UNIVERSIDADE FEDERAL DO CEARÁ

CENTRO DE CIÊNCIAS

DEPARTAMENTO DE COMPUTAÇÃO

PROGRAMA DE MESTRADO E DOUTORADO EM CIÊNCIA DA COMPUTAÇÃO

MESTRADO ACADÊMICO EM CIÊNCIA DA COMPUTAÇÃO

ÍCARO DA SILVA BARBOSA

FRAMEWORK PARA DESENVOLVIMENTO DE TACTICAL SHOOTERS EM REALIDADE VIRTUAL

FORTALEZA

2024

ÍCARO DA SILVA BARBOSA

FRAMEWORK PARA DESENVOLVIMENTO DE TACTICAL SHOOTERS EM REALIDADE VIRTUAL

Dissertação apresentada ao Curso de Mestrado Acadêmico em Ciência da Computação do Programa de Mestrado e Doutorado em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação.

Orientador: Prof. Dr. Creto Augusto Vidal.
Coorientador: Prof. Dr. Joaquim Bento Cavalcante Neto.

FORTALEZA

2024

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

B213f Barbosa, Ícaro da Silva.
Framework para desenvolvimento de tactical shooters em realidade virtual / Ícaro da Silva Barbosa. – 2024.
80 f. : il. color.

Dissertação (mestrado) – Universidade Federal do Ceará, Centro de Ciências, Programa de Pós-Graduação em Ciência da Computação, Fortaleza, 2024.
Orientação: Prof. Dr. Creto Augusto Vidal.
Coorientação: Prof. Dr. Joaquim Bento Cavalcante Neto.

1. Realidade virtual. 2. Tactical Shooters . 3. Framework. 4. Jogos digitais. I. Título.
CDD 005

ÍCARO DA SILVA BARBOSA

FRAMEWORK PARA DESENVOLVIMENTO DE TACTICAL SHOOTERS EM REALIDADE VIRTUAL

Dissertação apresentada ao Curso de Mestrado Acadêmico em Ciência da Computação do Programa de Mestrado e Doutorado em Ciência da Computação do Centro de Ciências da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Ciência da Computação. Área de Concentração: Ciência da Computação.

Aprovada em: 23/08/2024.

BANCA EXAMINADORA

Prof. Dr. Creto Augusto Vidal (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Joaquim Bento Cavalcante Neto (Coorientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Antonio José Melo Leite Júnior
Universidade Federal do Ceará (UFC)

Prof. Dr. George Allan Menezes Gomes
Universidade Federal do Ceará (UFC)

Prof. Dr. Alysson Diniz dos Santos
Universidade Federal do Ceará (UFC)

Aos meus pais e irmão, que me apoiaram, investiram e se alegraram em todas as conquistas desde o início da minha vida. Cada minuto, paciência, lágrima, sorriso, preocupação e orgulho que tiveram por mim me permitiram realizar esta jornada, sabendo que não estou sozinho e que sempre poderia contar com todos vocês.

Aos meus irmãos de vida, Franklyn Seabra, Marcos Felipe, João Castelo Branco e Franklin Alves, agradeço por todo o apoio que me deram, por toda paciência e por terem acreditado em mim. Todos os vossos conselhos e apoio me foram muito valiosos.

A todos que estiveram comigo de alguma forma em minha vida, meu eterno obrigado!

AGRADECIMENTOS

À Instituição Capes, pelo apoio financeiro com a manutenção da bolsa de auxílio e agradeço ao Conselho Nacional de Desenvolvimento Científico e Tecnológico pelo financiamento parcial da presente pesquisa através do Projeto CNPq Universal 406318/2023-9.

Ao Prof. Dr. Creto Augusto Vidal, pela excelente orientação e por todos os conselhos e apoios me dados ao longo de minha jornada e a paciência em tudo.

Aos professores participantes da banca examinadora Prof. Dr. Joaquim Bento Cavalcante Neto, Prof. Dr. Antônio José Melo Leite Júnior, Prof. Dr. George Allan Menezes Gomes e Prof. Dr. Alysson Diniz dos Santos pelo tempo, pelas valiosas colaborações e sugestões.

Aos colegas da turma de mestrado, pelas reflexões, críticas e sugestões recebidas.

"Buscar e aprender, na realidade, não são mais do que recordar". (Platão, 1578, p. 81d.)

RESUMO

Nos últimos anos, as tecnologias de Realidade Estendida tiveram um grande desenvolvimento, com lançamentos de Head-Mounted Displays mais acessíveis e poderosos computacionalmente, permitindo simulações mais elaboradas do que antigamente. No mercado de jogos, um dos gêneros mais lucrativos atualmente é o de Shooters. Entre os dez jogos com maior renda bruta na Steam em 2023, cinco são Shooters. O desenvolvimento de jogos é uma atividade desafiadora que demanda tempo e esforço, e por isso necessita de frameworks que auxiliem na redução do esforço e do tempo de desenvolvimento, aumentando a manutenibilidade do código e facilitando sua compreensão. Como há poucos frameworks para o desenvolvimento de Tactical Shooters em Realidade Virtual na literatura, este trabalho tem como objetivo propor um framework para esses jogos. Para isso, percorreram-se todas as etapas de criação de um framework: definição do modelo, definição da arquitetura e desenvolvimento do framework. Também foram criados três protótipos: dois baseados na arquitetura e um utilizando diretamente o framework. Para validar a usabilidade do framework, foi realizado um workshop com desenvolvedores de jogos, seguido de uma entrevista em grupo focal com todos os participantes. Como resultado, espera-se fomentar a criação de novos trabalhos que utilizem Realidade Virtual e frameworks para Tactical Shooters.

Palavras-chave: tactical shooters; realidade virtual; RV; framework.

ABSTRACT

In recent years, Extended Reality technologies have seen significant development, with the release of more affordable and computationally powerful Head-Mounted Displays, allowing for more elaborate simulations than before. One of the most lucrative genres in the gaming market is Shooters, with five out of the top ten grossing games on Steam in 2023 being Shooters. As game development is a challenging activity that demands time and effort, frameworks are needed to help reduce development time and effort, increase code maintainability, and make it easier to understand. Given the scarcity of frameworks for developing Tactical Shooters in Virtual Reality in the literature, this work aims to propose a framework for these games. All the steps for creating a framework were followed, including model definition, architecture definition, and framework development. Three prototypes were developed: two based on the architecture and one utilizing the framework. To validate the framework's usability, a workshop with game developers was held, culminating in a focus group interview with all participants. As a result, it is intended to promote the creation of new works that use Virtual Reality and frameworks for Tactical Shooters.

Keywords: tactical shooters; virtual reality; VR; framework.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

FPS	First Person Shooter
GQM	Goal, Question, Metrics
HMD	Head-Mounted Display
OXREF	Open XR for Education Framework
RA	Realidade Aumentada
RE	Realidade Estendida
RV	Realidade Virtual
SUS	System Usability Scale

SUMÁRIO

1 INTRODUÇÃO

Esta dissertação propõe um framework de software para auxiliar na implementação de Tactical Shooters de Realidade Virtual (RV).

1.1 Contextualização e justificativa

Com o crescente avanço das tecnologias de RV e a redução de custos dos equipamentos utilizados, como Head-Mounted Display (HMD) (Kelly *et al.*, 2022), essas tecnologias se tornam mais acessíveis para uso doméstico. Esse avanço possibilita o desenvolvimento de novas experiências, com mais graus de liberdade e melhor visualização 3D de ambientes reais (Banquiero *et al.*, 2023). Dessa forma, é possível perceber o crescimento de um mercado emergente para o desenvolvimento de jogos específicos em RV (Isabell *et al.*, 2020; LaValle, 2023).

Tactical Shooters são um dos gêneros mais lucrativos em plataformas como Steam[1] (Seção 2.2.1) e por buscar representar de forma mais realística possível os combates e funcionalidades dos equipamentos militares (definido na Seção 2.3), este gênero pode aproveitar das possibilidades de experiências proporcionadas pelas tecnologias de RV, aumentando a imersão do jogador[2] (Adams, 2014b).

Dado o caráter emergente do mercado de RV, há a necessidade de desenvolvimento de pesquisas em diversas áreas, incluindo a de Tactical Shooters em VR, já que, as pesquisas atuais avaliam apenas aspectos únicos de Tactical Shooters em RV (Monteiro *et al.*, 2020). Assim, a realização de pesquisas e frameworks de software que auxiliem o desenvolvimento e levem em conta as particularidades dessas plataformas se tornam importantes, por permitirem que o desenvolvedor tenha a liberdade de utilizar as ferramentas específicas dessa tecnologia (Wang; Nordmark, 2015), proporcionando melhor manutenção, reutilização e compartilhamento de módulos, fatores que muitas vezes impedem o avanço de projetos em RV (Kluge *et al.*, 2022).

Neste contexto, o presente trabalho visa propor um framework de software que auxilie no desenvolvimento de Tactical Shooters para RV, proporcionando uma melhor gestão da complexidade e modificabilidade do jogo.

[1] https://store.steampowered.com/

[2] Capacidade de se imergir no mundo virtual, tendo sensação de presença

1.2 Objetivos gerais e específicos

1.2.1 Objetivos Gerais

Propor um framework de software que auxilie no desenvolvimento de Tactical Shooters para RV, de forma que seja simples de utilizar, permita o desenvolvimento das ferramentas necessárias e seja fácil de modificar.

1.2.2 Objetivos Específicos

- Identificar os principais pontos para o desenvolvimento de Tactical Shooters;
- Definir o modelo de Tactical Shooters em RV;
- Definir a arquitetura de Tactical Shooters em RV;
- Desenvolver protótipos para validação da arquitetura;
- Validar os protótipos através de entrevistas semi-estruturadas com usuários;
- Definir o framework de Tactical Shooters em RV;
- Desenvolver um protótipo utilizando o framework; e
- Validar o framework com a apresentação de um workshop e grupo focal com desenvolvedores.

1.3 Abordagem metodológica

O presente trabalho foi dividido em duas fases: experimentação e refinamento. Essas fases foram necessárias devido à quantidade de atividades envolvidas no desenvolvimento de um framework. Na fase de experimentação, os principais elementos de um Tactical Shooter foram elencados, mapeados e validados, sendo esta a fase conceitual. Na fase de refinamento, o framework proposto foi construído e validado, sendo esta a fase prática.

A primeira fase foi dividida em coleta na literatura dos principais pontos para o desenvolvimento de Tactical Shooters. Com esses pontos, foi possível desenvolver o modelo de software, que serviu de base para o desenvolvimento da arquitetura de software. Utilizando essa arquitetura, foram desenvolvidos dois protótipos, que passaram por validação através de entrevistas semi-estruturadas com usuários.

Na segunda fase, a partir dos resultados obtidos na arquitetura de software, foi desenvolvido o framework. Após isso, foi criado um novo protótipo e realizado um workshop

com desenvolvedores, seguido de entrevista em grupo focal para validar a usabilidade do framework.

1.4 Organização da dissertação

Este trabalho está dividido em seis capítulos:

- Capítulo 2: Fundamentação Teórica - Apresenta os principais conceitos necessários para a compreensão da proposta e a base teórica desta pesquisa.
- Capítulo 3: Trabalhos Relacionados - Discute os trabalhos acadêmicos anteriores relacionados ao tema proposto.
- Capítulo 4: Metodologia - Apresenta a teoria e as validações da proposta, explicando as fases necessárias para a concepção dos resultados e os métodos utilizados.
- Capítulo 5: Desenvolvimento - Apresenta o desenvolvimento das fases apresentadas no capítulo 4, apresentando as atividades e discutindo seus resultados.
- Capítulo 6: Resultados - Relaciona e compara os resultados obtidos na pesquisa com os trabalhos anteriores apresentados no Capítulo 3.
- Capítulo 7: Conclusão - Resume a dissertação, pontuando os resultados obtidos, os trabalhos futuros, as limitações observadas e as contribuições esperadas.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo, é desenvolvida a fundamentação teórica do presente trabalho, explicando os principais conceitos e fundamentando os motivos da realização desta pesquisa.

2.1 O que são jogos digitais

Há diversas definições na academia para jogos, considerando várias áreas, como antropologia, filosofia, história e game design. Uma delas é a de Avedon e Sutton-Smith (1971, p. 405):

> Jogos são um execício de sistemas de controle voluntário, no qual há uma contestação entre poderes, confinado por regras em ordem de produzir um resultado desequilibrado.

Isso significa que jogos são atividades intencionais, nas quais há objetivos e conflitos, possuindo regras que levam a um desequilíbrio, onde haverá um vencedor e um perdedor.

Devido à relevância dada à separação entre a realidade em que o jogo ocorre, no presente trabalho será utilizada a definição desenvolvida por Adams (2014a, p. 3):

> Jogo é um tipo de atividade, conduzida dentro do contexto de uma realidade alternativa, em qual o(s) participante(s) tenta(m) atingir pelo menos uma meta arbitrária, com objetivo não trivial, seguindo de acordo com as regras.

Em outras palavras, jogos são atividades em que os eventos ocorrem em uma realidade virtual, onde há ações dos jogadores dentro das regras estabelecidas para atingir metas.

2.2 O que são Shooters

Partindo da definição de jogos, também existem algumas definições para shooters games, como a de Fagerholt e Lorentzon (2009, p. 1):

> (...) Um gênero de jogo focado no combate baseado em armas de fogo ou projéteis, no qual a ação é vista através dos olhos de um protagonista.

Dada a abrangência da definição, a que será utilizada no presente trabalho é a de Adams (2014b, p. 8):

> jogos focados na habilidade do jogador de realizar ações a longas distâncias, com ajuda de algum tipo de dispositivo mecânico, para atingir um efeito desejado, que normalmente é, mas nem sempre, um efeito destrutivo.

Em resumo, shooters são jogos, em duas dimensões (2D) ou três dimensões (3D), que utilizam dispositivos mecânicos, como controles, simulacros[1], armas reais etc., e permitem ao jogador realizar ações a longas distâncias, que podem ou não ter efeitos destrutivos. O jogo Battlefield 4[2], mostrado na Figura 1, é um exemplo de shooter em que as ações do jogador possuem efeitos destrutivos, enquanto o jogo Splatoon 3[3], desenvolvido para a plataforma Nintendo Switch, mostrado na Figura 2, é um exemplo de shooter em que os efeitos não são destrutivos.

Figura 1 – Imagem retirada do Shooter Battlefield 4

Fonte: https://www.ea.com/pt-br/games/battlefield/battlefield-4

2.2.1 Evolução dos Shooters

Jogos de tiro ao alvo sempre existiram desde que o ser humano aprendeu a atirar projéteis. Porém, é aceitável considerar a década de 1890 como a do início dos jogos shooters. Naquela década, eles eram jogados nos parques de diversões, e necessitavam de uma pessoa para reposicionar os alvos ao término de cada partida. Posteriormente, tais jogos passaram a utilizar partes eletromecânicas, diminuindo cada vez mais a necessidade de um operador humano para reiniciar a partida. Simulacros que utilizavam luz começaram a substituir as armas reais; o jogo Ray-o-light (Figura 3) de 1936 já utilizava tubos receptores a vácuo nos alvos para simular o tiro de arma real, em que as armas lançavam feixes de luz, e ao ser percebido pelos receptores, os alvos se comportavam como se tivessem sido atingidos. Tais mudanças na mecanização dos

[1] Equipamentos que imitam características, como forma, textura, peso ou sons, de armas reais
[2] https://www.ea.com/pt-br/games/battlefield/battlefield-4
[3] https://www.nintendo.com/pt-br/store/products/splatoon-3-switch/

Figura 2 – Imagem obtida do Shooter Splatoon 3

Fonte: https://splatoon.nintendo.com/en/gameplay/

shooters ocorreram até possibilitar partidas entre um humano e a máquina; o jogo Jungle Charlie (Figura 4), de 1971, foi um dos pioneiros a implementar essa interação, colocando o jogador para competir contra Jungle Charlie em quem acertava mais alvos, sendo o personagem Jungle Charlie controlado pela máquina (Wolf, 2012).

Figura 3 – Fotos do Ray-o-light, mostrando a máquina e o simulacro de rifle (imagem à esquerda), o display (imagem superior direita) e o alvo utilizado no arcade (imagem inferior direita)

Fonte: https://videogamehistorian.wordpress.com/tag/rayolite-rifle-range/

Figura 4 – Foto do arcade Jungle Charlie

Fonte: https://www.pinrepair.com/arcade/wildkin.htm

Com a evolução da tecnologia, os dispositivos se tornaram mais acessíveis. Em 1972, o Magnavox Odyssey (Figura 5) se tornou o primeiro console doméstico a possuir um receptor a vácuo embutido. Sempre que o jogador apertava o botão para disparar, o receptor se abria para identificar se a arma estava apontada para uma área da tela que emitia feixes de luz, contabilizando pontos para o jogador. Também controles alternativos começaram a ser utilizados em shooters, como o Wii Remote[4], Kinect[5] e webcams, diversificando as possíveis interações com os jogos.

Assim, nota-se que, ao longo da história, o gênero shooter começou a se dividir em diversos subgêneros, aproveitando-se da evolução tecnológica e do desenvolvimento de novos controles. Adams (2014b) descreveu alguns deles:

– **Shooting gallery:** Um dos subgêneros mais simples, que emula os shooters tradicionais

[4] https://www.nintendo.com/en-gb/Wii/Accessories/Accessories-Wii-Nintendo-UK-626430.html
[5] https://web.archive.org/web/20130112203900/http://www.xbox.com/en-US/kinect?xr=shellnav

Figura 5 – Foto do console Magnavox Odssey

Fonte: https://www.pinrepair.com/arcade/wildkin.htm

das antigas feiras da década de 1890. O objetivo principal do jogador é atirar em objetos a partir de uma distância fixa.

- **Rail shooters:** Nestes jogos, o avatar do jogador percorre caminhos pré-definidos, como se fossem trilhos, que podem possuir algumas ramificações. Este subgênero dá poucas opções de navegação ao jogador e pouco permite explorações. Normalmente são jogos solo ou cooperativos com dois jogadores. Um dos títulos mais famosos deste subgênero é o StarFox 64[6].
- **Survival horror:** Esses jogos aproveitam o poder gráfico dos hardwares modernos para reproduzir sangue e cenas grotescas. Nesse subgênero, a exploração do mapa e a resolução de problemas têm um papel mais importante do que o combate. Alguns títulos famosos são a série Silent Hill[7] e Resident Evil[8].
- **Arena games:** Esse subgênero é focado em partidas mata-mata[9] ou competições de times em uma área fechada, oferecendo armas e poderes aos jogadores. Títulos conhecidos desse subgênero são Fortnite[10], Team Fortress[11], Free Fire[12] e PUBG: Battlegrounds[13].

