André Salomão

Game Design: construindo um protótipo

Projeto de Conclusão de Curso (PCC) submetido ao Programa de Graduação da Universidade Federal de Santa Catarina para a obtenção do Grau de Bacharel em Design.
Orientador: Prof. Ms. Flávio Andaló.

Florianópolis
2017

Ficha de identificação da obra elaborada pelo Autor
através do Programa de Geração Automática da Biblioteca Universitária
da UFSC.

Salomão, André
Game Design: Construindo cenário 3D de jogo / André Salomão; orientador, Flávio Andaló - Florianópolis, SC, 2017.
139 p.
Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro de Comunicação e Expressão. Graduação em Design.
Inclui referências
1. Design. 2. Game Design. 3. Cenário 3D. 4. Metodologia de Jogos. I. Andaló, Flávio. II.
Universidade Federal de Santa Catarina. Graduação em Design.
III. Título.

André Salomão

***Game Design:* construindo um protótipo**

Este Projeto foi julgado adequado para obtenção do Título de "Bacharel em Design" e aprovado em sua forma final pelo Curso de Design da Universidade Federal de Santa Catarina

Florianópolis, 19 de junho de 2017.

_____
Prof. Luciano Patrício Souza de Castro, Dr.
Coordenador do Curso

Banca Examinadora:

_____
Prof. Flávio Andaló, Mestre
Orientador
Universidade Federal de Santa Catarina

_____
Prof.ª Mônica Stein, Dr. ª
Universidade Federal de Santa Catarina

_____
Prof. William Machado de Andrade, Doutor
Universidade Federal de Santa Catarina

AGRADECIMENTOS

Agradeço a minha família pelo apoio.
Um grande *Special Thanks* ao meu grande amigo Gustavo, por me aguentar dia após dia, durante anos.
Ao grupo *SaloRage* por me empurrarem a fazer essa ideia de trabalho.
Aos meus amigos do grupo *EDGE* em *Final Fantasy XIV*
Aos meus amigos do Design.
A todos os professores da área de Animação, por tudo que aconteceu nos últimos quatro anos.
À Professora Raquel Martinelli por fazer da aula de *GD* uma das melhores da faculdade e suas indicações.
À Professora Cristina Colombo Nunes por me ensinar a ver o cenário do ponto de vista de um arquiteto.
E a todos que eu esqueci de mencionar.

RESUMO

Este Projeto de Conclusão de Curso tem por objetivo abordar os diferentes aspectos do *game design* e como é importante conhecer e entender cada uma dessas áreas para o desenvolvimento de um *Protótipo de Jogo*. Além disso, a sua finalidade é criar/trazer uma experiência e emoção ao usuário que é o jogador nesse projeto, se utilizando de uma mistura de três diferentes metodologias para chegar a esse objetivo.

Palavras-chave: *Level Design*, Protótipo, Experiência.

ABSTRACT

The objective of this course conclusion project is to approach the different aspects of game design, and the importance of knowing them to better understand the development of Game's Prototype. Besides, this project uses 3D to create an experience and emotion for the user, the player in this project, by using a mix of three different methodologies.

Keywords: Level Design. Prototype. Experience.

LISTA DE FIGURAS

Figura 1 MDA Relação Designer e Player 22
Figura 2 MDA – Componentes 22
Figura 3 MDA Perspectiva Designer e Jogador 23
Figura 4 Elementos 24
Figura 5 Gráfico Flow 28
Figura 6 Projeto feito pelo autor para a Escola Melies 32
Figura 7 Modelo Personagem feito Zbrush 33
Figura 8 Programa Substance Painter 34
Figura 9 Projeto Portfólio pessoal Casa Abandonada 34
Figura 10 Exemplificação Assets na Unreal Engine 4 35
Figura 11 Jogo desenvolvido na Unreal 4 – Paragon 36
Figura 12 Jogo Left 4 Dead 37
Figura 13 Jogo Outlast 38
Figura 14 Arena de Rocket League 40
Figura 15 Mapa de um nível de Battlefield 4 40
Figura 16 Mapa Octógono de Rocket League 41
Figura 17 Mapa duas paredes de Rocket League 42
Figura 18 Mapa Battlefield 4 Modificado 43
Figura 19 Final Fantasy 6 Dentro e Fora da cidade 44
Figura 20 Exemplificação da relação objeto-jogador 47
Figura 21 Sistema de Regras por David Parlett 49
Figura 22 Cenário jogo Crysis 52
Figura 23 Imagem publicitaria Battlefield 1 52
Figura 24 Jogo Journey 53
Figura 25 Cena da Opera de Final Fantasy 6 54
Figura 26 Notre Dame em Assasin's Creed Unity 55
Figura 27 Notre Dame – Paris 56
Figura 28 Comparação entre Ator (esquerda) e Personagem (direita) 56
Figura 29 Cenário de Bioshock 57
Figura 30 Concept Arte Cenário God Of War 58
Figura 31 Referência de Ilha Maldivas 59
Figura 32 Referência Ilha temporal 59
Figura 33 Cenário do jogo Uncharted 4 60
Figura 34 Cenário ilha do jogo Uncharted 4 60
Figura 35 Arena de Rocket League 61
Figura 36 Capa do Monster Truck Madness 2 61
Figura 37 Planta baixa do Mapa de Doom 66
Figura 38 Cenário Mock-up Panorâmica 72
Figura 39 Visão Jogador no Mock-up 73
Figura 40 Plataforma do Cenário do Mock-up 73
Figura 41 Modelo Ilha 77
Figura 42 Concept Cenário Antigo 79
Figura 43 Concept novo cenário 79

Figura 44 Primeiro Modelo Plataforma Base 81
Figura 45 Plataforma Base v2.....81
Figura 46 Teste v2 na Unreal Engine 4.....82
Figura 47 Modelo Plataforma-Base Final.....83
Figura 48 Modelo Plataforma Base não otimizado.....84
Figura 49 Modelo Plataforma Base otimizado.....84
Figura 50 Plataforma Otimizada c/referência 85
Figura 51 Uvs Plataforma Base 85
Figura 52 Modelo Plataforma King of the Hill.....87
Figura 53 Uv's Plataforma King of The Hill.....88
Figura 54 Modelo Rampa e Plataforma Inferior 89
Figura 55 UVs Rampa 89
Figura 56 Uvs Suporte da Rampa 90
Figura 57 Uvs Plataforma Inferior.....90
Figura 58 Modelo Base do Poste 91
Figura 59 Modelo Refletores de Luz 91
Figura 60 Plataforma Base Textura Teste 2.....94
Figura 61 Plataforma Base Textura Teste 2.....94
Figura 62 Plataforma Base Textura Teste 3.....94
Figura 63 Mapa Height Plataforma Base v1 95
Figura 64 Teste Plataforma Base dentro da Unreal Engine 4 95
Figura 65 Mapa de Mask Plataforma-Base.....96
Figura 66 Teste com mapa de Mask 96
Figura 67 Teste Textura 1 Plataforma Base.....97
Figura 68 Teste Textura 2 Plataforma Base.....97
Figura 69 Teste Textura 3 Plataforma Base.....98
Figura 70 Teste Textura Circulo exterior.....98
Figura 71 Textura Detalhes.....99
Figura 72 Textura finalizada v1 Plataforma Base.....100
Figura 73 Mapa Height v2 100
Figura 74 Textura finalizada v2 Plaataforma Base 101
Figura 75 Mapa Height v3 102
Figura 76 Textura vFinal Plataforma Base 102
Figura 77 Mapas de Textura para importação na Unreal Engine 4.....103
Figura 78 Material da Plataforma Base dentro da Unreal Engine 4.....103
Figura 79 Plataforma Base texturizada dentro da Unreal Engine 4 104
Figura 80 Mapa Height + Textura Metal Plataforma King of The Hill 105
Figura 81 Teste Textura Linha de Chegada 105
Figura 82 Mapa Mask Plataforma King Of The Hill 106
Figura 83 Textura v2 Plataforma King Of The Hill.....106
Figura 84 Plataforma King Of The Hill teste dentro da Unreal Engine 4 107
Figura 85 Mapa de exportação para UE4.....107
Figura 86 Textura vFinal Plataforma King of The Hill 108
Figura 87 Rampa & Suporte Texturizada dentro da Unreal Engine 4 108
Figura 88 Plataforma Inferior dentro da Unreal Engine 4 109

Figura 89 Refletores texturizados dentro da Unreal Engine 4 109
Figura 90 Base do Poste Texturizada dentro da Unreal Engine 4 110
Figura 91 Configuração Light Source 111
Figura 92 Configuração Sky Sphere 111
Figura 93 Posicionamento das Spot Lights 112
Figura 94 Área afetada pela iluminação da Spot Light (em desenvolvimento) 112
Figura 95 Cenário iluminado com a ajuda de Point Light (em desenvolvimento) 113
Figura 96 Iluminação do cenário demonstrando a iluminação (em desenvolvimento) 113
Figura 97 Caixa do Volume da Luz Indireta 114
Figura 98 Sphere Reflection em cima da plataforma King of The Hill 115
Figura 99 Efeito da Caixa de Pós-Processamento 1 115
Figura 100 Efeito da Caixa de Pós-Processamento 2 116
Figura 101 Efeito da Caixa de Pós-Processamento 3 117
Figura 102 Caixa de configuração do Pós-Processamento 117
Figura 103 Bluprint Jogador 119
Figura 104 Blueprint Veículo 119
Figura 105 Blueprint para checar se o carro capotou ou não 120
Figura 106 Blueprint Update de Parâmetros 121
Figura 107 Bluprint Evento Respawn & Destruição Veículo 121
Figura 108 Blueprint Respawn carro quando capotado 122
Figura 109 Configuração Blueprint – Carro 1 123
Figura 110 Configuração Blueprint - Carro 2 124
Figura 111 Menu de Jogo 125
Figura 112 Blueprint Multiplayer 126
Figura 113 Blueprint Menu 126
Figura 114 Cenário Protótipo vFinal 130
Figura 115 Carro Protótipo para Teste 130
Figura 116 Cenário visão Wireframe 131
Figura 117 Visão Plataforma Inferior 131
Figura 118 Visão Rampa 132
Figura 119 Visão Plataforma-Base 132
Figura 120 Visão rampa acesso Plataforma King of The Hill 133
Figura 121 Visão Plataforma King of The Hill 133

LISTA DE ABREVIATURAS E SIGLAS

2D – Duas dimensões
3D – Três dimensões
PCC – Projeto Conclusão de Curso
HUD - *Heads-up display* (Tela de alerta ou *Interface* do jogo)
MDA – *Mechanism, Dynamic, Aesthetics*
SNES – Super Nintendo
RPG – *Role Play Game*
PS1 – Playstation *One*
PS2 – Playstation *Two*
PS3 – Playstation *Three*
PS4 – Playstation *Four*
NES – Nintendo *Entertainment System*
N64 – Nintendo 64
UE4 – Unreal Engine 4

SUMÁRIO

1 INTRODUÇÃO 16
1.1 Apresentação do Tema 16
1.2 Objetivo Geral 19
1.2.1 Objetivos Específicos 19
1.3 Justificativa 19
1.4 Metodologia Projetual 21
2 PROBLEMATIZAÇÃO 26
3 COLETAS E ANÁLISES 29
3.1 O que é um Jogo? 29
3.2 Tecnologia 31
3.2.1 Modelagem e Textura de *Assets* 32
3.2.2 Programa *Unreal Engine 4* 35
3.3 Tema 36
3.4 Estética 50
3.5 Material de Referência 55
3.6 *Level Design* 62
4 SÍNTESE 68
4.1 Processo de Conceito da Ideia 68
4.2 *Executive Summary* 70
4.3 *Mock-up* 71
5 DESENVOLVIMENTO 74
5.0.1 *Feedback* & Ajustes 74
5.1 Modelagem 75
5.1.1 *Assets* e UVs 76
5.1.2.1 Ilha e Consequências 77
5.1.2.2 Plataforma Base 78
5.1.2.3 Plataforma *King of The Hill* 86
5.1.2.4 Rampas e Plataforma Inferior 88
5.1.2.5 Objetos Secundários 90

5.2 Texturização 92
5.2.1 Plataforma-base 93
5.2.1 Plataforma *King of The Hill* 104
5.2.1 Rampas e Plataforma Inferior 108
5.2.1 Objetos Secundários 109
5.3 Iluminação e Pós-Processamento 110
5.4 *Mechanics* 118
6 *EXECUTIVE SUMMARY* VERSÃO FINAL 128
7 CONCLUSÃO 134
REFERÊNCIAS 138

1 INTRODUÇÃO

"Nós não nos olhamos como artistas, ao invés, como artesãos"
(Dillen & Plate)

Quando se desenvolve qualquer tipo de projeto, metodologias e diferentes ferramentas são utilizadas para chegar-se ao resultado final, nesse trabalho a ideia é utilizar os conhecimentos e metodologias envolvidas em Game Design.

Seguindo essas metodologias para o desenvolvimento de um protótipo de jogo, onde se demonstrou a importância do conhecimento teórico de Game Design para a construção de um produto que gere uma experiencia ao jogador, assim pretende-se trazer a seguinte ideia onde "O caso é que você não precisa replicar perfeitamente experiências reais para fazer um bom jogo (...) capture a essência dessas experiências para o seu jogo. " (SCHELL, 2015, p. 68).

Na parte prática do desenvolvimento de todo o processo, deve-se levar em conta que esse projeto e considerações estão sendo feitas e produzidas por apenas uma pessoa, que é o autor desse PCC, logo dificuldades que seriam separadas e resolvidas em grupos por pessoas diferentes, terão que ser tomadas em sua maioria individualmente.

1.1 Apresentação do Tema

A evolução da tecnologia produzida pelo ser humano no último século é algo singular, pois conseguiu-se evoluir em pouco tempo muito mais do que em milênios. No design não é algo diferente, a evolução também foi abrupta e, de acordo com Cardoso (2008) começou a sua expansão na primeira revolução industrial na Inglaterra e depois percorreu por diversas fases como *Arting and Crafting, Art Nouveau*, transitando pelo vanguardismo europeu e a Escola de Bauhaus, sofrendo diversas formas de influências, passando também pela fase de Modernidade e onde se encontra – na atualidade – na Pós-Modernidade; muita coisa evoluiu e se adaptou nos tempos atuais, e não apenas isso como também se expandiu para diferentes áreas.

Uma dessas novas áreas as quais o design tem se expandido é a área de criação e desenvolvimento de experiências como por exemplo vídeo games.

Ser um *designer* é saber tomar decisões. Independente de qual parte do processo de desenvolvimento de um jogo, qualquer decisão

tomada faz parte do design inteiro do jogo, logo os conhecimentos necessários para tornar-se um profissional da área, de acordo com Schell (2015), envolve alguns exemplos como:

- Animação: Jogos modernos e até antigos, personagens e plataformas, todos tem uma animação que servem para dar vida a esse objeto. Entender o poder dessa animação permite gerar novas ideias e inovações no tocante ao design do jogo;
- Arquitetura: A criação de um jogo muitas vezes é de todo um novo universo, seja ele baseado em locais reais como a série *Assassins's Creed* faz, ou a criação de um novo mundo totalmente diferente. Entender a relação entre pessoa e espaço é entender grande parte do processo de criação de novos mundos;
- *Brainstorming*: Assim como encontrado em muitos livros de desenvolvimento de design, o *brainstorm* na criação de jogo é algo primordial. O processo de tempestade de ideias é necessário para que se encontre 'aquela ideia' que vai ser usada para dar início ao desenvolvimento do projeto;
- Cinematografia: Entender conceitos de cinematografia, principalmente hoje onde a tecnologia permite que jogos fiquem cada vez mais perto de experiências de filmes como *The Last of Us*, é essencial para a criação de uma experiência emotiva para o usuário. Não apenas isso, mas ângulos de câmeras e *gameplay* também são exemplos de atividades dentro de um jogo que usam princípios de cinema aplicados aos jogos;
- Comunicação: Saber comunicar-se dentro de uma empresa e com o seu público-alvo é essencial para o desenvolvimento de um jogo;
- Roteiro criativo: Assim como por exemplos em filmes, jogos precisam de um roteiro, seja para elaborar a história, ou para a descrição do próprio mundo a ser criado, então conhecer princípios de roteiro ajudam a exemplificar e demonstrar o produto a ser ou sendo criado;
- Engenharia: A plataforma na qual se trabalha para a criação e desenvolvimento de um jogo é feito em base de múltiplas complexidades da engenharia. São essas engenharias que permitiram a criação de novas tecnologias e plataformas e fizeram com que os jogos evoluíssem ao longo do tempo. São essas plataformas que trazem os parâmetros e limitações que

os *games designers* têm que saber como lidar e para adaptar suas ideias;

- *Games*: Saber entender os jogos do passado, os porquês foram criados e da forma como o foram, quais suas ideias, temas, estéticas, decisões tomadas durante o desenvolvimento, todos esses detalhes servem como exemplos do que pode dar certo e o que pode dar errado na criação de um novo produto;
- História: Muitas vezes, assim como a arquitetura de um jogo pode ser baseada em uma época, saber entender as características de determinada época, o costume e hábitos de um povo por exemplo: vestimentas, características físicas, sociais e psicológicas, são importantes para uma possível replicação em jogos;
- Matemática: Todos os jogos têm alguma fórmula matemática, sejam em possíveis atributos numéricos dados ao personagem (como +5 de Inteligência), um sistema de pontuação de pontos (gols em Fifa 17), análises de riscos e probabilidades, mecânicas de jogos, entre outros. São apenas alguns exemplos onde conhecimento em matemática e fórmulas são necessárias para a aplicação dentro do jogo;
- Música & *Sound Design*: A música é a alma do jogo. Mesmo em jogos antigos de *Arcade* os sons davam um "ar novo" ao jogo, escutar as coisas explodindo porque você atirou nos invasores em *Missile Command*, dava emoção ao jogador. Escutar a trilha sonora da personagem *Celes* no teatro cantando em *Final Fantasy VI* é considerado por muitos como um dos momentos mais simbólicos da história de jogos, e isso quando a tecnologia ainda era primitiva e restringia o desenvolvimento. Hoje é possível colocar em jogos trilhas sonoras como as vistas no filme 'Senhor dos Anéis'; e saber utilizar isso é fundamental para realmente ressonar no jogador (a), de forma a fazê-lo gostar e se aprofundar no mesmo;
- Arte Visual: O jogo será composto de diversos elementos gráficos, seja por causa da HUD do jogo, símbolos, ícones, ou por questão de estilo, saber entender a linguagem gráfica de design é essencial para mostrar o *feeling* do jogo.

Existem mais (elementos e outros subelementos dentro dessas próprias características e é trabalhoso para o profissional da área dominar e ter total conhecimento sobre todos esses elementos. Conforme Dille &

Platten (2015, p. 17), “Uma grande parte do trabalho do designer é comunicar efetivamente sua visão”. Por isso em desenvolvimento de jogos, praticamente cada uma dessas áreas tem seus próprios departamentos e subdepartamentos, mas saber que essas áreas existem e o conceito e função de cada um diferencia um *designer* e o ajuda a fazer a diferença no seu trabalho é considerado

Logo para desenvolver o tema desse trabalho, que é a criação de um *level design* em 3D para um protótipo de jogo, não basta apenas saber a parte técnica ou desenvolver o cenário direto na ferramenta de desenvolvimento, é necessário um conhecimento geral de como a cadeia de desenvolvimento de um jogo funciona, quais suas características, procedimentos necessários e seus valores. Quanto maior o entendimento dessas áreas, mais fácil é criar um *level design* que possa ser aplicado dentro dessa longa cadeia processual que é a criação e desenvolvimento de um jogo.

Conforme Shell (2015, p.184), “(...) se você consegue entender e controlar como essa ilusão é formada na mente do jogador, você irá fazer com que pareçam reais, ou mais real que a própria realidade”.

Para entender o processo por trás da criação de um jogo, mais especificamente de um protótipo, deve-se ter em mente que o cenário é apenas uma parcela de toda a técnica a ser empregada, pois de acordo com Schell (2015, p. 51) “(...) a habilidade mais importante para um game designer é escutar”, e isso ajuda a compreender que esse desenvolvimento ‘parcelado’ irá se combinar com outros elementos, para criar efetivamente um produto de entretenimento.

1.2 Objetivo Geral

Construir um protótipo de jogo 3D.

1.2.1 Objetivos Específicos

- Desenvolver o processo criativo e prático de um jogo;
- Ter um produto final que deve resultar em uso de portfólio profissional.

1.3 Justificativa

“O propósito do design é resolver problemas, e game design não é uma exceção. Antes de você começar a pensar em ideias, você precisa ter certeza do porque está fazendo isso, e a afirmação do problema é um jeito de se deixar isso claro.

Boas afirmações de problemas dizem o seu objetivo e as suas restrições"
(Schell)

Desde a criação do Curso de Design, na UFSC o enfoque dado à animação e jogos de vídeo games foi norteado pela procura e interesse nessa área. Sendo assim, com o crescente interesse e aumento de público, bem como os relevantes ganhos monetários (para criadores, provedores, desenvolvedores e usuários) foi possível perceber que o curso tomou outros rumos e para melhor, focando no processo de criação de animação em 2 e 3D voltados para jogos recreativos e educativos.

Outro fator importante é o crescimento de jogos no mercado mundial e brasileiro. O site *Newzoo*[1] publicou em 2016 sua pesquisa relacionada à indústria de jogos, mostrando que o mercado global chegou a 99.6 bilhões de dólares, sendo a plataforma de computadores a que mais gira dinheiro, responsável por 32% desse número, enquanto os consoles chegam a 29% do total.

No Brasil, um dado trazido pelo *site* brasileiro G1[2], em 2015, mostra que a participação do Brasil nesse mercado ultrapassa 1 bilhão de dólares. A influência dos jogos é notória inclusive no fato da criação de eventos no Brasil como o *Brasil Game Show* - anualmente realizada em São Paulo - uma exposição para a publicação de jogos tanto brasileiros, quanto convidados estrangeiros no país.

Levando-se em conta essas informações, mostra-se que um investimento na área e um estudo maior na parte de design - como estudo de novas aplicações e tecnologias - e como ele pode entrar impulsionar o mercado são opções de negócios com chances de retorno.

Por isso nesse trabalho pretende-se mostrar que o *designer* é a pessoa responsável por qualquer tipo de conteúdo criativo dentro de um jogo, seguindo o raciocínio de que é como o diretor de um filme, se o jogador consegue usar algo, atirar em algo, modificar alguma coisa dentro da plataforma de um jogo, então é responsabilidade do *game designer* saber identificar esses elementos, adaptá-los da melhor forma possível para o público que está interagindo com o produto criado.

Afinal, jogos são apenas um meio para um fim mais importante, que é gerar experiências para o usuário, ou de acordo com Schell (2015, p.11) "(...) o jogo possibilita a experiência, mas não é a experiência em si", ou seja, há muitos elementos a serem explorados nessa área.

[1] Site especializado em dados estatísticos e métricas em geral que atende aos interessados em jogos, esportes eletrônicos e inteligência móvel.
[2] Site da Rede Globo de Comunicação.

Para aprender a parte teórica e prática de todo o processo envolvido na criação de um jogo, seja ele algo simples ou mais complexo, é necessário a criação de um cenário voltado para jogo, porque apesar de ser apenas uma parte da experiência total do usuário com o jogo, ainda é uma parte que é conectada a todas as outras.

Assim, também é no desenvolvimento de um jogo, onde cada área dentro de um desenvolvimento acaba de uma forma ou de outra influenciando as decisões que devem ser tomadas. Então, aprender a saber qual o problema a ser resolvido e seus objetivos, ajudam a dar diretriz ao trabalho, pois "(...) ao observar as próprias experiências e a dos outros, tentando se colocar no lugar deles, você começa a criar uma imagem de como a sua experiência se diferencia da dos outros" (Schell, p. 17).

1.4 Metodologia Projetual

> "Enquanto é frequentemente necessário focar em uma área, todo mundo, independente da disciplina, irá em algum ponto precisar considerar problemas fora daquela área: mecanismo básicos de um sistema de jogo, os abrangentes objetivos do design, ou os resultados experienciais desejados de um gameplay" (Hunicke, LeBlac e Zubek)

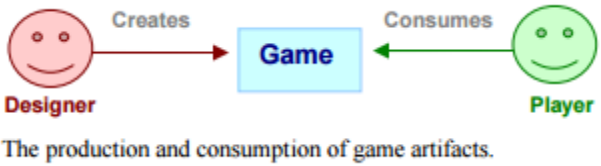
A base teórica principal é o artigo *MDA: A Formal Approach to Game Design and Game Research,* produzido por Robert Hunicke, Marc LeBlanc e Robert Zubek, que servirá como referência das aplicações teóricas e práticas, sendo que os autores Schell e Wallman serão utilizados como conteúdo de apoio e complementares.

Com relação ao MDA – Mecânicas, Dinâmica e Estéticas é um formato convencional de entender criação, produção e desenvolvimento de jogos; aquilo que faça a ligação entre o jogador e o *designer* e ajuda a desenvolver, criticar e gerar tecnicalidades sobre a pesquisa do jogo.

Esse papel científico ajudou, e ajuda, dentro de sua metodologia a esclarecer e reforçar o processo interativo de desenvolvedores, tornando mais fácil decompor o estudo de design dentro de uma grande variedade de jogos.

A meta é criar um jogo onde o sistema todo seja coerente entre si, e isso apenas é possível quando todas as partes de um jogo conseguem se relacionar como um todo. Decompor, entender e criar essa coerência requer um extenso trabalho em todos os níveis de abstração. As misturas dessas três metodologias irão servir como ferramentas para entender esse processo e tomar as decisões dentro do ciclo de produção de um jogo. Conforme é possível observar na imagem e a explicação a seguir:

Figura 1 MDA Relação Designer e Player

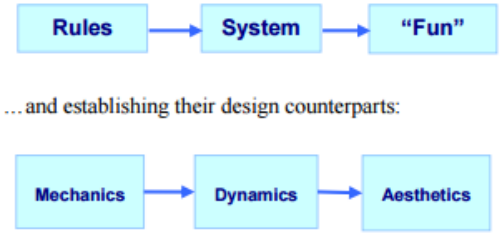


Fonte: *MDA: A Formal Approach to Game Design and Game Research* (2001).

O modo mais simples de mostrar como funciona o sistema é demonstrando que *designers* produzem o jogo e o jogador o consome. A diferença entre jogos e outros tipos de entretenimentos é que seu consumo é relativamente imprevisível.

MDA formaliza essa cadeia de relacionamentos quebrando-os em diferentes componentes conforme é possível observar:

Figura 2 MDA – Componentes



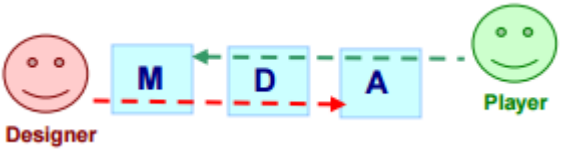
Fonte: *MDA: A Formal Approach to Game Design and Game Research* (2001).

- Regras (Rules) >Sistema (System) >Dinâmica (Fun)

E, estabelecendo suas contrapartidas em questão de design:

- Mecânicas(Mechanics)>Dinâmica(Dynamics)>Estética(Aesthetics)

Figura 3 MDA Perspectiva Designer e Jogador



Fonte: *MDA: A Formal Approach to Game Design and Game Research* (2001).

Além disso, quando se trabalha com jogos é necessário considerar ambas as perspectivas, tanto do *designer* quanto do jogador, porque qualquer mudança, seja ela pequena ou não, cria um efeito dominó que irá afetar a ambos. Pensando também no jogador deve-se incentivar a criação de design por experiência em vez de design por conteúdo.

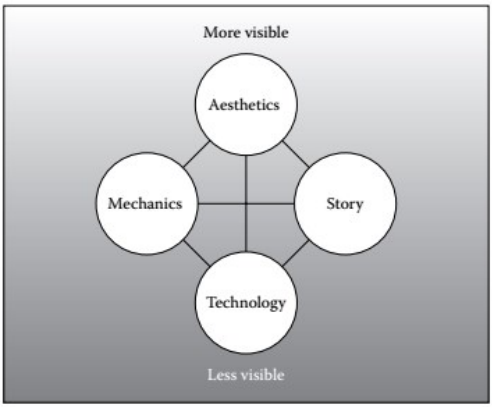
Ao usar essa metodologia, o ideal é olhar os jogos enquanto artefatos, onde o conteúdo do jogo é considerado o 'comportamento' e não a 'mídia', que é o canal de comunicação com o jogador.

