INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Jogo Sério Inclusivo para Cegos

LUÍS CARLOS TEIXEIRA COELHO
Junho de 2021

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

Jogo Sério Inclusivo para Cegos

Luís Coelho

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas Gráficos e Multimédia

Orientadora: Prof.ª Dr.ª Paula Escudeiro

Júri:
Presidente:
Joaquim dos Santos

Arguente:
Bruno Galasso

Porto, 30 de junho de 2021

Dedicatória

Dedicado a todos os que fizeram parte
desta experiência de luta constante.

Luís Coelho

Resumo

O presente relatório descreve o processo de investigação, desenvolvimento e teste de um jogo sério para cegos, no âmbito da unidade curricular DEI - Tese / Dissertação /Estágio (TMDEI) do segundo ano do mestrado em Engenharia Informática, com a área de especialização em Sistemas, Gráficos e Multimédia do Instituto Superior de Engenharia do Porto (ISEP).

São descritos os diferentes passos realizados para o desenvolvimento do projeto, nomeadamente a análise do problema, a definição dos objetivos e da abordagem a seguir, a investigação de lacunas e problemas no desenvolvimento de jogos acessíveis, e o desenho e implementação da solução.

Todo o processo é acompanhado por documentação e literatura que suporta as tomadas de decisão realizadas. Verificou-se no autor um crescimento do conhecimento relativo ao tema e às funcionalidades de acessibilidade, sendo complementado com o alcance dos objetivos definidos. Houve também uma elevada maturação ao nível das competências técnicas, muito devido aos desafios que apareceram ao longo do caminho e que obrigaram à aplicação de novas soluções. Constata-se assim a importância deste projeto não só para a sociedade como também no desenvolvimento do autor.

Palavras-chave: Jogo Sério, Inclusão, Acessibilidade, Cegos

Abstract

This report describes the process of research, development and testing of a serious game for the blind, within the scope of the DEI - Thesis / Dissertation / Internship (TMDEI) course of the second year of the Master's in Computer Engineering, within the specialization area in Systems, Graphics and Multimedia of the Superior Institute of Engineering of Porto (ISEP).

The different steps taken for the development of the project are here described, namely the analysis of the problem, the definition of objectives and the approach to be followed, the investigation of gaps and problems in the development of accessible games, and the design and implementation of the solution.

The entire process is accompanied by documentation and literature that supports the decision-making carried out. It was verified, in the author, a growth of knowledge related to the theme and the accessibility features, being complemented with the achievement of the defined objectives. There was also a high maturation in terms of technical skills, largely due to the challenges that appeared along the way, and that forced the application of new solutions. This shows the importance of this project not only for society, but also for the author's development.

Agradecimentos

Agradeço a todos aqueles que de uma forma ou de outra me apoiaram na realização deste projeto e do mestrado durante estes últimos 2 anos. Nomeadamente à minha namorada, pelo apoio, compreensão e ajuda durante as dificuldades que presenciei, à minha família que sempre me encorajou a atingir os meus objetivos e aos meus amigos e colegas de curso pelas experiências vividas ao longo destes 2 anos de aventura.

Por último, gostaria de agradecer à professora Paula Escudeiro, por acreditar, aceitar e apoiar o projeto apresentado, sendo mais uma das vozes que procura a integração desta comunidade no fantástico mundo dos jogos digitais.

Luís Coelho

Conteúdo

Lista de Figuras

Lista de Tabelas

Lista de Acrónimos

AHP	Analytic Hierarchy Process.
API's	Application Programming Interfaces.
CAMUL	Conceção e Autoria Multimédia.
COVID-19	SARS-CoV-2.
DAW	Digital Audio Workstation.
DSO	Disk Operating System.
DSRM	Design Science Research Methology.
FFE	Fuzzy Front End.
GASIG	Game Accessibility Special Interest Group.
IC	Índice de Consistência.
IGDA	International Game Developers Association.
IR	Índice Aleatório.
ISEP	Instituto Superior de Engenharia do Porto.
JAWS	Job Access with Speech.
LMMS	Linux MultiMedia Studio.
MIDI	Musical Instrument Digital Interface.
NPD	New Product Development.
NVDA	Non-Visual Desktop Access.
OMS	Organização Mundial de Saúde.
QDF	Quality Function Deployment.
QEF	Quantitative Evaluation Framework.
RC	Razão de Consistência.
SUS	System Usability Scale.
TMDEI	DEI - Tese / Dissertação / Estágio.
VST	Virtual Studio Technology.

Capítulo 1

Introdução

Esta dissertação encontra-se inserida na unidade curricular DEI - Tese / Dissertação / Estágio (TMDEI) do segundo ano do mestrado em Engenharia Informática, com a área de especialização em Sistemas, Gráficos e Multimédia do Instituto Superior de Engenharia do Porto (ISEP). São aqui documentados todos os processos, decisões e implementações realizados pelo autor, no âmbito do seu projeto.

Neste capítulo é realizado um enquadramento e contextualização do tema na área de jogos digitais para cegos, relacionado-o com o panorama que se vive atualmente nesta indústria. De seguida, é feita a descrição do problema e os objetivos a serem alcançados. Também são expostas as motivações do autor para o projeto, assim como a abordagem que teve para o seu desenvolvimento. Por fim, são descritos não só o planeamento delineado para a conclusão do projeto, como também a estrutura do presente documento.

1.1 Enquadramento / Contexto

Sendo o autor do documento um adepto da área gráfica e multimédia da engenharia informática, mais concretamente, da área da indústria de jogos digitais, faz todo o sentido o desenvolvimento deste projeto para dissertação de mestrado. Após a unidade curricular de Conceção e Autoria Multimédia (CAMUL), do segundo semestre de primeiro ano do mestrado, o autor ficou sensibilizado para a criação de tecnologias que possam abranger e ajudar um grande espetro de pessoas, que muitas vezes não tême novas tecnologias e ferramentas adaptadas às suas condições e problemas.

Nesta unidade curricular, a turma foi dividida em duas equipas, e o objetivo proposto foi a criação de uma aplicação que permitisse a transformação não só de texto escrito, como também de voz, em língua gestual. Foi desta forma que todo um novo mundo de desenvolvimento multimédia se abriu para o autor, o mundo da acessibilidade nas novas tecnologias, mais concretamente nos jogos digitais.

De acordo com o site da **PORDATA**, no último censo de deficiências visuais efetuado, existem 163.569 indivíduos com este tipo de problema, o que à data representava aproximadamente 1,58% da população do portuguesa (*População residente com deficiência segundo os Censos: total e por tipo de deficiência (2001)* s.d.). Mundialmente, com 36 milhões de cegos e 217 milhões com deficiências visuais moderadas a severas, o que representa uma percentagem de 0,49% e 2,95%, respetivamente (Ackland, Resnikoff e Bourne 2017).

Atualmente, devido à situação de confinamento provocada pela pandemia da SARS-CoV-2 (COVID-19), as pessoas passam mais tempo em casa, o que leva a um grande aumento na utilização de jogos digitais (Marketeer 2020). Surge assim um aumento daqueles que

procuram novas formas de entretenimento, agora que se encontram confinados nas suas habitações.

Para além do fator de diversão, os jogos servem como fuga da realidade, alívio de stress e estimulação mental, tendo efeitos muito benéficos na saúde mental, principalmente no contexto de confinamento (*How Gaming is Good for Your Mental Health During Lockdown* 2020).

Tendo em conta os dados apresentados, é assim imprescindível a implementação de técnicas que permitam integrar este grupo da população na indústria de jogos, e que esta se adapte às necessidades destes indivíduos. Revela-se especialmente importante agora, neste período em que as outras formas de entretenimento e de convívio social foram afetadas.

1.2 Descrição do Problema

A indústria de jogos digitais é a maior indústria de entretenimento a nível mundial, sendo que em 2019, foram gastos mais de 145 mil milhões de dólares globalmente (Richter 2020). E, apesar disso, muitos dos jogos que são lançados não possuem ferramentas nem tecnologias acessíveis que permitam que estes possam ser jogados, com facilidade, por indivíduos que possuem necessidades especiais (J. R. Aguado-Delgado et al. 2018).

Desta forma, são esquecidos e deixados de lado todos aqueles com uma panóplia de dificuldades inerentes. É assim importante, que os diversos estúdios de jogos e grandes desenvolvedoras possam aplicar as tecnologias e técnicas existentes no meio, de forma a facilitar a integração de todos, numa área que está em grande crescimento (Wijman 2020).

Este projeto pretende assim a criação de um jogo que possa ser facilmente jogado e completado por indivíduos cegos, sem qualquer tipo de constrangimento. Para além disso, procura-se também a investigação, estudo e seleção das melhores técnicas e tecnologias utilizadas no auxílio de cegos, de forma a que possam ser aplicadas posteriormente, no âmbito do design e implementação.

1.2.1 Objetivos

Tal como referido na secção 1.2, este projeto pretende o estudo e desenvolvimento de um jogo sério, que possa ser jogado por cegos, sem qualquer dificuldade. Assim, e de acordo com este preceito inicial, foram definidos os seguintes objetivos:

- **Identificação e Estudo de Técnicas Relevantes**: Identificação, estudo e análise de técnicas e tecnologias relevantes, no desenvolvimento de jogos para cegos e que conseguem melhorar a experiência de jogo;
- **Identificação e Análise dos problemas dos Utilizadores Principais**: Identificação e estudo dos principais problemas dos cegos, de forma a conseguir adaptar o projeto às suas necessidades;
- **Identificação de Lacunas existentes no Mercado**: Identificação e análise das lacunas de títulos existentes no mercado;
- **Estudo de Casos de Sucesso**: Estudo de casos de sucesso existentes na indústria de jogos, de forma a perceber quais as técnicas e tecnologias aplicadas;

- **Construção do Jogo Adaptado**: Desenho e implementação do jogo sério de acordo com as principais características analisadas, permitindo que todas as ações de jogo possam ser executadas por cegos;
- **Experimentação em Cenários Reais**: Experimentação e teste do jogo, de forma a poder simular da forma mais correta possível a sua implementação em contextos realistas.

1.2.2 Abordagem

A metodologia escolhida pelo autor para o desenvolvimento do projeto é a *Design Science Research Methology (DSRM)*, desenvolvida com o foco em sistemas informáticos. Encontra-se dividida em 6 atividades distintas: a identificação do problema e motivação, a definição dos objetivos, o desenho e desenvolvimento da solução, a demonstração, a avaliação e por fim a comunicação (Peffers et al. 2007).

Após a justificação da motivação do autor, é realizada uma descrição do problema. Esta fase corresponde à primeira atividade da metodologia, a **Identificação do Problema e Motivação** (Peffers et al. 2007).

Depois são definidos os objetivos necessários atingir para o sucesso do projeto, o que corresponde à atividade da **Definição dos Objetivos** e que procura aferir quais são possíveis de medir e de completar (Peffers et al. 2007).

A pesquisa realizada pelo autor pode ser caracterizada por diferentes camadas de investigação, também conhecidas por **Research Onion** (Saunders e Tosey 2013). Destas camadas, salientam-se as 3 mais relevantes, e que ajudam a definir o trabalho de investigação realizado:

- Metodologia - Pretende definir que tipo de métodos são utilizados: métodos quantitativos, qualitativos ou uma mistura de ambos (Saunders e Tosey 2013). O autor do documento utiliza métodos qualitativos, através da análise de entrevistas e opiniões obtidas da investigação e pesquisa de literatura (Tengli 2020).
- Estratégia - Refere-se às práticas que são usadas para desenvolver e mostrar resultados do trabalho feito (Saunders e Tosey 2013). Aqui o autor do documento prioriza pesquisa-ação, pois "[...] procura resolver desafios práticos da vida real, encontrando as soluções para os problemas através da participação de membros focados no estudo."[1] (Tengli 2020).
- Horizonte Temporal - Tal como definido por (Saunders e Tosey 2013), procura definir o período temporal no qual o investigador realiza a pesquisa, podendo ser longitudinal caso seja feito por um período de tempo elevado ou transversal caso procure resolver o problema num determinado período. A abordagem do autor segue o horizonte temporal longitudinal, já que vai procurar resolver o problema existente ao longo de um período de tempo, utilizando *feedback*[2] e análise da população em estudo, algo característico da estratégia de pesquisa-ação supramencionada (Saunders e Tosey 2013).

Depois, é realizado o levantamento de requisitos, o desenho da solução e a sua implementação, o que se encontra associado à atividade de **Desenho e Desenvolvimento da Solução**. Esta atividade é suportada por conhecimento teórico, obtido através da análise do estado da arte e de soluções atualmente praticadas (Peffers et al. 2007).

[1] Tradução livre do autor.

[2] *Feedback* é a informação obtida dos utilizadores, de acordo com a sua experiência (Haije 2021).

Relativamente ao processo de implementação, procurou-se aplicar metodologias ágeis. A utilização do método iterativo e incremental permite a divisão do trabalho em diferentes incrementos sequenciais, de versões estáveis do projeto (Larman 2012).

Posteriormente, é realizada a atividade de **Demonstração**, onde o autor demonstra o objeto criado aos utilizadores visados, de forma a simular as condições necessários para a resolução dos problemas (Peffers et al. 2007). Esta atividade, encontra-se dividida em 2 fases distintas:

- Na primeira fase, a versão produzida é testada pelo autor do projeto de forma a aferir os principais problemas funcionais e procurar *bugs*[3] no *software*[4]. Com o desenvolvimento, as versões produzidas passam também a ser testadas por participantes, de forma a receber *feedback* sobre a jogabilidade, interface, experiência de utilizador e outros problemas.
- Na segunda fase, as versões geradas são testadas pelos utilizadores visados no projeto - os cegos - ou participantes vendados, sendo assim possível usar a sua experiência para avaliar os fatores de inclusão criados e as técnicas aplicadas, assim como receber *feedback* de melhorias a realizar.

Depois, vem a fase de avaliação, onde o autor mede a forma como a solução implementada corresponde aos objetivos delineados. Isto é realizado através de análise de questionários, aplicados previamente na fase de demonstração, correspondendo este momento à atividade da **Avaliação** (Peffers et al. 2007).

Por fim, temos o desenvolvimento deste documento, onde é documentada toda a pesquisa realizada, artefactos construídos e desenhos realizados, o que se relaciona com a atividade de **Comunicação** (Peffers et al. 2007).

Aliado a estas abordagens ágeis, é utilizada uma aplicação de gestão de tarefas, de forma a documentar e rastear todas as tarefas realizadas, em resolução e por realizar. Usando esta abordagem, é também simples e rápido documentar todo o *feedback* obtido ao longo dos incrementos.

1.2.3 Questões de Investigação

As hipóteses de investigação definidas encontram-se intimamente ligadas aos objetivos definidos na secção 1.2.1. Tendo isto em conta, as hipóteses são as seguintes:

- A solução desenvolvida encontra-se totalmente adaptada às características dos cegos, permitindo que estes joguem sem qualquer obstáculo.
- Foram encontradas as principais lacunas no desenvolvimento de jogos para cegos, assim como as principais técnicas que colmatam os problemas destes.
- Os resultados dos cenários de experimentação obtiveram níveis excelentes em termos de usabilidade.

A partir destas 3 questões de investigação é possível perceber se o produto desenvolvido se encontra adaptado às condições dos cegos, se as principais lacunas no desenvolvimento foram colmatadas e se foi obtido obtido um nível de excelência na usabilidade. Partindo da análise e resposta de cada uma é possível aferir o nível de sucesso do projeto desenvolvido.

[3] *Bugs* são falhas, causadas por erros de programação e que levam a resultados inesperados (*What is a Software Bug?* 2017).

[4] *Software* são as instruções que controlam o computador e os programas (*Software* s.d.).

1.2.4 Planeamento de Trabalho

Este projeto foi dividido em 4 fases distintas e complementares: análise do estado da arte e desafios, desenho da solução, implementação e testes. Estas etapas foram realizadas paralelamente à escrita do presente documento.

Na primeira fase, **Análise do Estado da Arte e Desafios**, é feita uma investigação do estado da arte, onde são identificadas as principais tecnologias e técnicas utilizadas para a integração de cegos em jogos digitais. É também nesta fase, que são constatadas as suas principais dificuldades quando pretendem interagir com este tipo de produtos. Esta fase será desenvolvida entre os dias 4 de novembro de 2020 e 20 de fevereiro de 2021.

Na segunda fase, **Desenho da Solução**, são identificadas, avaliadas e selecionadas diferentes soluções a aplicar no projeto. Aqui, serão definidos os requisitos necessários para o cumprimento dos objetivos delineados e serão escolhidas as soluções que melhor se enquadram com o resultado pretendido. Será desenvolvida entre os dias 15 de fevereiro de 2021 e 15 de março de 2021.

Na terceira fase, **Implementação**, serão implementadas as soluções definidas durante a fase de desenho. Será desenvolvida entre os dias 15 de março de 2021 e 5 de junho de 2021.

Na quarta fase, **Testes**, que consiste na realização de testes da usabilidade da solução implementada, de forma a perceber como esta responde às necessidades dos utilizadores finais. É uma fase fundamental do projeto, na qual se pretende perceber se todos os mecanismos e técnicas aplicados conseguiram cumprir os objetivos que o projeto pretender solucionar, e ao mesmo tempo corrigir os problemas detetados pelos utilizadores. Será desenvolvida entre os dias 10 de maio de 2021 e 20 de junho de 2021.

Na figura 1.1, podemos ver sumariada a descrição feita anteriormente, através de um diagrama de *Gantt*.

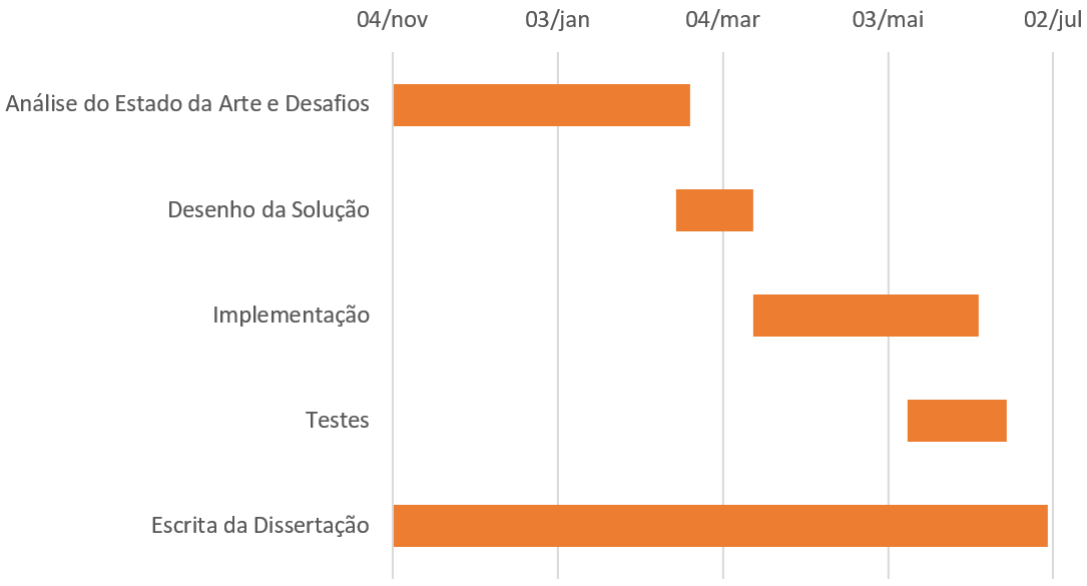


Figura 1.1: Diagrama de Gantt para Planeamento de Trabalho.

1.3 Estrutura do Documento

Este documento é composto por 8 capítulos distintos. Na Introdução é feita a contextualização da dissertação, seguida da descrição do problema, dos objetivos e da abordagem efetuada. Finaliza com a apresentação da estrutura do documento.

No segundo capítulo é apresentado um Estado da Arte, no qual é analisada a condição atual da indústria de jogos digitais para cegos, trabalhos relacionados e tecnologias existentes.

No terceiro capítulo, dedicado à Análise de Requisitos, é feita a identificação e classificação dos requisitos necessários para implementar o projeto.

No quarto capítulo, dedicado à Análise de Valor, é feita a identificação, estudo e seleção de oportunidades. É aqui definida a proposta de valor e são priorizados os aspetos mais relevantes de negócio.

No quinto capítulo, relativo ao Desenho da Solução, é realizada a concepção de todos os elementos arquiteturais e de jogo, e respetivas comparações com alternativas.

No sexto capítulo, a Implementação da Solução, são apresentados os detalhes relativos à implementação da solução desenhada.

No sétimo capítulo, referente à Avaliação da Solução, são definidas as diferentes grandezas que serão utilizadas para avaliar o projeto, assim como as hipóteses a serem testadas e respetiva metodologia de avaliação.

Por fim, no oitavo e último capítulo, são apresentadas as Conclusões, sendo feita uma apreciação global dos objetivos concretizados e sugestões para futuros desenvolvimentos, a serem realizados.

Capítulo 2

Estado da Arte

Este capítulo foca na contextualização, trabalhos relacionados e tecnologias existentes. Inicialmente é feita a contextualização da acessibilidade no mundo dos jogos digitais, o género de jogos existentes, os diferentes tipos de deficiências visuais e como a indústria se adaptou ao longo dos anos. Nos trabalhos relacionados são referidos jogos inovadores na acessibilidade para cegos. Por fim, são destacadas e analisadas as principais tecnologias utilizadas neste tipo de jogos e as alternativas que vão ser usadas no desenvolvimento do projeto.

2.1 Contextualização

Atualmente e de acordo com Porter (Porter e Kientz 2013), a diferença existente entre a área de interação humano-computador e a indústria de jogos digitais levou à baixa adoção das normas de acessibilidade existentes. Esta realidade é ainda mais percetível quando a indústria de jogos digitais é comparada com outras área das tecnologias de informação, onde houve uma adoção generalizada.

Porter explica que mesmo com a expansão dos jogos digitais para novas áreas, como a saúde e o ensino, os jogos comerciais continuam com grande carências relativamente à acessibilidade. Esta situação leva assim à exclusão de muitos indivíduos, de algo com cada vez mais importância na sociedade (Porter e Kientz 2013).

Como contratendência, o desejo por progressos na área da acessibilidade tem aumentando na sociedade, quer seja através da criação de leis ou através dos meios de comunicação social (J. Aguado-Delgado et al. 2020).

Porter (Porter e Kientz 2013) destaca ainda que as principais forças para a melhoria da acessibilidade na indústria de jogos são:

- **Pesquisa Académica**: Aqui, o principal tipo de trabalho encontra-se relacionado à criação de jogos com diferentes estratégia de acessibilidade (Porter e Kientz 2013).
- **Trabalho das Comunidades**: Já nos trabalhos das comunidades salienta-se os esforços realizados por indivíduos singulares e a formação da Game Accessibility Special Interest Group (GASIG) pela International Game Developers Association (IGDA). Esta e outras organizações semelhantes procuram a introdução de diferentes tipos de estratégias no combate aos desafios da acessibilidade (Porter e Kientz 2013). De acordo com Bierre (Bierre et al. 2005), de forma a tornar os jogos universalmente acessíveis, a GASIG definiu os seguintes objetivos:
  - Criação de comunidades com interesse no desenvolvimento da acessibilidade.

- Levantamento dos problemas e necessidades existentes nos diferentes grupos.
- Procura e investigação de novos métodos de acessibilidade.
- Revisão das necessidades ao ao longo da evolução da indústria dos jogos.
- Desenvolvimento de métodos de acessibilidade e a sua transmissão para a comunidade de desenvolvedores.

Desta forma e para Porter, a dificuldade da acessibilidade no mundo dos jogos digitais encontra-se associada à comunicação das melhores práticas para a indústria, sendo fundamental que estas respeitem as normas já praticadas e sejam claras e percetíveis (Porter e Kientz 2013).

2.1.1 Tipos de Jogos

Tal como referido por Aguardo-Delgado (J. Aguado-Delgado et al. 2020), os jogos são "[...] programas desenhados para entretenimento e diversão e que podem ser utilizados em diversas plataformas como consolas, computadores e telemóveis."[1]. A sua evolução encontra-se intimamente ligada ao aparecimento de novas tecnologias, ganhando cada vez mais importância como produtos multifacetados (J. Aguado-Delgado et al. 2020).

Yuan (Yuan 2009) explica que ao longo das últimas décadas houve um grande desenvolvimento das tecnologias que suportam os jogos digitais e que permitiram a resolução de muitas barreiras existentes. No entanto, também levou a uma maior complexidade nos controlos de jogo, o que por si só cria novas dificuldades de acessibilidade (Yuan 2009).

De forma a entender a ligação entre os jogos digitais e a forma como estes são adaptados a cegos é fundamental compreender a sua classificação. Estes são separados de acordo com características intrínsecas, sendo possível salientar os seguintes tipos (Yuan 2009):

- **Tiro em Primeira Pessoa**: Jogos de ação com a câmara em primeira pessoa e que envolve o ataque a inimigos com diferentes tipos de armas.
- **Estratégia**: Jogos de combate através da gestão de recursos, unidades e de estratégia. Este tipo de jogos podem ser jogados em tempo real ou através de turnos.
- **Desporto**: Simulação de diferentes tipos de desporto, como o futebol ou a natação.
- **Narrativa**: Criação e jogo de uma personagem que possui diferentes caminhos de evolução e que dependem das escolhas do jogador.
- **Puzzles**: Resolução de problemas através de lógica, estratégia e reconhecimento de padrões.
- **Corrida**: Simulação de corridas com diferentes tipos de veículos.
- **Ritmo**: Simulação de dança ou toque de instrumentos.
- **Aventura**: Investigação, exploração e interação com personagens, focando o jogo na história existente.

[1] Tradução livre do autor.

2.1.2 Classificação de Deficiências Visuais

Tal como referido na secção 1.1 e de acordo com o site da **PORDATA**, aproximadamente 1,58% da população do portuguesa é cega ou possui visão parcial (*População residente com deficiência segundo os Censos: total e por tipo de deficiência (2001)* s.d.).

Os indivíduos com deficiências visuais têm diferentes classificações de acordo com o grau de perda de visão. Segundo a Organização Mundial de Saúde (OMS), um indivíduo é considerado cego caso possua 5% ou menos de acuidade visual no melhor olho e com a melhor correção possível. Caso estes valores variem entre 5% e 30% de acuidade visual a deficiência é considerada visão parcial (Thylefors et al. 1995). As principais causa de deficiências visuais encontram-se relacionadas a doenças e problemas não tratados (*Vision impairment and blindness* s.d.), como por exemplo:

- Erros Refrativos.
- Cataratas.
- Degeneração macular relacionada ao envelhecimento.
- Glaucoma.
- Retinopatia diabética.

A prevalências destas doenças e problemas nas causas de deficiências visuais varia de acordo com zona no mundo e os países analisados, sendo muito mais comum encontrar deficiências visuais relacionadas a cataratas em países com baixos rendimentos (*Vision impairment and blindness* s.d.).

Outro fator importante nas deficiências visuais é a idade dos indivíduos, sendo que mundialmente 58% dos cegos têm mais que 60 anos idade (Thylefors et al. 1995). Isto aliado ao envelhecimento da população leva a que seja expectável o aumento do número de deficiências visuais (*Vision impairment and blindness* s.d.).

2.1.3 Lacunas e Problemas

De acordo com as entrevistas realizadas por Porter a indivíduos cegos (Porter e Kientz 2013), muitos dos problemas de acessibilidade devem-se à "[...] inabilidade para interagir com jogos usando soluções de tecnologia assistiva."[2]. No contexto de jogos digitais estas soluções tornavam-se desnecessárias, resultado da incapacidade na interpretação de informação. O autor notou ainda que muitas destas situações levam a pedidos de ajuda a terceiros, o que os participantes descreviam como uma "[...] diminuição do propósito de jogar o jogo por inteiro."[3] (Porter e Kientz 2013).

Segundo Bierre (Bierre et al. 2005), jogos não acessíveis levam a que jogadores cegos tenham como principais problemas: a resolução de tarefas e puzzles, a dificuldade em entender como os jogos são jogados, a incapacidade de interagir com equipamentos adaptados e a derrota contínua. Estes problemas, encontram-se associados a diversos motivos (Bierre et al. 2005):

- Pistas e informações que são fornecidas através de texto ou de imagens.
- Inexistência de tutoriais com explicação do funcionamento do jogo e das suas diferentes mecânicas.

[2] Tradução livre do autor.
[3] Tradução livre do autor

- Fraca documentação ou formatação com um elevado nível de complexidade.
- Jogos com suporte inexistente ou reduzido para equipamentos de acessibilidade.
- Falta de indicadores das diferentes situações de jogo. Como por exemplo, situações perigosas ou ações de valor acrescido.

De forma a entender as práticas de acessibilidade utilizadas, Porter realizou entrevistas a indivíduos de diversas posições da indústria de desenvolvimento de jogos. Os resultados permitem aferir como decisões no processo de desenvolvimento influenciam as questões de acessibilidade (Porter e Kientz 2013):

- Certas funcionalidades da acessibilidade têm maior probabilidade de serem desenvolvidas de acordo com a sua facilidade de implementação.
- Questões de acessibilidade apenas são mencionadas quando alguém as torna uma prioridade. Porter refere que segundo as entrevistas, os colaboradores com deficiências possibilitam a identificação de potenciais problemas de acessibilidade cedo.
- Pressão para adesão a padrões de acessibilidade vinda de executivos possui uma grande influência no processo de desenvolvimento.
- Utilização de *software* intermédio que permita tratar das funções de baixo nível e da implementação de funcionalidades de acessibilidade. Quando estas funções são tratadas por um *software* intermédio, as bases da acessibilidade já se encontram implementadas, sendo possível poupar no tempo, nos recursos e no dinheiro gasto em múltiplas implementações.