Atualmente, shooters são um dos gêneros que geram mais renda no mercado de games. Em 2023, quatro dos doze jogos que geraram as maiores rendas brutas do ano na Steam[14]

[6] https://www.nintendo.com/pt-pt/Jogos/Nintendo-64/Star-Fox-64-1093854.html
[7] https://www.konami.com/games/silenthill/us/en/
[8] https://game.capcom.com/residentevil/en/
[9] Partidas onde todos os jogadores se enfrentam, e o vencedor é aquele que permanecer vivo por último
[10] https://www.fortnite.com/?lang=pt-BR
[11] https://www.teamfortress.com/
[12] https://ff.garena.com/pt/
[13] https://pubg.com/pt-br/main
[14] https://store.steampowered.com/

eram shooters. Cinco dos onze jogos que obtiveram mais de 300 mil jogadores simultâneos na Steam também eram shooters (**??**), sendo eles Counter-Strike 2[15], PUBG: Battlegrounds, Destiny 2[16], Apex Legends[17] e Sons of The Forest[18]. Jogos como Counter-Strike 2 e PUBG: Battlegrounds já atingiram, respectivamente, 600 e 359 semanas consecutivas na tabela dos jogos mais vendidos da semana na Steam (Figura 6).

Figura 6 – Tela capturada da tabela dos 100 jogos mais vendidos da semana na steam até o dia 11/02/2024

Fonte: https://store.steampowered.com/charts/topselling/global

Estes dados, colhidos na plataforma da Steam, mostram o quão lucrativo e bem sucedido é o mercado de Shooters atualmente.

Dado o foco deste trabalho, na próxima seção será desenvolvido mais a fundo o subgênero Tactical Shooter.

2.3 Tactical Shooter

Tactical Shooter é um subgênero Shooter, que possui a principal característica de simular combates reais de forças militares, com representações realistas de armas e situações;

[15] https://www.counter-strike.net/cs2
[16] https://www.bungie.net/7/pt-br/Destiny/NewLight
[17] https://www.ea.com/pt-br/games/apex-legends
[18] https://store.steampowered.com/app/1326470/Sons_Of_The_Forest

dessa forma, as partidas se distanciam de jogabilidades irreais, como a possibilidade de segurar e disparar com diferentes armas em cada mão ao mesmo tempo, por exemplo.

A seguinte declaração foi dada pelos desenvolvedores do Tactical Shooter America's Army (Figura 7), jogo desenvolvido pelas Forças Armadas dos Estados Unidos, com uma versão que servia como propaganda do exército e uma versão utilizada no treinamento de seus oficiais (Game, 2004):

> Nós estamos bem preparados para o futuro da defesa modelando e simulando. E esse futuro tem cara de jogo. (Game, 2004, p. 21)

De acordo com eles, um jogo sério (simulador para treinamento militar), apesar de focar no aprendizado, antes de tudo, ainda é um jogo. Dessa forma, com algumas modificações, um jogo pode se tornar um jogo sério e vice-versa.

Tactical Shooters têm a característica de simular combates com equipamentos reais, proporcionando situações mais realistas; por esta razão, esses títulos também são utilizados em treinamentos militares (Game, 2004). Alguns exemplos famosos de Tactical Shooters são as franquias Battlefield, Call of Duty[19] e Tom Clancy's Ghost Recon[20].

2.4 Realidade virtual

O início da realidade virtual (RV) remonta à década de 1830, quando Sir Charles Wheatstone formalizou a estereoscopia (capacidade de ver imagens com profundidade) e inventou o estereoscópio, um dispositivo que permitia visualizar duas imagens fotografadas de pontos de vista diferentes, criando uma imagem 3D quando vistas por cada um dos olhos (Chavez; Davier, 2011). Mais tarde, em 1908, Lippman inventou a fotografia autostereoscópica, que proporciona uma imagem 3D sem a necessidade de utilizar óculos. Posteriormente, essas tecnologias foram desenvolvidas e aplicadas em cinema, fotografia, jogos, simulações realistas para treinamento em serviços perigosos, entre outros (Lafruit; Teratani, 2022).

O desenvolvimento de HMDs não é recente, mas, com a redução dos custos de produção, diversos modelos de HMDs começaram a ser criados tanto para uso diário quanto para jogos, como os HTC VIVE[21], Valve Index[22], Oculus Quest[23], e o mais recente Vision Pro[24].

[19] https://www.callofduty.com/br/pt
[20] https://www.ubisoft.com/pt-br/game/ghost-recon/breakpoint
[21] https://www.vive.com/
[22] https://www.valvesoftware.com/
[23] https://www.oculus.com/quest/refurbished/
[24] https://www.apple.com/apple-vision-pro/

Figura 7 – Imagem retirada do Tactical Shooter America's Army

Fonte: https://arstechnica.com/gaming/2019/01/army-video-games/

Esses capacetes geram imagens estereoscópicas do mundo virtual e possuem equipamentos que conseguem identificar os movimentos do capacete ou do usuário, além de controles que permitem interagir com o mundo virtual em experiências de RV. Atualmente, os preços variam desde os mais acessíveis, como o Oculus Quest 2 (Figura 8), que custa cerca de 300 dólares (**??**), até os mais caros, como o Vision Pro (Figura 9), que custa cerca de 3500 dólares (**??**).

Figura 8 – Imagem do Oculus Quest 2

Fonte: https://www.meta.com/quest/products/quest-2/

Figura 9 – Pessoa utilizando o Vision Pro

Fonte: https://www.apple.com/apple-vision-pro/

2.4.1 Tactical Shooters e RV

Tactical Shooters e RV possuem um forte vínculo ao longo da história com o desenvolvimento de jogos sérios para treinamento militar, como é o caso do Virtra Systems[25], sistemas de RV para treinamento policial desenvolvidos desde 1993, e o EasyAim Simulator[26], um simulador de RV voltado para o treinamento policial, lançado em 2020.

Dado os benefícios da implantação da RV em treinamentos militares, como diminuição de custos, redução de riscos, adaptação geográfica e uma capacidade de criação de ambientes reais e imersivos (Bc, 2024), justifica este vínculo militar dos Tactical Shooters.

Alguns exemplos de Tactical Shooters desenvolvidos para a RV serão descritos a seguir.

2.4.1.1 Sniper Elite VR

Sniper Elite é uma série popular desenvolvida pela Rebellion Developments desde 2005. A série é conhecida por seu foco na furtividade[27], ideal para missões de reconhecimento ou eliminação por atiradores de elite.

[25] https://www.virtra.com/
[26] http://easyaimsimulator.com/
[27] Capacidade de agir sem ser notado

Figura 10 – Imagem retirada do jogo Sniper Elite VR

Fonte: https://store.steampowered.com/app/752480/Sniper_Elite_VR/

A série possui dez títulos, incluindo um voltado para a realidade virtual (RV), o Sniper Elite VR. Este título está disponível para os sistemas Microsoft Windows[28], PlayStation VR[29] e Oculus Quest. Ele recebeu uma recepção média da crítica, com notas como 68/100[30] no Metacritic para a versão de PC e 6/10[31] na IGN. Na Steam, obteve 56% de análises positivas entre as 632 avaliações registradas até o momento em que este trabalho foi escrito.

Sniper Elite VR é ambientado na Segunda Guerra Mundial, narrando as memórias do protagonista. No jogo, o jogador pode usar rifles, granadas, metralhadoras e pistolas, precisando recarregar e trocar a munição manualmente. A necessidade de usar as mãos para mirar é o principal diferencial deste título em comparação aos outros jogos da série.

2.4.1.2 War Dust VR

War Dust VR é um jogo desenvolvido pela Raptor Lab, disponível para o sistema Microsoft Windows. War Dust é um jogo online de batalhas táticas entre dois times de 32 jogadores, cujo objetivo é conquistar pontos de controle e derrotar inimigos. Assim como em Sniper Elite VR, a recarga das armas e a mira são realizadas manualmente. Além disso, o jogador pode dirigir carros e lanchas, e pilotar helicópteros e jatos, tornando o jogo mais dinâmico.

War Dust possui 80% de análises positivas pela comunidade, das 1.469, na Steam,

[28] https://www.microsoft.com/pt-br/windows/
[29] https://www.playstation.com/pt-br/ps-vr/
[30] https://www.metacritic.com/game/sniper-elite-vr/
[31] https://br.ign.com/sniper-elite-vr

Figura 11 – Imagem retirada do jogo War Dust VR

Fonte: https://store.steampowered.com/app/752480/Sniper_Elite_VR/

até o presente momento em que este trabalho estava sendo escrito.

2.4.1.3 Superhot VR

Superhot é uma franquia de jogos desenvolvida pela Superhot Team, que começou na 7 Day First Person Shooter (FPS) Game Jam de 2013. No início de 2016, o jogo foi lançado, e em dezembro do mesmo ano foi lançada a versão em RV, o Superhot VR.

Figura 12 – Imagem retirada do jogo Superhot VR

Fonte: https://store.steampowered.com/app/617830/SUPERHOT_VR/

Superhot se destacou pela jogabilidade única, onde o tempo só avança quando o

jogador se move ou dispara, e a munição é limitada, sendo possível adquirir novas armas e munição apenas após eliminar inimigos. O objetivo do jogo é matar todos os inimigos em cada fase sem ser atingido por eles.

Superhot VR está disponível para as plataformas Microsoft Windows, PlayStation VR e Oculus Quest. A recepção da crítica foi positiva, com o jogo recebendo notas como 83/100 no Metacritic pela versão de PC e 9/10 pela IGN. Até o momento em que este trabalho foi escrito, Superhot VR possui 82% de análises positivas entre as 6916 análises realizadas na Steam.

2.5 Importância das arquiteturas de softwares

Dada a complexidade de elaboração de softwares, Shaw e Garlan (1996) desenvolveram, em 1996, o termo Arquitetura de Software, buscando flexibilidade, escalabilidade e facilidade de manutenção no desenvolvimento de sistemas. De acordo com a ISO/IEC/IEEE 42010:2022, Arquitetura de Software é a estrutura fundamental de um sistema de software, definindo seus componentes, relações, e os princípios e evolução do projeto (ISO/IEC/IEEE, 2022).

As arquiteturas de software desempenham um papel importante no desenvolvimento de jogos, auxiliando na gestão da complexidade e na obtenção de qualidade em performance, disponibilidade, segurança e adaptabilidade, conforme demonstrado por Wang e Nordmark (2015).

O trabalho de Wang e Nordmark (2015) foi embasado no método Goal, Question, Metrics (GQM) (Caldiera; Rombach, 1994), que separa a mensuração dos resultados em três fases:

- Goal: Definição do objetivo e o que se deseja mensurar;
- Question: Definição das perguntas que serão realizadas; e
- Metrics: Determinação de como as respostas às questões serão mensuradas.

Os autores definiram o objetivo do trabalho como sendo:

> Examinar como arquitetura de software é utilizada (...) a partir do ponto de vista de um desenvolvedor de jogos no contexto de desenvolvimento de jogos. (Wang; Nordmark, 2015, p 275)

Para as perguntas, foi elaborado um questionário com 20 afirmações que respondiam a 4 grupos de perguntas. Pelo foco desta dissertação, destaca-se apenas um desses grupos. Para as métricas, os participantes deveriam responder o grau de concordância com cada afirmação

utilizando a escala de Likert (Likert, 1932), com espaço para expressar opiniões sobre cada pergunta. A única pergunta do grupo destacado é:

– RQ1: Qual o papel que a arquitetura de software desempenha no desenvolvimento de software?

Para responder ao questionário, foram recrutados desenvolvedores de jogos no Estande Nórdico na Game Developer Conference[32] (GDC 2012) e por envio de e-mails diretamente a desenvolvedores de jogos.

A conclusão desse trabalho foi que, com base na RQ1, é possível afirmar que a arquitetura de software é crucial no desenvolvimento de jogos. Ela é fundamental para o gerenciamento da complexidade do software, atingindo qualidade em performance, disponibilidade, segurança e modificabilidade. Além disso, o conceito do jogo influencia diretamente na arquitetura de software, principalmente porque isso definirá a escolha do motor de jogos[33].

Portanto, a utilização de arquiteturas de software desempenha um papel importante no desenvolvimento e na manutenção de jogos, evitando que a dificuldade de desenvolvimento escale com o aumento do escopo.

2.6 Características principais de simuladores para treinamento militar

Por conta que Tactical Shooters também são utilizados em treinamentos militares, é razoável que eles também possuam características necessárias para estes treinamentos. Para isso Leite-Júnior *et al.* (2012) propuseram um modelo para simuladores de treinamento militares utilizando como base o sistema Virtra 300 LE. As características elencada na pesquisa foram:

– Simulação baseada em sequências de vídeo interativas ou ambientes 3D;
– Simulação de situações específicas para forçar o treinando a tomar decisões rápidas e adequadas, incluindo a execução de tiros ou a tomada de decisões alternativas, respeitando a doutrina previamente estabelecida;
– Soluções de comunicação, gerenciamento e controle baseadas em sistema modular (computador, tela, alto-falante e sistema de captura de tiro);
– Utilização de sistemas específicos para medir o desempenho do treinando, considerando variáveis como tempo de resposta a eventos e posições de tiro;

[32] https://gdconf.com/
[33] Ferramenta que ajuda no desenvolvimento de jogos, simplificando o desenvolvimento de cada parte do jogo (áudio, vídeo, tratamento dos comandos de entrada e saída etc.)

– Simulação considerando a influência real da gravidade para calcular trajetórias de balas disparadas.

3 TRABALHOS RELACIONADOS

Neste capítulo, são apresentados os trabalhos relacionados ao tema proposto nesta dissertação. Para isso, foi realizada uma pesquisa de estudos que envolvem RV, jogos e arquitetura de software.

Para a seleção dos artigos a seguir, foi utilizada a plataforma ResearchGate[1], empregando as palavras-chave "Shooter, Tactical Shooter, Virtual Reality, Framework, Software Architecture". As pesquisas foram realizadas em fevereiro de 2024. A escolha da plataforma ResearchGate se deu pela ampla disponibilidade de artigos em seu banco de dados, que inclui publicações de diversas editoras, além da possibilidade de contato direto com os autores para solicitar acesso aos artigos sem custos adicionais. As palavras-chave foram selecionadas por estarem alinhadas ao tema desta dissertação.

O processo de seleção dos artigos foi conduzido utilizando o método Quick and Dirty Review (Yi, 2014). Inicialmente, os artigos foram selecionados com base nos títulos e resumos (abstracts). Em seguida, uma leitura mais detalhada foi realizada para identificar e selecionar apenas os artigos diretamente relacionados ao tema proposto.

Durante a pesquisa, não foram encontrados trabalhos específicos que abordassem Tactical Shooters, possivelmente devido à escassez de conteúdos e ferramentas abertas para o desenvolvimento de RV (Abeywardena, 2023).

Na primeira fase de revisão, foram selecionados quatro artigos (mostrados na Tabela 1), dos quais apenas três foram mantidos na segunda fase da revisão, pois o artigo não escolhido não se foca em Tactical Shooters para RV.

3.0.1 OXREF: Open XR for Education Framework

O trabalho de Abeywardena (2023) desenvolve o Open XR for Education Framework (OXREF), um framework processual para o desenvolvimento de software educativo em geral em Realidade Estendida (RE). A OXREF é dividida em três camadas, conforme mostrado na Figura 13:

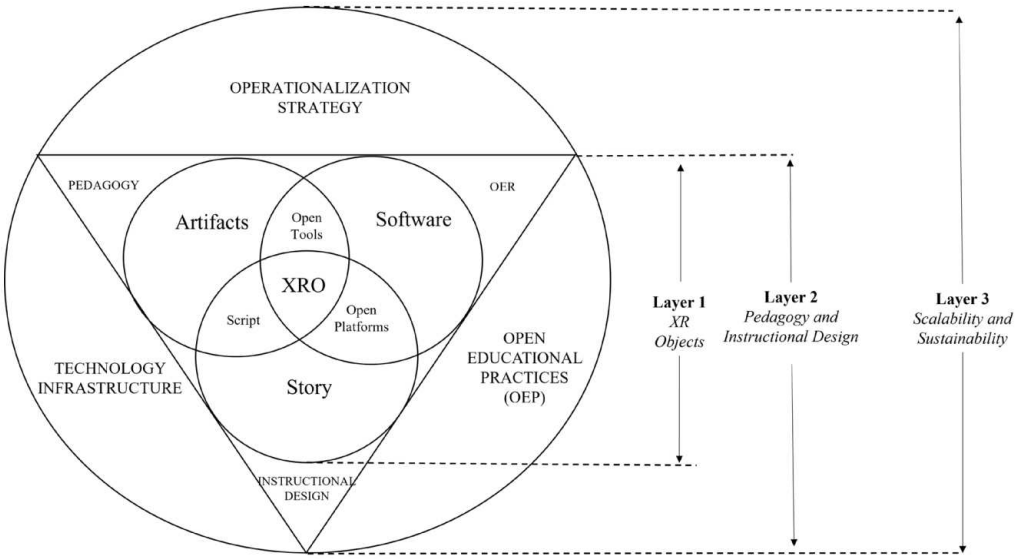
– A camada 1 trata da criação dos objetos necessários para o desenvolvimento de software em RE, incluindo histórias, artefatos e ferramentas de desenvolvimento e design a serem utilizados;

[1] https://www.researchgate.net/

Tabela 1 – Relação dos artigos selecionados na revisão realizada.

