UNIVERSIDADE ESTADUAL DO SUDOESTE DA BAHIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

Iran Freitas Ribeiro

DESENVOLVIMENTO DE SOFTWARE EDUCATIVO PARA O PROCESSAMENTO AUDITIVO: UMA FERRAMENTA PARA A ALFABETIZAÇÃO

VITÓRIA DA CONQUISTA – BAHIA
2018

Iran Freitas Ribeiro

DESENVOLVIMENTO DE SOFTWARE EDUCATIVO PARA O PROCESSAMENTO AUDITIVO: UMA FERRAMENTA PARA A ALFABETIZAÇÃO

Monografia apresentada ao Curso de Graduação em Ciência da Computação da Universidade Estadual do Sudoeste da Bahia, como requisito parcial para obtenção do Grau de Bacharel em Ciência da Computação.
Orientador(a): Prof. Dr. Alzira Ferreira da Silva
Coorientador: Prof. Dr. Ronei Guaresi

VITÓRIA DA CONQUISTA – BAHIA
2018

Iran Freitas Ribeiro

DESENVOLVIMENTO DE SOFTWARE EDUCATIVO PARA O PROCESSAMENTO AUDITIVO: UMA FERRAMENTA PARA A ALFABETIZAÇÃO

Aprovada em __/__/____

BANCA EXAMINADORA

_____
Prof. Dr. Alzira Ferreira da Silva
Universidade Estadual do Sudoeste da Bahia - UESB
Orientadora

_____
Prof. Dr. Ronei Guaresi
Universidade Estadual do Sudoeste da Bahia - UESB
Coorientador

_____
Prof. Dr. Gidevaldo Novais dos Santos
Universidade Estadual do Sudoeste da Bahia - UESB

DEDICATÓRIA

Dedico esta conquista a meu pais, Edson e Marizete, que tanto me apoiam em todas as escolhas da minha vida.

AGRADECIMENTOS

Agradeço principalmente a meu pai, Edson, pelo incentivo, confiança e suor derramado para que eu pudesse ter a melhor educação possível; e a minha mãe, Marizete, pelo carinho, paciência e compreensão.

Aos amigos da graduação, em especial, Jonas, Gevaldo, Carlos Alberto e Andrei, pelo companheirismo e resenhas dentro e fora da universidade que tornaram a realização deste e de outros trabalhos menos difícil.

Aos meus amigos do apartamento, atuais e antigos, que amenizaram a saudade que eu sinto de casa e tornaram a estadia nesta cidade menos dolorosa.

Aos meus professores do ensino fundamental e médio, especialmente Antônio Carlos e Gisele, que apostaram no meu potencial e me incentivaram a ir mais longe na carreira acadêmica.

À secretária do Colegiado de Computação, Celina, por todo o apoio desde o meu primeiro dia de aula no curso.

Aos mestres da universidade pelos conhecimentos que a mim foram passados durante a graduação, principalmente Alzira, minha orientadora, pelo tempo e orientação dedicados a este trabalho.

Ao meu coorientador Ronei, pela orientação e oportunidade no projeto de iniciação científica, que diretamente influenciou no resultado deste trabalho.

A todos vocês, meu muito obrigado!

RESUMO

O funcionamento do processamento auditivo das crianças dentro dos padrões esperados, especialmente o componente da alça fonológica da memória de trabalho, está intimamente relacionado à aquisição e ao aprendizado inicial da leitura e da escrita. Nesse sentido, o objetivo deste trabalho foi a construção de um *software* educativo, com enfoque na melhora da memória de trabalho, especialmente do processamento auditivo, que possa ser utilizado como uma ferramenta complementar para a alfabetização de crianças, em especial daquelas com atraso escolar. O *software* aqui apresentado trata-se de um jogo de memória auditiva, em que a criança precisa formar pares de fonemas correspondentes para poder pontuar e melhorar a sua posição no *ranking* de jogadores. Para o desenvolvimento da interação com o jogador e da interface, foram realizadas pesquisas bibliográficas acerca do design de *software* para crianças, além de análises de outros jogos educativos presentes no mercado. A implementação do *software* foi feita utilizando-se a metodologia *Scrum* e o padrão *MVC* em que mobilizou-se as seguintes tecnologias: linguagem de programação Java em conjunto com a biblioteca *JavaFX,* o *framework Scene Builder,* a *IDE NetBeans*, a plataforma de versionamento *Github*, o pacote de imagens *Kids Fantasy GUI* e os editores *Audacity* e *GIMP.*

Palavras-chave: Processamento auditivo. Software educativo. Leitura. Escrita.

