OS INGREDIENTES DE *OVERCOOKED* NO DESENVOLVIMENTO DE UM FRAMEWORK PARA JOGOS SÉRIOS

Reneu Morais Da Silva Júnior
Universidade Franciscana
Santa Maria - RS, Brasil
reneu.morais@gmail.com

Fabrício Tonetto Londero
Universidade Franciscana
Santa Maria - RS, Brasil
fabriciotonettolondero@gmail.com

***Resumo* - O presente trabalho elaborou um framework na *game engine* Unity 3D com o objetivo de auxiliar desenvolvedores que buscam produzir jogos, uma vez que tal atividade possui um alto grau de complexidade e exige tempo e conhecimento técnico. Como objetivo específico, o jogo digital *Overcooked* foi analisado e os elementos que o tornaram um sucesso entre os críticos e público em geral identificados para inserção no framework. Como a Unity 3D é repleta de botões, campos e funções que, por sua vez, podem intimidar usuários menos experientes, a ferramenta foi desenvolvida de forma que propõe ao desenvolvedor uma interface simples e intuitiva. Após a construção do framework, provou-se a eficácia da ferramenta ao desenvolver-se um protótipo funcional com êxito.**

Palavras-chave: Unity 3D, framework, Overcooked, jogos sérios.

1. INTRODUÇÃO

A indústria dos jogos digitais vem apresentando um dos maiores crescimentos entre todos os setores de produção nacional e internacional, segundo o 2º Censo da Indústria Brasileira de Jogos Digitais de 2018 [1], o número de desenvolvedoras passou de 142 para 375, um aumento de 164%. Além disso, o Censo também revelou que, somente em 2017 e 2018, foram produzidos 1.718 jogos no país. Desses, 43% foram desenvolvidos para dispositivos móveis, como celulares, 24% para computadores, 10% para plataformas de realidade virtual e realidade virtual aumentada e 5% para consoles de videogame. Dentro desse universo, foram 874 jogos educativos e 785 voltados para o entretenimento. Diante destes números, é visível o favoritismo dos jogos educativos, tornando o aprendizado muito mais produtivo e recompensador [2].

Em um mundo onde a tecnologia se instalou por toda parte, a juventude contemporânea vem interagindo cada vez mais cedo com smartphones, computadores e videogames. Diante disso, a tecnologia digital se mostra como uma ferramenta cada vez mais importante na produção e transformação cultural tanto na geração atual quanto nas futuras [3]. Entretanto, apesar da crescente adoção dessas ferramentas, a criação de jogos para o ensino ainda vem sendo pouco explorada e mais voltada para o conteúdo em si e carente de diversão [4]. Além do mais, em relação aos jogos educativos nacionais, não é incomum encontrar estudos com relato de jogos que são desenvolvidos de forma *ad-hoc*, ou seja, que não seguem processos formais de desenvolvimento de software [5]. O desenvolvimento *ad-hoc* consiste em desenvolver um software sem utilizar processos formais, não utilizar processos definidos, não gerar documentação, não executar fases previstas no desenvolvimento de software e não planejar e executar testes apropriados, o que pode custar mais caro posteriormente e gerar insatisfação do cliente que contratou o serviço [5]. Uma outra consequência da falta da utilização de processos definidos é a geração de sistemas não amigáveis para uso, ou seja, que foram feitos às pressas e que apresentam informações consideradas não fidedignas e com erros de funcionamento. Infelizmente, isso faz com que um número reduzido de usuários se interessem em adquirir ou utilizar o sistema desenvolvido, fazendo com que o projeto fracasse e investimento realizado seja desperdiçado [5].

O desenvolvimento de jogos *ad-hoc* vem rapidamente se tornando uma realidade cada vez mais presente tendo em vista a expansão dos jogos eletrônicos independentes (comumente referidos como *indie games* ou jogos indie) e o desenvolvimento de motores de jogos gratuitos e acessíveis, como a Unity 3D e Unreal Engine. Essa forma empírica de desenvolver jogos, apesar de cada vez mais comum, pode comprometer o resultado final do jogo, visto que esse processo informal pula etapas e uma sequência lógica e já bem estabelecida de desenvolvimento. Desta forma, um framework voltado para o desenvolvimento de jogos (educacionais e/ou voltados para o entretenimento) poderá auxiliar futuros profissionais e amantes de jogos digitais.

1.1 Objetivos

Diante do exposto, o objetivo do presente estudo foi produzir um framework que englobe as características do jogo digital *Overcooked* e que facilite o desenvolvimento de jogos educacionais e/ou voltados para o entretenimento. O framework emprega características do jogo citado anteriormente, sendo elas:

- Objetivos claros que desafiam o jogador sem frustrá-lo.
- Controles simples e acessíveis que permitem que qualquer pessoa consiga jogar.
- Partidas que não demandam muito tempo e ou sobrecarregam o jogador.
- Um código de programação que permitirá a reutilização do mesmo em jogos com objetivos e escopos diferentes.

A utilização de frameworks no desenvolvimento de jogos educacionais é um comportamento corriqueiro e que vem ganhando espaço no desenvolvimento de jogos independentes e educacionais visto que economiza recursos e tempo [2][7][14]. O framework proposto foi criado no motor de jogos Unity 3D e aplicado no desenvolvimento de um jogo educacional a fim de provar a sua aplicação em jogos com fins específicos.

2. FRAMEWORK

Um framework é um ponto comum que une códigos comuns entre projetos distintos de software e fornece uma funcionalidade genérica [4]. Essa ferramenta é fundamental no desenvolvimento de jogos de computador e tem como principal objetivo executar um número de tarefas de baixo nível e que são necessárias na maioria dos jogos [37]. Com o crescente uso da engenharia de software no desenvolvimento de jogos digitais, os recursos utilizados no desenvolvimento dos jogos digitais têm sido majoritariamente em frameworks de desenvolvimento, que também são conhecidos como motores de jogos, ou *game engines* [38]. A proposta principal desses motores é permitir que os recursos comuns a quase todos os jogos sejam reutilizados para cada novo jogo e software criado. Desta forma, é implementado a cada novo jogo apenas os seus requisitos particulares, fazendo com que o desenvolvedor ganhe tempo e foque mais nos seus objetivos específicos.

