagem do *Scrum* foi possível notar que a Etapa de Concepção não deveria se restringir apenas ao início do projeto, ela deveria se repetir sempre que necessário, fosse no início do projeto, para elucidar a ideia do jogo, ou mesmo para a geração de novas ideias de arte dentro de um ciclo.

Outras etapas que não estão presentes dentro do modelo tradicional do Scrum (RIS-ALA, 2018) e que foram inclusas para se adequar ao modelo de jogos estão presentes na imagem a seguir:

Figura 3 – Etapas de integração e testes



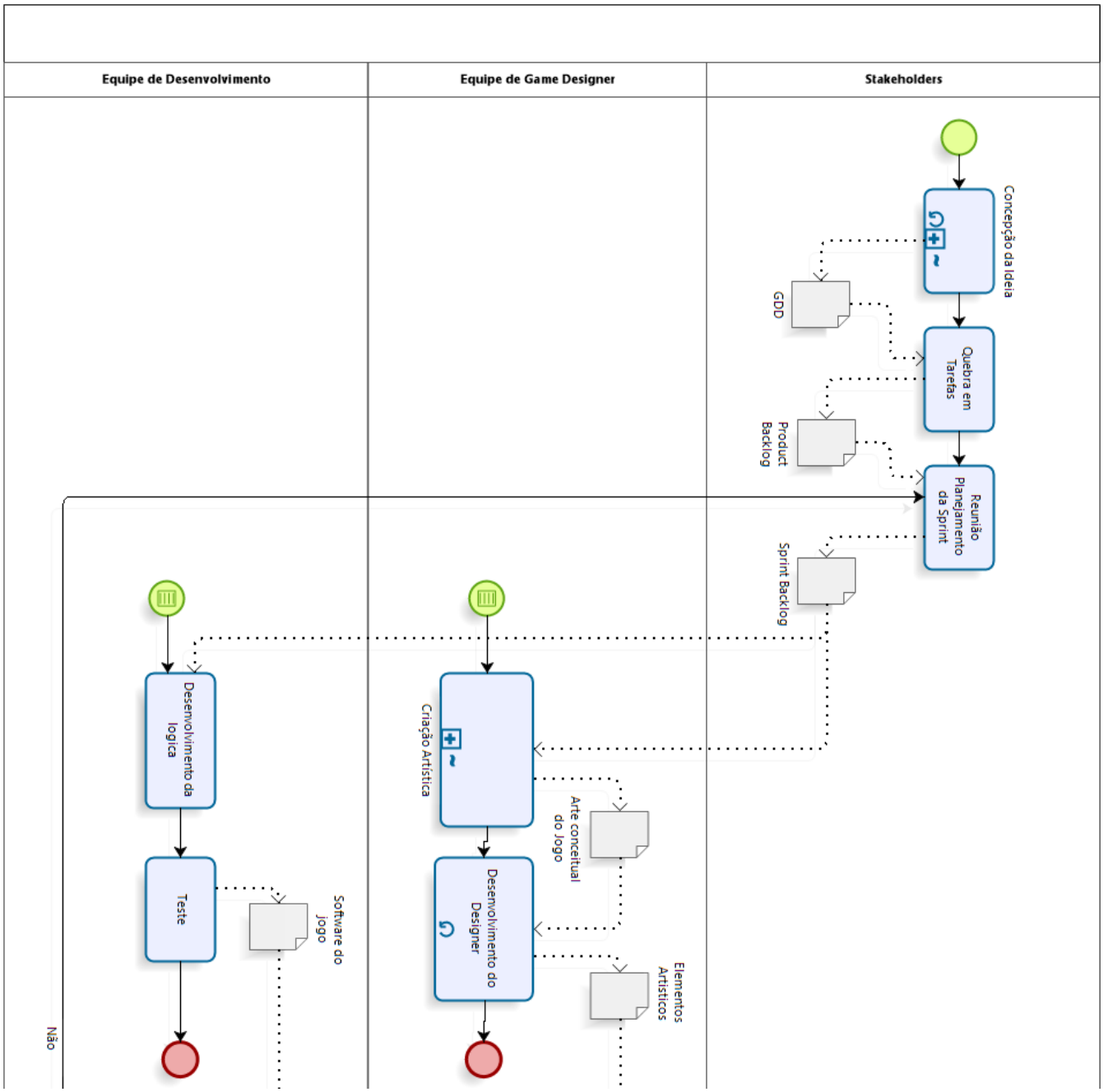
Fonte – Print da tela do processo no sistema Bizagi.

Ao final do desenvolvimento em paralelo do designer e do software do jogo é necessário que aconteça a integração destes elementos para gerar um componente dentro do jogo, para isso incluímos etapas dedicadas a esses momentos, bem como a adição de etapas de testes que auxiliam na busca de erros de jogabilidade, gráficos, level designer antes de entregar o produto final.

Mantivemos como base a notação BPMN para a construção do nosso processo que pode ser visto melhor pelo link presente no ANEXO B.

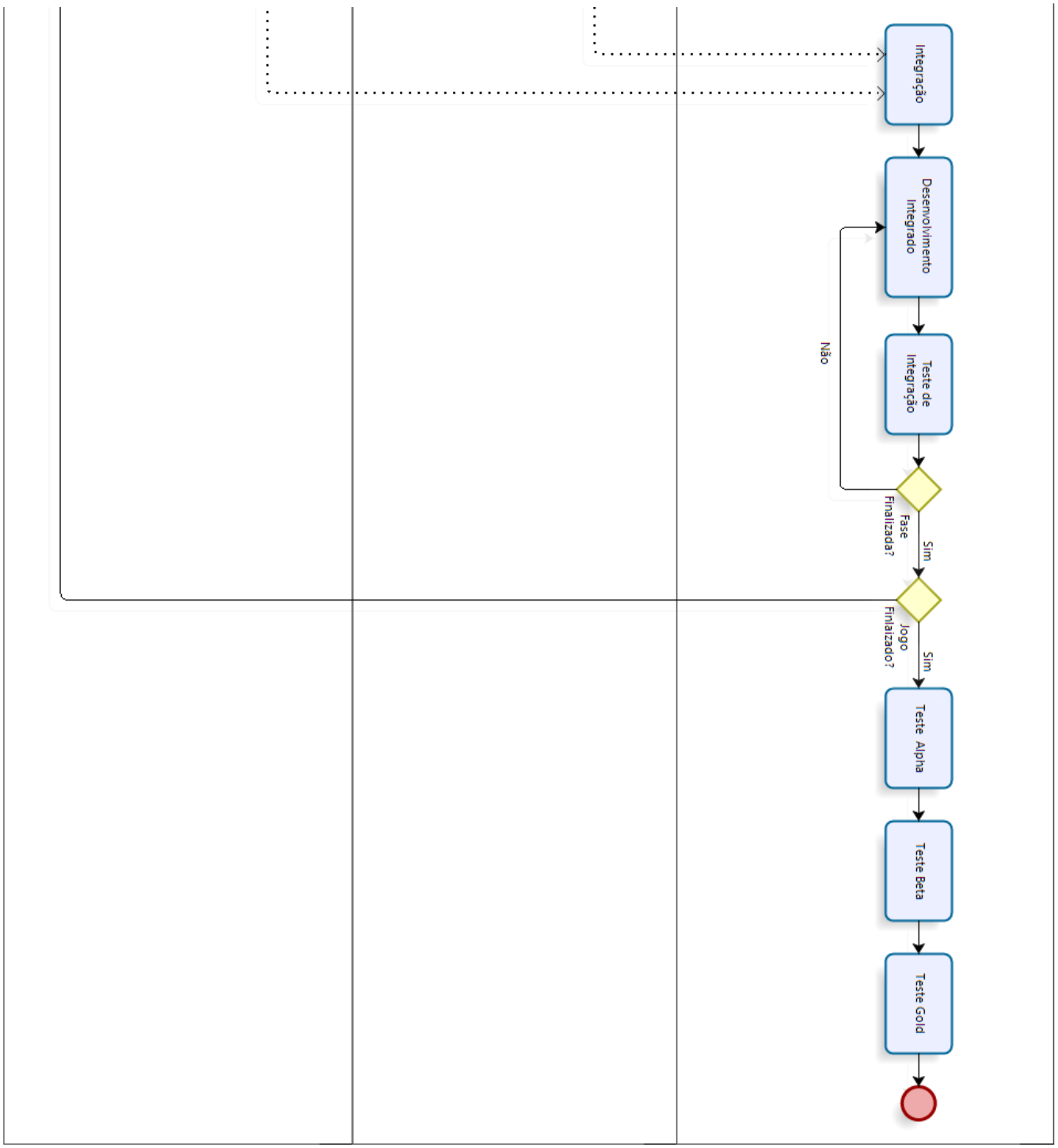
A estrutura do processo ficou distribuída da seguinte forma:

Figura 4 – Processo proposto - parte 1



Fonte – Print da tela do processo no sistema Bizagi.

Figura 5 – Processo proposto - parte 2



Fonte – Print da tela do processo no sistema Bizagi.

- **Concepção da Ideia** - Para que um jogo comece a ser produzido, as ideias precisam ser criadas e aprovadas pelas equipes de desenvolvimento, designer e o supervisor do projeto. O objetivo dessa etapa é definir o que será o jogo. Em paralelo a concepção da ideia, há o desenvolvimento da documentação do projeto, neste caso o *Game design document* (GDD) do jogo, que descreve em detalhes os conceitos e os principais elementos de jogabilidade deste.
- **Quebra em tarefas** - Nesta etapa, a equipe discute para definir uma lista com todas as funcionalidades e elementos desejados para o jogo, usando como referência do GDD. Dependendo do estilo do jogo essa lista vai variar, para um jogo estilo plataforma pode-se definir as fases e cenários que o jogo vai possuir, detalhar os personagens presentes em cada fase, os itens a serem coletados, para um jogo estilo *infinite run* (Corrida Infinita) será definido o cenário onde o jogo se passa, elementos a se coletar, personagens jogáveis. As ideias geradas aqui formarão o *Product Backlog* do jogo, este não precisa está completo no início do projeto.
- **Product Backlog** - É uma lista de funcionalidades desejadas de um produto, ou seja, os requisitos que um cliente espera receber ao final do projeto, descrito com sua própria linguagem. O ponto central do Scrum é a criação do *Product Backlog*, é nele que o projeto inicia.
- **Reunião Planejamento da Sprint** - Nesta etapa, a equipe se reúne com o *Scrum Master* para definir o que será feito na *Sprint*, levando consigo as funcionalidades e elementos já separados na etapa anterior presentes no *Product Backlog* e o tempo estimado que estas irão levar para serem produzidas. As decisões aqui tomadas são documentadas no *Sprint Backlog*.
- **Sprint Backlog** - É uma lista de tarefas que a equipe se compromete a realizar durante uma *Sprint*. Os itens do *Sprint Backlog* são extraídos do *Product Backlog*, pela equipe.
- **Criação Artística** - Esta etapa é um subprocesso Ad-hoc no qual as atividades contidas dentro dele não possuem uma ordem de execução definida, e se repete quantas vezes for necessário. As atividades deste subprocesso são: **Briefing, Identificação do Problema, Expansão das Ideias, Materialização, História e Aparência e Desenho**. A definição destas atividades está presente na seção 4.2. Este subprocesso vai guiar a equipe de *design* na construção dos elementos artísticos do jogo, nem todas as atividades precisam ser executadas para se ter um produto final de *designer*, o intuito do subprocesso é dar suporte a construção artística do que for necessário para o jogo, assim sendo este subprocesso pode ser executando quantas vezes forem necessárias durante todo o processo, seja na concepção ou nos ciclos de desenvolvimento, e termina sempre que gerar um produto de arte para o jogo.
- **Arte Conceitual do Jogo** - Segundo a comunidade artística *Concept Art* é a forma de ilustração na qual o objetivo principal é transportar uma representação visual de um *design*, ideia, e/ou modo para uso em filmes, vídeo games, ou revistas/livros antes do produto final.

- **Desenvolvimento do Designer** - Aqui a equipe de *design* vai produzir as artes finais, sendo elas *Sprites* 2D ou modelos 3D, baseados na Arte Conceitual do Jogo.
- **Desenvolvimento da Lógica** - Esta etapa acontece em paralelo a construção dos elementos artísticos, nela a equipe de software desenvolve as funcionalidades do jogo. A lógica por trás do jogo será feita nesta etapa.
- **Testes** - Estes testes estão ligados a funcionalidade lógica do jogo, se a *bugs* na execução de eventos como: correr, pular, atirar dentre outros.
- **Integração** - Nesta etapa, o conteúdo produzido pela equipe de design e pela equipe de *software*, serão mesclados de forma a juntar cada elemento artístico com a sua funcionalidade lógica dentro do jogo, bem como ligar os componentes aos seus elementos visuais como: texturas para o chão, cenário, céu, dentre outros elementos.
- **Desenvolvimento da Integração** - Esta é uma etapa crucial para o processo. Aqui as equipes de design e *software* se reúnem para fazer as adequações necessárias em ambos os lados, seja alguma funcionalidade que precise ser alterada para se encaixar ao elemento artístico, ou um elemento artístico que não se encaixou bem a funcionalidade e precisa ser alterado. Neste momento, as duas equipes devem estar mais próximas para discutirem o que precisa se adequar para que resulte no produto final.
- **Teste de Integração** - Esta etapa vai executar testes após a junção do design com o software, verificando funcionalidades e sua execução com a elementos artísticos.
- **Testes Alpha** - Primeira versão do seu jogo pronto para o teste. A versão *Alpha*, é feita para ajudar no apontamento dos diversos *bugs* que a equipe não identificou e dar uma amostra das funções ou características de certas funções do jogo.
- **Teste Beta** - *Beta* é a segunda rodada de testes do jogo. As versões betas costumam ser tanto abertas quanto fechadas e usando levantamentos como *feedback* relatados na versão *Alpha*, ela deve estar mais polida e perto da finalização, porém apresentando ainda *bugs* e coisas não finalizadas.
- **Teste Gold** - Esta é a versão final do jogo, neste ponto o jogo já pode ser lançado ao público pois o produto está totalmente acabado e podendo então os jogadores terem total acesso. Embora o jogo esteja totalmente terminado, existe a chance que o jogador possa encontrar *bugs*. Nesses momentos, os desenvolvedores devem liberar *patches* para corrigi-los.