ABSTRACT

The performance of the auditory processing of children according to the expected standards, especially the component of the phonological part of the working memory, is closely related to the initial acquisition and learning of reading and writing. Therefore, the objective of this work was the construction of an educational software, focusing on the improvement of working memory, especially of the auditory processing, that can be used as a complementary tool for children literacy, especially those with school backwardness.The software presented herein is an auditory memory game, in which the child needs to form pairs of matching phonemes in order to score and improve its position in the ranking of players. For developing the interaction with the player and the interface, we have done a bibliographic research about the software design for children, as well as analyzing other educational games on the market. The software implementation was done by using the Scrum methodology and the MVC standard in which the following technologies were mobilized: Java programming language in conjunction with the JavaFX library, the Scene Builder framework, the NetBeans IDE, the Github versioning platform, the Kids Fantasy GUI image package and the Audacity and GIMP editors.

Keywords: Auditory processing. Educational software. Reading. Writing.

Lista de Figuras

SUMÁRIO

1 INTRODUÇÃO

A área de educação infantil, especificamente nos primeiros anos de vida da criança, tem sido alvo de diversos estudos nos últimos anos, principalmente no que diz respeito às práticas pedagógicas utilizadas no processo de alfabetização.

Este estudo encontra-se na interface entre a Aquisição da Linguagem (especificamente a aquisição e o aprendizado da leitura e da escrita) e a Ciência da Computação. Baseados em achados psico/neurocientíficos acerca do aprendizado da leitura e da escrita, objetiva a elaboração e testagem de ferramenta tecnológica (provida de conhecimentos na área da Ciência da Computação) com o fim de acurar o processamento auditivo, aspecto caro à aquisição e ao aprendizado inicial da leitura e da escrita (Aquisição da Linguagem)

Em relação às competências acerca da apropriação da modalidade escrita, a alfabetização pode ser entendida como a aquisição do sistema convencional de escrita de maneira tal que, quando o indivíduo utiliza esse sistema em suas práticas sociais, tem-se o que Soares (2009) denomina de letramento. Para a autora “a alfabetização só tem sentido quando desenvolvida no contexto de práticas sociais de leitura e de escrita [...];” enquanto que o letramento, “só pode desenvolver-se na dependência da e por meio da aprendizagem do sistema de escrita” (SOARES, 2009, p. 97). Embora diversos métodos tenham sido utilizados na alfabetização infantil, o fracasso deles é evidente quando observadas as taxas de analfabetos funcionais divulgadas pelo Indicador de Alfabetismo Funcional (INAF, 2016). Uma das possíveis causas desse fracasso advém da dificuldade que algumas crianças possuem de processar cognitivamente/entender os fonemas que são apresentados pelo professor. O que, segundo Andrade (2008), em alguns casos pode ser caracterizado como um distúrbio do processamento auditivo.

Lemos (1999) afirma que “o termo processamento auditivo se refere a como o sistema auditivo […] recebe, analisa e organiza as informações auditivas”. Já para

Machado *et al*. (2011), esse nível de processamento cognitivo pode ser entendido como “o conjunto de habilidades específicas das quais o indivíduo depende para interpretar o que ouve”.

Essas habilidades, de acordo com Machado *et al*. (2001) e Engelmann e Ferreira (2009), têm seu desenvolvimento iniciado desde os primeiros anos de vida da criança, de forma que aqueles “com queixas de dificuldades escolares geralmente apresentam pior desempenho em testes de processamento auditivo em função do atraso na maturação das habilidades auditivas” (ENGELMANN e FERREIRA, 2009, p. 69).

Nesse sentido, percebe-se que o processo envolvido na aprendizagem, e consequentemente na alfabetização, está intimamente relacionado às habilidades descritas por Machado *et al*. (2001), acerca dos componentes do processamento auditivo: detecção, sensação, discriminação, localização, reconhecimento, compreensão, atenção e memória.

Por outro lado, a área da Ciência da Computação, que também está envolvida neste estudo, é uma área vasta que, principalmente nas duas últimas décadas, tornou-se indispensável nas mais diversas esferas da sociedade. No campo da educação, uma das aplicações mais interessantes dos conceitos e ferramentas da tecnologia da informação são os *softwares* educacionais.