Porter, conclui com o consenso geral dos participantes relativamente à utilização de *software* intermédio com funcionalidades de acessibilidade, sendo possível acelerar a adoção dos padrões de acessibilidade definidos para a indústria (Porter e Kientz 2013).

2.1.4 Desenho de Jogos para Cegos

De forma a perceber como as dificuldades dos cegos limitam as suas capacidades de jogo é necessário a definição de um modelo, que represente a interação destes com o sistema (Yuan 2009). Este modelo divide-se em 3 fases:

- **Receção de Estímulos**: Um jogo envia estímulos para um jogador de 3 formas distintas: visual, auditiva e tátil. Estes estímulos ainda podem ser divididos em 2 tipos (Yuan 2009):
  - **Estímulos Principais**: Estes estímulos têm de ser percecionados pelos jogadores, pois são fundamentais para que estes consigam jogar. Por exemplo, num jogo de futebol de consola é fundamental que o jogador consiga receber estímulos visuais, já que de outra forma não seria possível jogar.
  - **Estímulos Secundários**: Estímulos secundários servem como suporte aos estímulos principais, não sendo fundamentais para o jogo. Aqui, apesar de o jogador perder parte da experiência de jogo, continua a poder jogá-lo.
- **Determinação de Resposta**: Depois da receção dos diferentes estímulos, o jogador deve determinar qual é a melhor resposta, de acordo com as ações disponíveis (Yuan 2009).

- **Resposta**: Depois da decisão, o jogador deve responder fisicamente através da equipamentos de entrada e que comunicam com o sistema de jogo (Yuan 2009).

Os passos anteriormente descritos são repetidos até à vitória ou derrota do jogador. Esta interação, entre o jogador e o sistema encontra-se representada na figura 2.1:

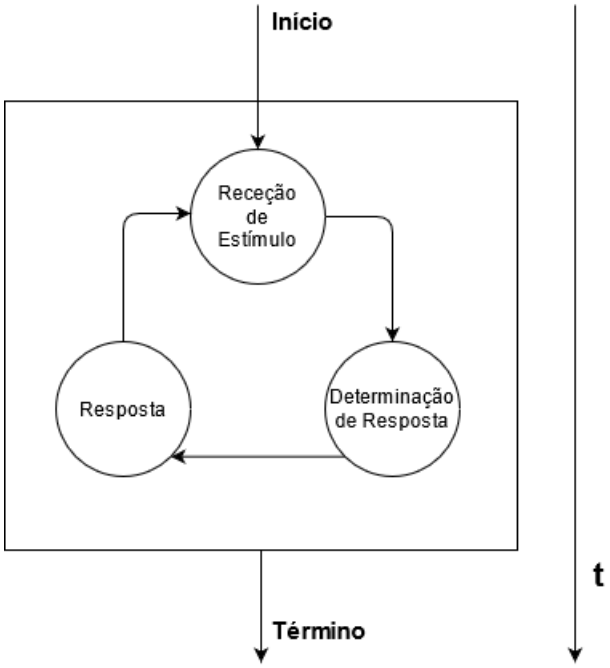


Figura 2.1: Máquina de estados finitos para o modelo de interação (Adaptado a partir de (Yuan 2009)).

Segundo Yuan (Yuan 2009) as principais estratégias de acessibilidade passam pela "[...] substituição de *feedback* visual por outra formas de *feedback* percecionadas."[4]. As mais relevantes são as seguintes:

- **Discurso**: Fala dos textos e dos menus jogo, podendo ser obtido através de leitores de ecrã ou de vozes customizadas.
- **Dicas de Áudio**: Uso de sons da vida real para proporcionar informação extra ao jogador ou dicas de como resolver os desafios.
- **Sonificação**: Utilização das características do áudio como a frequência ou o volume para passar informação ao jogador. A sonificação pode ser classificada em 3 tipos diferentes:
  - **Ícones Auditivos**: Efeitos de som que funcionam como indicadores de objetos ou de ações.
  - **Earcons**: Uso de tons do som para indicação de objetos ou de ações.
  - **Sonar**: Este mecanismo envia informação espacial da localização dos objetos.

Outras estratégias identificadas passam pela modificação dos visuais de jogo para aumentar o contraste, os esquemas de cor e o tamanho das fontes. Com isto é possível a criação de mecanismos de acessibilidade para indivíduos com visão reduzida (Yuan 2009).

[4] Tradução livre do autor

2.2 Trabalhos Relacionados

Nesta secção do estado da arte são descritos alguns dos trabalhos relacionados com o projeto. Estes foram escolhidos pela sua inovação na acessibilidade ou pelo seu foco na comunidade de cegos.

A sua pesquisa foi feita através de revisão de literatura onde são referidos como marcos de inovação na acessibilidade ou no desenvolvimento de jogos para cegos. Uma das bases de dados utilizadas na análise e pesquisa foi o *AudioGames.net* (*AudioGames* s.d.), dado a sua qualidade de repositório *online* de jogos de áudio.

Inicialmente são apresentados cada um dos trabalhos, com um resumo das suas principais vantagens. Por fim, realiza-se uma comparação entre eles, de forma a poder analisar as suas características diferenciadoras.

2.2.1 Resumo

De seguida são apresentadas alguns dos jogos com características semelhantes ao projeto a ser desenvolvido, quer através da semelhança de métodos como pela inovação e aplicação de estratégias de acessibilidade:

- **A Blind Legend**: Projeto colaborativo entre a *DOWiNO* e a *France Culture* de um jogo de ação e aventura para dispositivos móveis. Este projeto apenas faz uso de áudio, através da utilização de uma tecnologia inovadora, o som binaural, para guiar o jogador num mundo em 3 dimensões (*A Blind Legend* s.d.).
- **Terraformers**: É um jogo de aventura em primeira pessoa, tendo sido desenhado a pensar em cegos. Pode ser jogado com os gráficos ativados ou desativados e inova através da utilização de um sistema de sonar para indicação das distâncias de objetos ou de outras informações (*Terraformers* s.d.).
- **Shades of Doom**: Jogo de tiros em primeira pessoa desenvolvido pela *GMA Games*, onde são utilizadas pistas de áudio para encontrar inimigos e percorrer os diferentes níveis. É focado na utilização do som através de múltiplas camadas e dimensões e com o seu sistema de fala não é necessário qualquer leitor de ecrã (*Shades of Doom* s.d.).
- **Mortal Kombat 11**: Jogo de luta desenvolvido pela *NetherRealm Studios* e parte da série de *Mortal Kombat*. Tal como em outros jogos da série foca-se na luta entre diferentes personagens e é conhecido pela sua extrema violência e cenas explícitas (*Mortal Kombat11* s.d.). É um jogo com elevado detalhe no sistema de áudio, o que proporciona uma ótima experiência para jogadores cegos.
- **The Last of Us 2**: The Last of Us 2 é um jogo de ação e aventura desenvolvido pela *Naughty Dog* em exclusivo para a *PlayStation*. Continua a história do primeiro jogo da série e é jogado numa perspetiva de terceira pessoa e no papel de 2 protagonistas. Foi desenvolvido com o objetivo de assegurar que o maior número de fãs pudessem experienciar o jogo, sendo desenvolvidas diversas estratégia de acessibilidade como: conversão de texto em fala, assistência à navegação e avisos auditivos (*The Last of Us 2* s.d.).
- **The Division 2**: Este é um jogo narrativo de tiros desenvolvido pela *Massive Entertainment* com campanha, batalhas entre múltiplos jogadores e diferentes modos de

missão. Implementado com diversas funções de acessibilidade como a interpretação e fala de menus de jogo e a assistência de mira (*The Division 2* s.d.).

- **Gears 5**: Desenvolvido pela *The Coalition*, este é um jogo de tiros em terceira pessoa e que continua a história do anterior Gears 4 (*Gears 5* s.d.). Destaca-se pela grande lista de funcionalidades de acessibilidade e pela inclusão, possuindo leitores de ecrã e assistentes de mira (Ho 2019).

2.2.2 Tipos de Solução

Os jogos referidos na secção anterior podem ser englobados em dois conjuntos principais de soluções: Jogos de Áudio e Jogos Core. Estes dois conjuntos fazem parte de um grupo maior de três tipos de soluções, que apesar de abordagens distintas respondem a determinados pontos do problema, sendo importante perceber qual é a utilidade de cada um e quais são as principais vantagens da sua utilização.

Jogos de Áudio

Jogos de áudio são jogos onde o principal tipo de estímulo é auditivo, e são frequentes no âmbito do mercado de jogos para cegos. Podem ser divididos em 3 tipos conceitos (Archambault et al. 2007): Jogos de ritmo, jogos de experimentação musical e jogos que podem ser jogados sem qualquer imagem e com apenas áudio. Aqui, as imagens são substituídos por áudio e os textos e menus por fala, tornando-os acessíveis a cegos (Archambault et al. 2007; White, Fitzpatrick e McAllister 2008).

As vantagens da utilização desta solução são:

- Especialização na acessibilidade para cegos, procurando assim resolver os seu problemas através de soluções adaptadas.
- Desenvolvimento de novas tecnologias e funcionalidade, que podem ser utilizadas como padrões por todas a indústria.

As desvantagens da utilização desta solução são:

- Pequenas equipas de desenvolvimento.
- Pouco interesse na generalidade da comunidade de jogos.
- Mercado pequeno e com poucos lucros.

Neste tipo de solução encontram-se categorizados os seguintes jogos: **A Blind Legend**, **Terraformers** e **Shades of Doom**.

Jogos Core

Este tipos de jogos são focados num segmento de mercado especializado, maturando-se em comunidades especificas de fãs muito envolvidos, competitivos (Kuittinen et al. 2007) e que procuram sensações de adrenalina (IDGA 2008).

As vantagens da utilização desta solução são:

- Grandes equipas de desenvolvimento.
- Base de fãs pré-estabelecida.

- Mercado com lucros elevados e grande investimento.
- As soluções de acessibilidade têm grande impacto na adoção de padrões por parte da indústria.

As desvantagens da utilização desta solução são:

- Foco num público muito específico o leva a que as soluções desenhadas não se foquem na acessibilidade mas apenas no público idealizado.
- Funcionalidades de acessibilidade limitadas.

Aqui, os jogos encaixados são: **Mortal Kombat 11**, **The Last of Us 2**, **The Division 2** e **Gears 5**.

Jogos Casuais

Este tipos de jogos são focados no mercado generalizado de jogadores, procurando assim atingir todos os jogadores independentemente da idade ou de outras características. São também descritos como jogos que envolvem controlos menos complexos e uma jogabilidade mais simples (Kuittinen et al. 2007), proporcionando diversão e relaxamento (IDGA 2008).

As vantagens da utilização desta solução são:

- Baixos custos de desenvolvimento
- Crescente interesse por parte da comunidade de jogos.
- Mercado com lucros elevados.
- Elevado desenvolvimento em soluções de acessibilidade, visto que procuram atingir diferentes populações com diferentes características.

As desvantagens da utilização desta solução são:

- Influência limitada na aplicação de soluções de acessibilidade, devido à elevada oferta na indústria.

2.2.3 Comparação das Soluções

Nesta secção é realizada uma comparação e análise das diferentes soluções apresentadas na secção 2.2.1 através de 3 fatores: o público-alvo, as estratégia utilizadas para transmitir informação ao jogador e as funcionalidades de acessibilidade existentes, de acordo com 9 normas do GASIG.

Público-Alvo

Este parâmetro procura definir para quais comunidades estes jogos focaram o seu desenvolvimento. A tabela 2.1 apresenta esta categorização através de 2 fatores:

- **Comunidade dos Cegos**: Neste grupo estão incluídos todos jogos desenhados e desenvolvidos a pensar na comunidade de cegos. Estes jogos têm as suas características e implementações adaptadas para estas comunidades.
- **Comunidade Geral**: Aqui estão todos os jogos desenvolvidos para o público geral e sem especificação da comunidade. Apesar destes jogos serem desenvolvidos para uma grande audiência, não significa que exista ausência de características de acessibilidade.

Tabela 2.1: Público-Alvo dos jogos.

Jogos	Público-Alvo	
	Cegos	Geral
A Blind Legend	X	
Terraformers	X	X
Shades of Doom	X	
Mortal Kombat 11		X
The Last of Us 2		X
The Division 2		X
Gears 5		X

Com a análise da tabela 2.1 é possível aferir que os primeiros 3 jogos (Jogos de Áudio) tem como público-alvo os cegos, enquanto que os outros jogos (Jogos Comerciais) abrangem toda a comunidade de jogadores. É importante realçar o jogo *Terraformers*, que possui como público alvo os cegos e a comunidade geral. Esta dualidade deve-se à possibilidade de ativação e desativação dos gráficos.

Estratégias

O parâmetro das estratégias foca nas estratégias utilizadas pelos jogos para comunicar informação ao jogador e a forma como a captam. Os fatores utilizados foram os seguintes:

- **Tipo de Jogo**: Desenvolvido na secção 2.1.1 e que representa o género do jogo.
- **Feedback**: Explicado na secção 2.1.4 e que mostra como o jogo transmite a informação necessária para o jogador.
- **Controlos**: O tipo de equipamentos utilizados pelo jogador para interagir e responder às ações do sistema.

Tabela 2.2: Estratégias de acessibilidade existentes nos jogos.

Jogos	Estratégias				
	Tipo de Jogo	Feedback			Controlos
		Discurso	Pistas de Áudio	Sonificação	
A Blind Legend	Ação-Aventura	X	X		Tátil
Terraformers	Tiro 1º Pessoa	X		X	Teclado
Shades of Doom	Tiro 1º Pessoa	X	X	X	Teclado
Mortal Kombat 11	Luta	X	X		Comando
The Last of Us 2	Ação-Aventura	X	X	X	Comando
The Division 2	RPG	X	X		Comando
Gears 5	Tiro 3º Pessoa	X	X		Comando

A partir da análise da tabela 2.2 é possível destacar a panóplia de tipos de jogo e controlos utilizados, o que mostra que existem múltiplas estratégias para a implementação de jogos acessíveis. Relativamente ao *feedback*, é possível realçar os seguintes pontos:

- Uso generalizado de discurso para os diferentes menus de jogo e das pistas de áudio, sendo que o *Terraformers* é o único jogo que não as utiliza.
- Relativamente à sonificação, apenas é utilizada por 3 jogos, o *Terraformers*, o *Shades of Doom* e o *The Last of Us 2*.

Dos diferentes jogos apresentados, o *Shades of Doom* e o *The Last of Us 2* foram os que utilizaram o maior número de estratégias de acessibilidade, apresentando o uso de todos os tipo de *feedback*.

Funcionalidades de Acessibilidade

Para esta análise foram utilizados 9 fatores de acessibilidade, que seguem as normas do GASIG (*SIG top ten* 2019). Como estes fatores incluem principalmente características gráficas não são analisados os jogos de áudio, pois apenas se focam em características de áudio. Os fatores utilizados foram os seguintes:

- **Dificuldade**: Possibilitar ao jogador escolher o nível de dificuldade de jogo.
- **Alto Contraste**: Alto contraste entre o esquema de cores, destacando elementos importantes para o jogo.
- **Textos Legíveis**: Textos com tamanho grande e com formatos fáceis de ler, possibilitando ainda a escolha por parte do jogador.
- **Daltonismo**: Evitar a utilização de cores como informação importante para o jogo.
- **Controlos**: Possibilitar a utilização de diferentes tipos de dispositivos para o controlo do jogo.
- **Gestão de Áudio**: Possibilitar ao jogador ajustar o som individualmente, como os diálogos, a músicas e os efeitos de som.
- **Customização**: Possibilitar ao jogador a modificação manual dos botões e teclas utilizados nos equipamentos que controlam o jogo.
- **Menus Simples**: Menus de jogo com baixo nível de complexidade e que facilitem uma navegação intuitiva e rápida.
- **Informação**: A informação relativa às funcionalidade e sistemas de acessibilidade pode ser facilmente encontrada sem a necessidade de abrir o menu de opções.

Tabela 2.3: Funcionalidades de acessibilidade existentes nos jogos.

Jogos	**Funcionalidades**			
	Dificuldade	**Alto Contraste**	**Textos Legíveis**	**Daltonismo**
Mortal Kombat 11	X			X
The Last of Us 2	X	X	X	X
The Division 2	X	X	X	X
Gears 5	X	X	X	X

Tabela 2.3: Funcionalidades de acessibilidade existentes nos jogos (continuação).

Jogos	Funcionalidades				
	Controlos	Gestão Áudio	Customização	Menus Simples	Informação
Mortal Kombat 11	X	X	X		
The Last of Us 2		X	X	X	X
The Division 2	X	X	X	X	X
Gears 5	X	X	X	X	

Com a análise da tabela 2.3 é possível salientar os seguintes pontos:

- Em termos de múltiplos controlos, *The Last of Us 2* é o único jogo que nao apresenta esta funcionalidade. Esta situação encontra-se intimamente ligada à exclusividade com a *PlayStation* e os equipamentos que esta utiliza.
- Na funcionalidade de informação, apenas o *The Last of Us 2* e o *Division 2* conseguem cumprir o duplo objetivo de informar os jogadores e apresentar esta informação logo no ecrã inicial de jogo.
- O *Mortal Kombat 11* foi o jogo com o menor número de funcionalidades, tendo falhas a nível dos menus de jogo. Estas falham são relativas a menus complexos e com muita informação, textos com tamanho pequeno e baixo contraste na palete de cores utilizada.

2.3 Tecnologias

Nesta secção são apresentadas algumas das tecnologias utilizadas na área, assim como diferentes alternativas a serem utilizadas no desenvolvimento do projeto. Tratando-se de um projeto auto-proposto, não existe a necessidade de utilizar tecnologias pré-definidas, sendo um processo totalmente dependente das escolhas do autor. Desta forma, as tecnologias apresentadas procuram realçar um espetro global das diferentes opções possíveis para a realização do projeto.

2.3.1 Motores de Jogo

A primeira subsecção foca-se em motores de jogo, que são ferramentas imprescindíveis no rápido desenvolvimento de jogos digitais. A escolha desta ferramenta impacta todo o processo de desenvolvimento, sendo fundamental escolher a mais adequada a cada tipo de projeto.

- **Unity**: O *Unity* (Unity s.d.) é um motor de jogo que permite a criação de diferentes tipos de entretenimento interativo, sendo conhecido pela sua rápida capacidade prototipagem (Haas 2014). A sua primeira versão foi lançada a 6 de junho de 2005 na Dinamarca por David Helgason, Joachim Ante e Nicholas Francis. Foi desenvolvido

com o "[...] objetivo de criar um motor de jogo acessível, com ferramentas profissionais para desenvolvedores amadores."[5] (Haas 2014).

De acordo com Craighead (Craighead, Burke e Murphy 2008), as características que fazem do *Unity* um excelente escolha são:

- **Documentação**: Documentação completa de toda a plataforma, assim como exemplos da sua aplicação em múltiplos contextos. Esta documentação encontra-se atualizada, sendo possível obter para cada versão, a informação das funcionalidades existentes.
- **Comunidade**: Comunidade *online* ativa e que procura constantemente ajudar e auxiliar os novos desenvolvedores a usar a plataforma. Direção e equipa de desenvolvimento recetivos aos pedidos e *feedback* da comunidade, tendo sido muitas das funcionalidades existentes adicionadas a pedido dos utilizadores.
- **Usabilidade**: Para novos utilizadores o *Unity* é simples e fácil de utilizar. Todos os objetos do projeto estão organizados hierarquicamente, podendo ser arrastados para o ambiente de jogo. Estes podem ter associados a si comportamentos, físicas e até mesmo elementos de interface gráfica.
- **Distribuição**: A plataforma consegue gerar aplicações que podem ser complicadas para múltiplas plataformas, como *Android*, *Windows*, *WebGL* ou *iOS*. Facilitando a distribuição do jogo através dos binários e permitindo um maior alcance de plataformas, a partir de uma de desenvolvimento.
- **Preço**: Diferentes modalidades de preço consoante as necessidades do utilizador e o tipo de organização a utilizar, sendo no caso de pessoas singulares totalmente grátis, até um determinando rendimento anual.

- **Unreal**: O *Unreal* (Unreal s.d.) é um motor de jogo desenvolvido pela *Epic Games* e lançado em 1998, com o advento de jogos de tiro em primeira pessoa (Sanders 2016). Foi sucedido por 3 outros grande lançamentos, onde houve reescrita do código base e adição de novas funcionalidades, como a compatibilidade com *Android* e *iOS* e sistemas de luzes (Sanders 2016).

  É atualmente o principal motor de jogo na visualização realista de gráficos, vegetação e criação de terreno, sendo indicado para grandes projetos de desenvolvimento 3D pelas suas ferramentas de criação de materiais e optimização visual (Šmíd 2017). Algumas das principais características do motor de jogo são (Unreal s.d.):

  - Criação automática de flora, customização de luz e de terrenos e sistemas para controlo automático de massas de água.
  - Animação de personagens e gravação de movimentos.
  - Rápida renderização de gráficos, edição de materiais, fotorrealismo e reflexões precisas de luz.
  - Simulação do movimento de roupas, de cabelos e da destruição de objetos.
  - Ferramentas de inteligência artificial e uma robusta *framework* preparada para múltiplos jogadores.

[5] Tradução Livre do Autor

  - Motor de áudio especializado e suporte para realidade aumentada e realidade virtual.

- **Godot**: O *Godot* (Godot s.d.) é um motor de jogo totalmente grátis e de código aberto, desenvolvido por *Juan Linietsky* e *Ariel Manzur*, e com a primeira versão lançada em 2014. Aqui, é encontrada uma grande variedade de ferramentas que estimulam o processo inovativo e de colaboração, tendo os seguintes pontos como principais características (Godot s.d.):
  - Criação de ferramentas personalizadas.
  - Nós pré-criados, que facilitam o processo de desenvolvimento e podem ser alterados para ganhar novos comportamentos.
  - Um editor visual, onde todas a interface de utilizador pode ser construída.
  - Animação global e em tempo real dos gráficos, funções de pós-processamento e arquitetura focada na eficiência do processo de renderização.
  - Luzes para jogos em 2 dimensões, animação de imagens, editor de mapas do jogo e trabalho sobre píxeis.
  - Animações de todo o tipo de objetos, inclusive esqueletos de animação.
  - Lançamento para múltiplas plataformas, como *Windows*, *iOS*, *Linux*, *macOS* e *Android*.
  - Ferramentas de otimização de código e múltiplas linguagens de programação.
  - Integração com sistemas de controlo de versões.

Análise e Comparação

Nesta secção é feita uma análise comparativa entre os 3 motores de jogos, com o objetivo de perceber as principais diferenças, em diferentes campos. A tabela 2.4 apresenta as funcionalidades dadas pelos motores de jogo aos desenvolvedores, sendo auxiliares ao processo de desenvolvimento. Foram organizadas da seguinte forma:

- **Linguagens**: As diferentes linguagens de programação que os utilizadores podem utilizar nos motores de jogos.
- **Ambiente de Desenvolvimento**: Os sistemas operativos onde o motor de jogo pode ser utilizado, executando e compilando.
- **Mercado**: Existência de um mercado onde os desenvolvedores podem adquirir *plugins*, ferramentas e imagens.

Tabela 2.4: Funcionalidades para desenvolvedores.

Motor de Jogo	**Funcionalidades para Desenvolvedores**		
	Linguagens	**Ambiente de Desenvolvimento**	**Mercado**
Unity	C#	macOs, Windows, Linux	Sim
Unreal	C++	macOs, Windows, Linux	Sim
Godot	C#, C++, GDScript	macOs, Windows, Linux	Sim

A partir da análise da tabela 2.4, é possível aferir que o *Godot* é o motor de jogo que mais linguagens de programação suporta. Relativamente às outras funcionalidade, todos os motores de jogos possuem os principais ambientes de desenvolvimento e um mercado. É importante salientar que o mercado do *Unity* e do *Unreal* são os mais desenvolvidos e completos, com uma grande oferta.

A tabela 2.5 apresenta as funcionalidades de acessibilidade existentes nos motores de jogo. Foram definidos dois critérios principais de acessibilidade:

- **Leitores de Ecrã**: Interface de utilizador adaptada, de forma a que possa ser lido pelo sistema operativo. Isto possibilita aos leitores de ecrã a leitura correta das interfaces gráficas.
- **Scripting Visual**: Escrita de *scripts* sem linguagens de programação e através do uso de elementos gráficos.

Tabela 2.5: Acessibilidade dos motores de jogo.

Motor de Jogo	**Acessibilidade**	
	Leitores de Ecrã	**Scripting Visual**
Unity		X
Unreal	X	X
Godot		X

A análise da tabela 2.5 revela que de todos os motores de jogo, apenas o *Unreal* possui suporte nativo a leitores de ecrã. No entanto, todos os motores de jogo apresentam soluções relativas ao *scripting* visual.

Um dos fatores importantes no momento da escolha de um motor de jogo é o preço e as taxas aplicadas sobre os lucros. De forma a entender como cada um dos motores de jogo se enquadra, foi elaborada a tabela 2.6. Esta elaboração teve em conta os seguintes critérios:

- **Grátis**: Representa planos onde o utilizador do motor de jogo não possui a obrigatoriedade de pagar um valor inicial ou uma mensalidade para utilizar o motor de jogo. O plano grátis pode ainda se dividir em dois tipos:
  - **Sem Taxas**: No plano sem taxas, o utilizador não precisa de paga nada pelos lucros obtidos com o motor de jogo.
  - **Com Taxas**: Já no plano com taxas, o utilizador necessita de pagar uma percentagem dos lucros obtidos a utilizar o motor de jogo. Esta percentagem é definida de acordo com as políticas da empresa desenvolvedora.
- **Pago**: Nestes planos o utilizador necessita de pagar uma quantia inicial ou uma mensalidade/anuidade para utilizar o motor de jogo e a suas funcionalidades.

Tabela 2.6: Tipos de Conta dos motores de jogo.

Motor de Jogo	**Planos**		
	Grátis		**Pago**
	Sem Taxas	**Com Taxas**	
Unity	X		X
Unreal	X	X	
Godot	X		

A partir da análise da tabela 2.6 é possível destacar o *Godot*, sendo o único motor de jogo sem qualquer taxa ou pagamentos periódicos. Nos outros dois motores de jogo, a utilização é totalmente grátis até um determinado valor de lucro. A partir desse valor e no caso do *Unreal* existe uma taxa sobre os lucros obtidos, no do *Unity* é uma mensalidade.

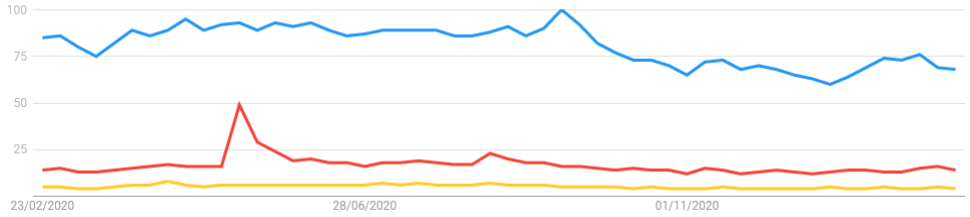


Figura 2.2: Popularidade dos motores de jogo nos últimos 12 meses (retirado de (*Google Trends* s.d.)).

A figura 2.2 representada o interesse por cada um dos motores busca, segundo o *Google Trends* (*Google Trends* s.d.). Esta visualização utiliza dados dos últimos 12 meses, em todo o mundo. A linha de cima representa o motor de jogo *Unity*, a linha do meio representa o *Unreal* e por fim, a linha de baixo representa o motor de jogo *Godot*.

Os valores da escala à esquerda representam o interesse de pesquisa relativo numa determinado momento. Isto significa que um termo que tenha o valor de 50 possui nesse momento metade da popularidade de um termo com o valor de 100.

A partir da análise desta figura, constata-se que o *Unity* foi o motor de jogo com maior interesse e popularidade nos últimos 12 meses, seguido pelo *Unreal* e o *Godot*. Esta análise salienta o interesse e popularidade deste, podendo ser explicado como o resultado de uma grande comunidade de utilizadores ativos.

Grandes comunidades de utilizadores, permitem a existência de elevados números de fóruns, de soluções já implementadas e de documentação, o que facilita a utilização do motor de jogo na resolução de diversos problemas.

2.3.2 Gestão de Projeto

Nesta subsecção são apresentadas tecnologias relacionadas à gestão, planeamento e organização de tarefas, fundamentais para o bom funcionamento e coordenação do projeto com o planeamento. São também referidas as tecnologias necessárias para controlo de versões e segurança da implementação.

- **Git**: O *Git* (Git s.d.) é um sistema de controlo de versões, desenvolvido por Linus Torvalds, que possibilita a gestão das diferentes versões de um projeto e ao mesmo tempo serve como *backup*[6]. Como este permite a associação de mensagens às diferentes versões, o processo de gestão das alterações realizadas fica facilitado.

  Tal como referido por (Loeliger e McCullough 2012), as características mais relevantes do *Git* são: a possibilidade do desenvolvimento de projetos paralelamente e em diferentes localizações, a rapidez e eficiência de sincronização da informação com o

[6] *Backup* é uma cópia de informação ou ficheiros, a ser usado caso a informação original fique corrompida ou seja apagada (Techopedia 2011).

repositório *online*, a integridade e confiança criadas através de funções de encriptação e a imutabilidade. Esta última característica leva à criação de um histórico de mudanças, tornando possível a comparação entre diferentes versões.

- **Bitbucket**: O *Bitbucket* (Bitbucket s.d.) é uma plataforma que permite a hospedagem de repositórios, sendo que se encontra intimamente ligado ao *git*. Com esta tecnologia, os repositórios são guardados na nuvem, o que permite um nível maior de segurança caso ocorram perdas e a acessibilidade a partir de qualquer lugar e em qualquer dispositivo.