7 EXECUTANDO O PROCESSO

Esta seção descreve como o processo foi aplicado no projeto *BigBang*, mostrando uma visão geral das etapas e os resultados obtidos nos ciclos executados, para este trabalho será

considerado o período de um semestre que estive acompanhando a equipe de desenvolvimento do projeto.

7.1 Projeto BigBang

O jogo *BigBang* foi proposto no contexto do mestrado de ensino de física da Universidade Estadual do Ceará em Quixadá e desenvolvido por alunos de graduação em parceria com a Universidade Federal do Ceará em Quixadá via projeto de extensão. O jogo tem como público-alvo alunos do ensino médio.

7.2 Conceito do Jogo

O jogo visa explicar a evolução do universo até o quinto estágio da teoria do *BigBang*. Esta teoria faz parte do ensino de ciências, mas esta é apresentada de forma muito superficial. Assim, o jogo pretende dar uma visão melhor do conteúdo que normalmente é apresentado aos alunos. O jogo também pretende ser material de apoio ao professor que pretende tornar suas aulas mais lúdicas.

7.3 Os personagens

O jogo tem 3 personagens principais e alguns personagens secundários. Os principais sãos: o herói Alltron, o vilão principal Hod e o general comandante do exército de Hod (e também vilão) Darkon. Os personagens secundários incluem: os minions integrantes do exército do Darkon e o bóson de Higgs. Os personagens principais são fictícios e não correspondem a nenhum elemento existente no universo. Dos personagens secundários, apenas os minions são fictícios, sendo o Bóson de Higgs uma partícula existente na teoria do Big Bang.

7.4 A equipe do Projeto

A equipe do projeto foi formada por quatro colaboradores: uma professora universitária e 3 alunos de graduação. Mesmo se tratando de uma equipe pequena, os participantes se dividiram em dois grupos para atuarem em paralelo na produção artística e no desenvolvimento da lógica, para outras atividades as funções eram atribuídas pela necessidade a um dos integrantes. E claro os clientes: um professor universitário e seu aluno de mestrado.

7.5 Processo na Equipe

Pelo fato de todos os integrantes fazerem parte do contexto da produção de software, foi natural para eles o entendimento e aceitação do processo, mediante seus conhecimentos prévios com processos de desenvolvimento. Entretanto projetos passados foram feitos por esta equipe sem seguir um processo bem definido, o que, segundo eles, se tornou frustrante lavando (junto com outros fatores como continuidade de bolsas) ao abandono de alguns projetos sem sua conclusão. Neste sentido, a ideia de seguir um processo novo mais focado no contexto de produção foi bem recebida por eles. Pelo fato da equipe ser composta por bolsistas, é estipulado que cada pessoa dedique doze horas semanais ao projeto, o que resulta em, no máximo, setenta e duas horas de atividade por sprint. Ficou definido que as sprints teriam duração de três semanas, tendo reuniões semanais como entrega parcial e reunião geral no fim da sprint. Apesar de não ser possível a presença constante do cliente durante o desenvolvimento, ele esteve presente em todas as reuniões de entrega, demonstrando satisfação com o método e os resultados gerados.

7.6 Concepção da Ideia

Está etapa demorou o tempo de uma sprint inteira já que ela é fundamental para todo o restante do projeto. Ela pode demorar menos porque o cliente já tinha parte da história do jogo definida cabendo à equipe as decisões sobre como transformar a história proposta em jogo. Inicialmente foi discutida a temática do jogo, quais os elementos artísticos que podem ser usados, qual a identidade visual, as cores, a mecânica de jogabilidade, qual os detalhes da história por trás do jogo. Para isso foram executadas as subatividades (*Briefing*, Identificação do Problema, Expansão das Ideias, Desenho (Rascunhos)) da etapa de concepção no levantamento das ideias, os resultados obtidos foram:

- O jogo teve uma temática voltada para a criação do universo, seguindo a teoria do BigBang e os princípios físicos ligados a isso.
- Foi dividido em fases, onde cada fase termina em um marco importante no resfriamento do universo.
- Tem como personagens principais: Alltron o herói do jogo, Darkon o general do exército do vilão Hod e Hod, o vilão.
- As fases teriam estilos diferentes de jogabilidade e mecânica.
- Primeiros rascunhos da identidade visual do jogo, a partir da temática, foram feitas algumas formas de como seria o cenário e os personagens.

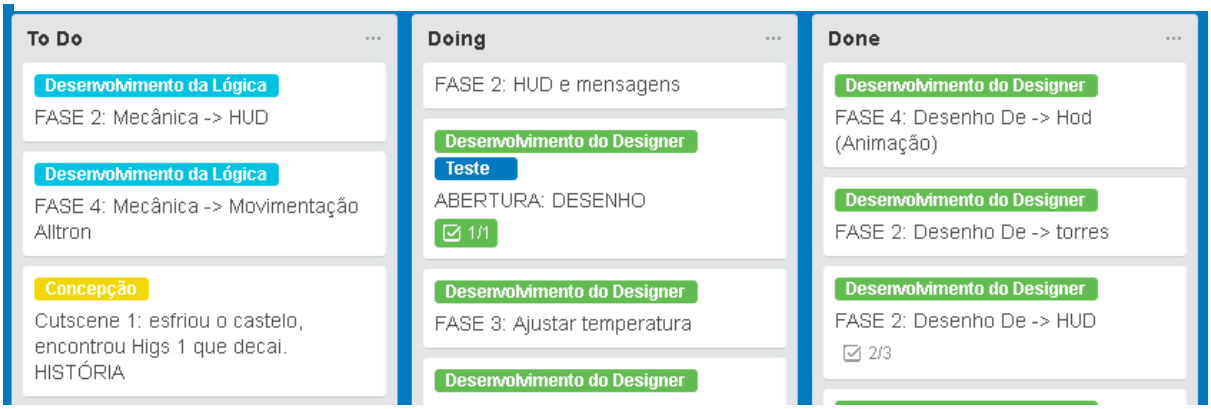
- A história do jogo e os detalhes das explicações técnicas sobre física seriam detalhadas nas *cutscenes* entre as fases.

Estas ideias geraram o GDD (Documento de Game Design), o primeiro artefato e documento base projeto.

7.7 Product Backlog

Mediante ideias bem definidas e visão geral do jogo, foram definidas as funcionalidades e tarefas a serem feitas para a construção do jogo em uma reunião conjunta com o cliente. Esta reunião levantou elementos e atividades para cada fase, foram definidas quatro fases para o jogo. Para facilitar a organização, o Trello foi utilizado como ferramenta de gerência. A criação das colunas o nome e o intuito delas dentro do quadro seguiram a metodologia Kaban (RADIGAN, 2017). Assim, foram criadas as colunas To Do, Doing e Done. Mediante o fato do processo incluir etapas específicas para a construção artística, não houve apenas uma atividade genérica para este intuito como é o comum em processos que só possuem atividades de desenvolvimento de software. Com o novo processo proposto, a equipe elencou atividades específicas para cada elemento de arte necessário para o jogo. Para facilitar a organização das atividades foram criadas tags para referenciar qual a área de atuação da atividade dentro do processo. Estas tags foram: Desenvolvimento do Designer; Desenvolvimento da Lógica; Concepção; Testes. A foto abaixo ilustra como ficou a configuração dos *cards* e colunas no Trello. Na figura é possível perceber que os nomes das atividades descrevem a fase do jogo a que elas estão relacionadas, qual o elemento do jogo e qual fase do processo a atividade pertence. Ex: Fase4: Desenho de -> Hod (animação) indica uma atividade de arte relacionada à animação do personagem Hod na fase 4 do jogo.