Artigo	Ano	Autores	Revista	Selecionado
OXREF: Open XR for Education Framework	2023	I. S. Abeywardena	The International Review of Research in Open and Distributed Learning	Sim
ZeusAR: a process and an architecture to automate the development of augmented reality serious games	2022	H. Marín-Vega, G. Alor-Hernández, L. O. Colombo-Mendoza, M. Bustos-López and R. Zataraín-Cabada	Multimedia Tools and Applications	Sim
Universal XR Framework Architecture Based on Open-Source XR Tools	2024	Y. Bondarenko, V. Kuts, S. Pizzagalli, K. Nutonen, N. Murray, E. O'Connell	Springer Proceedings in Business and Economics	Sim
A Generic Architecture and Validation Considerations for Tactical Combat Casualty Care Serious Games	2014	H. Feron, A. Lehmann, F. Josse	The Journal of Defense Modeling and Simulation Applications Methodology Technology	Não

Figura 13 – Diagrama de representação da OXREF, que separa o desenvolvimento em 3 camadas: objetos RE, Design pedagógico e instrucional e Escalabilidade e sustentabilidade.



Fonte: (Abeywardena, 2023)

– A camada 2 provê o design pedagógico e instrucional para o desenvolvimento, definindo as competências e conhecimentos a serem abordados pelo software, como pensamento

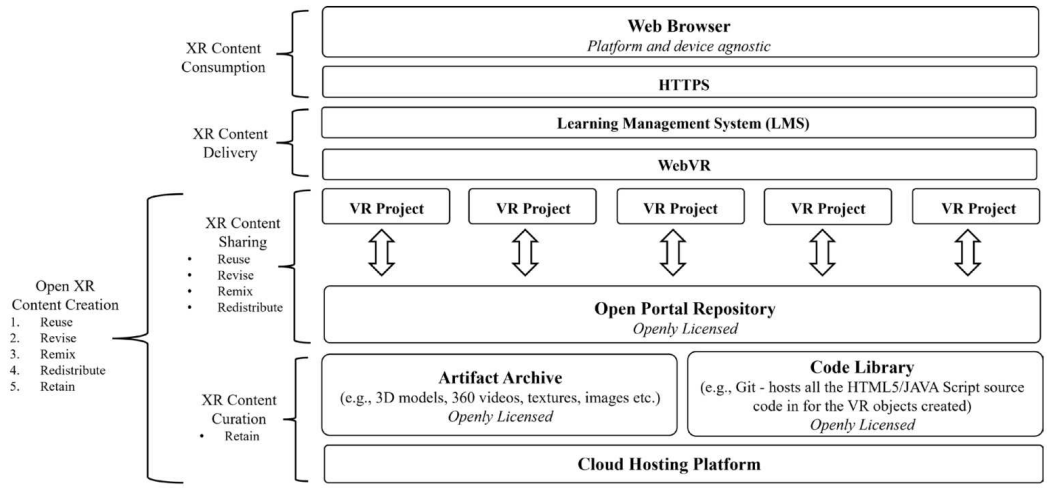
crítico e conhecimento técnico; e

– A camada 3 lida com a escalabilidade e sustentabilidade do projeto, determinando a tecnologia, infraestrutura e estratégias operacionais; em suma, esta camada trata da arquitetura do programa.

De acordo com Renz e Hilbig (2020), a escalabilidade e a sustentabilidade da aplicação são os maiores obstáculos para o uso de jogos sérios em RE na educação, pois muitos projetos pilotos financiados por governos e organizações não vão além da fase de protótipos (Kluge *et al.*, 2022) devido à falta de soluções para manutenção e compartilhamento de tecnologias entre os produtos de software educacionais.

Para enfrentar esses desafios, Abeywardena (2023) desenvolveu uma arquitetura de infraestrutura tecnológica, apresentada na Figura 14, para ser utilizada no desenvolvimento de projetos educacionais em tecnologia.

Figura 14 – Infraestrutura desenvolvida para a OXREF, possibilitando reusabilidade, revisão, edição e redistribuição de projetos.



Fonte: (Abeywardena, 2023)

Esta infraestrutura permite que o projeto seja reutilizado, revisado, editado e redistribuído, facilitando a colaboração entre diferentes instituições governamentais e privadas devido ao uso de ferramentas e arquiteturas de código aberto. Ela é dividida em:

– XR Content Curation, responsável pela plataforma de hospedagem, pelos arquivos de artefatos (modelos 2D, 3D, vídeos 360° etc.) e pela biblioteca de código a ser utilizada;
– XR Content Sharing, um portal de repositório onde é possível buscar por artefatos e códigos de projetos de RV;

– XR Content Delivery e XR Content Consumption, que tratam do consumo dos projetos, utilizando a especificação WebVR[2], permitindo que o projeto seja consumido por navegadores web independentemente da plataforma ou dispositivo.

Este trabalho se limita à definição do framework e da arquitetura, deixando a implementação da OXREF como trabalho futuro.

3.0.2 ZeusAR: a process and an architecture to automate the development of augmented reality serious games

O trabalho de Marin-Vega *et al.* (2022) desenvolve um processo, uma arquitetura para a adaptação de jogos sérios já existentes para RA e uma ferramenta, denominado ZeusAR, a partir da arquitetura desenvolvida. O processo é dividido em três fases:

– Fase de análise cuida da leitura dos dados do jogo existente, e da validação e correção da estrutura de seus arquivos;
– Fase de configuração lida com a customização e a incorporação de características de RA ao jogo sério, com a seleção do conteúdo a ser utilizado, com a seleção de bibliotecas que gerarão funcionalidades de RA, e com a obtenção e a aceitação das ferramentas de configuração do projeto; e
– Fase de geração gera a nova estrutura do jogo sério em RA e o arquivo para download.

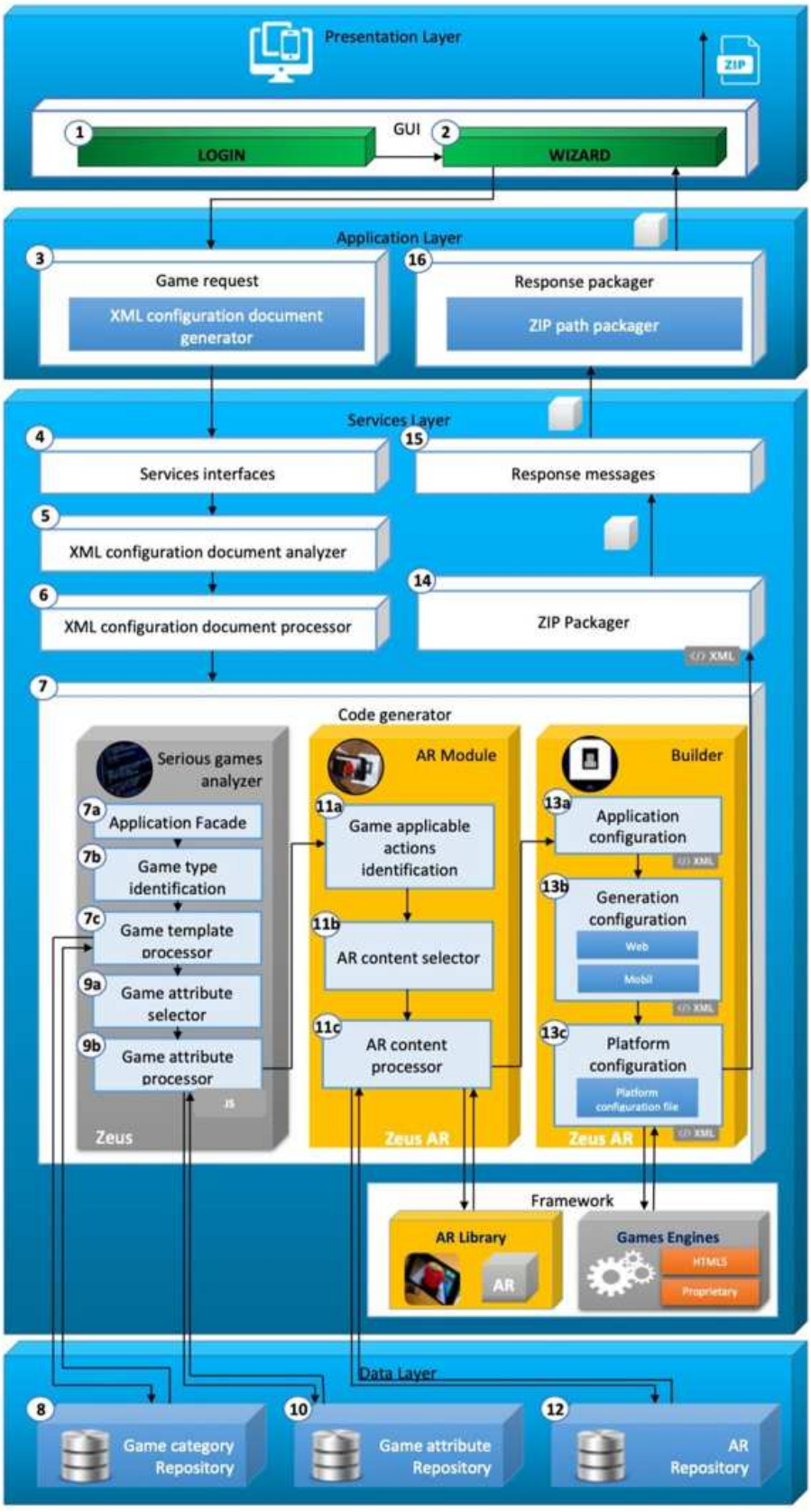
A partir desse processo, foi desenvolvida a arquitetura apresentada na Figura 15. A fim de possibilitar uma melhor organização de seus componentes e de suas respectivas interações, essa arquitetura é divida em quatro camadas:

– Camada de apresentação lida com a interface de usuário;
– Camada de aplicação serve de interface para acesso das informações da camada de serviços;
– Camada de serviços provê uma série de módulos para a geração do jogo sério, como o gerador de código; e
– Camada de dados cuida do armazenamento dos dados necessários para o desenvolvimento do jogo.

Para avaliar a ferramenta ZeusAR os autores utilizaram o método System Usability Scale (SUS) (Brooke *et al.*, 1996), similar à escala Likert, com dez professores e facilitadores de trigonometria básica. Os usuários foram divididos em dois grupos, assistidos por um líder de avaliação e um técnico da ZeusAR. Após isso, os participantes desenvolveram um jogo da

[2] API para desenvolvimento de RV e Realidade Aumentada (RA) em navegadores

Figura 15 – Arquitetura apresentada por Marin-Vega *et al.* (2022).



Fonte: (Marin-Vega *et al.*, 2022)

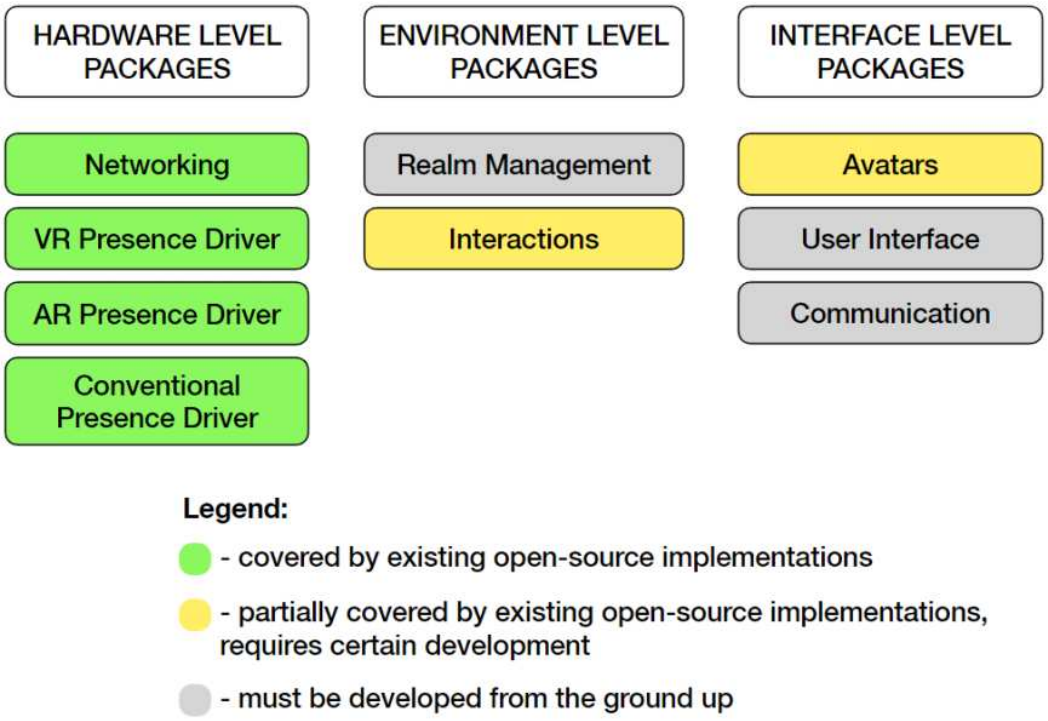
memória, um Sudoku e um jogo de cobras e escadas utilizando a ferramenta. Ao final desse processo, os participantes responderam ao questionário SUS.

Os resultados da pesquisa mostraram que a maioria dos participantes se sentiu confortável ao utilizar a ZeusAR, e que a ferramenta é de fato rápida e fácil de usar, havendo também um vídeo na plataforma Youtube mostrando o passo a passo de sua utilização[3].

3.0.3 Universal XR Framework Architecture Based on Open-Source XR Tools

O trabalho de Bondarenko *et al.* (2024) desenvolve uma arquitetura projetada para fornecer acessibilidade, modularidade e independência de hardware no desenvolvimento de aplicações de Realidade Estendida RE.

Figura 16 – Arquitetura apresentada por Bondarenko *et al.* (2024).



Fonte: (Bondarenko *et al.*, 2024)

A arquitetura apresentada na Figura 16 é divida em sete módulos, que são descritos a seguir:

- Networking – módulo responsável pela sincronização dos objetos e dos eventos que ocorrem no ambiente virtual dos usuários participantes;
- Drivers de presença para RV, RA e convencional – módulo que fornece as interfaces independentes de hardware;

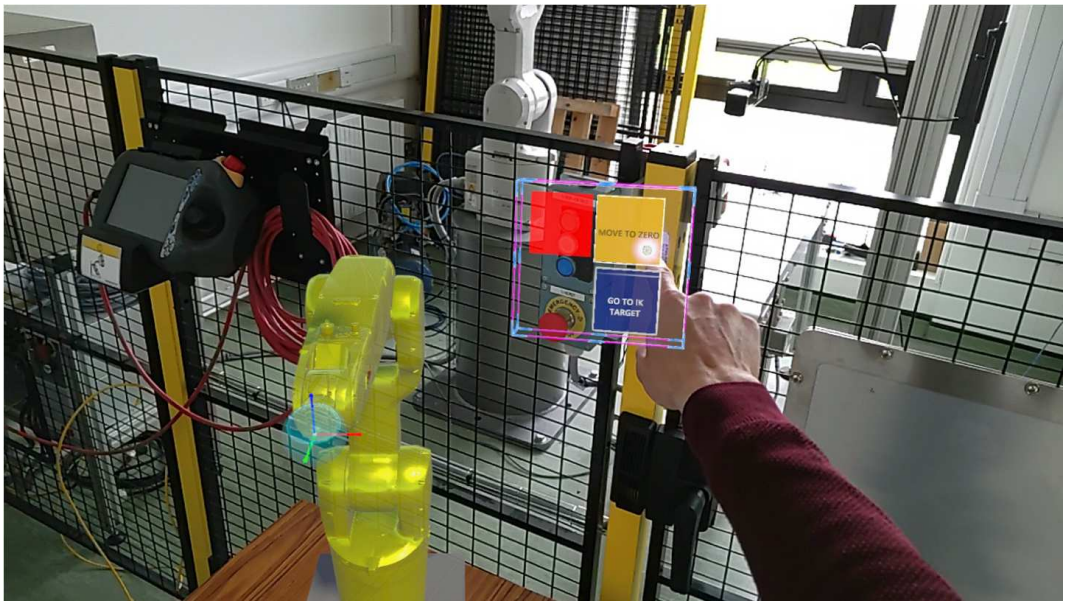
[3] https://www.youtube.com/watch?v=pCwRO-DD-L4

– Gerenciamento de ambiente – módulo que fornece uma interface para criação e edição de ambientes virtuais;
– Interações – módulo que gerencia as interações físicas do avatar do usuário com o ambiente e os objetos presentes nele;
– Avatares – módulo que oferece meios para importar, otimizar e editar o corpo virtual do usuário;
– Interface de usuário – módulo que provê a interface do usuário; e
– Comunicação – módulo que implementa ferramentas de chat de voz, streaming de vídeo e compartilhamento de tela dentro do ambiente virtual.

Com base nessa arquitetura, foi proposto um framework para a engine Unity, distribuído através do Unity Package Manager. Este framework consiste em um conjunto de pacotes de software abertos já existentes, como o Mirror[4], Mirage[5], DarkRift[6] e o framework Unity XR Plug-in[7].

Os autores apresentam dois possíveis casos de uso desenvolvidos utilizando o framework proposto: um software de interação humano-máquina utilizando RA (Figura 17) e um laboratório de engenharia industrial utilizando RV (Figura 18).

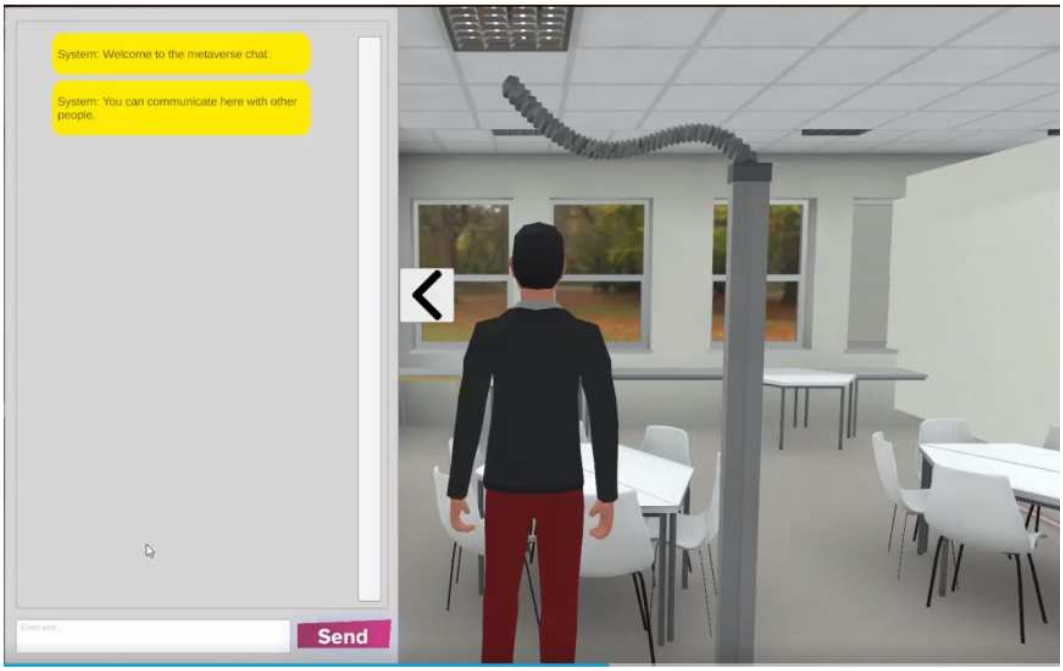
Figura 17 – Exemplo de interface em um cenário de interação humano-máquina.