De forma mais detalhada, o MDA determina que se olhe essas características com Lentes (*Lens*), que seria como ver diferentes perspectivas sobre diversos detalhes e decisões que *designers* precisam tomar que influenciam a interação final do jogador com o produto. Essas lentes nada mais são que pequenos grupos de perguntas a serem feitas sobre o *design* durante o ciclo de produção.

Para facilitar o entendimento de todos os processos de MDA, o livro *The Art of Game Design: a Book of Lenses* (SCHELL, 2015) contém diversas dessas lentes com uma série de perguntas para a serem feitas durante o processo de produção. Além disso, para facilitar o entendimento e aplicação dos mesmos, usou-se como recurso o fatiamento do MDA em quatro partes, que servem para exemplificar e detalhar melhor o processo como um todo, mantendo o sistema criado pela metodologia de MDA.

Nesse caso, têm-se quatro elementos básicos, semelhantes ao MDA:

Figura 4 Elementos



Fonte: SCHELL, 2015, p. 51

Entende-se *Mechanics* (Mecânica) como: Onde se foca no procedimentos e regras do jogo, descrevem os objetivos do mesmo, como os jogadores conseguem, ou não, chegar nesse objetivo final e o que acontece quando eles tentam. É essa a parte de um jogo que faz um jogo ser um jogo, porque difere de livros e filmes, os jogos envolvem mecânicas cruciais como o *gameplay*.

A seguir, tem-se *Story* (História): Essa é a sequência que se desenrola dentro do jogo. Pode ser algo linear ou divergir em vários segmentos e possibilidades. Pode ser um cenário onde algo aconteceu, e esse cenário reflete a história passada, como por exemplo um templo antigo com escrituras de um povo perdido.

Por *Aesthetics* (Estéticas): Aqui é como o jogo se parece visualmente, como ele repercute sonoramente, como as pessoas sentem o jogo e como emoções são geradas através dele. É onde existe a maior relação direta com a experiência que o jogador irá sentir, assim como demonstrado na figura 3.

Por fim, há a *Technology* (Tecnologia): Não necessariamente apenas tecnologia de ponta, como um PC caro que rode o jogo todo no máximo, mas sim qualquer material e interação que faz o jogo, seja ele por exemplo um papel de caneta para se jogar *Dungeous & Dragons*, ou um tabuleiro para jogar Banco Imobiliário. Tudo isso se encaixa na parte de tecnologia de um jogo; é isso que irá criar delimitações, possibilidades e impossibilidades que o jogo poderá ter.

Importante ressaltar que todos os elementos interagem entre si, nenhum deles funciona sozinho e nenhum é mais importante que o outro; é achar a coerência entre esse sistema que ajuda a criar um bom *design* de jogo. Ao utilizar esses quatro elementos, tem-se como referência que é utilizar uma lente que tenha "(...) um estoque do que o seu jogo realmente é feito de. Considere cada elemento separado e então todos eles juntos como um só"[3].

Durante todo o processo de decisão que envolve definir os vários elementos criados ou a serem criados no desenvolvimento do jogo, deve-se levar em consideração as lentes como um todo, porém seu uso irá depender da demanda e objetivo do jogo. Algumas dessas lentes serão apresentadas durante esse PCC.

Além do mais, é necessário um ciclo de produção, já que a produção do jogo não é algo linear e sim um ciclo constante de ajustes e reajustes.

Baseado no artigo de Jim Wallman[4] e utilizando o processo nomeado de *Formal Loop* do livro de Schell (2015) chega-se a um processo de metodologia para o desenvolvimento do jogo:

Passo 1: Decidir Objetivos e Problemas
Passo 2: *Brainstorm* de ideias
Passo 3: Definir a ideia
Passo 4: Examinar considerações de *design* sobre a escolha incluindo, mas não apenas:

- ✓ Estrutura do jogo
- ✓ Tipo de jogo
- ✓ Level
- ✓ Recursos
- ✓ Riscos

Passo 5: Testar o *Design*

- ✓ Objetivos foram alcançados?
- ✓ Reescrever ou Descartar ideia?

Passo 6: Escrever regras do jogo
Passo 7: Criação de protótipos para teste do Design para diminuir o risco de problema.

- ✓ Riscos do uso da plataforma

[3] Disponível em aplicativo específico chamado *Game Design: a deck of lenses*, que pode ser baixado via *Apple Store* e *GooglePlay*. Uso de Lens #9 *The Lens of The Elemental Tetrad.*
[4] "It's Only A Game" Game Design Methodology, Jim Wallman. **CLWG Fourth Annual Conference**, 1995.

- ✓ Limitações e possibilidades que plataforma traz para a ideia

Passo 8: Testar o Jogo

- ✓ Se estiver bom o suficiente, continuar produção normal
- ✓ Constatar novos problemas a serem resolvidos, ou novas soluções encontradas.

Passo 9: Receber *Feedback* de críticas em relação ao produto

Passo 10: Voltar para Passo 4 e ajeitar as escolhas propostas para se adequarem ao *feedback*.

Não se tem um número exato de quantos *Loops* serão necessários para que se chegue a uma ideia final, mas a ideia é que quanto maior o número de *loop* e considerações serem feitas durante o processo de tomada de decisão, menor serão os riscos de problemas ocorrerem durante a fase final de finalização do produto, pois conforme Schell (2015, p.94) a regra do *Loop* diz "quanto mais vezes você testar e melhorar o seu design, melhor o seu jogo será".

2 PROBLEMATIZAÇÃO

> "... entender verdadeiramente game design é saber entender uma complexa rede de criatividade, psicologia, arte, tecnologia e indústria".
> (Jesse Schell)

Durante o processo, nota-se que, apenas desenvolver um protótipo de jogo em 3D aleatório, envolve muito mais do que apenas o cenário, o universo do jogo, suas características, seus elementos principais, tudo isso influencia na criação do protótipo a ser desenvolvido.

Para tanto, é possível utilizar algumas lentes que ajudam a compreender e levantar questões necessárias sobre o desenvolvimento e produção do protótipo. Conforme a escolha de lentes[5] é preciso:

- Pense no seu jogo como solução a um problema.
- "Qual o problema, ou problemas, eu realmente estou tentando resolver? "

Saber compreender e reconhecer os problemas a serem resolvidos ajudam no fato de que muitas vezes se pensa constantemente nos detalhes internos na produção de um jogo, como estilo do personagem, *concepts*

[5] Disponível em aplicativo específico chamado *Game Design: a deck of lenses*, que pode ser baixado via *Apple Store* e *GooglePlay*. Uso de *Lens #14 The Lens of The Problem Statement.*

de elementos de cenário, história, e acaba esquecendo de uma coisa, com apoio de lentes[6], continua-se no seguinte raciocínio:

- Pare de pensar sobre o jogo e comece a pensar sobre a experiência que o jogador vai ter.
- "Quais experiências eu quero que o jogador tenha? "

Esse é um dos grandes desafios durante o *game design*: sentir ao mesmo tempo a experiência do jogo enquanto tenta-se entender quais os elementos e interação do mesmo estão causando essas experiências e o porquê.

Descobrir quais são os elementos chaves que realmente definem a experiência a ser criada e descobrir jeitos diferentes de torná-los parte do *game design*. Novamente, com o apoio de lentes[7] tem-se que:

- Olhar simultaneamente a estrutura do jogo e a experiência do jogador. É possível mudar o foco de um para o outro, mas é muito melhor vê o jogo e experenciá-lo holograficamente.
- "Como mudar elementos do jogo para melhorar a experiência do jogador? "

Seria possível citar diversas outras lentes, mas que independente de entender tudo, é necessário saber quais problemas a serem solucionados, quais emoções que se quer que o jogador sinta a partir das experiências que serão geradas.

Apesar disso, tudo engloba - em sua maioria - a parte conceitual na hora de criação do jogo, na prática a verdade é que muitas mudanças sofrerão efeito dessas perguntas e de muitas outras e é necessário se criar um produto, ou um ciclo de desenvolvimento, onde tem que se estar preparado para destruir e reconstruir os elementos criados, seja por efeito do marketing e do mercado, ou porque alguma peça simplesmente que fazia sentido no papel, não teve o mesmo efeito na prática.

Entender e prever esses riscos, conforme o disposto na Lente 16, e saber ir construindo o jogo dessa forma para evitar problemas sérios no

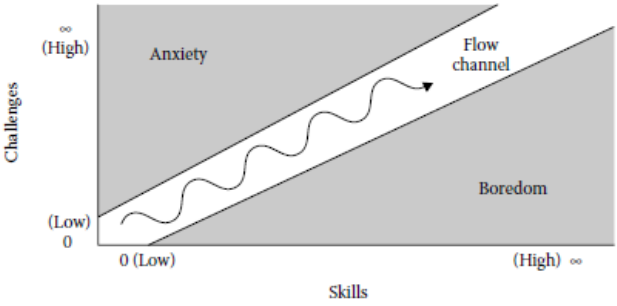
[6] Disponível em aplicativo específico chamado *Game Design: a deck of lenses*, que pode ser baixado via *Apple Store* e *GooglePlay*. Uso de *Lens #2 The Lens of Essential Experience.*
[7] Disponível em aplicativo específico chamado *Game Design: a deck of lenses*, que pode ser baixado via *Apple Store* e *GooglePlay*. Uso de *Lens #16 The Lens of Risk Mitigation.*

final do desenvolvimento, que vão comprometer a qualidade do produto final para o jogador.

Por essas razões que é necessário ter um objetivo claro, não ter distrações, ter um *feedback* direto e estar continuamente tentando desafiar o jogador a novas atividades e objetivos.

Por isso podemos usar esse exemplo de como se quer que um jogo se comporte na mente do seu jogador:

Figura 5 Gráfico Flow[8]



Fonte: SCHELL, 2015, p.141.

Significa criar uma experiência ao jogador onde é um ciclo repetitivo que sempre aumenta o desafio ao jogador, seguido por uma recompensa, de certa forma o jogador tende a ficar mais forte, o que torna o jogo talvez um pouco fácil e logo depois cresce de novo a dificuldade.

Esse ciclo acaba indo e voltando dentro do *design* de um jogo, muita tensão faz o jogador cansar, se o jogo é fácil de mais, torna-se um completo tédio, mas a oscilação entre eles é o que traz prazer ao jogador.

No final, entender como juntar todos esses elementos, saber posicioná-los dentro de um jogo, seja trabalhando com elementos físicos no cenário, ou características de *gameplay*, história do jogo etc., é essencial porque no final, assim como em dirigir um filme, ao criar o *design* e tentar apresentá-lo para o jogador é necessário saber como se comunicar e entregar as informações necessárias a eles de uma forma que fique claro e não haja erro no processo.

[8] Disponível em aplicativo específico chamado *Game Design: a deck of lenses*, que pode ser baixado via *Apple Store* e *GooglePlay*. Uso de *Lens #21 The Lens of Flow.*

Saber fazer essa transição entre jogo e jogador é a maior dificuldade para quem está começando a criar os primeiros projetos de jogos.

3 COLETAS E ANÁLISES

3.1 O que é um Jogo?

Definir o que é um jogo não é uma tarefa simples, existem inúmeras conceituações e nem todas são concordantes, porém, no geral há o consenso de acordo com Dille e Platten (2014) que afirmam que jogos são experiências compartilhadas e discutidas, que deixam uma impressão diferenciada de outros entretenimentos, já que o jogo usa o participante de forma ativa durante o processo, ou seja que seu objeto seja a diversão de seus usuários e que esta cause emoção e momentos de contentamento ou que seja visto como opção de lazer.

Já Schell (2015) conceitua que diferentemente da noção de trabalho, onde as pessoas exercem um ofício por questão de necessidade e/ou sobrevivência, jogar é algo que se faz de livre escolha, sem nenhuma obrigação (SCHELL, 2015). Sabendo que um jogo seria algo que surpreende e diverte um jogador, o que realmente significa então jogar? Seria algo tão simples quanto apenas uma espécie de exercício? Ou seria ter controle livre de movimento em um espaço criado dentro uma estrutura (o cenário do próprio jogo, por exemplo).

Não apenas isso, pois muitas vezes escolhe-se jogar em detrimento de outras atividades, pois de acordo com Schell (2015, p. 40) “Jogar é uma manipulação que aquiesce a curiosidade”, e já que essa ação atiça sensores de prazer e aguça nossa curiosidade, além de trabalhar com a imaginação em esferas nunca antes atingidas[9].

No final das contas, a definição de jogo pode ser apresentada de diferentes formas, entre diversos autores e livros e que acabam criando variadas soluções. Basicamente, um jogo é composto por:

- Objetivos;
- Conflitos;
- Regras;
- Derrota e Vitória;
- Interação;
- Desafios;

[9] Idem, uso de *Lens #6: The Lens of Curiosity*.

- História.

Todos são elementos que podem compor um jogo e que ajudam a chegar no objetivo final: que é criar uma experiência para o usuário, seja da forma que for; ao final todas essas expectativas atiçam nossa curiosidade para jogar, trazem uma nova surpresa e tem como objetivo final divertir o jogador. Decisões devem ser tomadas durante o desenrolar de um jogo, e essas devem seguir uma lógica de perguntas, tais como:

- Descobrir uma forma de ganhar do inimigo.
- Qual o caminho certo? Direita ou Esquerda?
- Como posso fazer mais pontos que meu adversário?
- Qual é o final dessa história?

Ao final, aparece algum elemento ou situação que cria um elo que liga todos esses pontos, independente dos objetivos e atividades dentro de um jogo; tudo se resume ao fato de estar-se apreciando o jogo pela diversão que é resolver os problemas dentro dele. No momento em que essa resolução de problemas for retirada do jogo, ele perde a sua função principal e acaba se tornando uma atividade qualquer e, nesse momento, o jogador perde o interesse e deixa o jogo para trás, e procura ir atrás de uma nova aventura para resolver[10].

Entendendo que jogo é uma atividade que envolve resolver problemas e achar soluções para desafios impostos pelo mesmo, e que no meio de todo esse processo, ele tem que surpreender, criar uma curiosidade e no final resultar em algo divertido para o jogador, uma definição que envolve todos esses pontos, pois conforme Schell (2015, p. 47), "Um jogo é uma atividade de soluções de problemas, abordada com uma atitude divertida". Assim como existem diferentes tipos de jogos, diferentes objetivos e propostas diferenciadas para atender os mais variados gostos, a definição do que é um jogo vai variar de jogador para jogador, cada um terá um foco em uma área que atraia mais atenção; consequentemente, o que cabe durante a criação e o desenvolvimento de um jogo é saber acertar onde serão ativadas essas áreas de interesse, prazer e do processo de curiosidade.

[10] Disponível em aplicativo específico chamado *Game Design: a deck of lenses*, que pode ser baixado via *Apple Store* e *GooglePlay*. Uso de *Lens #8: The Lens of Problem Solving.*

3.2 Tecnologia

> "A tecnologia que você escolhe para o seu jogo possibilita fazer certas coisas e proíbe de fazer outras coisas" (Jesse Schell).

Uma forma de entender a perspectiva por trás da tecnologia é conforme Schell (2015, p. 451) compreender a diferença entre *foundational* e *decorational technology.*

Foundational é a parte da tecnologia que está sempre em inovação e tentando criar novas maneiras de gerar experiências ao usuário. *Decorational* utiliza as tecnologias existentes e melhora a experiência de jogo.

Por exemplo, pode-se pensar que tecnologia fundacional são os consoles (*Xbox, Playstation, Nintendo*), onde geração após geração surgem novas plataformas que possibilitam a criação de novas experiências. Já a decorativa são formas inovadoras de aproveitar a tecnologia existente e apresentá-la de uma forma diferente para o jogador, como por exemplo as diferentes *Engines* que diferentes empresas usam para a produção dos jogos, mesmo que todos eles por exemplo rodem sempre em um Xbox e Playstation, são maneiras diferentes de se produzir e mostrar essa experiência, mas ainda usando a mesma tecnologia *foundational.*

Isso quer dizer que a tecnologia não está baseada apenas nos consoles ou aparelhos eletrônicos, já que um jogo de tabuleiro é também uma plataforma que tem como objetivo gerar uma experiência ao usuário, assim como uma mesa de xadrez, porque no final, é essa "fundação" que irá guiar o que pode ser feito e o que não pode ser feito dentro de um jogo.

Saber entender os limites da tecnologia é entender até onde sua ideia pode ser executada e quais os limites impostos a ela, evitando problemas futuros durante todo o processo de desenvolvimento de um jogo.

Nesse caso estuda-se mais sobre as tecnologias *decorational*, já que o objetivo desse trabalho é criar uma maneira diferente de produzir uma experiência ao jogador, mas não criar uma nova tecnologia fundamental.

Será utilizado como parte desse desenvolvimento a utilização de uma variedade de programas, mas como mais importante serão *3Ds Max*, da empresa *Autodesk*, *Photoshop*, da empresa *Adobe* e *Unreal 4*, da empresa *Epic Games*.

3.2.1 Modelagem e Textura de *Assets*

Essa é a primeira parte do desenvolvimento prático na criação de conteúdo diretamente ligado com o jogo, é agora que se utiliza dos *concepts* e *artworks* criados por artistas gráficos e começa a transformá-los no que se chama de *Assets*, objetos/personagens/animações, entre outros, para o desenvolvimento de um jogo.

O programa *3Ds Max,* desenvolvido pela Autodesk*,* é um desses *softwares* e será utilizado nesse projeto para exemplificar a escultura e criação desses *Assets*.

Tendo como exemplo a imagem a seguir, a casa foi totalmente modelada e criada dentro das possibilidades que o programa *3Ds Max* oferece:

Figura 6 Projeto feito pelo autor para a Escola Melies

Fonte: Autor, 2015.

Junto com o *3Ds Max*, também existe para complementar a criação de *Assets*, os programas *Zbrush* (Pixologic) e *Mudbox* (Autodesk). Ambos funcionam para complementar as funcionalidades do *3Ds Max* e suprir algumas necessidades, como por exemplo a criação de objetos mais orgânicos como personagens.

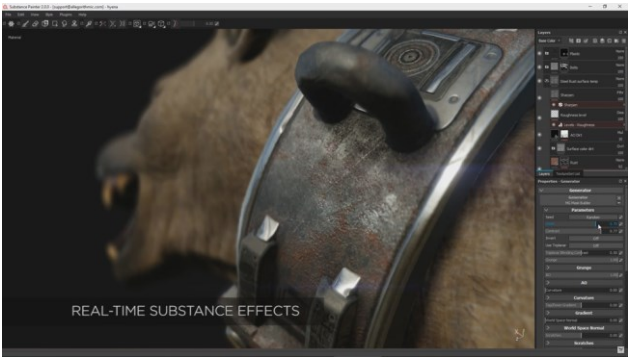
Figura 7 Modelo Personagem feito Zbrush

Fonte: http://www.zbrushcentral.com/ Usuário: Gilbertorbr

Mas o processo de criação de um *Asset* não passa só pela modelagem do mesmo, é necessário colocar o que se chama de textura, que é dar cor ao *Asset*. Nesse caso, se utiliza em sua maioria a ferramenta de *Photoshop* para a criação e edição de texturas.

Vale ressaltar que, apesar de muito utilizado o *Photoshop* para a edição das imagens e ajustes na textura, também são muito usados outros programas como *Substance Painter*, desenvolvido por *allegorithmic*, que ajuda a texturizar objetos e personagens com um maior controle sobre os detalhes da textura, de forma mais completa e pormenorizada do que as opções existentes dentro do *Photoshop*.

Figura 8 Programa Substance Painter



Fonte: http://www.digitalmediaworld.tv/

Com a criação de ambos (modelagem mais textura) é feito um primeiro teste dentro do *3Ds Max,* para se ter uma noção se a textura está funcionando antes de importar tudo para a *Unreal Engine 4*.

Figura 9 Projeto Portfólio pessoal Casa Abandonada

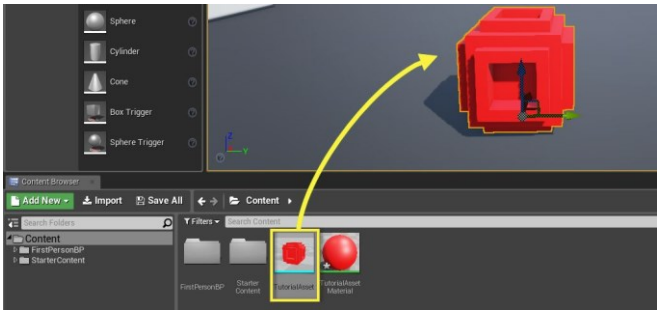
Fonte: Autor, 2016.

Após essa verificação, única coisa que falta é importar todos os *assets* desejados para dentro da ferramenta que realmente irá dar vida ao jogo, nesse caso o uso do *Unreal Engine 4*.

3.2.2 Programa *Unreal Engine 4*

Unreal Engine 4 é a ferramenta usada para juntar todos os Assets criados e será usada como a última plataforma de criação e desenvolvimento.

Figura 10 Exemplificação Assets na Unreal Engine 4



Fonte: https://docs.unrealengine.com

Trabalhar com uma *Engine* do tamanho que é a *Unreal 4* permite uma maior criatividade e diminui o nível de restrições sobre as funcionalidades do jogo.

O programa tem disponível um sistema de *Blueprint* que facilita a programação do jogo, mesmo para pessoas que não trabalham nessa área, permitindo assim a oportunidade de criar eventos e animações no jogo de uma forma menos complexa.

Além de permitir a importação de diversos modelos de programas já citados e de outros, não importando se seja um modelo de uma parede, ou um personagem junto com controles de animação, ou até a própria animação em si, é possível importar e colocar em outros objetos da cena.

Estudar e saber explorar os limites da *Engine* seja através da construção do protótipo de jogo ou das diferentes possibilidades de programação, influencia no que pode ou não ter no jogo.

Elementos que talvez funcionem de uma forma extremamente realista em uma imagem 3D, talvez em tempo real não seja possível de fazer.

Por esses motivos e outros se dá a escolha da *Unreal 4* para o projeto por se mostrar ser uma ferramenta harmoniosa e de alto nível, que inclusive é utilizada por diversas empresas e estúdios para

desenvolvimento de jogos e está disponível de graça pela empresa *Epic Games.*

Figura 11 Jogo desenvolvido na Unreal 4 – Paragon

Fonte: Epic Games[11]

3.3 Tema

> "O benefício primário de basear o seu design ao redor de um único tema é que todos os elementos do seu jogo irão reforçar uns aos outros, trabalhando em direção a um objetivo em comum" (Jesse Schell).

O tema é um dos primeiros passos no desenvolvimento de um jogo, sendo decidido até antes mesmo de dar origem a ideia do jogo (SCHELL, 2015). Considera-se que esse procedimento torna mais fácil decidir uma ideia de jogo sabendo qual o tema pelo qual deseja-se desenvolver.

Decidir o tema na fase inicial do desenvolvimento ajuda a criar delimitações em relação às ideias possíveis para o jogo e assim quando escolhido irá delimitar elementos, como personagens, objetos no cenário, estética etc. Para que haja um bom desenvolvimento, todos os elementos do jogo terão como objetivo enaltecer esse tema e trazê-lo para o jogador, permitindo assim uma maior imersão do mesmo com o jogo.

A escolha dos jogos *Left 4 Dead* e *Outlast* como referências para elaborar esse trabalho deve-se ao fato que ambos possuem elementos temáticos que conduzem o jogo.

[11] Imagem retirada do site oficial. Disponível em: http://epicgames.com Acesso em set – out 2016.

Usando a série *Left 4 Dead* como exemplo, o tema do jogo é bem simples, um cenário pós-apocalipse zumbi:

Figura 12 Jogo Left 4 Dead

Fonte: Valve[12]

A imagem apresenta um cenário caótico, com quatro personagens carregando armas pesadas e zumbis indo em direção a eles.

Nota-se que logo na abertura do jogo, nesse quadro aparecem todos os elementos que fornecem o tema principal do jogo, que é sobreviver em um universo composto de zumbis.

Usando o tema como delimitador de características, por exemplo não encontraremos em *Left 4 Dead* personagens como vampiros, ogros, ou qualquer tipo de elemento que desvie do tema central do jogo.

Também influencia a parte sonora do jogo, como por exemplo a música tema de *Mass Effect*, que é uma trilha sonora que tenta remeter ao fator de viagem espacial entre múltiplos locais no universo.

O tema também irá influenciar as possíveis formas de 'jogabilidade', jogos como *Outlast,* que é focado no gênero terror, houve a decisão de utilizar o jogo em primeira pessoa, porque seria a câmera que melhor coloca o jogador na situação do personagem, devido também a outras características do jogo como usar a câmera para documentar o que o personagem está vivenciando.

[12] Imagem retirada do site oficial da empresa desenvolvedora de software. Disponível em: http://www.valvesoftware.com/ Acesso em set. – out. 2016.

Figura 13 Jogo Outlast

Fonte: Red Barrels Studio[13]

No final, todos os elementos do jogo, sejam eles visuais, sonoros, ou como o jogador interage com o jogo irão dar suporte ao tema principal, criando um círculo de interação entre todos os elementos, onde um reforça o outro, assim pode-se considerar que o tema "(...) É a ideia que amarra o seu jogo inteiro – A ideia que todos os elementos precisam dar suporte" (SCHELL, 2015, p.59).

3.5 Mecânicas

De acordo com Hunicke (2001, p. 03), "Mecânicas são as diversas ações, comportamentos e controles instrumentais proporcionadas ao jogador dentro do contexto do jogo. Junto com o conteúdo do jogo, as mecânicas suportam a dinâmica do gameplay".

As mecânicas do jogo são o núcleo do jogo; são as interações e relações entre todos os outros componentes do jogo como *aesthetics*, *tecnology* e *story* quando nenhuma delas está mais presente.

Por exemplo, em um jogo de pôquer, as mecânicas do jogo incluem embaralhar as cartas, a opção de apostar ou não de acordo com a sua mão, e entre outros pequenos detalhes, e dessas mecânicas também surge a possibilidade de blefar vindo do próprio jogador.

Em um jogo como *Call of Duty* e *Battlefield,* as mecânicas envolvem as próprias armas do jogo, quantidade de munição que o personagem tem e pode carregar, lugares onde os personagens revivem

[13] Imagem retirada do jogo oficial da empresa desenvolvedora de videogame. Disponível em: <https://www.redbarrelsgames.com/> Acesso em set. – out. 2016.

depois de morrerem, e utilizando o cenário a seu favor eles criam estratégias como ficar escondido atrás de um arbusto esperando o momento certo de atirar.

Em *Rocket League*, diferentes tipos de arenas criam diferente tipos de mecânicas, com o *boost* do jogo é possível aumentar a velocidade máxima do carro, mas ao mesmo tempo pode ser usado para voar e acertar a bola em um ponto mais alto, a própria bola é uma mecânica do jogo pela forma com a qual se comporta, com efeito da gravidade ou não, se é redonda ou quadrada, entre outros. Isso muda a forma com a qual interage-se com os carros dentro do jogo.

Além disso, é possível adicionar outros elementos como tempo de jogo, ou um placar mantendo a pontuação de cada time para se criar um modo mais competitivo entre dois times, incluindo também a possibilidade de se andar pelas paredes o que permite diferentes movimentos ao jogador e novas possibilidades de jogada.

Todos esses são modelos simples de mecânicas de jogo, onde saber ajustar esses elementos faz com que se consiga melhorar o dinamismo do mesmo. Em todo o caso, deve-se manter em mente que no final, o jogo deverá estar balanceado dentro de suas características, porque um jogo balanceado estará dentro dos parâmetros impostos pela figura 5.