Figura 6 – Listas do Projeto no Trello



Fonte – Print da tela do sistema Trello.

7.8 Analise do desenvolvimento

A coleta de dados para essa análise se deu por meio de avaliação informal (FREITAS, 2017) com os participantes, se utilizando de observação direta e troca de opiniões, sugestões e questionamentos nas reuniões.

O processo de maneira geral funcionou de maneira parcial. A principal vantagem do processo foi a inclusão de atividades específicas de artes, permitindo que desenvolvedores e artistas pudessem perceber as dependências entre suas atividades. Outra vantagem é a percepção da separação em relação ao volume de trabalho. Um dos membros tinha habilidade de trabalhar tanto na parte artística quanto na de desenvolvimento e seu tempo foi utilizado para balancear carga entre essas áreas.

A separação de atividades do GDD em atividades do backlog da sprint logo no início do desenvolvimento permitiu ter uma boa dimensão do tamanho do trabalho a ser feito.

Tivemos como pontos positivos a possibilidade de acompanhamento conjunto por todos os participantes, a estrutura do processo deixava claro a separação das atividades o que deixava mais fácil para as reuniões de entrega mostrar o andamento do projeto para os clientes em questão. Saber que existia uma estrutura a ser seguida para gerar um produto final ajudou a equipe a não ficar perdida, em certos momentos de reuniões era possível ver uma dispersão da ideia inicial levava o projeto para outro rumo, algo natural no mundo do desenvolvimento artístico, mas o processo conseguiu manter a equipe focada nesses momentos para que visualizassem o jogo como um produto de software, assim sendo, almejando o resultado final alinhado com o que fora definido. Nos momentos de integração da arte com a mecânica a equipe mostrou facilidade, já que era indicado que fizessem essa tarefa juntos para facilitar a discussão de possíveis mudanças no momento da integração, o que costumou gerar novas ideias que persistiram no componente final.

Entre os principais problemas estão o gerenciamento de tempo e a atualização das atividades. Foi possível perceber que entregas parciais toda semana não estimulavam a produtividade, especialmente devido ao fato de que a equipe tinha tempo de dedicação parcial e dependia de prazos de aulas e provas que eram independentes do processo. A falta de flexibilidade não era compatível com os horários e obrigações acadêmicas que cada um dos participantes possuía. Muitas vezes, as atividades do projeto eram negligenciadas devido a prazos de trabalhos ou provas e outras vezes a equipe precisava fazer bem mais que 12h de atividades para conseguir entregar algo na semana final do prazo. A produtividade de um aluno

de graduação e de membros inexperientes em equipes de desenvolvimento é muito variável, assim, sprints de 3 semanas trazem uma dificuldade maior para o cumprimento de prazos, especialmente quando parte das atividades é feita de forma não presencial.

Foi possível notar a falta de atualização do Trello por parte dos membros. Muitas vezes as atividades presentes no board não representavam seu status atual e eram atualizadas apenas nos dias de reunião presencial com o professor, dificultando o controle. Neste sentido era necessária uma revisão no fim de cada Sprint verificando quais atividades foram feitas ou não, daquelas listadas para a Sprint atual.

Foi possível perceber que existiram falhas tanto na execução do processo quanto na sua definição. Por isso, mudanças foram propostas.

7.9 Mudanças no Processo

Analisando as conclusões, avaliações e reuniões feitas durante a execução do processo, tivemos como resultado mudanças em sua estrutura bem como em alguns aspectos da sua metodologia.

A primeira mudança foi pensar em como seria possível aumentar a flexibilidade do processo, pois foi notório que a estrutura de certa forma empresarial do Scrum não se adequou a realidade acadêmica, onde os horários e dedicação de tempo são diferentes no decorrer da Sprint. Para isso, foi pensado na criação de marcos, com a definição de uma data de entrega para um determinado módulo, dentro deste intervalo de tempo trabalharemos com sprints para que seja possível alocar atividades do backlog em cada uma delas, entretanto o tamanho dessa sprint não é definida, ela pode ser curta ou mais longa a depender da necessidade da equipe, sendo que nestas sprints não existem entregáveis, os entregáveis acontecem apenas nas datas finais de cada marco que define a entrega de um modulo do projeto, assim sendo conseguimos manter o controle de atividades por sprints mas com um intervalo variável de tempo para sua conclusão, pois fica a cargo dos responsáveis pela atividade se auto gerenciar para concluir as atividades no tempo do módulo mas sem ter um tempo máximo para cada atividade, pois como já foi mencionado a dedicação de tempo de cada participante pode ser variada, fazendo com que alguns possam terminar atividades em sprints de uma semana quando outros possam levar duas sprints para terminar.

Outra mudança pertinente foi a possibilidade de criação de outras linhas de divisão para o desenvolvimento, como já mencionando a construção de um jogo necessita de áreas multidisciplinares, neste sentido abrindo espaço para Produção Musical: encarregado de construir a trilha sonora e efeitos sonoros dentro do jogo; Level Designer: responsável por criar

os mapas e fases do jogo; Tester: responsável por testar todos os componentes criados, se certificando que esteja tudo dentro do padrão de jogabilidade; Estes ramos de atuação ou qualquer outro que se veja necessário para a construção de um jogo, poderá ser alocado dentro do processo juntamente com as já existentes áreas de Produção Artísticas e de Programação, modularizando ainda mais o processo e gerando maior especificidade nas funções resultando em um entregável de qualidade.

7.10 Resultados

Ao analisar os resultados da execução do processo, foi possível notar uma falta de engajamento da equipe com o mesmo. Embora o processo tivesse uma metodologia bem formulada e etapas bem definidas, durante sua execução as coisas nem sempre seguiram o modelo proposto. Foi possível notar que muitas vezes as atividades ficavam confusas (desorganizadas) dentro do trello, por exemplo, existiam várias atividades a serem feitas que estariam rodando em paralelo, mas não era possível saber precisamente a qual componente elas pertenciam ou se já tinham começado a acontecer. Assim, não era possível saber por exemplo se a mecânica de movimento do Alltron já estava feita ou se suas Sprites de ataque já estavam em produção. Para isso, era preciso sair verificando todos os *cards* até encontrar os *cards* ligados textualmente ao componente. Para isso minimizar esse problema, foi utilizado este padrão de título "FASE 2: Desenho De -> Alltron" onde são definidos a qual fase e componente esse *card* referenciava, mas isso gerava muitas vezes confusão para o gerenciamento. O ideal é que os *cards* pudessem ser filtrados e agrupados por componente do jogo e/ou por tipo ou fase do jogo.