3. JOGOS EDUCATIVOS

Um jogo é uma atividade lúdica onde os integrantes da atividade participam de uma situação de engajamento social num tempo e espaço determinado, com características próprias

delimitadas pelas próprias regras de participação na situação "imaginária" [7]. Jogos educativos são atividades lúdicas que oferecem oportunidades de continuar trabalhando e reforçando o conteúdo de aula através da participação mais efetiva, buscando sempre um melhor desempenho do aluno [6]. Diante de uma sociedade cada vez mais digitalizada e conectada, os jogos educativos digitais proporcionam aos professores métodos dinamizados para estimular e engajar os estudantes. Hoje, essa ferramenta vem ganhando cada vez mais espaço em todas as castas do ensino nacional [13], e o seu desenvolvimento pode auxiliar futuros alunos pesquisadores. A importância dos jogos educativos no ensino fundamental, por exemplo, vai muito além aprendizado e engajamento dos alunos, como também apresenta contribuições no desenvolvimento infantil que abrange aspectos cognitivos, afetivos, físico-motores e morais [13]. Num contexto educacional superior, não é incomum o uso de jogos educativos no ensino de, entre outras áreas, computação ou programação [12][14].

No entanto, essa ferramenta também vem ganhando impulso em contextos não educacionais. O sucesso de uma iniciativa empreendedora, por exemplo, depende de realçar abordagens que propiciem a inovação e, uma delas é o *Design Thinking* [15]. Segundo Tim Brown [16], a vantagem do *Design Thinking* é que ele ressalta o ser humano antes do produto ou serviço propriamente dito, onde a empatia conduz o empreendedor a pensar em soluções dirigidas às pessoas. Neste contexto, um grupo de pesquisadores da Universidade Federal do Rio Grande do Sul (UFRGS) desenvolveu um jogo para os profissionais da área do ensino superior e do empreendedorismo que une o lúdico ao ensino da abordagem *Design Thinking* [15]. Design thinking é uma abordagem prática-criativa que tem o objetivo de resolver problemas em diversas áreas empresariais, principalmente no desenvolvimento de produtos e serviços, e age com base na coletividade colaborativa do desenvolvimento dos projetos [15]. Essa abordagem consiste em quatro etapas: a imersão, ideação, prototipagem e desenvolvimento [15]. Os resultados deste estudo mostraram que o jogo foi bem aceito pelos participantes e que ensinou o conteúdo relacionado à *Design Thinking* de forma eficiente, fazendo com que os jogadores inclusive desejassem aprender desta forma em detrimento às outras.

Alguns outros exemplos de jogos sérios em contextos profissionais e acadêmicos são InViVo [31] e Scratch [32]. InViVo "Surgical Simulation Game" foi desenvolvido para os profissionais da área da saúde e consiste em simulações que empregam casos reais recriados como pacientes virtuais interativos para permitir que os médicos identifiquem uma abordagem prática para a identificação precoce de sintomas e a aplicação do gerenciamento baseado em evidências (melhores práticas) e diretrizes baseadas em evidências [31]. As simulações são usadas pelos profissionais de saúde e alunos de medicina para avaliá-los sobre desempenho em casos antigos ou casos potenciais, além de permitir que as decisões clínicas tomadas no cenário sejam desconstruídas e analisadas, levando a melhores resultados para os pacientes, com mortalidade e responsabilidade reduzidas [31]. Em outro espectro, Scratch é um software de desenvolvimento orientado a objetos e utilizado para criar e desenvolver jogos de forma simples [32]. O Scratch é usado em mais de 150 países e está disponível em mais de 40 línguas, além de ser utilizado por algumas universidades do mundo todo para ensinar a programação e desenvolvimento de jogos, sendo uma delas, inclusive, a própria Harvard University [32][33].

Diante dos contextos expostos acima, o presente artigo visa, na próxima seção, apresentar e analisar o jogo em estudo, Overcooked, para posteriormente identificar os elementos que o compõe. Esses elementos serão posteriormente utilizados em um esboço de framework que será voltado para o desenvolvimento de jogos sérios com aplicação educacional.

4. OVERCOOKED

Overcooked é um jogo digital desenvolvido em 2016 pela Ghost Town Games para ser jogado com um a quatro jogadores. O objetivo do jogo é coletar o maior número possível de moedas até do final da partida. As moedas são adquiridas de acordo com o número de pedidos entregues. Ao finalizar o tempo, os jogadores são então classificados em um sistema de 3 estrelas com base no número de moedas adquiridas. Salienta-se que o aspecto de cozinhar é dificultado pelos *layouts* da cozinha, que alteram gradativamente e em maior complexidade conforme o passar dos níveis. Cada cenário possui estações para adquirir ingredientes, áreas de preparação como fogões e fornos, janelas para entregar pedidos e louças que geralmente são separadas no cenário, exigindo tempo para se mover entre elas. No decorrer da partida, os jogadores são obrigados a assumir papéis diferentes dentro da cozinha para alcançarem pontuações mais altas, podendo um jogador ser responsável por cortar os ingredientes, um por lavar a louça, etc.

Do mesmo modo, também podem haver (ou não) outros obstáculos ou desafios, tais como a cozinha estar separada por uma faixa de pedestres, com os pedestres potencialmente ficando no caminho do jogador e consequentemente atrapalhando o deslocamento do fluxo do mesmo. Em outro cenário, a cozinha fica em dois caminhões se deslocando em ritmos diferentes por uma estrada, o que muitas vezes impossibilita (ou atrasa) a passagem de uma metade da cozinha para a outra.

Overcooked utiliza uma abordagem intuitiva e exige o uso de apenas dois botões, o que o tornou um sucesso entre os críticos e amantes de jogos. Graças a essa abordagem incauta, o Overcooked vem sendo empregado em estudos de diversas áreas, tais como na comunicação e ensino de línguas estrangeiras [8], comportamento do desperdício de alimentos [9], linguagens figurativas em revistas de críticas de jogos [10], desenvolvimento de jogos em realidade virtual [11], dentre outras.

4.1. As principais características do overcooked

Como qualquer jogo clássico, Overcooked possui um limite de tempo, pontuação e colaboração entre participantes. À medida em que a tensão aumenta no decorrer do jogo, mais obstáculos ficam no caminho do jogador, desde fogo a ratos. Sendo assim, apenas uma equipe comunicativa obtém êxito.

Overcooked possui visão *top-down* (com visão superior, com a câmera bem no alto apontando para o cenário), em perspectiva fornecendo uma visão geral do jogo. Cada fase detém um limite de tempo preestabelecido (2 a 4 minutos) e se resume a preparar os pratos solicitados, sendo que cada pedido possui um limite de tempo próprio e a sua elaboração requer ingredientes distintos (Figura 1).