  Tal como salientado pela plataforma do *Bitbucket*, existe integração com diversas tecnologias:

  - **Planeamento** das tarefas a realizar, através de plataformas como o *Jira* e o *Trello*.
  - **Colaboração** entre utilizadores, através da integração com múltiplos *IDE's*[7] e funcionalidades de revisão de código.
  - **Lançamento de Versões** através da automatização de processos de testes e de *builds* produzidas.

- **Trello**: A plataforma *Trello* (Trello s.d.) permite organizar e gerir as tarefas realizadas ao longo do projeto, sendo possível verificar o estado de execução em que se encontram. Tal como referido por Johnson (Johnson 2017), a plataforma elimina a necessidade de instalação de programas e possui diferentes tipos de planos (Trello s.d.), consoante as necessidades dos utilizadores. Com a plataforma é possível criar listas, que permitem a divisão das tarefas pelo seu grau de desenvolvimento ou por temas; e sub-grupos/cartas. Estes sub-grupos podem conter imagens, comentários, etapas ou descrições (Johnson 2017).

  O método utilizado na plataforma, também conhecido por *Kanban*, foi desenvolvido com o objetivo de eliminar a perda de tempo e otimizar a produtividade (Ostergaard 2016). Assim, torna-se possível registar e manter em destaque todas as atividades relacionadas ao projeto (Johnson 2017).

2.3.3 Áudio

Nesta subsecção são apresentadas algumas das tecnologias utilizadas na produção e controlo de áudio em jogos. Tal como referido secção 2.1.4, este é um dos pontos chave dos jogos para cegos.

- **Audacity**: O *Audacity* (Audacity s.d.) é um editor de áudio totalmente grátis, podendo ser utilizado em *Windows*, *macOS* e *Linux*. Este *software* possui o seu código-fonte totalmente aberto, sendo desenvolvido por grupos de voluntários da comunidade. As principais características da ferramenta são:

  - **Gravação**: Gravação de voz através de microfone e digitalização de gravações de outros meios.

[7] *IDE's - Integrated Development Environment* é uma ferramenta que permite, entre outras funcionalidades, a edição e correção de código, o destaque da sintaxe utilizada nas respetivas linguagens de programação e a organização do projeto (*What is an IDE?* s.d.).

  - **Exportação de Ficheiros**: Importação de ficheiros de som e exportação de ficheiros em múltiplos formatos.
  - **Plugins**[8]: Integração com diversos *plugins*, para aplicação de efeitos e suporte à criação de *plugins* personalizados.
  - **Edição**: Fácil edição de áudio, com funcionalidades de corte, cópia e colagem.
  - **Efeitos**: Aplicação de efeitos sobre o áudio, em tempo real.
  - **Acessibilidade**: Possibilidade de controlar seleções e menus através de uma grande variedade de atalhos de teclado.
  - **Análise**: Ferramentas adequadas para visualização e análise de frequência.

- **LMMS**: O *Linux MultiMedia Studio (LMMS)* (LMMS s.d.) é um *Digital Audio Workstation (DAW)*[9] que permite a geração, sintetização e automação de áudio numa interface simples. É uma tecnologia grátis, de código aberto, sendo desenvolvida pela comunidade e com suporte para *Windows*, *macOS* e *Linux*.

  Os diferentes sons podem ser modificados com o uso de uma grande variedade de *plugins*, que fazem uso de efeitos sonoros e instrumentos para facilmente criar trilhas sonoras. Com esta ferramenta é também possível combinar trilhas sonoras para a criação de músicas e exportá-las em diferentes formatos.

  Esta tecnologia é também caracterizada pela boa integração e controlo que possui. A composição das notas utilizadas, pode ser efetuada através do teclado ou via *Musical Instrument Digital Interface (MIDI)*[10].

- **FL Studio**: Tal como o LMMS, o *FL Studio* (Studio s.d.) é um DAW, que permite a rápida criação de música, através de uma extensa livraria de amostras e exemplos, livres de direitos autorais.

  Possui diversas ferramentas para mistura com alto nível de complexidade, flexibilidade na sequenciação de diferentes faixas de música, automatização da informação enviada para *plugins*, uma grande variedade de instrumentos nativos e efeitos sonoros e a implementação de padrões pré-definidos, que permitem o uso de instrumentos criados por terceiros.

Análise e Comparação

Nesta secção é feita uma análise comparativa entre os 3 programas, de forma a perceber as diferenças mais significativas entres eles. A tabela 2.7 apresenta quais as funcionalidades oferecidas pelos programas analisados, sendo organizadas da seguinte forma:

- **Suporte Virtual Studio Technology (VST)**[11]: Possibilidade de utilizar e importar *plugins* de VST.

[8] *Plugins* são softwares adicionados a um programa base e que permitem a adição de funcionalidades ou melhoraria daquelas já existentes (Hope 2020).

[9] *DAW* é *software* utilizado com o objetivo de gravar, editar e misturar áudio digital (*What Is A DAW Program? Digital Audio Workstation* 2021).

[10] *MIDI* é um formato desenvolvido como padrão para a reprodução de música em instrumentos digitais e no computador, permitindo a criação do som de qualquer instrumento (Airy e Parr 2001; Cataltepe, Yaslan e Sonmez 2007).

[11] VST são efeitos de áudio que permitem a simulação de equipamentos criados em estúdio (*What Is a VST? All Your VST Questions Answered!* 2020).

- **Suporte MIDI**: Possibilidade de controlar os efeitos e o áudio através de de controladores MIDI.
- **Instrumentos Virtuais**: Possibilidade de aplicar áudio de instrumentos virtuais.
- **Gravação**: Possibilidade de gravar áudio através de microfone.

Tabela 2.7: Funcionalidades dos programas de áudio.

Programa	**Funcionalidades**			
	Suporte VST	**Suporte MIDI**	**Instrumentos Virtuais**	**Gravação**
Audacity	X			X
LMMS	X	X	X	
FL Studio	X	X	X	X

A partir da análise da tabela 2.7, pode-se aferir que o *FL Studio* é o programa mais completo, tendo suporte para as 4 funcionalidades analisadas. Todos o programas possuem suporte para *plugins VST*, já relativamente ao suporte de controlos *MIDI* e integração com instrumentos virtuais o *Audacity* não possui nenhuma das funcionalidades. Na gravação de áudio, o *LMMS* é o único programa sem a funcionalidade.

A tabela 2.8 apresenta funcionalidades extra dos programas, aumentando a sua transparência, o seu alcance e a fidelização. Estas funcionalidade estão organizada da seguinte forma:

- **Código Aberto**: Possibilidade de modificação e utilização do *software* de forma totalmente grátis.
- **Multiplataforma**: Possibilidade de execução em diversas plataformas ou sistemas, como *Windows*, *macOS* e *Linux*.
- **Atualizações**: Existência de atualizações e manutenção constante dos programas, através da adição de novas funcionalidades e correção de *bugs*.

Tabela 2.8: Extras dos programas de áudio.

Programa	**Extras**		
	Código Aberto	**Multiplataforma**	**Atualizações**
Audacity	X	X	X
LMMS	X	X	X
FL Studio		X	X

Ao analisar a tabela 2.8 pode-se aferir que o *Audacity* e o *LMMS* possuem todas as funcionalidades extra, sendo os programas mais completos neste aspeto. No caso do *FL Studio*, o seu código-fonte não é aberto a modificações.

De forma a perceber os diferentes planos de contas, foi criada a tabela 2.9. A análise teve em conta 2 planos diferentes - o **Plano Grátis** e o **Plano Pago**:

Da análise da tabela 2.9 pode-se aferir que o *FL Studio* é o único programa com uma versão grátis e uma versão paga. Todavia, é importante referir que no caso do *Audacity* e do *LMMS*, as versões grátis possuem todas as funcionalidades, enquanto que a versão grátis do *FL Studio* possui muitas limitações.

Tabela 2.9: Tipos de conta dos programas de áudio.

Programa	Planos	
	Grátis	Pago
Audacity	X	
LMMS	X	
FL Studio	X	X

Por fim, com a tabela 2.10 é possível entender como os diferentes programas se comportam a nível de usabilidade. Para isto foram escolhidos 3 critérios:

- **Efeitos em Tempo Real**: Aplicação de efeitos sonoros ao áudio é imediata, sendo feitas as alterações diretamente.
- **Portabilidade**: Alta portabilidade do programa, o que permite a sua rápida instalação e utilização.
- **Arraste de Menus**: Possibilidade de arraste de menus para diferentes zonas do ecrã, o que permite ao utilizador escolher a disposição do seu ambiente de trabalho.

Tabela 2.10: Usabilidade dos programas de áudio.

Programa	Usabilidade		
	Efeitos em Tempo Real	Portabilidade	Arraste de Menus
Audacity		X	
LMMS	X	X	X
FL Studio	X		X

Com a tabela 2.10 percebe-se que o *FL Studio* é a plataforma mais capaz, utilizando os 3 critérios de usabilidade. O *FL Studio* peca em termos de portabilidade, devido ao seus extensos ficheiros de instalação. O *Audacity* apenas possui o critério de portabilidade.

2.3.4 Acessibilidade

Nesta subsecção são apresentadas as tecnologias intimamente ligadas ao processo de acessibilidade para cegos e que se focam em receber e enviar informação corretamente. Desta forma, são tecnologias que o utilizador deve adquirir para melhorar a sua integração.

- **Leitores de Ecrã**: Um leitor de ecrã é um *software* que se encontra ligado às ações do sistema operativo, obtendo a partir dele informações dos textos, ícones e menus que aparecem no ecrã (Watson 2020).

  A informação obtida pelo *software* através do sistema operativo é enviada ao utilizador através de duas formas: voz e *braille* (Watson 2020). Para comunicar através de voz o programa utiliza conversores de texto para voz; para a conversão para *braille* é necessária a existência de equipamentos externos, que consigam representar esta informação (Watson 2020). A maioria dos utilizadores procuram a solução de áudio, porque a transformação para *braille* é um processo lento e que possui a inconveniência de equipamentos e custos extra (Stockman e Metatla 2008). De seguida são apresentados alguns exemplos de leitores de ecrã:

  - *Voice Over* (Over s.d.) criado pela *Apple* para *macOS* e *iOS*.

  - *Job Access with Speech (JAWS)* (JAWS s.d.) da *Freedom Scientific* para *Windows* e *Disk Operating System (DSO)*.
  - *Non-Visual Desktop Access (NVDA)* (NVDA s.d.) criado pelo *NonVisual Desktop Access project* para *Windows* e de código aberto.
  - *Dolphin Supernova* (Supernova s.d.) da *Dolphin Computer Access Inc.* para *Windows* e *macOS*.

- **Texto para Voz**: As ferramentas de texto para voz possibilitam a transformação de texto escrito em voz. Estes sistemas têm um grande número de aplicações desde leitura de texto para cegos, centros de atendimento automático, relato de notícias ou indicações automatizadas de direções (Taylor 2009). De seguida são apresentadas algumas aplicações e *Application Programming Interfaces (API's)* de processamento de texto para voz:
  - Google Text to Speech API (Speech s.d.) da *Google*, com uma seleção de mais de 220 vozes em 40 idiomas distintos e alta fidelidade e exclusividade.
  - VoiceForge API (VoiceForge s.d.) da *Cepstral*, com transformação para áudio de alta qualidade e para qualquer tipo de dispositivo.
  - Amazon Polly API (Polly s.d.) da *Amazon*, focado na alta qualidade do serviço, baixos tempo de resposta e grande suporte de vozes e linguagens.
  - Microsoft Cognitive Services (Services s.d.) da *Microsoft*, com mais de 200 vozes em 50 idiomas, criando variantes que falam naturalmente.

- **Sensores Hápticos**: Cegos podem usar interfaces hápticas em diferentes tipos de programas, como por exemplo experiências simuladas em laboratório ou na transformação de sinais visuais para representação tátil (Colwell et al. 1998; Sjöström 2001). Tal como referido por (Perret e Vander Poorten 2018), a sensação do toque é muito complexa, podendo ser divida em: sensações de textura, temperatura e forma; e sensações de resistência e força. As luvas hápticas juntam os dois conceitos proporcionando uma forma alternativa de receção de *feedback*.

- **Sistemas Braille**: Sistemas de *braille* permitem aos cegos a substituição tátil das formas das letras, através de uma série de pontos levantados e que podem ser lidos com os dedos (Sadato 2005). Aqui existem diversos equipamentos adaptados (*Computer Science learning for school students* s.d.):
  - Teclados de *braille* que permitem ao utilizador escrever instruções e texto em *braille*.
  - Monitores que lêem a informação do computador, apresentando-a num ecrã que se atualiza em *braille*.

Capítulo 3

Análise de Requisitos

Este capítulo foca-se no levantamento dos requisitos do projeto através da análise do problema e das necessidades dos utilizadores, realizados nas secções 1.2 e 2.1.3 respetivamente. De seguida é feita uma classificação destes, de acordo com o seu tipo: não funcionais ou funcionais. Por fim e de acordo com o requisitos funcionais delineados são criados os casos de uso.

De seguida apresentam-se todos os requisitos definidos:

- **R01**: Os jogadores devem poder alterar os níveis de dificuldade do jogo.
- **R02**: Os jogadores devem poder alterar o contraste entre os esquemas de cor.
- **R03**: Os jogadores devem poder alterar o tamanho das letras.
- **R04**: As cores de jogo não devem ser utilizadas para transmitir informação importante.
- **R05**: Deve ser possível utilizar diferentes tipos de equipamentos para controlar as ações de jogo.
- **R06**: O jogador deve pode ajustar individualmente as definições de áudio, como música, diálogo e efeitos sonoros.
- **R07**: O jogador deve poder alterar manualmente a configuração do botões e teclas utilizadas para controlar as ações de jogo.
- **R08**: Os menus de jogo devem possuir um baixo nível de complexidade para permitir uma navegação rápida e intuitiva.
- **R09**: O jogo deve alertar o jogador sobre as funcionalidades de acessibilidade existentes durante o acesso ao jogo.
- **R10**: O jogo deve enviar pistas de áudio para transmitir informação ao jogador.
- **R11**: Todos os menus de jogo e texto devem ser convertidos para fala.
- **R12**: O jogo deve suportar a língua portuguesa e inglesa.
- **R13**: Todas as ações de jogo devem transmitir *feedback* auditivo.
- **R14**: O jogo deve possuir um tutorial para a explicação das mecânicas de jogo.
- **R15**: O jogador deve poder pausar o jogo em qualquer momento.
- **R16**: O jogo deve funcionar em múltiplas plataformas
- **R17**: O jogo deve enviar informação simples e concisa.

- **R18**: O jogo deve enviar *feedback* através de sonificação para o jogador.
- **R19**: O jogo deve se ajustar ao tamanho dos ecrãs.
- **R20**: Todo o conteúdo deve estar relacionado ao jogo.
- **R21**: O jogador deve poder desligar e ligar a opção que converte menus em fala.
- **R22**: Todos as informações de áudio devem ser transmitidas em tempo real.
- **R23**: Todos as conversões para fala devem ser precisas.

3.1 Requisitos Não Funcionais

Requisito não funcionais são requisitos que representam restrições e níveis de qualidade do *software*, expressando como este se vai comportar e que característica possui (Cysneiros s.d.).

Dos requisitos listados anteriormente foi possível definir os seguintes requisitos como requisitos não funcionais:

- **R04**: As cores de jogo não devem ser utilizadas para transmitir informação importante.
- **R05**: Deve ser possível utilizar diferentes tipos de equipamentos para controlar as ações de jogo.
- **R08**: Os menus de jogo devem possuir um baixo nível de complexidade para permitir uma navegação rápida e intuitiva.
- **R09**: O jogo deve alertar o jogador sobre as funcionalidades de acessibilidade existentes durante o acesso ao jogo.
- **R10**: O jogo deve enviar pistas de áudio para transmitir informação ao jogador.
- **R11**: Todos os menus de jogo e texto devem ser convertidos para fala.
- **R12**: O jogo deve suportar a língua portuguesa e inglesa.
- **R13**: Todas as ações de jogo devem transmitir *feedback* auditivo.
- **R14**: O jogo deve possuir um tutorial para a explicação das mecânicas de jogo.
- **R16**: O jogo deve funcionar em múltiplas plataformas
- **R17**: O jogo deve enviar informação simples e concisa.
- **R18**: O jogo deve enviar *feedback* através de sonificação para o jogador.
- **R19**: O jogo deve-se ajustar ao tamanho dos ecrãs.
- **R20**: Todo o conteúdo deve estar relacionado ao jogo.
- **R22**: Todos as informações de áudio devem ser transmitidas em tempo real.
- **R23**: Todos as conversões para fala devem ser precisas.

3.1.1 FURPS+

FURPS+ é um modelo proposto por Robert Grady and Hewlett-Packar (Khosravi e Guéhéneuc 2004) que permite classificar detalhadamente os requisitos não funcionais. Cada uma das letras deste acrónimo representam uma escala de análise, dividindo-se da seguinte forma: **F**uncionalidade, **U**sabilidade, Confiabilidade "**R**eliability", **P**erformance e **S**uportabilidade. O "+"refere-se a restrições extra de desenho, implementação ou de interface (Eeles 2005).

Funcionalidade

Requisitos de funcionalidade estão relacionados às funções fornecidas de acordo com as necessidade de um sistema (Khosravi e Guéhéneuc 2004).

Os requisitos de funcionalidade encontrados foram os seguintes:

- **R04**: As cores de jogo não devem ser utilizadas para transmitir informação importante.
- **R09**: O jogo deve alertar o jogador sobre as funcionalidades de acessibilidade existentes durante o acesso ao jogo.
- **R10**: O jogo deve enviar pistas de áudio para transmitir informação ao jogador.
- **R11**: Todos os menus de jogo e texto devem ser convertidos para fala.
- **R13**: Todas as ações de jogo devem transmitir *feedback* auditivo.
- **R14**: O jogo deve possuir um tutorial para a explicação das mecânicas de jogo.
- **R18**: O jogo deve enviar *feedback* através de sonificação para o jogador.

Usabilidade

Requisitos de usabilidade encontram-se relacionados a aspetos de estética e de consistência numa interface e a facilidade na sua compreensão (Eeles 2005).

Os requisitos de usabilidade associados foram os seguintes:

- **R08**: Os menus de jogo devem possuir um baixo nível de complexidade para permitir uma navegação rápida e intuitiva.
- **R17**: O jogo deve enviar informação simples e concisa.
- **R19**: O jogo deve-se ajustar ao tamanho dos ecrãs.
- **R20**: Todo o conteúdo deve estar relacionado ao jogo.

Confiabilidade

Requisitos de confiabilidade estão associados à capacidade do sistema manter a sua performance e precisão (Eeles 2005).

O requisito de confiabilidade encontrado foi o seguinte:

- **R23**: Todas as conversões para fala devem ser precisas.

Performance

A performance encontra-se associada às características temporais do *software*, como tempos de resposta ou de transferência (Eeles 2005).

O requisito de performance encontrado foi o seguinte:

- **R22**: Todos as informações de áudio devem ser transmitidas em tempo real.

Suportabilidade

Suportabilidade encontra-se relacionada à capacidade do sistema em ser configurado, adaptado e compatível com outros sistemas (Eeles 2005).

Os requisitos ligados à suportabilidade foram os seguintes:

- **R05**: Deve ser possível utilizar diferentes tipos de equipamentos para controlar as ações de jogo.
- **R16**: O jogo deve funcionar em múltiplas plataformas

Restrições (+)

O "+"é usado para outras categorias como restrições, que podem ser físicas, de implementação ou de desenho (Eeles 2005).

O requisito ligado às restrições foi o seguinte:

- **R12**: O jogo deve suportar a língua portuguesa e inglesa.

3.2 Requisitos Funcionais

Requisito funcionais são requisitos que representam o que um *software* deve realizar, utilizando materiais de entrada para produzir produtos de saída (Cysneiros s.d.).

Dos requisitos listados anteriormente foi possível definir os seguintes 7 requisitos como requisitos funcionais:

- **R01**: Os jogadores devem poder alterar os níveis de dificuldade do jogo.
- **R02**: Os jogadores devem poder alterar o contraste entre os esquemas de cor.
- **R03**: Os jogadores devem poder alterar o tamanho das letras.
- **R06**: O jogador deve pode ajustar individualmente as definições de áudio, como música, diálogo e efeitos sonoros.
- **R07**: O jogador deve poder alterar manualmente a configuração do botões e teclas utilizadas para controlar as ações de jogo.
- **R15**: O jogador deve poder pausar o jogo em qualquer momento.
- **R21**: O jogador deve poder desligar e ligar a opção que converte menus em fala.

3.2.1 Casos de Uso

De acordo os requisitos funcionais identificados na secção 3.2, desenvolveu-se a figura 3.1 com os casos de uso criados:

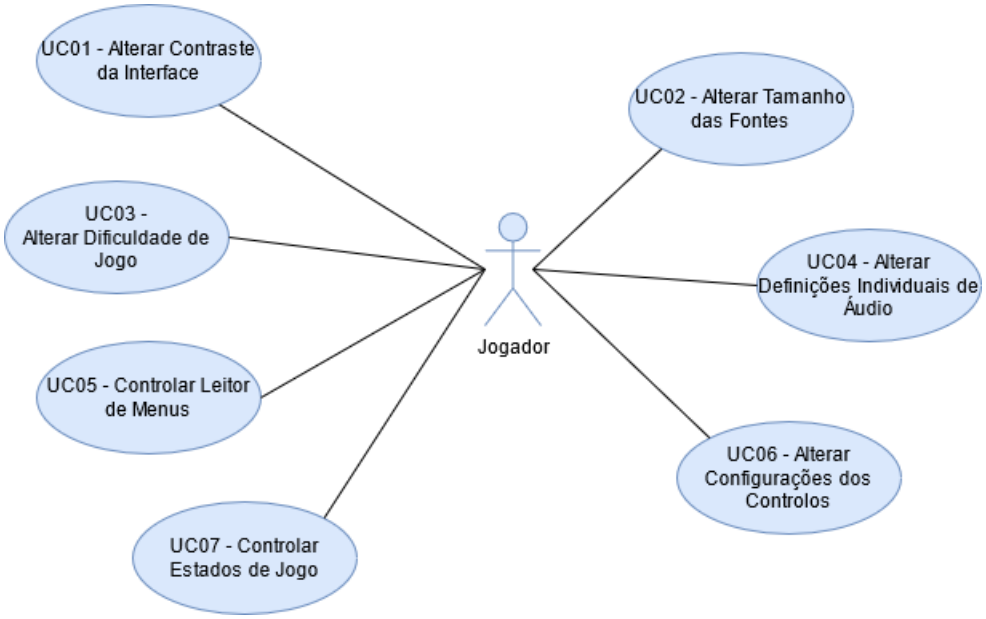


Figura 3.1: Diagrama de Casos de Uso.

- **UC01 - Alterar Contraste da Interface**: O jogador altera o contraste da interface de jogo.
- **UC02 - Alterar Tamanho das Fontes**: O jogador altera o tamanho das fontes utilizadas no texto do jogo.
- **UC03 - Alterar Dificuldade de Jogo**: O jogador altera a dificuldade de jogo.
- **UC04 - Alterar Definições Individuais de Áudio**: O jogador alterar os diferentes elementos de áudio individualmente
- **UC05 - Controlar Leitor de Menus**: O jogador ativa e desativa o leitores de menus.
- **UC06 - Alterar Configurações dos Controlos**: O jogador altera as configurações existentes dos controlos de jogo.
- **UC07 - Controlar Estados de Jogo**: O jogador controla os diferentes estados de jogo.

A criação dos casos de uso baseou-se nas principais necessidades e lacunas existentes na área, sendo necessária a fase de geração e seleção de ideias, desenvolvida no capítulo 4, para a escolha do tema e de outros aspetos fundamentais de jogo.

Capítulo 4

Análise de Valor

Neste capítulo é feita a análise do valor da solução a ser adotada para a resolução do problema. Inicialmente é feita uma identificação e análise das oportunidades existentes, de acordo com a conjuntura atual. De seguida e de acordo com a análise de oportunidades realizada é feita uma seleção de ideias para a escolha da que melhor se enquadra com o projeto a ser realizado.

Também é definido o valor da solução a implementar, através da utilização da proposta de valor e do modelo de negócio CANVAS. Por fim é utilizado o Quality Function Deployment (QDF) para transformar todas as necessidades existentes em valor real.

4.1 Processo de Inovação

Segundo Koen (Koen et al. 2002), o processo de inovação pode ser dividido em 3 fases distintas, que se encontram representadas na figura 4.1:

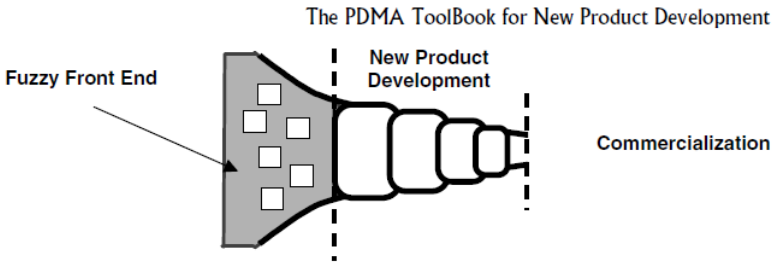


Figura 4.1: Divisão do Processo de Inovação (retirado de (Koen et al. 2002)).

- **Fuzzy Front End (FFE)**: A primeira fase, o **Fuzzy Front End**, representa um estado não estruturado, caótico e experimental, onde existem variáveis incertas de comercialização e financiamento e onde se procura a melhoria e fortificação de conceitos de negócio (Koen et al. 2002).
- **New Product Development (NPD)**: A segunda fase, o **New Product Development**, encontra-se com planos concretos e orientados a objetivos específicos, onde todas as datas são definidas e existem equipas focadas no desenvolvimento e teste de produtos (Koen et al. 2002).
- **Comercialização**: A terceira e última fase, a **Comercialização**, representa-se como o resultado final das duas fases anteriores, sendo o objetivo final do processo inovativo a comercialização do produto desenvolvido (Koen et al. 2002).

4.2 New Concept Development

O *New Concept Development* é um modelo não linear de inovação, criado por Peter Koen e a sua equipa (Koen et al. 2002), com o objetivo de criar conhecimento e terminologia comum no FFE. Este modelo baseia-se em 3 partes fundamentais:

- **O Motor**: O motor encontra-se relacionado à liderança, o plano estratégico e a cultura existentes numa empresa e que vão influenciar diretamente os 5 elementos controláveis (Koen et al. 2002).
- **O Ambiente**: O ambiente representa fatores externos não controlavéis que afetam todo o processo de inovação, como leis, política, economia e competidores (Koen et al. 2002).
- **Os 5 Elementos Controláveis**: Os 5 elementos controláveis representam 5 atividades controladas do FFE, compostas pela identificação de oportunidades, análise de oportunidades, geração de ideias, seleção de ideias e definição de conceitos(Koen et al. 2002).

4.2.1 Identificação de Oportunidades

Esta é a fase onde se identificam as diferentes oportunidades existentes, sendo nelas que são alocados os recursos. Esta oportunidade podem ser enquadradas em diferentes contextos, como respostas à concorrências através de processos mais eficazes, melhoria e a alterações a serviços já existentes ou até a criação de raiz de novos produtos e processos (Koen et al. 2002).

Ardichvili (Ardichvili, Cardozo e Ray 2003) acrescenta ainda que a fase de identificação de oportunidades pode ser dividido em 3 processos: **perceção** de necessidades e falhas existentes no mercado, **descoberta** de espaços de mercado sem soluções e **criação** de soluções totalmente diferenciadas.

O atual projeto enquadra-se no processo de **perceção** de necessidades e falhas existentes no mercado. As oportunidades identificadas têm assim origem nos seguintes fatores:

- Grande crescimento da indústria de jogo digitais, que foi ainda exacerbado pela situação de pandemia vivida atualmente (Marketeer 2020). Como é explicado na secção 1.1, este aumento destaca cada vez mais a necessidade de existência de características de acessibilidade nos jogos, de forma a que estes possam ser jogados por todos sem obstáculos.
- Soluções inexistentes ou incompletas de acessibilidade na indústria de jogos digitais. E tal como demonstrado na secção 2.2.3 estas lacunas acontecem mesmo nos jogos com maior financiamento e investimento, o que leva à inabilidade dos cegos na interação com jogos digitais, como se constatou na secção 2.1.3.

4.2.2 Análise de Oportunidades

Esta é a fase onde se confirma que as oportunidades identificadas são de facto as mais corretas a seguir (Koen et al. 2002). É aqui que se procura obter informação para diminuir as variáveis de incerteza de mercado e de atratividade. São utilizadas técnicas que permitem fazer a análise de oportunidades como (Koen et al. 2002): as principais necessidades dos

consumidores, análise da concorrência e das suas estratégias, segmentação do mercado para análises mais detalhadas ou análise SWOT.

A análise SWOT é uma ferramenta utilizada para o planeamento e gestão estratégica, através do estudo e apresentação de dimensões internas e externas que podem afetar uma organização (Gürel e Tat 2017; Pickton e Wright 1998). Através da utilização das 2 dimensões com fatores positivos e negativos, é possível criar uma matriz de 4 componentes distintos e que criam o seu nome (Gürel e Tat 2017):

- **Strengths**: Forças são características positivas que adicionam valor e se tornam distintas quando comparadas a outras soluções.
- **Weaknesses**: Fraquezas são características negativas que criam lacunas e desvantagens em relação a outras soluções.
- **Opportunities**: Oportunidades representam situações favoráveis e positivas para uma atividade e que permitem obter vantagens.
- **Threats**: Ameaças são situações negativas que prejudicam uma atividade e que criam situações de desvantagem.