Com base nesses fatos e como uma forma de sanar esses problemas, a solução proposta foi de construir uma ferramenta que desse suporte a execução do processo. Neste sentido, a ferramenta seria baseada no processo para que pudesse atender as necessidades de organização das tarefas ligadas aos componentes, assim como a distribuição mais clara e fácil das atividades ligadas a programação e arte por componente. A seguir está a descrição da ferramenta Game Control Lab (GCL), desenvolvida neste trabalho.

8 CRIANDO A FERRAMENTA

Inicialmente foi necessário definir o escopo do GCL, a fim de definir quais elementos ela deveria possuir com base na experiência vivenciada na execução do processo, e no processo em si.

A primeira coisa a ser definida sobre a GCL foi que ela deveria ter uma representação dos *Game Objects*. A ideia era representar os elementos que compunham um jogo dentro do GCL, por exemplo, imagine que se deseje criar um jogo estilo *Super Mario*. Esse jogo tem como gênero plataforma, então dentro dele é possível encontrar coisas como pisos flutuantes, quadrados se movendo, o cenário passando ao fundo e, claro, o herói do jogo como personagem principal. Neste caso, cada um desses objetos compõem o jogo como um todo e cada um deles possuem características que se dividem em artísticas e de programação. Tomemos o herói do jogo como exemplo, é possível definir que ele terá uma estrutura humanoide e uma roupa de bombeiro. Também é possível definir que ele consiga correr e pular em todas as direções. Neste caso, a parte visual do personagem que seria o desenho do seu corpo e do seu estilo estaria ligado a produção artística, assim como toda a parte voltada a mecânica de movimentação e habilidade especial estaria ligado à programação. Neste sentido, podemos dizer que um *Game Object* é composto basicamente de arte e programação.

Assim, a representação de *Game Objects* dentro do GCL segue as especificações definidas no processo proposto, pois nele é definido que o desenvolvimento dos elementos de arte e mecânica são feitos em paralelo contendo atividades de integração dos elementos no jogo como resultado final. Assim, as tarefas agora estariam diretamente ligadas a um *Game Object* e seriam do tipo Arte ou Programação. Isso já sanaria um dos problemas encontrados na gestão do processo, que era a dificuldade para saber se determinado componente do jogo já estaria pronto, pois para isso era necessário sair verificando todos os *cards* atrás de encontrar aqueles que tivessem ligação com tal componente. Agora é possível ver se todos os *cards* de um *Game Object* estão terminados indicando que o *Game Object* está terminado também.

Analisando os resultados da execução do processo, também foi possível identificar que, embora o Trello permitisse aplicar filtro, se tornou caótica a visualização das tarefas que estavam próximas da sua data de entrega, quais estavam prestes a atrasar, não era possível ter uma visualização cronológica de como estava o andamento do projeto. Logo chegamos à conclusão que o GCL deveria possuir um sistema de *TimeLine* para auxiliar na gestão das tarefas, facilitando a visualização cronológica, bem como a análise rápida dos estados nos quais se encontram as tarefas por meio de filtros aplicáveis a *TimeLine* do projeto.

O resultado desta etapa de planejamento e concepção do GCL é documento de requisitos que pode ser visto no ANEXO A. Foi também definido que o GCL usaria o Trello

como API para o *Backend*, tendo em vista que seria mais prático remodelar o Trello para atender as nossas necessidades já que ele possui várias funcionalidades que precisaríamos fazer do zero, como por exemplo criar uma tarefa com descrição e data de entrega.

8.1 Validação da Viabilidade

Tendo um escopo bem definido, era necessário validar a viabilidade do GCL, pois até o momento a escolha de sua construção ainda vinha do resultado isolado de uma execução do processo e dos estudos da literatura de processos no desenvolvimento de jogos. Para tal, pessoas relacionadas ao mercado de desenvolvimento de jogos e que trabalham com o desenvolvimento de jogos e possuem o conhecimento prático foram contatadas. O contato com algumas empresas de Fortaleza foi realizado e os entrevistados foram convidados a ouvir brevemente sobre a ideia do projeto e tiveram a possibilidade de saber maiores detalhes sobre o GCL, caso fosse do interesse dela. Como resultado, foram obtidas duas respostas de interesse em participar da validação da viabilidade.

A entrevista ocorreu de maneira aberta, tendo como foco duas perguntas gerais: Você utilizaria o GCL como ferramenta de auxílio no desenvolvimento? Ela teria realmente aplicabilidade num senário real da produção de jogos?

Um dos entrevistados respondeu usando como exemplo a utilização do GCL dentro da empresa a fim de ver como ela poderia se encaixar no cenário da sua empresa. Em um primeiro momento, ele não entendeu muito bem o fator de flexibilidade a qual a ferramenta se propõe, como pode ser visto nesse trecho onde ele diz “parece interessante no geral, mas não sei se é algo que eu usaria na minha empresa geralmente a gente não divide as tarefas associadas a *Game Objects*, a semântica é diferente um artista, por exemplo, só vê/foca nos *cards* de arte”. Mas ao explicar que a aplicabilidade do *Game Object* estaria mais ligada ao fator organizacional e que dentro do GCL existiria uma sessão onde seria possível visualizar apenas os *cards* ligados a si onde o usuário em questão estaria marcado como responsável pela tarefa, ele reavaliou de maneira positiva a abordagem de *Game Objects* dentro da feramente ao dizer “a visão de *Game Object* pode ter essa utilidade gerencial também, dependendo de quem for utilizar. no caso de algum gerente de projeto ou game designer, que precisam ter uma visão mais holística, pode ser útil também”.

Seguindo para o depoimento do segundo entrevistado, ele também se mostrou positivo quanto a abordagem do *Game Object*. Também fez uma indicação importante sobre como demonstrar o progresso dos *Game Objects* neste trecho “Na tela inicial, por exemplo, de olhar para um GO você já saber o quão próximo de ser finalizado ele está”. Ao falar sobre como

ele faz a gestão hoje e fazendo um comparativo com o GCL ele disse “E eu confesso ter achado interessante a ideia da ferramenta. Atualmente, o controle que eu faço usa uma penca de programas gratuitos e é relativamente desorganizado demais para o meu gosto, uma ferramenta para centralizar tudo isso cairia bem. Então sim, eu teria interesse em tal ferramenta”.

Como conclusão do feedback levantado, foi possível validar o GCL com base em sua aplicabilidade em um cenário real, dando assim embasamento para sua construção.

8.2 Desenvolvendo

A partir de um escopo previamente definido e tendo validado a ideia do GCL, iniciou-se o processo de desenvolvimento em si. Devido ao curto período de tempo para o desenvolvimento, o foco da produção foi voltado para as funcionalidades do GCL deixando inicialmente de lado seus fatores estéticos ligados a interface. Neste sentido, os primeiros passos no desenvolvimento foram para fazer a integração com a api do trello. As requisições do trello possuem um padrão de autenticação de aplicações que desejem fazer integração com sua plataforma que solicita o *token* e a *key*. É com base nesse esquema de *token e key* que o usuário pode permitir que o GCL acesse as informações do seu quadro no trello. *O token* é gerado pelo trello para cada usuário que deseja fazer requisições a api, já a *key* é uma chave única da aplicação. Assim, o usuário do GCL precisa obrigatoriamente possuir uma conta dentro do trello para poder usar a ferramenta, já que ao utilizar a api o GCL necessita de um *token* gerando a partir do login e aceitação da aplicação pelo trello.