Ainda que haja algumas divergências quanto à definição dessa categoria de *software*, uma ferramenta pode ser considerada educativa quando seu objetivo principal se volta para o ensino de um determinado assunto. *Softwares* educativos têm sido amplamente utilizados nas últimas décadas, como uma forma de auxiliar o processo de alfabetização de crianças, uma vez que por um lado elas são estimuladas muito precocemente ao uso de computadores e, por outro, os *softwares* possuem animações e gráficos visuais que favorecem a atenção e encorajam a criança a interagir com o *software* mais tempo que algumas terapias tradicionais (BALEN; MASSIGNANI; SCHILLO, 2008).

Os primeiros estudos que relacionavam o processamento auditivo às questões de dificuldades de aprendizagem são da década de 1990, embora já na década de 1970 Tallal e Piercy (1974) mostraram como a afasia, uma doença que afeta o sistema auditivo, influenciava a percepção das consoantes por parte das crianças observadas no estudo.

As primeiras pesquisas na área de *software* educativo, por volta de 1990, tratavam de avaliar a eficácia de alguns *softwares* disponíveis no mercado, de apontar as principais dificuldades de utilizá-los nas escolas, o potencial que os mesmos possuíam no campo da educação e indicavam alguns requisitos necessários ao desenvolvimento de um *software* educativo de qualidade.

Relacionados às áreas da Linguística, mais especificamente da área da Aquisição da Linguagem, e do processamento auditivo, diversos *softwares* de treinamento auditivo foram desenvolvidos, sendo o *Fast ForWord* um dos pioneiros e mais conhecidos. Balen, Massignani e Schillo (2008, p. 572) o testaram em crianças brasileiras e concluíram que, embora seja possível perceber sua efetividade, “são necessários novas pesquisas com uma amostra maior de participantes para verificar a eficácia deste *software* em crianças brasileiras”.

Recentemente, Ojanen *et al*. (2015) apresentaram um estudo em que um jogo, denominado de *GraphoGame*, foi testado em algumas escolas de países africanos. O estudo mostrou que, embora o *software* tenha sido desenvolvido inicialmente para crianças finlandesas com dificuldades de leitura, essa ferramenta também poderia ser utilizada em algumas escolas africanas, uma vez que boa parte das línguas daquele continente tem seu sistema de escrita similar ao da Finlândia.

Os problemas dos *softwares* citados acima, *Fast ForWord* e *GraphoGame*, estão em: ser muito caro, no caso do primeiro e ainda não estar disponível no mercado brasileiro, no caso do segundo. Em função disso, empreendemos este estudo com o intuito de disponibilizar para o mercado brasileiro um produto educativo efetivo e viável para qualificar o ensino em nosso país.

A alfabetização plena possui um papel fundamental na formação do indivíduo, uma vez que a leitura o influencia a analisar a sociedade, ampliar visões e interpretações sobre o mundo e melhorar as funções cognitivas (DEHAENE, 2012), além de tornar o indivíduo capaz de participar e transformar a sociedade onde está inserido (KRUG, 2015). Nesse sentido, o principal objetivo deste trabalho foi a construção de um *software* educativo, denominado Jogo de Memória Auditiva, com enfoque na melhora da memória de trabalho, especialmente a do processamento auditivo, que possa ser utilizado como uma ferramenta complementar à alfabetização de crianças.

2 REFERENCIAL TEÓRICO

2.1 PROCESSAMENTO AUDITIVO E SUA RELAÇÃO COM AQUISIÇÃO E APRENDIZADO INICIAL DE LEITURA E ESCRITA

Durante o processo inicial de aquisição e aprendizagem da leitura e da escrita, as funções executivas atuam de maneira determinante e são fundamentais para o sucesso do aprendizado. Uma importante função executiva a ser citada é a Memória de Trabalho, a qual Piper (2013) destaca tratar-se de um componente da função executiva responsável por armazenar e reter temporariamente a informação enquanto uma determinada atividade está sendo realizada, oferecendo suporte às tarefas cognitivas, como, por exemplo, o aprendizado da leitura e a escrita.

