

Os ingredientes de *overcooked* no desenvolvimento de um framework para jogos sérios

Overcooked ingredients in the development of a framework for serious games

Reneu Moraes da SILVA JÚNIOR¹
Cássio Fernandes LEMOS²
Fabrício Tonetto LONDERO³

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.

Abstract

The present work elaborated a framework in the game engine Unity 3D with the objective of assisting developers who seek to produce games, since such activity has a high degree of complexity and requires time and technical knowledge. As a specific objective, the digital game *Overcooked* was analyzed and the elements that made it a success among critics and the general public identified for insertion in the framework. As Unity 3D is full of buttons, fields and functions that, in turn, can intimidate less experienced users, the tool was developed in a way that offers the developer a simple and intuitive interface. After the construction of the framework, the effectiveness of the tool was proven by successfully developing a functional prototype.

Keywords: Unity 3D. Framework. *Overcooked*. Serious Games.

¹ Graduado em Jogos Digitais pela Universidade Franciscana (UFN/RS).
E-mail: reneu.morais@gmail.com

² Mestre em Artes Visuais pela Universidade Federal de Santa Maria (UFSM/RS). Professor do curso de Jogos Digitais da Universidade Franciscana (UFN/RS). E-mail: cassio.f.lemos@gmail.com

³ Mestre em Ciência da Computação pela Universidade Federal de Santa Maria (UFSM/RS). Professor do curso de Jogos Digitais da Universidade Franciscana (UFN/RS).
E-mail: fabriciotonettolondero@gmail.com

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 (SAKUDA; FORTIM, 2018), 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 (FERNANDES, 2010).

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 (RIVA, 2009). 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 (SIQUEIRA; NASCIMENTO; PORTELA, 2018). 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.

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. 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

interesse em adquirir ou utilizar o sistema desenvolvido, fazendo com que o projeto fracasse e investimento realizado seja desperdiçado (BATTISTELLA, 2016).

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 poderá auxiliar futuros profissionais e amantes de jogos digitais.

Diante do exposto, o objetivo do presente estudo foi produzir um framework que englobe as características do *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 (FERNANDES, 2010) (LEAL, 2005) (MONCLAR; SILVA; XEXÉO, 2018). 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.

Framework

Um framework é um ponto comum que une códigos comuns entre projetos distintos de software e fornece uma funcionalidade genérica (SIQUEIRA; NASCIMENTO; PORTELA, 2018). 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 (PESSOA, 2001). 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* (FEIJÓ; DREUX; RAMALHO; BATTAIOLA, 2001). 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.

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” (LEAL, 2005). 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 (PARREIRA JUNIOR; FARIAS, 2016). 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, 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 (BATISTA; DIAS, 2012).

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* (LORENZI; RIBEIRO; KURTZ, 2018). Segundo Tim Brown (2017), 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*. 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. Essa abordagem consiste em quatro etapas: a imersão, ideação, prototipagem e desenvolvimento (LORENZI; RIBEIRO; KURTZ, 2018). 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 (GORBANEY; AGUDELO-LONDONO; GONZALEZ, 2018) e Scratch (MALAN; LEITNER, 2007). 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.

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 (GORBANEY; AGUDELO-LONDONO; GONZALEZ, 2018). Em outro espectro, Scratch é um software utilizado para criar jogos de forma simples. 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 (MALAN; LEITNER, 2007) (WOLZ; MALAN; MALONEY, 2009).

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 (DUNCAN, 2016) (CHRIS, 2017).

Do mesmo modo, também pode 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 conseqüentemente 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, vem sendo empregado em estudos de diversas áreas, tais como na comunicação e ensino de línguas estrangeiras (ROACH; UTAMI, 2005), comportamento do desperdício de alimentos (VERLOOP, 2018), linguagens figurativas em revistas de críticas de jogos (MERKL, 2019), desenvolvimento de jogos em realidade virtual (TANG; TRAN; BRIGGS, 2018), dentre outras.

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 (SOUZA, 1998). 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 (BECK, 2010). 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 (BECK, 2010). Segundo o autor Mike Cohn (2012), 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.

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. 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 (COHN, 2012).

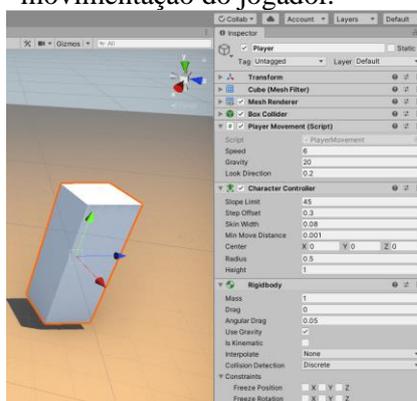
O último valor, que é a coragem, é necessário, pois dispensar um código já bem concretizado, iniciar do zero e realizar mudanças drásticas requer coragem por parte do desenvolvedor. Neste contexto, o emprego desta metodologia facilitou o desenvolvimento bem-sucedido do framework tendo em vista o grau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

Desenvolvimento do framework

O motor de jogos Unity 3D foi utilizado para desenvolver o framework almejado. 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 (AXON, 2018) (TAKAHASHI; TICCITIELLO, 2018).