3.5.1 Espaço

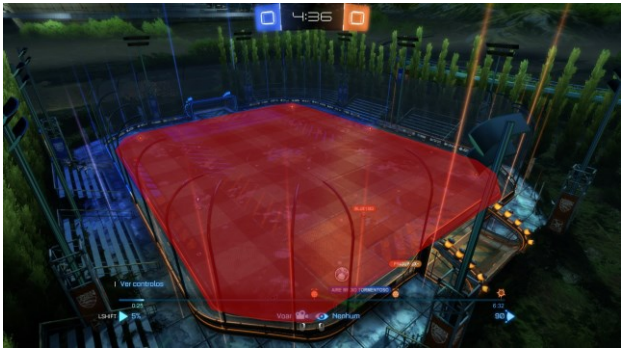
Todo jogo acontece dentro de um espaço, ele define os lugares dentro do jogo e como cada um desses locais se relacionam com os outros. Como uma mecânica de jogo é necessário retirar toda a estética e simplesmente analisá-lo como algo com linhas que formam o espaço e determinar quais os elementos estarão dentro.

Essas linhas normalmente são de acordo com Schell (2015, p. 159):

- Discretas ou contínuas;
- Funcionam dentro de dimensões 2D ou 3D;
- Tem áreas que funcionam como fronteira para outras áreas que podem ou não estar conectadas.

Usando as arenas do *Rocket League* como exemplo, pode-se reduzir a versão mais simples dela, em um formato retangular (marcado pela área vermelha), tal qual a imagem a seguir:

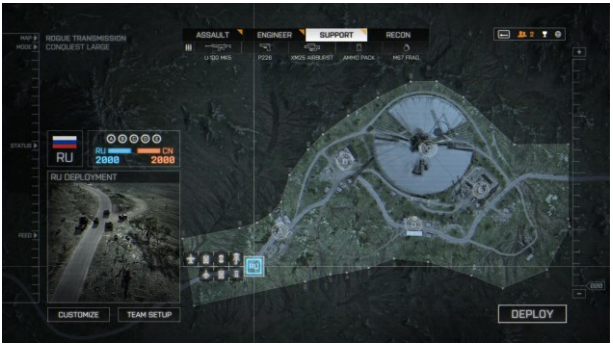
Figura 14 Arena de Rocket League



Fonte: Psyonix[14] (original modificado pelo Autor)

Na série *Battlefield*, essas linhas ficam mais evidentes porque o próprio estúdio do jogo delimita essas linhas no mapa, muito mais para delimitar o espaço em que os aviões podem voar e não saírem da zona de combate.

Figura 15 Mapa de um nível de Battlefield 4



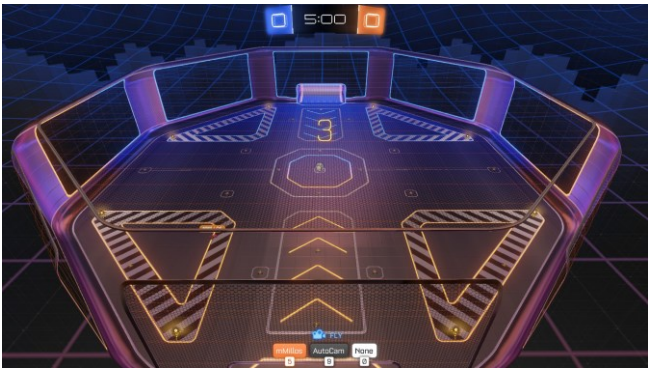
Fonte: Eletronic Arts[15]

[14] Imagem retirada do jogo oficial da empresa. Disponível em: <http://psyonix.com> Acesso em set – out 2016.
[15] Imagem retirada do jogo oficial da empresa. Disponível em: <http://www.ea.com> Acesso em set – out 2016.

Voltando para os exemplos de mapas de *Rocket League*, o estúdio se utilizou dessa habilidade de mudar as linhas que formam o campo para criar diferentes arenas que por consequência criavam todo um novo dinamismo para uma partida dentro do jogo.

O mapa que se destaca por essa diferença é o mapa em formato de um octógono, tal como a imagem a seguir:

Figura 16 Mapa Octógono de Rocket League



Fonte: Psyonix[16]

A simples mudança dessas linhas do mapa, criou todo um novo dinamismo para o jogo, do gênero: algumas bolas que batiam para a parede e iam para o gol, agora têm uma reação diferente ou andar pela parede ficou mais difícil por causa das pontas que foram colocadas etc., dessa forma o tamanho do mapa aumentou consideravelmente.

Por causa do *feedback* dos jogadores, o estúdio realizou um 'reparo' no mapa que foi tirar a ponta dos octógonos e deixá-los um pouco arredondados para facilitar o andar do carro pela parede, já que essa é uma das características mais importantes do jogo e o mapa estava dificultando esse tipo de *gameplay*.

Mudar só as linhas externas que formam o espaço não é o único método de modificar a dinâmica do jogo; em outro mapa customizado de *Rocket League*, além de diminuir o tamanho do mapa, o estúdio adicionou duas paredes extras no meio do campo, como se estivesse separando o

[16] Imagem retirada do jogo oficial da empresa. Disponível em: <http://psyonix.com> Acesso em set – out 2016.

mapa em três caminhos diferentes para se chegar ao gol, ao meio e nas laterais, conforme é possível verificar na imagem a seguir:

Figura 17 Mapa duas paredes de Rocket League

Fonte: Psyonix[17]

Muitos jogos têm seus espaços elaborados de forma mais complexa do que os exemplos citados de *Rocke League*, e esses acabam criando zonas dentro de linhas gerais que delimitam o mapa do jogo, por exemplo, no próprio mapa de *Battlefield* - já citado - existem "espaços dentro de espaços".

[17] Imagem retirada do jogo oficial da empresa. Disponível em: <http://psyonix.com> Acesso em set – out 2016.

Figura 18 Mapa Battlefield 4 Modificado

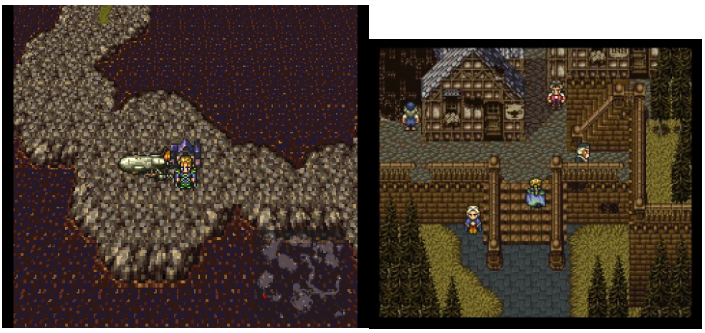
Fonte: Eletronic Arts[18]

É possível perceber que dentro do próprio mapa existem zonas (marcadas em vermelho) dentro da linha geral do mapa (marcado em amarelo). Mais adiante, nas zonas em vermelho - existem outras pequenas zonas dentro; dentro de um prédio é possível ver cada andar do prédio como uma pequena zona de *gameplay*.

Alguns jogos, separam esses espaços em zonas não conectadas, mas que são a continuação ou complementação do local de onde elas estavam, por exemplo jogos antigos de RPG usavam muito dessa técnica pelas limitações da tecnologia da época, em *Final Fantasy 6*, o personagem principal andava pelo *World Map* e quando precisava entrar em uma cidade ou uma caverna, ele acessava um ícone no mapa que o levava para essa zona, uma complementação do mapa mundial, mas que não está necessariamente conectada.

[18] Imagem retirada do jogo oficial da empresa. Disponível em: < http://www.ea.com> Acesso em set – out 2016.

Figura 19 Final Fantasy 6 Dentro e Fora da cidade



Fonte: Square Enix[19]

Certamente em jogos o espaço 2D não é o único a ser considerado na hora da produção de cada mapa. Após essa primeira construção começa-se a adicionar os elementos de 3D nesse espaço, como altura.

No Caso de *Rocket League*, as arenas são fechadas no seu topo para evitar que o carro saia voando para o infinito e acabe fora da arena proposta para a competição entre dois times. Já em *Battlefield*, ao sair da área limitadora, tanto pelas laterais, quanto voando, o jogador é punido e levado de volta para dentro dos parâmetros impostos pelo jogo. Outro bom exemplo está em *Mario Kart*, quando um jogador é derrubado para fora da pista delimitada pelo jogo, um personagem traz o jogador de volta para o ponto no qual ele caiu.

No final, uma vez que se consegue manipular o espaço abstrato de uma forma que encaixe no jogo, é que se começa a aplicar a parte de *aesthetics* nele. O ideal é fazer os dois ao mesmo tempo e nesse processo conseguir entender e prever a experiência que o jogador terá, como todos eles se relacionam; dessa forma, simplifica-se a tomada de decisões sobre o formato que o universo do jogo terá.

3.5.2 Tempo

Na vida real não é possível controlar o tempo da forma que se quer, tentar pausar ou voltar atrás, ir adiante para ver como ficou o resultado de alguma coisa, entre outras situações. Porém, dentro de um jogo, é possível dar esse controle ao jogador ou utilizá-lo como parte da mecânica do jogo.

[19] Imagem retirada do jogo oficial da empresa. Disponível em: < http://www.square-enix.com/> Acesso em set. – out. 2016.

Existem diversas formas de implementar o tempo como uma mecânica, diversos jogos de tabuleiros e RPG usam turnos para representar o tempo de jogo.

A maioria dos jogos, no entanto, utilizam um tempo contínuo, principalmente jogos de ação como *Battlefield,* assim como grande parte dos jogos de esportes. Mesmo assim é possível misturar ambos, o jogo de Xadrez por exemplo utiliza um sistema de turno entre os dois jogadores, mas cada um deles tem um tempo para poder realizar a ação.

Tempo também pode ser usado como um limitador, por exemplo a duração de uma partida de futebol é de noventa minutos e tudo deve acontecer dentro desse pequeno espaço de tempo, em outros casos é quase o contrário, em jogos de corrida o jogador quer terminar o mais rápido possível porque isso significa que ele vai ser o primeiro e, consequentemente, ganhar a partida.

Já alguns jogos trabalham com tempo como o núcleo do *gameplay* de seu jogo, o jogo *Braid* envolve controlar o próprio tempo com o personagem, voltando no tempo para fazer ajustes que ajudem o jogador a avançar no jogo.

No final, jogos criam a possibilidade de poder manipular o tempo de uma forma que não é possível na realidade, seja dentro do próprio *gameplay* ou nos momentos onde aperta-se o botão de pausa no controle; tempo pode parecer algo invisível e intangível, mas ele está sempre presente dentro de um jogo, conforme é possível verificar no aplicativo[20] de Lentes.

3.5.3 Objetos

O livro *The Art of Game Design A Book of Lenses* (SCHELL, 2015, p. 165) classifica os objetos como o equivalente aos substantivos das mecânicas de jogos. Significa que tudo que o jogador pode ver e manipular dentro do espaço proposto pelo jogo, cabem nessa categoria de objetos.

Em cada objeto inserido, todos vão conter de alguma forma algum tipo de atributo (*attribute*), que significa quais as informações que um objeto vai ter, nem que seja a sua posição no mapa.

[20] Disponível em aplicativo específico chamado *Game Design: a deck of lenses*, que pode ser baixado via *Apple Store* e *GooglePlay*. Uso *de Lens #27 The Lens of Time.*

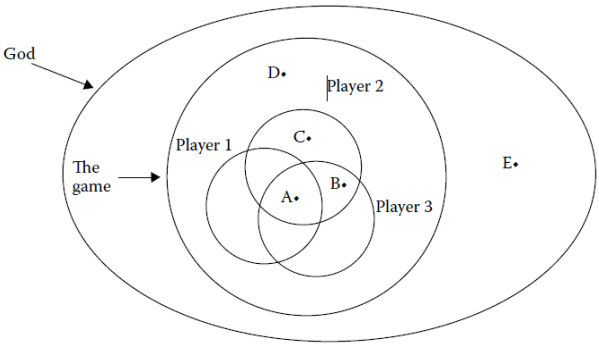
Por exemplo, dentro de um RPG uma espada pode ter um valor de ataque entre o mínimo (5) e o máximo (10), os valores são fixos, não mudam se o objeto não mudar, mesmo recebendo um *upgrade* por exemplo, mas o valor entre eles, este sim é constante porque depende de uma série de cálculos que irão mudar constantemente como o jogador joga, onde o mesmo sabe que um ataque de 5 pode não destruir o adversário, mas um ataque de 8 sim, mas não tem como ter certeza apenas com essas variantes, que irá derrotar o adversário com um ataque apenas.

Se objetos são substantivos, seus atributos (*atributes*) e suas condições (*states*) funcionam como adjetivos. Eles podem ser estáticos (como a cor de um objeto), nunca mudando durante todo o jogo, como também pode ser dinâmico, como por exemplo a Rainha de um jogo de xadrez que tem a habilidade de quase todas as outras peças.

Esse design do sistema de cada função de objetos é uma teia de variáveis e elementos que é possível tornar tudo confuso tanto para quem está desenvolvendo o jogo quanto quem está jogando o mesmo. Por isso que escolher exatamente como e o quê mostrar ao jogador é essencial para a experiência dele durante o jogo para não o sobrecarregar.

Por essa razão às vezes é necessário criar um sistema de “segredo” tanto por questões de *gameplay*, quanto para não exagerar na quantidade de informação. Logo, decidir o que é informação pública para todos os jogadores e/ou só para alguns deles é fundamental para o andamento do jogo. Para isso é necessário criar pontos de vistas de dentro do jogo e considerar o próprio jogo como uma dessas visões, porque dependendo da visão e o objetivo de cada jogo, cada um terá a informação necessária apenas e não a imagem do quadro inteiro.

Figura 20 Exemplificação da relação objeto-jogador



Fonte: Schell, p. 169.

A imagem acima representa aproximadamente os pontos de vista de cada jogador em um jogo, obviamente nem todos os jogos seguirão o mesmo modelo, cada jogo irá adaptá-lo conforme as suas necessidades.

- A: É a informação pública que todos terão acesso durante o jogo.
- B: Corresponde à informação que os jogadores compartilham entre si, mas não quer dizer que todos os jogadores têm essa informação.
- C: É a informação que apenas o próprio jogador tem conhecimento.
- D: É a informação que o jogo sabe que existe, mas os jogadores não, ou ainda não descobriram.
- E: Apenas informações geradas aleatoriamente devido a possíveis atos aleatórios que apenas "Deus" tem conhecimento.

3.5.4 Ações

Pode-se considerar que ações dentro do jogo é o equivalente ao verbo para a linguística, ou seja, "Um jogo sem ação é como uma frase

sem verbo – nada acontece”[21], ou seja dentro dos ensinamento e teorias sobre lentes, essa é a mais específica sobre a ação em si e isso exemplifica tudo que o seu jogador pode fazer dentro do seu jogo. Em *Rocket League*, seria acelerar, frear, usar o turbo, todos esses funcionariam como ações básicas do jogo. Outro jeito seria usar essas ferramentas de uma forma mais estratégica, como usar o turbo para voar e em qual direção, tentando acertar a bola em certo ponto no ar, utilizando essas ações básicas a seu favor para chegar no objetivo, acertar a bola, entre outros.

Essas ações estratégicas geradas se chamam de *emergency gameplay* (jogabilidade emergente), ações que emergem naturalmente quando se joga, muitas vezes pela criatividade do jogador ou por planejamento do *designer* para uma maior complexidade no *gameplay*.

Existem diversas formas de se tentar produzir esse *gameplay* emergente, como adicionar mais “verbos”, ou seja, novas ações que possibilitem interagir com as já existentes, criando novas possibilidades de ações. O maior problema desse tipo de pensamento é que isso pode ocasionar uma sobrecarga ao jogador pelas diversas mecânicas que o jogo acaba tendo, ou essas novas mecânicas não se integram bem entre elas, criando algo confuso para o jogador.

Outra maneira de pensar é fazer que as ações existentes possam ser aplicadas de forma diferente com objetos diferentes na cena, exemplo disso são as duas paredes no mapa citado na figura 17, esses dois novos objetos na cena adicionam diferentes formas de se mover com o carro e com a própria bola do jogo para criar situações de possibilidade de gol.

Esse exemplo também complementa a ideia de criar novas possibilidades para se chegar ao objetivo final, se a maneira melhor e mais fácil de se fazer gol fosse sempre de uma forma só, todo mundo abusaria desse sistema para poder ganhar a partida, adicionando esses novos objetos e novos “verbos” (turbo, voar etc.) já citados, torna o jogo mais difícil de se balancear, mas ao mesmo tempo cria novas opções ao jogador.

No final, cada diferente tipo de gênero de jogo terá suas metodologias que funcionam ou não para melhorar a ação do jogador, e essa é uma das dificuldades: saber limitar o jogador no que ele pode e não pode fazer e interagir.

Por isso, alguns jogos se parecem muito iguais a outros e são considerados derivativos, porque suas ações são semelhantes; essa função

[21] Disponível em aplicativo específico chamado *Game Design: a deck of lenses*, que pode ser baixado via *Apple Store* e *GooglePlay*. Uso de *Lens #31 The Lens of Action.*

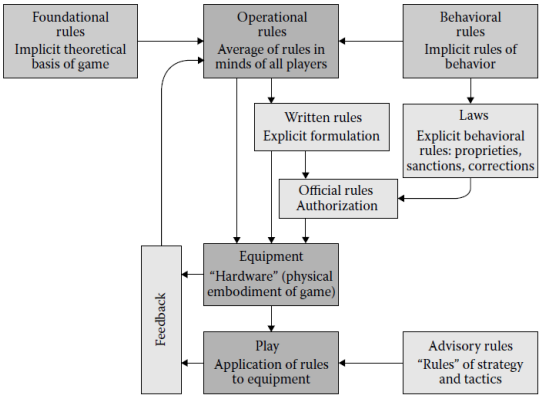
do jogo é tão importante que uma simples mudança, pode ocasionar em um efeito dominó que resultará em um jogo completamente diferente.

3.5.5 Regras

Regras são a ‘cola’ que gruda e molda todas as mecânicas citadas e muitos outros detalhes dentro de um jogo, realmente engloba os elementos e os transformam em um verdadeiro jogo.

Para analisar melhor como funciona o sistema de regras David Parlett, um historiador de jogos, criou um sistema com diferentes tipos de regras que estão envolvidas dentro da mecânica de um jogo (SHELL, 2015).

Figura 21 Sistema de Regras por David Parlett



Fonte: Schell, 2015, p. 175.

Regra Operacionais: É simplesmente a regra base que significa “O que os jogadores fazem para jogarem o jogo”, quando um jogador entende essas regras básicas, ele consegue começar a jogar.

Regras Fundacionais: Essas regras, são as regras matemáticas que criam a fundação para as regras operacionais.

Regras Comportamentais: São regras que estão implícitas no *gameplay*, o que é conhecido no dia a dia como espírito esportivo, assim como *foundational rules*, elas também se remetem às *regras operacionais*.

Regras Escritas: É exatamente o livro de regras de um jogo, um documento que os jogadores recebem com uma lista de orientações do jogo, seja do que pode fazer ou o que não pode fazer. Apesar disso, jogos de hoje tem evitado esse tipo de forma de ensinar os jogadores a jogarem, porque na maioria dos casos, as pessoas preferem ver e sentir do que apenas ler em palavras o jogo.

Leis: São as regras compostas para ambientes mais competitivos, onde a premiação é de valor e precisa-se deixar bem explicito as regras do torneio. A maioria tem como propósito deixar claro quais as configurações do jogo serão usadas para o campeonato.

Regras Oficiais: São criadas quando se encontra uma necessidade de unir *wrriten rules* com *laws*. Ao longo do tempo, essas *official rules* acabam virando *written rules*.

Regras sugeridas: Também chamadas de *strategy rules (regras estratégicas),* são apenas dicas ditas pelos jogos ou pela comunidade do jogo para ajudar a pessoa a entender e jogar melhor o jogo.

Claro que para cada jogo podem existir diversos tipos de regras, durante diferentes partes do jogo. Regras frequentemente mudam de acordo com o modo de jogo que está a ser jogado.

No final, a verdade é que jogos são compostos por múltiplos tipos de regras, mas tem algo que é a fundação de todas elas: o objetivo final do jogo. Pode ser que seja uma sequência de objetivos, e para cada um desses objetivos, um conjunto de regras será necessário, porque se o objetivo do jogo não for claro, por consequência as regras podem se tornar confusas e o jogador ficar perdido durante o jogo.

Porque o objetivo final é o grande motivador que faz qualquer pessoa querer jogar o jogo até o final, querer viver essa experiência, seja pelo objetivo final da história, por uma partida de futebol ou pela exploração, todos esses e muitos outros fatores se resumem à ambição e motivação do jogador de completar o jogo e sem as regras para delimitar como é possível chegar no final, um jogo simplesmente não é um jogo.

3.4 Estética

> "Aesthetics: Demonstra como o seu jogo parece, se escuta, cheira, saboreia e se sente...já que eles têm a relação mais direta com a experiência do jogador"
> (Jesse Schell)

Assim como diz MDA, essa é a última etapa e a que tem o maior contato com o jogador, logo é aqui que todas as ideias de design,

conceitos, abstrações, começam a se transformar de fato em algo concreto, é aqui que o visual do seu jogo começa a ganhar corpo, o áudio ganha sua vida, e as primeiras emoções e experiências[22] começam a virar algo mais concreto.

Para ajudar a caracterizar melhor, o MDA separa uma pequena lista de adjetivos que podem ser utilizados para o jogo, mas não necessariamente exclusivo a apenas eles:

- ✓ *Sensation* (Sensação);
- ✓ *Fantasy* (Fantasia);
- ✓ *Narrative* (Narrativa);
- ✓ *Challenge* (Desafios);
- ✓ *Fellowship* (Sociedade);
- ✓ *Discovery* (Descobrimento);
- ✓ *Expression* (Expressão),

Cada jogo, busca diversos tipos de estéticas diferentes, alguns dão foco em um mais que o outro, outrem não levam em conta, o importante é saber fazer a combinação e a proporção necessária de cada elemento para que se crie uma boa experiência para o usuário.

Muitas vezes pensa-se que o visual do jogo é apenas uma pequena camada por cima das outras partes do jogo, como história, mecânica e esquecem que um jogo é uma experiência completa, todos os elementos são equivalentes e por isso o visual de um jogo pode chamar a atenção do jogador pelo belo visual, conforme aplicativo[23].

Crysis, lançado em 2007, era aclamado pelos seus gráficos e um grande motivo pelo qual fez sucesso foi pela sua aparência, tanto que o jogo lançado na sua época, raros computadores conseguiam rodá-lo em seu potencial máximo.

[22] Disponível em aplicativo específico chamado *Game Design: a deck of lenses*, que pode ser baixado via *Apple Store* e *GooglePlay*. Uso de *Lens #1 The Lens of Emotion e Lens #2 The Lens of Essential Experience.*

[23] Idem, uso de *Lens #71 Beauty*.

Figura 22 Cenário jogo Crysis

Fonte: Eletronic Arts.[24]

A forma pela qual o jogo foi composto, acabou criando uma interação entre todos os elementos que o fizeram tornar-se um jogo mais apelativo visualmente, principalmente comparando a qualidade dos gráficos do jogo com a própria realidade, em comparação aos outros.

Levando isso em conta, considera-se que elementos visuais também vão ser usados para reforçar o tema do jogo.

Figura 23 Imagem publicitaria Battlefield 1

Fonte: Eletronic Arts.[25]

[24] Imagem retirada do jogo oficial da empresa. Disponível em: <http://www.ea.com> Acesso em set – out 2016.

[25] Imagem retirada do jogo oficial da empresa. Disponível em: <http://www.ea.com> Acesso em set – out 2016.

Nota-se que o estúdio *Dice*, utilizou de ferramenta visuais para reforçar fortemente os elementos da Primeira Guerra Mundial dentro do novo jogo da série *Battlefield*, clássicos aviões, armas e cenários da época incorporam parte do jogo. Fazendo com que esses elementos visuais aumentem a expectativa de uma experiência emocionante para o jogo.

Alguns jogos como *Journey*, do estúdio *Thegamecompany*, utilizam fortemente elementos visuais como alto fator para a narrativa do seu jogo.

Figura 24 Jogo Journey

Fonte: Thegamecompany[26].

Onde o jogo anda sozinho e é rodeado por um vasto deserto com dunas, ruínas antigas, cavernas e com fortes ventos onde o objetivo é chegar no topo da montanha vista no horizonte. Mas a parte mais importante é a experiência pela qual o jogo quer que você sinta ao descobrir quem você é, o que o lugar significa e qual o propósito de tudo.

Por isso que nessa parte, ter um bom material de *artwork* para a criação do jogo é essencial por alguns motivos como: deixar a ideia clara para as outras pessoas - do que estiverem construindo o jogo, saberem expressar as emoções dentro do jogo e replicarem da melhor maneira para o usuário.

O áudio pode e é uma ferramenta extremamente importante para criar um maior estímulo ao jogador e fazê-lo imergir totalmente em seu jogo, como em *Battlefield 1*, exemplo citado pelo seu potencial visual

[26] Imagem retirada do jogo oficial da empresa. Disponível em: <http://www.thatgamecompany.com> Acesso em set – out 2016.

remetente à Primeira Guerra Mundial, o que seria desse jogo se o jogador enquanto tivesse jogando, não escutasse os tiros da sua própria arma?

Pessoas que não trabalham na área e/ou usuários domésticos pressupõem que o áudio do jogo vem apenas depois de toda a sua criação, e isso está errado, assim como na animação, o áudio do jogo em sua maioria já começa a ser criado na hora da conceptualização do jogo, porque com o áudio é possível começar a sentir como a ideia começa a se expressar e isso faz com que várias decisões de seu subconsciente sobre como o jogo deve se sentir, ou seja, sua atmosfera começa a tomar um corpo.

O poder do áudio do áudio é tão forte que mesmo quando *Final Fantasy 6* foi lançado para SNES e a capacidade de áudio era extremamente limitada, *Nobuo Uematsu*, um dos maiores compositores de músicas para a série *Final Fantasy*, conseguiu colocar o som de uma orquestra dentro de um sistema limitado que era SNES e dentro do jogo possibilitou uma das cenas mais marcantes da historia

Figura 25 Cena da Opera de Final Fantasy 6

Fonte: Square Enix[27]

Final Fantasy 6 contém uma cena onde a personagem Celes deve cantar na ópera e apesar de ser impossível colocar a voz da personagem dentro do sistema, conseguiram através das notas da música replicar o

[27] Imagem retirada do jogo oficial da empresa. Disponível em: < http://www.square-enix.com/> Acesso em set. – out. 2016.

mais perto possível disso e gerar uma experiência e emoção ao jogador nesse momento do jogo.

Esse é um ótimo exemplo onde tanto a parte visual, composição de cenário e história conseguem coexistir em uma mesma atmosfera criando uma excelente experiência. Por isso que com o som, se elemento ou cenário do jogo parece estar entrando em conflito com o áudio do jogo, e se entende que o áudio do jogo é o ideal para a ideia do jogo, é uma excelente indicação de que esses elementos do jogo devem ser adaptados.

Por esses motivos que essa é a parte que tem um maior contato com o jogador, é nessa área que realmente se juntam todos os elementos, tanto visuais como sonoros, para se criar uma atmosfera ao jogo, essa atmosfera é tanto invisível como intangível, mas de alguma forma irá acolher o jogador e transformá-lo como parte desse mundo fictício[28].

3.5 Material de Referência

Em todo processo de desenvolvimento de uma nova ideia, principalmente durante a parte de *brainstorm*, é necessário uma ampla pesquisa de materiais que possam servir como referência para a ideia a ser criada.

Por exemplo, os jogos como *Assasin's Creed*, têm referências de locais que realmente existem ou existiram no mundo, criados dentro do jogo de uma forma realista para a exploração do jogador:

Figura 26 Notre Dame em Assasin's Creed Unity

Fonte: Ubisoft[29]

[28] Disponível em aplicativo específico chamado *Game Design: a deck of lenses*, que pode ser baixado via *Apple Store* e *GooglePlay*. Uso de *Lens #94 - The Lens of Atmosphere.*

[29] Imagem retirada do jogo oficial da empresa. Disponível em: < https://www.ubisoft.com/pt-br/> Acesso em set. – out. 2016.

Figura 27 Notre Dame – Paris

Fonte: Autor, 2013.