Fonte: (Bondarenko *et al.*, 2024)

[4] https://github.com/MirrorNetworking/Mirror
[5] https://github.com/MirageNet/Mirage
[6] https://github.com/DarkRiftNetworking/DarkRift
[7] https://docs.unity3d.com/Manual/XRPluginArchitecture.html

Figura 18 – Laboratório virtual de engenharia industrial.



Fonte: (Bondarenko *et al.*, 2024)

O estudo de caso de interação humano-máquina apresenta uma interface onde o usuário pode interagir com a representação de um robô industrial através de RA, comparando a experiência do usuário ao manipular um robô real ou virtual. Já o caso de uso do laboratório de engenharia industrial é um jogo sério, multiplayer, onde educadores e alunos, remotamente, estão em uma sala de aula de laboratório de engenharia industrial, utilizando chat de texto e voz para se comunicar, permitindo interações com equipamentos virtuais e minijogos, como o de pneumáticas.

Como resultado, os autores propuseram uma arquitetura e um framework para o desenvolvimento de jogos sérios utilizando RE, que poderá ser utilizado por pesquisadores e instituições educacionais, devido ao uso de pacotes livres. Como trabalhos futuros, os autores pretendem finalizar o desenvolvimento do framework e validá-lo através de um estudo de caso acadêmico.

3.1 Presente trabalho

Como mostrado por Renz e Hilbig (2020), há escassez de materiais acadêmicos desenvolvendo frameworks e arquiteturas para auxiliar no desenvolvimento de jogos para RV, resultando em poucos artigos sobre este tema.

O trabalho proposto por Abeywardena (2023) foca em estabelecer um framework

processual auxiliando todas as etapas do desenvolvimento de jogos sérios para RE, não apenas a programação. No entanto, devido à abrangência da proposta, ele não cobre exatamente a programação de Tactical Shooters em RV, necessitando utilizar outro framework adicional. Além disso, o framework carece de validação, que será realizada como trabalho futuro.

A arquitetura desenvolvida por Marin-Vega *et al.* (2022) foca no desenvolvimento de jogos sérios para RA, validada utilizando o método SUS com professores e facilitadores de trigonometria básica. Entretanto, essa arquitetura não pode ser utilizada para desenvolver Tactical Shooters em RV, pois possui outra finalidade.

O framework proposto por Bondarenko *et al.* (2024) visa auxiliar o desenvolvimento de jogos sérios em RE, utilizando pacotes abertos, permitindo acessibilidade e modularidade aos jogos desenvolvidos. Todavia, o framework ainda está em desenvolvimento e carece de validação, não permitindo saber com certeza se consegue abranger Tactical Shooters em RV.

Dessa forma, este trabalho objetiva contribuir para o desenvolvimento de jogos, apresentando um framework de software para Tactical Shooters em RV, para que, com a diminuição da complexidade no desenvolvimento de jogos desse gênero, o desenvolvedor possa focar nas ferramentas que serão acrescentadas. E também, pela proximidade dos Tactical Shooters aos simuladores de treinamento militar, deseja-se contribuir para a superação de desafios para a implementação de RV a estes treinamentos, como fidelidade da simulação e integração às estruturas de treinamentos (Bc, 2024). A Tabela 2 compara os trabalhos descritos com a presente dissertação.

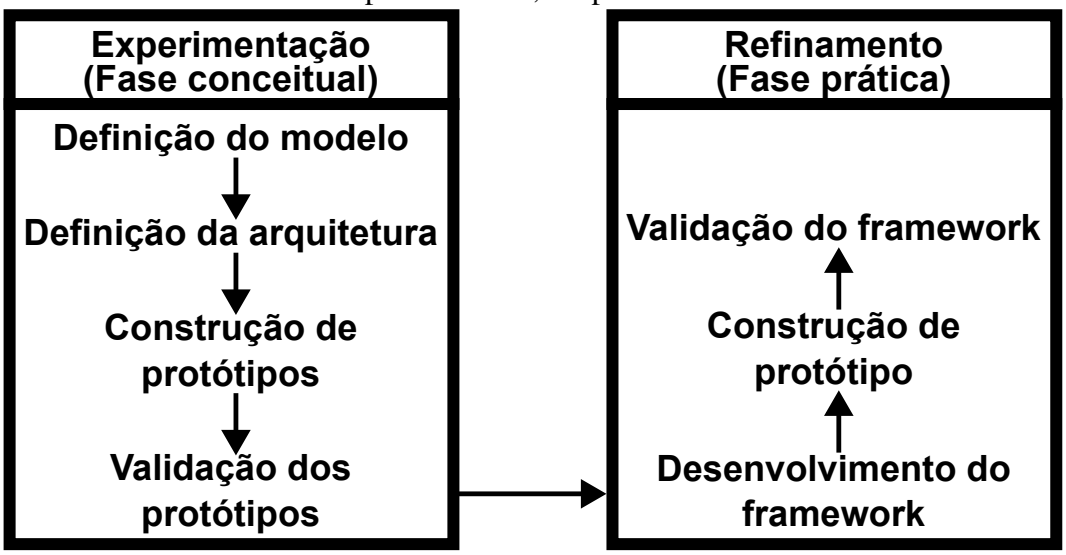
Tabela 2 – Comparação entre as propostas apresentadas na Seção 3 e a presente proposta de framework de software.

Framework/ Arquitetura	Foco	Validações	Tipo de aplicações	Plataforma
Abeywardena (2023)	Criação de jogos sérios escaláveis	Sem validações	Jogos sérios	RE
(Marin-Vega *et al.*, 2022)	Adaptação de jogos sérios	Questionário SUS realizado com professores e faciltadores de trigonometria básica	Jogos sérios	RA
Bondarenko *et al.* (2024)	Criação de jogos sérios	Framework em desenvolvimento	Jogos sérios	RE
Proposta atual	Criação de Tactical Shooters	Entrevistas semi-estruturadas para validação dos protótipos e realização de workshop e entrevista de grupo focal com desenvolvedores de jogos para validação do framework de software	Tactical Shooters	RV

4 METODOLOGIA

Para o desenvolvimento do framework, por conta da quantidade de atividades necessárias para desenvolver um do mesmo, este trabalho precisou ser dividido em duas fases: uma fase de experimentação e outra de refinamento, como apresentada na Figura 19.

Figura 19 – Etapas necessárias ao desenvolvimento do framework de software. A primeira fase (à esquerda) é a fase de Experimentação, onde foram realizados as atividades conceituais, que são as definições do modelo (que elenca as principais características do programa) e da arquitetura (que compila as características definidas no modelo nos módulos necessários do programa). Já a segunda fase (à direita) é a fase de Refinamento, onde foi desenvolvido a atividade prática, que a partir dos conceitos desenvolvidos na primeira fase, foi possível desenvolver o framework.



Fonte: Imagem desenvolvida pelo autor.

Na fase de experimentação, foram definidos o modelo e a arquitetura. Após a definição da arquitetura, foi possível desenvolver dois protótipos: um estande de tiros fechado, e um estande de tiro 360º. Por fim, foram realizados testes e entrevistas semi-estruturadas com usuários para validação dos protótipos.

Na fase de refinamento, foi desenvolvido o framework de software a partir dos resultados da fase de experimentação. Com esse framework, foi construído mais um protótipo, o Tiro ao Polvo. Por fim, para validar o framework de software, foi realizado um workshop com desenvolvedores de jogos e uma entrevista de grupo focal com os participantes do workshop.

4.1 Framework, arquitetura e modelo

Na academia, há diversas definições de frameworks, algumas delas são descritas a seguir:

- "Conjunto de classes que incorpora um design abstrato para soluções para uma família de problemas relacionados, e suporta o reuso em um nível maior de granularidade do que classes."(Johnson; Foote, 1988);
- "Design reutilizável para um aplicativo, ou subsistema, representado por um conjunto de classes abstratas e a maneira como elas colaboram."(Mattsson, 1998);
- "Conjunto integrado de artefatos de software (como classes, objetos e componentes) que colaboram para fornecer uma arquitetura reusável para uma família de aplicações relacionadas."(Schmidt *et al.*, 2004).

Por ser uma descrição mais próxima ao trabalho realizado, será utilizada a definição de Schmidt *et al.* (2004). Dessa forma, o framework de software precisa ter como base uma arquitetura definida para uma família de aplicações, sendo o framework uma implementação genérica da arquitetura.

De acordo com a ISO/IEC/IEEE (2022), arquiteturas são "a estrutura fundamental ou o esqueleto de um sistema de software, que define seus componentes, suas relações e seus princípios de projeto e evolução.". A arquitetura precisa seguir requisitos funcionais e não funcionais definidos para a família de aplicações definida, sendo o conjunto desses requisitos o modelo.

Mesmo o framework sendo a implementação genérica da arquitetura, é possível desenvolver aplicações a partir da arquitetura, porém, não garantindo o reuso do código escrito como na utilização do framework.

4.2 Entrevista Semi Estruturada e grupo focal

Na literatura científica, há dois tipos de pesquisas: a pesquisa quantitativa, que busca medir as variáveis e as interferências, quantificando-as a partir de amostras de uma população; e a pesquisa qualitativa, que busca colher informações mais profundas e subjetivas.

De acordo com Dias (2000), os principais métodos utilizados em pesquisas qualitativas são:

- Entrevistas: As entrevistas podem ser divididas em três categorias: estruturadas, não

estruturadas e semi-estruturadas. As entrevistas estruturadas e não estruturadas são dois extremos de um continuum. Nas entrevistas estruturadas, as perguntas são fixas e não podem ser alteradas, enquanto nas entrevistas não estruturadas, não há uma ordem fixa de perguntas, permitindo que o entrevistador adapte a entrevista de acordo com cada entrevistado. A entrevista semi-estruturada se encontra entre esses extremos, com uma ordem definida de perguntas, mas permitindo ao entrevistador adaptar as perguntas conforme necessário (Wildemuth, 2009).

- Técnicas projetivas: São técnicas utilizadas quando o pesquisador considera difícil para o participante descrever ou explicar seus sentimentos ou as razões por trás de suas ações. Para isso, o pesquisador apresenta um estímulo ambíguo e observa as reações dos participantes.
- Grupo focal: São entrevistas realizadas com um pequeno grupo de pessoas, coletando informações para avaliar conceitos ou identificar problemas.

A entrevista semi-estruturada pode ser definida como uma entrevista com perguntas pré-definidas, mas cuja ordem pode ser modificada com base na percepção do entrevistador sobre o que é mais apropriado a cada momento da conversa. A formulação das perguntas pode ser ajustada, explicações podem ser fornecidas, perguntas específicas que parecerem inadequadas para um entrevistado em particular podem ser omitidas, e perguntas adicionais podem ser incluídas conforme necessário durante a conversa (Robson, 2002).

4.3 Padrões de projetos

Padrões de projetos são soluções já testadas para problemas comuns no desenvolvimento (Sommerville, 2011), eles descrevem melhores práticas de modo que possam ser reutilizados. Os padrões mais conhecidos foram definidos por Gamma *et al.* (1995), dentre os quais, três foram utilizados no framework de software desenvolvido:

- **Singleton**: Padrão de projeto criacional que garante que a classe possua apenas uma instância, sendo ela acessível globalmente. Apesar de dificultar a modularidade do código, este padrão foi escolhido por sua simplicidade de implementação e uso para criação de interfaces acessíveis em qualquer parte do código;
- **Observer**: Padrão de projeto comportamental que cria um mecanismo em que instâncias de classes possam se inscrever e receber atualizações de eventos ocorridos na instância publicadora. Este padrão pede que haja as classes publicadoras, que irão informar sobre seus eventos, e as classes assinantes, que irão se inscrever na publicadora para receber

suas atualizações. Este padrão trás modularidade e escalabilidade de código ao framework de software;

– **Template Method**: Padrão de projeto comportamental que define a estrutura do algoritmo em uma superclasse e permite que suas subclasses sobrescrevam partes específicas do algoritmo. Este padrão foi escolhido pela vantagem de proporcionar reutilização do código da superclasse por subclasses, agilizando a escrita dos códigos pelo desenvolvedor.

5 DESENVOLVIMENTO

Neste capítulo será descrito o passo a passo do desenvolvimento do framework de software proposto.

5.1 Fase de experimentação

Na fase inicial para a definição da arquitetura, foram utilizadas características necessárias em Tactical Shooters, discutidas no Capítulo 2. Para tal, foi preciso realizar três etapas: a de definição do modelo, a de definição da arquitetura e a de construção de protótipos, que são descritas a seguir.

5.1.1 Definição conceitual do modelo

Em seu livro Fundamentals of Shooter Game Design, Adams (2014b) elencou os seguintes elementos essenciais que jogos de tiro precisam ter:

- **E1**: A natureza da interação, que define o objetivo do jogador para vencer o jogo.
- **E2**: A natureza dos alvos, definindo os tipos, seus comportamentos e ações ao serem acertados (emissão de som, animação de queda, por exemplo) ou não (fugir do jogador, disparar contra o jogador, por exemplo).
- **E3**: A natureza da representação do jogador (avatar), seus comportamentos e possibilidades de interação com os elementos do jogo (ambiente, armas, alvos, etc.).

Como discutido na Seção 2.3, Tactical Shooters se aproximam muito de simuladores para treinamento militar. Dessa maneira, é possível relacionar elementos desses simuladores com os Tactical Shooters.

Como discutido na seção 2.6, Leite-Júnior *et al.* (2012) elencaram os principais elementos que simuladores militares precisam ter:

- **E4**: Simulação baseada em sequências de vídeo interativas ou ambientes 3D;
- **E5**: Simulação de situações específicas para forçar o treinando a tomar decisões rápidas e adequadas, incluindo a execução de tiros ou a tomada de decisões alternativas, respeitando a doutrina previamente estabelecida;
- **E6**: Soluções de comunicação, gerenciamento e controle baseadas em sistema modular (computador, tela, alto-falante e sistema de captura de tiro);
- **E7**: Utilização de sistemas específicos para medir o desempenho do treinando, conside-

rando variáveis como tempo de resposta a eventos e posições de tiro;

– **E8**: Simulação considerando a influência real da gravidade para calcular trajetórias de balas disparadas.

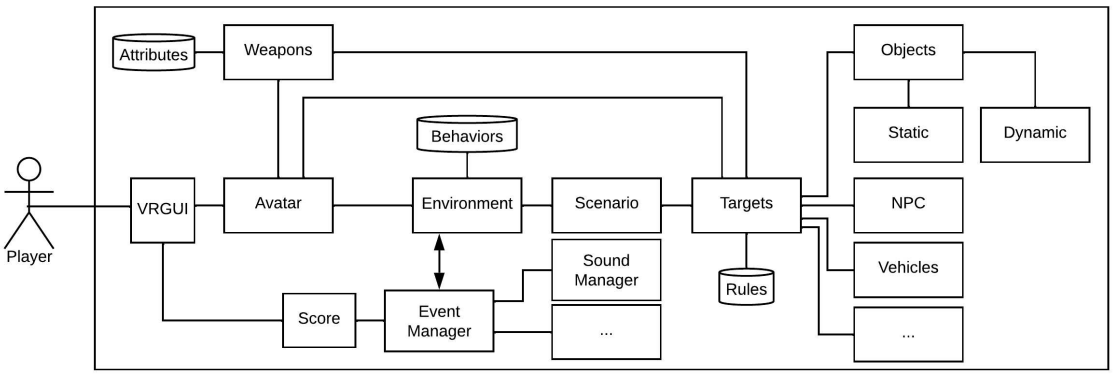
A partir dos pontos elencados, foi definido o seguinte modelo:

– Ambientes 3D bem definidos com respostas diretas às decisões do jogador, abrangendo o *E4* e *E5*.
– Comportamentos de avatar bem estabelecidos envolvendo todas as possibilidades de interação com o ambiente, compreendendo o *E3*.
– Comportamentos bem definidos dos alvos presentes no ambiente, representando *E2*, *E5*.
– Boa gestão das respostas visuais e sonoras necessárias para representar as diversas ocorrências no jogo, incorporando *E3*, *E6* e *E8*.
– Clara definição dos objetivos, com formas de medir o progresso, como cálculo de pontuação e desempenho geral do jogador, como definido nos *E1* e *E7*.

5.1.2 Definição da arquitetura

Com o modelo definido, foi possível desenvolver a arquitetura para Tactical Shooters, abrangendo todos os pontos definidos, separando em módulos de forma que consigam contemplar o modelo.

Figura 20 – Arquitetura definida.



Fonte: Imagem desenvolvida pelo autor.

A arquitetura, apresentada na Figura 20, consiste nos seguintes elementos:

– O **Player** representa o usuário que jogará o jogo. Ele utilizará dispositivos de realidade virtual imersiva, como HMD e seus controles, para se inserir e interagir com o ambiente virtual.