Dentre os diferentes tipos de Memória de Trabalho existentes, daremos ênfase à Memória de Trabalho Fonológica (ou auditiva), que, segundo Baddeley (2003), processa as informações que são verbalmente codificadas e viabiliza o armazenamento temporário dos resultados do processamento fonológico para a decodificação de palavras durante a leitura e a codificação durante a escrita.

Nesse sentido, Costa (2010, p. 14) afirma que crianças com dificuldades de leitura apresentam, em geral, um processo de decodificação das palavras mais lento e cansativo que o de outras crianças na mesma faixa etária. Além disso, a autora defende que “a pessoa com dificuldades de leitura tem problemas com o código fonológico na memória de trabalho e não consegue traduzir a informação visual em forma fonológica”.

Como referido acima, um dos papéis essenciais ao funcionamento da Memória de Trabalho Fonológica é executado pelo processamento auditivo. Diversas definições para o termo foram apresentadas nas últimas décadas e as mais recentes defendem que processamento auditivo pode ser entendido como o conjunto de habilidades específicas das quais o indivíduo depende para interpretar o que ouve (MACHADO et *al*, 2009) ou um conjunto de operações que o sistema auditivo realiza (ALVAREZ, 1997 *apud* ENGELMANN e FERREIRA, 2009).

Neste trabalho, entretanto, será utilizada a definição defendida por Lemos (1999) entre outros autores (NEVES, SCHOCHAT, 2005; ENGELMANN, FERREIRA,

2009), em que o processamento auditivo é definido como a parte do sistema nervoso auditivo, mais precisamente sistema auditivo periférico e central, que é capaz de decodificar e compreender a fala.

De maneira geral, o processamento auditivo possui três momentos diferentes: a percepção auditiva (que capta os sons das palavras), a discriminação auditiva (que forma grupos de sons a partir de suas semelhanças) e a memória auditiva (que tanto armazena quanto recupera informações importantes dos sons captados) (CAPELLINI, GERMANO, CARDOSO, 2008).

Além disso, como uma habilidade cognitiva, o processamento auditivo se modifica, otimiza e amadurece ao longo tempo (EUGÊNIO, ESCALDA, LEMOS, 2012). Espera-se, assim, que a habilidade de processamento auditivo de um bebê seja inferior a de uma criança de 5 anos, que, de maneira similar, é menos desenvolvida que a de um adulto. E embora não exista um consenso acerca do amadurecimento do processamento auditivo, alguns autores defendem que seu início dá-se já nos primeiros anos de vida do indivíduo até os 4 anos de idade, com cada uma de suas experiências sonoras, enquanto outros afirmam que o mesmo ocorre até os 6 anos (ENGELMANN, FERREIRA, 2009). Ou seja, o processamento auditivo em crianças em processo de alfabetização é variável de criança para criança, de modo que as que têm baixos escores em testes de processamento auditivo podem ter mais dificuldades de aquisição e aprendizado inicial de leitura e escrita. Além desse aspecto, como a memória de trabalho é um componente variável, estudos atestam que programas interventivos podem melhorar essa função executiva (DEHAENE, 2012).

Esse processo de maturação do sistema auditivo nem sempre ocorre como deveria, de maneira tal que algumas de suas funções não são executadas normalmente. Quando isso acontece, tem-se o que Andrade (2008) denomina de distúrbio do processamento auditivo, que pode ter diversas causas. Machado *et al*. (2009) afirma que alterações neurológicas ou alterações sensoriais auditivas são uma delas. Alvarez, Caetano e Nastas (1997, *apud* Lemos, 1999) afirmam que algumas das prováveis causas para a referida desordem são: dificuldades durante a gestação e o nascimento; febre alta durante a primeira infância; otites frequentes nos três primeiros anos de vida; hereditariedade e falhas genéticas.

Independente da causa, crianças com problemas e distúrbios no processamento auditivo podem apresentar deficit na fala (CHANDRASEKARAN,

KRAUS, 2010), problemas de leitura e atrasos na linguagem, entre outras dificuldades na aprendizagem escolar (MACHADO *et al*., 2011). Ainda, Engelmann e Ferreira (2009, p. 69) afirmam que “indivíduos com queixas de dificuldades escolares geralmente apresentam pior desempenho em testes de processamento auditivo em função do atraso na maturação das habilidades auditivas”.