Não apenas em cenários, mas muitos personagens são baseados em perfis de atores reais como por exemplo Miranda de *Mass Effect* 2 e 3 é modelado e criado baseando-se na atriz Yvonne Strahosvki.

Figura 28 Comparação entre Ator (esquerda) e Personagem (direita)

Fonte: BuzzFeed e Bioware[30]

Estilos visuais de jogos também são influenciados muitas vezes por diferentes estilos de artes que foram criados e marcaram a história do ser humano. A série *Bioshock* é um material de referência sobre esse assunto,

[30] Imagens retiradas do site. Disponível em: <https://www.buzzfeed.com/themetalcat/the-real-life-actors-behind-mass-effect-characters-12spw?utm_term=.uq1950pVL9#.njZxQ9XzAx> Acesso em set – out 2016.

sendo que *Bioshock* 1 e 2 tem uma influência da *Art Deco*, conforme dito por Pete Worth (2013) em seu artigo sobre o assunto.

Figura 29 Cenário de Bioshock

Fonte: Irrational Games[31]

Ken Levine, um dos cofundadores de *Irrational Games* e a liderança da criação da série *Bioshock*, cita em uma entrevista[32] ao jornal *NY Times*, que a inspiração para o jogo foram brinquedos do parque de diversão da Disney como "The Haunted Mansion" e "Pirates of the Caribbean", bem como experiências para os protótipos que eles vieram a fazer. Nessa mesma entrevista ele cita o livro "The Devil in the White City", de Erik Larson, como referência inicial para decidir o período em que os jogos se passariam.

A série *God Of War* é outro exemplo de um jogo altamente baseado em referências, durante a série inteira, todos os cenários são inspirados na mitologia grega e luta-se contra deuses da mesma mitologia em sua história, inclusive o novo jogo da série se passará no universo da mitologia nórdica e isso influencia a conceptualização do cenário e do próprio personagem e suas características, inclusive essas referências servem para sustentar o tema da série.

[31] Imagem retirada do jogo oficial da empresa. Disponível em: <http://www.thunderboltgames.com/feature/bioshock-the-art-deco-design-of-rapture> Acesso em set – out 2016.
[32] Entrevista dada em 2013, acessível no site.

Figura 30 Concept Arte Cenário *God Of War*

Fonte: God of War[33]

Fazer pesquisa de referências para os objetos, personagens, cenários, estilo etc., é parte crucial do processo de desenvolvimento, porque é desse passo que ficará mais claro para qual direção o desenvolvimento do jogo está indo, e ajudará a criar mais delimitações e possibilidades do que se pode fazer e o que não se pode fazer. Esse passo, assim como os outros, requer uma intensidade grande do *Loop*, até que se encontre a experiência certa e baseada nela consiga-se desenvolver *concept arts* e continuar o desenvolvimento do jogo.

Com essa ideia em mente foi feito uma pesquisa e algumas imagens surgiram como inspiração para o projeto a ser desenvolvido nesse PCC, levando-se em conta que assim como o processo inteiro, a parte de pesquisa estará sempre aberta a novas opções e não apenas delimitadas as opções mostradas a seguir.

[33] Imagem retirada do site oficial da empresa. Disponível em: <http://godofwar.playstation.com/> Acesso em set.- out. 2016.

Figura 31 Referência de Ilha Maldivas

Fonte: Ilhas Maldivas[34]

Inspiração de uma ilha para servir como referência no desenvolvimento do cenário

Figura 32 Referência Ilha temporal

Fonte: https://wallpaperscraft.com/image/sea_coast_excitement_cloudy_clouds_bad_weather_foam_52140_1920x1080.jpg

Esse exemplo serve como inspiração para o clima a qual o cenário terá, esse clima mais escuro, uma pré-tempestade, como mar reage nesse tipo de tempo e o céu reflete essa mudança de humor do cenário.

Na imagem, a seguir, tirada do jogo *Uncharted* 4, *Drake*, o personagem principal do jogo está em uma ilha logo após o acontecimento de uma tempestade.

[34] Imagem retirada do site. Disponível em: <http://antesdemorrer.com.br/wp-content/uploads/2014/11/ilhas_maldivas.jpg> Acesso em set.- out. 2016.

Figura 33 Cenário do jogo Uncharted 4

Fonte: Sony[35]

Ainda usando *Uncharted 4* como referência, essa é uma imagem de uma das ilhas do jogo, serve como inspiração para o relevo do mapa e seu estilo de vegetação.

Figura 34 Cenário ilha do jogo Uncharted 4

Fonte: Sony[36]

Rocket League é a inspiração por ser um jogo atual, onde jogadores usam carros em uma partida de futebol em vez de humanos e assim como em uma partida de futebol normal, o objetivo é fazer o maior número de gols e ganhar a partida.

[35] Imagem retirada do jogo oficial da empresa. Disponível em: <http://www.sony.com.> Acesso em set – out 2016.
[36] Idem.

Figura 35 Arena de Rocket League

Fonte: Psyonix[37]

Monster Truck Madness 2 serve como inspiração pelo seu modo de jogo onde o objetivo é se manter no topo do cenário o maior tempo possível, o ganhador é a pessoa que fizer a maior pontuação por ficar nesse tempo.

Figura 36 Capa do Monster Truck Madness 2

Fonte: Microsoft[38]

Durante o processo inteiro da criação do cenário para o projeto, inúmeras referências fotográficas, de filmes, livros e afins serão utilizados para o desenvolvimento e criação dos *Assets* que irão compor o cenário

[37] Imagem retirada do jogo oficial da empresa. Disponível em: <http://psyonix.com> Acesso em set – out 2016.
[38] Imagem retirada do site oficial da empresa. Disponível em: <http://microsft.com> Acesso em set – out 2016.

até a sua finalização, os citados acima são apenas alguns dos exemplos de referências que serviram de inspiração tanto para a ideia conceitual desse protótipo, quanto para a futura prática do mesmo.

Apesar das diferenças de estruturas de jogos e exemplos díspares aqui apresentados, tal explanação se faz necessária para a retirada de determinados elementos (estética, mecânica e história) que irão servir de referência ao protótipo em questão.

3.6 *Level Design*

> "Frequentemente as ideias mais inovativas não vêm de ir além dos limites convencionais, mas sim olhando de uma maneira diferente, ou utilizando-o para fazer algo completamente inesperado" (Dille & Platten)

O *Level designer* é a função que faz o papel de projetar os diferentes cenários que compõem o universo para o jogo. Colocar inimigos em alguma posição, ou um item coletável em cima de uma mesa, munição para o jogador seguir adiante, até criar ativadores de eventos, como o botão de um elevador, ou a criação de um obstáculo natural a ser vencido como uma tempestade de neve que influencia na forma com que o jogador desenvolva aquela parte.

Todos esses e muitos outros são elementos que compõem um possível cenário para um jogo, o cenário que irá ampliar a importância do *gameplay* e complementá-lo de certa forma.

Assim, o autor Schell (2015, p. 368-371) faz a mesma comparação, *Level designers* e arquitetos são como primos, nenhum dos dois cria uma experiência diretamente, mas se usam de controle indireto para guiar as pessoas para terem essa experiência.

Por isso que é importante entender qual a experiência que o jogo tem como objetivo para poder montar um espaço de acordo com ele, pode-se começar com alguns princípios básicos como:

- ✓ Linear: O jogo tem como característica sempre fazer o jogador ir adiante e nunca ficar dando voltas no cenário, o cenário é composto de uma forma que o jogador sempre segue para a frente. Exemplo disso são os antigos jogos de Mario do SNES como o *Super Mario World*, todos os cenários são da esquerda para a direita em sua maioria. Nem sempre isso é bom, *Final Fantasy XIII* é um RPG com um design extremamente linear, e o jogo foi extremamente criticado, porque não combina com o

gênero RPG um jogo linear que constringe o movimento do jogador.

- ✓ Grid: Esse tipo de formato é muitas vezes usado em design de jogos de tabuleiros, como por exemplo o Xadrez, ele é um formato fácil do jogador entender e mantem tudo proporcional. A série *Fire Emblem* é um exemplo de jogo que usa essa Grid para suplementar todo o cenário e seu *gameplay* estratégico. Mas não é só usado para jogos de tabuleiro, os primeiros jogos de *Zelda* no NES, os cenários eram separados em pequenas zonas (quadros) e se juntasse todos um do lado do outro, se teria esse formato Grid.
- ✓ *Web:* Significa formar o jogo que permita ao jogador chegar a seu objetivo de diferentes formas/caminhos; seja por diferentes zonas ou por uma forma diferente de se jogar. *Deus Ex* por exemplo, utiliza o cenário com suas mecânicas de jogo para dar a opção ao jogador de chegar ao seu objetivo sem matar ninguém e sem chamar a atenção de nenhum inimigo, ou de fazer exatamente o contrário, destruindo tudo e fazendo o maior número possível de barulho.
- ✓ *Divided Space:* Isso funciona quando o jogo tem como objetivo a tentativa de replicar um mapa real. Funciona de uma forma que se corta esse mapa em zonas, mas que estão sempre conectadas diretamente. *Zelda Ocarina of Time* usa esse sistema para tentar criar o universo na época, todas os cenários têm sua própria zona, mas todos ao mesmo tempo estão interligados diretamente uns aos outros.

Leve-se em conta de que isso são apenas alguns dos princípios básicos, e nem todo o jogo é preto/branco e escolhe apenas um método, muitos dos próprios jogos citados acima têm uma combinação entre dois ou mais estilos na composição de cenários e do próprio jogo em si.

Por isso durante a composição do cenário, é importante saber lidar com cada elemento da cena, porque um bom design de cenário irá causar algumas reações. Fará o jogador sentir que o cenário está vivo, completo, confortável, liberto, eterno e o mais importante, um bom design nunca passará a sensação de que existe conflito interno com contradições.

Se um jogo por natureza tem como ser uma boa experiência e divertido, um design que faz exatamente ao contrário é uma contradição interna, compor um cenário é saber lidar com essas contradições e não

arranjar desculpas para encaixá-las na cena, ou forçar a fazer com que ela se torne algo natural.

Com isso em mente para chegar a esse objetivo é necessária a criação de diversas interações e observações das mesmas de como os objetos e o cenário estão se relacionando com o jogador, para poder chegar-se ao design final, ou seja, a aplicação da regra de *Loop* na composição de cenário.

Conforme é possível perceber através da citação de Shell (2015, p. 372) tem-se o contato com Christopher Alexander, arquiteto que devotou a sua vida a estudar como os espaços se relacionam com a pessoa e que criou alguns padrões que podem ser utilizados na composição do cenário para jogos.

Assim temos, *Strong Centers* que é chamar a atenção do jogador para um elemento da cena, como por exemplo um *Landmark*, fazer com que todos os objetos da cena seja diretamente ou indiretamente remetam-se a esse objeto central. Não funciona apenas como composição de cenário, pode ser utilizado também na parte de história, colocando o personagem como o centro do universo e também para dá ênfase ao objetivo final do jogo.

Depois há a *Boundaries* que remete muito à limitação do espaço que o jogador tem acesso citado no capitulo de 3.5.1, que é criar essas regras de onde o jogador pode ou não pode ir. A seguir, há *Alternating Repetition* que é criar um ciclo que se alterna dentro da sequência de cenários, diretamente ligado com a figura 5 no capitulo 2, onde envolve criar esse ciclo para o jogador, como uma sequência do tipo aventura - boss - aventura - *boss*.

Seguindo o raciocínio, passa-se para o *Positive Space* que é saber misturar os elementos tanto de primeiro plano, quanto de segundo plano, de uma forma que se complementem e criem um ambiente bonito ao jogador, exemplo disso é a figura 35 de *Rocket League* que usa o estádio como primeiro plano, e os outros elementos do cenário como segundo plano a ponto de se misturarem criando algo completo e bonito.

Com *Good Shape* cria-se um formato do cenário que seja agradável. Mais perceptível quando visto pela parte dos elementos visuais do cenário, mas dá para ver e sentir quando algo aprece fora de lugar ou de proporção. Um bom cenário será algo sólido e com uma boa nuança.

Junto vem o *Local Symmetries* que não está relacionado à simetria do cenário como um todo, mas sim visto do ponto mais específico a objetos na cena, de uma forma que eles construído e posicionados passam a sensação de algo orgânico.

Por conseguinte, surge *Deep interlock and ambiguity* que é saber quando conectar dois objetos de uma forma que um é definido pelo outro, de uma forma que caso um deles esteja faltando, o outro perde o seu sentido. Esse tipo de interação pode ser utilizado para aprimorar o componente histórico do local, quando uma parte da arquitetura estiver faltando, seja por motivo de destruição ou qualquer outro, ou quando claramente esses dois objetos juntos dão um significado para a cena, ou um contexto ao personagem.

Dando andamento ao raciocínio, tem-se o *Contrast* que é a composição de cenário, contraste não envolve apenas o uso de cores, seja no cenário ou nos objetos que compõem o mesmo. Contraste existe entre os personagens, com suas personalidades por exemplo, nas recompensas e punições. Dentro do cenário, também é possível utilizar como um método de contar a diferença do local, como uma parte da cidade que faz a transição entre a zona rica e a zona pobre.

No padrão *Gadrients* demonstra-se a curva de desafio do cenário, por exemplo, cenários no início do jogo dentem a ser tecnicamente mais fáceis que os encontrados na última parte do jogo, saber fazer essa transição do mais fácil para o mais difícil é elemento importante durante a criação do cenário.

A seguir, tem-se *Roughness* que é quando um jogo é composto de uma forma muito perfeita, acaba tirando a imersão do jogador, pelo fato de que parece algo muito artificial. Às vezes adiciona-se um toque caótico ao cenário, tirando ele um pouco da ordem, cria-se um dinamismo mais natural que o jogador já está acostumado na realidade.

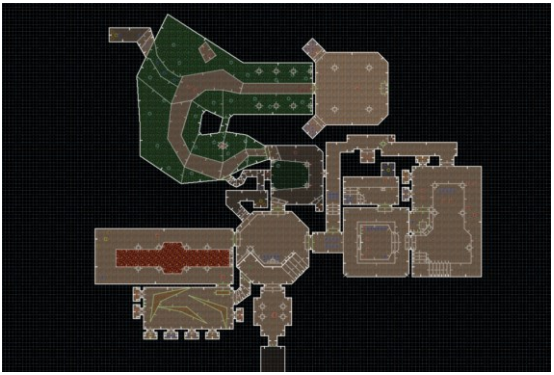
Para finalizar, há *Echoes* que é saber ligar os elementos do cenário com outras mecânicas, o que faz com que se crie um eco entre elas, uma conexão. Quando um *boss* é um pássaro gigante por exemplo, e seus inimigos mais fracos são pequenos pássaros, forma-se um eco entre ambos. E, *Not-separateness* é quando um objeto, ou algo da cena, realmente está conectado ao ambiente criado, como se fosse parte dele. Todas as regras do jogo terão essa propriedade por exemplo, assim como os elementos que compõem o jogo e, consequentemente, o cenário.

Nota-se que esses padrões, mesmos que estejam simplificados, não são utilizados só para o cenário, diversos exemplos deles podem ser utilizados em outras áreas de design de um jogo ou apenas uma parte de um sistema que é colocado no jogo porque o jogador precisa de uma plataforma, e sim parte de todo um ecossistema interligado dentro da criação de um bom design no geral.

Apesar de usar muitos elementos da arquitetura como visto anteriormente, não quer dizer necessariamente que todos os cenários dos

jogos são fiéis à arquitetura do mundo real, alguns realmente são como vistos na figura 26, mas muitos em sua maioria quando analisada a "planta baixa", nota-se que ninguém construiria algo assim no mundo real.

Figura 37 Planta baixa do Mapa de Doom



Fonte: Battzcavehttps[39]

Isso funciona pelo fato da mente humana não conseguir propriamente traduzir espaços em 3D em mapas 2D durante a atividade do jogo, porque pensamos em espaço como algo relativo e não como algo absoluto. Por essa razão na maioria dos casos quando é criado um espaço em 3D, não é que ele precise ser totalmente realista, ou iguais às plantas em 2D, mas sim como o jogador se sente ao interagir com tal cenário.

Por essa razão que é importante levar em consideração a questão de proporção do cenário em relação ao jogador e a câmera utilizada, se um objeto for grande demais para o cenário, esse será um dos primeiros que o jogador irá notar, ou que a textura de tijolos da parede está grande e totalmente irrealista, por exemplo.

Essa proporção como dita antes é influenciada muito por como o jogador vê o jogo, se o jogo é em primeira pessoa, ou em terceira pessoa, onde está posicionada a câmera do carro, dentro ou fora do *cockpit*. Tudo isso muda a dimensão do cenário para o jogador e consequentemente como ele reage aos objetos da cena.

[39] Imagem retirada do site. Disponível em: https://battzcave.wordpress.com/2016/04/10/leveldesignofvideogames01-doom/ Acesso em set – out 2016.

Por essa razão é importante ver a composição do cenário tanto do ponto de vista do jogador quanto da parte técnica; deve-se ter consciência de levar em consideração todos os elementos do jogo, seja ele da parte mecânica, estética e tecnológica, tudo isso irá no final influenciar o desenvolvimento.

No final, o objeto é sempre organizar e compor os cenários de uma forma que permita ao próprio jogo, crescer dentro do jogador e criar essa imersão que é algo divertido e interessante. *Level Design* no final das contas é exercer tudo isso em minúsculos detalhes, por isso que em cada jogo, ele irá mudar para se adaptar a situação, mas o conceito de todos eles serão muito parecidos, tornando mais fácil tomar as decisões certas e claras para o desenvolvimento do projeto.

4 SÍNTESE

> "Se você tem uma ideia de jogo que não teve sucesso, não é necessariamente verdade que a ideia é um fracasso. Talvez só precise ser feito de uma maneira melhor, ou em um momento melhor" – Dave Hagewood

Flint Dille e John Zuur citam em seu livro *The Ultimate Guide to Video Game Writting and Design,* a criação de um documento que eles nomearam de *Executive Summary,* que é uma documentação da ideia dos jogos que eles apresentam para poderem vender suas ideias, ou apresenta-las para seus superiores. Já Jesse Schell no livro *The Art of Game Design A Book of Lenses,* cita que não existe um *template* mágico para esse documento que seja utilizado universalmente pela indústria de jogos.

Apesar disso, esses documentos são utilizados, e o único propósito deles é para se manter organizado, não esquecendo das ideias e decisões que já foram tomadas e também para ajudar a comunicar aos outros a ideia que está a ser desenvolvida.

Levando ambos em consideração, para facilitar a demonstração das decisões tomadas, foi adaptado um documento contendo apenas as informações pertinentes a este trabalho, sem o escopo abrangente de um jogo inteiro, mas dando um foco aos elementos mais importantes.

4.1 Processo de Conceito da Ideia

A ideia começou simplesmente como um novo estádio para o jogo *Rocket League*[40]. Criado pelo estúdio *Psyonix* e lançado em julho de 2015, *Rocket League* hoje é um jogo de grande sucesso que envolve colocar os jogadores no controle de um carro contra outros jogadores, onde o modo de jogo básico é uma partida de futebol, só que com carros. O jogo inclusive foi originado de um *mod*[41] para *Unreal Tournament*, que depois virou um jogo publicado pelo próprio estúdio chamado de *Supersonic Acrobatic Rocket-Powered Battle-Cars* em 2008, o que em um sentido, se tornou um conceito básico do que se tornaria *Rocket League*.

A ideia do PCC seria apenas criar um estádio diferente onde se pudesse jogar uma partida dentro do jogo; foi imaginado - como o jogo não tinha nada relacionado ao mar - que o tema do mapa poderia ser o mar.

[40] Informação acessível no site, constante nas referências.
[41] Informação acessível no site, constante nas referências.

Então surgiram alguns problemas com essa ideia, a mais importante seria porque não modificar o mapa de uma forma com que, assim como em *Rocket League* tem mapa para basquete e *hockey*, para um modo de jogo totalmente diferente?

Foi então que surgiu a ideia de não ficar apenas preso ao conceito de esportes e suas limitações impostas por ser algo direcionado a *Rocket League*.

Sendo assim, se decidiu incorporar elementos do jogo *Monster Truck Madness 2*. Jogo criado pelo estúdio *Terminal Reality* e publicado pela Microsoft, era um jogo simples de corrida com carros do estilo *Monster Truck*. Mas não era a corrida em si o elemento buscado para a incorporação do projeto, dentro do jogo existia um modo chamado *Rumble*, que era um minijogo com um simples conceito de que o ganhador era o jogador com mais pontos, e para fazer pontos, era necessário se manter com o carro em cima da plataforma delimitada no cenário. Logo essas regras geravam o fato de que o jogador tinha que conseguir manter o próprio carro dentro dessa limitação, enquanto tinha que tentar tirar os jogos, usando o próprio carro para empurrar, e cuidar de si mesmo para não ser empurrado para forma da plataforma.

Com esse processo em mente, a ideia começou a ganhar vida própria, deixando de ser algo para *Rocket League*, e se tornando algo que conseguisse se sustentar por conta própria, com influência e inspiração de ambos os jogos citados.

Nota-se que esse processo inteiro foi utilizado todo o sistema proposto durante o trabalho desenvolvido nesse documento, inclusive essas perguntas estão dentro do parâmetro de ver o jogo em forma de lentes e ao fato das decisões teriam sido tomadas em forma de ciclos, como mostra o sistema de *Loop* proposto no início.

Por isso com o conceito definido, ficou clara a tomada das próximas decisões, e que foram todas baseadas nas argumentações feitas durante o desenvolvimento do estudo de *game design,* que no final se tornam o documento do *Executive Summary*.

Esse capítulo, é o final do primeiro *loop* de decisões do projeto inteiro, e a partir dele, junto com a apresentação do projeto, se dará o primeiro *feedback* completo e recomeçará o ciclo fazendo as adaptações necessárias até se chegar ao produto final, o protótipo de jogo em 3D baseado nesse conceito, na entrega do PCC2 em junho de 2017.

4.2 *Executive Summary*

Titulo	*King of the Hill Arena* (*Working in Progress*)
Gênero	*Multiplayer, Racing, Sport, Online, Fast-Paced, Casual, Indie, Competitive, Co-op.*
Plataformas	Jogo desenvolvido no PC para ser rodado dentro do sistema Windows 10.
Tecnologia	Uso da ferramenta *Unreal Engine 4* disponibilizada de graça pela Epic Games para a criação de jogos. Nesse caso, a ferramenta será utilizada pelo seu potencial de uso na área de jogos 3D
Referências	*Rocket League* e *Monster Truck Madness 2*
Conceito	Jogo tem como objetivo criar experiências utilizando uma atmosfera competitiva entre os jogadores onde apenas um deles poderá sair vencedor. Localizado em uma ilha isolada, com câmeras ao redor para fazer a transmissão da partida, os jogadores devem disputar quem fica no topo mais tempo entre eles para ver quem será coroado o *King of the Hill.*
Tema	Competição pelo título de *King of the Hill*
Ambientação	Cenário 3D, localizado em uma ilha isolada no meio do oceano, onde foi construído uma arena. O mar do cenário está agitado, devido a possibilidade de tempestade, o tempo também está nublado, com uma leve chuva acontecendo com possibilidade de se tornar uma tempestade. A arena é fechada, mas suas paredes são transparentes, permitindo ao jogador ver o oceano e a vegetação nativa. Dentro da arena existe a plataforma principal onde os jogadores terão que competir por um espaço nela, ao redor da arena existe vegetações colocadas para esconder possíveis turbos e também para servirem como obstáculos aos jogadores

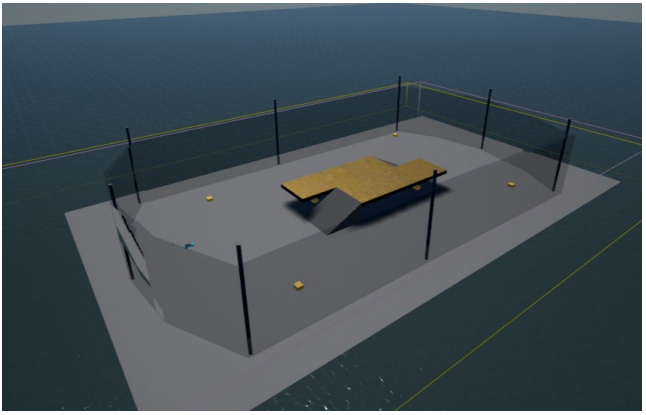
Estéticas	Jogo terá uma aparência com tendências realísticas, com a maioria dos objetos dentro de uma proporção realista, levando em consideração as possibilidades permitidas pela *Unreal Engine 4.*
Som	Jogo terá uma trilha sonora para dar ênfase a atmosfera de disputa entre os jogadores. Inclui também sons pela utilização de turbo e ao choque entre carros, além dos sons conhecidos feitos por um carro na realidade, como o barulho do motor aquecendo.
Objetivos	Jogador terá como objetivo competir contra até 4 jogadores (total de 5), pelo maior tempo possível de permanência na plataforma mais alta do cenário, quanto maior tempo ele permanecer, maior número de pontos o que aumentará sua chance de ganhar o jogo.
Objetivos Secundários	Possibilidade de destruir os inimigos e de acumular turbos pegos pelo mapa são mecânicas que o jogador terá que leva em conta para aumentar suas chances de ganhar a partida.
Mecânicas	Tempo de 5 a 30 Minutos por partida. Habilidade de controlar um carro. Atingir alta velocidade com o carro. Possibilidade de destruir os inimigos. Andar pela parede. Captura e uso de turbo para aumentar a velocidade. Ajuste de câmera em relação ao carro.

4.3 *Mock-up*

Baseado em todo o estudo e, principalmente, na ideia que virou o *Executive Summary* foi iniciado um *mock-up* para começar a ter um protótipo e de como o mesmo poderá a se tornar, e para que logo no início do desenvolvimento se descubra problemas tanto mecânicos, estéticos, tecnológicos e outros para diminuir o risco de problemas encontrados na parte final do desenvolvimento

Em estágio inicial de desenvolvimento, o protótipo será composto apenas com elementos simples, ou seja, apenas modelos que depois serão substituídos pelos modelos finais que farão parte do produto final.

Figura 38 Cenário Mock-up Panorâmica

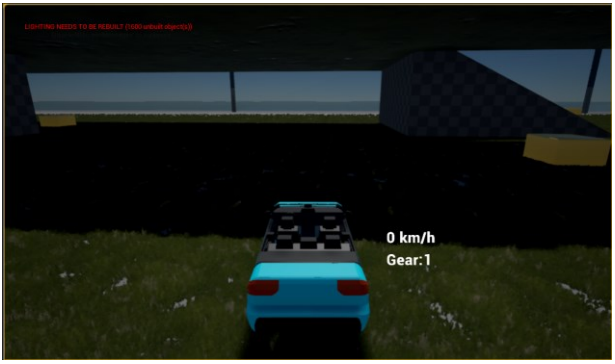


Fonte: Autor, 2016

Com essa imagem, podemos visualizar o formato próximo que a arena terá. O quadrado em cima do mar é apenas um substituto para o que depois virá a ser o modelo da ilha.

As paredes simbolizam o limite da arena, e estão transparentes para demonstrar que o jogador irá poder visualizar o horizonte, mesmo no produto final. E no centro se encontra a plataforma principal, onde os jogadores lutarão por um espaço e, consequentemente, ganhar o jogo.

Figura 39 Visão Jogador no Mock-up

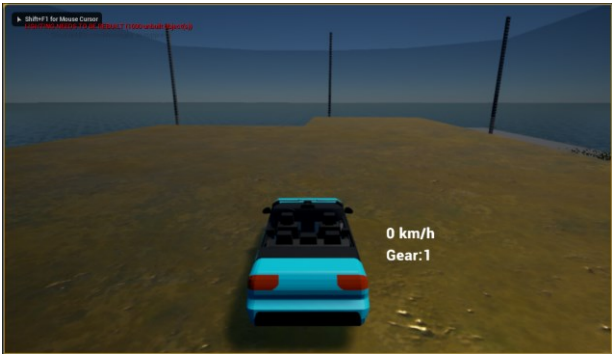


Fonte: Autor, 2016

Nessa imagem, coloca-se na possível câmera do jogador, e mais perto do cenário já se vê uma possibilidade de grama a ser utilizada no produto final, mas que nesse momento tem como propósito apenas ser demonstrativo da vegetação do local. O jogador também se encontra embaixo da plataforma, tendo-se a ideia de que ele poderá vincular também nessa área.