A figura 4.2 representa a análise SWOT das oportunidades identificadas na secção 4.2.1:

Figura 4.2: Análise SWOT.

Forças

- *Funcionalidades focadas na acessibilidade* - com o foco do projeto em funcionalidades de acessibilidade, existe diferenciação em relação à grande maioria dos jogos digitais

existentes no mercado e ao mesmo tempo consegue captar um grande número de novos e antigos jogadores com problemas visuais.

- *Pouca burocracia* - sendo o projeto desenvolvido por apenas o autor não existe burocracia associada à tomada de decisões e escolhas, o que permite rapidez e grande adaptação, que vai de encontro com a próxima força descrita.
- *Abertura à mudança* - permite um elevado nível de adaptação às mudanças existentes na indústria, sendo possível a modificação rápida e eficaz de funcionalidades para estarem de acordo com as normas e necessidades existentes no mercado.

Fraquezas

- *Falta de investimento financeiro* - tratando-se de um projeto académico não existe qualquer tipo de financiamento, portanto o autor é obrigado a se adaptar às condições e tecnologias grátis. A não utilização de *software* licenciado pode levar a atrasos na resposta a necessidades e na implementação de funcionalidades.
- *Equipa de desenvolvimento reduzida a um elemento* - sendo a equipa de desenvolvimento constituída apenas pelo autor da presente dissertação, as funcionalidades a serem implementadas estão limitadas pelo tempo disponível. Assim sendo, existe uma situação de desvantagem em relação a outros projetos que possuem equipas com vários elementos.
- *Falta de relevância dentro da indústria* - como se trata de um projeto académico sem produtores e sem nome, no período inicial do seu lançamento não existe tração de uso. Tanto através dos jogadores como também através da criação de boa normas na indústria.

Oportunidades

- *Aumento da regulação relativa à acessibilidade em jogos* - o aumento da regulação permite o destaque daqueles que já possuem este tipo de funcionalidade implementadas. Tornando-se pioneiros e referências dentro da indústria e encontrando-se em posições de destaque quando comparados com outras soluções.
- *Crescimento da indústria de jogo digitais, relacionado ao contexto atual de pandemia* - este crescimento, tal como referenciado na secção 1.1, levou a um aumento de jogadores durante o período de pandemia, o que consequentemente levou ao aumento de jogadores com deficiências visuais e que necessitam de soluções adaptadas às suas necessidades.
- *Soluções existentes na indústria incompletas* - a inexistência e soluções incompletas, tal como analisado em mais detalhe na secção 2.2.3, revela-se uma ótima oportunidade para o desenvolvimento de um jogo que permita colmatar estas lacunas existentes nas funcionalidades para cegos.
- *Indústria com foco cada vez maior na acessibilidade* - tal como referido na oportunidade do aumento da regulação, um foco maior na acessibilidade leva à utilização de jogos com estas características como exemplos e referências, tornando-se uma ótima oportunidade para o desenvolvimento de um jogo acessível.

Ameaças

- *Soluções preconcebidas de acessibilidade* - a utilização de soluções preconcebidas de acessibilidade, através do uso de *software* intermédio, leva a grande diminuição do tempo necessário para implementar este tipo de funcionalidades. Desta forma, torna-se muito difícil competir e adaptar em tempo útil às necessidades de mercado.
- *Mercado altamente competitivo* - como foi visto na secção 1.1, o mercado de jogos digitais é bastante competitivo e com imensa oferta. Desta forma é difícil destacar-se num meio tão maduro, existindo elevada competição pela tecnologia e soluções mais avançadas, que possam atrair o maior número possível de jogadores.

4.2.3 Geração de Ideias

O processo de geração de ideias é a fase onde se criam, desenvolvem e concretizam ideias (Koen et al. 2002). O desenvolvimento e criação destas ideias não se trata de um processo instantâneo e automático, sendo o seu resultado um processo de desenvolvimento contínuo com inúmeras alterações e melhorias (Koen et al. 2002). Este processo pode ser melhorado através do contacto com os utilizadores finais, parceiros de negócios e as equipas de desenvolvimento (Koen et al. 2002).

Para este projeto, tratando-se apenas de um desenvolvedor sem parceiros de negócio, o foco para a geração de ideias tem de ser os utilizadores finais, neste caso os cegos. Tendo em conta que as lacunas referidas nas secções 2.1.3 e 2.2.3 já se encontram contabilizadas através do processo de análise de requisitos realizado no capítulo 3, sobra apenas o tema de jogo, referenciado no 2.1.4. Assim sendo, foram geradas as seguintes ideias:

- Desenvolvimento de um jogo com o tema **Jogo Casual**, onde o foco se encontra na utilização de discurso para descrição do jogo e sonificação, através de ícones auditivos e *earcons*, que transmitem informação ao jogador.
- Desenvolvimento de um jogo com o tema **Jogo de Tiros**, onde o foco se encontra na localização espacial por sonificação dos inimigos e discurso do ambiente de jogo.
- Desenvolvimento de um jogo com o tema **Jogo de Aventura**, onde o foco se encontra na resolução de puzzles, investigação de enigmas e exploração, através do uso de dicas de áudio e discurso.

As 3 ideias geradas, mesmo tendo abordagens diferentes ao problema, cumprem os objetivos definidos na secção 1.2.1. Posto isto, torna-se necessário uma avaliação das que mais se adequam ao projeto, tendo em conta critérios pré-definidos.

4.2.4 Seleção de Ideias

A fase procura a seleção das ideias que melhor cumprem os objetivos e mais valor trazem a uma organização (Koen et al. 2002). Assim sendo, é fundamental a escolha de uma boa ideia para o sucesso de um negócio (Koen et al. 2002).

De forma a atingir os objetivos propostos com esta dissertação, foi utilizado uma ferramenta de apoio à seleção de ideias, o método Analytic Hierarchy Process (AHP), criado por Thomas Saaty (Saaty 1987). Este método foca-se em 3 fases distintas: a construção de uma hierarquia que represente o problema, o estabelecimento de relações e comparação entre as

diferentes alternativas e critérios e por fim a preocupação com a consistência das relações de dependência existentes entre os elementos da hierarquia (Saaty 1987).

Árvore Hierárquica de Decisão

Tal como referido na secção 4.2.3, tendo em conta a análise de requisitos já realizada no capítulo 3, o próximo passo é a decisão do tema do jogo. Assim, na análise através do AHP, o **objetivo** é a escolha do tema de jogo, as **alternativas** selecionadas encontram-se definidas na secção 4.2.3 e para a escolha destas definiram-se os seguintes **critérios**:

- **Popularidade**: Este critério encontra-se relacionado à popularidade de um tema de jogo dentro da comunidade. A popularidade está associada à adesão que um determinado jogo tem, sendo fundamental para que este possa ganhar destaque e visibilidade no ramo. Uma alternativa é melhor que outra, se o seu nível de popularidade for superior.
- **Complexidade**: O critério da complexidade relaciona-se à complexidade de implementação e de manutenção do jogo. Este critério permite enquadrar qual o tema de jogo que possui um maio nível de complexidade no seu desenvolvimento e manutenção. As alternativas que possuem uma menor complexidade de desenvolvimento são superiores às outras.
- **Tempo de Desenvolvimento**: Este critério encontra-se relacionado ao tempo de desenvolvimento do jogo. Tratando-se de um projeto académico com datas previstas de conclusão, é fundamental que o escopo delineado seja fazível neste período de tempo. Assim sendo, a alternativa que possuir um menor tempo de desenvolvimento é superior às restantes.

Na figura 4.3 encontram-se representados as 3 fases da estrutura hierárquica do AHP. O objetivo situa-se no topo da estrutura, os critérios na linha do meio e as diferentes alternativas no fundo:

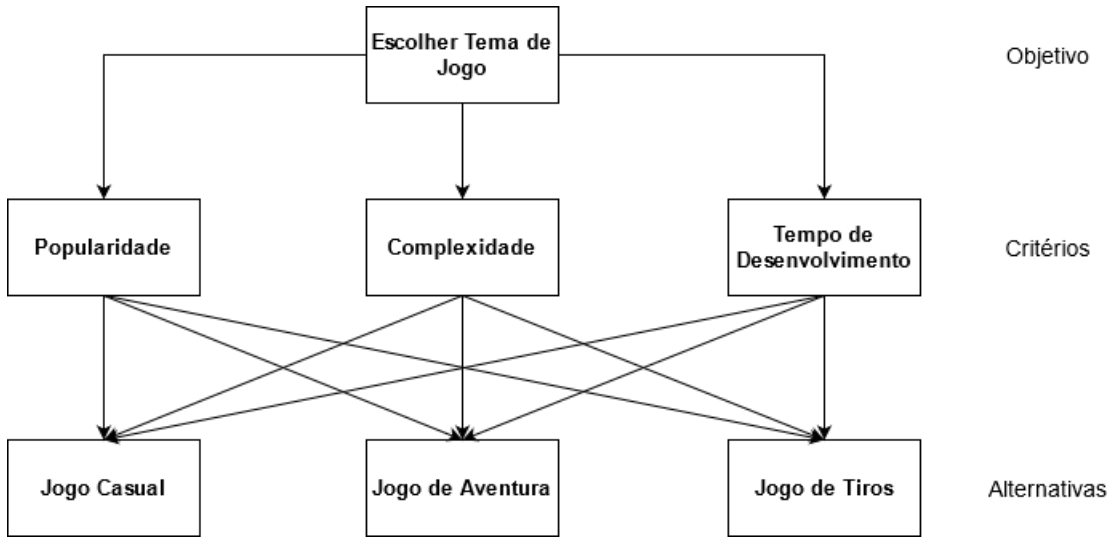


Figura 4.3: Estrutura Hierárquica AHP.

Definição de Alternativas e Critérios

Para a comparação das diferentes alternativas e critérios é necessário a utilização de uma escala que permita definir os diferentes níveis de importância. Esta escala encontra-se representada na tabela 4.1:

Tabela 4.1: Escala fundamental dos níveis de importância (adaptada de (Saaty 1987)).

Nível de Importância	Definição
1	Importância igual
3	Importância moderadamente superior
5	Importância fortemente superior
7	Importância muito fortemente superior
9	Importância extremamente superior
2, 4, 6, 8	Valores de importância Intermédios

Aqui, encontram-se representados 4 níveis intermédios intercalados com 5 níveis principais e que são explicados da seguinte forma (Saaty 1987):

- **Nível 1**: A importância ambos as atividades contribuem igualmente para a os objetivos.
- **Nível 3**: A importância de uma atividade para os objetivos é moderadamente mais forte que outra.
- **Nível 5**: A importância de uma atividade para os objetivos é mais forte que outra.
- **Nível 7**: A importância de uma atividade para os objetivos é muito mais forte que outra e os seus resultados podem ser observados na prática.
- **Nível 9**: A importância de uma atividade para os objetivos é extremamente mais forte que outra, sendo a maior ordem de afirmação possível.

Para efeitos de simplicidade os critérios selecionados são representados com o seguinte formato:

- **A**: Popularidade
- **B**: Complexidade
- **C**: Tempo de Desenvolvimento

Tabela 4.2: Prioridade Relativa de cada Critério.

	A	**B**	**C**	**Prioridade Relativa**
A	1	5	0,33	0,28
B	0,2	1	0,14	0,08
C	3	7	1	0,64

Através da análise da tabela 4.2 é possível destacar que:

- O critério Popularidade é mais importante que o critério Complexidade.
- O critério Tempo de Desenvolvimento é moderadamente mais importante que o critério Popularidade.

- O critério Tempo de Desenvolvimento é muito mais importante que o critério Complexidade.

Também é possível aferir a ordem de prioridade relativa de cada um dos critérios: **Tempo de Desenvolvimento** -> **Popularidade** -> **Complexidade**.

Para o autor, o critério mais importante é o **Tempo de Desenvolvimento**, pois este é um fator essencial para o término do projeto em tempo útil. Assim, é fundamental a escolha de uma alternativa que permita um tempo de desenvolvimento baixo. De seguida, vem a **Popularidade**, que leva à criação de visibilidade para o produto, aumenta o número de novos utilizadores e facilita a inserção na indústria de jogos digitais. Por fim encontra-se a **Complexidade** que afere se a alternativa escolhida é simples de ser implementada e mantida. Como se trata apenas de um desenvolvedor, a manutenção e implementação tomam um valor menos relevante.

Depois da identificação da prioridade relativa de cada critério é necessário a identificação das prioridade de cada alternativa, por cada critério. Por questões de simplicidade as alternativas selecionadas são identificadas da seguinte forma:

- **X**: Jogo Casual
- **Y**: Jogo de Aventura
- **Z**: Jogo de Tiros

Tabela 4.3: Importância Relativa por Critério - Popularidade.

	X	**Y**	**Z**	**Prioridade Relativa**
X	1	0,33	0,14	0,09
Y	3	1	0,25	0,21
Z	7	4	1	0,70

Através da análise da tabela 4.3 é possível destacar que:

- A alternativa Jogo de Aventura é moderadamente mais importante que a alternativa Jogo Casual.
- A alternativa Jogo de Tiros é muito mais importante que a alternativa Jogo Casual.
- A alternativa Jogo de Tiros é moderadamente mais importante que a alternativa Jogo de Aventura.

Também é possível aferir no critério de Popularidade, a ordem de prioridade relativa de cada uma das alternativas: **Jogo de Tiros** -> **Jogo de Aventura** -> **Jogo Casual**.

Para o autor e utilizando os dados de (McCarthy e Richter 2015), a alternativa mais importante no critério de popularidade é o **Jogo de Tiros**, seguida pelo **Jogo de Aventura** e por fim o **Jogo Casual**.

O projeto a ser desenvolvido tem como plataforma essencial o computador, sendo assim utilizados dados para este tipo de ambiente, caso os dados utilizados fossem referentes à popularidade para telemóveis, os resultados seriam o oposto, estando a alternativa de **Jogo Casual** como a mais popular e o **Jogo de Aventura** como o menos popular. No mercado de jogos para telemóveis, os jogos casuais dominam totalmente as vendas (Hill 2021).

Através da análise da tabela 4.4 é possível destacar que:

Tabela 4.4: Importância Relativa por Critério - Complexidade.

	X	**Y**	**Z**	**Prioridade Relativa**
X	1	5	4	0,68
Y	0,2	1	0,5	0,12
Z	0,25	2	1	0,20

- A alternativa Jogo Casual é mais importante que a alternativa Jogo de Aventura.
- A alternativa Jogo Casual é moderadamente mais importante que a alternativa Jogo de Tiros.
- A alternativa Jogo de Tiros é fracamente mais importante que a alternativa Jogo de Aventura.

Também é possível aferir no critério Complexidade, a ordem de prioridade relativa de cada uma das alternativas: **Jogo Casual** -> **Jogo de Tiros** -> **Jogo de Aventura**.

Para o autor a alternativa mais importante no critério de complexidade é o **Jogo Casual**. Estes resultados são expectáveis uma vez que esta alternativa possui mecânicas menos complexas de desenvolvimento o que também permite uma fácil manutenção. As outras alternativas são mais complexas devido à sua associação com físicas de mundo real, movimentação de câmara e controlo de entidade de jogo no caso da alternativa **Jogo de Tiros** e a criação de mundo virtual, níveis e enigmas no caso da alternativa **Jogo de Aventura**.

Tabela 4.5: Importância Relativa por Critério - Tempo de Desenvolvimento.

	X	**Y**	**Z**	**Prioridade Relativa**
X	1	6	5	0,72
Y	0,17	1	0,5	0,10
Z	0,2	2	1	0,18

Através da análise da tabela 4.5 é possível destacar que:

- A alternativa Jogo Casual é muito mais importante que a alternativa Jogo de Aventura.
- A alternativa Jogo Casual é moderadamente mais importante que a alternativa Jogo de Tiros.
- A alternativa Jogo de Tiros é fracamente mais importante que a alternativa Jogo de Aventura.

Também é possível aferir no critério do Tempo de Desenvolvimento, a ordem de prioridade relativa de cada uma das alternativas: **Jogo Casual** -> **Jogo de Tiros** -> **Jogo de Aventura**.

Para o autor a alternativa mais importante no critério de tempo de Desenvolvimento é o **Jogo Casual**. Isto pode ser explicado pelo escopo temporal reduzido das implementações necessárias durante o seu desenvolvimento. As alternativas de **Jogo de Tiros** e **Jogo de Aventura** possuem um maior tempo de desenvolvimento associado à própria natureza do seu tema, com implementações de mundo de jogo ou mecânicas que necessitam de maior um maior investimento temporal.

Depois da identificação da importância relativa da cada alternativa, por critério é necessário identificar qual a alternativa que melhor responde ao problema da estrutura hierárquica. A

tabela 4.6 representa a importância relativa de cada uma das soluções por cada um dos critérios e a prioridade relativa destes critérios:

Tabela 4.6: Importância Relativa das Alternativas por Critério

	A	**B**	**C**
X	0,09	0,68	0,72
Y	0,21	0,12	0,10
Z	0,70	0,20	0,18
Prioridade Relativa	0,28	0,08	0,64

Para o cálculo final da prioridade relativa das alternativas tendo em conta todos os critérios é necessário multiplicar a matriz composta das alternativas, representada na tabela 4.6, pelo vetor de pesos de critério calculado na tabela 4.2 e que também se encontra representado na última linha da tabela anterior. Com estes cálculos é possível obter a prioridade relativa das alternativas, representada na tabela 4.7:

Tabela 4.7: Prioridade Relativa das Alternativas.

	Prioridade Relativa
X	0,54
Y	0,14
Z	0,32

A partir da análise da tabela 4.7 é possível constatar que a melhor alternativa é a **X**, o **Jogo Casual**, que possui um valor de prioridade relativa superior aos das outras alternativas.

Avaliação de Consistência

Para confirmar os resultados obtidos é necessário fazer uma avaliação da consistência das prioridades relativas obtidas. Para fazer esta avaliação é necessário calcular a Razão de Consistência (RC), que é obtida através da fórmula 4.1:

$$RC = \frac{IC}{IR} \tag{4.1}$$

Se o valor do RC for superior a 0,1 então os cálculos feitos anteriormente não são confiáveis, pois não apresentam valores consistentes. Estas avaliações de consistência baseiam-se no pressuposto que se um critério A é superior a um critério B, então o critério A também é superior ao critério C.

Para o cálculo do Índice de Consistência (IC), é necessário saber o valor de n e de λmax através da fórmula 4.2:

$$IC = \frac{\lambda max - n}{n - 1} \tag{4.2}$$

O valor de λmax é obtido pela utilização do maior valor próprio da matriz de prioridades de critérios, apresentada na tabela 4.2. Este cálculo pode ser obtido através da equação $Ax = \lambda \max x$, onde x representa o vetor de prioridades de cada critério e Ax a multiplicação entre este vetor e a matriz de prioridades não normalizada. Os resultados desta multiplicação encontram-se representados na tabela 4.8:

Tabela 4.8: Matriz de Consistência

A	0,87
B	0,22
C	2,01

Com os valores da matriz de consistência e com o vetor de prioridades de cada critério é possível calcular o valor próprio através da equação 4.3:

$$\lambda max = \frac{0,87/0,28 + 0,22/0,08 + 2,01/0,64}{3} = 3,07 \tag{4.3}$$

Com o valor de λmax, é agora possível retornar à fórmula do IC onde se assume que n=3, valor do número de critérios a serem utilizados, para obter a equação 4.4:

$$IC = \frac{3,07 - 3}{3 - 1} = 0,03 \tag{4.4}$$

Com o valor do IC calculado, basta apenas determinar o valor do Índice Aleatório (IR), que é obtido através da tabela 4.9, com o número de critérios que são utilizados (n).

Tabela 4.9: Tabela de Índice Aleatório.

N	1	2	3	4	5	6	7	8	9	10
IR	0,00	0,00	0,58	0,90	1,12	1,24	1,32	1,41	1,45	1,49

Através da análise da tabela 4.9 é possível aferir que para 3 critérios, o IR possui um valor de 0,58, o que já possibilita o cálculo do RC através da equação 4.5:

$$RC = \frac{0,03}{0,58} = 0,06 \tag{4.5}$$

Como o valor obtido através do cálculo do RC é inferior a 0,1, é possível concluir que os valores da prioridades relativas são consistentes. Desta forma validam-se os resultados obtidos pelos cálculos do método AHP, sendo o **Jogo Casual** a melhor alternativa para o tema do jogo a ser desenvolvido.

4.3 Valor da Solução

Após a identificação a solução a ser implementada é fundamental perceber o valor que esta cria para os seus utilizadores finais, no caso específico deste projeto, os cegos e indivíduos com deficiências visuais. Para isto será utilizado o conceito de proposta de valor e o modelo de negócio CANVAS.

4.3.1 Proposta de Valor

O CANVAS da proposta de valor é um modelo que explica as dores dos utilizadores e respetivamente os fatores que permitem a diminuição destas e criação de ganho (Osterwalder et al. 2014). Desta forma, o modelo divide-se em 2 partes: a dos utilizadores, que explica

os seus problemas e a proposta de valor, que explica como se pretende criar valor para o utilizador. A figura 4.4 representa esta modelo:

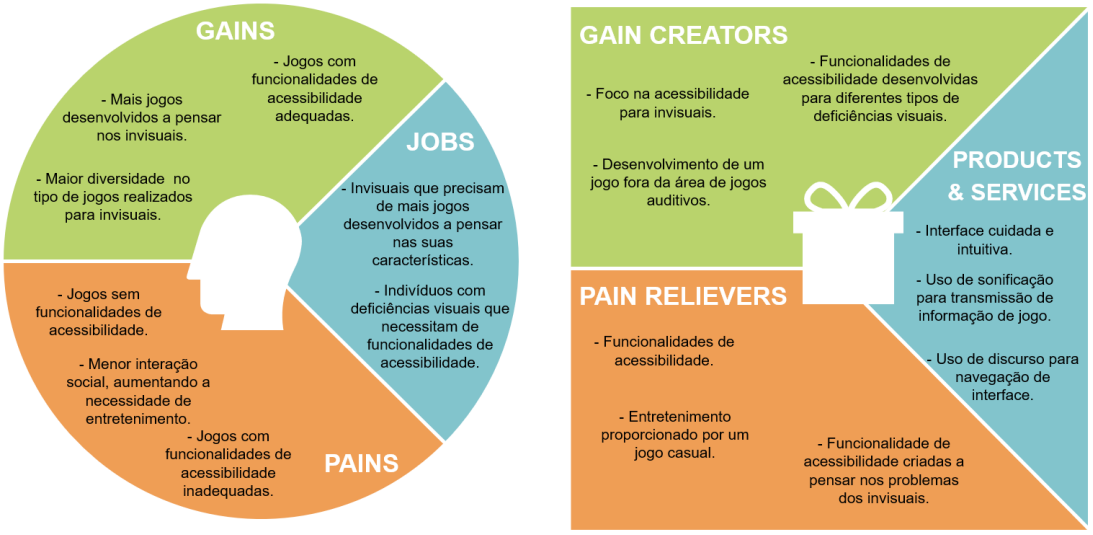


Figura 4.4: Proposta de Valor.

De acordo com a análise da figura 4.4 é possível destacar a seguinte informação sobre o lado dos utilizadores:

- **Ganhos**: Os utilizadores procuram mais jogos desenvolvidos a pensar nas características dos cegos, com mais funcionalidades de acessibilidades e com maior diversidade no tema destes.
- **Dores**: Os principais problemas, tal como salientado na secção 2.1.3, encontram-se nos jogos sem qualquer características de acessibilidade, o baixo nível de interação social causado pela pandemia e que leva ao aumento da necessidade de entretenimento e os jogos com funcionalidades inadequadas.
- **Trabalhos**: Os trabalhos procurados por cegos e indivíduos com deficiências visuais encontram-se associados ao desenvolvimento de jogos com características associados aos seus problemas.

E sobre o lado da proposta de valor:

- **Criador de Ganhos**: A proposta pretender criar valor a estes utilizadores, através do foco nos seus problemas e no desenvolvimento de funcionalidade de acessibilidade para diferentes tipos de deficiências visuais.
- **Produtos e Serviços**: Os produtos oferecidos variam entre interfaces cuidadas e intuitivas, sonificação para a transmissão de informação necessário para o processo de jogo e o uso de discurso para ajudar o utilizador a navegar pela interface e os menus de jogo.
- **Limitador de Dores**: As principais dores destes utilizadores são colmatadas através da criação de funcionalidades de acessibilidade, altos níveis de entretenimento com um jogo casual e funcionalidades especialmente pensadas nas suas dificuldades.

4.3.2 Modelo de Negócio CANVAS

O modelo de negócio CANVAS, proposto por Alexander Osterwalder, pode conceptualizar as características de uma organização através de 3 fatores (Joyce e Paquin 2016): como as suas diferentes partes se organizam para entregar valor ao consumidores, como estas se conectam dentro da organização e como a organização gerar valor através destas ligações. O Modelo de Negócio CANVAS representado na figura 4.5 explica como o produto desenvolvido nesta dissertação se comporta e insere num negócio:

Figura 4.5: Modelo de Negóco CANVAS.

Da análise da figura 4.5 é possível retirar a seguinte informação:

- **Parceiros Principais**: Aqui encontram-se os principais parceiros associados ao desenvolvimento deste produto, sendo assim incluídos como parceiros: as lojas *online* onde o produto vai ser disponibilizado e as comunidades de cegos, que servem como elo de ligação.
- **Atividades Principais**: As atividade principais estão relacionadas a uma contínua análise das necessidades dos jogadores cegos, desenvolvimento de funcionalidades de acessibilidade que sigam normas criadas e criação de novo conteúdo associado ao jogo.
- **Recursos Principais**: Aos recursos necessários para a entrega da proposta de valor encontram-se relacionados à equipa de desenvolvimento e *marketing*, que desenvolve o *software* e a publicidade, que no caso concreto deste projeto é constituída por apenas um elemento, o jogo com as funcionalidade de acessibilidade, as lojas *online* onde o produto é disponibilizado e as redes sociais onde é realizado o *marketing*.
- **Proposta de Valor**: A proposta de valor procura a criação de um produto com uma interface cuidada e intuitiva, onde são utilizadas técnicas de sonificação, que permitem a transmissão de informação de jogo, uso de discurso para ajuda dos clientes durante

a navegação pelos menus, alto nível de entretenimento proporcionado por um jogo casual e funcionalidades de acessibilidade para diferentes tipos de deficiências visuais.

- **Relação com Clientes**: A relação com o cliente será efetuada através de comunidades *online*, por email e através das redes sociais.
- **Canais**: Os canais representam a forma como a proposta de valor é levado para o cliente e encontra-se associado a questionários de satisfação do uso do produto, a venda *online* do produto e *marketing* através das redes sociais.
- **Segmentos de Clientes**: Segmentação refere-se aos segmentos de mercado visados com o produto, que no caso deste projeto são os cegos e os indivíduos com deficiências visuais.
- **Custos**: Os custos encontram-se relacionados ao processo de *marketing* e às taxas de infraestrutura para a disponibilização do produto *online*.
- **Fontes de Receita**: Como se trata de um produto totalmente grátis não existem fontes de receita associadas.

4.3.3 Quality Function Deployment

O QDF é um sistema feito para o desenho de produtos de acordo com as necessidades dos consumidores (Roberts s.d.). O seu processo teórico começou a ser desenvolvido na Mitsubishi em 1972, procurando desenvolver os seguintes fatores (Roberts s.d.):

- Melhor entendimento das necessidades dos consumidores.
- Melhoria da organização e desenvolvimento de projetos.
- Diminuição dos problemas
- Aumento da produtividade.

O primeiro passo consiste na definição das necessidades dos utilizadores. Para o contexto do projeto a ser realizado foram identificadas as seguintes necessidades tendo por bases as lacunas e problemas identificados na secção 2.1.3:

- **Menu Intuitivos e de Fácil Navegação**: O jogo a ser desenvolvido deve possuir menus intuitivos e que permitam um fácil navegação entre eles.
- **Comunicação da Informação da Interface**: O produto deve comunicar a informação existente na interface para o utilizador através de discurso.
- **Comunicação da Informação de Jogo**: O produto deve comunicar a informação de jogo para o utilizador, utilizando técnicas como sonificação ou dicas de áudio.
- **Possibilidade de Modificação da Interface**: O utilizador deve ter a possibilidade de modificar diversos aspetos e configurações da interface, para que esta possa ser adaptada a diferentes características.
- **Comunicação da Informação em Tempo Real**: A comunicação da informação visual através de áudio deve ser realizada em tempo real e sem perdas de informação.

A cada uma destas necessidades foram atribuídos valores de importância absoluta, de 1 a 5, onde representa a mínima importância para o utilizador e 5 representa a importância

máxima. De seguida foi feita a sua normalização, obtendo-se os resultados apresentados na tabela 4.10:

Tabela 4.10: Necessidades dos utilizadores classificadas por importância.

Necessidades	**Importância (Absoluta)**	**Percentagem**
Menu Fáceis e Intuitivos	3	15%
Informação de Interface	5	25%
Informação de Jogo	5	25%
Modificação de Interface	3	15%
Informação em Tempo Real	4	20%

As características de engenharia identificadas que procuram responder às necessidades dos utilizadores foram as seguintes:

- **Interface Gráfica Intuitiva**: Característica associada a interfaces intuitivas e que permitem a sua fácil utilização.
- **Leitor de Menus**: Característica associada à existência de leitores de menu, que permite a leitura dos diferentes menus e interfaces existentes no jogo.
- **Opções de Jogo**: Característica associada a diferentes opções de modificação de interface e de definições de jogo.
- **Tempo de Resposta**: Característica associado ao tempo de resposta das diferentes ações de jogo e da comunicação de informação por áudio.
- **Interface Gráfica Simples**: Característica associado a interfaces simples, com informação concisa, concreta e condensada.