Nesse sentido, Machado *et al*. (2009) apontam a necessidade de se questionar sobre uma possível relação entre as desordens do processamento auditivo e um distúrbio de leitura e escrita, tendo em vista que a habilidade de ouvir e interpretar corretamente o que foi ouvido é fundamental para o processo de aprendizagem (NEVES e SCHOCHAT, 2005).

Um ambiente linguístico auditivamente estimulante é, normalmente, suficiente para a aquisição da fala, entretanto, conforme Dehaene (2012), a escrita é uma invenção cultural recente para a qual o cérebro humano não evoluiu e que o adequado processamento dos fonemas é condição para a alfabetização. Dessa forma, a simples exposição à forma escrita é, em geral, insuficiente para que se aprenda a ler e escrever. Para o autor, uma das condições para que esse aprendizado ocorra é a associação entre grafemas e fonemas. De acordo com o autor, os estudos sugerem estreita relação entre deficit no processamento fonológico e dislexia, patologia que impõe dificuldade acentuada de aprendizado.

Dessa forma, é possível perceber a importância da memória auditiva (componente da memória de trabalho) no processo de aprendizado da leitura (UEHARA, LANDEIRA-FERNANDEZ, 2010), uma vez que esta competência relaciona-se com a habilidade de decodificar uma palavra e a habilidade de compreender um texto escrito, conectando às ideias e aos conhecimentos já presentes na memória de longo prazo.

Segundo Guaresi (2017), sistemas alfabéticos de escrita representam a fala a partir de seu componente fonológico, assim, de maneira geral, cada som que produzimos tem uma correspondência escrita. Entretanto, “não há uma transparência plena entre grafemas e fonemas na Língua Portuguesa, afinal, a escrita representa a fala, mas não exatamente tal qual como percebida” (GUARESI, 2017, p.39).

Da mesma maneira, captação e processamento inadequado do som, como deficit em testes de memória de trabalho, podem prejudicar o aprendizado da leitura e da escrita. O som que a criança precisa ouvir e interpretar deve ser de qualidade e

o mais claro possível. Por isso, uma das preocupações durante o desenvolvimento do Jogo de Memória Auditiva foi que os áudios que a criança iria ouvir fosse livre de ruídos. Além disso, durante a utilização do jogo, o uso de um dispositivo de fone é indispensável a fim de evitar que sons externos influenciem no áudio ouvido.

Em suma, a literatura científica atesta que o processamento auditivo, em especial a capacidade de reter informações acústicas na memória de trabalho, está estreitamente relacionado à aquisição e ao aprendizado da leitura e da escrita (PINKERING, 2001). Por outro lado, outro conjunto de estudos sugere que competências do âmbito cognitivo podem ser melhoradas por meio de programas interventivos (DEHAENE, 2012). Esses aspectos justificam o delineamento do jogo aqui apresentado.

2.2 SOFTWARE EDUCATIVO

Dentre os diferentes tipos de *softwares* que podem ser desenvolvidos, um merece destaque nesta pesquisa. O *software* educativo ou educacional é assim caracterizado quando um *software* está inserido em “contextos de ensino-aprendizagem” (MORAIS, 2003, p. 21). Ainda, de acordo com Morais (2003), o principal objetivo ao se utilizar *software* educativo é tornar o ensino e a aprendizagem mais fáceis, independentemente do conteúdo didático que está sendo ensinado.

Lucena (1992, *apud* TEIXEIRA e BRANDÃO, 2003, p. 2) defende que *software* educacional “é todo aquele programa que possa ser usado para algum objetivo educacional, pedagogicamente defensável, por professores e alunos, qualquer que seja a natureza ou finalidade para o qual tenha sido criado”. Segundo Teixeira e Brandão (2003), para sua finalidade educacional, o *software* precisa ser avaliado desde a sua interface e adequação pedagógica, bem como necessita atender às necessidades dos usuários.

Chaves (2005), contudo, afirma que há uma dificuldade em tentar-se conceituar definitivamente o termo “*software* educativo”, tendo em vista que até mesmo um *software* que não tenha sido inicialmente construído para o fim didático, pode ser considerado como educativo por alguma situação ou utilização específica.

De toda forma, e considerando a observação feita por Chaves (2005), Marques (1996, *apud* GIL e MENEZES, 2004, p. 1) afirmam que a utilização dos *softwares* no meio educacional "tem que ser entendida e condicionada com as características que possui esse *software*, com o uso que se vai fazer dele, com a adequação ao contexto e, principalmente, à sua correcta integração nas actividades lectivas". Sendo assim, a definição de *software* educativo adotada aqui será aquela defendida por Lucena (1992, *apud* TEIXEIRA e BRANDÃO, 2003).