Assim como visto na imagem de cima, os quadrados amarelos representam possíveis locais de captura de turbo, que o jogador poderá adquirir.

Figura 40 Plataforma do Cenário do Mock-up



Fonte: Autor, 2016

Nessa última imagem, nota-se o carro já está em cima da plataforma e é o primeiro teste para ver se a plataforma está grande demais com relação ao modelo do carro, ou se tem que aumentar de tamanho de acordo com a quantidade de jogadores; os ferros vistos dentro das paredes representam os holofotes que serão instalados para dar iluminação à arena e assim por diante.

Essas são as primeiras decisões tomadas, ainda no início do estágio de desenvolvimento; mais decisões e testes terão que ser feitos para se chegar ao melhor resultado, com isso o protótipo ainda deve sofrer algumas alterações de proporção, tamanho, mudança de câmera etc.

Assim, durante o desenvolvimento, quando a modelagem for terminando, esses objetos - hoje na cena - serão substituídos por versões melhoradas e com a qualidade desejada para a finalização do cenário 3D.

5 DESENVOLVIMENTO

Com o fim do primeiro grande *loop* com a criação do *mock-up*, se deu início ao segundo grande *loop*, após o primeiro *feedback* recebido, com os ajustes e desenvolvimentos necessários, projetando a finalização do protótipo.

5.0.1 *Feedback* & Ajustes

Após a criação do *mock-up*, o *feedback* positivo foi recebido e com a aprovação da ideia, foi dada a continuidade ao cronograma programado e se começou a modelar o cenário.

Durante o desenvolvimento do projeto, a primeira mudança veio na remoção da ideia de a ilha dar suporte ao jogador e servir de acesso para a plataforma principal. Com isso, foi adicionada uma plataforma menor além da plataforma já existente, onde a plataforma antiga agora faz o papel da ilha e a plataforma menor faz o papel da plataforma principal.

Ao longo dos ciclos de desenvolvimentos utilizando essa ideia, notou-se que punir o jogador fazendo-o cair sempre ao mar sem uma segunda chance, acabou se tornando mais frustrante do que um desafio interessante. Para tentar consertar esse problema no design do jogo se criou uma plataforma abaixo, que permite, até um limite, ajudar a não punir com tanta severidade, um erro do jogador. Por consequência, duas rampas foram adicionadas para permitir ao jogador voltar da plataforma inferior para a plataforma de base, o que faz com que o mesmo volte a brigar pela plataforma principal.

Além de algumas mudanças no conceito do projeto, onde o principal objeto do projeto é ter um protótipo jogável, alguns elementos, principalmente estéticos, como efeito de chuva, raios, efeitos de partículas, fumaça, alguns *foleys* de som, foram acrescentados para o refinamento do produto; foram colocados com baixa prioridade e não estarão no protótipo final, mas estariam no jogo finalizado.

Todas essas mudanças foram feitas durante o ciclo de produção do protótipo com a intenção de criar uma experiência de jogo que, assim como diz MDA, acabe gerando uma sensação no jogador, dentro de uma narrativa de jogo, em um mundo de fantasia, onde na partida se encontre desafios, seja imposto pelo mapa ou por outros jogadores, e qualquer outro elemento que possa vim a ajudar a criar a melhor experiência ao jogador e serão melhor detalhadas nos próximos capítulos.

5.1 Modelagem

Após o início dado com a criação de um simples *mock-up*, e feitas as observações e *feedbacks* necessários, se iniciou a fase de criar os modelos finais a serem utilizados no produto final que substituíram os colocados na cena.

Essa parte do processo teve diversas influências e ajustes foram feitos de acordo com os parâmetros colocados na metodologia utilizada.

Nessa fase, o jogador tem um maior contato com a parte estética e com o resultado final da jogabilidade ao controlar o carro no protótipo, enquanto o desenvolvedor terá acesso a detalhes que não passam pelo jogador, como por exemplo diferentes versões das plataformas desenvolvidas e testes feitos para chegar a qualidade necessária para o protótipo.

Pelo lado do desenvolvedor, notou-se uma grande influência das mecânicas do jogo na hora de modelar os objetos principais da cena, seus formatos e tamanhos foram modificados inúmeras vezes durante o processo inteiro.

Múltiplos ciclos dentro do desenvolvimento foram feitos para que o resultado final conseguisse suprir as necessidades postas pelas regras do jogo e suas mecânicas, mas ao mesmo tempo é necessário seguir algumas regras para que cada objeto não acabe sobrecarregando a *Unreal Engine 4* de forma desnecessária e o projeto acabe tendo uma performance pior dentro da tecnologia optada para o jogo, afetando a interação do jogador com o produto final.

Cronograma seguido nessa parte do desenvolvimento foi começar com a criação da ilha principal, seguido pelas plataformas dentro da arena,

e depois complementando com objetos na cena, como poste de luz, rampas etc.

5.1.1 *Assets* e UVs

Essa área de desenvolvimento, a parte que o desenvolvedor tem uma maior interação são com as regras "invisíveis" ao jogador, mas que são necessárias serem seguidas durante os ciclos de desenvolvimento dos *assets* para que a interação jogo & jogador seja a melhor possível.

Praticamente todas as áreas de um design de um jogo influência nos *Assets.* Serviram de base para a criação da Estética do jogo, influenciaram as Mecânicas do protótipo, e a criação deles teve que ser levada em conta regras impostas pela Tecnologia, nesse caso, a *Unreal Engine 4* e todos os *Assets* tiveram que ser criados tendo em mente o propósito de reforçar o tema principal do jogo, ou seja, a disputa pelo título de *King of The Hills.*

Antes de começar o desenvolvimento desses *Assets*, regras vindas da parte de Tecnologia tiveram que serem definidas para evitar problemas durante o ciclo de produção final do produto:

- ✓ Menor número possível de polígonos em cada *Assets*, trabalhando assim com que é chamado de *Asset Low-Poly*. Nesse projeto a quantidade de *assets* compondo a cena é menor do que em uma produção de alta escala, por isso tecnicamente seria possível utilizar *assets* com uma quantidade de polígono muito maior, mas para se criar o hábito de sempre procurar otimizar o jogo e esse protótipo servir de exemplo para próximos, todos os *assets* foram desenvolvidos para que tivessem a menor quantidade de polígonos possíveis.
- ✓ As faces de um objeto que não irão aparecer para o jogador, tiveram de ser deletadas, mesmo que o objeto tivesse ficado com a aparência de "cortado". Essa técnica por exemplo reduz a quantidade de vezes que a *Unreal Engine 4* calcula a iluminação e suas consequências em cada *asset*, então se uma face for eliminada e o objeto é multiplicado por 10 na cena, são 10 faces a menos que a plataforma tem que calcular.
- ✓ Levou-se em conta na criação do modelo que pequenos detalhes normalmente feitos no mesmo, foram feitos apenas na textura do objeto para evitar uma multiplicação do número de faces dentro de cada *Assets*.

O ciclo de produção dessa parte se notabilizou pelos *loops* entre os programas *3Ds Max* e *Unreal Engine 4*, com o modelo sendo criado no primeiro e aplicado com testes no segundo, detalhes e ajustes foram sendo feitos e adaptados no decorrer da produção utilizando esse ciclo, até se chegar ao modelo final.

Além de modelar os objetos finais, teve-se que levar em consideração a parte de transição para a próxima fase de desenvolvimento, que é a texturização, para isso o modelo já estará pronto e optimizado, utilizando os parâmetros impostos anteriormente, e como adicional é foi necessário adaptar as UVs de cada objeto da seguinte forma:

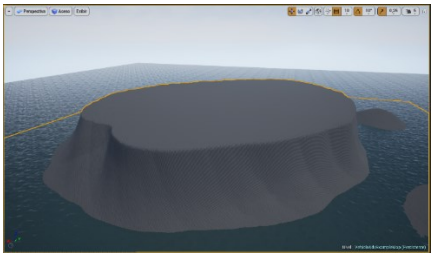
- ✓ Organizou-se a malha do objeto para que fosse possível aplicar a textura no objeto.
- ✓ Ocupou-se o maior espaço possível com o objetivo de acomodar o maior número de informações para textura e para o cálculo de iluminação dentro da *Unreal Engine 4*.
- ✓ Manteve-se a proporção de cada Uv para não haver distorção do material aplicado no objeto.

5.1.2.1 Ilha e Consequências

Primeiro grande *asset* a ser criado para o protótipo do jogo foi a ilha que daria o suporte para o cenário inteiro do jogo, utilizando-se de referências citadas anteriormente, como base para a criação desse modelo.

Depois de inúmeras iterações do formato, tamanho, proporção, e testes em diferentes modelos criados para fazer o papel principal da ilha, esse foi o resultado escolhido.

Figura 41 Modelo Ilha



Fonte: Autor, 2017.

Mas independente do formato, tamanho e qualquer outro parâmetro na criação da ilha, um problema sério acabou acontecendo nessa fase do desenvolvimento, toda vez que a cena criada no *mock-up* era incorporada ao cenário e colocada na ilha - e os primeiros testes eram feitos com o carro - todo *feedback* recebido dizia a mesma coisa: *"Cenário e modo de jogo alternativo de Rocket League"*

A ideia do projeto é ser um protótipo de jogo com inspirações em outros jogos e não parecer uma cópia ou modo alternativo do mesmo.

Então para adaptar o projeto e fugir desse mantra, uma decisão teve que ser tomada, e a ideia de colocar a arena em uma ilha foi totalmente descartada, criando consequências para todos os próximos *assets* a serem criados para o projeto.

Com isso a ideia do projeto foi adaptada para que os limites impostos pela parede da arena fossem removidos e o cenário passou a ser no meio do oceano, sem a ilha, criando toda uma nova dinâmica nas mecânicas do jogo, por que agora sem o limite das paredes, sem o "piso" da ilha para dar suporte aos jogadores, ao cair da plataforma, o preço a ser pago era muito mais alto já que os jogadores agora caem na água.

Consequentemente, as ideias de *assets* para as plataformas sofreram alterações no *concept*, sem nem ao menos terem sido criados; por esse motivo que é importante fazer o protótipo e ir criando e adaptando o projeto de acordo com o *feedback* e resultado recebidos, pois graças a esse *feedback* no início do desenvolvimento do protótipo final teve-se tempo suficiente de adaptar o projeto com os ajustes necessários.

Se o *feedback* tivesse sido feito apenas no final da produção do protótipo, não haveria tempo para um ajuste dessa magnitude e a única opção seria tentar mascarar esse problema ao invés de solucioná-lo, como foi feito nesse protótipo.

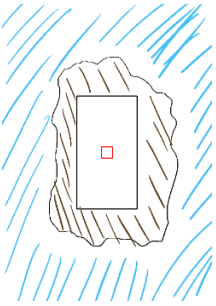
5.1.2.2 Plataforma Base

Com as mudanças impostas na remoção da ilha no cenário dando suporte à arena, o design da plataforma principal teve que ser mudado e algumas adaptações foram feitas.

Antes, a plataforma que servia de base para o jogador subir na plataforma principal, era a própria ilha, com a remoção da ilha, o que antes era a plataforma principal de disputa dos jogadores, acabou se tornando a plataforma base que permitiu o acesso a uma plataforma, agora menor do que era antes em conceito, que é a área de maior contestação entre os jogadores.

Primeira grande adaptação feita foi mudar o formato do cenário, antes o formato era uma ilha, com uma arena em formato retangular delimitando a área de jogabilidade do jogador, com outra plataforma de formato quadrado no centro.

Figura 42 Concept Cenário Antigo

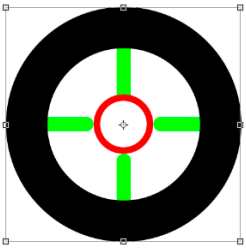


Fonte: Autor, 2017.

Aqui a mecânica de jogo era simples, o jogador teria controle do carro para movê-lo como necessário dentro do espaço largo delimitado pela arena e tentar subir dentro da plataforma localizada no centro e brigar por um espaço dentro da mesma.

Como a mudança de design e a ideia de alto risco e alta recompensa (imposta pelo perigo de cair no mar ao tentar brigar pelo ponto mais alto do mapa), algumas mudanças estruturais tiveram que ser feitas, principalmente em relação à forma e ao tamanho do cenário.

Figura 43 Concept novo cenário



Fonte: Autor, 2017.

Para isso seguiu-se o raciocínio proposto por Schell, onde para poder se visualizar a parte de mecânica do jogo, é necessário retirar a estética do jogo e apenas analisá-lo como linhas que formam o espaço e determinam os elementos dentro do jogo.

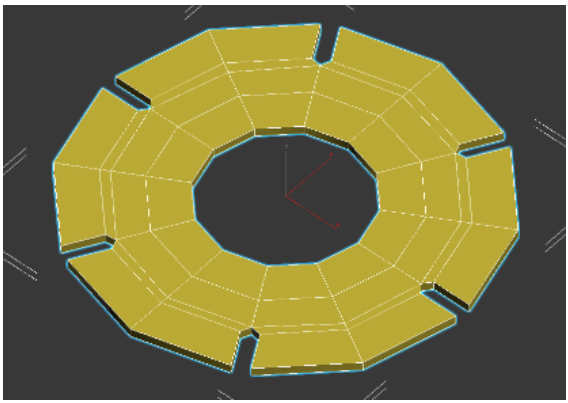
A área preta representa a plataforma base, o círculo vermelho representa a nova plataforma a qual os jogadores deverão disputar espaço, e área verde representa as rampas de acesso a essa pequena área, fazendo com que essas linhas sirvam de fronteiras para outras áreas, onde todas estão conectadas em sequência uma as outras.

O motivo da transformação de um formato de quadrado para redondo é que com a redução de espaço, a interação do jogador com a mecânica de dirigir um carro afetou como a área seria construída. Por isso a escolha por algo mais oval, ao contrário de uma área quadrada onde o jogador não teria maneabilidade para andar com o carro, assim uma plataforma oval condiz, em um espaço menor, com o comportamento físico de um carro, por exemplo quando o jogador tenta manobrar o carro ao fazer a curva, a rota feita pelo carro é, em condições normais, curvilínea. Antes isso não era um fator porque a área delimitada pela arena era maior e as paredes da arena tinham um pouco de curva para ajudar a manobrar o carro.

Seguindo essas condições, a visão que se tem para a criação do cenário e suas mecânicas de jogo é baseada no espaço do tipo *Web*, com elementos de *Grid*, onde a ideia é fazer com que o círculo vermelho seja considerado o “Strong Centers” do cenário e que as outras plataformas sirvam de *Boundaries*, limitando o espaço de jogo do jogador, ditando também as regras de jogo, de forma que tenha uma *Local Symmetries* no cenário, fazendo com que o cenário seja balanceado e orgânico.

Com esses novos parâmetros e condições a serem levadas em conta, iniciou-se os primeiros desenvolvimentos do *asset* final da plataforma-base e o primeiro resultado foi a figura a seguir:

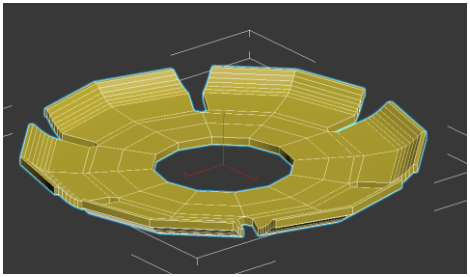
Figura 44 Primeiro Modelo Plataforma Base



Fonte: Autor, 2017.

Esse foi a primeira interação da plataforma e imediatamente após a criação foi implementado dentro da *Unreal Engine 4* e após testes, se notou que era muito fácil para os jogadores caírem da plataforma, pois qualquer erro a plataforma punia de forma desnecessária, com o jogador caindo, e em um jogo que os jogadores irão se chocar bastante, se a cada 5 minutos de jogo, o jogador passar 4 minutos fora porque ele não para de cair, acaba criando uma péssima mecânica de jogo e tira a diversão de jogar.

Figura 45 Plataforma Base v2



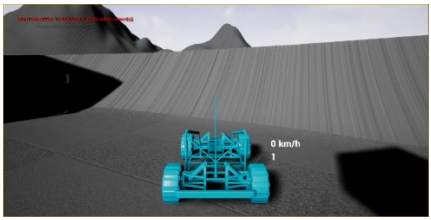
Fonte: Autor, 2017.

Para arrumar esse problema, foi criada uma pequena rampa em cada pedaço da borda da plataforma, semelhante formato ao que era usado para delimitar a área quando a ideia ainda era uma ilha, essa ideia arruma

o problema anterior e enriquece as mecânicas de jogo e a dinâmica do jogador em relação ao cenário, porque agora ele pode utilizar as rampas para se esquivar de algum jogador, ou ganhar vantagem de velocidade para subir nas rampas que levam à plataforma principal.

Com isso levou-se esse cenário para dentro da *Unreal Engine 4* para novos testes e *feedbacks,* como é possível observar na imagem a seguir:

Figura 46 Teste v2 na Unreal Engine 4



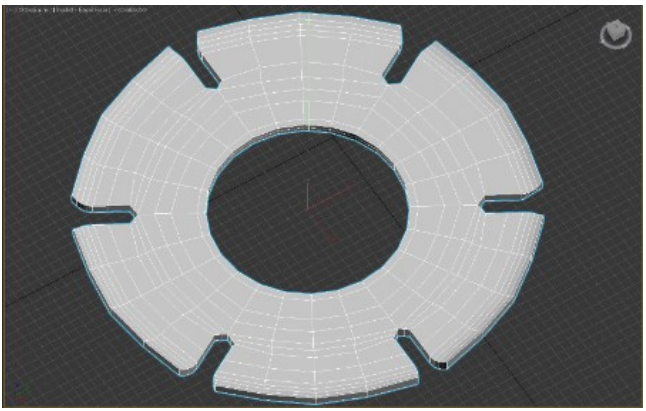
Fonte: Autor, 2017.

Obviamente que apenas duas interações não seriam suficientes para encontrar o formato ideal; como previsto encontrou-se novos problemas, a rampa estava muito alta e tornava praticamente impossível o jogador cair, ou seja, saiu-se de um extremo para o outro.

Apesar disso, o conceito estava funcionando para a ideia e o objetivo desejado, faltava apenas acertar os parâmetros da plataforma para que se chegasse ao resultado esperado, que era a ideal interação com o jogador.

Após diversos ajustes e testes feitos na plataforma, concluiu-se o desenvolvimento da plataforma-base com o resultado a seguir:

Figura 47 Modelo Plataforma-Base Final



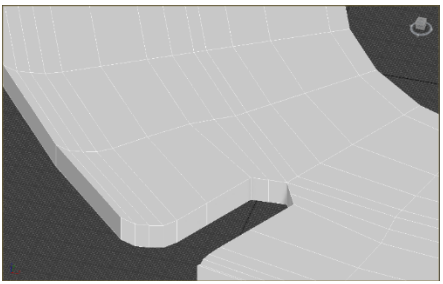
Fonte: Autor, 2017.

Algumas pequenas mudanças, como diminuição da rampa nas bordas e o aumento do espaço entre segmentos da plataforma, mais o espaço vazio no meio, foram os três maiores fatores de ajustes e optimização da plataforma para que tivéssemos uma mecânica de jogo mais perto possível da ideal.

Essa última versão, após testes e *feedback* positivos, foi levada para a próxima etapa de desenvolvimento que foi preparar o *asset* para receber texturas, a parte estética visual do modelo.

Nota-se que na Figura 47, a quantidade enorme de polígonos existentes na plataforma, em cena isolada, não seria um problema, mas como o projeto tem em mente um jogo completo, toda otimização ajuda, e qualquer coisa que é desnecessária, ao acumular, acaba custando a performance da *Unreal Engine* e, consequentemente, do próprio jogador.

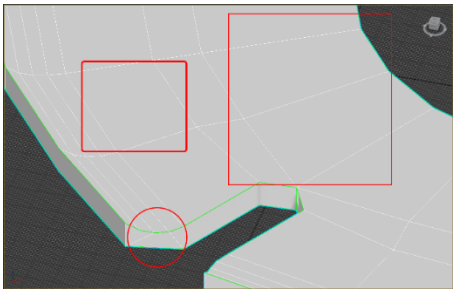
Figura 48 Modelo Plataforma Base não otimizado



Fonte: Autor, 2017.

Essa é a primeira figura do objeto, sem nenhuma alteração, a seguir, o mesmo objeto, optimizado e pronto para receber a textura e implementação dentro da *Unreal Engine 4.*

Figura 49 Modelo Plataforma Base otimizado

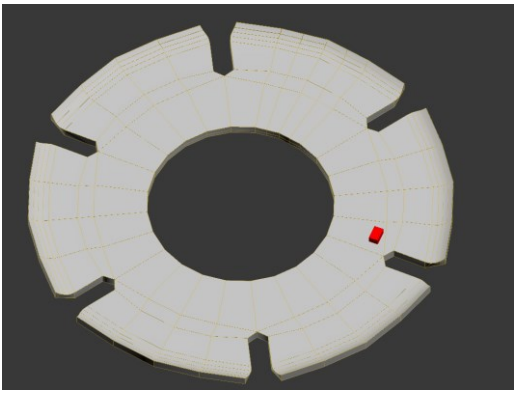


Fonte: Autor, 2017.

As áreas marcadas pelas geometrias em vermelho demonstram as áreas mais afetadas pela redução de polígonos, nota-se que em nenhuma das áreas, a redução mudou o terreno no qual os carros passam, inclusive a área delimitada pelo círculo, teve uma alteração significativa, mas se vê que a rampa em si onde o carro passa, continua intacta, mantendo seu formato curvilíneo que permite ao carro ter uma transição suave como em uma rampa. Essa mudança foi apenas em um quadrante da plataforma, agora aplicado em todos os outros, a quantidade de cálculos necessários removidos que teriam que ser feitos pela *Unreal Engine 4*, acabou sendo exponencialmente importante considerando que muito mais objetos

foram implementados na cena e cada mudança teve um fator relevante na performance.

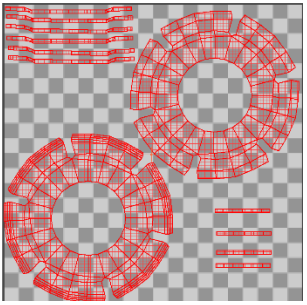
Figura 50 Plataforma Otimizada c/referência



Fonte: Autor, 2017

Nessa imagem nota-se o objeto polido e em vermelho, está a referência do carro utilizado para decidir-se o tamanho da plataforma em sua versão final. Com o objeto polido e optimizado, próximo passo foi preparar o *asset* para a fase de texturização, para isso houve a necessidade de ajustar seu UV de acordo com os parâmetros definidos anteriormente.

Figura 51 Uvs Plataforma Base



Fonte: Autor, 2017.

Após todos esses desenvolvimentos, a plataforma-base ficou pronta para o próximo passo do desenvolvimento do protótipo que foi criar as suas texturas.

5.1.2.3 Plataforma *King of The Hill*

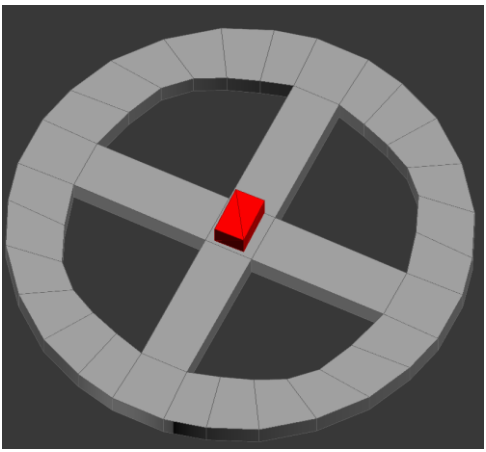
Com o fim da ilha, surgiu a necessidade de criar um novo dinamismo na mecânica do jogo, com a plataforma que antes era a principal virando a plataforma-base e fazendo o papel da ilha, assim apenas uma plataforma tornava redundante o tema de *King of The Hill*, já que todos tecnicamente já começariam no ponto mais alto do mapa, então para seguir o tema e reforçá-lo foi criada uma pequena plataforma, acessível por quatro rampas no mapa, onde essa plataforma deveria ser de difícil manobra, e deveria ser algo dentro do jogo que fosse de alto risco, mas que oferecesse uma recompensa (vitória/maior quantidade de ponto).

Para chegar-se a esse objetivo, alguns elementos foram levados em conta, como o ponto mais importante do mapa, ela deve criar o efeito de *Echo*, *Not-separateness* e, principalmente, de *Strong Centers*, essas três características, seja pelo formato do objeto ou estética, têm que ser remetidas ao objeto para que ele realmente demonstre ser o local mais desejado do cenário.

Sendo assim com o espaço no meio na plataforma de base já projetado tendo em mente essa nova pequena plataforma que se torna agora o objetivo principal, o desenvolvimento acabou sendo menos complicado do que a base principal, já que alguns parâmetros como tamanho e proporção já foram resolvidos no desenvolvimento da plataforma anterior, logo a quantidade de testes e ciclos produzidos nesse *asset* acabou sendo mais objetivo e reduzido, com os ajustes sendo apenas de proporção e se o carro conseguiria manobrar dentro da plataforma.

Sendo assim, o *asset* final foi produzido e modelado de acordo com a imagem a seguir:

Figura 52 Modelo Plataforma King of the Hill



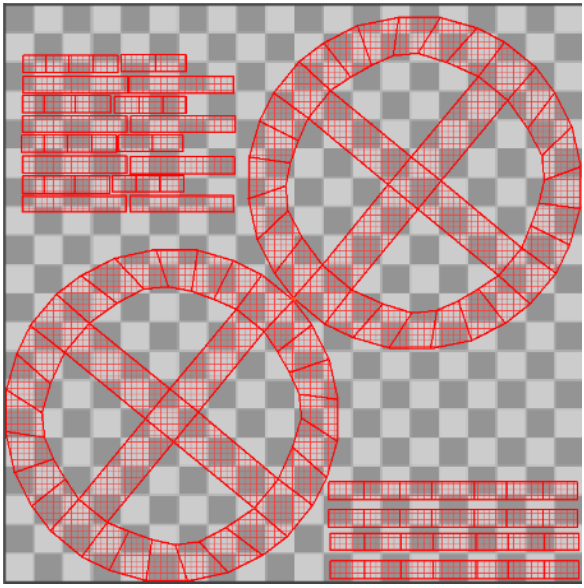
Fonte: Autor, 2017.

Já que a plataforma de base tem que obrigatoriamente ter uma rampa na borda para evitar os problemas citados anteriormente, o espaço na plataforma criada foi apenas o suficiente para um carro manobrar, com certa dificuldade, e em caso de choque ou empurrão de outro oponente, a chance de derrubá-lo aumentava, mas ao mesmo tempo, se o jogador não arriscasse em nenhum momento a subir na plataforma principal, ele não ganharia o jogo.

Ajustes na malha do *asset* para otimizá-lo ao jogo não foram muito necessários porque diferentemente da plataforma-base, onde sua forma mudou bastante durante o desenvolvimento, a plataforma principal desde o início teve seu processo já em mente de ter o mínimo possível de polígonos, e ajuda também o fato de que é uma plataforma mais simples comparada a anterior.

Com o processo de modelagem finalizado para a plataforma principal, apenas os ajustes de preparação para receber a parte estética visual foram feitos e preparado para o próximo passo de desenvolvimento.

Figura 53 Uv's Plataforma King of The Hill



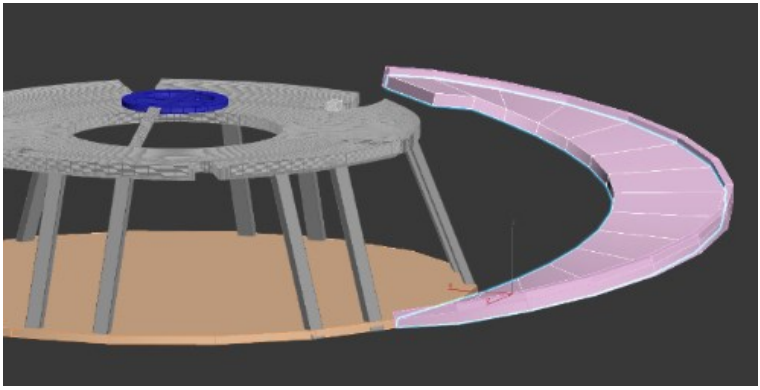
Fonte: Autor, 2017.