Após essa etapa, um protótipo do GCL foi criado para testar as requisições a api. Fazendo algumas chamadas, foi possível o nome dos *cards* que existiam no quadro e criar um novo *card*. Integração funcionando, deu-se o início do trabalho de fato em cima dos requisitos do GCL. O primeiro foi a representação do *Game Object* dentro do GCL. Já que o GCL utiliza como backend a estrutura do trello (todos os *cards* são salvos no trello e não em uma base de dados do GCL), não existia muita liberdade para mudar os componentes dos quadros e *cards*. O que foi feito foi remodelar a forma como o GCL trata esses componentes. A solução para a representação do *Game Object* foi usar as listas. Uma lista no trello contém vários *cards* e possui um nome e essa estrutura foi utilizada para criar *cards* atrelados. Para separar entre *cards* de arte e programação dentro da lista, as *tags* do trello foram utilizadas. Assim, foi possível filtrar em uma lista os *cards* de determinada *tag*. Esse foi o mesmo princípio utilizado para acompanhar o estado do *card*, então foram criadas ao todo 5 *tags* e foram elas: feito, desenvolvendo, fazer, programação e arte.

A lógica do GCL girava em torno de requisições ao trello e o tratamento do json de retorno. Mas para melhorar o desempenho, em alguns momentos, um objeto era passado diretamente entre os componentes para evitar um sobrecarga de carregamento na mesma página.

Após todos as funções de criação, edição e leitura estarem funcionais, a UX do GCL foi tratada. Em alguns momentos da edição de um *card*, por exemplo, era possível editar, mas o sistema se mantinha na mesma página, sem nenhum botão de voltar (apenas o do navegador). Com o passar do tempo, através do uso e do teste do GCL, foi possível notar um desconforto com o número de passos necessários para fazer uma ação. Por exemplo, neste caso da edição de um *card* dentro de um *Game Object*, para ver esse *card* era preciso ir para a seção de *Game Objects*, clicar no *Game Object* que possuía aquela tarefa e escolher em qual área ele estava (arte ou programação), para então encontrá-lo e, ao clicar em detalhes desse *card*, havia um redirecionamento para uma nova página. A solução para resolver esse problema foi uma reformulação da navegação e exibição de informações do sistema e atualmente os detalhes do *card* são exibidos na mesma página onde é possível ver o *Game Object*. Botões de retorno nas páginas de edição e criação foram adicionados e levam exatamente para o *Game Object* pai. Os formulários de criação do Projeto e do *Game Object* deixaram de ser uma nova página e se tornaram um modal simples. Existia uma janela ao clicar em um *Game Object* na qual era possível escolher entre arte e programação e que redirecionava para uma nova página e isso tornava custoso para ir criar atividades de arte misturada com atividades de programação, pois o usuário teria que voltar a entrar no *Game Object* para escolher o tipo antes da criação, então adicionamos um menu lateral de navegação à página do *Game Object*. Assim ficou mais cômodo transitar entre as áreas.

Em seguida a *timeline* foi adicionada ao GCL. Optou-se por utilizar uma biblioteca que auxiliasse na construção e manipulação desse componente. A biblioteca escolhida foi a vis.js (B.V., 2018), que possui uma documentação bem detalhada e um código fácil de implementar.

O intuito da *timeline* era exibir todos os *cards* do projeto de maneira que com uma rápida visualização o gerente do projeto e todos os demais integrantes pudessem perceber visualmente o andamento do projeto ao notar os *cards* atrasados, feitos e pendentes. Para isso criamos a seguinte lógica:

- Primeiro todas as datas de entrega dos *cards* foram comparadas com o dia atual;

- Se a data de entrega do *card* estiver inferior ao dia atual e seu status for diferente de feito, quer dizer que a atividade está atrasada e o *card* então recebe um background vermelho;
- Se a data de entrega do *card* estiver superior ao dia atual e seu status for diferente de feito, quer dizer que a atividade ainda está pendente, mas possui tempo para ser entregue então o *card* recebe um background branco;
- Se o *card* não possuir uma data de entrega, então ele aparece no dia atual da *timeline* como um ponto flutuando sem ligação a um dia;
- Se o status do *card* for igual a feito, então ele recebe um background verde.

Após feita a disposição dos *cards* na *timeline* foram criados os filtros. Os filtros servem para identificar mais precisamente um cenário especifico dentro do projeto. Os filtros possíveis são:

- Por status: feito, a fazer, desenvolvendo e todos.
- Por área: Arte, Programação e todos.
- Por responsável: Meus Cards e todos.

Existe intercessão na aplicação dos filtros. Por exemplo, é possível filtras todos os *cards* de programação que estão feitos ou filtrar apenas os Meus Cards que estão a fazer.

O GCL então foi hospeda no firebase. Já que seu *backend* está no servidor do trello e todas as requisições são feitas a ele, o GCL funciona como uma página estática sem precisar de integração com o banco do firebase. Este é o **link de acesso ao GCL** https://pgcl-1995.firebaseapp.com/login. Devido ao fato de que a usabilidade do GCL não foi muito trabalhada, uma página externa com um pequeno **tutorial de primeiros passos** foi criada e se encontra neste link https://sites.google.com/view/gcl-primeiro-acesso/página-inicial, lá é possível ter uma visão inicial do GCL.

8.3 Avaliação de Uso

Terminada a primeira versão do GCL, ele foi colocado em fase inicial de teste a fim de validar a sua aplicabilidade na prática. O foco era avaliar a funcionalidade do GCL já que os fatores de usabilidade ainda precisam ser melhores trabalhado e ficarão para trabalhos futuros. Para isso, pessoas do meio de desenvolvimento indie e desenvolvimento de software foram contatadas

Infelizmente, os dois profissionais ligados ao mercado de jogos consultados na avaliação de viabilidade não estavam disponíveis para fazer a avaliação final do GCL. Entretanto, um engenheiro de software que atua no mercado em Recife se disponibilizou a dar um feedback sobre seu uso.

Para a coleta de informações ele seguiu alguns tópicos aos quais deveria avaliar em sua análise sobre o GCL, foram eles:

- A metodologia de *Game Objects;*
- Usabilidade;
- Sistema de *TimeLine;*
- Pontos positivos e negativos.