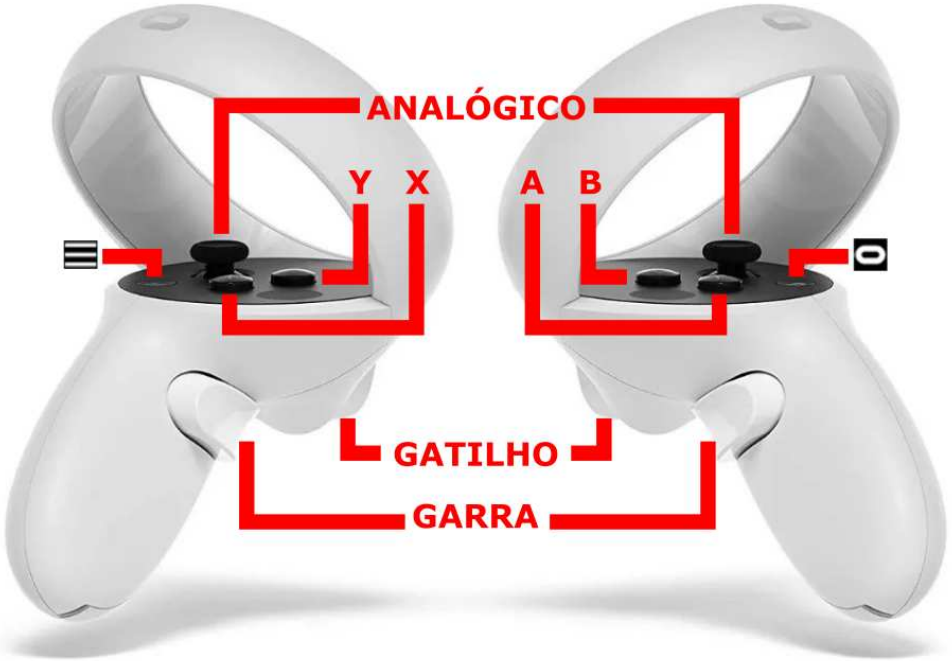
- A **VRGUI** é a interface utilizada pelo **Player**, que será responsável por capturar os movimentos do **Player** por meio de controles, permitir a interação com o ambiente virtual e retornar estímulos, sejam sonoros ou visuais, para a imersão.
- As interações realizadas pelo **Player**, a partir do uso dos controles e de sua própria movimentação no ambiente físico, são mapeadas para o ambiente virtual a partir da **VRGUI**, sendo replicadas pelo **Avatar**, que será o representante do **Player** no ambiente virtual, responsável por replicar suas ações. Este módulo encapsula todas as ações que o **Player** pode realizar, como a implementação da movimentação e interação dos controles.
- O **Avatar** interagirá com as **Weapons**, que são as representações das armas dentro do jogo.
- As **Weapons** possuem **Attributes**, que proporcionarão o comportamento e características de armas reais, como cadência[1], range[2] e quantidade de munição (Adams, 2014b). Exemplos de **Attributes** são a quantidade total de munição que a arma comporta, tendo que recarregar toda vez que a munição carregada na arma acabar; ou também restrições de períodos que o **Player** consegue atirar, simulando a cadência da arma.
- O **Scenario** é o conjunto de todos os objetos presentes no ambiente virtual.
- Todos os objetos são **Targets**, isto é, tudo presente no **Scenario** pode interagir com as ações das **Weapons**, definidas pelas **Rules**, que são os comportamentos dos **Targets** ao serem atingidos ou não. Como exemplo, uma lâmpada explode quando é acertada por uma pistola, enquanto uma parede é destruída ao explodir uma granada próxima a ela.
- O **Environment** é o módulo responsável por capturar todas as ações realizadas no *Scenario*, gerando os dados e eventos próprios de cada ação realizada dentro do **Scenario** e será gerenciado pelo **Behaviors**, entidades que farão o controle das comunicações entre os módulos que compõem o **Environment**, e gerencia o progresso do **Player**.
- O **Event Manager** receberá os dados gerados pelo *Environment* e irá processá-los para gerar os efeitos visuais, auditivos ou hápticos que serão tratados pelo **manager** específico, como o **Sound Manager** gerenciando o som, **Visual Effect Manager** gerenciando os efeitos visuais, **Haptic Manager** gerenciando respostas hápticas dos controles e **Physics Manager** gerenciando a física do jogo, além de fazer os cálculos dos scores do **Player**.

[1] Quantidade de disparos por minuto que a arma consegue realizar
[2] Distância que o disparo consegue percorrer

5.1.3 Construção de protótipos

Com a definição da arquitetura, foi possível desenvolver dois protótipos para uma avaliação inicial da facilidade e extensibilidade no uso da arquitetura. Os protótipos criados foram o *Estande de tiro fechado* e o *Estande de tiro 360º*. Esses protótipos foram escolhidos por fazerem parte de treinamentos militar e também serem utilizados nos tutoriais de Tactical Shooters como Call of Duty. Todos os protótipos utilizam o cálculo de raycast[3] implementado pelo motor de jogo Unity para simular o disparo da arma e a física do projétil.

Figura 21 – Relação dos botões presentes nos controles do Oculus Quest 2. Em ambos os controles, na frente, há o botão de gatilho, e no lado contrário à palma da mão, o botão de garra. Na parte superior do controle direito, estão os botões A, B e Quest (identificado pelo símbolo da Oculus), juntamente com o analógico. Já o controle esquerdo possui os botões X, Y e o menu em apps e experiências (identificado pelo símbolo de três linhas), também com um analógico.



Fonte: Imagem desenvolvida pelo autor.

A organização dos controles do Oculus Quest 2 segue a mostrada na Figura 21, que foi considerada na construção dos protótipos.

[3] Algoritmo que identifica a interseção entre objetos com linhas. Nele, é definido a posição inicial e a direção da linha, realizado o cálculo de interseção com todos os objetos presente na cena.

5.1.3.1 Protótipo 1 - Estande de tiro fechado

Este protótipo simula um estande de tiro, em uma sala fechada, em que o jogador fica atrás de uma bancada e de frente para um alvo (Figura 22). Ele permite que o jogador tenha um primeiro contato com o equipamento e se acostume com seus controles.

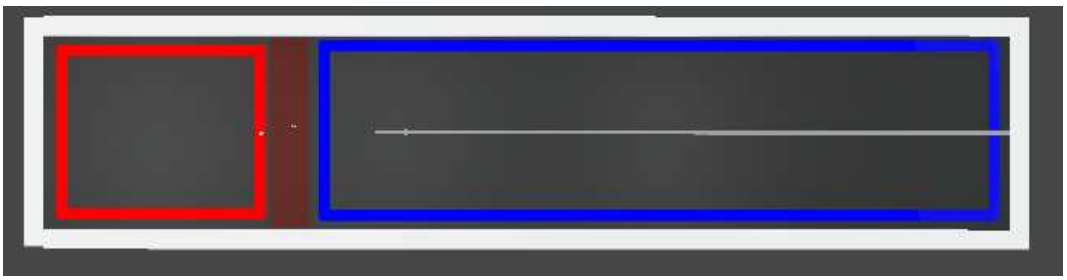
Figura 22 – Visão do jogador dentro do estande de tiro fechado.

Fonte: Imagem realizada pelo autor.

O jogador pode efetuar até trinta disparos em tempo ilimitado. A cada dez disparos, o alvo se afasta dentro do retângulo azul definido na Figura 23, e, após trinta disparos, o alvo retorna à posição inicial, e tanto a pontuação quanto as marcas de bala no alvo são exibidas no display do HMD. Para disparar, o jogador precisa pressionar o botão-gatilho (Figura 21), fazendo com que seja realizado o cálculo de raycast implementado pelo motor de jogo Unity.

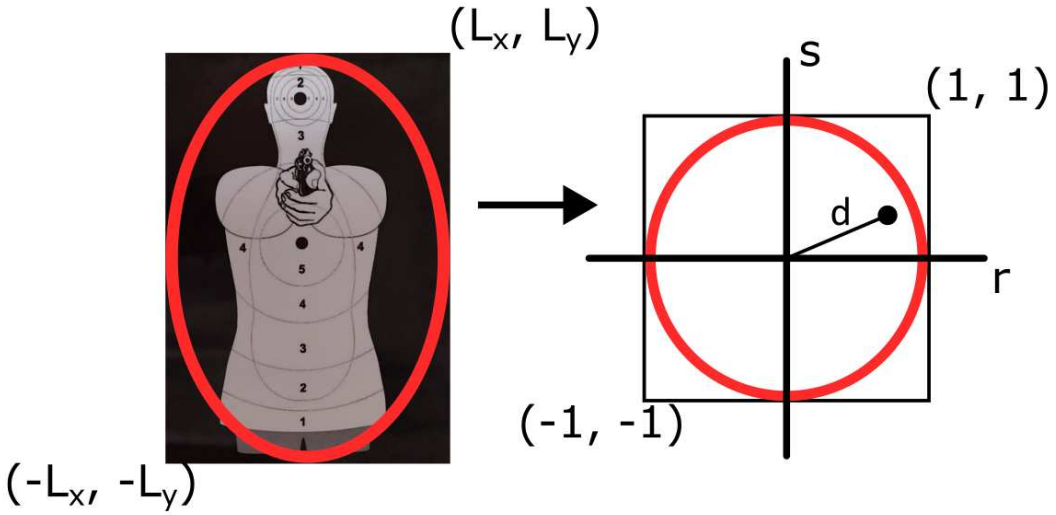
A pontuação é tanto maior quanto mais próxima a marca do tiro estiver do centro do alvo (o centro do peito que aparece na imagem mostrada na Figura 24). A região efetiva do

Figura 23 – Planta baixa do estande de tiro fechado; o quadrado vermelho representa a área em que o jogador pode caminha, e o retângulo azul representa a área em que o alvo se move.



Fonte: Imagem desenvolvida pelo autor.

Figura 24 – Cálculo da pontuação, onde as linhas vermelhas representam o threshold mínimo para pontuar.



Fonte: Imagem desenvolvida pelo autor.

alvo é considerada como um retângulo de arestas iguais a $2L_x$ e $2L_y$. Para calcular a pontuação de um dado disparo, primeiramente, determina-se a distância euclidiana entre o centro desse quadrado e a marca da bala, isto é, $d = \sqrt{r^2 + s^2}$, onde $r = \frac{x}{L_x}$, e $s = \frac{y}{L_y}$, em seguida, e substitui-se d na fórmula de pontuação $P = 5\dot{(}1 - min(d, 1))$. Assim, o círculo vermelho em coordenadas normalizadas ou a correspondente elipse vermelha em coordenadas não normalizadas delimitam a zona de pontuação. Note que, quando $d = 0$ (tiro no centro do alvo) a pontuação $P = 5$.

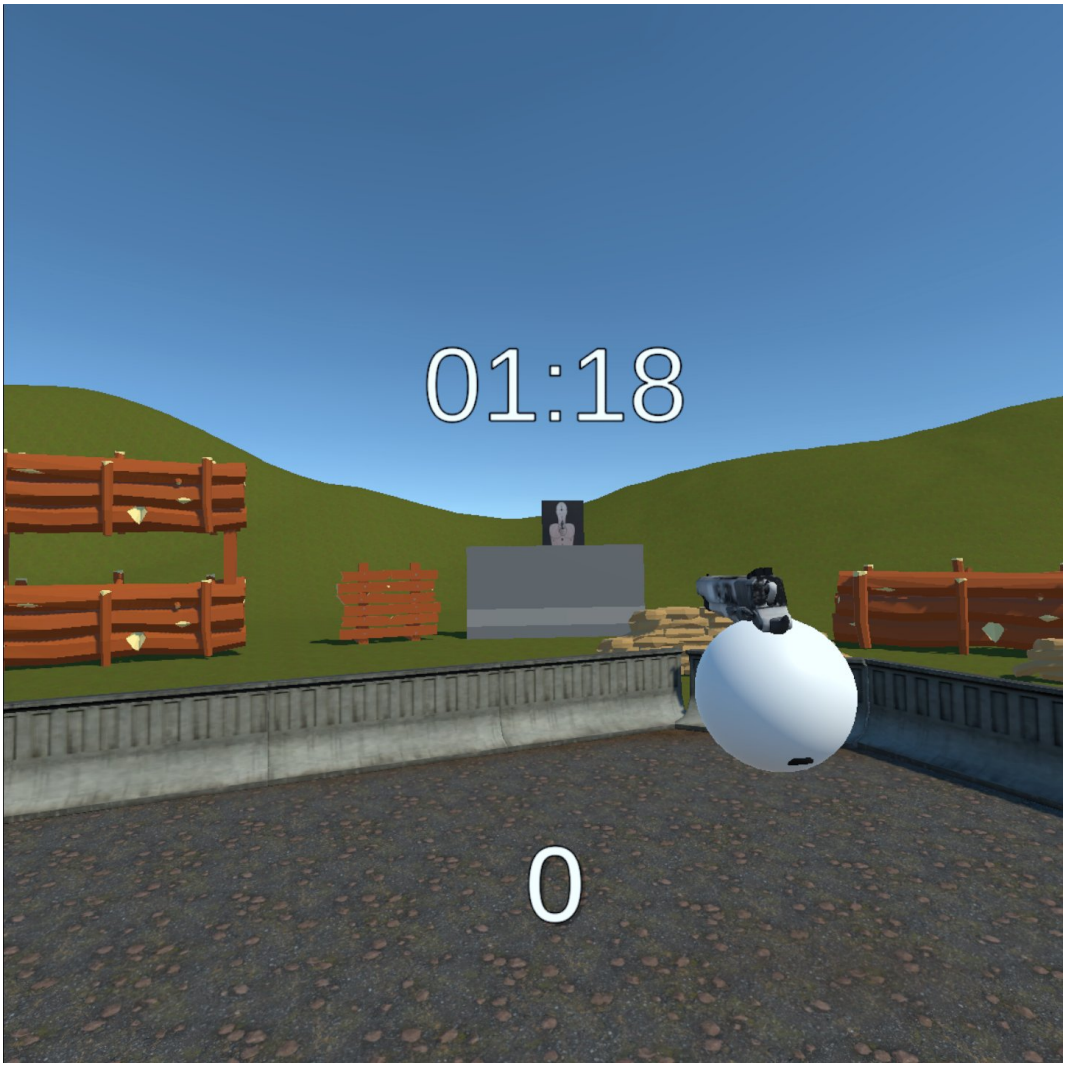
5.1.3.2 Protótipo 2 - Estande aberto 360º

Neste protótipo, o jogador se encontra em um ambiente aberto, rodeado por barricadas. Os alvos aparecem um por vez em locais predefinidos dentro da área do círculo azul representado na Figura 26, e emitem som para que o jogador perceba o alvo a ser atingido. O

alvo permanece na cena até ser acertado pelo jogador, momento em que um novo alvo aparece aleatoriamente. O objetivo desse protótipo é que o jogador acerte a maior quantidade de alvos possíveis.

Para acertar os alvos, o jogador precisa pressionar o botão-gatilho (Figura 21), fazendo com que seja realizado o cálculo de raycast implementado pelo motor de jogo Unity.

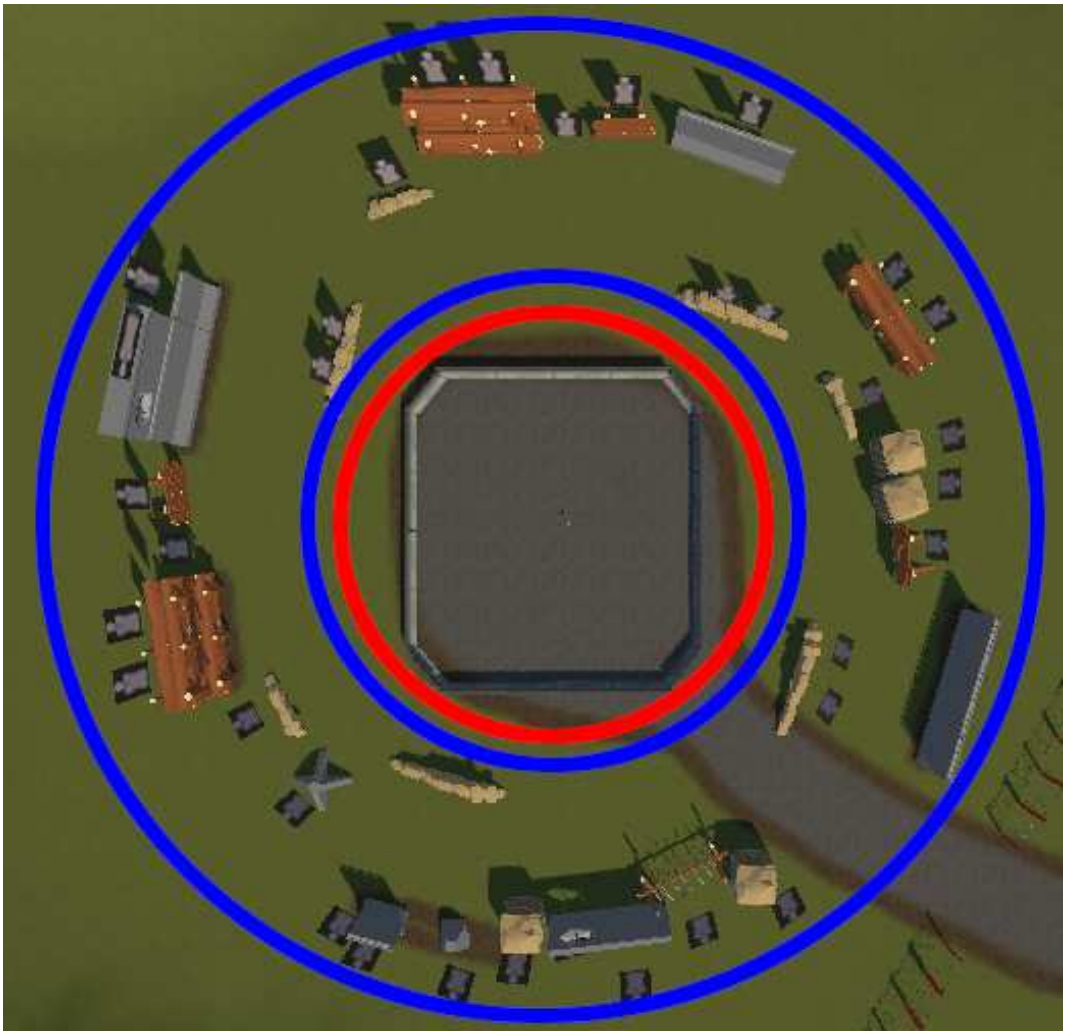
Figura 25 – Visão do jogador dentro do estande de tiro aberto 360º.



Fonte: Imagem realizada pelo autor.