Figura 1 – Pontuação do jogador (1), pratos solicitados (2) e tempo de jogo restante (3). Imagem: *Overcooked*

A movimentação dos chefs se dá nos eixos X e Y. Há dois botões principais no jogo: um para pegar/soltar os objetos do jogo e um para realizar ações (cortar/preparar um ingrediente, lavar louça, etc.). O jogador também conta com um pequeno *dash* (uma espécie de arranque momentâneo, aumentando sua velocidade temporariamente) para ajudá-lo a se deslocar rapidamente de um lado da cozinha até o outro. Isto posto, mesmo esbanjando simplicidade, a proposta de cada cenário somado aos pedidos que

se acumulam e os relógios de serviço que começam a se esgotar, o caos se instala, que é o ápice do jogo e um dos pilares do seu game design.

Apesar da possibilidade de se jogar sozinho, a grande proposta do jogo é a cooperação, como em qualquer cozinha comercial. Neste escopo, o jogo incentiva e recompensa o trabalho em equipe ao apresentar desafios temporais e físicos que, por sua vez, são mais facilmente vencidos conforme o andamento do jogo e a sinergia e comunicação entre jogadores melhora. Essa melhora comunicativa e sinérgica está ligada a teoria do *game flow* (estado de fluxo), que é o estado psicológico de imersão completa, quando o jogo equilibra poderes e desafios mantendo a atenção do jogador [24].

Nesse quesito, o estado de fluxo é o canal que o jogador se encontra entre a ansiedade e tédio. O jogador pode permanecer no fluxo se ele acreditar que as suas habilidades estão à altura do desafio em questão. Se as habilidades de um jogador melhorarem mais rapidamente do que o jogo pode se adaptar e apresentar novos desafios, o jogador cairá no tédio. Se novos desafios sobrecarregam a habilidade em desenvolvimento, o jogador é dominado pela ansiedade e sai do fluxo [24]. Como cada nível propõe novos cenários e desafios diferentes, o jogador se encontra mais facilmente dentro deste canal psicológico.

Diante disso, fica evidente o porquê do sucesso do Overcooked: objetivos claros e desafiadores, controles simples e acessíveis e partidas que não demandam muito tempo e nem sobrecarregam o jogador.

4.2. Estilo de arte

A arte nos jogos vem ganhando um destaque especial, visto que influencia a percepção, sensação e experiência geral que o jogador recebe visualmente do sistema de jogo [17]. Overcooked emprega um estilo cartoon 3D de poucos polígonos *(low poly)* com visuais coloridos e uma interface de usuário limpa, intuitiva e layouts de ícones.

O potencial visual de qualquer jogo é essencial para munir a experiência desejada com jogabilidade interessante e significativa [18]. Se o estilo de arte de um jogo obtiver êxito em chamar as atenções dos jogadores, ele também irá capturar as suas emoções, proporcionará emoção, um sentimento de conexão ou até mesmo transmitirá uma atmosfera memorável [18].

Essa linha de pensamento contribui diretamente com o que é chamado de "sensação de jogo" (*game feel*) [19], que é composto por uma série de elementos que levam o jogador ao nível de imersão, envolvimento total e fluxo durante a sua experiência lúdica. O *game feel* não acrescenta nada no jogo em termos de mecânicas e regras e sim na experiência e sensação, sendo algo que empodera o jogador, fazendo com que ele sinta que está em controle do universo no qual está inserido. Especificamente, *game feel* pode ser considerado o senso tátil e cinestésico de manipular um objeto virtual. É a sensação de controle em um jogo. Um bom *game feel* é aquele que torna um jogo fácil de aprender, mas difícil de dominar [25].

Game feel também pode ser considerado uma "interação momento a momento", que é uma das razões pelas quais um conjunto simples de instruções sobre como criar deliberadamente uma sensação de jogo eficaz é ilusório [26]. Há também a linha de pensamento de que *game feel* é aquele que é "intuitivo" e que encoraja o *game flow* citado anteriormente - isto é, que os controles de um jogo devem parecer o menos mediado e o mais invisível possível [26], detalhe este que fica explícito na simplicidade dos controles do jogo em estudo.

No caso de Overcooked, onde cada cenário (nível) possui uma temática distinta, é importante que o contexto no qual o jogador está inserido esteja a par desses conceitos, afinal, um dos personagens principais do jogo não é só aquele que está sob o controle do jogador, mas também os cenários em si.

4.3. Cenários

O objetivo de um cenário não é que ele deve ser obrigatoriamente bonito, encantador ou impressionante, ele pode ser totalmente o oposto e ainda assim atingir um alto nível estético [17]. Sendo assim, a sua importância não se encontra no seu aspecto fisionômico e sim na construção da experiência de jogo. No caso de Overcooked onde a proposta é instaurar um ambiente de bagunça e comunicação entre os jogadores, os cenários demandam muito mais do que um visual elaborado.

Os cenários de jogos definem o clima e contêm elementos visuais que aumentam a atmosfera, proporcionam uma sensação de realismo e geralmente fazem o mundo do jogo parecer vivo [18], que fica nítido em Overcooked com seus cenários que variam entre ambientes árticos, espaciais e navios piratas (Figura 2).

Diferentemente de inúmeros jogos clássicos e contemporâneos, os desenvolvedores de Overcooked criaram personagens genéricos que não diferem entre um e outro senão no quesito visual. Essa decisão por parte dos desenvolvedores é justamente o que dá mais importância aos cenários, uma vez que cada fase possui mecânicas, layouts e estéticas distintas (Figura 2).

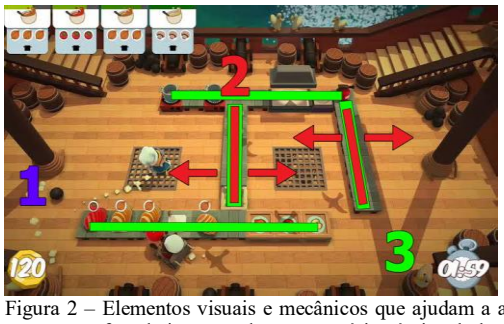

Figura 2 – Elementos visuais e mecânicos que ajudam a aumentar a imersão e atmosfera do jogo, sendo neste cenário náutico: bolas de canhão (1), balcões que se movem conforme o navio balança, layout do cenário (3). Imagem: *Overcooked.*

Diante do exposto, é importante que um jogo exiba um mundo detalhado, fornecendo um contexto que consiste na identidade do personagem (neste caso, os cenários), no objetivo, nas possibilidades, e em todo o necessário para experimentar a mecânica do sistema, sendo este o principal motivo do jogo existir [21].