Segundo ele, o GCL se mostrou simples e intuitiva de usar, como pode ser visto nesse trecho onde ele diz "A ferramenta se mostrou útil mesmo estando em aprimoramento em relação ao seu desenvolvimento, apresenta de forma simples e intuitiva uma forma de pensar, organizar e controlar as atividades do processo de desenvolvimento de jogos baseando-se em atividades voltadas ao *Game Objects* que farão parte do software (gamer) que será desenvolvido". Quanto à possibilidade de alocar os responsáveis as tarefas criadas ele disse: "A proposta de inicialmente trabalhar na criação de um backlog e atribuir atividades aos membros do projeto se apresenta de forma fundamental para que cada membro possa saber e compreender a importância e acompanhar o progresso do desenvolvimento" mostrando que dentro de uma equipe auto gerenciável todos precisam estar sincronizados com o projeto, ao mesmo tempo que ver apenas suas atividades é uma outra forma de focar apenas nas suas especificações como no caso de um ilustrador que tem um foco maior em apenas criar o que está sendo pedido. Isso mostra que o GCL possui a flexibilidade que ela se propunha nesse aspecto. Sobre a *timeline* do GCL ele considerou ser uma funcionalidade importante para o gerenciamento do projeto, por conseguir visualizar rapidamente as atividades e assim ter um controle maior do andamento do projeto como pode ser visto nesse trecho "Se mostrou relevante a possibilidade de aplicar filtro a *timeline* do projeto e não menos importante, aglutinar os grupos de atividades por *status* de execução, fazendo lembrar o quadro kanban, essa visualização permite manter um melhor controle das tarefas em execução e também as que não estão em execução (desenvolvimento), além de fornecer uma boa visibilidade para o usuário de suas tarefas e responsabilidades". Como forma de análise qualitativa pedimos que ele listasse alguns pontos positivos, negativos e melhorias que julgasse necessário para o GCL, como ponto positivo ele mencionou:

- Boa apresentação das atividades (quadros se mostram de forma organizada e com boa completude de informações).
- Filtros correspondentes a modelos ágeis de controle e gestão de atividades (semelhança ao modelo kanban).
- Boa disposição e organização de atividades direcionados a criação de *Game Objects* (disposição das tarefas de maneira intuitiva para cada object).
- Separação entre projeto de design e projeto de codificação.
- Visualização intuitiva das tarefas em uma *timeline* com opções de filtros de forma dinâmica.

É possível notar sucesso na escolha da metodologia voltada ao *Game Object* e, embora o design do GCL ainda não tenha sido explorado mais a fundo, já é possível concluir que esse trabalho representa um primeiro passo na criação de uma ferramenta mais robusta seguindo o caminho correto quanto a forma de exibição adotados.

Sobre os pontos negativos ele indicou:

- Visualização de respostas do sistema mal localizadas (*pop-ups* de erros ou de ação bem-sucedidas estão mal localizados na tela).

Este é um ponto no qual o GCL precisa melhorar. O sistema de *feedbacks* para o usuário como um todo ainda é muito limitado. Na maioria das vezes, são utilizados redirecionamentos e atualização da própria página sem um retorno de mensagem e com apenas o novo *card* surgindo na tela.

Quanto às melhorias ele listou:

- Fornecer a criação de *tags* ou *label* para posterior filtragem.
- Apresentar a possibilidade de alteração de status diretamente na *timeline*, com função de arrastar (diminuir a quantidade de passos para mudança de status de uma tarefa).
- Atribuir atividade para si com um clique. ("atribua para mim" disponível no rótulo da atividade).

De maneira geral, foi possível notar fatores ligados a usabilidade do GCL. A criação de novas *tags* precisa ser estudada tendo em vista que o trello possui um limite de espaço para *tags* em um quadro. Quanto à mudança de *status* na *timeline*, este foi um ponto discutido no seu desenvolvimento e ficou decidido redirecionar para a página de edição do *card* justamente

para se tornar o mais direto possível. Entretanto a utilização intuitiva de arrastar um *card* dentro da *timeline* aparenta ser mais usual, assim como atribuir uma tarefa para si com um clique. Estes pontos citados com certeza irão entrar na lista de tarefas a serem implementas para a melhoria do GCL, como, por exemplo, a interação com os *cards* na *timeline* que poderia ser resolvido com um modal a partir do clique em um *card*, para não haver redirecionamento e ter uma edição rápida na tela.

9 CONCLUSÃO

Com base nas validações do GCL e na execução do processo levando em consideração as melhorias levantadas para os dois, é possível concluir que obtivemos sucesso na proposta do trabalho, primeiro quanto ao processo que se mostrou útil na construção de um jogo, as mudanças implantadas no Scrum para a inclusão de etapas que atendessem as necessidades no desenvolvimento de um jogo surtiram efeito na prática, auxiliando a equipe em momentos como a concepção diferenciada com uma metodologia mais voltada para o pensamento criativo e do designer, além da separação clara das atividades por áreas de atuação, juntamente com a possibilidade de executar a qualquer momento os subprocessos que auxiliavam na discussão e reavaliação das ideias. O processo conseguiu guiar a equipe rumo a construção de um jogo bem definido na sua etapa de concepção, passando por ciclos de desenvolvimento, caminhando para um produto final de qualidade.

Com os erros da execução do processo, conseguimos então chegar em um modelo melhor com mudanças na sua estrutura, mas para que o processo conseguisse gerar um produto de qualidade concluímos que esta precisava de um suporte, uma ferramenta que auxiliasse na sua execução, eis que está foi desenvolvida, ela se baseou totalmente no processo seguindo sua metodologia, assim sendo um suporte para o processo. Sua aplicabilidade pratica se mostrou um sucesso, possuir um processo bem definido e ter o GCL seguindo esse processo foi a chave para estes resultados, um dos momentos onde foi possível ver este resultado foi na grande aceitação da ideologia de *Game Objects* dentro o GCL, que foi a união do desenvolvimento em paralelo das áreas de produção artística e programação definidos pelo processo juntamente com a modelagem deste dentro do GCL.

Sem dúvidas o GCL ainda precisa de melhoras em vários aspectos de experiência do usuário e usabilidade, como por exemplo poder fazer a transição e mudança de status das tarefas apenas pela *timeline*, melhorar o sistema de feedback para o usuário ao fazer operações dentro da ferramenta, implementar sistema de urgência onde possa indicar que uma tarefa

precisa de prioridade na produção, sistema de atualização por notificação onde os participantes de alguma tarefa de um *Game Object* seriam notificados sobre uma nova tarefa concluída do *Game Object* em questão, espaço para comentários dentro da tarefa para discursões sobre a tarefa em questão, opção de escolha das cores dentro do GCL podendo mudar as cores que caracterizam as atividades de mecânica e designer.

O processo precisa ser executado mais vezes para refinar ainda mais as necessidades na prática e assim chegar em um conjunto que possa ser largamente usado pela comunidade de jogos.

Como trabalhos futuros estamos considerando a integração do GCL com uma ferramenta de construção de GDD, a ideia é ter dentro do *Game Object* uma referência direta a documentação deste Game Object, por exemplo no caso de um *Game Object* Herói do jogo, dentro desse *Game Object* temos os *cards* de arte, mecânica e teremos também um elemento descrição que conterá o texto de concepção do herói do jogo descrevendo como ele foi pensado, como deve ser seu estilo visual, sua mecânica de movimentação ataque, seu contexto histórico dentro do roteiro do jogo e outras informações relevantes que estejam no GDD ligadas a esse *Game Object*.

REFERÊNCIAS

AMBLER, S. W. **Modelagem ágil**: práticas eficazes para a programação extrema e o processo unificado. Porto Alegre: Bookman, 2009.

B.V., Almende. **Vis.js**. Disponível em: <http://visjs.org/>. Acesso em: 13 Maio 2018.

BIZAGI. **Bizagi time to digital**. Disponível em: <https://portal.bizagi.com/>. Acesso em: 16 Out. 2016.

DIAS, RAPHAEL. **Produção de Jogos**. Disponível em: <https://producaodejogos.com/game-designer/>. Acesso em: 30 Jun. 2018.

FREITAS, Luiz Carlos et al. **Avaliação educacional**: caminhando pela contramão. Rio de janeiro: Vozes Limitada, 2017.