Cada partida tem duração de 1 minuto e 30 segundos e possui quantidade ilimitada de munição. Após esse tempo, qualquer alvo que esteja visível desaparece, e a pontuação do jogador é mostrada no HMD, calculada como a quantidade de alvos acertados dentro do tempo.

Figura 26 – Planta baixa do estande aberto 360º: O círculo vermelho representa a área em que o jogador pode se locomover, e o círculo azul é a área onde os alvos aparecem.



Fonte: Imagem desenvolvida pelo autor.

5.1.4 Análise dos protótipos

Para validar os protótipos de Tactical Shooters desenvolvidos a partir da arquitetura proposta, foram selecionados seis usuários por amostragem por conveniência. Os testes foram realizados em dias e locais diferentes, conforme a disponibilidade de cada participante. Os testes ocorreram em um local residencial no dia 28 de maio de 2023, na Secretaria da Segurança Pública de Defesa Social do Estado do Ceará no dia 30 de maio de 2023, e no Departamento de Estatística de Matemática Aplicada - bloco 910 no dia 1 de junho de 2023. Os perfis dos participantes serão descritos a seguir:

– **Usuário 1 (*U1*)**: Homem de 24 anos, estudante de Direito, joga casualmente vários estilos de jogos, incluindo FPS. Já havia usado óculos de RV anteriormente e não possui

experiência com armas reais.

- **Usuário 2 (*U2*)**: Mulher de 35 anos, assistente técnica em perícia forense, atualmente não joga nenhum tipo de jogo eletrônico, mas gostava de jogar na adolescência. Nunca tinha usado nenhum dispositivo de RV e tinha pouca experiência com armas reais. Relatou que possuía miopia.
- **Usuário 3 (*U3*)**: Homem de 57 anos, policial, não joga nenhum tipo de jogo eletrônico. Já havia experimentado um HMD anteriormente e manipula armas reais há 21 anos.
- **Usuário 4 (*U4*)**: Homem de 41 anos, policial, não joga jogos eletrônicos, mas pratica Airsoft. Nunca havia usado qualquer dispositivo de RV anteriormente e tem 10 anos de experiência com armas reais.
- **Usuário 5 (*U5*)**: Homem de 24 anos, estudante de Ciência da Computação. Joga jogos casuais e se considera experiente em FPS, mas nunca tinha usado HMDs e não tem experiência com armas reais.
- **Usuário 6 (*U6*)**: Homem de 24 anos, estudante de Ciência da Computação, joga jogos casuais. Já tinha usado HMDs anteriormente e não tem experiência com armas reais.

Como cada usuário realizou os testes em locais e dias diferentes, a Tabela 3 mostra a relação entre as condições do ambiente em que foram realizados os testes e a experiência com manipulação de armas e com RV de cada um.

Os testes foram realizados em duas partes. Na primeira parte, os usuários jogaram os protótipos, realizando as seguintes etapas:

- Antes de começar o primeiro teste de cada um dos protótipos, foram explicadas as regras e sistemas de pontuações específicas dos mesmos.
- Cada participante testou cada protótipo individualmente e separadamente para não haver influências de terceiros.
- Cada participante jogou duas vezes em cada um dos protótipos.

Na segunda parte do teste, foi realizada uma entrevista semi-estruturada para coletar a opinião do usuário sobre sua experiência com o protótipo, realizando as perguntas:

- Você gostou da experiencia?
- O que mais gostou na experiencia?
- O que não gostou na experiencia?
- Sentiu dificuldade, medo ou adrenalina na experiencia?
- Como poderia melhorar a experiencia?

Tabela 3 – Relação entre as experiências com jogos e RV dos usuários e as particularidades dos espaços em que foram realizados os testes.

Usuários	Experiência com jogos	Experiência com RV	Experiência com armas de fogo	Especificidades do espaço e do usuário
U1	Casual	Já utilizou	Sem experiência	2.6 m² com tevê e móveis ao redor
U2	Somente durante a adolescência	Nunca jogou	Já participou de aulas de tiro, porém não utiliza armas de fogo no dia a dia	Sem objetos ao redor, mas ambiente barulhento e pessoa com miopia
U3	Não joga	Já utilizou	21 anos	Sem objetos ao redor, mas ambiente barulhento
U4	Não joga	Nunca jogou	10 anos	Sem objetos ao redor, mas ambiente barulhento
U5	Casual e experiente em Shooters	Nunca jogou	Sem experiência	Computadores e móveis ao redor
U6	Casual	Já utilizou	Sem experiência	Computadores e móveis ao redor

5.1.5 Resultados

Os resultados específicos, obtidos a partir da análise das entrevistas semi-estruturadas, serão descritos abaixo, com os principais pontos levantados pelos participantes, seguidos de seus comentários.

Todos os participantes gostaram de usar os protótipos e se sentiram imersos na experiência, como pode ser avaliado nos relatos a seguir:

- *U1* - "Achei bem interessante. Para quem é acostumado a jogar todos os tipos de jogos, jogos normais e jogos de realidade virtual, é muito bacana essa experiência. Eu gostei pois está bem próximo da realidade: a arma está boa, os gráficos estão bons também."
- *U2* - "Foi bem diferente, bastante interessante, foi muito realista e conseguir localizar os alvos foi bem interessante. Eu gostei muito. Foi bem desafiador com os movimentos que você faz, foi muito bom!"
- *U3* - Como o ambiente de testes onde *U3* participou era barulhento e havia pouco espaço, ele comentou: "Muito bom! Muito bom! Não ouvi o disparo devido ao barulho e acrescentar para a nossa realidade uma coisa mais dinâmica (ambiente com mais espaço para movimentação e com alvos se locomovendo), não que seja parado, em que você ande e tenha outro ambiente. O parado (ambiente com pouco espaço de movimentação e alvos

estáticos) é muito bom, mas se acrescentar o dinâmico fica melhor."

- *U4* - "Muito bom, uma experiência muito boa. Simula realmente uma situação real, se o tiro for estático, parado. Realmente simula a massa e a alça de mira de uma arma de fogo de verdade."
- *U5* - "É muito legal porque é muito próximo da experiência que você tem no jogo mesmo. Porque, por exemplo, você mira, você tem muito mais liberdade com a arma do que você teria com o gamepad ou mouse. Então, a possibilidade de você fazer uma jogada, de virar rápido, de fazer tudo isso é muito mais ágil do que eu faria em um controle ou mouse e teclado. O mouse e teclado já são muito ágeis e muito precisos, mas com a precisão do controle do VR é muito melhor, muito melhor mesmo! E fora a experiência de você estar em uma perspectiva de primeira pessoa e de fazer tudo isso é bem mais imersivo."
- *U6* - Enquanto realizava os testes, o participante *U6*, analisando o modelo 3D da arma, percebeu que havia a alça da arma, e que ele poderia ajudá-lo a mirar. *U6* comentou: "O que eu mais gostei foi a riqueza de detalhes, a dinâmica do jogo em que os alvos ficam aparecendo em locais aleatórios. Mas toda aquela sensação trás um certo realismo. E também porque o jogo possui os que treinam não só a questão visual, como também pela escuta dos barulhos que indicam onde o alvo está e que temos que girar em 360°."

No entanto, os participantes que não tinham experiência com armas reais relataram dificuldades ao mirar:

- *U1* - "A dificuldade que eu senti foi a experiência com tiro. Eu acho que para alguém que teve um treinamento com armas, isso se tornaria mais fácil por o jogo ser próximo à realidade. Então, para mim que não tive nenhum treinamento com armas, foi um pouco mais difícil. Nos primeiros tiros eu tive que ir pegando o jeito, errando os tiros para depois, no decorrer do jogo, eu ir me acostumando e conseguindo atirar bem."
- *U5* - "A falta do rastro da bala foi algo difícil, porque eu não sabia o quanto calibrar a arma."
- *U6* - "No começo eu fiquei um pouco confuso. Eu não tenho nenhuma experiência com armas. Eu não gostei de saber que a mira tem uma fenda, no topo da arma, que a gente usa como se fosse para encaixar no ponto em que a gente quer disparar."

Em particular, os participantes *U5* e *U6* tiveram a impressão de que estavam mirando corretamente, mas o tiro não acertava o local que eles esperavam. Essa impressão não foi relatada pelos participantes que tinham experiência com armas (*U2*, *U3* e *U4*). Podemos supor, assim,

que essa impressão pode ter sido causada pela falta de experiência em como mirar com uma arma real.

O participante *U3* relatou que achou interessantes os protótipos, inclusive indicando que o Estande de Tiro Fechado refletia bem o treinamento policial. Ele disse que "Aquele do disparo (o protótipo Estande de Tiro Fechado) a gente faz esse treinamento. Tantos metros, e vai aumentando e aumentando. A gente faz esse treinamento no real."

O participante *U4* relatou que o jogo/simulador é bem próximo ao treinamento real, porém não gostou tanto, indicando que os cenários empregados apresentavam poucos objetos presentes nas possíveis situações do dia a dia de um policial: "O cenário poderia ser mais evoluído, com mais obstáculos. O que poderia melhorar seria colocar outros tipos de cenários, outros tipos de biomas. Por exemplo, simular um cenário de uma via pública, um cenário da cidade, pois o cenário que está aqui (jogo/simulador) é de interior. Poderia colocar ruas com postes de iluminação, pessoas passando, para dificultar; evoluir o treinamento, pois nosso treinamento real é assim". No entanto, apesar de ser um jogo e não um simulador, ele indicou que a experiência fornecida pelos protótipos atuais já seria bastante adequada a iniciantes na prática de tiro: "para pessoas que não têm experiência, os cenários atuais já estão ideais".

Os candidatos *U2*, *U5* e *U6* disseram que sentiram algumas dificuldades para identificar os alvos. Por exemplo, a candidata *U2* indicou que provavelmente sentiu isso devido à miopia. Já *U5* informou que sua dificuldade foi por conta das cores utilizadas, que muitas vezes faziam alguns alvos virtuais se misturarem visualmente aos objetos presentes especificamente no cenário do Estande Aberto 360º: "O que eu não gostei foi a questão da paleta de cores. Por exemplo, aparece um inimigo que está em um cartaz que possui um fundo cinza, mas o muro possui quase a mesma paleta. Então, em boa parte eu tive dificuldade de identificar o que estava aparecendo atrás, se era inimigo mesmo ou era complemento do muro; porque como você se vira muito rápido, você tende a ver um destaque de cor e não há esse destaque necessário. Então, isso acabou me confundindo em vários aspectos nesse cenário."Do mesmo modo, *U6*, também relatou essa mesma dificuldade por conta das cores presentes nesse protótipo; entretanto, ele não achou que dificultou a identificação dos alvos. No entanto, aparentemente tal fato não atrapalhou a diversão, já que *U3* - que, conforme já dito, teve problemas em ouvir devidamente os sons emitidos pelos alvos - acabou gostando mais especificamente desse protótipo.

Considerando especificamente a presença de objetos ao redor do espaço físico utilizado, os participantes *U1* e *U5* se sentiram tensos por terem medo de esbarrar em algo perto

dos limites de locomoção.

5.2 Fase de refinamento

Após essa primeira validação dos protótipos, percebeu-se que os Tactical Shooters desenvolvidos a partir da arquitetura são divertidos e imersivos. Dessa forma, foi possível desenvolver o framework de software utilizando a arquitetura. Após a finalização do framework de software, desenvolveu-se mais um protótipo, o Tiro ao Polvo, e também foi realizado um workshop com desenvolvedores de jogos para validar a usabilidade do framework de software. Estes passos serão descritos a seguir.

5.2.1 Desevolvimento do Framework

Definida a arquitetura, foi possível desenvolver o framework de software. Para tal, utilizou-se como base três padrões de projetos já consolidados na academia: Singleton, Observer e Template Method. Esses padrões foram escolhidos devido à facilidade de uso e implementação do Singleton, à modularidade proporcionada pelo Observer e à reutilização de código oferecida pelo Template Method. A seguir, o framework de software, seu desenvolvimento, uso e validação serão discutidos.

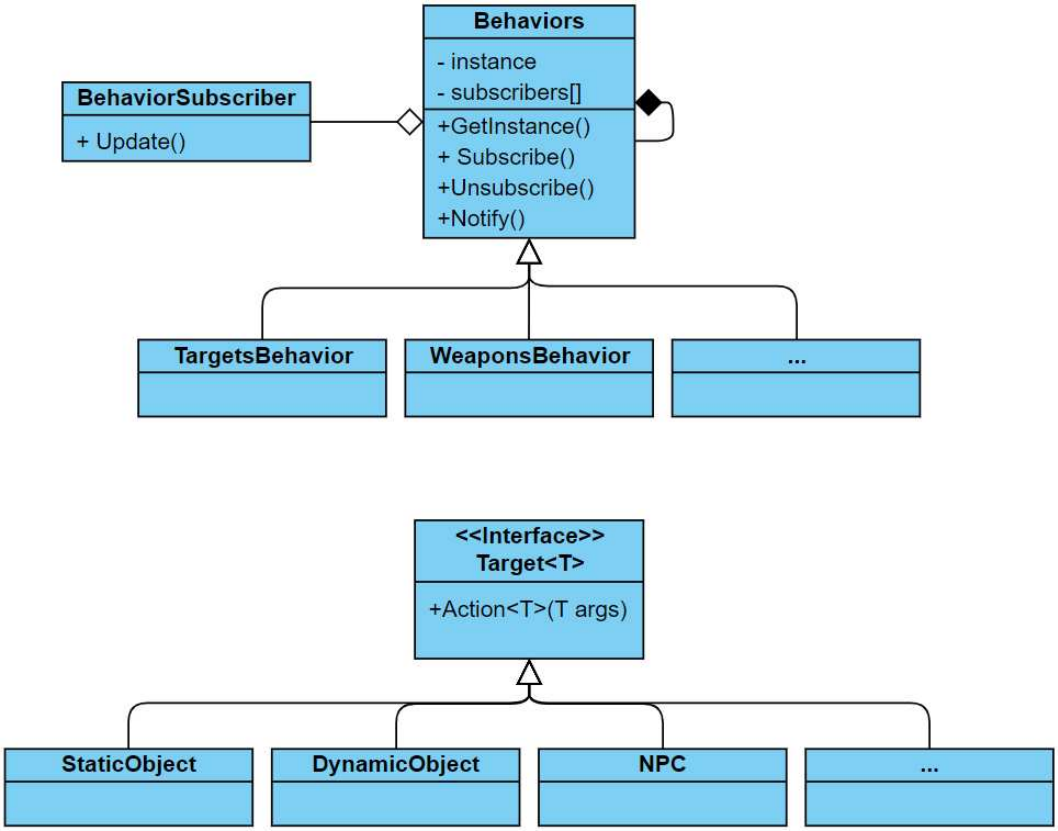
5.2.1.1 Descrição do Framework de software

Como mostrado na Figura 27, o framework de software foi definido utilizando os padrões Observer, Singleton e Template Method, dividido entre as classes Behaviors, Behavior-Subscriber e Target.

As classes Behaviors funcionam como interface de atualização dos módulos da arquitetura. Sendo assim, deverá haver uma instância de Behavior para cada módulo que deseja enviar atualizações. Elas utilizam o padrão Observer para implementar as interfaces de comunicação dos módulos, funcionando como classes publicadoras[4]. O padrão Singleton foi escolhido para que qualquer classe dentro do projeto possa se cadastrar nelas, recebendo as atualizações referentes ao módulo desejado, já que cada Behavior possuirá apenas uma instância global, e para que todas as classes do módulo possam utilizá-la como interface para publicar suas atualizações.

[4] Classes do padrão Observer responsáveis por atualizar as classes cadastradas com as atualizações de seus dados.

Figura 27 – Diagrama de classe UML do Framework de Software desenvolvido.



Fonte: Digrama de classe UML desenvolvida pelo autor.

A classe BehaviorSubscriber também utiliza o padrão Observer e é a classe que se inscreverá nas Behaviors. Toda outra classe que desejar receber atualizações de algum módulo precisa herdar da BehaviorSubscriber.

Já as classes Target são abstratas, de modo que toda classe herdada precise implementar o método Action, responsável por estabelecer as ações de seus objetos quando forem acertados por disparos das armas. A classe Target utiliza o padrão Template Method para dar ao programador liberdade para definir quais dados serão importantes de passar como parâmetro no método Action, além de permitir a reutilização de código entre as subclasses.

5.2.1.2 Implementação do Framework de software

O framework de software independe da linguagem e ferramenta utilizada, dando liberdade ao desenvolvedor de utilizá-lo em qualquer ambiente. No trabalho atual, por utilizar o motor de jogo Unity, a implementação do framework de software foi realizada utilizando a linguagem C#. Por conta disso, foi tomada a decição de utilizar o tipo Delegate do C# para implementar o padrão Observer (Microsoft, 2022), onde as classes assinantes cadastram seus métodos em atributos Delegate das classes publicadoras; assim, toda vez que um atributo

Delegate da classe publicadora for invocado, serão chamados os métodos cadastrados nele. Isso dá a vantagem de que qualquer classe possa cadastrar métodos nas classes publicadoras, sem que haja necessidade de implementar a classe BehaviorSubscriber.

5.2.2 Novo protótipo: Tiro ao polvo

Com o framework de software definido, para uma primeira validação, foi desenvolvido o protótipo Tiro ao polvo. Neste protótipo, o jogador se encontra em um espaço aberto, em que à sua frente está um polvo, que o rodeará dentro do círculo azul, como representado na Imagem 29, atirando projéteis na direção do jogador. O objetivo desse protótipo é acertar a maior quantidade de vezes o polvo.

Aqui, o jogador possui tempo e munição ilimitados, mas toda vez que o polvo é acertado, ele percorre metade do círculo azul ao redor do jogador (Imagem 29), passa a rodear o jogador mais rapidamente, diminui o intervalo entre os disparos e troca o sentido em que rodeia o jogador. A partir do terceiro acerto, o polvo passa de tempos em tempos a trocar o sentido em que rodeia o jogador. Caso um disparo atinja o jogador, ele perde uma vida, e ao chegar a zero, o jogo acaba. O jogador possui a possibilidade de acertar o projétil disparado pelo polvo, que desaparecerá e não causará dano. Ao fim da partida, a pontuação será mostrada, onde cada acerto no polvo soma 10 pontos.