4.4. Mecânicas

Sabe-se da importância do estilo de arte e cenários bem arquitetados, as mecânicas do jogo também o são. As mecânicas são itens essenciais no game design e necessárias para a implementação do mundo virtual. Sem mecânicas, não existe jogo, tendo em vista que elas são responsáveis por gerar desafios para os jogadores resolverem e determinar os efeitos das suas ações. Dessa forma, é responsabilidade do game designer construir mecânicas que gerem uma experiência divertida, desafiadora e principalmente balanceada [27]. Há inúmeras definições para mecânicas de jogos, pode-se dizer que a mecânica do jogo é um sistema/simulação baseado em regras que facilitam e incentivam o usuário a explorar e aprender as propriedades de seu espaço através do uso de mecanismos de *feedback* [28]. A mecânica também pode ser definida como as várias ações, comportamentos e mecanismos de controle oferecidos ao jogador em um contexto de jogo [29].

Isto posto, o jogo em estudo pode ser considerado um jogo assimétrico, sem simetria forçada em cada personagem, pois ele utiliza os itens do próprio cenário no design dos cenários (e.g. *layouts* estranhos, pratos sujos, fogos aleatórios, balcões em constante deslocação, etc.). Essas características forçam os jogadores a trabalharem juntos e se adaptarem constantemente.

Existem cinco tipos diferentes de mecânicas e o termo mecânica passou a indicar muitos tipos diferentes de relacionamentos subjacentes entre entidades nos jogos [30]:

- **Físico:** A mecânica mais comum é definida pela física - a ciência do movimento e da força. No caso de Overcooked, isso acontece desde o chão deslizante até os pedestres que empurram os jogadores.

- **Economia interna:** A mecânica das transações envolvendo elementos do jogo que são coletados, consumidos e negociados. A economia interna de um jogo normalmente engloba itens facilmente identificados como recursos: pontos, dinheiro, energia, munição, saúde, poder mágico, etc.

- **Mecanismos de progressão:** Em muitos jogos, o design de níveis determina como um jogador pode se mover pelo mundo. Tradicionalmente, o jogador precisa chegar a um lugar específico para completar o nível. Nesse tipo de jogo, o progresso do jogador é rigidamente controlado por vários mecanismos que bloqueiam ou desbloqueiam o acesso a determinadas áreas.

- **Manobras táticas:** Os jogos podem ter mecanismos que lidam com a colocação de unidades/objetos no cenário para vantagens ofensivas ou defensivas. Tratando-se de Overcooked, isso ocorre em posicionar ingredientes/pratos estrategicamente pela cozinha, além do posicionamento no cenário.

- **Interação social:** A maioria dos jogos não obrigavam a interação social entre jogadores até recentemente. Hoje, muitos jogos online incluem mecânicas que recompensam a interação e cooperação entre jogadores. Trocar recursos e mecânicas que incentivam a cooperação ou o conflito entre jogadores são exemplos aplicáveis e presentes em Overcooked.

Diante do exposto, é importante que as mecânicas centrais do jogo em estudo sejam identificadas para posteriormente serem utilizadas em um framework voltado para o desenvolvimento de jogos. Neste caso, há cinco mecânicas fundamentais para o funcionamento de Overcooked:

- **Mecânicas do jogador**: movimentar-se pelo cenário, pegar/soltar objetos, fazer alguma ação com os objetos;
- **Mecânicas do cenário**: Gerar um pedido (para o jogador produzir), produzir ingredientes para tal pedido;

Ao analisar essas mecânicas dentro do jogo, é possível constatar que na sua forma crua elas podem não passar de animações mascarando algumas simples linhas de código, o que possivelmente tornará a sua reutilização em jogos diferentes algo prático e que consome menos tempo (Figura 3).

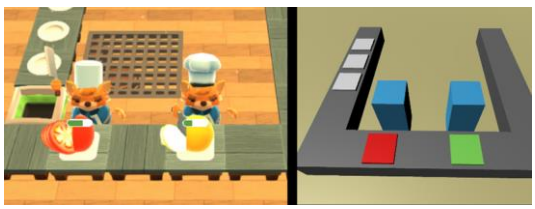

Figura 3 (Overcooked –esq., figura criada pelo autor –dir.) – Animação dos jogadores cortando os ingredientes (esquerda) e os mesmos visto sem as texturas e animações (direita).

5. METODOLOGIA

O presente trabalho utilizou uma abordagem explicativa para descrever as características do jogo e desenvolveu um framework que facilita o desenvolvimento de jogos educacionais e/ou sérios. Um framework é um projeto abstrato orientado a objetos que pode ser adaptado segundo as necessidades da aplicação [34]. Sabendo das dificuldades enfrentadas pelos desenvolvedores de jogos, tais como a falta de acesso ao ensino formal, falta de conhecimento, gerenciamento de tempo, entre outros, o presente framework foi desenvolvido.

O desenvolvimento se deu com a metodologia XP, que é uma abordagem leve para times de desenvolvimento de software de tamanho pequeno a médio e que desenvolve softwares com requisitos vagos e que se modificam rapidamente [35]. A metodologia XP é método sistêmico que se inicia por um plano geral, ou seja, compreender o objetivo como um todo e sem muitos detalhes para, posteriormente, repartir os objetivos em partes, para iniciar o seu desenvolvimento [35]. Segundo o autor Mike Cohn [36], essa abordagem é uma mudança no paradigma tradicional e tem como base quatro valores: comunicação, simplicidade, feedback e coragem. A comunicação pode ser ou não o motivo de sucesso em quase qualquer projeto, inclusive nos projetos de software. Neste contexto, a comunicação é um dos valores mais importantes dessa metodologia [36].

O segundo valor de manter a simplicidade é difícil e outro princípio do XP. Assim, o ideal é que se faça a coisa mais simples que pode funcionar e, mesmo assim, continua sendo um dos princípios mais difíceis de se alcançar, afinal, muitas vezes é necessário fazer algo complexo pensando no futuro [36]. O feedback não é só necessário, como também um dos princípios que fazem com que o XP tenha sucesso, pois é de extrema importância o feedback do cliente visto que os ciclos são curtos e a integração contínua [36].

O último valor, que é a coragem, é necessário, pois dispensar um código já bem concretizado, iniciar do zero e realizar mudanças drásticas requer coragem por parte do desenvolvedor [36]. Neste contexto, o emprego desta metodologia facilitou o desenvolvimento bem-sucedido do framework tendo em vista o grau elevado de conhecimento técnico necessário e tempo disponível.