As figuras 4.6 e 4.7 apresentam duas escalas, que permitem respetivamente representar as relações entre as necessidades dos consumidores e as características de engenharia e a correlação entre as diferentes características de engenharia:

Matriz de Relação		
	Relação Forte	9
	Relação Média	3
	Relação Fraca	1
	Sem Relação	0

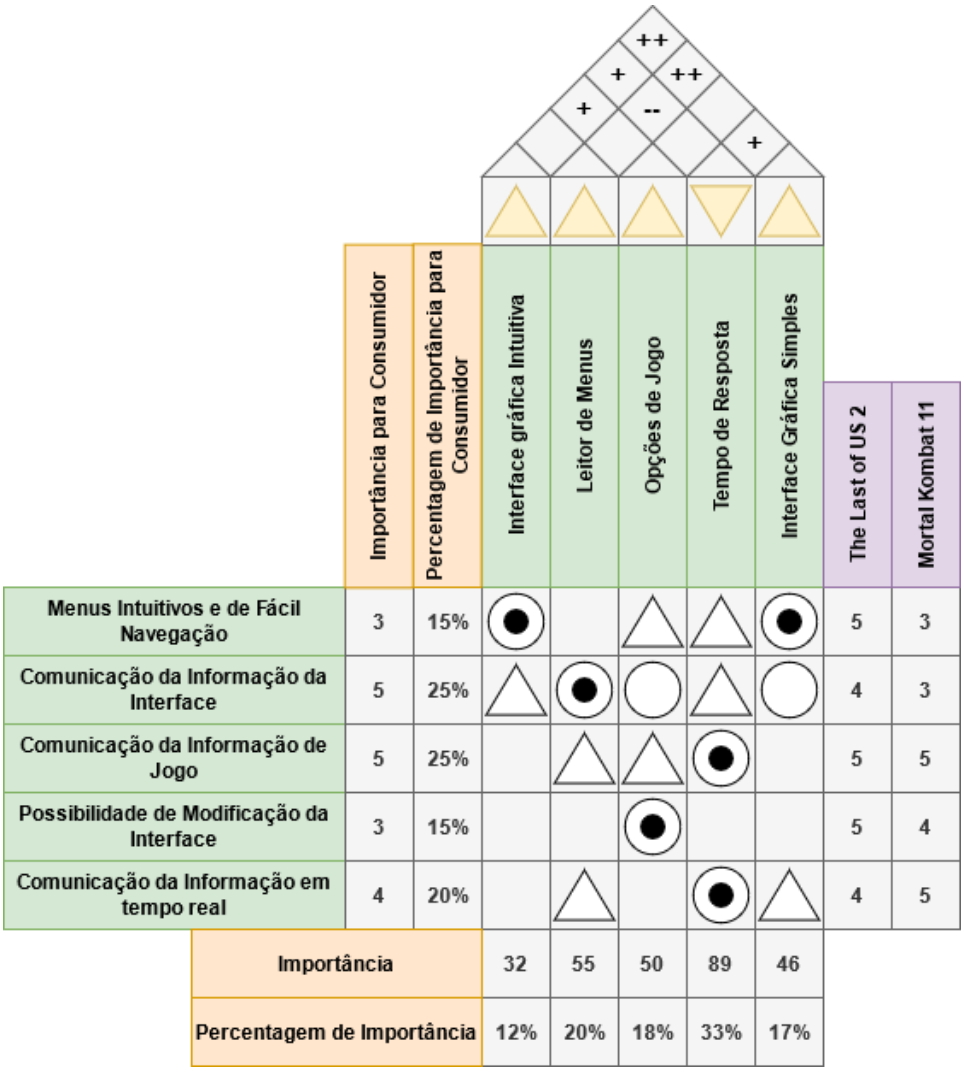
Figura 4.6: Matriz de Relação.

Matriz de Correlação	
+ +	Fortemente Positiva
+	Positiva
-	Negativa
- -	Fortemente Negativa
	Sem Correlação

Figura 4.7: Matriz de Correlação.

De acordo com a análise da figura 4.8 é possível retirar as seguintes ilações:

- O Tempo de Resposta é a característica de engenharia que apresenta maior grau de importância.
- A Interface Gráfica Intuitiva é a característica de engenharia que apresenta o menor grau de importância.
- As característica de engenharia Leitor de Menu e Tempo de Resposta possuem uma correlação fortemente negativa.
- As característica de engenharia Interface Gráfica Intuitiva e Interface Gráfica Simples possuem uma correlação fortemente positiva.
- As característica de engenharia Leitor de Menus e Interface Gráfica Simples possuem uma correlação fortemente positiva.
- O **The Last of Us 2** possui ótimos resultados em todas as necessidades dos utilizadores. Já o **Mortal Kombat 11** falha principalmente na usabilidade de interface e na sua comunicação aos jogadores.



	Importância para Consumidor	Percentagem de Importância para Consumidor	Interface gráfica Intuitiva	Leitor de Menus	Opções de Jogo	Tempo de Resposta	Interface Gráfica Simples	The Last of US 2	Mortal Kombat 11
Menus Intuitivos e de Fácil Navegação	3	15%	●		△	△	◉	5	3
Comunicação da Informação da Interface	5	25%	△	◉	○	△	○	4	3
Comunicação da Informação de Jogo	5	25%		△	△	◉		5	5
Possibilidade de Modificação da Interface	3	15%			◉			5	4
Comunicação da Informação em tempo real	4	20%		△		◉	△	4	5
Importância			32	55	50	89	46		
Percentagem de Importância			12%	20%	18%	33%	17%		

Figura 4.8: Quality Function Deployment.

Capítulo 5

Desenho da Solução

Nesta secção são especificadas as mecânicas do jogo, recompensas e a tabela de ações. É também complementado com o fluxograma de jogo e o mapa de navegação da interface. Por fim é feito o *storyboard* de modo a pré-visualizar os diferentes ecrãs de jogo.

5.1 Mecânicas de Jogo

Este é um jogo de casual de ação para computador, onde o jogador tem de destruir inimigos antes que estes o destruam, sem limite de tempo e que só termina quando o jogador é derrotado. É pensado e desenvolvido para que possa ser jogado sem qualquer estímulo visual, estando totalmente adaptado para indivíduos cegos.

As principais mecânicas de jogo são:

- **Alteração de Direção**: O jogador pode alterar a sua direção para cada um dos quatro pontos cardeais - norte, sul, este e oeste. Desta forma apenas pode rodar 90º para cada um dos lados e nunca um valor intermédio. Com estes valores fixos de rotação, possibilita-se um melhor controlo das direções do jogador, permitindo que este se encontre sempre direcionado para um dos quatro principais pontos cardeais. Os inimigos nascem sempre na direção de um destes 4 pontos, dirigindo-se em linha reta para a posição do jogador.
- **Ataque**: O jogador pode atacar, o que envia um ataque instantâneo na direção para o qual este se encontra virado e que atinge o objeto mais próximo deste. Sendo um ataque instantâneo e sem qualquer tipo de atraso existe uma maior sensação do controlo por parte do jogador. Apesar do ataque instantâneo e de forma a impedir múltiplos ataques por parte do jogador, é criado um mecanismo de atraso entre ataques, tendo assim de esperar um determinado valor fixo para voltar a atacar.
- **Perceção de Inimigos**: Para percecionar a direção dos inimigos, o jogador deve estar atento ao som emitido por estes. Assim, e dependendo do volume com que o som chega aos dois ouvidos do jogador, este deve triangular a sua posição no mundo 3D. Por exemplo, caso o volume do som seja mais alto no ouvido direito do que no esquerdo, automaticamente o jogador sabe que o inimigo se encontra posicionado à sua direita, sendo apenas necessário fazer uma rotação de 90º para a direita e atacar o inimigo. No caso oposto, caso o volume do som seja mais elevado no ouvido esquerdo, o jogado apenas necessita de fazer uma rotação de 90º graus à esquerda.

  Nos casos em que o som do inimigo chega ao dois ouvidos do jogador exatamente com o mesmo volume, significa que este se encontra atrás do jogador ou à frente.

O jogador para se adaptar à situação, precisa então de realizar uma rotação de 90º graus para um dos lados, percebendo automaticamente onde se encontra o inimigo. Note-se, caso este realize uma rotação para a direita, se o som ficar com um volume superior no ouvido esquerdo significa que o inimigo se encontra à sua esquerda, se ficar com um volume superior no ouvido direito significa que o inimigo se encontra à sua direita.

- **Destruição de Inimigos**: Para destruir um inimigo, o jogador deve atingi-lo com um dos seu ataques, o que permite a eliminação direta deste. A eliminação de inimigo recompensa o jogador com o total de um ponto.
- **Receção de Dano**: O jogador perde automaticamente um ponto de vida caso não consiga destruir um inimigo antes que este alcance a sua posição. O inimigo é também destruído e nasce automaticamente um novo para ser enfrentado.
- **Morte**: Caso o jogador perca todas as 5 vidas disponíveis no início de cada jogo, morre e a ronda é automaticamente terminada, aparecendo o ecrã de derrota onde o jogador pode visualizar o resultado obtido.
- **Multiplicação de Pontos**: Quando o jogador faz um determinado número de pontos seguidos, entra em cena um multiplicador que permite multiplicar o número de pontos obtidos. Para ganhar pontos e ativar o multiplicador o jogador necessita de duas condições - não pode ser atingido por inimigos e não pode falhar ataques, tendo de atingir inimigos com todos os ataques realizados. Caso falhe qualquer uma destas condições o multiplicador volta a 1 e o resultado é atualizado de acordo com o valor atingido.

  Por exemplo, caso o jogador falhe um ataque com um total acumulado de 11 pontos e um multiplicador de 3, o total de pontos convertido fica a 33 (11 * 3 = 33).

  Existem 5 níveis no multiplicador:

  - **Nível 1**: Nível base e que multiplica os pontos por 1.
  - **Nível 2**: Segundo nível, atingido quando o jogador alcança os 5 pontos e que os multiplica por 2.
  - **Nível 3**: Terceiro nível, atingido quando o jogador alcança o valor de 10 pontos, multiplicando-os por 3.
  - **Nível 4**: Quarto nível, alcançado quando o jogador atinge um total de 20 pontos, multiplicando-os por 4.
  - **Nível 5**: Quinto e último nível, obtido quando o jogador consegue um total de 40 pontos e que os multiplica por 5.

  A partir do quinto nível, todos os pontos feitos pelo jogador são multiplicados por 5 até que falhe um ataque ou seja atingido por um inimigo.

  O mecanismo de multiplicação de pontos é mais uma estratégia utilizada para que o jogador procure atingir todos os seus ataques e ao mesmo tempo não seja atingido, já que a junção destas duas situações permite uma recompensa imediata e elevada.

5.2 Matriz de Recompensas

A última mecânica de jogo referida na secção 5.1, multiplicação de pontos, cria uma estratégia de recompensa e eficiência que pode ser vista na figura 5.1:

O Jogador	
Acerta todos os ataques	Não acerta todos os ataques
Consegue subir os níveis de multiplicador, permitindo a multiplicação dos pontos obtidos	Não consegue subir os níveis de multiplicador, ganhando apenas 1 ponto de cada vez
Recebe muitos pontos rapidamente	Recebe poucos pontos de cada vez

Figura 5.1: Matriz de Recompensas do Multiplicador.

Caso o jogador procure acertar todos os seus ataques, através de um jogo calmo e de perceção auditiva é recompensado através do mecanismo de multiplicação de pontos, que no nível máximo permite ganhar 5 pontos por cada inimigo destruído.

No entanto caso este ataque múltiplas vezes em diversas direções, apesar de ter uma probabilidade de atingir o inimigo mais rapidamente, ganha apenas um ponto por cada inimigo destruído, visto que sempre que falha um inimigo, o multiplicador retorna ao nível inicial. Torna-se assim muito mais eficiente usar o *feedback* auditivo obtido.

5.3 Tabela de Ações

A tabela de ações representa detalhadamente todos os objetos de jogabilidade, assim como as ações que estes realizam e o seu respetivo gatilho. Cada uma destas ações levam a um determinado resultado de jogo e são praticamente todas acionadas pelo jogador através das suas decisões, tal como se pode analisar na figura 5.2:

ID	Gatilho	Objeto	Ação	Resultado
1	Evento Inicial	Inimigo	Nasce	Um novo inimigo nasce numa das direções
2	Jogador	Inimigo	Jogador ataca Inimigo	O jogador ganha um ponto O inimigo é destruído Nasce um novo inimigo numa direção distinta
3	Inimigo	Inimigo	Inimigo Ataca	O jogador perde um ponto de vida O inimigo é destruído Nasce um novo inimigo numa direção distinta O multiplicador volta ao nível inicial
4	Jogador	Inimigo	Jogador ataca diversos inimigo seguidos	O jogador ganha um multiplicador que lhe permite multiplicar os pontos obtidos
5	Jogador	Mapa de Jogo	Jogador altera Direção	Rotação do jogador é alterada representando a direção do jogador
6	Jogador	Mapa de Jogo	Jogador Ataca	O multiplicador volta ao nível inicial
7	Inimigo	Inimigo	Inimigo ataca Jogador com uma vida	O jogador é derrotado A ronda termina O resultado final é apresentado

Figura 5.2: Tabela de Ações.

5.4 Fluxograma de Jogo

Na figura 5.3 temos acesso ao fluxograma associado à jogabilidade de jogo e que representa o conjunto principal de decisões que ocorrem durante uma rotina normal.

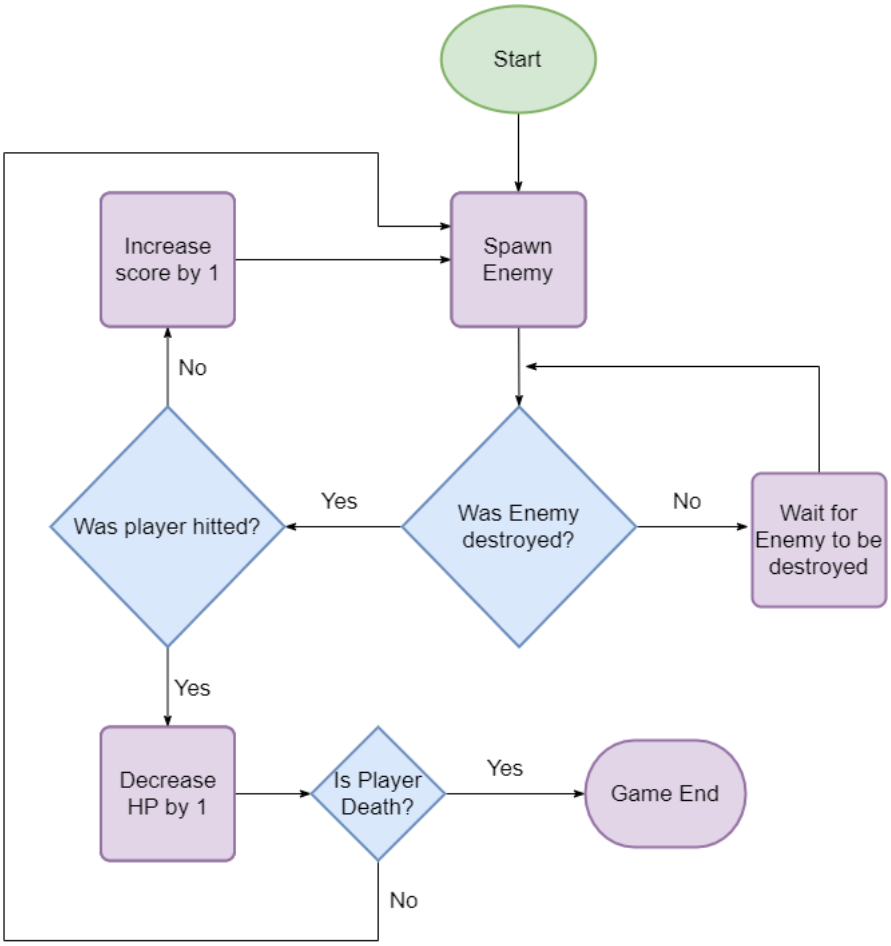


Figura 5.3: Fluxograma de Jogo.

Depois de uma partida começar o fluxo é iniciado, destacando-se em 3 secções distintas:

- Criação de um inimigo que verifica ciclicamente se já foi destruído, o que leva à repetição do processo até que a condição seja verdadeira.
- Quando a condição fica verdadeira e o inimigo se encontra destruído, verifica-se uma nova condição para aferir se o jogador foi atacado. Se o resultado for negativo, então a pontuação do jogador aumenta e mais um inimigo é criado. Caso o jogador tenha sido atacado, então o valor da sua vida é reduzido.
- Por fim e seguindo o fluxo de quando o jogador é atacado, ocorre mais um processo de decisão, onde se verifica se o jogador morreu após a redução de vida. Caso a resposta seja sim o jogo termina, se a resposta for não um novo inimigo é criado.

5.5 Mapa de Navegação

A figura 5.4 representa o mapa de navegação pelos ecrãs de jogo, mostrando todo o circuito de menus e ecrãs pelos quais o jogador pode navegar:

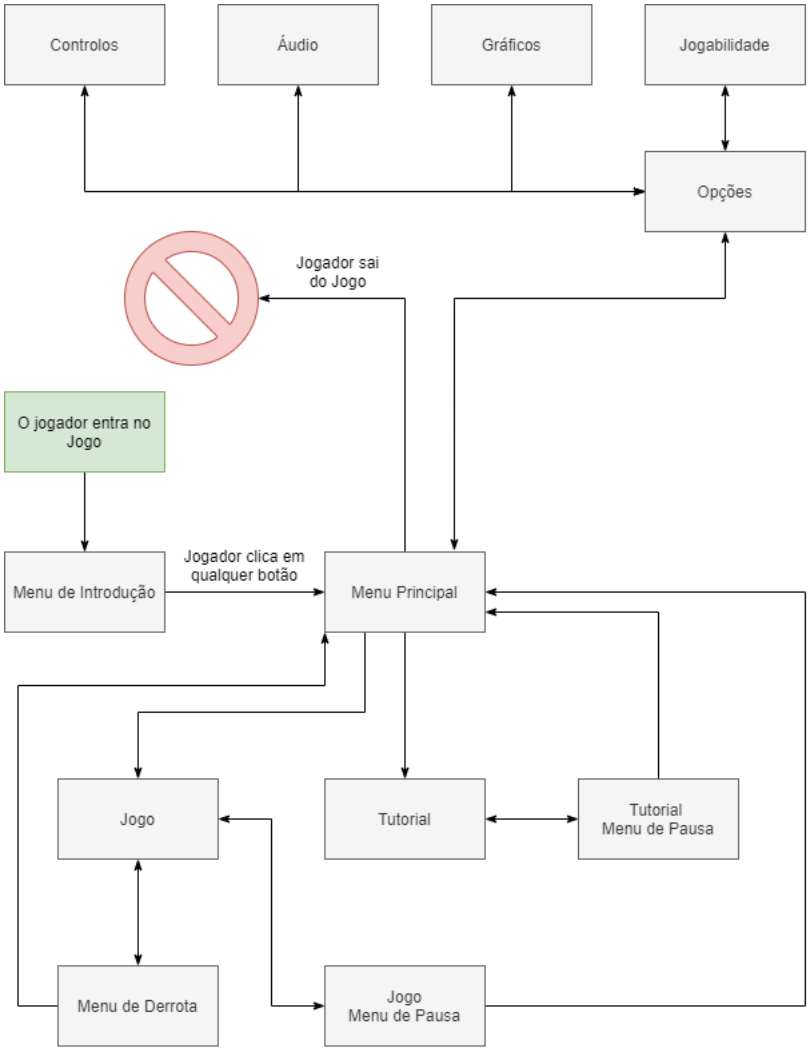


Figura 5.4: Mapa de Navegação de Menus de Jogo.

Quando o jogador abre o jogo, é recebido com um menu de introdução que o direciona para o menu principal, quando qualquer tecla ou botão é pressionado. Do menu principal o jogador tem 4 caminhos distintos de navegação:

- **Jogo**: Caso escolha o caminho do jogo, o fluxo referido na secção anterior é inicializado, sendo possível ao jogador aceder ao menu de pausa e consequentemente ao menu principal. Após a derrota do jogador é também possível aceder ao menu de derrota, que permite a navegação para um novo jogo ou para o menu principal.
- **Tutorial**: Caso o jogador clique no Tutorial é inicializado o processo associado. No tutorial o jogador tem acesso ao menu de pausa, que pode utilizar para retornar ao tutorial ou ao menu principal.

- **Opções**: Caso clique no menu das opções, ficam disponíveis 4 menus diferentes com 4 grupos de funcionalidades distintas - os controlos de jogo, o áudio, os gráficos e a jogabilidade. O jogador pode navegar entres estes e o menu de opções.
- **Sair**: Caso o jogador clique em sair a aplicação é encerrada e este sai do jogo.

5.6 Storyboard

A *storyboard* permite uma análise e visualização gráfica dos menus referidos na secção anterior através de ilustrações que fornecem uma pré-visualização do resultado final pretendido.

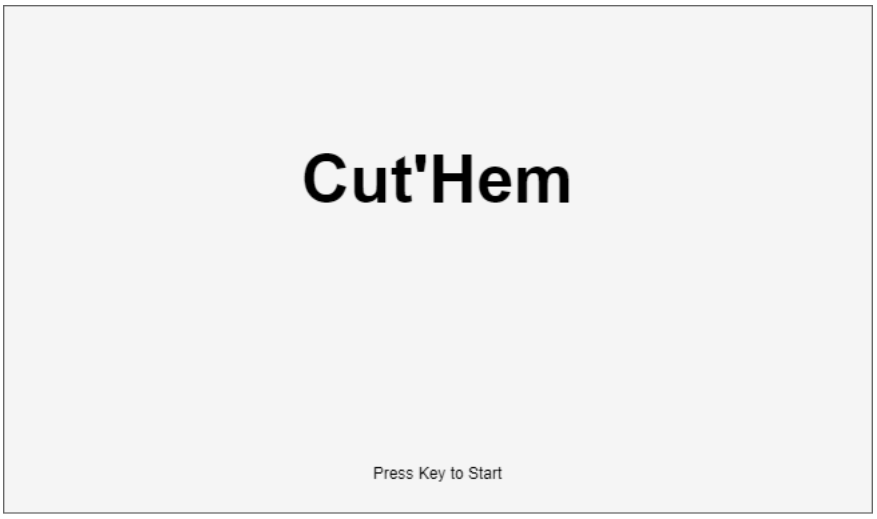


Figura 5.5: Storyboard - Menu de Introdução

Na figura 5.5, é possível ver o ecrã de introdução. Este é um ecrã que apenas pode ser visto quando se inicializa o jogo e que espera o *input* do jogador para continuar na direção do menu principal.

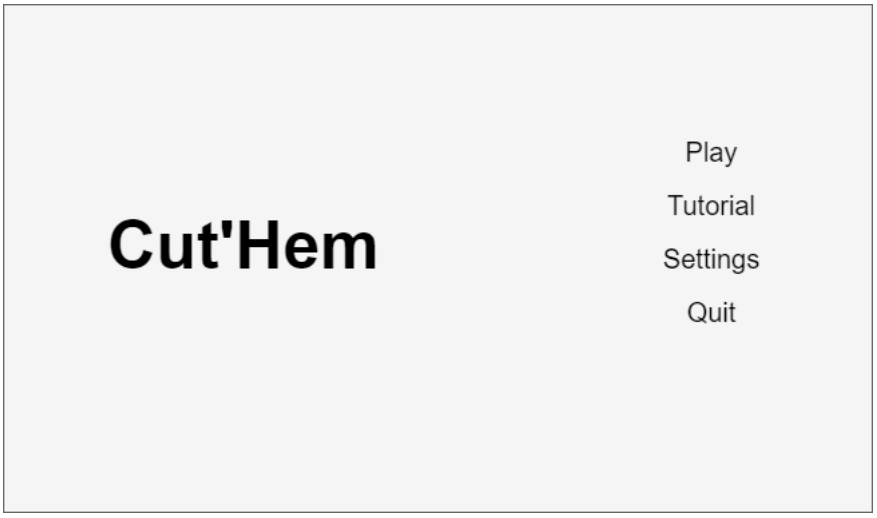


Figura 5.6: Storyboard - Menu Principal

Na figura 5.6 é possível visualizar o menu principal. Possui 4 botões distintos para aceder a outras partes do jogo: o jogo onde se enfrentam os inimigos, o tutorial para a explicação

de mecânicas, o menu de opções onde as definições de acessibilidade são configuradas e o sair para terminar a execução da aplicação.

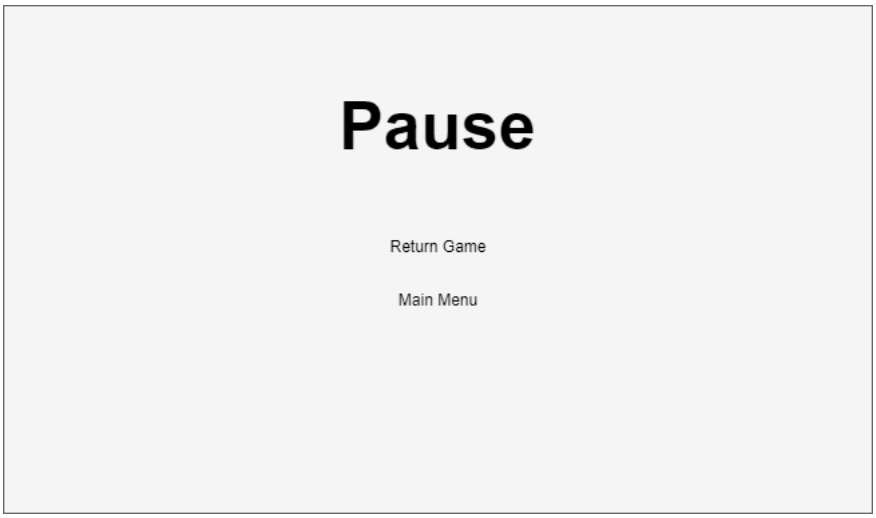


Figura 5.7: Storyboard - Menu de Pausa

A figura 5.7 representa o menu de pausa do jogo e do tutorial e apenas apresenta dois botões: o primeiro permite a retorna ao jogo/tutorial e o segundo volta ao menu principal de jogo.

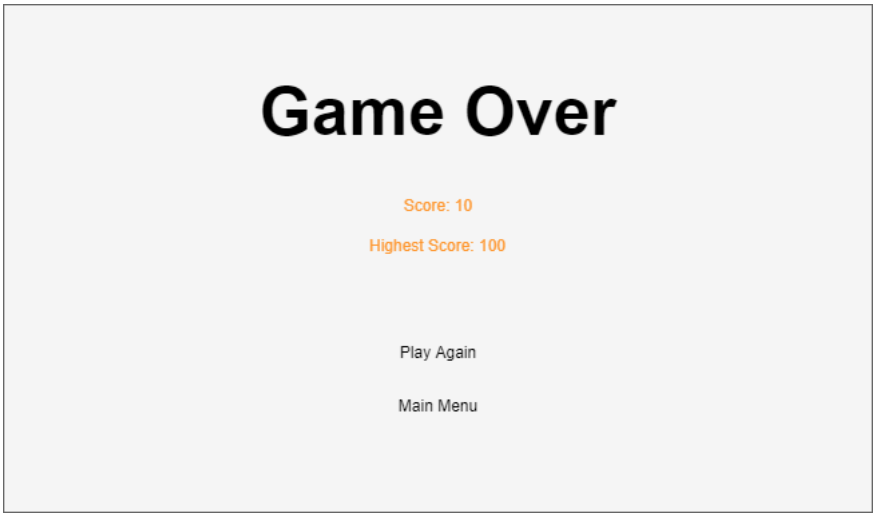


Figura 5.8: Storyboard - Menu de Final de Jogo

A figura 5.8 representa o menu de final de jogo/derrota, apresentando 4 elementos de interface principais:

- **Pontuação**: Este elemento permite ver a pontuação obtida pelo jogador depois de uma partida.
- **Pontuação mais Alta**: Este elemento permite ver a pontuação mais alta obtida pelo jogador para a dificuldade atualmente selecionada.
- **Jogar de Novo**: Permite ao jogador começar uma nova partida.
- **Menu Principal**: Permite o retorno para o menu principal de jogo.

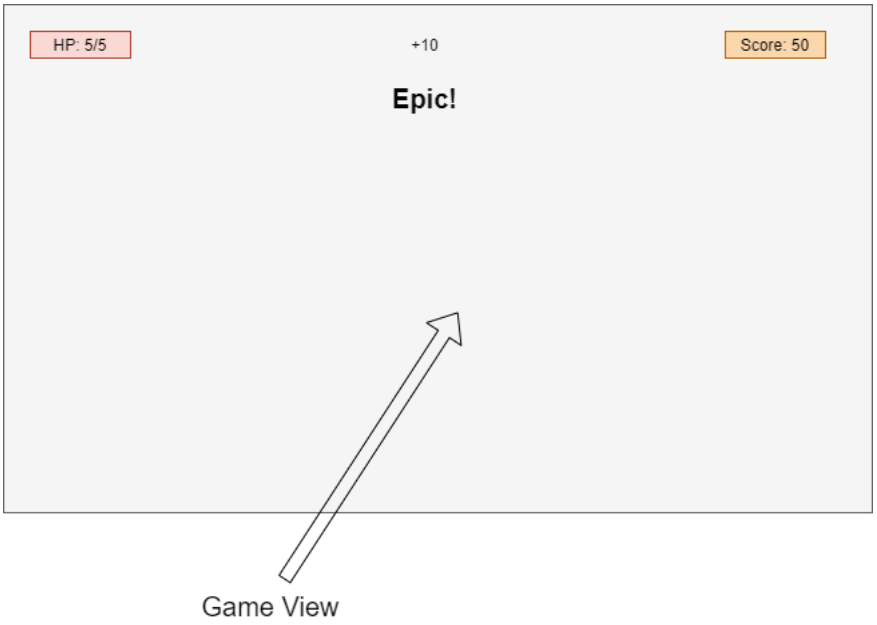


Figura 5.9: Storyboard - Menu de Jogo

A figura 5.9 representa a interface fornecida durante um jogo, que possui os seguintes elementos:

- A vida atual e máxima do jogador.
- A pontuação atual do jogador.
- A pontuação obtida em sequência.
- O multiplicador ativo.