5.2.3 Utilização do Framework de software no Tiro ao Polvo

Para a elaboração do protótipo Tiro ao Polvo, foi utilizado o framework de software proposto, seguindo a arquitetura de software definida na Seção 5.1.2. Para isso, foram utilizadas as bibliotecas XR Interaction Toolkit[5] e XR Core Utilities[6] do motor de jogos Unity para a implementação da *VRGUI* e das interações do *Avatar* com o ambiente e com a *VRGUI*.

A *Weapon* foi implementada como uma classe, contendo o método *Shoot*, chamado toda vez que o gatilho do controle direito do Quest 2 fosse pressionado (Figura 21). Ao ser chamado, o método calcula, a partir da posição do controle no mundo virtual, um raio utilizando o cálculo de raycast, simulando o caminho percorrido pelo disparo, procurando o primeiro objeto *Target* que intercepte o raio. Essa função é calculada pelo gerenciador de física do motor de jogos Unity, que representa o *Physics Manager* da arquitetura. Caso encontre, o método

[5] Biblioteca que implementa funções de interações em RE. https://docs.unity3d.com/Packages/com.unity.xr.interaction.toolkit3.0/
[6] Biblioteca que implementa os principais métodos utilizados no desenvolvimento de RE. https://docs.unity3d.com/Packages/com.unity.xr.core-utils2.2/manual/index.html

Figura 28 – Visão do jogador dentro do protótipo tiro ao polvo.

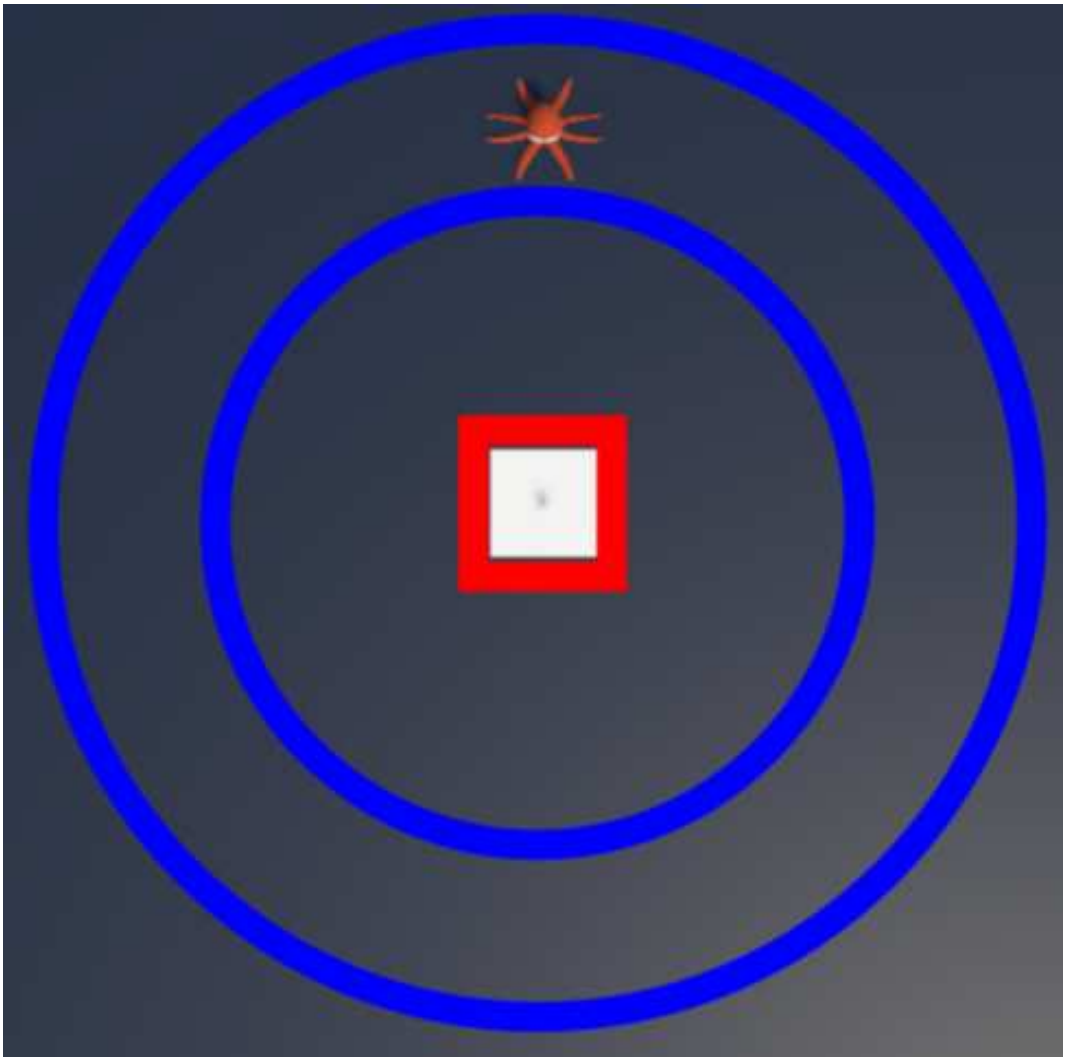
Fonte: Imagem realizada pelo autor.

Action<int>(int args) do *Target* encontrado é chamado, em que args é o valor de quanto de vida é retirado do polvo pelo disparo, que no caso dessa arma inicial, retira um ponto de vida. Ao fim, um som é emitido utilizando o componente Audio Source da Unity, representando o *Sound Manager*.

Para o polvo, foi criada a classe *NPC*, que possui métodos representando suas *Rules*. A cada frame, o polvo rotaciona ao redor do *Avatar* a uma distância definida. Também guarda a quantidade de vezes que foi acertado pelo **Player**, para que, a partir do terceiro acerto, comece a trocar o sentido em que caminha de forma aleatória. Seu método *Action<int>(int args)* o faz percorrer a distância de metade do círculo azul ao redor (Imagem 29) do *Avatar* e utiliza o *BehaviourTargets* para informar que foi acertado por um disparo.

A classe responsável por guardar a pontuação do jogador é a *Environment*, que se cadastra na *BehaviourTargets*, e ao receber a atualização de que houve acerto em algum alvo,

Figura 29 – Planta baixa do protótipo Tiro ao Polvo. O círculo azul representa a área em que o polvo rodeia o jogador, enquanto o quadrado vermelho representa a área em que o jogador pode se locomover.



Fonte: Imagem realizada pelo autor.

acrescenta pontos ao jogador.

5.2.4 Workshop

Para a devida validação do framework de software junto ao seu público-alvo, desenvolvedores de jogos, foi realizado um workshop com duração de duas horas no Laboratório 1 do prédio do curso de Sistemas e Mídias Digitais da UFC no dia 06/05/2024.

A partir do workshop foi desejado validar o quão fácil seria utilizar o framework proposto, e receber ideias de melhorias, para assim, saber se o framework pode ser utilizado no desenvolvimento de Tactical Shooters de RV. A seguir, são apresentadas as características dos desenvolvedores que participaram do workshop:

- **Perfil 1 (*P1*)**: Desenvolvedor homem de 20 anos de idade, graduando em Ciência da Computação pela Universidade de Fortaleza, com 1 ano de experiência com programação e 4 meses com o motor de jogo Godot. Afirmou saber definir o que são Tactical Shooters e não possuía experiência com desenvolvimento em RV.
- **Perfil 2 (*P2*)**: Desenvolvedor homem de 49 anos de idade, doutorando em Ciência da Computação pela UFC, com 20 anos de experiência com programação e 5 meses com o motor de jogo Unity. Afirmou saber definir o que são Tactical Shooters e não possuía experiência com desenvolvimento em RV.
- **Perfil 3 (*P3*)**: Desenvolvedor homem de 19 anos de idade, graduando em Engenharia Elétrica pela UFC, com 1 ano e meio de experiência com programação e 8 meses com o motor de jogo Unity. Afirmou saber definir o que são Tactical Shooters e não possuía experiência com desenvolvimento em RV.
- **Perfil 4 (*P4*)**: Desenvolvedor homem de 20 anos de idade, graduando em Sistemas e Mídias Digitais pela UFC, com 4 anos de experiência com programação e que utilizou o motor de jogo Unity apenas em um workshop de 1 dia. Afirmou não saber definir o que são Tactical Shooters e não possuía experiência com desenvolvimento em RV.
- **Perfil 5 (*P5*)**: Desenvolvedor homem de 22 anos de idade, graduando em Sistemas e Mídias Digitais pela UFC, com 5 anos de experiência com programação e 2 meses com o motor de jogo Unity. Afirmou não saber definir o que são Tactical Shooters e possuía experiência com desenvolvimento em RV por participar de um grupo de pesquisa em RV do seu curso.
- **Perfil 6 (*P6*)**: Desenvolvedora mulher de 25 anos de idade, graduanda em Sistemas e Mídias Digitais pela UFC, com 1 ano de experiência com programação e sem experiência com motores de jogos. Afirmou não saber definir o que são Tactical Shooters e não possuía experiência com desenvolvimento em RV.

O workshop foi separado em cinco momentos, sendo eles:

- Entrevista inicial para colher as informações pessoais e experiências dos participantes em desenvolvimento de jogos e em RV.
- Apresentação do protótipo Estande de tiro aberto 360º, descrito na Seção 5.1.3.2, permitindo aos participantes jogá-lo (Figuras 30 e 31). Este protótipo foi escolhido pois é o único com tempo pré-determinado, o que possibilita um teste com tempo controlado, e por aparecerem alvos ao redor, o que torna a experiência mais dinâmica (como mostrado na

Figura 30 – Participante do workshop jogando protótipo.

Fonte: Fotografia desenvolvida pelos autores.

Figura 31 – Participante do workshop jogando protótipo.

Fonte: Fotografia realizada pelo autor.

Seção 5.1.5).

– Apresentação de contextualização sobre o que são Tactical Shooters e apresentação do framework de software (Figura 32). A apresentação utilizada no workshop está disponível

Figura 32 – Apresentação do framework.

Fonte: Fotografia realizada pelo autor.

Figura 33 – Grupo focal.

Fonte: Fotografia realizada pelo autor.

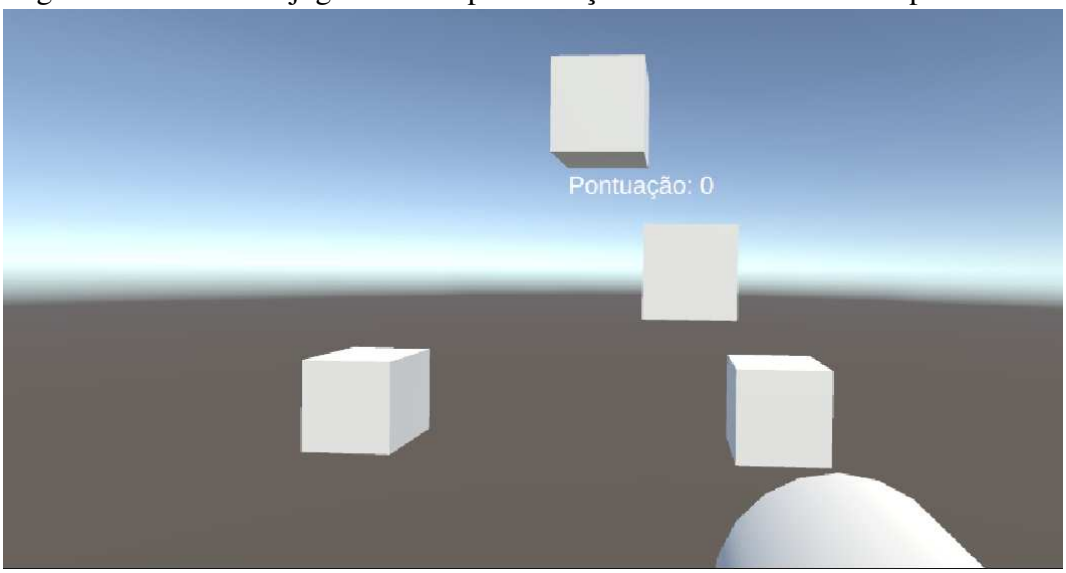
no Apêndice A.

– Implementação de um jogo teste utilizando o framework Unity. Neste momento, o palestrante codificou o jogo teste, explicou, discutiu e tirou dúvidas dos participantes. Por exemplo, ao codificar o método *Action* da classe *Target*, foram discutidas possibilidades de respostas visuais e cálculo de pontuação, mostrando exemplos de como poderiam ser implementados e tirando dúvidas sobre como ocorria a comunicação entre as classes do jogo. Dessa forma, todos participaram da programação. Após o desenvolvimento do jogo teste, os participantes foram convidados a testá-lo.
– Realização da entrevista de grupo focal para colher a opinião dos participantes sobre o framework de software apresentado (Figura 33). A entrevista de grupo focal foi escolhida dado o tamanho do grupo de participantes, que apesar de não ser grande, era suficiente para a validação desejada.

5.2.4.1 Implementação realizada no workshop

Na fase de implementação, foi escolhido fazer um mini jogo de tiro ao alvo (Figura 34), onde à frente do jogador há alguns cubos que o jogador precisa acertar para fazê-los desaparecer e pontuar. Este mini jogo foi escolhido devido à facilidade de implementação e à baixa quantidade de requisitos para desenvolver.

Figura 34 – Visão do jogador na implementação realizada no workshop.



Fonte: Imagem capturada pelo autor.

Neste protótipo, um cilindro representa a *Weapon* utilizada pelo *Avatar*, seguindo as mesmas regras do protótipo Tiro ao Polvo (Seção 5.2.3). Os cubos foram implementados

como *Targets*, nos quais, quando seu método *Action* é chamado, o cubo é removido da cena e é chamado o método de atualização do *TargetBehaviors*, que por sua vez chamará o método *Update* dos objetos cadastrados. Por fim, foi implementada a classe *Environment*, que guarda e faz os cálculos da pontuação; esta classe se cadastra no *TargetBehaviors*, e toda vez que o método *Update* da classe *TargetBehaviors* é chamado, a classe *Environment* acrescenta um ponto ao jogador.

5.2.4.2 Perguntas do grupo focal

Foram realizadas quatro perguntas para os participantes do grupo focal, sendo as duas primeiras perguntas mais gerais para servir como "quebra de gelo"(Souza, 2020), fazendo com que os participantes se sentissem confortáveis de expressar suas opiniões, e as duas últimas perguntas mais especificas, seguindo uma técnica de funil (Gondim, 2002), sendo elas:

- ***Q1***: Qual sua opinião sobre o framework de software apresentado?
- ***Q2***: Acredita se haveria algum cenário em que o framework de software apresentado não contemplaria?
- ***Q3***: Qual foi sua maior dificuldade em entender o framework de software apresentado?
- ***Q4***: Haveria alguma sugestão para o melhoramento do framework de software apresentado?

Após realizar cada uma das perguntas, foi deixado que cada participante que se sentisse confortável respondesse à pergunta e discutissem a resposta um do outro, de forma que pudessem complementar o raciocínio da resposta.

5.2.4.3 Resultados do grupo focal

Para a pergunta *Q1*, sobre a opinião dos participantes em relação ao framework de software, *P5* afirmou que ficou impressionado o quão rápido foi possível desenvolver um protótipo funcional, levando 2 horas para conseguir explicar um sistema como esse, em que normalmente, projetos em RV demoram muito para serem desenvolvidos. Já a **P6** comentou que a presença do diagrama de UML facilitou o entendimento do framework de software, que apesar de não conseguir reproduzir o código implementado no workshop, conseguiria entender para conseguir gerir ou realizar um trabalho de UX[7] em harmonia com os desenvolvedores.

Para a pergunta *Q2*, sobre cenários não contemplados pelo framework de software, os participantes não conseguiram responder com certeza, mas *P5* apontou que em RV há várias

[7] User Experience

possibilidades que podem ser implementadas, já que é possível adaptar várias situações para *RV*, assim, a depender da situação, pode ser um desafio utilizar a arquitetura. Os participantes *P2* e *P5* levantaram o exemplo da movimentação do *Avatar*, onde não estava tão explícito como ficaria a implementação, porém há várias implementações em RV que implementam essa funcionalidade, como a movimentação livre[8] e teleport[9].

Já na pergunta *Q3*, sobre as dificuldades tidas para entender o framework de software, *P2* comentou que por sua falta de maturidade na utilização da Unity, a dificuldade que ele teve foi de separar o que era da API de RV da Unity e o que era implementação do framework, mas o *P5* comentou que era possível perceber que havia uma divisão entre a API da unity com o framework de software.

Por fim, na pergunta *Q4*, sobre sugestões de melhoramento do framework de software, o participante *P5* aconselhou a implementar novos recursos, como alguma espécie de mira ou qualquer tipo de suporte que torne o jogo mais acessível. Já **P2** acrescentou que poderia ser implementado o feedback tátil dos controles por exemplo. E a partir dessa última resposta, *P6* acrescentou se poderia desenvolver um manual que desse sugestões de implementações aos desenvolvedores que pudessem ser implementados utilizando o framework de software.

5.2.4.4 Discussão

Nem todos os participantes responderam todas as perguntas, porém, podemos perceber que os participantes que fazem o curso de Sistemas e Mídias Digitais, *P5* e *P6*, responderam à maior quantidade de perguntas, mesmo não possuindo muita experiência com desenvolvimento, principalmente o *P5*. Podemos supor que isso ocorreu porque o curso de Sistemas e Mídias Digitais é o curso mais próximo ao desenvolvimento de jogos dentre os cursos dos 6 participantes. Além disso, a experiência com desenvolvimento em RV do *P5* pode ter dado a ele uma opinião mais fundamentada sobre o framework de software proposto.

A partir da *Q1*, sobre a opinião dos participantes em relação ao framework de software proposto, podemos perceber que o framework de software apresentado pôde ser facilmente entendido, pelo comentário de *P6*, mesmo por alguém que não programadora, além de conseguir desenvolver a base de Tactical Shooter de RV em pouco tempo, pelo comentário de *P5*. Isso também fortalece os apontamentos de Wang e Nordmark (2015) sobre a importância de arquiteturas

[8] Movimentação livre é quando o Avatar imita a real movimentação do jogador no mundo real.