6. DESENVOLVIMENTO

O motor de jogos Unity 3D foi utilizado para desenvolver o framework almejado. Unity 3D é um motor de jogos desenvolvido pela Unity Technologies e tem como finalidade o desenvolvimento de jogos e aplicativos para mais de 25 plataformas. Essa ferramenta gratuita pode ser utilizada na criação de jogos em realidade virtual, tridimensional (3D), bidimensional (2D), realidade aumentada, simulações e softwares com outros fins [39][40]. Além disto, a Unity 3D vem sendo adotada pelas mais variadas indústrias, como no desenvolvimento de filmes, automotivo, arquitetura, engenharia e construção [40].

O desenvolvimento do projeto iniciou com a programação das mecânicas do jogador anteriormente identificadas. A programação foi realizada no software Microsoft Visual Studio 2019, que é um aplicativo de desenvolvimento integrado da Microsoft, utilizado para desenvolver programas de computador, sites, aplicativos e serviços da web e aplicativos móveis. Um ambiente provisório em 3D foi gerado e nele criado um cubo. O cubo gerado recebeu alguns componentes necessários para a sua posterior utilização como jogador, sendo esses componentes um *box collider* (para o objeto detectar colisões), *character controller* (componente que facilita a movimentação de um objeto em 2D ou 3D), *rigidbody3D* (permite a manipulação da massa, rotação,

gravidade, etc.) e um código de programação (para programar o input do jogador, entre outras funções) (Figura 4).

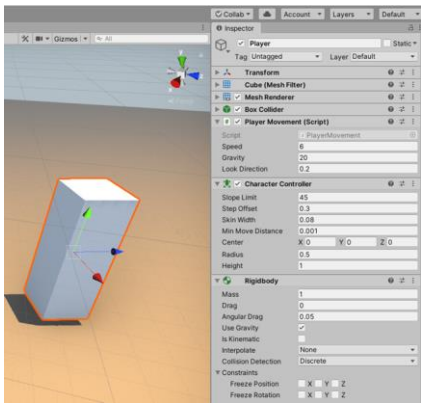

Figura 4 – O cubo gerado e os seus respectivos componentes necessários para a movimentação do jogador. Imagem disponibilizada pelo autor.

A movimentação e ações básicas do jogador, que ocorrem pelos comandos do teclado e/ou controle, foram programadas no componente "PlayerMovement", que é um código desenvolvido pelo autor na linguagem de programação C#.

A próxima mecânica do jogador a ser desenvolvida é a de pegar e soltar objetos. Um outro cubo, dessa vez invisível, foi acrescentado um pouco à frente do jogador. Esse cubo recebeu apenas dois componentes: um *box collider* e um código de programação. A programação dessa mecânica foi feita de modo que caso alguns objetos específicos entrassem dentro daquele raio e o jogador apertasse a tecla espaço, aquele objeto ficaria preso na frente do jogador até que o mesmo apertasse a tecla espaço novamente, simulando a mecânica de pegar e soltar objetos.

A última mecânica do jogador e, talvez uma das mais importantes, foi elaborada de forma semelhante. Utilizando o mesmo *box collider* anterior, outro código de programação foi gerado com o objetivo de detectar objetos específicos dentro do seu raio. A programação desse objeto específico seria o local onde o jogador pode exercer alguma ação. Em Overcooked, só é possível cortar e preparar ingredientes em cima de objetos pré-estabelecidos (fogão, tábua de cortar, etc.), portanto, essa mecânica é necessária para o andamento do jogo. A programação desta mecânica foi feita de modo que o jogador só possa fazer a ação ali caso esteja dentro do seu raio de alcance e com algum objeto em cima. Ao atender esses requisitos, o jogador segura a tecla B por 3 segundos para então realizar a ação. Frisa-se que a tecla B foi escolhida devido a sua proximidade com a tecla espaço, entretanto, o projeto possibilita ao desenvolvedor a escolha de qualquer tecla. Ao segurar a tecla B por três segundos, optou-se por mudar o objeto de cor para visualmente notificar o jogador da ação bem-sucedida.

Com as três mecânicas do jogador desenvolvidas, implementadas e testadas com êxito, direcionou-se, então, as atenções para a implementação das mecânicas de cenário.

Após a implementação desses espaços de instanciação e blocos para delimitação do local de jogo, direcionou-se o foco do estudo para desenvolver-se um sistema de pedidos e local de entrega. Para isso, foi necessário elaborar um sistema flexível que possibilita o desenvolvedor a criar pedidos e escolher os seus ingredientes base, desta forma:

- Objeto base **A** + objeto base **B** = objeto resultado **C**

Nesse contexto, os objetos base são adquiridos em seus respectivos locais de criação, levados ao local de ação e após a sua execução originam o objeto resultado. Ao chegar a esse desfecho, elaborou-se uma interface de fácil manuseio e intuitiva cuja finalidade é gerar, de forma aleatória, pedidos preestabelecidos pelo desenvolvedor. Na Figura 5, é possível observar a interface desenvolvida: os objetos solicitados (letras em vermelho), os seus respectivos requisitos (letras em verde), e as barras superiores são o tempo necessário de cada objeto solicitado. Esse tempo é determinado pelo desenvolvedor

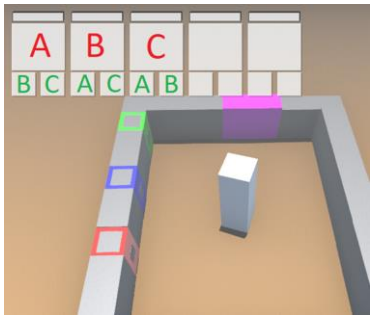

Figure 5 – Interface desenvolvida com o pedido, o seu tempo e ingredientes necessários. Imagem disponibilizada pelo autor.

Esse sistema de pedidos é gerenciado por um código principal que contém o tempo de jogo, pontuação e quais pedidos e objetos base estão sendo solicitados (Figura 6). Ao receber um determinado pedido, o jogador deverá juntar os respectivos ingredientes base no local de execução, realizar a ação e posteriormente levar o objeto resultado no local de entrega. Cada pedido possui um tempo próprio de entrega, portanto, cabe ao jogador decidir qual pedido deverá ser preparado e entregue primeiro. Esse tempo de pedido é imposto de forma aleatória, tendo um tempo mínimo e máximo de entrega que é estabelecido pelo desenvolvedor. Ao entregar o objeto no local de entrega, o código principal varre todos os pedidos solicitados na interface para conferir se há correspondência entre objeto solicitado e entregue. Em caso afirmativo, o jogador recebe pontos e o pedido correto é desligado na interface. Caso contrário, o jogador perde pontos e o pedido permanece ativo na interface até que acabe o seu tempo ou o jogador faça a entrega correta.