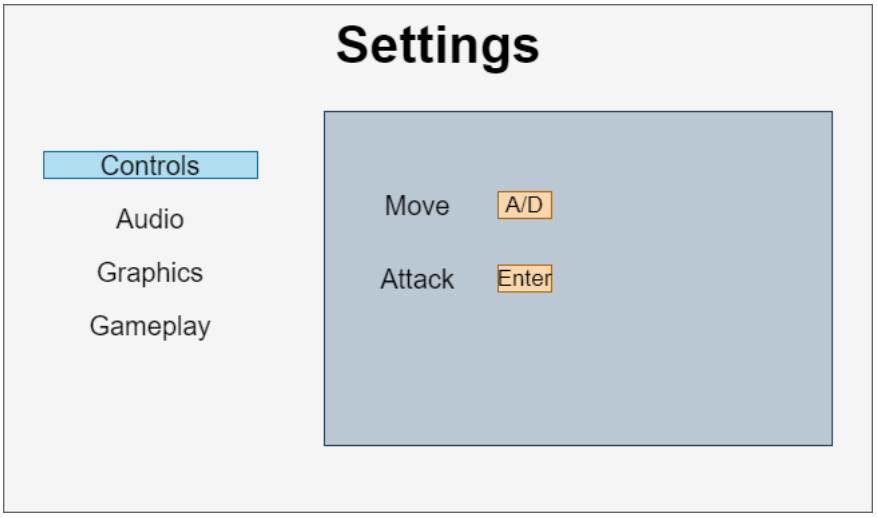


Figura 5.10: Storyboard - Menu de Definições (Controlos)

A figura 5.10 representa o menu de controlos nas opções. Este menu permite alterar o tipos de controlos utilizados para nas ações de jogo. Desta forma existem as ações (movimentação e ataque) e os respetivos botões associados a estas, que o jogador pode modificar.

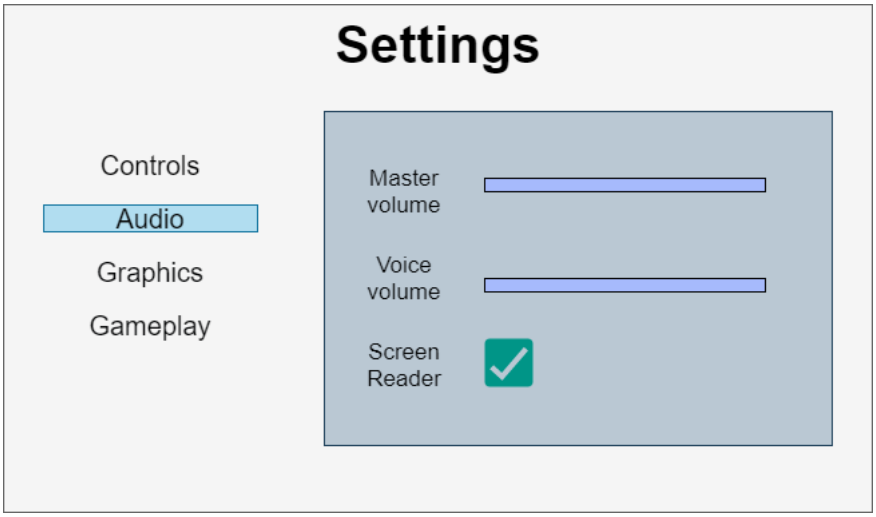


Figura 5.11: Storyboard - Menu de Definições (Áudio)

A figura 5.11 representa o menu de áudio nas opções. Este menu permite alterar o volume de diferentes tipos de áudio individualmente, sendo assim possível ao jogador modificar o volume global de jogo ou apenas o volume de voz. Neste menu é também possível ligar e desligar o leitor de menus existente.

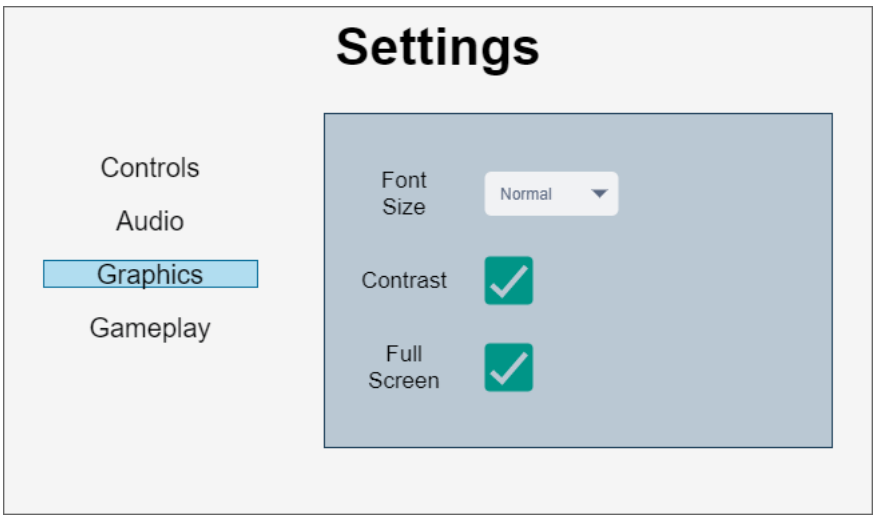


Figura 5.12: Storyboard - Menu de Definições (Gráficos)

A figura 5.12 representa o menu de gráficos nas opções. Neste menu o jogador pode fazer modificações ao espeto gráfica do jogo, como o tamanho da fonte utilizada, a existência de contraste na interface ou até mesmo a ativação de ecrã completo.

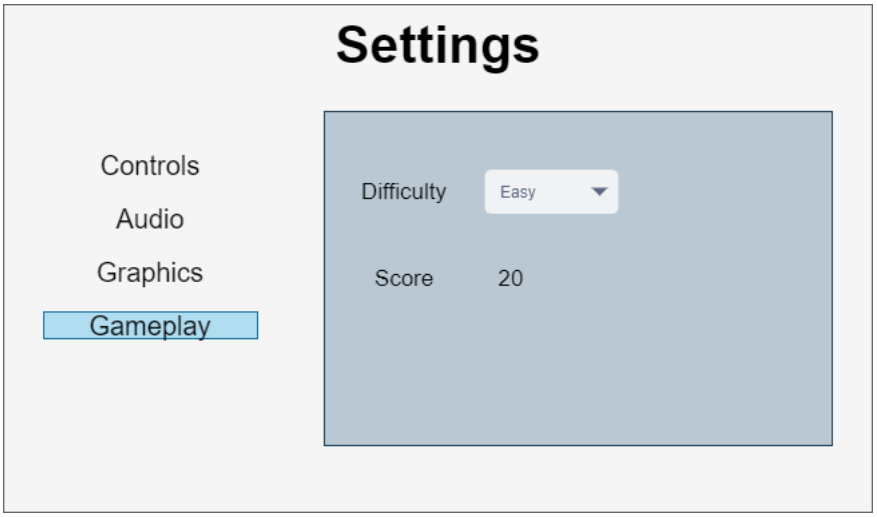


Figura 5.13: Storyboard - Menu de Definições (Gameplay)

A figura 5.13 representa o menu de jogabilidade nas opções. Neste menu o jogador pode alterar o nível de dificuldade do jogo, assim como ver a pontuação mais alta em cada um destes níveis de dificuldade.

Capítulo 6

Implementação da Solução

Neste capítulo são apresentados os detalhes relacionados com a implementação da solução definida de forma a atingir os objetivos criados. É, primeiramente, discutido para cada caso de uso referido na secção 3.2.1 a descrição da implementação, através da apresentação visual do *Unity* com o respetivo código criado em C#. Por fim é feita uma análise da implementação do sistema de voz no jogo.

6.1 Implementação dos Casos de Uso

Nesta secção são apresentados os 7 casos de uso definidos, sendo inicialmente apresentada a sua implementação visual no *Unity*, seguido do código associado no C#.

6.1.1 UC01 - Alterar Contraste da Interface

Para a implementação deste caso de uso foi necessário a criação de um *GameObject* do tipo *Toggle*, que permite ao jogador marcar e desmarcar uma seleção. Este objeto foi adicionado ao painel dos elementos gráficos no menu de opções. Tal como se pode verificar na figura 6.1, sempre que o valor deste *toggle* é alterado um evento chama o método "SetUIConstrast".

Figura 6.1: Implementação da interface para definição de Contraste.

Este método, que pode ser analisado na figura 6.2, recebe um *bool* com o valor atual da seleção, significando isto que quando é verdadeiro a opção de contraste está selecionada. De acordo com o valor recebido, o método vai alterar a cor de *background* do menu de opções e principal. No final o valor do contraste é guardado numa variável dos *PlayerPrefs* para que a mudança possa ser persistida entre sessões e as diferentes cenas de jogo.

```
/// <summary>
/// Set UI Contrast checkbox value
/// </summary>
/// <param name="isContrast">bool</param>
0 referências
public void SetUIConstrast(bool isContrast)
{
    if (graphicsPanel.activeSelf)
    {
        SetUIConstrastChanges(isContrast);

        if (isContrast)
        {
            AudioManager.instance.PlayAudio(AudioType.Graphics_On, AudioChannel.Voice);
        }
        else
        {
            AudioManager.instance.PlayAudio(AudioType.Graphics_Off, AudioChannel.Voice);
        }
    }
}

/// <summary>
/// Update the UI Contrast values in the different panels
/// </summary>
/// <param name="isContrast">bool</param>
2 referências
private void SetUIConstrastChanges(bool isContrast)
{
    //If isContrast variable is set to true, change menus background color to black
    if (isContrast)
    {
        mainMenuPanel.GetComponent<Image>().color = Color.black;
        optionsMenuPanel.GetComponent<Image>().color = Color.black;
    }
    else
    {
        //Color 487EB0 (Kind of Blue)
        mainMenuPanel.GetComponent<Image>().color = new Color(0.282353f, 0.4941176f, 0.6901961f, 1);
        optionsMenuPanel.GetComponent<Image>().color = new Color(0.282353f, 0.4941176f, 0.6901961f, 1);
    }

    int isConstrastValue = isContrast ? 1 : 0;
    PlayerPrefs.SetInt("UIContrast", isConstrastValue);
    PlayerPrefs.Save();
}
```

Figura 6.2: Implementação do código necessário para alteração da definição de Contraste.

6.1.2 UC02 - Alterar Tamanho das Fontes

Na implementação deste caso de uso foram criados múltiplos *GameObjects* do tipo *Text* e *Button* unidos por um objeto superior comum. Quando o jogador pressiona o botão da esquerda, o método "LeftSelected"é chamado através de um evento, caso clique no botão da direita, o método "RightSelected"é o que passa a ser chamado, tal como se pode ver na figura 6.3.

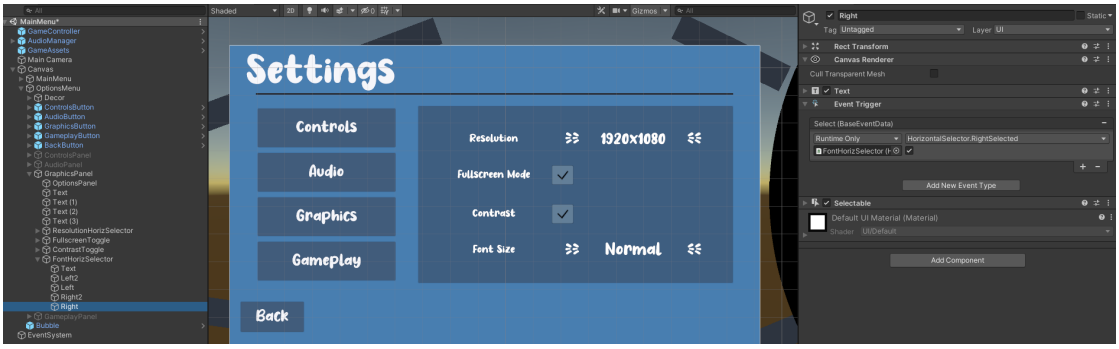


Figura 6.3: Implementação da interface para definição de Fontes.

O método, que pode ser analisado em profundidade na figura 6.4 obtêm o *index* selecionado, de acordo com o tipo de *SelectorType* e chama o método "SetFontSize".

```
/// <summary>
/// When clicked on the right increment on value on the index
/// Returning to zero when arriving the last array element
/// </summary>
/// <param name="firstSelectable">bool to do internal UI updated</param>
0 referências
public void RightSelected(bool firstSelectable)
{
    if (firstSelectable)
        rightSelectable.gameObject.GetComponent<Text>().enabled = true;
    else
        rightSelectable.gameObject.GetComponent<Text>().enabled = false;

    if (index == information.Count - 1)
        index = 0;
    else
        index++;

    text.text = information[index];
    SelectorFunctions();
}

/// <summary>
/// Update different settings based on the actual selector type
/// </summary>
2 referências
private void SelectorFunctions()
{
    switch (selectorType)
    {
        case SelectorType.Resolution:
            mainMenuController.SetScreenResolution(index);
            break;
        case SelectorType.Font:
            mainMenuController.SetFontSize(index);
            break;
        case SelectorType.Difficulty:
            mainMenuController.SetGameDifficulty(index);
            break;
    }
}
```

Figura 6.4: Implementação do código para seleção do índice de Fonte.

Este método que se encontra na figura 6.5 obtêm o valor do *index* enviado, transformando-o num *bool*: caso o *index* seja igual a 0, as fontes usam o tamanho base, de outra forma usam o tamanho máximo possível. De seguida o método atualiza todos os *GameObjects* do tipo *Text* com a nova configuração e guarda os valores nos *PlayerPrefs*.

```
/// <summary>
/// Update the Font Size index value
/// </summary>
/// <param name="fontSizeIndex">int</param>
1 referência
public void SetFontSize(int fontSizeIndex)
{
    if (graphicsPanel.activeSelf)
    {
        SetFontSizeChanges(fontSizeIndex);

        switch (fontSizeIndex)
        {
            case 0:
                AudioManager.instance.PlayAudio(AudioType.Graphics_Normal, AudioChannel.Voice);
                break;
            case 1:
                AudioManager.instance.PlayAudio(AudioType.Graphics_Big, AudioChannel.Voice);
                break;
        }
    }
}

/// <summary>
/// Update the font size values in the scene text labels
/// </summary>
/// <param name="fontSizeIndex">font size index int</param>
2 referências
private void SetFontSizeChanges(int fontSizeIndex)
{
    //If fontSizeIndex is equal to 0, font size value is set normal
    bool isBestFit = fontSizeIndex == 0 ? false : true;

    foreach (Text someLabel in textLabels)
    {
        someLabel.resizeTextForBestFit = isBestFit;
    }

    PlayerPrefs.SetInt("FontSizeIndex", fontSizeIndex);
    PlayerPrefs.Save();
}
```

Figura 6.5: Implementação do código para a atualização da Interface Textual existente.

6.1.3 UC03 - Alterar Dificuldade de Jogo

Na implementação deste caso de uso foram criados múltiplos *GameObjects* para formar uma estrutura semelhante à estrutura criada no caso de uso anterior. Essa estrutura pode ser analisada na figura 6.6 e chama os mesmo métodos do caso de uso anterior, através da classe comum "HorizontalSelector".

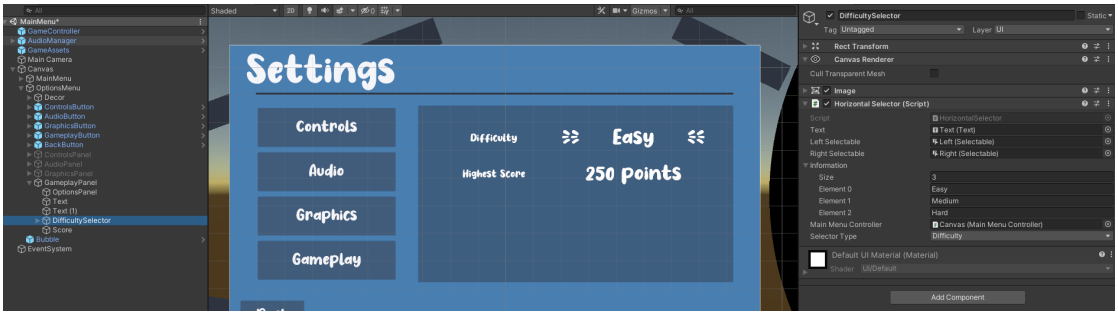


Figura 6.6: Implementação da interface para definição de Dificuldades.

Tal como se pode analisar na figura 6.4, a classe comum chama o método "SetGameDifficulty"que pode ser visualizado na figura 6.7. Este método divide-se em dois processos distintos. No primeiro processo o método atualiza a interface com a pontuação mais alta, de acordo com a dificuldade selecionada. De seguida guarda a dificuldade para que seja persistida nos *PlayerPrefs*.

```
/// <summary>
/// Update Game difficulty Index value
/// </summary>
/// <param name="gameDifficulty">int</param>
1 referência
public void SetGameDifficulty(int gameDifficulty)
{
    if (gameplayPanel.activeSelf)
    {
        SetGameDifficultyChanges(gameDifficulty);

        switch (gameDifficulty)
        {
            case 0:
                AudioManager.instance.PlayAudio(AudioType.Gameplay_Easy, AudioChannel.Voice);
                break;
            case 1:
                AudioManager.instance.PlayAudio(AudioType.Gameplay_Medium, AudioChannel.Voice);
                break;
            case 2:
                AudioManager.instance.PlayAudio(AudioType.Gameplay_Hard, AudioChannel.Voice);
                break;
        }
    }
}

/// <summary>
/// Update Gameplayed Highest score based on the game difficulty
/// </summary>
/// <param name="gameDifficulty">index associated with game difficulty</param>
2 referências
private void SetGameDifficultyChanges(int gameDifficulty)
{
    int highestScoreVal = 0;
    switch (gameDifficulty)
    {
        case 0:
            highestScoreVal = PlayerPrefs.GetInt("HighestScore-Easy");
            break;
        case 1:
            highestScoreVal = PlayerPrefs.GetInt("HighestScore-Medium");
            break;
        case 2:
            highestScoreVal = PlayerPrefs.GetInt("HighestScore-Hard");
            break;
    }
    highestScore.text = highestScoreVal + " points";

    PlayerPrefs.SetInt("DifficultyLevel", gameDifficulty);
    PlayerPrefs.Save();
}
```

Figura 6.7: Implementação do código para a atualização da dificuldade atual.

6.1.4 UC04 - Alterar Definições Individuais de Áudio

Para a implementação deste caso de uso foi necessário a criação de um *GameObject* do tipo *Slider*, que permite ao jogador utilizar um *slider* para escolher valores entre dois limites. Este objeto foi adicionado ao painel dos elementos áudio no menu de opções. Tal como se pode verificar na figura 6.8, sempre que o valor deste *slider* é alterado um evento chama o método "SetMasterVolume".

Figura 6.8: Implementação da interface para definição de Volumes.

O método apresentado na figura 6.9, realiza a atualização de volume chamando o método "SetVolume", representado na figura 6.10, e passando dois parâmetros: o valor do volume não normalizado e o *AudioChannel* (*Master*, *Sfx*, *Audio*).

```
/// <summary>
/// Set Master volume
/// </summary>
/// <param name="value">float volume</param>
0 referências
public void SetMasterVolume(float value)
{
    if (audioPanel.activeSelf)
    {
        AudioManager.instance.StopAudio(AudioType.Audio_Master, AudioChannel.Voice);
        StopVoiceCoroutines();
        AudioManager.instance.SetVolume(value, AudioChannel.Master);
        if (AudioManager.instance.readScreen == 1)
            StartCoroutine(AudioManager.instance.PlayVoiceWithScore(null, (int)value * 10, false));
    }
}

/// <summary>
/// Set Voice volume
/// </summary>
/// <param name="value">float volume</param>
0 referências
public void SetVoiceVolume(float value)
{
    if (audioPanel.activeSelf)
    {
        AudioManager.instance.StopAudio(AudioType.Audio_Voice, AudioChannel.Voice);
        StopVoiceCoroutines();
        AudioManager.instance.SetVolume(value, AudioChannel.Voice);
        if (AudioManager.instance.readScreen == 1)
            StartCoroutine(AudioManager.instance.PlayVoiceWithScore(null, (int)value * 10, false));
    }
}
```

Figura 6.9: Implementação do código para a atualização de volume personalizado.

Este método atualiza as variáveis de volume por canal dentro do controlador de áudio, primeiro normalizando-as através de uma divisão por 10 e guardando o valor obtido para ser persistido nos *PlayerPrefs*.

```
/// <summary>
/// Change the actual volume for a specific audio channel
/// </summary>
/// <param name="volume">float with the volume</param>
/// <param name="audioChannel">AudioChannel</param>
2 referências
public void SetVolume(float volume, AudioChannel audioChannel)
{
    switch (audioChannel)
    {
        case AudioChannel.Master:
            masterVolumePercent = volume / 10;
            PlayerPrefs.SetFloat("MasterVolume", masterVolumePercent);
            break;
        case AudioChannel.Voice:
            voiceVolumePercent = volume / 10;
            PlayerPrefs.SetFloat("VoiceVolume", voiceVolumePercent);
            break;
        case AudioChannel.Sfx:
            sfxVolumePercent = volume / 10;
            PlayerPrefs.SetFloat("SfxVolume", sfxVolumePercent);
            break;
    }

    PlayerPrefs.Save();
}
```

Figura 6.10: Implementação do código para a atualização de volume no controlador de áudio.

6.1.5 UC05 - Controlar Leitor de Menus

Para a implementação deste caso de uso foi necessário a criação de um *GameObject* do tipo *Toggle*, que permite ao jogador marcar e desmarcar uma seleção. Este objeto foi adicionado ao painel dos elementos de áudio no menu de opções. Tal como se pode verificar na figura 6.11, sempre que o valor deste *toggle* é alterado um evento chama o método "SetScreenReader".

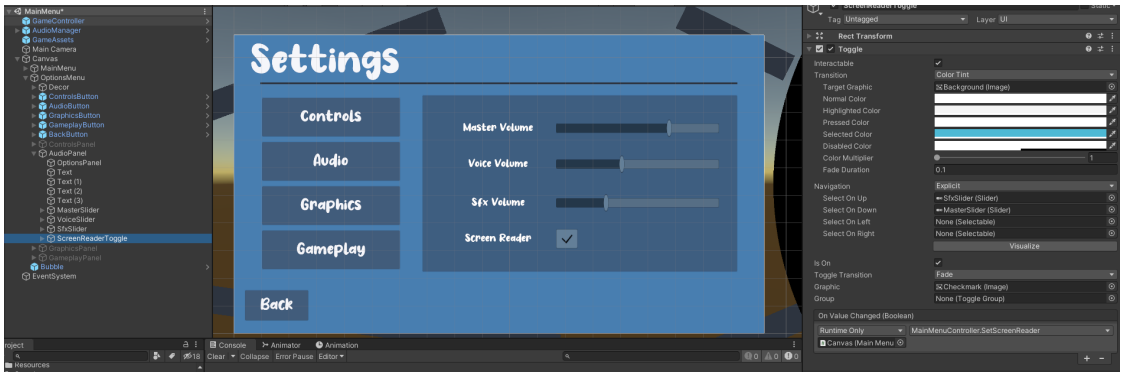


Figura 6.11: Implementação da interface para definição do Screen Reader.

Este método, representado na figura 6.12, recebe um valor *bool* do *toggle*, indicando se este se encontra selecionado ou não, que por sua vez chama o método "SetScreenReader"do *AudioManager* para atualizar as suas variáveis internas.

```
/// <summary>
/// Set Screen reader checkbox
/// </summary>
/// <param name="value">screen reader check</param>
0 referências
public void SetScreenReader(bool value)
{
    if (audioPanel.activeSelf)
    {
        AudioManager.instance.SetScreenReader(value);
        if (value)
            AudioManager.instance.PlayAudio(AudioType.Graphics_On, AudioChannel.Voice);
        else
            AudioManager.instance.PlayAudio(AudioType.Graphics_Off, AudioChannel.Voice);
    }
}
```

Figura 6.12: Implementação do código para a atualização da definição do Leitor de Ecrã.

E tal como provado pela figura 6.13, antes de realizar leituras de ecrã o método verifica se a opção de leitura de menus no *AudioManager* se encontra ativa, saindo do método caso não se encontre.

```
/// <summary>
/// Play Sound based on the type and channel
/// </summary>
/// <param name="_type">AudioType to be played</param>
/// <param name="channel">AudioChannel</param>
30 referências
public void PlayAudio(AudioType _type, AudioChannel channel)
{
    //Return if the read screen option is off
    if (readScreen != 1 && channel == AudioChannel.Voice) return;

    //Update the AudioJobs being played
    AddJob(new AudioJob(AudioAction.START, _type, channel));
}
```

Figura 6.13: Verificação se a variável de leitura de ecrã se encontra ativa.

6.1.6 UC06 - Alterar Configurações dos Controlos

Para a implementação deste caso de uso foi necessário a criação de um *GameObject* do tipo *Button*, que permite ao jogador clicar para realizar uma ação. Este objeto foi adicionado ao painel dos controlos no menu de opções. Tal como se pode verificar na figura 6.14, sempre que este *Button* é pressionado um evento chama o método "StartInteractiveRebind".

Figura 6.14: Implementação da interface para definição dos Controlos.

Tal como se encontra representado na figura 6.15, é chamado o método "PerformInteractiveRebinding"que guarda o controlo selecionado pelo utilizador caso este seja válido.

```
private void PerformInteractiveRebind(InputAction action, int bindingIndex, bool allCompositeParts = false)
{
    m_RebindOperation?.Cancel(); // Will null out m_RebindOperation.

    void CleanUp()
    {
        m_RebindOperation?.Dispose();
        m_RebindOperation = null;
    }

    // Configure the rebind.
    m_RebindOperation = action.PerformInteractiveRebinding(bindingIndex)
        .WithControlsExcluding("<Mouse>/leftButton")
        .WithControlsExcluding("<Mouse>/rightButton")
        .WithControlsExcluding("<Mouse>/press")
        .WithControlsExcluding("<Pointer>/position")
        .WithCancelingThrough("*/{Cancel}")
        .OnCancel(
            operation =>
            {
                m_RebindStopEvent?.Invoke(this, operation);
                m_RebindOverlay?.SetActive(false);
                UpdateBindingDisplay();
                CleanUp();
            })
        .OnComplete(
            operation =>
            {
                m_RebindOverlay?.SetActive(false);
                m_RebindStopEvent?.Invoke(this, operation);

                UpdateBindingDisplay();
                CleanUp();
```

Figura 6.15: Implementação do código para definição de novos Controlos.

O excerto de código representado na figura 6.16 mostra a atualização dos *PlayerPrefs* com os novos controlos de utilizador no momento de desativação e a atualização do *InputActionAsset* com os controlos que se encontram persistidos no momento de criação.

```
public class RebindSaveLoad : MonoBehaviour
{
    //Public variables
    public InputActionAsset actions;

    Mensagem do Unity | 0 referências
    public void OnEnable()
    {
        //Get the rebinds from the player prefs and load them in the InputActionAsset
        var rebinds = PlayerPrefs.GetString("Rebinds");
        if (!string.IsNullOrEmpty(rebinds))
            actions.LoadBindingOverridesFromJson(rebinds);
    }

    Mensagem do Unity | 0 referências
    public void OnDisable()
    {
        //Get the associated bindings and save them on the player prefs
        var rebinds = actions.SaveBindingOverridesAsJson();
        PlayerPrefs.SetString("Rebinds", rebinds);
    }
}
```

Figura 6.16: Implementação do código para guardar e carregar controlos.

6.1.7 UC07 - Controlar Estados de Jogo

Para a implementação deste caso de uso foi necessário a adição da *Input Controls Action* correta ao "Cancel"do *Input System UI Input Module* do *Unity*, tal como representado na figura 6.17. Isto permite chamar um evento com contexto, tal como representado na figura 6.18.

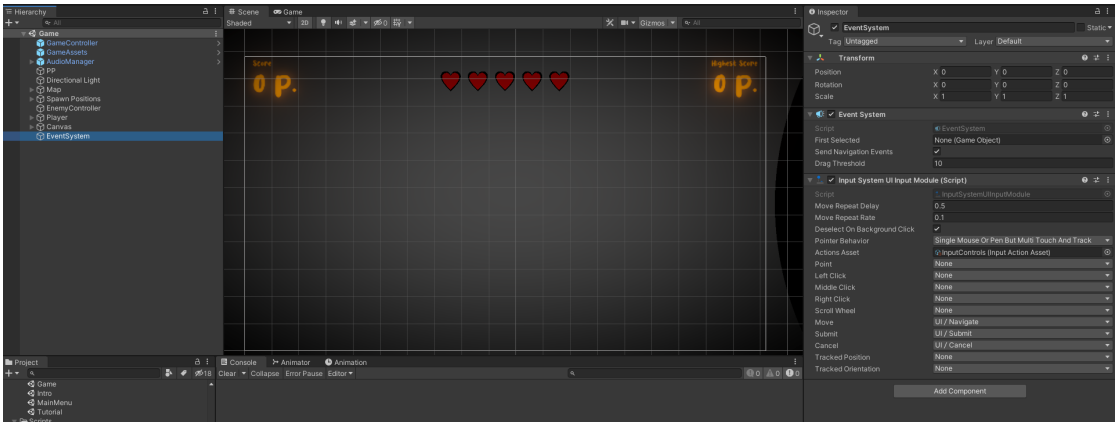


Figura 6.17: Implementação da interface para definição de Pausa.