[9] Teleport é quando a movimentação ela é realizada pelo jogador ao se teleportar ao apontar para um espaço em que seja possível estar.

e frameworks de software para o desenvolvimento de jogos. Além disso, a presença de uma boa documentação ajuda no entendimento geral da arquitetura para diferentes atribuições do projeto, não somente para os desenvolvedores. Isso foi apontado por *P6*, que, apesar de não ter tanta experiência com desenvolvimento em geral ou em RV, conseguiria conversar sobre o framework de software com desenvolvedores sem problemas de entendimento.

Para a pergunta *Q2*, sobre se haveria algum cenário não contemplado pelo framework de software, podemos perceber que os participantes tiveram dificuldade de respondê-la, provavelmente pela falta de experiência com o framework, chegando a respostas inconclusivas. Mas essa pergunta ainda serviu para que os participantes se desinibissem para emitir suas opiniões. Quanto aos apontamentos sobre movimentação, podemos perceber que é necessário um detalhamento maior da arquitetura, deixando pontos como o da movimentação mais explícitos. Isso porque, como foi apresentado, estava implícito que a movimentação seria implementada pelo *Avatar* a partir dos comandos dados pelo *Player* ao utilizar o *VRGUI*.

Na pergunta *Q3*, sobre as dificuldades em entender o framework de software, é possível perceber que, para conseguir utilizar a arquitetura sem tantos problemas de entendimento, é necessário um conhecimento básico da engine que esteja sendo utilizada e sua API de RV. *P2*, que não possuía experiência com desenvolvimento em RV, levantou essa dificuldade de separar o que é implementação da API e o que é implementação do framework de software. Por outro lado, *P5*, que possui experiência com desenvolvimento em RV, conseguiu perceber melhor a divisão de cada coisa.

Com as respostas da pergunta *Q4*, sobre sugestões de melhoramento para o framework de software, percebe-se que se pode estender a pesquisa para alternativas de facilitação da utilização do framework de software, com desenvolvimento de tutoriais e manuais que tragam sugestões de implementações utilizando as mais diversas ferramentas disponíveis nos HMDs, como resposta tátil dos controles, melhorando assim a documentação.

5.2.5 Implementação pós-workshop

Atualmente, dois alunos de graduação em Ciência da Computação da UFC, Diego Caracas e Edson Coelho, estão desenvolvendo um protótipo utilizando o framework de software apresentado neste trabalho (Figura 35).

O intuito é desenvolver um protótipo mais lúdico, voltado para crianças, utilizando novas técnicas como a remoção do raycast do cálculo de interseção dos disparos realizados pelo

Figura 35 – Desenvolvedor testando o protótipo em desenvolvimento.

Fonte: fotografia realizada pelos alunos de graduação Edson Caracas e Diego Coelho.

jogador para utilizar o cálculo de colisão física entre objetos, implementado pelo motor de jogos Unity.

Dado o público-alvo, os modelos utilizados (Figura 36) e os comportamentos das *Weapons* não são realistas. Por exemplo, o disparo utilizado possui tamanho e velocidade (*Attributes*) próprios para possuírem características lúdicas, sendo grandes e lentos.

Figura 36 – Visão do jogador.

Fonte: Imagem capturada pelos alunos de graduação Edson Caracas e Diego Coelho.

6 RESULTADOS

Como resultado deste trabalho, foi possível chegar a um framework de software para desenvolvimento de Tactical Shooters utilizando RV. Este trabalho se diferencia nos seguintes pontos em relação aos trabalhos apresentados na Seção 3 e comparados na Tabela 2:

- O framework processual, OXREF, apresentado por Abeywardena (2023), tem como foco o desenvolvimento de softwares educativos em geral em RE, tendo em vista a utilização de ferramentas e arquiteturas de licenças abertas, possibilitando que conteúdos já desenvolvidos possam ser reutilizados por outros projetos educacionais. Não houve testes para a validação do framework.
- Já Marin-Vega *et al.* (2022) apresentam uma arquitetura, a ZeusAR, para a adaptação de jogos sérios para RA. Na avaliação da arquitetura, realizaram um questionário utilizando o método SUS com dez professores e facilitadores de trigonometria básica, com resultados promissores, onde a maioria dos participantes se sentiram confortáveis ao utilizar a ZeusAR, conseguindo validar a rapidez e facilidade de uso da mesma.
- O trabalho realizado por Bondarenko *et al.* (2024) teve como objetivo realizar uma arquitetura e uma proposta de framework de software para desenvolvimento de jogos sérios para RE. Para oferecer acessibilidade, modularidade e independência de hardware, a arquitetura foi pensada na utilização de bibliotecas abertas. Por fim, Bondarenko *et al.* (2024) apresentaram duas propostas de implementação utilizando o framework proposto, que, porém, ainda será implementado.

A solução aqui apresentada se foca em todo o processo de desenvolvimento de um framework de software para Tactical Shooters em RV, além de apresentar protótipos utilizando a arquitetura e o framework de software desenvolvidos. Também foi realizado workshop e entrevista para validar a usabilidade do framework de software proposto. A partir dessas atividades, é possível perceber a flexibilidade da arquitetura e do framework de software em desenvolver diferentes tipos de Shooters e Tactical Shooters, e também que há facilidade de manutenção, já que, como relatado na entrevista de grupo focal, o framework de software é fácil de ser utilizado e entendido, mesmo para grupos diferentes no desenvolvimento de jogos. A partir desta pesquisa não foi possível mensurar a escalabilidade, nem saber os limites do framework de software, necessitando a criação de novos protótipos, evoluir os já existentes e realizar novos workshops com desenvolvedores de jogos mais experientes com desenvolvimento RV e Shooters. Também não foi possível realizar a validação da qualidade dos protótipos

desenvolvidos, precisando de validações futuras.

Com esta pesquisa, espera-se uma contribuição ao desenvolvimento de jogos e Tactical Shooters de RV, fomentando o desenvolvimento de novas pesquisas e tecnologias a partir do facilitamento do desenvolvimento.

7 CONSIDERAÇÕES FINAIS

O gênero Shooters vem evoluindo há muito tempo, antes mesmo do desenvolvimento de jogos eletrônicos, sendo dividido atualmente por vários subgêneros, sendo um deles o Tactical Shooters, que por conta de buscar representações realísticas dos combates, é o subgênero mais próximo de jogos sérios para treinamento militar.

Apesar de ser um dos mais famosos gêneros de plataformas como PC e fomentar o mercado de jogos, há poucos trabalhos que tenham como objetivo facilitar o desenvolvimento de Shooters para RE, principalmente para Tactical Shooters para RV. Porém, isso não é uma questão exclusiva de Shooters; jogos sérios também possuem escassez de trabalhos acadêmicos, o que impede o desenvolvimento de novos jogos para RE.

A partir deste trabalho foi possível propor um framework de software passando por todas as etapas: construção do modelo conceitual, da arquitetura de software e do framework de software propriamente dito. Para o modelo, foram utilizados como base trabalhos anteriores, e a partir desses resultados foi possível desenvolver a arquitetura. Com a definição da arquitetura, foram construídos dois protótipos, Estande de tiro fechado e Estande aberto 360º, que foram testados e validados por usuários para analisar a usabilidade da arquitetura no desenvolvimento de Tactical Shooters em RV.

Por fim, foi desenvolvido um framework de software que permitiu a criação de mais um protótipo, Tiro ao Polvo, utilizado para exemplificar a utilização do framework de software. Após, foi realizado um workshop com desenvolvedores de jogos para poder validar e avaliar o framework de software. No final do workshop, foi realizada uma entrevista de grupo focal, onde foram colhidas as opiniões dos participantes sobre diferentes pontos do framework de software. A entrevista resultou em validações de informações importantes sobre o framework de software, possíveis melhorias e trabalhos futuros.

A partir do workshop, é possível notar que o framework de software facilita o desenvolvimento de Tactical Shooters em RV e é fácil de entendê-lo, além de conseguir uma gama de implementações para Tactical Shooters. Porém, se faz necessário mais pesquisas para conseguir mapear os limites do framwork de software, já que os participantes do grupo focal não conseguiram estimar cenários em que pudessem ser contemplados.

Como trabalhos futuros, necessita-se de mais implementações utilizando o framework de software para ser possível mapear seus limites e escalabilidade, que atualmente já está em andamento (Seção 5.2.5), realizando novos workshops com perfis de desenvolvedores

com mais experiência com desenvolvimento de jogos e em RV, e desenvolvendo novos protótipos e evoluindo os já existentes, contemplando novas funcionalidades, além de reimplementá-la em outros motores de jogos, como a Unreal. Também é desejado avaliar a qualidade dos jogos desenvolvidos com o framework de software, comparando com outros frameworks de software presentes no mercado que possam ser adaptados para desenvolvimento de Tactical Shooters. Por fim, realizar testes para saber se outros gêneros de jogos conseguem ser implementados utilizando o framework de software aqui apresentado.

As fragilidades deste trabalho que puderam ser percebidas são uma amostra pequena de desenvolvedores que participaram do workshop, mas que suas opiniões foram condizentes à presente pesquisa. Também a pequena quantidade de jogos desenvolvidos com o framework de software, mas que puderam ser utilizados para validar diversos pontos da arquitetura e do framework de software no desenvolvimento de Shooters e Tactical Shooters.

Por fim, como contribuições, espera-se que seja fomentado o desenvolvimento de mais arquiteturas e frameworks de software para desenvolvimento de aplicações em RE, principalmente na área de jogos, e em Tactical Shooters, uma vez que esse é um mercado que vem crescendo e fomentando o mercado de jogos em geral. Também destacar mais uma vez a necessidade de desenvolvimento de mais trabalhos acadêmicos na área.

REFERÊNCIAS

ABEYWARDENA, I. S. Oxref: Open xr for education framework. **The International Review of Research in Open and Distributed Learning**, v. 24, n. 3, p. 185–206, Sep. 2023. Disponível em: https://www.irrodl.org/index.php/irrodl/article/view/7109.

ADAMS, E. **Fundamentals of game design**. [*S. l.*]: Pearson Education, 2014.

ADAMS, E. **Fundamentals of Shooter Game Design**. [*S. l.*]: New Riders, 2014.

Apple2024 APPLE. 2024. Disponível em: https://www.apple.com/shop/buy-vision/apple-vision-pro.

AVEDON, E. M.; SUTTON-SMITH, B. **The Study of Games. John Wileys & Sons**. [*S. l.*]: Inc, 1971.

BANQUIERO, M.; VALDEOLIVAS, G.; TRINCADO, S.; GARCíA, N.; JUAN, M.-C. Passthrough mixed reality with oculus quest 2: A case study on learning piano. **IEEE MultiMedia**, v. 30, n. 2, p. 60–69, 2023.

BC, A. Enhancing military training through vr applications. **International Scientific Journal of Engineering and Management**, v. 03, p. 1–9, 05 2024.

BONDARENKO, Y.; KUTS, V.; PIZZAGALLI, S.; NUTONEN, K.; MURRAY, N.; O'CONNELL, E. Universal xr framework architecture based on open-source xr tools. In: DIECK, M. C. tom; JUNG, T.; KIM, Y.-S. (Ed.). **XR and Metaverse**. Cham: Springer Nature Switzerland, 2024. p. 87–98. ISBN 978-3-031-50559-1.

BROOKE, J. *et al.* Sus-a quick and dirty usability scale. **Usability evaluation in industry**, London, England, v. 189, n. 194, p. 4–7, 1996.

CALDIERA, V. R. B. G.; ROMBACH, H. D. The goal question metric approach. **Encyclopedia of software engineering**, p. 528–532, 1994.

CHAVEZ, I. C.; DAVIER, Y. W. C. Visual resonance: Fine art and 3d stereo pictoriality. In: **2011 Eighth International Conference Computer Graphics, Imaging and Visualization**. [*S. l.*: *s. n.*], 2011. p. 25–32.

DIAS, C. A. Grupo focal: técnica de coleta de dados em pesquisas qualitativas. **Informação & Sociedade**, Universidade Federal da Paraíba-Programa de Pós-Graduação em Ciência da ..., v. 10, n. 2, 2000.

FAGERHOLT, E.; LORENTZON, M. Beyond the hud-user interfaces for increased player immersion in fps games. 2009.

GAME, P. America's army pc game vision and realization. p. 40, 2004.

GAMMA, E.; HELM, R.; JOHNSON, R.; VLISSIDES, J. **Design patterns: elements of reusable object-oriented software**. [*S. l.*]: Pearson Deutschland GmbH, 1995.

GONDIM, S. M. G. Grupos focais como técnica de investigação qualitativa: desafios metodológicos. **Paidéia (Ribeirão Preto)**, SciELO Brasil, v. 12, p. 149–161, 2002.

ISABELL, W.; SIMONS, A.; STIEGLITZ, S. Virtual reality. **Business & Information Systems Engineering**, Springer Nature BV, v. 62, n. 5, p. 455–461, 2020.

ISO/IEC/IEEE. Iso/iec/ieee-42010:2022. **System and software engineering**, 2022.

JOHNSON, R. E.; FOOTE, B. Designing reusable classes. **Journal of object-oriented programming**, v. 1, n. 2, p. 22–35, 1988.

KELLY, J. W.; DOTY, T. A.; AMBOURN, M.; CHEREP, L. A. Distance perception in the oculus quest and oculus quest 2. **Frontiers in Virtual Reality**, Frontiers, v. 3, p. 850471, 2022.

KLUGE, M. G.; MALTBY, S.; KEYNES, A.; NALIVAIKO, E.; EVANS, D. J.; WALKER, F. R. Current state and general perceptions of the use of extended reality (xr) technology at the university of newcastle: Interviews and surveys from staff and students. **SAGE Open**, SAGE Publications Sage CA: Los Angeles, CA, v. 12, n. 2, p. 21582440221093348, 2022.

LAFRUIT, G.; TERATANI, M. **Virtual Reality and Light Field Immersive Video technologies for Real-World applications**: Iet, the institute of engineering and technology. [*S. l.*: *s. n.*], 2022.

LAVALLE, S. M. **Virtual Reality**. Cambridge: Cambridge University Press, 2023. xi–xiii p.

LEITE-JúNIOR, A. J. M.; GOMES, G. A. M.; JUNIOR, N. A. C.; SANTOS, A. D. d.; VIDAL, C. A.; CAVALCANTE-NETO, J. B.; GATTASS, M. System model for shooting training based on interactive video, three-dimensional computer graphics and laser ray capture. In: **2012 14th Symposium on Virtual and Augmented Reality**. [*S. l.*: *s. n.*], 2012. p. 254–260.

LIKERT, R. A technique for the measurement of attitudes. **Archives of psychology**, 1932.

MARIN-VEGA, H.; ALOR-HERNANDEZ, G.; COLOMBO-MENDOZA, L. O.; BUSTOS-LOPEZ, M.; ZATARAIN-CABADA, R. Zeusar: a process and an architecture to automate the development of augmented reality serious games. **Multimedia Tools and Applications**, Springer, v. 81, n. 2, p. 2901–2935, 2022.

MATTSSON, M. Object-oriented frameworks - a survey of methodological issues. 10 1998.

Meta2024 META. 2024. Disponível em: https://www.meta.com/quest/products/quest-2/.

MICROSOFT. **Delegates (C Programming Guide)**. 2022. https://learn.microsoft.com/en-us/dotnet/csharp/programming-guide/delegates/. Acessado: 26/06/2024.

MONTEIRO, D.; LIANG, H.-N.; WANG, J.; CHEN, H.; BAGHAEI, N. An in-depth exploration of the effect of 2d/3d views and controller types on first person shooter games in virtual reality. In: **2020 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)**. [*S. l.*: *s. n.*], 2020. p. 713–724.

RENZ, A.; HILBIG, R. Prerequisites for artificial intelligence in further education: Identification of drivers, barriers, and business models of educational technology companies. **International Journal of Educational Technology in Higher Education**, SpringerOpen, v. 17, n. 1, p. 1–21, 2020.

ROBSON, C. **Real world research**: A resource for social scientists and practitioner-researchers. [*S. l.*]: Wiley-Blackwell, 2002.

SCHMIDT, D. C.; GOKHALE, A.; NATARAJAN, B. Leveraging application frameworks: why frameworks are important and how to apply them effectively. **Queue**, ACM New York, NY, USA, v. 2, n. 5, p. 66–75, 2004.

SHAW, M.; GARLAN, D. **Software architecture**: perspectives on an emerging discipline. [*S. l.*]: Prentice-Hall, Inc., 1996.

SOMMERVILLE, I. **Engenharia de Software.** [*S. l.*]: Pearson Education–BR, 2011.

SOUZA, L. K. d. Recomendações para a realização de grupos focais na pesquisa qualitativa. **Psi UNISC. Santa Cruz do Sul**, v. 4ª, p. 52–66, 2020.

Steam2024 STEAM. 2024. Disponível em: https://store.steampowered.com/sale/BestOf2023.

WANG, A. I.; NORDMARK, N. Software architectures and the creative processes in game development. In: CHORIANOPOULOS, K.; DIVITINI, M.; HAUGE, J. B.; JACCHERI, L.; MALAKA, R. (Ed.). **Entertainment Computing - ICEC 2015**. Cham: Springer International Publishing, 2015. p. 272–285. ISBN 978-3-319-24589-8.

WILDEMUTH, B. M. **Applications of social research methods to questions in information and library science**. [*S. l.*]: Libraries Unlimited, 2009.

WOLF, M. J. Battlezone and the origins of first-person shooting games. **Guns, Grenades, and Grunts**, p. 25, 2012.

YI, J. S. Qndreview: read 100 chi papers in 7 hours. In: **CHI '14 Extended Abstracts on Human Factors in Computing Systems**. New York, NY, USA: Association for Computing Machinery, 2014. (CHI EA '14), p. 805–814. ISBN 9781450324748. Disponível em: https://doi.org/10.1145/2559206.2578884.

APÊNDICE A – SLIDES UTILIZADOS NO WORKSHOP

– Primeira versão: https://1drv.ms/p/s!ArUsBq0NXtKTgaMGTljr3BLvaUSjfQ?e=LxcVjG

– Versão final: https://1drv.ms/p/s!ArUsBq0NXtKTgaMIxfBxdJdAegJKEg?e=UtHVLy

APÊNDICE B – REPOSITÓRIO DO PACOTE PARA UNITY3D

https://github.com/icaroslb/rv_tactical_shooter