Figura 6 - O código principal cuja função é designar novos pedidos, verificar os objetos entregues, configurar o tempo de partida e pontuação do jogador. Imagem disponibilizada pelo autor.

Para agilizar o desenvolvimento da programação desejada, todos os objetos do trabalho, por padrão, são cubos. Entretanto, o que diferencia um cubo do outro é o seu respectivo material. No Unity 3D, materiais são a forma como a superfície de um objeto é projetado na tela, ou seja, eles não alteram o formato do objeto. Sendo assim, cada objeto distinto recebeu uma cor específica, o que facilitou a diferenciação dos objetos entre si e no cenário. Todos os objetos possuem materiais distintos, sendo nesse caso as cores diferentes, para melhor diferenciá-los entre si. Os cubos menores são os objetos base (criados pelos cubos instanciadores), que podem ser pegos/soltos pelo jogador. Após colocar as

combinações corretas na caixa de ação e executar o comando, o objeto resultado é criado (nesse caso, o cubo com material laranja) (Figura 7).

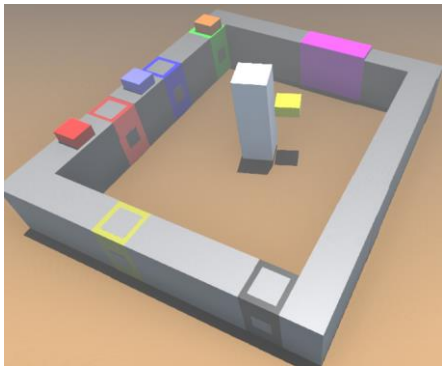

Figura 7 - Versão final do protótipo. Imagem disponibilizada pelo autor.

Ao integrar todas as mecânicas descritas e testá-las no cenário provisório, percebeu-se que o protótipo era condizente com as características do Overcooked e se comportava de forma responsiva e satisfatória. Tal resultado sinalizou que a programação ora desenvolvida poderia ser organizada de forma que posteriormente pudesse ser facilmente reutilizada por futuros desenvolvedores. Para isso, desenvolveu-se um pequeno sistema que integra discretamente a interface do Unity 3D, na forma de janela, a programação realizada anteriormente (Figuras 8). O framework desenvolvido no presente trabalho recebeu a nomenclatura "RMFramework", sendo as letras RM as siglas do nome do autor.

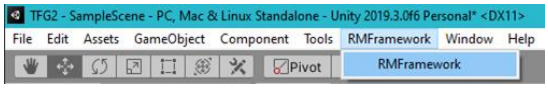

Figura 12 – Ao instalar a presente ferramenta no projeto, o desenvolvedor contará com uma nova opção na sua interface do Unity (RMFramework). Imagem disponibilizada pelo autor.

O framework contém 9 botões que, ao clicá-los, possibilitam ao desenvolvedor a instanciação dos objetos descritos ao longo da seção de desenvolvimento. Para fins de praticidade e melhor entendimento do usuário, cada botão na nova lista possui um *tooltip* que descreve brevemente a função daquele objeto selecionado (Figura 9). Um *tooltip* é uma moldura flutuante que aparece quando se passa o mouse sobre um elemento da interface (pode ser uma palavra em um texto, um botão, link etc.) e que contém uma explicação adicional sobre aquele elemento [41].

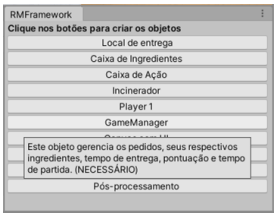

Figura 9 – O framework instalado no Unity 3D. Cada botão possui uma breve descrição para melhor entendimento. Imagem disponibilizada pelo autor.

Como é possível observar na imagem acima, os *tooltips* também notificam quando o objeto é ou não necessário para o funcionamento do jogo. Ao instanciar os objetos desejados, o usuário deve, então, verificar o inspetor de cada um deles a fim de averiguar se algum destes necessita ser atrelado a algum objeto dentro do Unity (Figura 10) O inspetor é uma janela na interface que contém todas as informações e componentes de um objeto selecionado no Unity 3D.

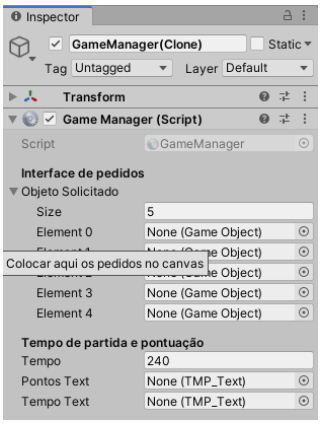

Figura 10 - Inspetor do novo objeto instanciado pelo desenvolvedor utilizando o presente framework. Imagem disponibilizada pelo autor.

Com o sistema desenvolvido e organizado, o eixo do estudo se moveu para colocar em prática e testar a eficácia do framework elaborado.

7. Testes e Resultados

Sabe-se que o desenvolvimento de jogos é uma tarefa muitas vezes árdua. É exigido que o desenvolvedor tenha conhecimento em diversas áreas para além da programação em si, tais como, arte, sonografia, narrativa, entre outros [42]. Diante desse contexto, buscou-se desenvolver um jogo pequeno utilizando a ferramenta produzida. Visto que já se possuía um protótipo funcional, o próximo passo era conceber uma breve narrativa para situar o jogador naquele contexto e justificar o grande objetivo do jogo.

Atentando para o sistema de misturas e pedidos desenvolvido anteriormente e o cenário provisório que já gozava de múltiplas cores, decidiu-se que o jogo teria como finalidade ensinar as cores e suas respectivas combinações aos usuários. Apesar de o público alvo ser crianças a partir de 6 anos de idade, pessoas de todos os estágios de vida poderão desfrutar dessa experiência digital. Diante desse contexto, optou-se por situar o jogo em um comércio de tintas onde clientes solicitam ao jogador uma determinada cor de tinta. Ao receber esse pedido, o jogador terá que, então, misturar outras cores a fim de obter aquela solicitada pelo cliente. Essa esquematização encaixa-se sublimemente no sistema já desenvolvido e permite que os jogadores aprendam a produzir determinadas cores a partir das cores primárias, o que dá ao jogo um aspecto educacional discreto.