Os *softwares* educativos podem ser classificados em diversos tipos, sendo os tutoriais, os de simulação e os jogos educativos os mais utilizados. De acordo com Morais (2003), esses três tipos possuem uma característica diferente da maioria dos *softwares* educativos, uma vez que permitem a seus usuários construir o conhecimento tendo o professor à disposição para o saneamento de possíveis dúvidas.

Considerando-se o que defende Morais (2003) e Jucá (2006), tem-se a seguir os três tipos de *softwares* educativos anteriormente citados: a) Tutoriais: apresentam, de uma forma diferente, o conteúdo dado pelo professor na sala de aula, por exemplo, através de imagens, sons e animações; b) Simulação: apresentam a modelagem de um sistema ou situação real, utilizando gráficos e imagens, além de oferecerem um ambiente exploratório onde o usuário pode tomar decisões e comprovar, em tempo real, as consequências de cada uma delas; c) Jogos educativos: nesse tipo de *software* o usuário adota estratégias para chegar a um objetivo predeterminado, sendo necessário que ele possua algumas habilidades que variam de acordo cada tipo de jogo, como destreza, associação de ideias, comparações lógicas e raciocínio.

Nesse sentido, tendo em vista o que foi exposto acerca dos tipos de *softwares* educativos, o *software* aqui desenvolvido classifica-se como jogo educativo, categoria que será mais bem descrita a seguir.

2.2.1 Jogos educativos e sua aplicabilidade à escolarização

Segundo Prensky (2012, *apud* RAMOS, 2015), os jogos de uma maneira geral, sendo eles eletrônicos ou não, possuem algumas características em comum:

regras, objetivos, resultados e *feedback*, desafio, interação e enredo.

Em sua pesquisa, Juul (2010) analisou uma série de conceitos de jogos e chegou a uma definição similar à que foi dada por Ramos. Para Juul (2010), um jogo possui seis pontos principais:

- Regras: são baseados em regras;
- Resultados variáveis e quantificáveis: possuem resultados que podem variar e são quantificáveis;
- Valor atribuído aos possíveis resultados: os possíveis resultados recebem valores diferentes, podendo ser positivos ou não;
- Esforço do jogador: o jogador precisa se esforçar para influenciar nos resultados;
- Jogador relacionado ao resultado: jogadores estão relacionados aos resultados no sentido de que serão vencedores e "felizes" se um resultado positivo acontecer, e perdedores e "infelizes" caso um resultado negativo aconteça;
- Consequências negociáveis: o mesmo jogo pode ser executado com ou sem consequências na vida real.

Nesse sentido, um jogo eletrônico pode ser entendido como aquele que transfere essas características presentes nos jogos para o meio virtual, seja ele um computador, um celular ou um console. Deve-se ressaltar, contudo, que as características citadas acima podem não estar presentes em todos os jogos, mas podem ser compartilhadas pela maioria deles.

Da mesma forma, um jogo educativo seria aquele que utilizaria dessas características comuns dos jogos para atingir um fim educacional. De fato, conforme Tarouco (2004, p. 2) jogos educativos são "todas aquelas aplicações que puderem ser utilizadas para algum objetivo educacional ou estiverem pedagogicamente embasadas".

Deve-se enfatizar que, no presente trabalho, o termo “jogo educativo” será utilizado como sinônimo de “jogo educacional” em que ambos referem-se ao jogos no meio eletrônico.

De acordo com Gros (2007, p. 19), muitas pessoas enxergam o aprendizado como uma obrigação para as crianças, fazendo “com que a aprendizagem seja encarada como um trabalho” e consequentemente gerando desinteresse no aluno. Dessa forma, Tarouco (2004) afirma que os jogos são um elemento que podem contribuir para o “resgate” do interesse dos alunos, mudando assim a maneira como eles encaram o aprendizado.

Nesse sentido, Neto e Fonseca (2013, p.2) defendem que o ato de jogar, sendo o jogo educativo ou não, é importante para o aluno, principalmente durante a infância. Para eles, “jogar é considerado uma importante atividade para o desenvolvimento psicológico, social e cognitivo.”