Este evento é enviado sempre que o jogador clica numa tecla representada como formato de cancelamento de ações. O excerto de código na figura 6.18 verifica se as ações foram completamente executadas, podendo resultar em 3 situações distintas:

- O jogador encontra-se em jogo, o que leva a que este pause e o menu de pausa abra.
- O jogador encontra-se no menu de pausa, o que leva a que o jogo recomece e o menu feche.
- O jogador encontra-se no ecrã de derrota, o que leva com que o jogo abra o menu principal.

```
/// <summary>
/// InputAction UI scheme onCancel action
/// </summary>
/// <param name="context">CallbackContext</param>
0 referências
public void OnCancel(InputAction.CallbackContext context)
{
    //If te action was performed
    if (context.performed)
    {
        //If the game is in the results menu return to the main menu
        if (gameController.actualState == GameState.ResultsMenu)
        {
            StartCoroutine(LeaveToMenu());
        }
        //If it is playing the game, pause the game
        else if (gameController.actualState == GameState.Playing)
        {
            //Update the panels
            gameController.actualState = GameState.PauseMenu;
            gameScreen.SetActive(false);
            pauseScreen.SetActive(true);

            //Change selected game objected to the score selectable
            EventSystem.current.SetSelectedGameObject(null);
            EventSystem.current.SetSelectedGameObject(resumeButton);
        }
        //If the game is paused, return to play
        else if (gameController.actualState == GameState.PauseMenu)
        {
            ReturnToGame();
        }
    }
}
```

Figura 6.18: Implementação do código para fluxo de cancelamento.

6.2 Implementação de Voz

Nesta secção é apresentada a implementação do sistema de voz utilizado no projeto e que possibilita a tradução de todos os menus e textos existentes. Na primeira é apresentado a implementação de voz nos menus de jogo e na segunda parte é explicada a implementação de voz em números.

6.2.1 Voz dos Menus

Para a implementação de vozes para tradução dos diferentes textos e menus para voz, foi criado no *AudioManager* uma lista com todos os tipos de vozes e sons, tendo cada um destes a um *AudioSource* específico associado. Por exemplo, todas os áudios com voz encontram-se associados ao *AudioSource* "VoiceSource". Com este processo tornou-se possível centralizar todas as vozes criadas num só local, como podemos ver na figura 6.19.

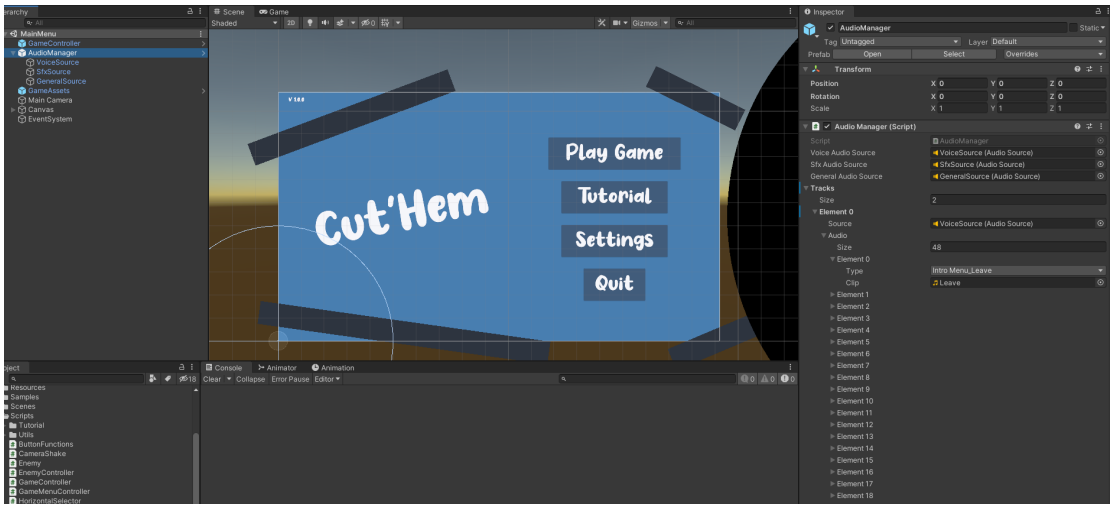


Figura 6.19: Implementação dos diversos áudios numa lista centralizada do *AudioManager*

O código representado na figura 6.20 mostra o processo da leitura de um áudio. Inicialmente, e caso exista, o áudio que se encontra no *AudioSouce* necessário para o novo áudio é removido, de seguida e de acordo com o *AudioJob* enviado é obtido o *clip* de som baseado na lista representada na 6.19. O volume é ajustado de acordo com o tipo de *AudioChannel* associado e por fim e de acordo com a ação enviado é: reproduzido, parado ou recomeçado.

Este processo é muito útil, pois permite que não existam duas ou mais vozes a serem executadas ao mesmo tempo, facilitando o processo auditivo ao jogador. Este consegue assim entender muito mais facilmente todas as leituras de menu.

```
/// <summary>
/// Update the AudioJobs being played
/// </summary>
/// <param name="_job">AudioJob</param>
3 referências
private void AddJob(AudioJob _job)
{
    RemoveConflictingJobs(_job.type);

    IEnumerator _jobRunner = RunAudioJob(_job);
    jobTable.Add(_job.type, _jobRunner);
    StartCoroutine(_jobRunner);
}

/// <summary>
/// Run audio job to be updated
/// </summary>
/// <param name="_job">AudioJob</param>
/// <returns>IEnumerator</returns>
1 referência
private IEnumerator RunAudioJob(AudioJob _job)
{
    //Obtain the AudioTrack based on the job type
    AudioTrack _track = (AudioTrack)audioTable[_job.type];
    _track.source.clip = GetAudioClipFromTrack(_job.type, _track);

    //Update the track volume based on the audioChannel
    switch (_job.channel)
    {
        case AudioChannel.Master:
            _track.source.volume = 0.05f * masterVolumePercent;
            break;
        case AudioChannel.Voice:
            _track.source.volume = 0.5f * voiceVolumePercent * masterVolumePercent;
            break;
    }

    //Swicth case depending on the audio job action
    switch (_job.action)
    {
        case AudioAction.START:
            _track.source.Play();
            break;
        case AudioAction.STOP:
            _track.source.Stop();
            break;
        case AudioAction.RESTART:
            _track.source.Stop();
            _track.source.Play();
            break;
    }

    //Remove job type form the hash table
    jobTable.Remove(_job.type);

    yield return null;
```

Figura 6.20: Implementação de código para paragem de áudios em conflito

6.2.2 Voz dos Números

Dada a variabilidade de números existentes, torna-se impossível usar o método anterior para a representação em fala de informação que contenha números, como por exemplo, a pontuação do jogador. Por isso foi criado um novo processo que permite a transformação de qualquer número para voz. Este processo transforma os algarismo únicos e número falados em códigos, seguindo a lista criada e representada na figura 6.21.

```
/// <summary>
/// Number Extense            Code
/// - 0                    - Zero            - A0
/// - 1                    - One             - A1
/// - 2                    - Two             - A2
/// - 3                    - Three           - A3
/// - 4                    - Four            - A4
/// - 5                    - Five            - A5
/// - 6                    - Six             - A6
/// - 7                    - Seven           - A7
/// - 8                    - Eight           - A8
/// - 9                    - Nine            - A9
/// - 10                   - Ten             - B0
/// - 11                   - Eleven          - B1
/// - 12                   - Twelve          - B2
/// - 13                   - Thirteen        - B3
/// - 14                   - FourTeen        - B4
/// - 15                   - Fifteen         - B5
/// - 16                   - Sixteen         - B6
/// - 17                   - Seventeen       - B7
/// - 18                   - Eighteen        - B8
/// - 19                   - Nineteen        - B9
/// - 20                   - Twenty          - C2
/// - 30                   - Thirty          - C3
/// - 40                   - Forty           - C4
/// - 50                   - Fifty           - C5
/// - 60                   - Sixty           - C6
/// - 70                   - Seventy         - C7
/// - 80                   - Eighty          - C8
/// - 90                   - Ninety          - C9
/// - 100                  - Hundred         - D0
/// - 1000                 - Thousand        - D1
/// - 1000000              - Million         - D2
/// - 1000000000           - Billion         - D3
/// - 1000000000000        - Trillion        - D4
/// </summary>
```

Figura 6.21: Representação dos números e algarismos falados

O método representado na figura 6.22 recebe o número global dividido de 3 em 3 algarismos, transformando os números de cada grupo em códigos. Para isto faz uso de algumas propriedades como o seu tamanho fixo de 3 e a posição no número global. Por exemplo, caso o índice recebido seja 3, significa que se encontra no grupo dos milhões, logo o último código a ser colocado é o "D2".

E sendo que são grupos de 3 algarismos, o primeiro algarismo representa sempre um valor nas centenas, o segundo as dezenas e o terceiro as unidades. Por exemplo, com o número 325 temos "three; hundred; twenty; five", que com a nossa lista de códigos se transforma em "A3";"D0";"C2";"A5".

Aplicando esta lógica iterativamente, torna-se possível ler qualquer número desejado.

```
/*
 * Based on the chunk size and value a different code will be generated:
 * - 0 value is silent, so no code will be generated
 * - Decimal values will have a special code based on the second digit, number between 10 and 19 have special denominations,
 * the others will change with 10's increments
 * - Hundred values will be the digit value (1,2,3,4..) with "Hundred" code
 * - Units will be just the digit number
 */
char[] trimElements = element.Trim().ToCharArray();
for (int i = 0; i < trimElements.Length; i++)
{
    if (trimElements.Length == 3)
    {
        if (i == 0 && trimElements[i] != '0')
            result += "A" + trimElements[i] + ";" + "D0;";
        else if (i == 1 && trimElements[i] == '1')
            result += "B" + trimElements[i + 1] + ";";
        else if (i == 1 && trimElements[i] != '0')
            result += "C" + trimElements[i] + ";";
        else if (i == 2 && trimElements[i - 1] != '1' && trimElements[i] != '0')
            result += "A" + trimElements[i] + ";";
    }
    else if (trimElements.Length == 2)
    {
        if (i == 0 && trimElements[i] == '1')
            result += "B" + trimElements[i + 1] + ";";
        else if (i == 0 && trimElements[i] != '0')
            result += "C" + trimElements[i] + ";";
        else if (i == 1 && trimElements[i - 1] != '1' && trimElements[i] != '0')
            result += "A" + trimElements[i] + ";";
    }
    else if (trimElements.Length == 1)
    {
        if (i == 0 && trimElements[i] != '0')
            result += "A" + trimElements[i] + ";";
    }
}

//Based on the chunk associated index, a different code will be generated (Billion, Million, Thousand...)
switch (index)
{
    case 2:
        result += "D1;";
        break;
    case 3:
        result += "D2;";
        break;
    case 4:
        result += "D3;";
        break;
    case 5:
        result += "D4;";
        break;
    default:
        break;
```

Figura 6.22: Transformação para códigos

O método representado na figura 6.23 apenas necessita de ler o código criado, separando os diversos elementos com o uso do ";"e associando a letra inicial (A, B, C, D) à lista desejada e o número ao índice necessário, que possui o áudio associado.

```
public IEnumerator PlayVoiceWithScore(AudioClip audio, int number, bool? isPoints = false, bool? isHP = false)
{
    //Check if voice reader is available
    if (readScreen == 1)
    {
        //Set the voice volume
        float volume = 0.5f * voiceVolumePercent * masterVolumePercent;
        //Get the string generated code with the voice association
        string finalCode = NumberCodeGenerator(number);
        //Split the generated string by ';'
        string[] soundCodes = finalCode.TrimEnd(';').Split(';');

        //If initial AudioClip is not null play it and wait for it to finish
        if (audio != null)
        {
            voiceAudioSource.clip = audio;
            voiceAudioSource.volume = volume;
            voiceAudioSource.Play();

            yield return new WaitWhile(() => voiceAudioSource.isPlaying);
        }

        //For each code in the splitted auxiliary array
        foreach (string code in soundCodes)
        {
            char[] auxValues = code.ToCharArray();
            string auxCode = auxValues[0].ToString(); //List Code Type (A,B,C,D)
            int auxCodeNumber = int.Parse(auxValues[1].ToString()); //List Code Number (1,2,3,4...)
            voiceAudioSource.volume = volume;

            //Switch case to define to the correct list AudioClip and number
            switch (auxCode)
            {
                case "A":
                    voiceAudioSource.clip = GameAssets.i.aList[auxCodeNumber];
                    voiceAudioSource.Play();
                    yield return new WaitWhile(() => voiceAudioSource.isPlaying);
                    break;
                case "B":
                    voiceAudioSource.clip = GameAssets.i.bList[auxCodeNumber];
                    voiceAudioSource.Play();
                    yield return new WaitWhile(() => voiceAudioSource.isPlaying);
                    break;
                case "C":
                    voiceAudioSource.clip = GameAssets.i.cList[auxCodeNumber - 2];
                    voiceAudioSource.Play();
                    yield return new WaitWhile(() => voiceAudioSource.isPlaying);
                    break;
                case "D":
                    voiceAudioSource.clip = GameAssets.i.dList[auxCodeNumber];
                    voiceAudioSource.Play();
                    yield return new WaitWhile(() => voiceAudioSource.isPlaying);
                    break;
            }
        }

        //Check if the "Points" voice over needs to be read
        if ((isPoints ?? false) && number != 1)
        {
            voiceAudioSource.clip = GameAssets.i.points;
            voiceAudioSource.volume = volume;
            voiceAudioSource.Play();
        }
```

Figura 6.23: Leitor de códigos

Capítulo 7

Avaliação da Solução

Neste capítulo é feita a descrição das experiências e avaliação realizada à solução implementada. Inicialmente são definidas as grandezas e informações a serem utilizadas para a avaliação do trabalho, seguida de uma explicação da metodologia de avaliação utilizada para testar as hipóteses de investigação e os seus respetivos resultados.

7.1 Indicadores e Fontes de Informação

Os indicadores e fontes de informação utilizados são diferentes consoante a dimensão estudada: qualidade e usabilidade.

Para a avaliação de usabilidade são aplicados questionários a participantes cegos e visuais vendados. A partir das respostas dadas nos questionários é possível avaliar de que forma os objetivos de usabilidade definidos no projeto se encontram completos. Para além disso, os questionários são utilizados para responder a alguns dos requisitos definidos no Quantitative Evaluation Framework (QEF).

Para a avaliação qualidade é utilizada uma autoavaliação do autor da dissertação, classificando cada um dos objetivos definidos de acordo com o cumprimento das métricas definidas através do QEF.

7.2 Metodologia de Avaliação

Para as metodologias de avaliação são utilizadas duas metodologias: **Avaliação de Qualidade** com o QEF e a **Avaliação de Usabilidade** com questionários a utilizadores.

A forma de avaliação utilizada nos questionários classifica as respostas dadas em relação à usabilidade do jogo, em 5 níveis diferentes:

- **Nível 1**: Discordo Totalmente.
- **Nível 2**: Discordo.
- **Nível 3**: Sem Opinião / Indiferente.
- **Nível 4**: Concordo.
- **Nível 5**: Concordo Totalmente.

Relativamente ao inquérito, foram criadas 10 questões relacionadas com o projeto, considerando que as questões de número par tem um sentido negativo e as questões de número ímpar tem um sentido positivo, como se pode analisar na tabela 7.1:

Tabela 7.1: Questionário de Usabilidade.

ID	Questão
1	O texto é simples e conciso?
2	A navegação pela interface de jogo é complexa?
3	As opções de acessibilidade são devidamente assinaladas?
4	É difícil aceder ao menu de opções?
5	É fácil configurar as diferentes opções de acessibilidade?
6	É complexo começar um novo jogo?
7	O tutorial explica as mecânicas base do jogo?
8	A jogabilidade é complexa e inadequada?
9	O jogo fornece corretamente diferentes tipos de feedbacks auditivos ao jogador?
10	O jogo teve uma má performance (Velocidade, erros e compatibilidade)?

A partir das respostas dadas são feitos alguns cálculos para a sua normalização, primeiro numa escala de 0 e 40 e depois através da multiplicação por 2,5 para a escala de pontuação de System Usability Scale (SUS) (Bangor, Kortum e Miller 2008). É também importante durante a normalização ter atenção às perguntas de sentido positivo e negativo. Para isso é necessário seguir um conjunto de critérios:

- **Perguntas Pares**: x - 1, onde x representa o valor obtido no questionário.
- **Perguntas Ímpares**: 5 - x, onde x representa o valor obtido no questionário.

De acordo com esta escala é definido o grau de usabilidade obtido com o projeto, através do sistema de classificação representado na tabela 7.2:

Tabela 7.2: Sistema de Classificação SUS (Adaptado de Bangor (Bangor, Kortum e Miller 2008).).

Escala	Classificação	Definição
>80,3	A	Excelente
68 - 80,3	B	Bom
68	C	Aceitável
51 - 68	D	Pobre
<51	F	Péssimo

A média das respostas dadas são também calculadas individualmente para responder a métricas de avaliação do QEF, correspondendo a percentagem obtida a um valor de cumprimento específico.

Para a metodologia de avaliação qualitativa é utilizado o QEF. A avaliação através deste modelo diz o quão próximo dos objetivos definidos inicialmente se encontra a solução desenhada (Escudeiro e Bidarra 2008).

Esta avaliação é possível de acordo com os requisitos definidos no QEF, que possuem métricas avaliadas de acordo com o seu cumprimento. Estas métricas possuem duas formas de avaliação (Escudeiro e Bidarra 2008):

- **Qualidade**: Métricas que utilizem a qualidade serão avaliadas de acordo com as características técnicas e tecnológicas desenvolvidas. Esta avaliação é feita através de uma escala de percentagens [0, 25, 50, 75, 100], que possui em cada um dos valores definidos uma das características desenvolvidas.

- **Usabilidade**: Métricas que utilizem a usabilidade são avaliadas através de questionários respondidos por diferentes tipos de utilizadores, tal como explicado na secção 7.1.

A partir da escala de percentagens é obtido o valor de cumprimento do requisito e de acordo com o peso de cada um dos requisitos[1] é calculado o valor final da qualidade da solução (Escudeiro e Bidarra 2008).

Cada um dos requisitos definidos para o projeto em questão, encontram-se agrupados por **Dimensões**, tendo sido consideradas as seguintes:

- **Funcionalidade**: Representa os requisitos funcionais e de funcionalidades no projeto.
- **Suportabilidade**: Representa os requisitos de suportabilidade no projeto.
- **Usabilidade**: Representa os requisitos de usabilidade no projeto.

Para a dimensão da **Funcionalidade** foram definidos 3 fatores: **Acessibilidade**, **Jogabilidade** e **Interação com Utilizador**. Estes 3 fatores perfazem um total de 22 requisitos, que podem ser analisados juntamente com a escala de avaliação de requisitos no anexo A.1.1.

Para a dimensão da **Suportabilidade** foram definidos 3 fatores: **Jogabilidade**, **Manutenibilidade** e **Adaptabilidade**. Estes 3 fatores perfazem um total de 8 requisitos, que podem ser analisados juntamente com a escala de avaliação de requisitos no anexo A.1.2.

Para a dimensão da **Usabilidade** foram definidos 3 fatores: **Navegação**, **Qualidade de Conteúdo** e **Integridade**. Estes 3 fatores perfazem um total de 11 requisitos, que podem ser analisados juntamente com a escala de avaliação de requisitos no anexo A.1.3.

7.3 Resultados

Nesta secção são analisados os resultados das metodologias de avaliação de qualidade e usabilidade. É a partir desta análise que é possível perceber se os objetivos traçados e as questões de investigação foram cumpridos.

7.3.1 Usabilidade da Solução

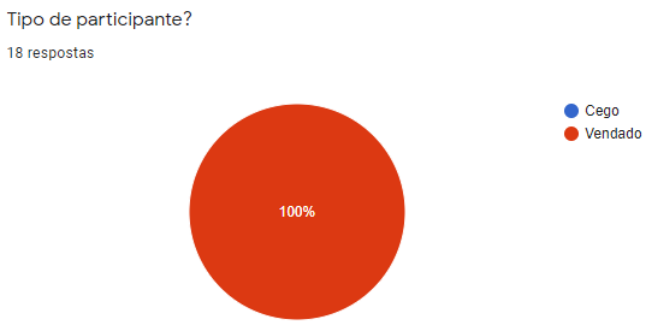


Figura 7.1: Questionário - Tipos de Participante

A avaliação de usabilidade é efetuada através das respostas de utilizadores a questionários. Para o questionário realizado houve um total de 18 respostas, sendo que todos os utilizadores

[1] Cada requisito possui um valor de relevância que varia entre 0 e 10: 0, 2, 4, 6, 8, 10.

eram vendados e não cegos tal como se pode constatar na figura 7.1, onde o vermelho representa os utilizadores vendados.

O passo seguinte envolveu a recolha de todas as respostas dadas numa tabela de 18 por 10, onde 18 representa o número total de utilizadores que responderam ao inquérito e 10 o número total de questões. O resultado desta recolha e junção pode ser visualizado na figura 7.2:

	Question									
	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
User01	4	2	5	2	4	2	3	2	5	1
User02	5	1	5	1	5	1	5	1	5	1
User03	5	1	5	1	5	1	2	1	4	1
User04	5	2	5	1	4	1	5	2	5	1
User05	5	2	4	1	4	1	4	1	5	1
User06	5	3	5	1	4	1	4	3	5	1
User07	5	1	4	1	4	1	5	2	4	1
User08	5	1	5	1	4	1	5	1	5	1
User09	5	1	4	1	5	1	4	1	5	1
User10	5	2	4	1	4	1	3	2	5	1
User11	5	1	5	1	4	1	4	2	4	1
User12	4	2	4	2	4	2	4	2	4	2
User13	5	1	5	1	5	1	5	1	5	1
User14	5	1	5	1	4	1	5	1	5	1
User15	5	3	5	1	3	1	5	1	5	2
User16	5	1	5	1	5	1	5	1	5	1
User17	5	1	4	1	1	1	5	2	4	1
User18	5	1	5	1	4	1	4	1	5	1

Figura 7.2: Questionário - Tabela de Resultados

Depois da criação da tabela é necessário transformar as respostas de cada um dos utilizadores para uma escala de 0 a 40. Esta transformação foi obtida para cada um destes através da equação representada na figura 7.3, onde é tido em conta as respostas de sentido positivo e as respostas de sentido negativo.

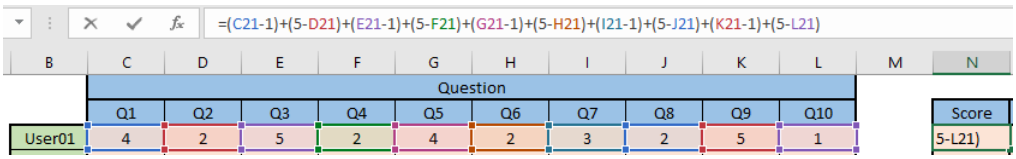


Figura 7.3: Fórmula para Perguntas Positivas e Negativas

Depois da transformação para a escala de 0 a 40 é apenas necessário multiplicar os valores obtidos na escala por 2.5, movendo assim para uma escala de 0 a 100, o que representa o resultado *SUS*. Esta alteração de escala pode ser analisada na figura 7.4.

Score	SUS Score
32	80
40	100
36	90
37	92,5
36	90
34	85
36	90
39	97,5
38	95
34	85
36	90
30	75
40	100
39	97,5
35	87,5
40	100
33	82,5
38	95

Average	90,69444

Figura 7.4: Questionário - Avaliação de Usabilidade

Após a normalização dos resultados e cálculo da média de todas as respostas do questionário foi obtida uma avaliação de 90,7 em 100, tal como se pode analisar na figura 7.4. Seguindo o sistema de classificação *SUS*, apresentado na tabela 7.2, é possível constatar que o sistema obteve uma classificação com um nível **Excelente** (90,7 > 80,3) na medida de usabilidade, respondendo assim à terceira questão de investigação levantada na secção 1.2.3.

7.3.2 Qualidade da Solução

Os resultados finais da avaliação através da metodologia *QEF* estão representados nas figuras 7.5 e 7.6:

q	D	qi	Dimension	Qj	Wij (Fact	Factor	rwjk (requirement weight k in Factor j) {2, 4, 6, 8, 10}	Requirement	wfk % requirement fulfillment k) [0,100]
97%	1,00	97,73	Funcionalidade	100,00	0,32	Acessibilidade	10,00	FA01 - Permite Alterar Contraste da Interface	100
							10,00	FA02 - Permite Alterar Tamanho das Fontes	100
							10,00	FA03 - Permite Alterar Dificuldade de Jogo	100
							10,00	FA04 - Permite Alterar Definições Individuais de Áudio	100
							10,00	FA05 - Permite Controlar Leitor de Menus	100
							10,00	FA06 - Permite Alterar Configurações dos Controlos	100
							10,00	FA07 - O jogo notifica o jogador relativamente às funcionalidades de acessibilidade existentes durante o acesso	100
				93,75	0,36	Jogabilidade	10,00	FG01 - O jogador controla os diferentes estados de jogo	100
							10,00	FG02 - O jogo possui um tutorial para a explicação das mecânicas de jogo	100
							10,00	FG03 - Todas as ações de jogo trasmitem feedback auditivo.	50
							10,00	FG04 - O jogador consegue localizar a direção dos inimigos através do som	100
							10,00	FG05 - O jogador consegue alterar a sua direção	100
							10,00	FG06 - O jogador consegue alterar destruir os inimigos	100
							10,00	FG07 - O jogador consegue ver o seu resultado no final de uma batalha	100
							10,00	FG08 - O jogador consegue ver o seu melhor resultado para cada nível de dificuldade	100
				100,00	0,32	Interação com Utilizador	10,00	FIU01 - O jogador tem acesso a um menu principal	100
							8,00	FIU02 - O jogo possui um menu de pausa	100
							10,00	FIU03 - A interface é intuitiva	100
							10,00	FIU04 - Todos os menus de jogo e texto devem ser convertidos para fala	100
							10,00	FIU05 - O jogo envia feedback através de sonificação para o jogador	100
							10,00	FIU06 - O jogo envia pistas de áudio para transmitir informação ao jogador	100
							8,00	FIU07 - O jogo possui um menu de derrota com as pontuação	100
		100	Suportabilidade	100,00	0,25	Jogabilidade	10,00	SG01 - O jogador sente que os resultados das diferentes ações realizadas em jogo são justas	100
							8,00	SG02 - O jogador controla as diferentes ações de jogo	100
				100,00	0,25	Manutenibilidade	8,00	SM01 - A implementação permite a inclusão de novas funcionalidades	100
							6,00	SM02 - A implementação encontra-se acompanhada de documentação	100
				100,00	0,50	Adaptabilidade	10,00	SA01 - O jogo ajusta-se à resolução de diferentes tamanhos de ecrã	100
							10,00	SA02 - O jogo funciona em múltiplas plataformas	100
							6,00	SA03 - O jogo funciona em múltiplos browsers	100
							10,00	SA04 - É possível jogar com diferentes tipos de equipamentos	100

Figura 7.5: Resultados do QEF - Parte 1

		90,45	Usabilidade	100,00	0,18	Navegação	10,00	UN01 - A interface de utilizador é simples e rápida	100
							10,00	UN02 - O jogo não tem erros durante o funcionamento e os problemas inesperados são tratados	100
				85,00	0,64	Qualidade de Conteúdo	8,00	UQC01 - Todas as mensagens são simples de perceber	100
							4,00	UQC02 - Todo o conteúdo está relacionado ao jogo	100
							10,00	UQC03 - Os textos são simples e concisos	100
							10,00	UQC04 - O jogo fornece informação suficiente para que o jogador possa começar a jogar	100
							10,00	UQC05 - O jogo suporta a língua portuguesa e inglesa	50
							8,00	UQC06 - A interface de utilizador é visualmente consistente ao longo do jogo	50
							10,00	UQC07 - As cores de jogo não são utilizadas para transmitir informação importante.	100
				100,00	0,18	Integridade	10,00	UI01 - Todas as conversões de fala são precisas	100
							10,00	UI02 - As informações de áudio são transmitidas em tempos real	100

Figura 7.6: Resultados do QEF - Parte 2

Existe um total de 41 requisitos distintos a ser analisados e avaliados individualmente. Cada um deles possui uma escala de avaliação própria, sendo que alguns deles são avaliados de acordo com as classificações obtidas na análise de usabilidade. Estas escalas encontram-se representada no anexo A.

Dos 22 requisitos da dimensão de **Funcionalidade** apenas um não se encontra realizado a 100%, possuindo apenas uma avaliação de 50%. Este requisito encontra-se no fator de **Jogabilidade** e procurar avaliar se "Todas as ações de jogo transmitem *feedback* auditivo".

Este requisito possui esta avaliação, já que algumas das ações do utilizador não transmitem *feedback* auditivo, como por exemplo, a rotação para outra direção. Globalmente a dimensão da funcionalidade está **implementada a 97,7% da solução ideal**.

Todos os 8 requisitos da dimensão de **Suportabilidade** encontram-se realizados a 100%. Fazendo com que esta seja a dimensão mais próxima da solução ideal, tendo sido **implementada a 100%**.