Ao delimitar o escopo da experiência almejada, mudou-se o foco do estudo para então encontrar meios de concretizar essa concepção de forma visual. Sabe-se do caráter multidisciplinar da indústria de jogos, portanto, buscou-se alunos do curso de Jogos Digitais da Universidade Franciscana para a implementação gráfica pretendida. Três alunos optaram por participar do desenvolvimento. Um aluno se encarregou de modelar (em 3D) um personagem e fazer a sua animação digital, um aluno se encarregou de modelar (em 3D) o cenário do jogo e uma aluna se encarregou de desenhar a arte da interface do jogo.

Ao concluírem-se os novos elementos visuais do jogo, os mesmos foram implementados no projeto e o foco do trabalho deslocou-se para a última etapa de testes e ajustes necessários. Posto que a programação do jogo mostrou-se responsiva e comportando-se de forma previsível, essa etapa final do desenvolvimento também objetivou produzir a sonoplastia do projeto e acrescentar mecânicas de feedback, tais como partículas de efeito. O autor optou por adquirir a trilha sonora e efeitos

sonoros, de forma gratuita, em páginas online de sonoplastia para jogos.

Com os novos componentes visuais e sonoros instalados (Figura 11), voltaram-se as atenções, novamente, para a etapa de testes, também conhecida como *playtesting*. *Playtesting* é uma abordagem que consiste em analisar o funcionamento de um determinado jogo em relação as suas características inerentes e analisa elementos como jogabilidade, grau de diversão e dificuldade, entre outros [44]. Esses testes são essenciais para alcançar um nível satisfatório de qualidade e asseguram o funcionamento correto do protótipo em relação aos objetivos e características projetadas no início do desenvolvimento [44], além de ser a etapa mais importante da metodologia XP.

Optou-se por realizar o *playtesting* de forma independente de modo que um novo estudo possa ser realizado posteriormente com o público alvo. Contudo, frisa-se que o objetivo do presente estudo foi desenvolver um jogo com o framework produzido, fato esse que se concretizou com êxito. A realização do *playtesting* pelo autor foi um fator altamente positivo e eficiente. Conforme o andamento dos testes, foi possível realizar prontamente inúmeros ajustes conforme as necessidades se apresentaram, sendo os exemplos mais comuns ajustes no tempo de entrega dos pedidos, velocidade de movimentação do jogador e a detecção de colisão dos objetos. O fato de o autor ter desenvolvido o protótipo seguramente encurtou o tempo entre encontrar irregularidades no jogo e a aplicação dos ajustes necessários. Em contrapartida, sabe-se da extrema importância que os *playtesters* (aqueles que aplicam o *playtesting*) agregam ao desenvolvimento de jogos dado a relevância do seu feedback crítico [44], uma vez que não participaram diretamente da produção do jogo e, portanto, não possuem o mesmo viés dos desenvolvedores.

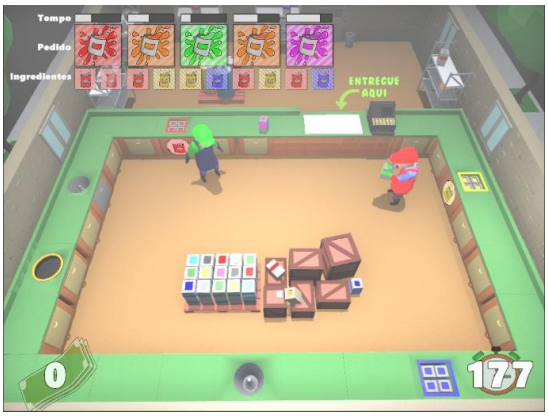

Figura 11 – Estado final do protótipo desenvolvido com o framework.

8. CONCLUSÃO

Desenvolver jogos é uma tarefa penosa, porém muito recompensadora. Essa arte exige uma gama de conhecimento técnico nas mais variadas áreas de conhecimento, desde a programação, arte, sonografia, narrativa, entre outros [42]. É sabido que o desenvolvimento de um jogo, por mais simples e menor que seja o seu escopo, pode custar ao(s) desenvolvedor(es) dezenas ou até centenas de horas de dedicação. Também não é incomum projetos que não vingam e acabam postergado ou até mesmo abandonado. Diante desse contexto, buscou-se criar um framework para auxiliar desenvolvedores iniciantes e experientes. A fim de evidenciar a eficácia da ferramenta produzida, um jogo protótipo foi desenvolvido, testado e finalizado com êxito.

Um framework padrão para produzir jogos com as características do Overcooked foi desenvolvido com sucesso. Esse framework poderá ser utilizado por desenvolvedores de jogos, iniciantes ou não, para criar jogos educativos ou de entretenimento de maneira satisfatória. Além desse framework servir como um ponto de partida para aqueles que desejam ingressar no mundo de desenvolvimento de jogos, os desenvolvedores com experiência na área poderão utilizar e alterar o framework de acordo com as suas necessidades especificas. Seja quem adotar este processo, os agentes que o utilizarem serão beneficiados, pois poderão reutilizar o que for desenvolvido em vários projetos, que por sua vez permitirá uma evolução e melhoria contínua do processo.

9. REFERÊNCIAS

[1] L. Sakuda e I. Fortim (Orgs.). II Censo da Indústria Brasileira de Jogos Digitais. Ministério da Cultura: Brasília, 2018.

[2] N. Fernandes. Uso de jogos educacionais no processo de ensino e de aprendizagem, 2010.

[3] C. Riva. Novos tempos, novas crianças. 2009. http://www.dihoje.com.br/dihoje2009/?pg=noticia&id=1360. Acesso em 23 nov. 2019.

[4] K. Siqueira, E. Nascimento e C. Portela. Uma Proposta de Processo Padrão para Apoiar o Desenvolvimento de Jogos Educativos. SBC – Proceedings of SBGames 2018 — ISSN: 2179-2259.

[5] P. Battistella. ENgAGED: Um Processo de Desenvolvimento de Jogos para Ensino em Computação. Tese de Doutorado do Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Santa Catarina. UFSC, 2016.

[6] W. M. Parreira Júnior e R. V. Farias. O Aprendizado Através De Jogos Educativos. SIED – Simpósio Internacional De Educação A Distancia, 2016.

[7] T. Leal. Jogos: alternativas didáticas para brincar alfabetizando (ou alfabetizar brincando?). Alfabetização: apropriação do sistema de escrita alfabética. Belo Horizonte: Autêntica, 2005.

[8] A. Roach e Y. Utami. Using Video Game To Enhance English Communication Skills. Proceedings of the Fifth International Seminar on English Language and Teaching (ISELT-5), 2017.

[9] C. Verloop. Developing a serious game as a tool for collecting data on food waste behavior. Creative Technology, University of Twente, 2018. Acessado em 02/10/2019. Disponível em https://essay.utwente.nl/75846/1/Verloop_BA_EWI.pdf.