O desenvolvimento do projeto iniciou com a programação das mecânicas do jogador anteriormente identificadas. 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 1).

Figura 1 – O cubo gerado e os seus respectivos componentes necessários para a movimentação do jogador.



Fonte: Screenshot da Unity durante o desenvolvimento do Framework

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 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.

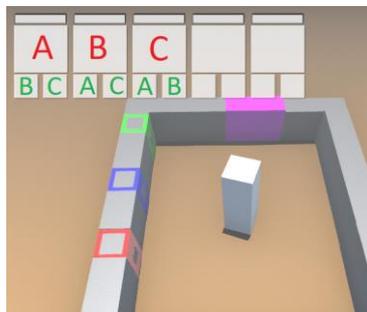
Com as três mecânicas do jogador desenvolvidas, implementadas e testadas com êxito, direcionou-se, então, as atenções para a implementação das mecânicas de cenário.

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 2, 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.

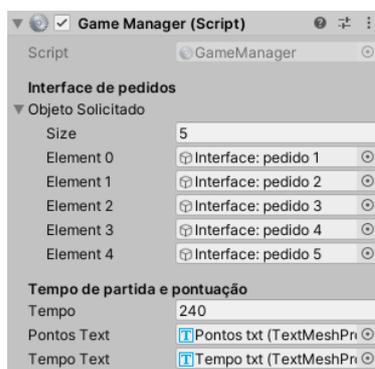
Figure 2 – Interface desenvolvida com o pedido, o seu tempo e ingrediente necessários.



Fonte: Screenshot da Unity durante o desenvolvimento do Framework.

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 3). 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.

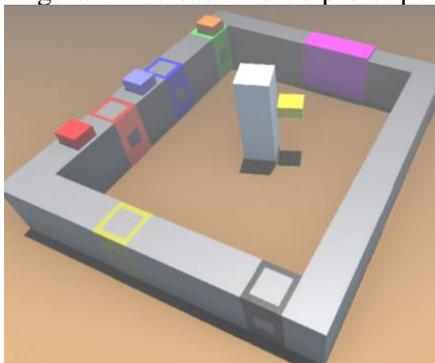
Figura 3 - O código principal cuja função é designar novos pedidos, verificar os objetos entregues, configurar o tempo de partida e pontuação do jogador.



Fonte: Screenshot da Unity durante o desenvolvimento do Framework.

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 4).

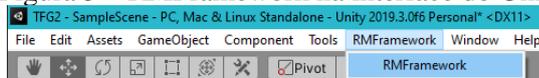
Figura 4 - Versão final do protótipo.



Fonte: Screenshot da Unity durante o desenvolvimento do Framework.

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 5). O framework desenvolvido no presente trabalho recebeu a nomenclatura “RMFramework”.

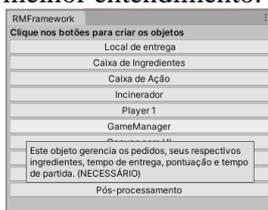
Figura 5 – RMFramework na interface do Unity.



Fonte: Screenshot da Unity durante o desenvolvimento do Framework.

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 6). 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 (CHRISTENSSON, 2006).

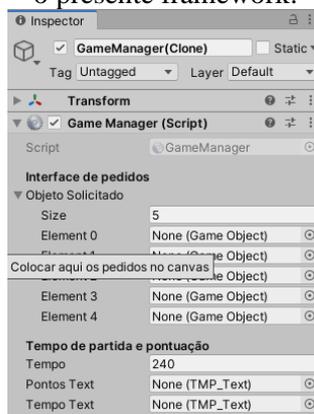
Figura 6 – O framework instalado no Unity 3D. Cada botão possui uma breve descrição para melhor entendimento.



Fonte: Screenshot da Unity durante o desenvolvimento do Framework.

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 no Unity (Figura 7) O inspetor é uma janela na interface que contém todas as informações e componentes de um objeto selecionado no Unity 3D.

Figura 7 - Inspetor do novo objeto instanciado pelo desenvolvedor utilizando o presente framework.



Fonte: Screenshot da Unity durante o desenvolvimento do Framework.

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

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 (COELHO; BACELAR; RODRIGUES NETO; JACOB, 2015). 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 para a implementação gráfica pretendida.

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 8), 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 às suas características inerentes e analisa elementos como jogabilidade, grau de diversão e dificuldade, entre outros. 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 (CHOI et. al, 2016), 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 (CHOI et. al, 2016), uma vez que não participaram diretamente da produção do jogo e, portanto, não possuem o mesmo viés dos desenvolvedores.

Figura 8 – Estado final do protótipo desenvolvido com o framework.



Fonte: Screenshot do produto desenvolvido em funcionamento.

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 (COELHO; BACELAR; RODRIGUES NETO; JACOB, 2015). 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 específicas. 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.

Referências

AXON, S. **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/>.