5.1.2.4 Rampas e Plataforma Inferior

Durante o desenvolvimento, alguns *feedbacks* recebidos davam conta de que mesmo com os ajustes feitos para evitar que o jogador caísse com muita facilidade, quando o jogador caía na água, o preço a ser pago era alto, pois ele perdia controle do carro e apenas esperava voltar para a disputa.

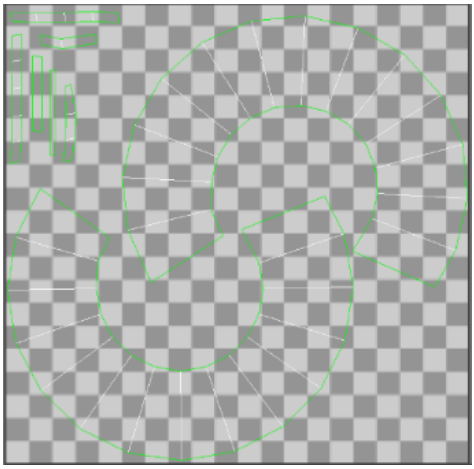
Para tornar mais interessante a punição e não apenas um evento de espera do jogador, se adicionou uma plataforma mais abaixo, onde ao invés do jogador esperar para ter o controle do carro novamente, ele cai nessa plataforma e tem que levar o seu carro, através de duas novas rampas adicionadas, de volta para área de disputa. Essa ideia não removeu completamente a punição anterior, em certas partes do mapa ainda é possível o jogador cair direto na água e pagar a punição maior, apenas o que foi feito foi adicionar um novo tipo de punição, criando uma dinâmica diferente por adicionar diferentes tipos de punições, umas mais leves, outras mais pesadas ao jogador.

Figura 54 Modelo Rampa e Plataforma Inferior



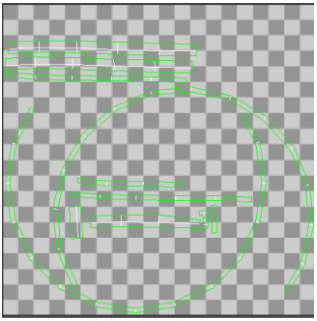
Fonte: Autor, 2017

Figura 55 UVs Rampa



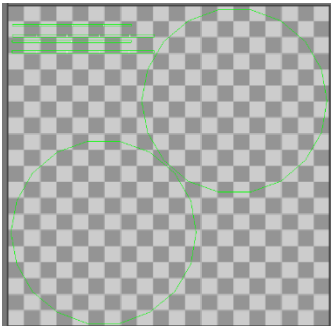
Fonte: Autor, 2017

Figura 56 Uvs Suporte da Rampa



Fonte: Autor, 2017

Figura 57 Uvs Plataforma Inferior



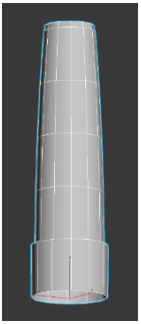
Fonte: Autor, 2017

5.1.2.5 Objetos Secundários

O protótipo contém alguns objetos secundários como postes, que são objetos simples, mas que complementam o cenário, além disso, há dois objetos secundários com uma complexidade um pouco maior que merecem um pouco mais de atenção.

O primeiro é a base do poste que o jogador vê quando está presente na plataforma inferior.

Figura 58 Modelo Base do Poste

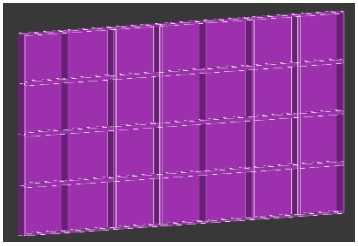


Fonte: Autor, 2017

Nota-se o mínimo possível de uso de polígonos para criar o formato desejado, e inclusive na parte inferior, o polígono foi deletado já que o jogador nunca o verá sendo que essa parte estará submersa na água.

O segundo objeto de importância são os refletores criados que servem como fonte primária de iluminação do cenário.

Figura 59 Modelo Refletores de Luz



Fonte: Autor, 2017

Ambos objetos tiveram que passar pelo processo de otimização e criação de Uvs para criar texturas customizadas para eles.

Apesar de serem cruciais para o contexto do cenário e a ideia por trás da criação da plataforma, ambos os objetos, desde o início, nunca foram pensados e modelados como se o jogador fosse vê-los ou focá-los

durante o jogo, diferentemente das plataformas e rampas que fazem ativamente parte de todos os elementos dos jogos, esses objetos e outros pequenos secundários têm como função principal apenas a estética e não influenciam a mecânica do jogo diretamente.

5.2 Texturização

Texturizando os objetos modelados, começa-se a visualizar a parte onde o jogador tem um maior contato, que é a parte de *Aesthetic* (Estética) do jogo. Nessa parte, o objetivo é criar texturas e materiais que condizem com o que foi decidido anteriormente no sumário executivo.

A parte de proporção foi adquirida durante o processo de modelagem, logo aqui todas as texturas e materiais criados tinham que seguir uma lógica mais realista e com elementos que remetessem a nossa realidade hoje, e não por exemplo metais futurísticos ou pedaços de madeiras da época medieval.

No decorrer dessa etapa, foi necessário a criação e recriação de diferentes estilos de texturas para os objetos, até que se chegasse a um resultado final que atendesse a tal necessidade.

Nesse processo, foi exportado do programa *3Ds Max,* os modelos prontos para dentro do programa *Substance Painter 2* onde foi possível criar uma textura customizada para cada objeto da cena que requeriam uma maior complexidade em sua textura.

O processo foi separado em 3 fases na criação da textura para cada objeto.

- ✓ Mapa *Height*

No mapa de *Height* foram detalhados os *assets*, onde foi necessário criar um efeito de profundidade; esse efeito foi criado na textura do objeto para evitar aumentar a quantidade de polígonos exponencialmente em um objeto.

- ✓ Material/*Base Color/Emmisive Color*

Nessa parte do procedimento foi colocada a cor e o material verdadeiro em cada parte do objeto, ou seja, se uma área é metal, coloca-se um material de metal correspondente, incluído também nessa área o material de emissão de luz que compõem alguns *assets* da cena.

- ✓ *Details*

Fase final da texturização foi colocar pequenos detalhes que ajudaram a enriquecer o material. Por exemplo: marca de pneus nas plataformas deixadas por “jogos anteriores”, efeitos de arranhões nos metais devido também aos próprios carros passando por cima, entre outros.

Com o processo de criação finalizado, a parte final é exportar essas texturas de uma forma que seja possível remontar o material dentro da *Unreal Engine 4* e para isso, diferentes mapas de texturas serão criados para completar essa parte do desenvolvimento:

1. *Base Color Map*
2. *Normal Map*
3. *Occlusion / Roughness / Metallic Map*
4. *Emmisive Color Map* (Quando necessário)

5.2.1 Plataforma-base

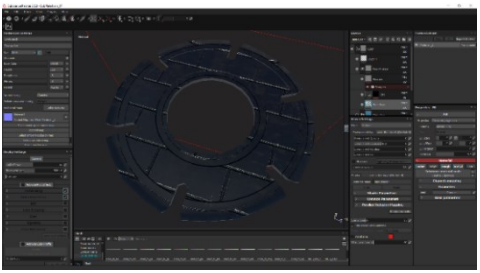
Objeto que mais sofreu alterações em sua textura foi a plataforma-base, que acabou servindo de referência para todos os outros objetos seguirem na cena a mesma lógica e mantivessem a estética do cenário.

Como esse objeto serviria de referência para todos os outros objetos do protótipo, acertar suas estéticas e como o jogador reagirá, era o passo mais importante do desenvolvimento das texturas do cenário, pelo fato de que essa parte é onde o jogador tem o maior contato e interações com o jogo desenvolvido.

Tendo em mente esse desafio, o desenvolvimento desse objeto, teve uma quantidade maior de *loops* na produção de seu material, o que parecia estar bom, nem sempre refletia o mesmo resultado dentro da *Unreal Engine 4.*

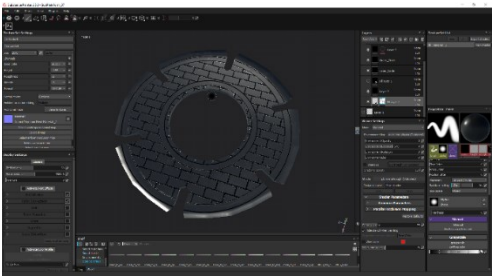
Primeiro passo então foi começar pela criação do mapa de *Height* com Base Color, para visualizar as ideias em prática e que servisse como um norteamento para chegar-se ao resultado que daqui para a frente ditaria a estética do jogo.

Figura 60 Plataforma Base Textura Teste 2



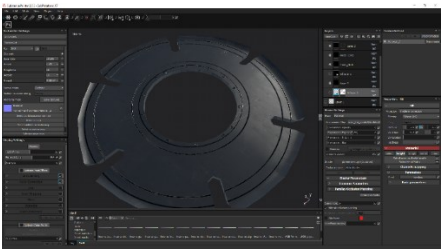
Fonte: Autor, 2017

Figura 61 Plataforma Base Textura Teste 2



Fonte: Autor, 2017

Figura 62 Plataforma Base Textura Teste 3

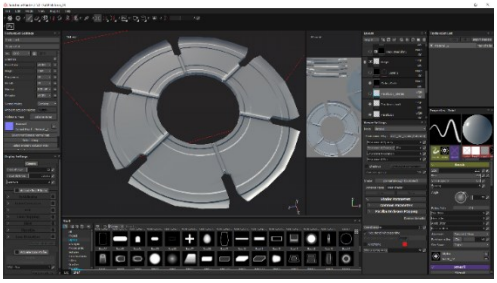


Fonte: Autor, 2017

Durante o desenvolvimento inicial o objetivo da estética do jogo era ter tendências realistas, então algumas questões arquitetônicas foram levadas em consideração para chegar ao resultado final.

Para isso no resultado foi criado entre os espaços, uma espécie de "liga de metal" que simula como se a plataforma não fosse apenas um objeto, e sim composta de várias pequenas partes - seis para ser exato - que estão conectadas por essa liga de metal.

Figura 63 Mapa Height Plataforma Base v1



Fonte: Autor, 2017

Com esse detalhe em mente, e tendo o primeiro resultado satisfatório criado, se seguiu com o ciclo e exportou-se para dentro da *Unreal Engine 4* para ver se algum erro, que antes não estava aparente, ainda existia no *asset*.

Figura 64 Teste Plataforma Base dentro da Unreal Engine 4

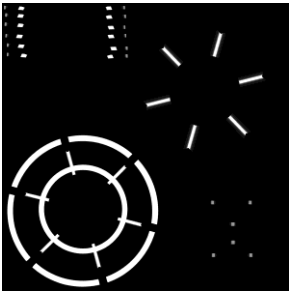
Fonte: Autor, 2017

O resultado estava aceitável e não foi notado nenhum problema a princípio, logo iniciou-se a segunda fase do desenvolvimento da textura do objeto.

Primeiro passo para a criação das texturas foi a criação de *Mask* baseado no mapa de *Height* criado anteriormente, permitindo colocar texturas separadas em cada parte do objeto sem ter que pintar manualmente e correr o risco de sobrepor uma parte a outra.

Exportar o mapa de *Height* para dentro do programa *Photoshop* permitiu criar o mapa de *Mask,* que foi então reimportado para dentro do *Substance Painter 2*.

Figura 65 Mapa de Mask Plataforma-Base



Fonte: Autor, 2017

O resultado desse processo pode ser visto na figura a seguir, onde foram utilizados diferentes materiais como teste.

Figura 66 Teste com mapa de Mask

Fonte: Autor, 2017

Obviamente, as texturas do objeto não seriam assim, mas essa imagem serve de referência para mostrar que o *Mask* utilizou o mapa de *Height* e que estava funcionando corretamente.

Assim, várias ideias foram testadas para a plataforma: puro metal, puro concreto, mistura de concreto com metal, diferentes tipos de metais e concretos etc.

Nas imagens a seguir mostra-se algum dos exemplos que foram pensados e testados durante o desenvolvimento do material dessa plataforma.

Figura 67 Teste Textura 1 Plataforma Base

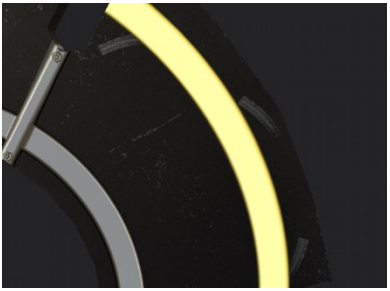
Fonte: Autor, 2017

Figura 68 Teste Textura 2 Plataforma Base

Fonte: Autor, 2017

Outro teste feito com um mapa emissivo de luz no círculo exterior, com a plataforma sendo em sua maioria com um tipo de concreto, e o círculo interior sendo de metal.

Figura 69 Teste Textura 3 Plataforma Base

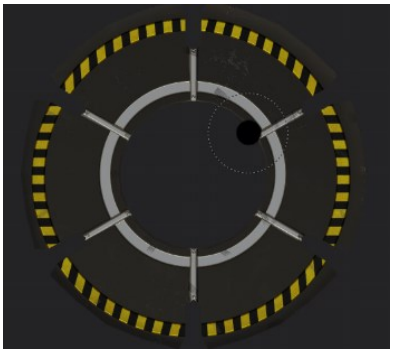


Fonte: Autor, 2017

A cada passo dado, chegava-se ao material ideal que se visualizava para a plataforma. Apesar de haver uma ideia mais concreta nessa fase do desenvolvimento sobre qual material seria predominante, faltava decidir o que o círculo interno e externo seriam. Por trás da ideia do círculo externo estava o fato de que era importante deixar claro ao jogador que aquilo era uma zona de perigo, perto daquilo, ou passando dessa linha, a chance de cair aumentava drasticamente.

Com essa ideia em mente, se criou uma linha preta & amarela para simbolizar o perigo pelo qual o jogador passaria ao movimentar-se e/ou acessasse essa área da plataforma.

Figura 70 Teste Textura Circulo exterior



Fonte: Autor, 2017

A incerteza e diversas mudanças drásticas de materiais testados do início ao fim nessa plataforma, apenas reforçaram a importância que era acertar a estética desse material.

Foram feitos mais alguns pequenos ajustes no material e assim chegou-se à fase final do desenvolvimento, onde foram colocados os detalhes finais para enriquecer a experiência do usuário e melhorar a estética do jogo.

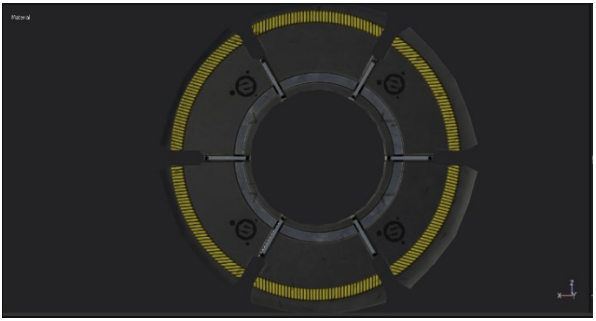
Figura 71 Textura Detalhes

Fonte: Autor, 2017

Nessa imagem, nota-se os detalhes implementados ao material, o símbolo do programa *Substance Painter 2* que simbolizam a propaganda na plataforma (ideia que veio a partir do fato de que a disputa é um evento televisionado), incluiu-se também setas que indicariam o que na época do desenvolvimento eram as entradas das rampas para a plataforma menor, e também as marcas dos pneus deixadas por carros em disputas passadas.

Com isso, a primeira versão final, acabou resultando na imagem a seguir:

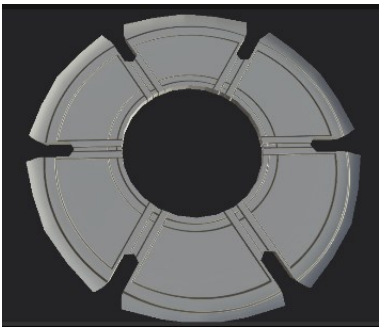
Figura 72 Textura finalizada v1 Plataforma Base



Fonte: Autor, 2017

Mais tarde no desenvolvimento, após a criação da plataforma menor e a colocação preliminar do sistema de iluminação, notou-se alguns problemas de proporção, principalmente no círculo interior e nos ligamentos de metal colocado nessa plataforma, para isso teve-se que voltar no mapa de *Height* e recriá-lo novamente com os ajustes necessários.

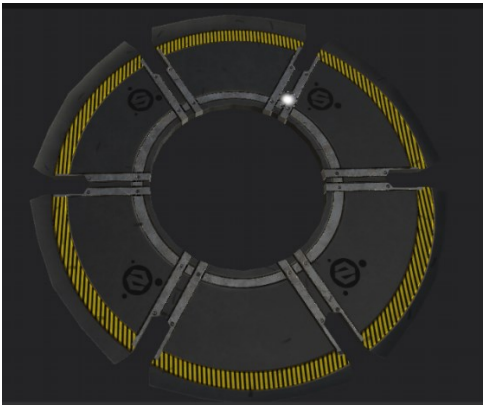
Figura 73 Mapa Height v2



Fonte: Autor, 2017

Novo mapa de *Height* baseado no *feedback* recebido, algumas pequenas mudanças, mas junto com outros ajustes feitos aos materiais e detalhes, a segunda versão final do *asset* texturizado ficou assim:

Figura 74 Textura finalizada v2 Plaataforma Base



Fonte: Autor, 2017

Nessa segunda versão, retirou-se as setas indicando a rampa por dois motivos, o primeiro sendo que as posições das rampas mudaram de lugar durante os testes e também porque se concluiu que colocar essas setas indicava uma falta de inteligência do jogador em não saber identificar a rampa para subir na plataforma principal.

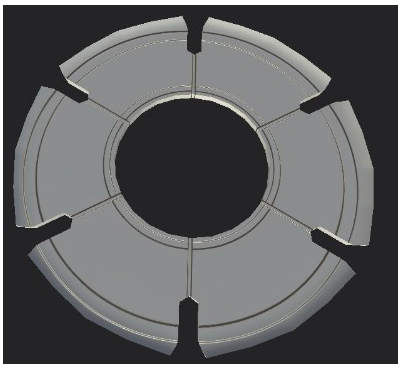
Após a criação das texturas para a plataforma *King of The Hill,* houve a necessidade de voltar e fazer uma terceira e final revisão em relação às texturas da plataforma de base com o *feedback* recebido.

Feedback recebido apontou ainda a necessidade de arrumar alguns problemas de proporção e deu a ideia de em vez da plataforma ser um metal menos refletivo, como estava até agora, que se criasse um material que fizesse o efeito similar de asfalto.

O produto final então foi a obtenção de uma plataforma inteira de asfalto, que teria ligamentos de metais, o círculo exterior continuava com o aviso de listras amarelas e pretas, e o círculo interior, inspirado pela implementação na plataforma *King of the Hill*, passou a ser um mapa emissivo de luz amarela. Os detalhes se mantiveram os mesmos apenas sofrendo pequenos ajustes em suas posições e tamanhos.

Tendo isso em mente o novo mapa de *Height* acabou resultando na imagem a seguir:

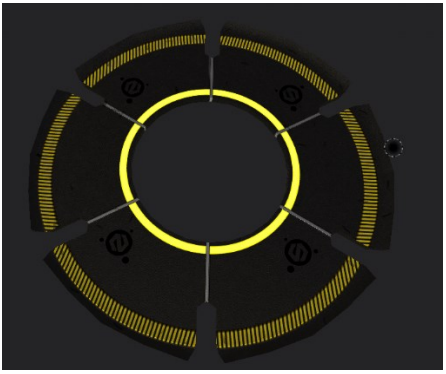
Figura 75 Mapa Height v3



Fonte: Autor, 2017

Semelhante à primeira versão, mas ajustado de acordo com a proporção necessária, e para finalizar o *asset,* a versão final da textura com os detalhes e o material de asfalto acabou resultando na plataforma-base com a seguinte estética:

Figura 76 Textura vFinal Plataforma Base

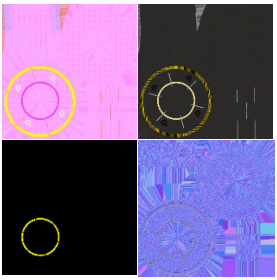


Fonte: Autor, 2017

Com o processo de texturização finalizado, o último passo foi exportar os arquivos para a recriação do material dentro da *Unreal Engine 4:*

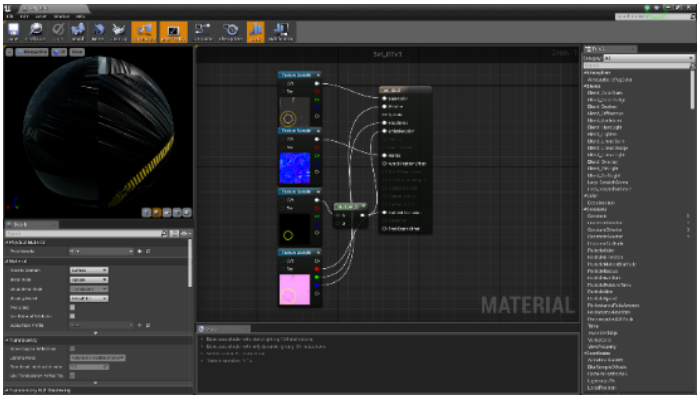
- ✓ *Base_Color Map*
- ✓ *Emissive (Se tiver) Map*
- ✓ *Normal Map*
- ✓ *Occlusion (Red Channel) / Roughness (Green Channel) / Metallic (Blue Channel)*

Figura 77 Mapas de Textura para importação na Unreal Engine 4



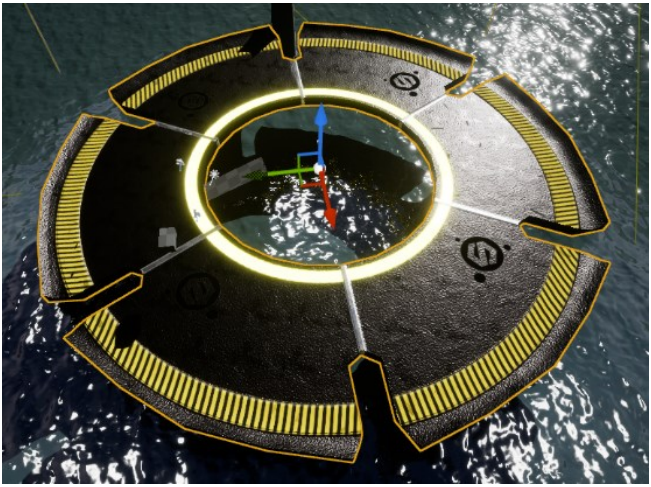
Fonte: Autor, 2017

Figura 78 Material da Plataforma Base dentro da Unreal Engine 4



Fonte: Autor, 2017

Figura 79 Plataforma Base texturizada dentro da Unreal Engine 4



Fonte: Autor, 2017

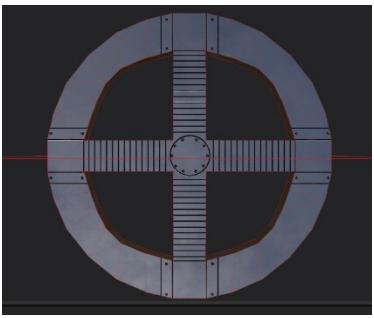
5.2.1 Plataforma *King of The Hill*

Com a plataforma de base completamente texturizada, fazer a plataforma principal foi um processo mais rápido e simples pelo fato de que diferente da plataforma-base, testes como os primeiros vistos não foram mais necessários.

Logo o processo foi mais simples, bastando apenas pegar a ideia por trás da outra plataforma e aplicar uma estética semelhante, mas dando alguns retoques para diferenciá-la da plataforma-base, mas com a importância de se tornar, através da textura, o *Strong Center* do cenário.

Assim, como anteriormente, o primeiro passo foi criar o mapa de *Height* que resultou ao final em algo semelhante à figura a seguir:

Figura 80 Mapa Height + Textura Metal Plataforma King of The Hill



Fonte: Autor, 2017

Nesse exemplo, um material de metal foi aplicado para se ter uma noção de como o mapa de *Height* estava se comportando.

No meio do desenvolvimento veio uma sugestão de *feedback* de *talvez* adicionar em algum lugar do material, uma espécie de "linha de chegada", para demonstrar ao jogador que ali era o objetivo final, mas apesar de alguns testes terem sido feitos, em nenhum momento a estética de uma linha de chegada acabou combinando com o resto dos objetos feito no cenário, incluindo a plataforma-base.

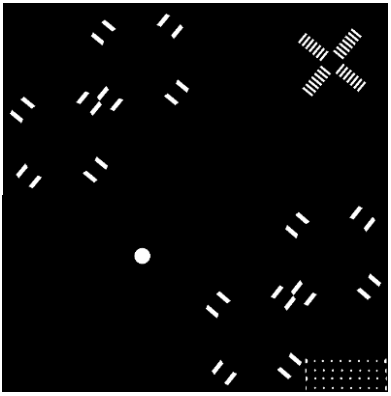
Figura 81 Teste Textura Linha de Chegada

Fonte: Autor, 2017

Mas a sugestão não foi descartada, a estética do objeto realmente tinha que chamar a atenção do jogador para mostrar que essa plataforma era o objetivo final. Foi então que junto com a textura na época criada para a outra plataforma, resolveu-se colocar luzes ao redor, com o efeito de minilâmpadas, que chamassem a atenção do jogador. Junto nessa fase

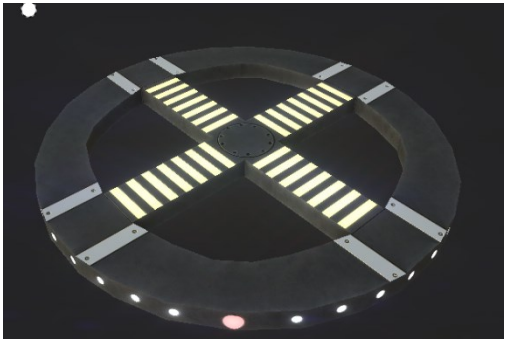
do desenvolvimento aproveita-se e já se aplica os mapas de *Mask* assim como na plataforma de base:

Figura 82 Mapa Mask Plataforma King Of The Hill



Fonte: Autor, 2017

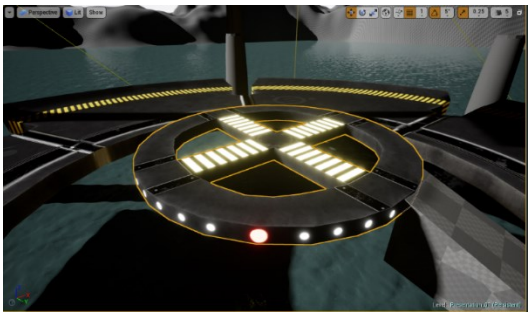
Figura 83 Textura v2 Plataforma King Of The Hill



Fonte: Autor, 2017

Essas luzes servem a dois propósitos, o primeiro de chamar a atenção, e o segundo como o protótipo terá um cenário a noite, também ajuda na iluminação do cenário.

Figura 84 Plataforma King Of The Hill teste dentro da Unreal Engine 4

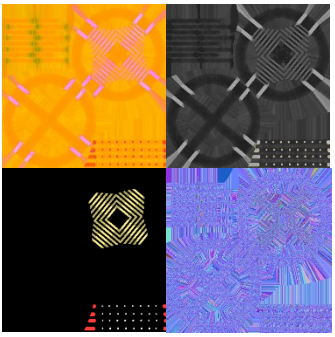


Fonte: Autor, 2017

A imagem acima, o cenário ainda não está na iluminação correta, mas já se nota dentro do jogo que a ideia está funcionando, chamando a atenção do jogador, e ajudando a iluminar a cena.