Dos 11 requisitos da dimensão de **Usabilidade** apenas dois não se encontram implementados a 100%, ambos possuindo uma avaliação de 50%. Encontram-se no fator de **Qualidade de Conteúdo** e procuram avaliar se "O jogo suporta a língua portuguesa e inglesa"e se "A interface é visualmente consistente ao longo do jogo". O primeiro requisito possui esta avaliação visto que apenas suporta a língua inglesa, o segundo possui algumas inconsistências nas cores das fontes utilizadas. Globalmente a dimensão da usabilidade está **implementada a 90,5% da solução ideal**, levando a que esta seja a dimensão mais afastada.

É assim possível constatar que o sistema obteve uma implementação geral de **97% da solução ideal**, o que também permite responder à primeira questão de investigação levantada na secção 1.2.3.

Capítulo 8

Conclusões

Nesta secção do relatório é realizado um resumo do trabalho que foi desenvolvido, enquadrando os contributos e objetivos atingidos com os que foram inicialmente idealizados. São identificadas e explicadas as limitações existentes e recomendações para trabalho que pode ser realizado e melhorado futuramente. Por fim é feita uma apreciação final do trabalho realizado e uma introspectiva global sobre o tema abordado.

8.1 Contributos

Os principais contributos e benefícios deste projeto para a sociedade, nomeadamente na área dos cegos são:

- **Identificação de Técnicas Relevantes**: Identificação das técnicas e tecnologias mais relevantes para a integração dos cegos nos jogos digitais produzidos. Sendo desta forma possível selecionar e identificar aquelas que são as tecnologias mais significativas e que mais impactam a vida daqueles que mais precisam.
- **Criação de um Jogo Digital Adaptado às Necessidades dos Cegos**: Com deste projeto, tornou-se possível criar um jogo que se encontra adaptado a cegos e às suas necessidades. Com isto, é proporcionado um meio de integração a uma comunidade que não possui as mesmas oportunidades na indústria.
- **Prova de Conceito**: Através da criação deste jogo digital, é criada uma prova de conceito para produtores e desenvolvedores de jogos, podendo ser adaptada para a criação de produtos a pensar nesta comunidade. Pode servir de base e fonte de conhecimento, para todos aqueles que pretendem implementar este tipo de sistemas nos seus jogos, ou até mesmo influenciar perspetivas existentes, no sentido de tornarem mais inclusivas para este tema.

Em suma e com este projeto, o autor não só contribuiu com a criação de um jogo inclusivo e adaptado às características dos cegos, como também captou atenção da indústria para os problemas desta comunidade. Ao mesmo tempo apresenta soluções e tecnologias que podem ser integradas e adaptadas, em projetos existentes ou em fase de design.

8.2 Objetivos

Na reta final do projeto e de forma a avaliar o sucesso deste é necessário fazer uma análise do cumprimento dos objetivos definidos inicialmente. Ao longo da secção 1.2.1 foram definidos 6 objetivos, que são:

- Identificação e Estudo de Técnicas Relevantes
- Identificação e Análise dos problemas dos Utilizadores Principais
- Identificação de Lacunas existentes no Mercado
- Estudo de Casos de Sucesso
- Construção do Jogo Adaptado
- Experimentação em Cenários Reais

Para a realização do primeiro objetivo foram identificadas e estudadas diversas fontes de literatura no estado da arte e que permitiram perceber quais são as principais técnicas utilizadas que colmatam problemas no desenvolvimento de jogos para cegos, tal como referido na secção 2.1.4.

O segundo objetivo, focado na identificação e análise dos principais problemas dos cegos foi realizado com sucesso. Através de análise e estudo de entrevistas realizadas, como se pode aferir na secção 2.1.3.

O terceiro objetivo de identificação das lacunas existentes de mercado foi realizado através da análise do estado da arte e soluções existentes no mercado, feita na secção 2.2.1.

O quarto objetivo do estudo de casos de sucesso foi também desenvolvida na secção 2.2.1 através da verificação das melhores técnicas e características de acessibilidade.

O quinto objetivo, da construção do jogo adaptado foi também concluído com sucesso. Algo que pode ser confirmado através dos resultados de qualidade e usabilidade obtidos no capítulo 7. Onde a solução real se encontra com **97%** do valor da solução ideal e onde foram obtidos **90,7** em 100 na escala de avaliação de *SUS*.

O sexto e último objetivo foi atingido com relativo sucesso. Apesar da classificação de **Excelência** obtida através da avaliação de usabilidade pela metodologia *SUS*, todos os 18 participantes do inquérito eram indivíduos vendados, sendo que nenhum representa na totalidade o utilizador final, o indivíduo cego.

A conclusão com sucesso dos primeiros 3 objetivos responde positivamente à segunda questão de investigação levantada na secção 1.2.3.

8.3 Limitações e Trabalho Futuro

Apesar do sucesso no cumprimentos dos diversos objetivos ainda existem algumas melhorias e trabalho futuro que podem ser realizados. Destaca-se assim os seguintes:

- **Suporte para Múltiplas Linguagens**: Neste momento o jogo apenas suporta a língua inglesa, sendo que qualquer jogador que não compreenda o inglês fica totalmente impossibilitado de, por exemplo, navegar pelos menus fluidamente. Esta é uma melhoria bastante relevante uma vez que aumenta as fronteiras e o alcance que um projeto como este pode ter. Para além disso é muito importante que projetos como este atinjam comunidades que desfavorecidas, que muitas vezes não possuem os mesmo recursos, sendo assim a língua uma barreira.
- **Tornar a leitura de ecrãs escalável e automática**: Neste momento, caso aja a necessidade de adicionar novos menus é necessário criar manualmente um áudio de voz.

Isto pode causar múltiplos problemas, como inconsistências nas falas e um processo demoroso e burocrático. A solução passa por utilizar uma das *API's* que atualmente fornecem serviços de texto para voz.

- **Adição de uma Componente Online**: Uma das componentes mais importantes nos jogos digitais dos dias de hoje é o formato *online*. É assim fundamental que esta componente seja adicionada ao jogo, levando à criação de novas mecânicas de competição ou cooperação.

8.4 Apreciação Final

O planeamento de tarefas foi cumprido assim como todas as principais funcionalidades idealizadas, desta forma o autor considera que se tratou de um projeto de sucesso. Esta experiência deve-se assim aos diversos intervenientes que direta ou indiretamente o ajudaram e apoiaram sem hesitação.

Ao longo da realização do projeto houve sempre uma sensação de confiança e motivação para o seu desenvolvimento. Isto deve-se principalmente a duas razões: o desafio proporcionado ao autor e a importância que pequenos projetos como este podem ter na vida de muitos indivíduos.

São projetos como este que muitas mudam opiniões e atitudes erradas para determinados temas. E sem dúvida que o autor considera este projeto como uma abertura para o mundo de acessibilidade. A partir da leitura de diversas fontes de literaturas e informação houve um aumento da perceção do quanto pequenas implementações e funcionalidades afetam a vida de outras pessoas. E isto é algo que o autor leva consigo para a vida independentemente do sucesso que o projeto pudesse ter.

Como nota final é assim importante destacar toda a experiência não só pela sua positividade, como também pelas as competências técnicas e principalmente humanas que foram desenvolvidas no decorrer do projeto.

Bibliografia

A Blind Legend (s.d.). url: http://www.ablindlegend.com/en/home-2/.

Ackland, Peter, Serge Resnikoff e Rupert Bourne (2017). *World blindness and visual impairment: despite many successes, the problem is growing*. url: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5820628/.

Aguado-Delgado, Juan et al. (2020). «Accessibility in video games: a systematic review». Em: *Universal Access in the Information Society* 19.1, pp. 169–193.

Aguado-Delgado, Juan R. et al. (2018). «Accessibility in video games: a systematic review». Em: *Universal Access in the Information Society* 19.1, pp. 1–1. doi: 10.1007/s10209-018-0628-2. url: https://portal.uah.es/portal/page/portal/epd2_profesores/prof23288/publicaciones/UAIS-D-17-00071R1-Editorial_Comments.pdf.

Airy, Samuel e Judy M. Parr (2001). «MIDI, Music and Me: Students' Perspectives on Composing with MIDI». Em: *Music Education Research* 3.1, pp. 41–49. doi: 10.1080/14613800020029941. eprint: https://doi.org/10.1080/14613800020029941. url: https://doi.org/10.1080/14613800020029941.

Archambault, Dominique et al. (2007). «Computer games and visually impaired people». Em: *Upgrade* 8.2, pp. 43–53.

Ardichvili, Alexander, Richard Cardozo e Sourav Ray (2003). «A theory of entrepreneurial opportunity identification and development». Em: *Journal of Business venturing* 18.1, pp. 105–123.

Audacity (s.d.). *Audacity*. Accessed: 2021-02-19. url: https://www.audacityteam.org/.

AudioGames (s.d.). url: https://audiogames.net/.

Bangor, Aaron, Philip T Kortum e James T Miller (2008). «An empirical evaluation of the system usability scale». Em: *Intl. Journal of Human–Computer Interaction* 24.6, pp. 574–594.

Bierre, Kevin et al. (2005). «Game not over: Accessibility issues in video games». Em: *Proc. of the 3rd International Conference on Universal Access in Human-Computer Interaction*, pp. 22–27.

Bitbucket (s.d.). *Bitbucket*. Accessed: 2021-02-19. url: https://bitbucket.org/.

Cataltepe, Zehra, Yusuf Yaslan e Abdullah Sonmez (2007). «Music Genre Classification Using MIDI and Audio Features». Em: *EURASIP Journal on Advances in Signal Processing* 2007.1. doi: 10.1155/2007/36409. url: https://link.springer.com/content/pdf/10.1155/2007/36409.pdf.

Colwell, Chetz et al. (1998). «Haptic virtual reality for blind computer users». Em: *Proceedings of the third international ACM conference on Assistive technologies*, pp. 92–99.

Computer Science learning for school students (s.d.). url: https://www.teach-ict.com/as_a2_ict_new/ocr/AS_G061/312_software_hardware/specialist_hwsw/miniweb/pg5.htm.

Craighead, Jeff, Jennifer Burke e Robin Murphy (2008). «Using the unity game engine to develop sarge: a case study». Em: *Proceedings of the 2008 Simulation Workshop at the International Conference on Intelligent Robots and Systems (IROS 2008)*.

Cysneiros, Luiz Marcio (s.d.). «Requisitos não funcionais: da elicitação ao modelo conceitual». Em: (). url: http://www-di.inf.puc-rio.br/~julio/Tese%20-%205.pdf.

Eeles, Peter (2005). «Capturing architectural requirements». Em: *IBM Rational developer works*.

Escudeiro, Paula e José Bidarra (2008). «Quantitative evaluation framework (QEF)». Em: *Conselho Editorial/Consejo Editorial* 16.

Gears 5 (s.d.). url: https://www.gears5.com/.

Git (s.d.). *Git*. Accessed: 2021-02-19. url: https://git-scm.com/.

Godot (s.d.). *Godot*. Accessed: 2021-02-19. url: https://godotengine.org/.

Google Trends (s.d.). url: https://trends.google.com/trends/.

Gürel, Emet e Merba Tat (2017). «SWOT analysis: a theoretical review.» Em: *Journal of International Social Research* 10.51.

Haas, John (2014). «A history of the unity game engine». Em: *Diss. WORCESTER POLYTECHNIC INSTITUTE*.

Haije, Erin Gilliam (jan. de 2021). *The Complete Guide to User Feedback*. url: https://mopinion.com/user-feedback/#what-is-user-feedback.

Hill, Simon (mar. de 2021). *Games Rule The App Stores: Most Popular Genres Revealed 2021*. url: https://www.localizedirect.com/posts/most-popular-game-genres-revealed.

Ho, Vanessa (dez. de 2019). *Gears 5 sets a higher bar for accessibility with features that mean more people can play the game*. url: https://news.microsoft.com/features/gears-5-higher-bar-accessibility-more-people-can-play/.

Hope, Computer (fev. de 2020). *What is a Plugin?* url: https://www.computerhope.com/jargon/p/plugin.htm.

How Gaming is Good for Your Mental Health During Lockdown (nov. de 2020). url: https://www.embryodigital.co.uk/how-gaming-is-good-for-your-mental-health-during-lockdown/.

IDGA (2008). «2008-2009 Casual Games White Paper». Em: url: https://www.org.id.tue.nl/IFIP-TC14/documents/IGDACasualGames-WhitePaper-2008.pdf.

JAWS (s.d.). *JAWS*. Accessed: 2021-02-21. url: https://www.freedomscientific.com/products/software/jaws/.

Johnson, Heather A. (abr. de 2017). *Trello*. Accessed: 2021-02-19. url: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5370621/.

Joyce, Alexandre e Raymond L Paquin (2016). «The triple layered business model canvas: A tool to design more sustainable business models». Em: *Journal of cleaner production* 135, pp. 1474–1486.

Khosravi, Khashayar e Yann-Gaël Guéhéneuc (2004). «A quality model for design patterns». Em: *German Industry Standard*.

Koen, P. et al. (2002). «1 Fuzzy Front End : Effective Methods , Tools , and Techniques». Em:

Kuittinen, Jussi et al. (nov. de 2007). «Casual games discussion». Em: pp. 105–112. doi: 10.1145/1328202.1328221.

Larman, Craig (2012). *Agile and iterative development: a managers guide*. Addison-Wesley. url: https://books.google.pt/books?hl=pt-PT&lr=&id=76rnV5Exs50C&oi=fnd&pg=PA1&dq=iterative%20and%20incremental%20development&ots=oaY6_oLRR_&sig=ACvYXmfYNDHBbnbFd2snag1Bc_Q&redir_esc=y#v=onepage&q&f=false.

LMMS (s.d.). *LMMS*. Accessed: 2021-02-20. url: https://lmms.io/.

Loeliger, J. e M. McCullough (2012). *Version Control with Git: Powerful Tools and Techniques for Collaborative Software Development*. O'Reilly Media, pp. 2–4. isbn: 9781449345051. url: https://books.google.pt/books?id=aM7-Oxo3qdQC.

Marketeer, Por (set. de 2020). *O crescimento do mercado de jogos na pandemia e a contribuição dos jogos electrónicos para a saúde mental*. url: https://marketeer.sapo.pt/o-crescimento-do-mercado-de-jogos-na-pandemia-e-a-contribuicao-dos-jogos-electronicos-para-a-saude-mental.

McCarthy, Niall e Felix Richter (jun. de 2015). *Infographic: America's Favorite Video Game Genres*. url: https://www.statista.com/chart/3599/americas-favorite-video-game-genres/.

Mortal Kombat11 (s.d.). url: https://www.mortalkombat.com/.

NVDA (s.d.). *NVDA*. Accessed: 2021-02-21. url: https://www.nvaccess.org/.

Ostergaard, Kirsten (2016). «Applying Kanban principles to electronic resource acquisitions with Trello». Em: *Journal of Electronic Resources Librarianship* 28.1, pp. 48–52. doi: 10.1080/1941126X.2016.1130464. eprint: https://doi.org/10.1080/1941126X.2016.1130464. url: https://doi.org/10.1080/1941126X.2016.1130464.

Osterwalder, Alexander et al. (2014). *Value proposition design: How to create products and services customers want*. John Wiley & Sons.

Over, Voice (s.d.). *Voice Over*. Accessed: 2021-02-21. url: https://www.apple.com/accessibility/vision/.

Peffers, Ken et al. (2007). «A Design Science Research Methodology for Information Systems Research». Em: *Journal of Management Information Systems* 24.3, pp. 45–77. doi: 10.2753/MIS0742-1222240302. eprint: https://doi.org/10.2753/MIS0742-1222240302. url: https://doi.org/10.2753/MIS0742-1222240302.

Perret, Jérôme e Emmanuel Vander Poorten (2018). «Touching virtual reality: a review of haptic gloves». Em: *ACTUATOR 2018; 16th International Conference on New Actuators*. VDE, pp. 1–5.

Pickton, David W e Sheila Wright (1998). «What's swot in strategic analysis?» Em: *Strategic change* 7.2, pp. 101–109.

Polly, Amazon (s.d.). *Amazon Polly*. Accessed: 2021-02-21. url: https://docs.aws.amazon.com/polly/latest/dg/what-is.html.

População residente com deficiência segundo os Censos: total e por tipo de deficiência (2001) (s.d.). url: https://www.pordata.pt/Portugal/Popula%C3%A7%C3%A3o%20residente%20com%20defici%C3%AAncia%20segundo%20os%20Censos%20total%20e%20por%20tipo%20de%20defici%C3%AAncia%20(2001)-1239-9822.

Porter, John R e Julie A Kientz (2013). «An empirical study of issues and barriers to mainstream video game accessibility». Em: *Proceedings of the 15th international ACM SIGACCESS conference on computers and accessibility*, pp. 1–8.

Richter, Felix (set. de 2020). *Gaming: The Most Lucrative Entertainment Industry By Far*. url: https://www.statista.com/chart/22392/global-revenue-of-selected-entertainment-industry-sectors/.

Roberts, Paul (s.d.). «Quality Function Deployment». Em: (). url: https://www.coursehero.com/file/12615334/QFD-Warwick/.

Saaty, Roseanna W (1987). «The analytic hierarchy process—what it is and how it is used». Em: *Mathematical modelling* 9.3-5, pp. 161–176.

Sadato, Norihiro (2005). «How the blind "see" Braille: lessons from functional magnetic resonance imaging». Em: *The Neuroscientist* 11.6, pp. 577–582.

Sanders, Andrew (2016). *An introduction to Unreal engine 4*. CRC Press.

Saunders, Mnk e P. Tosey (2013). «The Layers of Research Design». Em: url: https://www.academia.edu/4107831/The_Layers_of_Research_Design.

Services, Microsoft Cognitive (s.d.). *Microsoft Cognitive Services*. Accessed: 2021-02-21. url: https://azure.microsoft.com/pt-pt/services/cognitive-services/text-to-speech/.

Shades of Doom (s.d.). url: http://www.gmagames.com/sod.html.

SIG top ten (jun. de 2019). url: https://igda-gasig.org/how/sig-top-ten/.

Sjöström, Calle (2001). «Using haptics in computer interfaces for blind people». Em: *CHI'01 extended abstracts on Human Factors in Computing Systems*, pp. 245–246.

Šmíd, Antonín (2017). «Comparison of unity and unreal engine». Em: *Czech Technical University in Prague*, pp. 41–61.

Software (s.d.). url: https://dictionary.cambridge.org/pt/dicionario/ingles/software.

Speech, Google Text to (s.d.). *Google Text to Speech*. Accessed: 2021-02-21. url: https://cloud.google.com/text-to-speech.

Stockman, Tony e Oussama Metatla (2008). «The influence of screen-readers on web cognition». Em: *Proceeding of Accessible design in the digital world conference (ADDW 2008), York, UK*.

Studio, FL (s.d.). *FL Studio*. Accessed: 2021-02-20. url: https://www.image-line.com/.

Supernova, Dolphin (s.d.). *Dolphin Supernova*. Accessed: 2021-02-21. url: https://yourdolphin.com/products/individuals/families/supernova.

Taylor, Paul (2009). *Text-to-speech synthesis*. Cambridge university press.

Techopedia (jul. de 2011). *What is a Backup? - Definition from Techopedia*. url: https://www.techopedia.com/definition/1056/backup.

Tengli, Mahesh Bhimashankar (ago. de 2020). *Blog 132-Research Onion: A Systematic Approach to Designing Research Methodology*. url: https://www.aesanetwork.org/research-onion-a-systematic-approach-to-designing-research-methodology/.

Terraformers (s.d.). url: http://terraformers.nu/.

The Division 2 (s.d.). url: https://www.ubisoft.com/en-gb/game/the-division/the-division-2.

The Last of Us 2 (s.d.). url: https://www.playstation.com/pt-pt/games/the-last-of-us-part-ii/.

Thylefors, B et al. (1995). *Global data on blindness*. url: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2486591.

Trello (s.d.). *Trello*. Accessed: 2021-02-19. url: https://trello.com/.

Unity (s.d.). *Unity*. Accessed: 2021-02-19. url: https://unity.com/.

Unreal (s.d.). *Unreal*. Accessed: 2021-02-19. url: https://www.unrealengine.com.

Vision impairment and blindness (s.d.). url: https://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment.

VoiceForge (s.d.). *VoiceForge*. Accessed: 2021-02-21. url: https://www.voiceforge.com/.

Watson, Léonie (fev. de 2020). *What is a screen reader?* url: https://www.nomensa.com/blog/2005/what-screen-reader.

What Is A DAW Program? Digital Audio Workstation (fev. de 2021). url: https://mixbutton.com/home-recording-articles/what-is-a-daw-program/.

What is a Software Bug? (Jul. de 2017). url: https://www.goodfirms.co/glossary/software-bug/.

What Is a VST? All Your VST Questions Answered! (Set. de 2020). url: https://www.careersinmusic.com/what-is-a-vst/.

What is an IDE? (S.d.). url: https://opensource.com/resources/what-ide.

White, Gareth R., Geraldine Fitzpatrick e Graham McAllister (2008). «Toward Accessible 3D Virtual Environments for the Blind and Visually Impaired». Em: *Proceedings of the 3rd International Conference on Digital Interactive Media in Entertainment and Arts*. DIMEA '08. Athens, Greece: Association for Computing Machinery, pp. 134–141. isbn: 9781605582481. doi: 10.1145/1413634.1413663. url: https://doi.org/10.1145/1413634.1413663.

Wijman, Tom (mai. de 2020). *The World's 2.7 Billion Gamers Will Spend $159.3 Billion on Games in 2020; The Market Will Surpass $200 Billion by 2023*. url: https://newzoo.com/insights/articles/newzoo-games-market-numbers-revenues-and-audience-2020-2023/.

Yuan, Bei (2009). «Towards generalized accessibility of video games for the visually impaired». Tese de doutoramento.

Apêndice A

Quantitative Evaluation FrameWork

A.1 Escala de Avaliação de Requisitos

Escalas de avaliação de requisitos para os diferentes fatores. É a partir destas escalas que se avalia o grau de cumprimentos de cada um dos diferentes requisitos avaliados no capítulo 7.

A.1.1 Funcionalidade

	Wfk - Fulifilment (%)				
Requirement	**0**	**25**	**50**	**75**	**100**
FA01 - Permite Alterar Contraste da Interface	Não Implementado	-	-	-	Implementado
FA02 - Permite Alterar Tamanho das Fontes	Não Implementado	-	-	-	Implementado
FA03 - Permite Alterar Dificuldade de Jogo	Não Implementado	-	-	-	Implementado
FA04 - Permite Alterar Definições Individuais de Áudio	Não Implementado	-	O jogador consegue alterar o volume globalmente	-	Implementado
FA05 - Permite Controlar Leitor de Menus	Não Implementado	-	-	-	Implementado
FA06 - Permite Alterar Configurações dos Controlos	Não Implementado	-	-	-	Implementado
FA07 - O jogo notifica o jogador relativamente às funcionalidades de acessibilidade existentes durante o acesso	Não Implementado	-	O jogo apenas notifica o jogador das funcionalidade de acessibilidade e não da sua localização	-	Implementado

Figura A.1: Escala de Avaliação de Requisitos para Fator Acessibilidade

	Wfk - Fullfilment (%)				
Requirement	**0**	**25**	**50**	**75**	**100**
FG01 - O jogador controla os diferentes estados de jogo	Não Implementado	-	O jogador apenas consegue pausar em alguns momentos	-	Implementado
FG02 - O jogo possui um tutorial para a explicação das mecânicas de jogo	Não Implementado	-	-	-	Implementado
FG03 - Todas as ações de jogo trasmitem feedback auditivo.	Não Implementado	-	Apenas algumas das ações transmitem feedback auditivo	-	Implementado
FG04 - O jogador consegue localizar a direção dos inimigos através do som	Não Implementado	-	O jogador ouve os inimigos mas não consegue determinar a direção	-	Implementado
FG05 - O jogador consegue alterar a sua direção	Não Implementado	-	-	-	Implementado
FG06 - O jogador consegue alterar destruir os inimigos	Não Implementado	-	-	-	Implementado
FG07 - O jogador consegue ver o seu resultado no final de uma batalha	Não Implementado	-	-	-	Implementado
FG08 - O jogador consegue ver o seu melhor resultado para cada nível de dificuldade	Não Implementado	-	O jogador apenas consegue ver o seu melhor resultado obtido globalmente	-	Implementado

Figura A.2: Escala de Avaliação de Requisitos para Fator Jogabilidade

	Wfk - Fullfilment (%)				
Requirement	**0**	**25**	**50**	**75**	**100**
FIU01 - O jogador tem acesso a um menu principal	Não Implementado	-	-	-	Implementado
FIU02 - O jogo possui um menu de pausa	Não Implementado	-	-	-	Implementado
FIU03 - A interface é intuitiva	0-20% of positive questionaries	20-40% of positive questionaries	40-60% of positive questionaries	60-80% of positive questionaries	80-100% of positive questionaries
FIU04 - Todos os menus de jogo e texto devem ser convertidos para fala	Não Implementado	-	Apenas alguns dos menus e textos são convertidos para fala	-	Implementado
FIU05 - O jogo envia feedback através de sonificação para o jogador	Não Implementado	-	-	-	Implementado
FIU06 - O jogo envia pistas de áudio para transmitir informação ao jogador	Não Implementado	-	-	-	Implementado
FIU07 - O jogo possui um menu de derrota com as pontuação	Não Implementado	-	O jogo possui um menu de derrota, mas não é apresentada a pontuação final do jogador	-	Implementado

Figura A.3: Escala de Avaliação de Requisitos para Fator Interação com Utilizador

A.1.2 Suportabilidade

	Wfk - Fullfilment (%)				
Requirement	**0**	**25**	**50**	**75**	**100**
SG01 - O jogador sente que os resultados das diferentes ações realizadas em jogo são justas	0-20% of positive questionaries	20-40% of positive questionaries	40-60% of positive questionaries	60-80% of positive questionaries	80-100% of positive questionaries
SG02 - O jogador controla as diferentes ações de jogo	Não Implementado	-	O jogador apenas tem a liberdidade de decisão de algumas das ações de jogo	-	Implementado

Figura A.4: Escala de Avaliação de Requisitos para Fator Jogabilidade (Suportabilidade)

	Wfk - Fullfilment (%)				
Requirement	**0**	**25**	**50**	**75**	**100**
SM01 - A implementação permite a inclusão de novas funcionalidades	Não Implementado	-	-	-	Implementado
SM02 - A implementação encontra-se acompanhada de documentação	Não Implementado	-	Apenas alguma da implementação se encontra documentada	-	Implementado

Figura A.5: Escala de Avaliação de Requisitos para Fator Manutenibilidade

	Wfk - Fullfilment (%)				
Requirement	**0**	**25**	**50**	**75**	**100**
SA01 - O jogo ajusta-se à resolução de diferentes tamanhos de ecrã	Não Implementado	-	Ajusta-se mas com problemas	-	Implementado
SA02 - O jogo funciona em múltiplas plataformas	Não Implementado	-	1 Plataforma	-	2 Plataformas
SA03 - O jogo funciona em múltiplos browsers	Não Implementado	1 Browser	2 Browsers	-	3 Browser
SA04 - É possível jogar com diferentes tipos de equipamentos	Não Implementado	-	1 equipamento de input	-	2 equipamentos de input

Figura A.6: Escala de Avaliação de Requisitos para Fator Adaptabilidade

A.1.3 Usabilidade

	Wfk - Fullfilment (%)				
Requirement	**0**	**25**	**50**	**75**	**100**
UN01 - A interface de utilizador é simples e rápida	0-20% of positive questionaries	20-40% of positive questionaries	40-60% of positive questionaries	60-80% of positive questionaries	80-100% of positive questionaries
UN02 - O jogo não tem erros durante o funcionamento e os problemas inesperados são tratados	0-20% of positive questionaries	20-40% of positive questionaries	40-60% of positive questionaries	60-80% of positive questionaries	80-100% of positive questionaries

Figura A.7: Escala de Avaliação de Requisitos para Fator Navegação

	Wfk - Fullfilment (%)				
Requirement	**0**	**25**	**50**	**75**	**100**
UQC01 - Todas as mensagens são simples de perceber	0-20% of positive questionaries	20-40% of positive questionaries	40-60% of positive questionaries	60-80% of positive questionaries	80-100% of positive questionaries
UQC02 - Todo o conteúdo está relacionado ao jogo	Não Implementado	-	Algum do conteúdo existente não é relevante	-	Implementado
UQC03 - Os textos são simples e concisos	0-20% of positive questionaries	20-40% of positive questionaries	40-60% of positive questionaries	60-80% of positive questionaries	80-100% of positive questionaries
UQC04 - O jogo fornece informação suficiente para que o jogador possa começar a jogar	0-20% of positive questionaries	20-40% of positive questionaries	40-60% of positive questionaries	60-80% of positive questionaries	80-100% of positive questionaries
UQC05 - O jogo suporta a língua portuguesa e inglesa	Não Implementado	-	O jogo tem suporte para apenas 1 das línguas	-	O jogo possui suporte para as 2 línguas
UQC06 - A interface de utilizador é visualmente consistente ao longo do jogo	Não Implementado	-	Algumas inconsistências visuais	-	Implementado
UQC07 - As cores de jogo não são utilizadas para transmitir informação importante.	Não Implementado	-	-	-	Implementado

Figura A.8: Escala de Avaliação de Requisitos para Fator Qualidade de Conteúdo

	Wfk - Fullfilment (%)				
Requirement	**0**	**25**	**50**	**75**	**100**
UI01 - Todas as conversões de fala são precisas	Não Implementado	-	Algumas das conversões de texto para fala são imprecisas	-	Implementado
UI02 - As informações de áudio são transmitidas em tempos real	Não Implementado	-	Apenas algumas das informações são transmitidas em tempo real	-	Implementado

Figura A.9: Escala de Avaliação de Requisitos para Fator Integridade