HAMADA, O. Considerações sobre o desenvolvimento de jogos por meio do framework scrum. **REFAS-Revista FATEC**, Zona Sul, 2016, p. 1–19.

KEITH, C. **Agile game development with Scrum**. [S.l]: Pearson Education, 2010.

LAUBISCH, A.; E CLUA. **Scrum4games**: uma aplicação do scrum para projetos de games focada em game design. IX Proceedings do SBGames 2010. Florianópolis, 2010. p. 178–187.

MARCELO, Antônio; PESCUITE, Julio. **Design de Jogos**: fundamentos. Rio de Janeiro: Brasport, 2009.

PEREIRA, G. A. **Projeto e desenvolvimento de jogos computacionais**. 2006. Trabalho de Conclusão - Graduação em Ciência da Computação. Joinville: Universidade do Estado de Santa Catarina, 2006.

RADIGAN, Dan. **Kanban**.. Disponível em: <https://br.atlassian.com/agile/kanban>. Acesso em: 22 abr. 2017.

RIS-ALA, Rafael. **Metodologia scrum modelada em BPMN**. Disponível em: <https://media.licdn.com/dms/image/C5612AQH3FJYqK7m_0g/article-inline_image-shrink_1500_2232/0?e=2130105600&v=beta&t=F6CkUsAPh8WJT7BkQU4BKwx8_4fcRulWPVBlVCHGiNM>. Acesso em: 30 Jun. 2018.

SANTOS, R., GÓES, V.; ALMEIDA., L. Metodologia origame: um processo de desenvolvimento de jogos. **XI Simpósio Brasileiro de Jogos e Entretenimento Digital.** Brasılia-DF, 2012.

SATO, Adriana Kei Ohashi. **Game design e prototipagem**: conceitos e aplicações ao longo do processo projetual. Proceedings do SBGames, 2010. p. 74-84.

SCHWABER, Ken. **Guia Scrum**. Disponível em: <https://www.scrum.org/>. Acesso em: 30 Jun. 2018.

SILVA, D. E. d. S., SOUZA, I. T. d., CAMARGO, T. Metodologias ágeis para o desenvolvimento de software: aplicação e o uso da metodologia scrum em contraste ao modelo tradicional de gerenciamento de projetos. **Revista Computação Aplicada-UnG**, 2013. p. 39–46.

VOSS, C., TSIKRIKTSIS, N., FROHLICH, M.. Case research in operations management. **International journal of operations & production management**, 2002, p. 195–219.

WALLAS, Graham. **The art of thought**. [S.l: s.n], 1926.

ANEXO A – DOCUMENTO DE REQUISITOS DA FERRAMENTA

1. INTRODUÇÃO

a. Propósito do Documento

Este documento contém a especificação de requisitos para o sistema GCL (Game Control Lab), que auxiliará na gestão de projetos voltados para a área de jogos.

b. Escopo do Produto

O sistema tem como objetivo auxiliar no gerenciamento de projetos na área de jogos, como: inserir, excluir, modificar, consultar *cards* (tarefas).

c. Visão Geral do documento

Este documento apresenta uma descrição geral da ferramenta, e logo em seguida descreve suas funcionalidades especificando as entradas e saídas para todos os requisitos funcionais. Faz também uma descrição sucinta dos requisitos não funcionais contidos neste sistema.

2. DESCRIÇÃO GERAL

A ferramenta do site GCL gerencia os Cards ligados aos *Game Objects*, onde somente os usuários logados terão permissão para inserir, modificar, excluir e consultar estes *cards*, poderão também adicionar outros integrantes ao projeto para participar das tarefas. Os usuários poderão fazer todas as operações citadas aos *cards*, além de visualizar um na *timeline* do projeto quais estão atrasados, feitos e pendentes.

a. Funções do Produto

Gerenciamento dos *Cards*: inserir, modificar, excluir e consultar os *cards* no projeto.

Gerenciamento de Membros: adicionar e excluir membros ligados ao *card*.

Gerenciamento dos *Game Objects*: inserir e excluir um *Game Object* do projeto.

Gerenciamento da Time Line: Habilitar ou desabilitar filtros para ter várias visualizações diferentes dos *cards* em uma linha temporal.

3. REQUISITOS

a. Requisitos Funcionais

RF.1: Cadastro de Card

Descrição: Todos os participantes do projeto poderão cadastrar um novo *card*.

Entrada: Nome, Descrição, Data de Entrega, Status, Submodulo e Responsável.

Processo: O cadastro será incluído na board do projeto por meio da api.

Saída: Mensagem de confirmação bem-sucedido do cadastro caso tenha sido efetuado com sucesso, senão, mensagem de erro.

RF.2: Modificação do Card

Descrição: O usuário altera os campos que deseja modificar e aplica as mudanças.

Entrada: Campo desejado e novo dado.

Processo: Atualização do *card*.

Saída: redirecionamento para a página do *Game Object* ao qual pertence.

RF.3: Exclusão do Card

Descrição: Os participantes do projeto poderão excluir os *cards*.

Entrada: Seleção do *card*.

Processo: A ferramenta faz uma requisição direta a board passando o ID do *card* para que seja deletado.

Saída: redirecionamento para a página do *Game Object* ao qual pertence.

RF.5: Consulta de Cards

Descrição: Os participantes do projeto poderão visualizar os *cards*.

Entrada: Seleção do *card* na página de meus *cards*, dentro de um *Game Object* ou na *timeline*.

Processo: A ferramenta faz uma requisição direta ao quadro passando o ID do *card* para que seja carregado seus dados.

Saída: redirecionamento para a página de detalhes do *card*.

RF.6: Cadastro de Game Object

Descrição: Os membros do projeto podem criar um novo *Game Object*.

Entrada: Nome.

Processo: O cadastro será incluído no quadro do projeto por meio da api.

Saída: O modal desaparece exibindo a lista de *Game Objects* do projeto.

RF.7: Excluir de Game Object

Descrição: Os membros do projeto podem deletar um *Game Object*.

Entrada: Entrar dentro do *Game Object* e selecionar o botão de exclusão.

Processo: O *Game Object* juntamente com todos os *cards* ligados a ele serão deletados do projeto.

Saída: redirecionamento para a página de *Game Objects* do projeto.

RF.8: Cadastro de Projeto

Descrição: Os membros que utilizam a ferramenta podem criar um novo Projeto.

Entrada: Nome.

Processo: O cadastro será incluído na base do trello por meio da api.

Saída: O modal desaparece exibindo a página inicial do projeto.

b. Requisitos Não-Funcionais

RNF.1: Software

A api utilizada será a do trello e servirá como *backend*, já que ela tem uma boa documentação e seu funcionamento se encaixa em muitos aspectos da ferramenta.

RNF.2: Linguagem de Programação

A ferramenta será feita em Angular 4, para que a ferramenta funcione como uma web app, para isso é indicado uma linguagem que já trabalhe com single pages.

RNF.3: Hardware

A ferramenta ficara hospedada em sum servidor da Firebase oferecido gratuitamente pela google.

ANEXO B – IMAGEM DO PROCESSO

Para melhor visualização acesse o link:
https://drive.google.com/file/d/0B9h2AGA5exr4SnR4Q0RZeGdxWnM/view?usp=sharing