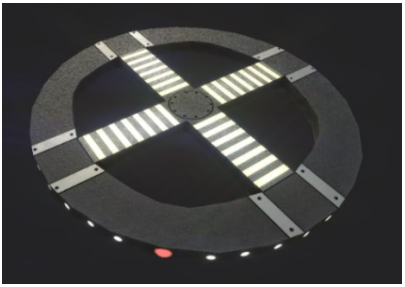
Após a ideia principal ter funcionado, a única coisa faltando era adaptar a textura para a nova ideia do asfalto, assim como foi feito na plataforma-base e exportar para *Unreal Engine 4*.

Figura 85 Mapa de exportação para UE4



Fonte: Autor, 2017

Figura 86 Textura vFinal Plataforma King of The Hill



Fonte: Autor, 2017

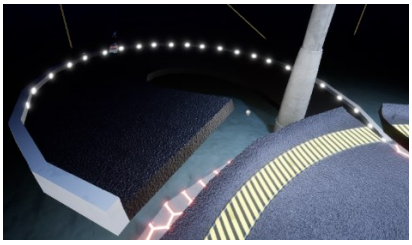
5.2.1 Rampas e Plataforma Inferior

Aqui, já se tem uma ideia cristalizada de como serão as rampas e a plataforma inferior, em função de toda a evolução do protótipo.

Logo no desenvolvimento da rampa, foi criada a sua parte principal e o suporte para evitar que o jogador caísse na água. A primeira foi feita de asfalto, seguindo a mesma lógica das plataformas, a parte de suporte é feita de metal, também seguindo a lógica na criação das estruturas de suporte das plataformas, com a adição das luzes que iluminam a plataforma.

Como o cenário é noturno e tudo está 100% escuro, os holofotes de iluminação estão mais para dentro dando foco nas plataformas principais, assim essas luzes servem de fonte secundárias para o jogador poder ver o mínimo possível da rampa e poder subir a plataforma principal.

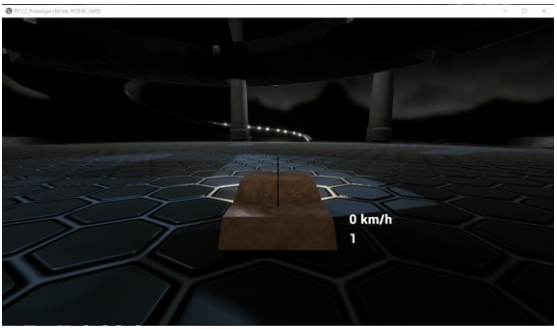
Figura 87 Rampa & Suporte Texturizada dentro da Unreal Engine 4



Fonte: Autor, 2017

Já na plataforma inferior foi criada uma estética diferente para seu resultado, pois era preciso chamar menos atenção para essa plataforma. Então, apenas uma simples plataforma com um material de hexágonos foi colocada em cena, sendo o suficiente para a plataforma conseguir se misturar com o cenário como um todo.

Figura 88 Plataforma Inferior dentro da Unreal Engine 4



Fonte: Autor, 2017

5.2.1 Objetos Secundários

Assim como na parte de modelagem, apenas os objetos secundários anteriores tiveram uma atenção extra. A base do poste foi criada com a intenção de parecer uma estrutura firme, mas já fragilizada por possíveis impactos de carros, por isso o poste contém detalhes de rachaduras e o holofote é todo de metal, mas onde a iluminação sai tem um mapa de luz emissiva em branco semelhante ao mapa emissivo utilizado nas plataformas principais.

Figura 89 Refletores texturizados dentro da Unreal Engine 4

Fonte: Autor, 2017

Figura 90 Base do Poste Texturizada dentro da Unreal Engine 4

Fonte: Autor, 2017

5.3 Iluminação e Pós-Processamento

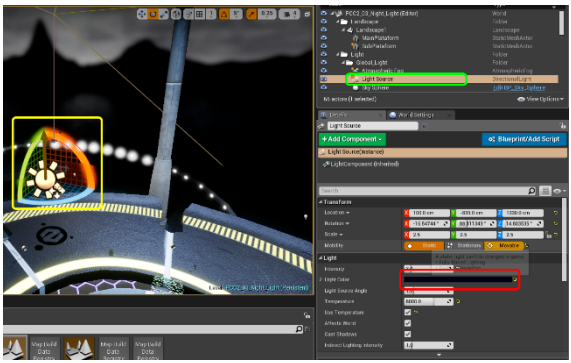
A iluminação foi projetada para ser um dos efeitos de estética mais importantes para que o cenário parecesse um evento televisionado para o mundo inteiro, normalmente shows assim contêm múltiplas fontes de iluminação, cores diferentes para cada carro, um espetáculo com fogos de artifícios e qualquer outro efeito de um show espetacular.

Como a ideia do projeto é limitado a um protótipo, considerou-se que essa parte faz parte do refinamento da parte estética do jogo completo e por isso essa área foi focada na parte básica da iluminação, que foi tornar o cenário jogável, no ambiente noturno proposto anteriormente.

Primeiro passo do desenvolvimento foi a mudança do ambiente diurno para o noturno e como *Unreal Engine 4* não possui um sistema automático de mudança de dia para a noite, ajustes manuais tiveram que ser feitos.

Primeira mudança foi selecionar o próprio sol da cena e mudar sua rotação para cima como indicado na imagem abaixo e mudar a luz transmitida pelo sol para a cor preta.

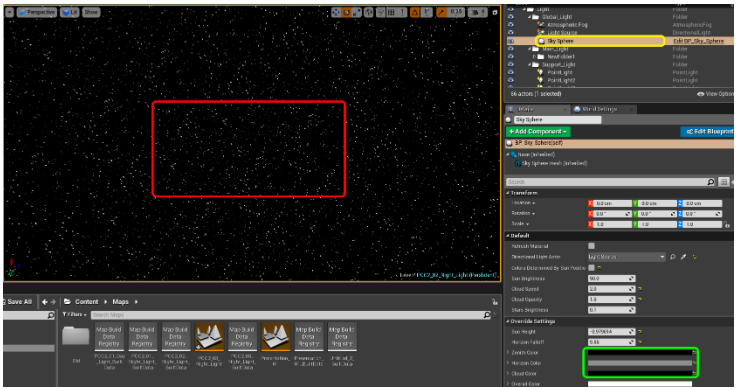
Figura 91 Configuração Light Source



Fonte: Autor, 2017

Próximo foi mudar a cor da *Sky Sphere*, objeto que engloba o cenário, fazendo as alterações nele como indicado na imagem abaixo e acabar fazendo a transformação final do cenário para a noite e inclusive fazendo aparecer estrelas no céu.

Figura 92 Configuração Sky Sphere

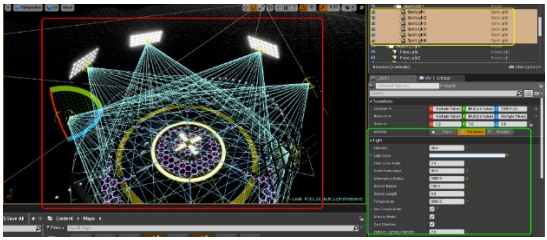


Fonte: Autor, 2017

Assim, o cenário está totalmente encoberto e escuro sem o jogador poder ver nada; seguindo o desenvolvimento foram adicionadas as luzes principais da cena, assim a fonte principal de iluminação vem da forte

emissão de luz dos holofotes. Para isso foram utilizados para cada holofote um *Spot Light* que funciona para simular uma luz sendo projetada de um ponto de origem, a ideia como mostra na imagem abaixo foi tentar encobrir as plataformas do cenário inteiro com essas *Spot Lights*.

Figura 93 Posicionamento das Spot Lights



Fonte: Autor, 2017

Alguns ajustes foram feitos nas *Spot Lights* como: o ângulo de amplitude para área que elas vão afetar e a mudança para *Stationary Light*, assim como ajustes na luz da cor por temperatura, que nesse caso foi 9000.

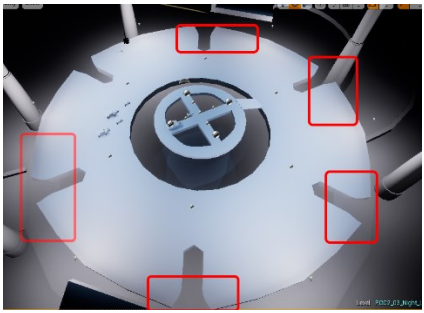
Mas como a ser visto na próxima imagem, apenas as *Spot Lights* não acabam dando o resultado visual desejado para o jogo, algumas áreas das plataformas acabaram ficando escuras e isso atrapalha o jogador, ainda mais uma plataforma que já é preta pelo asfalto, a notar onde termina a plataforma, ocasionando em quedas desnecessárias.

Figura 94 Área afetada pela iluminação da Spot Light (em desenvolvimento)

Fonte: Autor, 2017

Por isso é comum em jogos, onde a luz principal acaba iluminado pouco uma área do cenário, colocar-se *Point Lights* nas áreas mais escuras, para dar uma visão mais clara para o jogador. Na prática, não existe nenhuma fonte de luz para explicar essa iluminação, mas esses pequenos ajustes são necessários para criar uma melhor experiência ao usuário, mesmo que não pareça real. Com a adição de *Point Lights* acaba-se diminuindo o impacto das sombras e a falta de iluminação como visto na imagem a seguir.

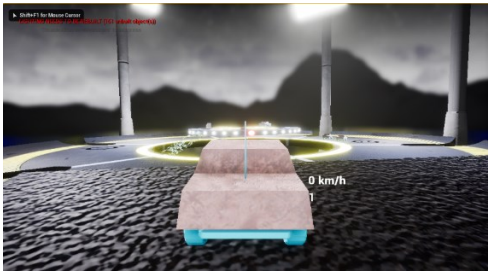
Figura 95 Cenário iluminado com a ajuda de Point Light (em desenvolvimento)



Fonte: Autor, 2017

Resultado final da combinação de ambas acaba resultando em um cenário mais iluminado e com menos partes escuras que só atrapalham o jogador como visto na imagem a seguir.

Figura 96 Iluminação do cenário demonstrando a iluminação (em desenvolvimento)



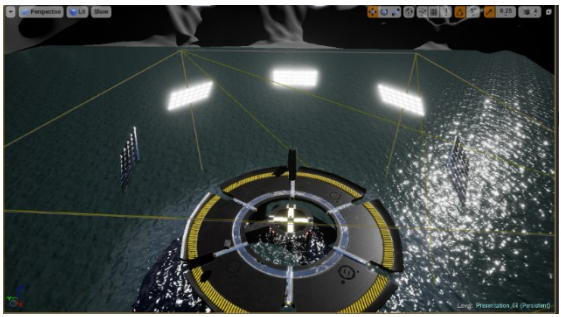
Fonte: Autor, 2017

Com a parte de iluminação montada, existem dois detalhes mais técnicos necessários para uma melhor performance do jogo pela *Unreal Engine 4* que envolvem duas opções.

A primeira é em relação a *Lightmass Capture Volume*, que impede que luzes indiretas sejam calculadas infinitamente pelo cenário e as contém apenas dentro de uma caixa, impedindo uma quantidade absurda de cálculos desnecessários.

Para isso, foi criada uma caixa em volta do cenário para indicar ao programa que luzes indiretas devem ser calculadas apenas dentro dessa área, e fora dela apenas uma vez, melhorando a performance do jogo.

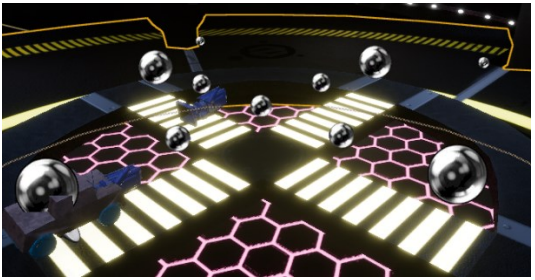
Figura 97 Caixa do Volume da Luz Indireta



Fonte: Autor, 2017

Segundo detalhe importante é o comportamento dos materiais no quesito de refletir a iluminação: dentro da *Unreal*, existe uma outra ferramenta chamada de *Sphere Reflection Capture* que permite dar ao objeto as propriedades de reflexão necessárias para um comportamento mais realístico, por exemplo dos materiais metálicos colocados nas plataformas e no holofote. Essa função tem muito pouco peso na performance do jogo porque todos os cálculos necessários feitos para o comportamento da cena são calculados antes do jogador dar *play* no jogo, logo é possível espalhar esse objeto pelo cenário inteiro sem comprometer a qualidade de jogo.

Figura 98 Sphere Reflection em cima da plataforma King of The Hill



Fonte: Autor, 2017

Após todo esse desenvolvimento, o que falta agora é acertar alguns detalhes, e para isso se utiliza um volume de pós-processamento que aplica um "filtro" na visão de quem estiver dentro do volume e que ajeita diversas opções dentro do cenário como *Depth of Field*, efeito de *Bloom,* alguns detalhes de iluminação global entre outras opções.

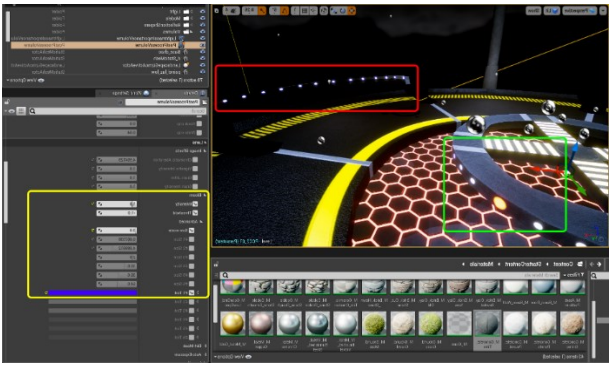
Primeiro ajuste foi feito no *Color Grading*, que ajuda a fazer algumas correções nas cores ou melhorar a qualidade de outras em certas áreas; nesse caso, pequenos ajustes foram feitos, como aumentar levemente o contraste e a saturação do azul, para dar uma maior ênfase as cores dos círculos presentes na plataforma.

Figura 99 Efeito da Caixa de Pós-Processamento 1

Fonte: Autor, 2017

Próximo ajuste foi o efeito das lentes colocadas na cena, como as lâmpadas nas rampas e a iluminação na plataforma principal e na plataforma-base.

Figura 100 Efeito da Caixa de Pós-Processamento 2

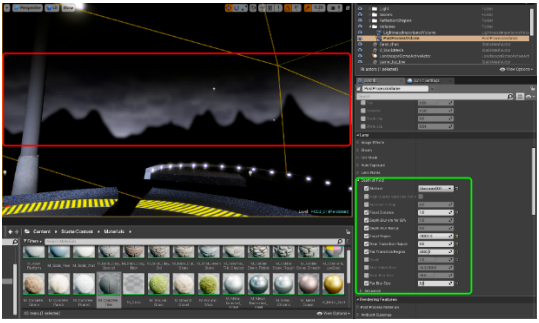


Fonte: Autor, 2017

Ajustes de intensidade do *bloom* e como ele se comportava foram as alterações feitas, além de ser incrementado todos os efeitos com um leve toque de azul, mais notáveis nas lâmpadas que eram originalmente brancas.

Últimos ajustes feitos nas lentes, foi a utilização do *Depth of Field*, esse efeito foi utilizado para borrar objetos a distâncias longas da plataforma, nesse caso as montanhas nas redondezas do local, e todas as opções feitas foram para melhorar a qualidade desse efeito de *blur* e ao mesmo tempo não afetar objetos dentro da própria cena quando o jogador estivesse jogando, por exemplo as rampas do outro lado do mapa estivessem com esse efeito durante o jogo.

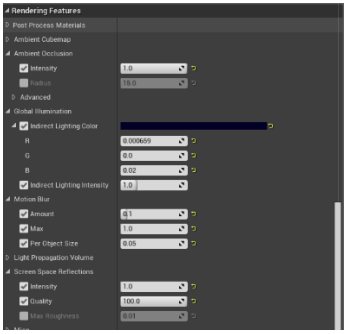
Figura 101 Efeito da Caixa de Pós-Processamento 3



Fonte: Autor, 2017

Por último, alguns efeitos na hora de renderização do jogo foram ajustados, o primeiro incluiu *Ambient Occlusion*, difícil de ser notado durante o jogo sem a pessoa estar ativamente procurando por ele, é um efeito que diz aproximadamente como a luz deve se comportar em certa parte do objeto baseado na luz e seu ambiente. O segundo efeito aplicado foi um efeito de *Global Illumination* e a cor aplicada vindo dele, próximo efeito foi de *Motion Blur*, esse mais notado quando o jogador dirigir o carro, é o efeito de "blur" quando qualquer objeto está em movimento, e sua quantidade e valores foram ajustados nessa caixa de valores e, por final *Screen Space Reflections*, outro ajuste que indica como as reflexões se comportam na cena, principalmente em relação aos metais.

Figura 102 Caixa de configuração do Pós-Processamento



Fonte: Autor, 2017

Com esses detalhes ajustados, a parte de estética do protótipo foi concluída, faltando apenas ajustar configurações de Mecânicas que incluem o carro e como o mesmo se comporta na mão do jogador.

5.4 *Mechanics*

Essa parte do desenvolvimento demonstrou como foram arrumadas as configurações de jogabilidade dentro do protótipo; essas opções incluíram a física, exemplo disso foi a gravidade que afeta o carro, ou como as rodas do carro se comportam e o quanto elas giram ou não quando o jogador faz uma curva, por exemplo.

Nessa parte do desenvolvimento, o MDA demonstra a parte mecânica e a dinâmica do jogo que fica por trás da estética do jogo, é pouco notável pelo ponto de vista do jogador, todas as opções e ajustes feitos nessa parte do desenvolvimento são mais claras para o desenvolvedor e a maioria de seus detalhes, o jogador não irá ver como funciona, principalmente no caso de usos de *blueprints*, o jogador verá o resultado dela, mas não todos os procedimentos para que tal evento aconteça.

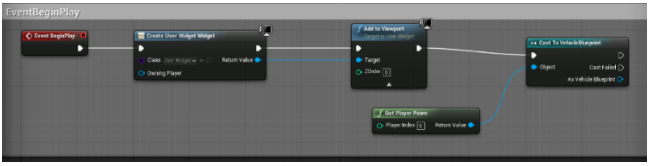
As opções foram muitas e diversos ajustes e testes foram feitos durante esse processo, tudo foi arrumado segundo as opções e ajustado o necessário até ter o resultado desejado, fazendo-se o uso de múltiplos *loops* pequenos, até que se chegasse ao resultado apropriado do protótipo.

Por parte final, foi necessária a criação de algumas *blueprints* que dizem aonde o jogador começa o jogo, o que acontece quando o carro sai da plataforma e outros eventos necessários para que o protótipo seja operacional.

Para facilitar os testes de customização dos controles do carro, foram feitas as modificações na *BluePrint* referentes ao carro e ao jogador para que em dada condição, o jogador fosse teletransportado para o ponto de origem onde começou o jogo. Essa modificação e criação na *Blueprint* tem dois objetivos, o primeiro como mecânica do jogo, caso o jogador seja derrubado para fora da plataforma ele “renasça” novamente dentro da mesma depois de um tempo, e o segundo caso aconteça algum erro durante o desenvolvimento do jogo, e o desenvolvedor fique ‘preso’ de alguma forma, essa *blueprint* também o levará para o ponto de origem.

Para criar-se esse processo inteiro, o primeiro passo foi criar uma *blueprint* que a *Unreal Engine 4* irá considerar como o jogador.

Figura 103 Bluprint Jogador

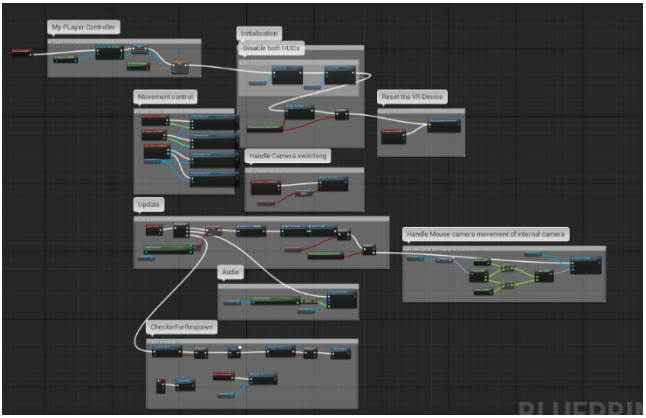


Fonte: Autor, 2017

Com essa *blueprint* temos o jogador funcionando perfeitamente dentro da cena, podendo andar com o carro e interagir com os objetos, apenas o que foi feito até aqui foi transformar o que antes estava em programação, para o sistema de *BluePrint* que facilita o desenvolvimento do restante.

Segundo passo foi adicionar esse novo controle dentro da *blueprint* do veículo para que ele possa reconhecer o novo jogador criado. Dentro da *blueprint* do veículo, a maioria dos eventos e configurações já vieram dentro do próprio projeto que esse protótipo está utilizando como base.

Figura 104 Blueprint Veículo



Fonte: Autor, 2017

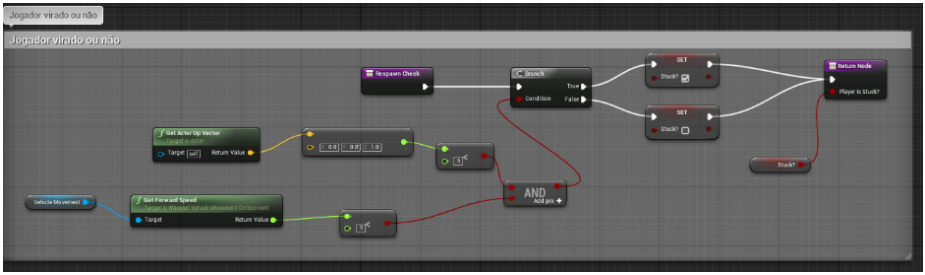
O que foi adicionado de novo foi o “My Player Controller” que tem como função pegar a posição de origem do jogador colocado dentro do cenário do protótipo e fazer a *blueprint* reconhecer esse local para

futuros usos. Essa função é necessária para saber onde recolocar o jogador na cena.

Segundo passo foi começar a produção no caso de alguma coisa acontecer, o jogador será teletransportado para esse ponto de origem. O primeiro obstáculo que acontece frequentemente é o jogador capotar o carro e não conseguir desvirá-lo, então quando isso acontecer, o jogo automaticamente irá 'destruir' o carro e trazê-lo de volta ao ponto de origem.

Para isso dentro da *blueprint* do veículo se criou um novo evento que se chamou de *Respawn Check*, que servirá para ver se o carro está virado de alguma forma ou não, basicamente ele checará o eixo Z do carro e se o valor for menor que 0.5, ele irá ativar o evento de destruir o carro e botar o jogador no ponto de origem.

Figura 105 Blueprint para checar se o carro capotou ou não

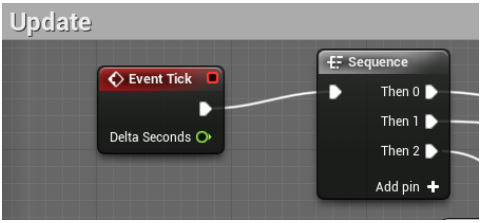


Fonte: Autor, 2017

Com esse evento criado, é necessário fazer a *blueprint* do veículo fazer a checagem, ou *update* da posição do carro, e caso o valor do eixo Z realmente esteja menor que 0.5, ele irá ativar a sequência.

Primeiro como visto na figura abaixo, tem uma parte chamada *Update*, que é a parte da *blueprint* que faz a checagem de diversos parâmetros e caso esses parâmetros mudem, ele ativa novos eventos.

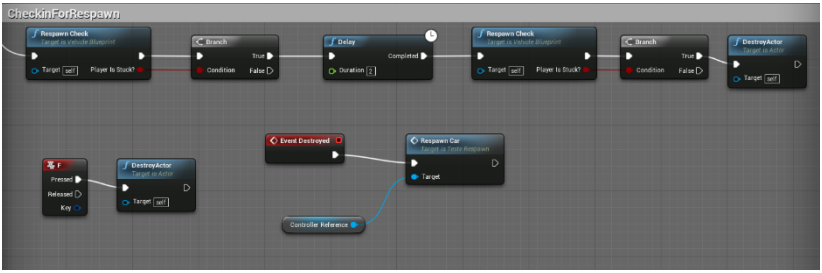
Figura 106 Blueprint Update de Parâmetros



Fonte: Autor, 2017

Nisso, adicionou-se um novo parâmetro, simbolizado na lista do *Sequence* como "Then 2" e, desse ponto de partida criou-se o evento necessário para checar se o carro realmente virou ou não.

Figura 107 Bluprint Evento Respawn & Destruição Veículo



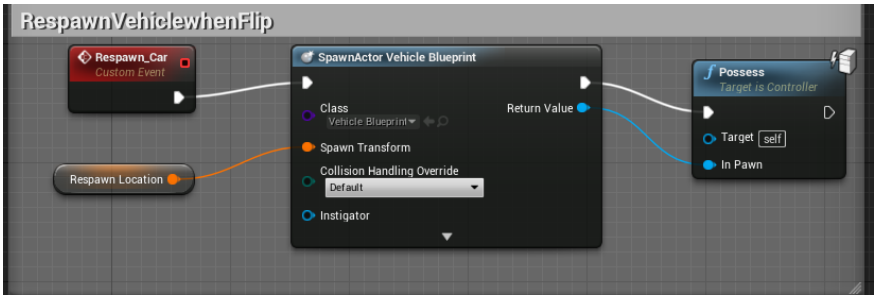
Fonte: Autor, 2017

Essa checagem dá ao jogador um tempo (nesse caso está colocado em 2 segundos), para ver se ele consegue colocar o carro de volta no eixo correto, caso contrário, ativará o evento de *DestroyActor*, que destrói o carro, fazendo com que o carro renasça na cena.

Junto nessa área se criou um pequeno evento, tipo teste e caso o carro fique preso por motivos desconhecidos, que é toda vez que a tecla "F" for apertada, o carro será destruído, como a *blueprint* está configurada para toda vez que o carro é destruído, voltar para o ponto de origem, mas nenhuma alteração foi necessária, essa *hotkey* serve mais para testes e futuros erros que possam vir a acontecer durante o desenvolvimento e teste do protótipo.

A última coisa faltando é fazer com que o controle do jogador seja colocado de volta para o carro quando ele volta para o ponto de origem, se não o carro volta, mas o jogador não tem controle dele. Para arrumar esse erro dentro da *BluePrint* criada para o jogador, adicionou-se um novo evento quando o carro dá *Respawn*, o jogador assume controle novamente.

Figura 108 Blueprint Respawn carro quando capotado

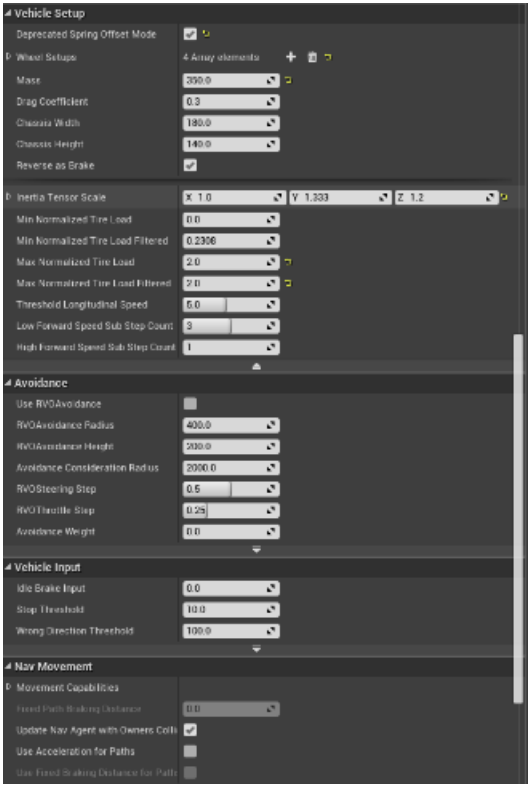


Fonte: Autor, 2017

Com a configuração da *blueprint* para fazer o jogador renascer na cena por diversos fatores, o último passo da parte de mecânica desse protótipo foi fazer pequenos ajustes nas mecânicas do carro que influenciam diretamente no *gameplay* do jogo e como o jogador interage com o carro e o cenário.