BACELAR, Wagner A. Coelho Lucas R.; RODRIGUES NETO, José M.; JACOB JR, Antonio FL. **A Maldição de Bibi Costa:** desenvolvimento de um jogo voltado para a cultura amazônica. SBC – Proceedings of SBGames 2015 | ISSN: 2179-2259.

BATISTA, Drielly Adrean; DIAS, Carmen Lúcia. O processo de ensino e de aprendizagem através dos jogos educativos no ensino fundamental. **Revista Colloquium Humanarum**, São Paulo. 2012. p. 975-982.

BATTAIOLA, André Luiz et al. **Desenvolvimento de jogos em computadores e celulares.** RITA, v. 8, n. 2, p. 7-46, 2001.

BATTISTELLA, P.. **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.

BECK, K. **Programação extrema explicada:** acolha as mudanças. Porto Alegre: Bookman, 2010.

BROWN, T. Design Thinking. **Uma metodologia poderosa para decretar o fim das velhas ideias.** Rio de Janeiro: Alta Books, 2017.

CHOI, Judeth Oden et al. **Playtesting with a Purpose.** In: Proceedings of the 2016 annual symposium on computer-human interaction in play. 2016. p. 254-265.

CHRIS, K. **Road to the igf:** ghost town games' Overcooked. Gamasutra, https://www.gamasutra.com/view/news/290885/Road_to_the_IGF_Ghost_Town_Game_s_Overcooked.php 2017.

CHRISTENSSON, P. **Tooltip definition.** TechTerms. 2006. Acessado em 16/05/2020. Disponível em <https://techterms.com/definition/tooltip/>.

COHN, M. **Desenvolvimento de software com Scrum:** aplicando métodos ágeis com sucesso. Porto Alegre:Bookman, 2012.

Duncan, P. **Game design deep dive:** building truly cooperative play in Overcooked. Gamasutra,

https://www.gamasutra.com/view/news/279715/Game_Design_Deep_Dive_Building_truly_cooperative_play_in_Overcooked.php 2016.

FERNANDES, N. **Uso de jogos educacionais no processo de ensino e de aprendizagem**, 2010.

GORBANEV, Iouri et al. **A systematic review of serious games in medical education: quality of evidence and pedagogical strategy**. *Medical education online*, v. 23, n. 1, p. 1438718, 2018.

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

LORENZI, Fabiana; RIBEIRO, Vinícius Vargas; KURTZ, Gabriela Birnfeld. **RPG Educacional para o ensino de Design Thinking**. SBGMAES, Proceedings, p. 1239-1248, 2018.

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.

MERKL, P. **Figurative language in online game reviews**, Bachelor thesis of the University of Pardubice - Faculty of Arts and Philosophy, 2019.

MONCLAR, R.; SILVA, M; XEXÉO, G. **Jogos com propósito para o ensino de programação**. SBC – Proceedings of SBGames 2018 — ISSN: 2179-2259.

PARREIRA JÚNIOR, Walteno Martins; DE FARIAS, Renato Vasconcellos. **O aprendizado através de jogos educativos**. SIED: EnPED-Simpósio Internacional de Educação a Distância e Encontro de Pesquisadores em Educação a Distância, 2016.

PESSOA, C. **wGem: um motor de desenvolvimento de jogos para dispositivos móveis**. Undergraduate Final Year Project, UFPE, 2001.

RIVA, C. **Novos tempos, novas crianças**. 2009.
<http://www.dihoje.com.br/dihoje2009/?pg=noticia&id=1360>. Acesso em 23 nov. 2019.

ROACH, Alex; UTAMI, Yeski. **Using video game to enhance English communication skills**. Proceedings of ISELT FBS Universitas Negeri Padang, v. 5, p. 200-204, 2017.

SAKUDA, Luiz Ojima; FORTIM, Ivelise. **II censo da indústria brasileira de jogos digitais**. Ministério da Cultura: Brasília, 2018.

SIQUEIRA, Kelven; NASCIMENTO, Elison; PORTELA, Carlos. **Uma proposta de processo padrão para apoiar o desenvolvimento de jogos educativos**. Anais do Simpósio Brasileiro de Jogos e Entretenimento Digital, Foz do Iguaçu, 2018.

SOUZA, Cleidson Ronald Botelho de et al. **Um framework para editores de diagramas cooperativos baseados em anotações**. 1998. 105f. Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação, Campinas, SPC.

TAKAHASHI, Dean. John Riccitiello **Q&A**: how unity ceo views epic's fortnite success. VentureBeat. Archived from the original on September, v. 17, 2018.

TANG, David Yuhua; TRAN, Jimmy Ngoc; BRIGGS, Sean Robert. **Ben-tomo!** A Cooperative VR Cooking Game. 2018.

VERLOOP, Christiaan. **Developing a serious game as a tool for collecting data on food waste behavior**. 2018. Trabalho de Conclusão de Curso. University of Twente.

WOLZ, Ursula et al. **Starting with scratch in CS 1**. In: Proceedings of the 40th ACM technical symposium on Computer science education. 2009. p. 2-3.