[10] P. Merkl. Figurative Language in Online Game Reviews, Bachelor thesis of the University of Pardubice - Faculty of Arts and Philosophy, 2019.

[11] D. Tang, J. Tran e S. Briggs. Ben-tomo! A Cooperative VR Cooking Game (2018). Acessado em 02/10/2019. Disponível em https://digitalcommons.wpi.edu/mqp-all/6599.

[12] N. Carneiro1, A. Machado, C. Laureano, R. Cavalcante e W. Viana. Net.Aura: Design e Aplicacao de um Jogo de Realidade Aumentada no Ensino de Redes de Computadores. SBC – Proceedings of SBGames 2018 — ISSN: 2179-2259.

[13] D. Batista e C. Dias. O Processo De Ensino E De Aprendizagem Através Dos Jogos Educativos No Ensino Fundamental. Encontro de Ensino, Pesquisa e Extensão, Presidente Prudente, 2012.

[14] R. Monclar, M. Silva e G. Xexéo. Jogos com Propósito para o Ensino de Programação. SBC – Proceedings of SBGames 2018 — ISSN: 2179-2259.

[15] F. Lorenzi, V. Ribeiro e G. Kurtz. RPG Educacional para o ensino de Design Thinking. SBC – Proceedings of SBGames 2018 — ISSN: 2179-2259.

[16] T. Brown. Design Thinking. Uma Metodologia Poderosa Para Decretar o Fim das Velhas Ideias. Rio de Janeiro: Alta Books, 2017.

[17] E. Alejandro e P. Domínguez, Art Direction In Games. Acessado em 09/10/2019. Disponível em pancredad.com.

[18] T. Meigs. Ultimate Game Design. Building Game Worlds. USA: Mc Graw Hill/Osborne, 2003.

[19] S. Swink. Game Feel. A game designer's game to virtual sensation. USA: Morgan Kaufmann Publishers, 2009.

[20] S. Hayward. Cinema Studies. Key Concepts. USA: Routledge, 1996.

[21] N. Egentfeldt, S. Smith e P. Tosca. Understanding Video Games. NY: Routledge Taylor & Francis Group, 2009.

[22] P. Duncan. Game Design Deep Dive: Building truly cooperative play in Overcooked. *Gamasutra*, 2016.

[23] K. Chris. Road to the IGF: Ghost Town Games' Overcooked. *Gamasutra*, 2017.

[24] L. J. Crockett. HTML5 Canvas, User Illusions, and Game Flow. Augsburg College, Minneapolis MN, USA, 2014.

[25] S. Swink. Game Feel: A Game Designer's Guide To Virtual Sensation. CRC Press Taylor & Francis Group, ISBN 13: 978-1-138-40325-3, 2009.

[26] J. Marcotte. Queering Control(lers) Through Reflective Game Design Practices. The international journal of computer game research, volume 18 issue 3, ISSN:1604-7982, 2018.

[27] A. Ernest. Fundamentals of Game Design. Second Edition. Berkeley, CA: Peachpit Press/New Riders, 2009

[28] D. Cook. What are game mechanics? lostgarden.com, disponível em http://lostgarden.com/2006/10/what-are-gamemechanics.html, 2006.

[29] R. Hunicke, M. LeBlanc e R. Zubek. MDA: A Formal Approach to Game Design and Game Research. http://www.cs.northwestern.edu/~hunicke/MDA.pdf, 2004.

[30] A. Ernest. Game Mechanics Advanced Game Design. First Edition. Berkeley, CA: New Riders Games, 2012

[31] I. Gorbanev, S. Agudelo-Londono, RA Gonzalez, 2018. A systematic review ofserious games in medical education: quality of evidence and ped-agogical strategy. Med Educ Online 23:1438718.

[32] Malan, D. ,Leitner, H., "Scratch for Budding Computer Scientists", in Proc. of the 38th SIGCSE Technical Symposium on Computer Science Education, pp. 223-227, Covington, KY, 2007.

[33] U. Wolz, H. H. Leitner, D. J. Malan, and J. Maloney. Starting with Scratch in CS1. ACM SIGCSE Bulletin, 41(1):2–3, Mar. 2009.

[34] C. Souza. Um framework para editores de diagramas cooperativos baseados em anotações. 1998. 105f. Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação, Campinas, SP. Disponível em: http://www.repositorio.unicamp.br/handle/REPOSIP/275954.

[35] K. Beck. Programação extrema explicada: acolha as mudanças. Porto Alegre: Bookman, 2010.

[36] M. Cohn. Desenvolvimento de Software com Scrum: Aplicando métodos ágeis com sucesso. Porto Alegre:Bookman, 2012.

[37] C. Pessoa, wGEM: Um Framework de Desenvolvimento de Jogos para Dispositivos Móveis, Dissertação de Mestrado, Pernambuco, Novembro, 2001.

[38] B. Feijó, M. Dreux, G. Ramalho, A. Battaiola. Desenvolvimento de Jogos em Computadores e Celulares. Florianópolis, SIBGRAPI 2001.

[39] S. Axon. Unity at 10: For better—or worse—game development has never been easier. Ars Technica. Setembro, 2018. Disponível em https://arstechnica.com/gaming/2016/09/unity-at-10-for-better-or-worse-game-development-has-never-been-easier/.

[40] D. Takahashi. John Riccitiello Q&A: How Unity CEO views Epic's Fortnite success. VentureBeat. September, 2018. Disponível em https://venturebeat.com/2018/09/15/john-riccitiello-interview-how-unity-ceo-views-epics-fortnite-success/.

[41] P. Christensson. "Tooltip Definition." TechTerms. 2006. Accessado em 16/05/2020. Disponível em https://techterms.com/definition/tooltip/.

[42] W. Coelho, L. Bacelar, J. Neto, A. Jacob. A Maldição de Bibi Costa: desenvolvimento de um jogo voltado para a cultura amazônica. SBC – Proceedings of SBGames 2015 | ISSN: 2179-2259.

[43] E. Zimmerman, N. Fortugno. Soapbox: Learning to Play to Learn - Lessons in Educational Game Design. 2005. Accessado em 20/05/2020. Disponível em: https://www.gamasutra.com/view/feature/130686/soapbox_learning_to_play_to_learn_.php

[44] J. Choi, J. Forlizzi, M. Christel, R. Moeller, M. Bates, and J. Hammer. Playtesting with a Purpose. In Proceedings of the 2016 annual symposium on computer-human interaction in play. ACM, 254–265. 2016.