Para isso continua-se a modificar os parâmetros dentro da *blueprint* do veículo, mas como está a utilizar um *template* já criado pela *Unreal Engine 4*, esses parâmetros de modificação do carro já vêm prontos e o necessário apenas é mudar os valores desses parâmetros para surgir efeitos na jogabilidade do carro na cena.

Figura 109 Configuração Blueprint – Carro 1

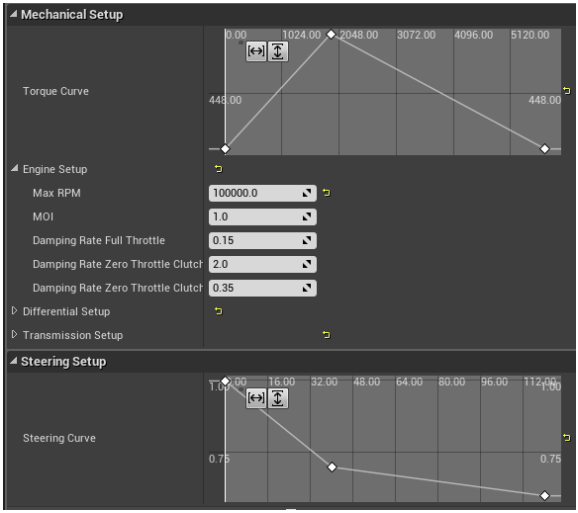


Fonte: Autor, 2017

Nessa parte do desenvolvimento não foi necessário inventar nenhum novo procedimento, o que foi feito foi o balanceamento de todos os valores da imagem anterior até chegar ao resultado necessário para o protótipo.

Independente dos valores numéricos, que sim ajudam a fazer pequenos ajustes no comportamento do carro, a parte que influenciou mais a modificação do carro foram os dois gráficos disponíveis na próxima figura.

Figura 110 Configuração Blueprint - Carro 2



Fonte: Autor, 2017

Foi modificando esses gráficos que o comportamento do carro teve um impacto maior nas mecânicas do jogo e como o jogador o controla durante a partida. Nesses gráficos foram feitos leves ajustes e isso influenciou o quanto o carro consegue virar na hora de fazer curvas, e seu comportamento no geral durante o jogo.

Essa parte do desenvolvimento foi de tentativa e erro, mas o objetivo das mudanças foi deixar o carro mais leve e rápido, mas ao mesmo tempo não o fazer perder a maneabilidade de poder fazer a curva e ter precisão de controlar o carro dentro da plataforma menor, onde os jogadores terão a maior disputa.

Com a mecânica do carro ajeitada, próximo e último passo foi criar o sistema de *multiplayer* para que outros três jogadores pudessem se juntar à partida e o protótipo se tornasse realmente jogável e testável com mais de um jogador.

Para fazer isso, se criou o menu do jogo, com alguns exemplos de opções que devem existir em um jogo, como tela de configurações e comando de controles existentes dentro do jogo.

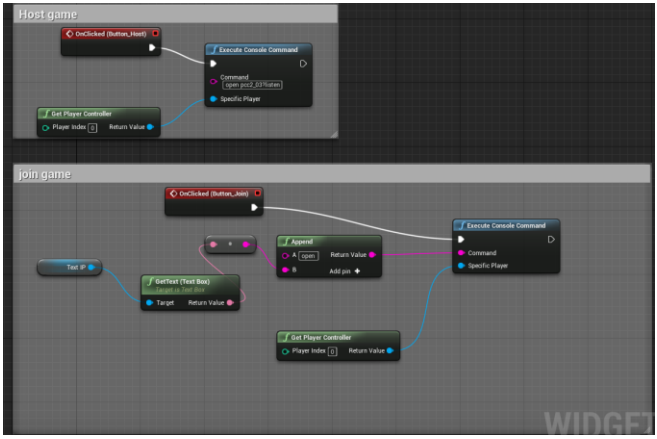
Figura 111 Menu de Jogo

Fonte: Autor, 2017

Host & Join Game fazem parte da configuração do *multiplayer, Options* é para abrir as opções de jogo, *Quit* é para fechar o jogo, e na direita estão as descrições dos comandos para controle do carro e de eventos criados para testes dentro do protótipo.

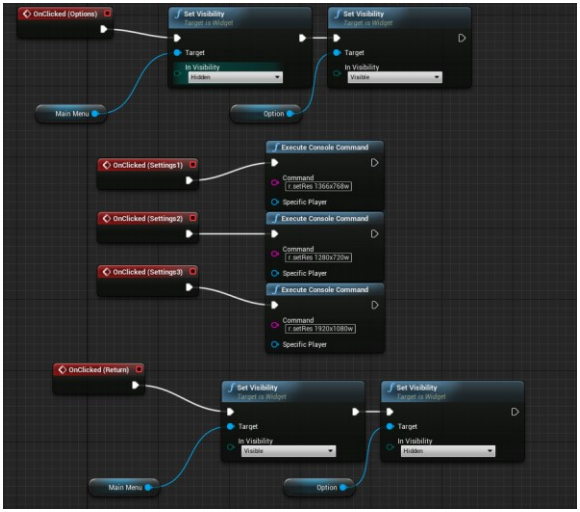
Na criação do *multiplayer* e do menu, fica evidente os dois lados do desenvolvimento de um jogo, do jogador & desenvolvedor, o jogador vê apenas o menu, a estética dele, sua aparência e o que ele faz e para onde leva o jogador, já o desenvolvedor vê como realmente faz funcionar esse menu com *multiplayer* como nas imagens a seguir.

Figura 112 Blueprint Multiplayer



Fonte: Autor, 2017

Figura 113 Blueprint Menu



Fonte: Autor, 2017

Dessa forma, o protótipo está 100% jogável, com *multiplayer* de até quatro pessoas funcionando, com um menu para entrada e saída de jogo, com o carro sendo jogável e possível de se disputar o título de *King Of The Hill.*

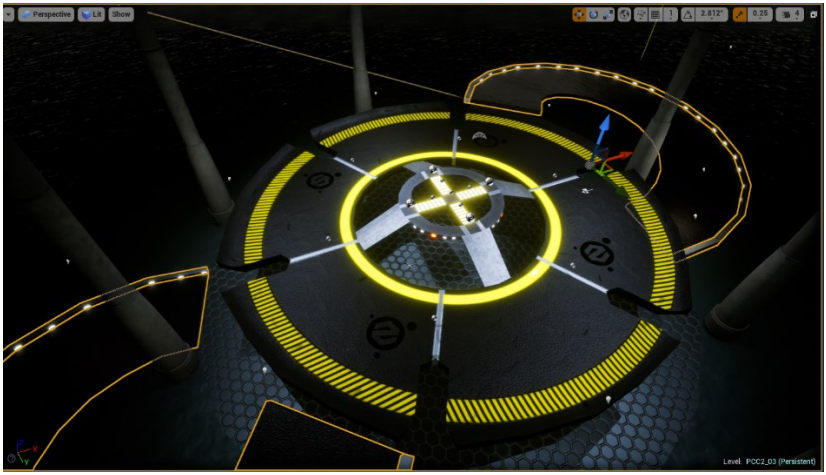
6 *EXECUTIVE SUMMARY* VERSÃO FINAL

O primeiro *executive summary* foi desenvolvido quando a ideia do projeto estava em fase de criação de conceitos e ideias, sendo que a maioria delas não haviam sido testadas; com a finalização do produto, pôde-se atualizar o sumário para ter uma indicação do que o produto final realmente é.

Titulo	*King of the Hill Arena*
Gênero	*Multiplayer, Racing, Sport, Online, Fast-Paced, Casual, Indie, Competitive, Co-op.*
Plataformas	Jogo desenvolvido na *Unreal Engine 4*, com finalidade de rodar independente no sistema *Windows* 10.
Tecnologia	Uso da ferramenta *Unreal Engine 4* versão 4.15.3 disponibilizada de graça pela empresa *Epic Games* para a criação de jogos. Ferramenta utilizada pelo enorme suporte da comunidade e dos desenvolvedores para a ferramenta.
Referências	*Rocket League* e *Monster Truck Madness 2*
Conceito	Jogo tem como objetivo criar experiências utilizando uma atmosfera competitiva entre os jogadores onde apenas um deles poderá sair vencedor. Localizado em uma plataforma complexa localizada no meio do oceano, o evento da disputa será televisionado (*Não incluído no protótipo), os jogadores devem disputar quem consegue a maior quantidade de pontos na plataforma mais alta do complexo e no final ser coroado o *King of the Hill.*
Tema	Competição pelo título de *King of the Hill.*
Ambientação	Cenário 3D, localizado em um conjunto no meio do oceano, onde foi construído um complexo de plataformas. O mar do cenário está agitado, devido a

	possibilidade de tempestade, o tempo também está nublado, com possibilidade de chuva. (*Não incluído no protótipo) Existem três plataformas, onde a menor fica no ponto mais alto do mapa, a segunda dá acesso a essa plataforma principal, e a última evita que os jogadores sempre caiam direto na água.
Estéticas	Jogo terá uma aparência com tendências realísticas, com a maioria dos objetos dentro de uma proporção realista, levando em consideração as possibilidades permitidas pela *Unreal Engine 4.*
Som	Jogo terá uma trilha sonora para dar ênfase à atmosfera de disputa entre os jogadores. Inclui também sons pela utilização de turbo e ao choque entre carros, além dos sons conhecidos feitos por um carro na realidade, como o barulho do motor aquecendo. *Não incluído no protótipo.
Objetivos	Jogador terá como objetivo competir contra até 3 jogadores (total de 4), pelo maior tempo possível de permanência na plataforma mais alta do cenário, quanto maior tempo ele permanecer, maior número de pontos ele fará e aumentará sua chance de ganhar o jogo.
Objetivos Secundários	Possibilidade de derrubar jogadores para plataformas inferiores ou até para o mar e de se utilizar partes do mapa para criação de estratégias que levem a vitória.
Mecânicas	Tempo de 5 a 10 Minutos por partida Habilidade de controlar um carro. Atingir alta velocidade com o carro. Possibilidade de chocar-se com inimigos Andar pelas bordas das plataformas Ajuste de câmera em relação ao carro.

Figura 114 Cenário Protótipo vFinal

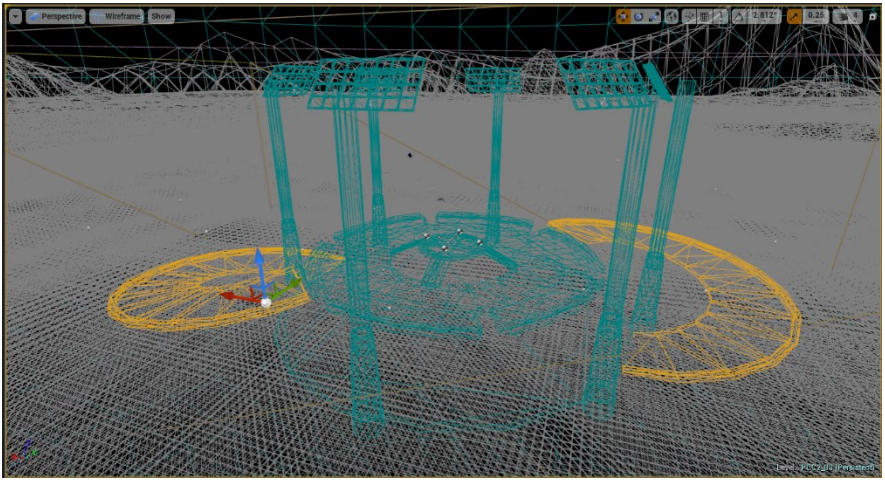


Fonte: Autor, 2017

Figura 115 Carro Protótipo para Teste

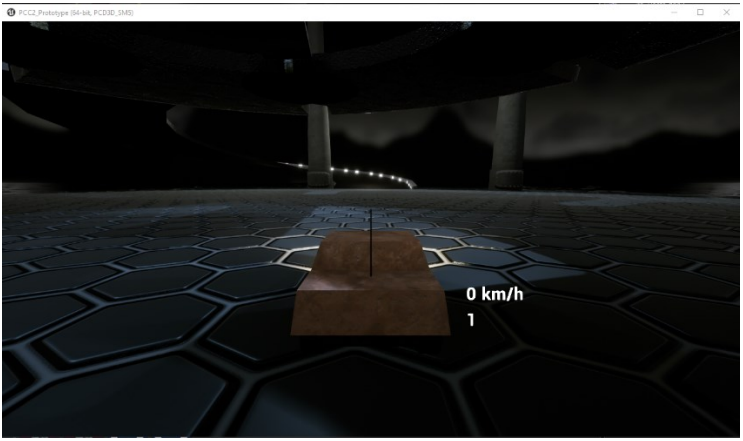
Fonte: Autor, 2017

Figura 116 Cenário visão Wireframe



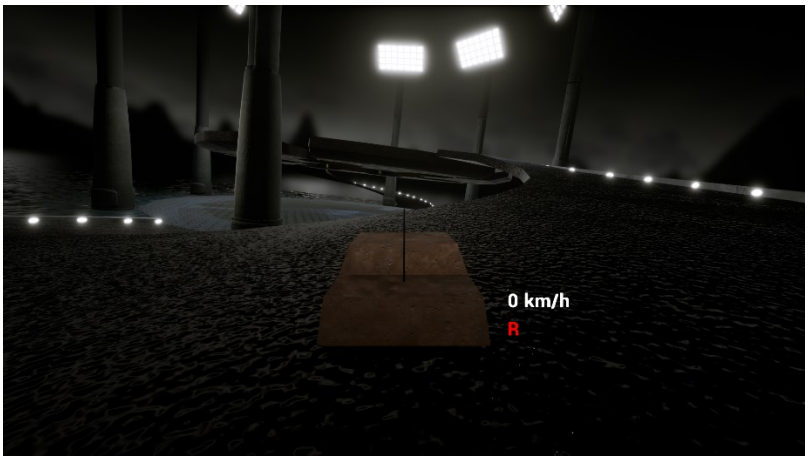
Fonte: Autor, 2017

Figura 117 Visão Plataforma Inferior



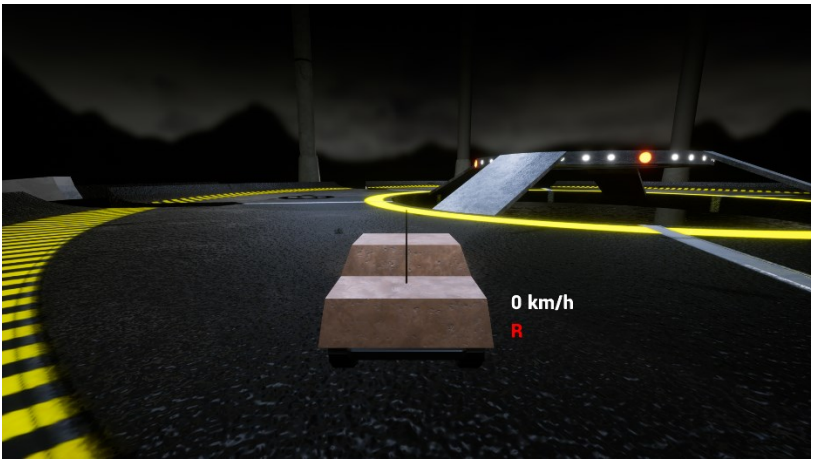
Fonte: Autor, 2017

Figura 118 Visão Rampa



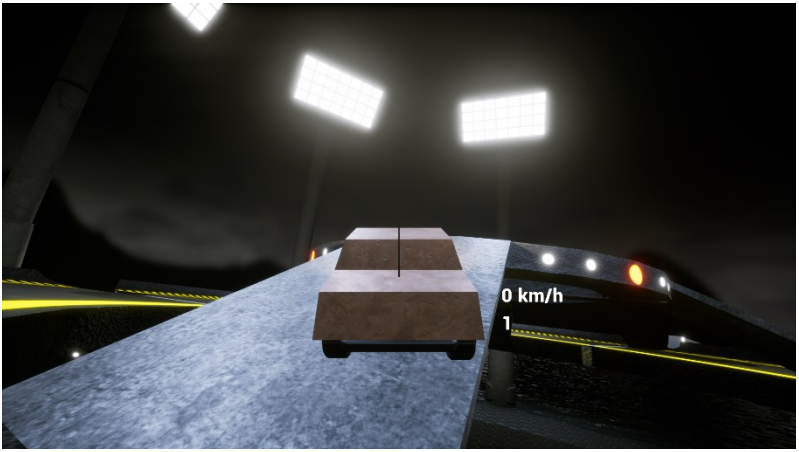
Fonte: Autor, 2017

Figura 119 Visão Plataforma-Base



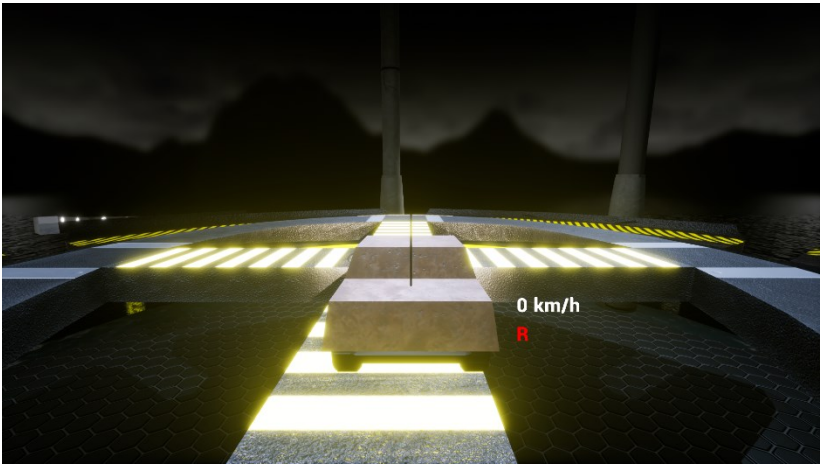
Fonte: Autor, 2017

Figura 120 Visão rampa acesso Plataforma King of The Hill



Fonte: Autor, 2017

Figura 121 Visão Plataforma King of The Hill



Fonte: Autor, 2017

7 CONCLUSÃO

Após um semestre de desenvolvimento no protótipo proposto no início do projeto, notou-se a importância da criação de uma metodologia e da influência que elementos do *game design* produzem na qualidade final do produto.

O projeto foi guiado pela metodologia proposta e, atendeu os seus objetivos que eram produzir um protótipo de um jogo baseado nos conceitos de Game Design.

Levando-se em consideração a metodologia foi possível fazer ajustes durante o desenvolvimento. Tendo isso em mente, diversos ajustes deram certos, outros nem tanto e em partes do desenvolvimento se encontraram muitos problemas.

Considerando-se isso funcionaram o uso da metodologia no geral como forma de guiar e nortear as decisões tomadas durante o processo de desenvolvimento do jogo, conforme o seguinte *check-list* das ações positivas realizadas:

- ✓ Fica clara a visão que o desenvolvedor tem do jogo, e como o jogador tem a experiência ao jogar;
- ✓ Sistema de *Loops* aplicado em todos os aspectos de decisões e construções de *assets* e elementos pertinentes ao projeto, demonstrou a importância de construir e descontruir é importante para que a escolha final seja melhor resultado possível, demonstrado pelas inúmeras versões de *assets* criados durante o desenvolvimento e decisões tomadas como a exclusão da ilha;
- ✓ Protótipo desenvolvimento é 100% jogável, inclusive com multiplayer para até 4 jogadores no total;

Continua-se ainda com os elementos citados durante a metodologia que nortearam as áreas de desenvolvimentos, *Story, Mechanics, Aesthetics e Tecnology e Thema*:

- ✓ Da parte de tecnologia da *Unreal Engine 4*, facilitou o processo de desenvolver um novo jogo dentro de outro já existente por já ter disponível as ferramentas e constarem muitos detalhes já prontos, como por exemplo: o carro funcionando, precisando apenas de ajustes para o protótipo;
- ✓ Da mecânica do jogo, foi possível modificar o carro, tanto sua aparência, quanto o modo pelo qual ele se comporta na cena e

seu controle pelo jogador, da forma que se imagina que o produto final poderá vir a ser - atingindo o objetivo do protótipo de demonstrar essa parte do jogo;

- ✓ A parte de estética, ficou clara a direção na qual o jogo final terá em termos de aparências e elementos figurativos do cenário;
- ✓ O protótipo, da forma como o cenário foi construído, com a plataforma inferior, a plataforma-base e a plataforma mais alta, e com sua estética, ajudam a reforçar o tema do jogo, que é o título de *King Of The Hill*;
- ✓ Apesar de não estar 100% pronto em nenhuma das quatro áreas propostas, o protótipo consegue deixar claro a direção em que cada uma dessas áreas têm visando o jogo completo e finalizado.

Apresenta-se a seguir, o seguinte check-list que diz respeito ao que não deu certo ou não atendeu as expectativas e/ou problemas apresentados durante o desenvolvimento:

- ✓ Apesar do sistema de *Loops* ter funcionado, pela falta de tempo disponível, não foi possível fazer uso desse sistema mais vezes, e em partes do desenvolvimento decisões foram tomadas mais cedo para não se ficar preso em uma parte e atrasar o resto;
- ✓ Grande parte da estética do jogo foi sacrificada para que focar na parte de Mecânica do jogo, por exemplo: durante o desenvolvimento se escolheu deixar o carro em um estado mais básico visualmente para poder focar em como o jogador controla o carro e como o carro se comporta dentro do cenário e nas físicas do jogo, o que consequentemente acabou sacrificando um pouco do elemento *Story*, já que faltaram elementos estéticos que dessem suporte.
- ✓ O ambiente de trabalho proposto entre a interação dos programas *3Ds Max, Photoshop, Substancie Painter e Unreal Engine 4,* apesar de todos serem necessários, o fato do sistema de *Loop* muitas vezes ter que passar pelos quatro programas, acabou consumindo um tempo acima do esperado e restringindo o tempo.
- ✓ Por mais que a *Unreal Engine 4* tenha possibilitado muitas coisas de forma mais fácil e simples, problemas técnicos foram encontrados durante o desenvolvimento, como por exemplo durante a criação do *multiplayer* onde ocorreram vários problemas de controle e erros, já durante a iluminação do cenário ficou pesado demais e acabou comprometendo a

performance do jogo e entre outras áreas, e cada um desses erros foi um tempo que poderia ser utilizado para mais *loops* ou testes em outras áreas, mas que foi gasto para arrumar os problemas ocasionados pelo programa.

Com a conclusão do primeiro grande *loop* feito na criação doo *mock-up*, o segundo grande *loop* foi finalizado e apresentado durante o PCC2 e resultou no protótipo visto anteriormente. Por isso, pensando no terceiro grande ciclo de produção do jogo, para se chegar a um melhor resultado e baseado no *feedback*, percebeu-se alguns pontos devem ser levados em consideração:

I. Tecnologia:

- ✓ Optimização do jogo para rodar em máquinas mais fracas.

II. Mechanics:

- ✓ Com a inclusão do *multiplayer*, mecânicas do carro como suas colisões e controle, problemas nessas áreas ficaram mais explícitos e ajustes terão que serem feitos baseados nesses novos *feedbacks;*
- ✓ Possível inclusão de outros tipos de carros com características diferentes, por exemplo um carro maior só que mais lento e difícil de ser derrubado, outro carro menor, só que mais rápido e extremamente frágil;
- ✓ Efeitos meteorológicos que podem ter na partida como por exemplo chuva intensa e neve;
- ✓ Contador de tempo de partida e *score* de cada jogador;
- ✓ Execução de eventos de penalidades, como por exemplo quando o jogador cair na água.

III. Aethetics & Story

Nessa parte, muitas das mudanças estéticas a serem feitas vão diretamente influenciar e dar suporte a história do jogo.

- ✓ Adicionar mais propagandas ao redor do jogo, como possíveis telões, além de adicionar mais propagandas ainda nas próprias plataformas;

- ✓ Melhorar a qualidade de algumas texturas e consertar a proporções de outras, principalmente do asfalto;
- ✓ Melhorar a iluminação do cenário que está precária em algumas partes do protótipo, como por exemplo nas rampas de acesso à plataforma-base, problema também resolvido com a adição de telões de propaganda como mais uma fonte de iluminação do cenário, e adicionar holofotes ao redor do cenário, melhorando o "show de luzes" que um espetáculo televisionado deve ter;
- ✓ Adicionar os efeitos previstos anteriormente de ambiente, como forte chuva, tempestade, raios, possível névoa, neve etc.;
- ✓ Criação do cenário ao redor da plataforma, como navios passando, possíveis animais pulando da água, em vez de apenas montanhas desfocadas;
- ✓ Criação estética do carro que o jogador controla na cena;
- ✓ Criação de água realista para o cenário em vez de apenas uma textura que simula a aparência de água. Podendo customizar a água de acordo com o ambiente proposto (tempestade, neve).

Tendo em vista os aspectos observados, sem a metodologia proposta, a conclusão do trabalho não teria alcançado a qualidade do protótipo atual, visto que além dos ajustes necessários, surgiram outros elementos e variáveis que foram importantes na maneabilidade do projeto.

E, para finalizar, espera-se que o protótipo e o relatório deixado, consigam servir de exemplos e referências para futuros alunos que estejam interessados na área de Games no Design, e que aqui encontrem o material desejado para pesquisa.

REFERÊNCIAS

CARDOSO, R. **Uma introdução à história do design.** São Paulo: Bloucher, 2008.

DILLE, F.; PLATTEN, J. Z. **The Ultimate Guide To Video Game Writing And Design**. New York: Crow Group, 2014.

GOLDBERG, H. Ken Levine talks about his new video game. **The News York Times.** New York, 21 mar. 2013. Disponível em: <http://artsbeat.blogs.nytimes.com/2013/03/21/bioshock-infinite-ken-levine-talks-about-his-new-video-game/?_r=1> Acessado em set – out 2016.

HUNICKE, R.; LEBLAC, M.; ZUBEK, R. "MDA: A Formal Approach to Game Design and Game Research. In: **Game Developers Conference, San Jose**, 2001. Disponível em: <http://www.mccormick.northwestern.edu/eecs/computer-science/>. Acessado em out. /Nov. 2016.

MANCUSO, F. Mercado de Games fatura cerca de 1 bilhão de dólares por ano no Brasil. **Jornal da Globo.** São Paulo, 08 out. 2015.Disponível em: <http://g1.globo.com/jornal-da-globo/noticia/2015/10/mercado-de-games-fatura-cerca-de-us-1-bilhao-por-ano-no-brasil.html> Acessado em set – out 2016.

MORRISON, A. The making of Rocket League. **Rocket Paper Shotgun.** 02 set. 2015. Disponível em: <https://www.rockpapershotgun.com/2015/09/02/rocket-league-making-of/> Acessado em set – out 2016.

NEWZOO. Global games Market reaches 99.6 billion 2016 mobile generating. **Newzoo**, San Francisco, 21 abr. 2016. Disponível em: <https://newzoo.com/insights/articles/global-games-market-reaches-99-6-billion-2016-mobile-generating-37/> Acessado em set – out 2016.

SCHELL, J. **The Art of Game Design:** a Book of Lenses. USA: Elsevier, 2015.

SILVA, Eli Lopes da. **Elaboração de trabalhos acadêmicos:** normas, dicas e erros comuns. Florianópolis: Ed. Do Autor, 2016.

VIDEO GAMES: The Movie. Direção e Produção: Jeremy Snead. Estados Unidos: Jeremy Snead, 18 Jul. 2014. Disponível em: <http://netflix.com> Acessado em set. 2016.

WALLMAN, J. "It's Only A Game" Game Design Methodology. In: **CLWG Fourth Annual Conference**, 1995. Disponível em: <http://www.jimwallman.org.uk/> Acessado em out. - nov.2016.

WAWRO, A. Why some old designs are Worth revisiting: a Rocket League story. 21 jul. 2015. Disponível em: <http://www.gamasutra.com/view/news/248925/Why_some_old_designs_are_worth_revisiting_A_Rocket_League_story.php> Acessado em set. – out. 2016.

WORTH, P. Bioshock: the Art Deco design of Rapture. 28 mar. 2013. Disponível em < http://www.thunderboltgames.com/feature/bioshock-the-art-deco-design-of-rapture> Acesso em set. – out. 2016.