Como uma subcategoria de *software* educativo, os jogos têm uma ligeira vantagem quando comparados a outros tipos de *software*. Além de elementos gráficos, sonoros e outras multimídias que por si só atraem a atenção dos alunos, eles possuem algumas particularidades que se destacam. Na maioria deles é possível encontrar elementos como desafios e objetivos a serem cumpridos que, por sua vez, estão inseridos dentro um enredo envolvente. Essa capacidade de prender a atenção do aluno é o que torna a utilização dos jogos educativos nas salas de aula tão interessante. Referindo-se a jogos educativos, Neto (2012, p. 2) afirma: “a utilização dos mesmos na educação pode proporcionar ao aluno motivação, estimulando também hábitos de persistência no enfrentamento de desafios e desenvolvimento de tarefas”.

Entretanto, um cuidado especial deve ser tomado durante o desenvolvimento desse tipo de *software*. A atratividade não pode tirar o foco do jogador de tal maneira que ele jogue “apenas” pelo prazer e diversão proporcionados pelo jogo. Segundo Neto (2012), os jovens podem passar horas em um jogo e, assim, sendo ele

educativo, essas horas gastas, além de divertir, precisam atingir o fim principal para o qual ele foi desenvolvido: o de educar. Como Tarouco (2004, p. 3) explica, o "[...]grande desafio é apoiar o aluno para que sua atenção não seja desviada somente para a competição, deixando de lado os conceitos a serem desenvolvidos".

Embora considere os jogos uma ferramenta eficiente para a educação, Tarouco faz uma ressalva importante com relação ao seu uso:

> Todavia, é importante ressaltar a idéia de que o uso de recursos tecnológicos, dentre eles o jogo educacional, não pode ser feito sem um conhecimento prévio do mesmo e que esse conhecimento deve sempre estar atrelado a princípios teórico-metodológicos claros e bem fundamentado. Daí a importância dos professores dominarem as tecnologias e fazerem uma análise cuidadosa e criteriosa dos materiais a serem utilizados, tendo em vista os objetivos que se quer alcançar (TAROUCO, 2004, p. 2).

Atualmente, com a expansão do conhecimento que se tem acerca de jogos educativos e seu potencial como ferramenta educadora, é possível encontrar diversos tipos de jogos comerciais no mercado. A maioria deles, entretanto, não foi construído tendo como foco alguma dificuldade específica de aprendizado, como, por exemplo, a leitura e a escrita. Isso, segundo Lima (2010, *apud* Guardiola, 1998), pode fazer com que os resultados esperados pelo professor não sejam alcançados. Nesse sentido, Savi (2008, p. 2) afirma a necessidade de os jogos educativos apresentarem "objetivos de aprendizagem bem definidos e ensinar conteúdo das disciplinas aos usuários".

É importante enfatizar que, embora alguns jogos possam não ter um objetivo educacional específico eles ainda possuem sua importância pedagógica, como já foi abordado neste trabalho. A questão aqui é que, com um objetivo de aprendizado mais específico, é possível analisar o desenvolvimento da criança com relação ao conteúdo abordado no jogo, bem como possíveis intervenções que precisam ser feitas para que o ensino do conteúdo abordado seja mais eficaz.

Dependendo de como a interação do jogo é construída, a criança pode aprender de forma mais independente, em relação à figura do professor. "Os jogos educacionais se baseiam numa abordagem autodirigida, isto é, aquela em que o sujeito aprende por si só, através da descoberta de relações e da interação com o

software" (TAROUCO, 2004, p.2). Nesses casos, Falkembach (2006) afirma que o professor tem o papel de selecionar os *softwares* mais adequados para a abordagem dos conteúdos e que desenvolvam outras habilidades em seus alunos, tais como: curiosidade, atenção e criatividade.

Uma das questões observadas durante o desenvolvimento do Jogo de Memória Auditiva foi como manter um equilíbrio entre os momentos dedicados à aprendizagem e elementos que mantivessem as crianças jogando e interessadas no jogo. Assim, enquanto se divertiam estariam também aprendendo.

2.2.2 Jogos educativos disponíveis no mercado acerca do processamento auditivo

A seguir serão apresentados jogos educativos, com enfoque no processamento auditivo, que foram encontrados na literatura e nas pesquisas bibliográficas feitas durante a realização deste trabalho.

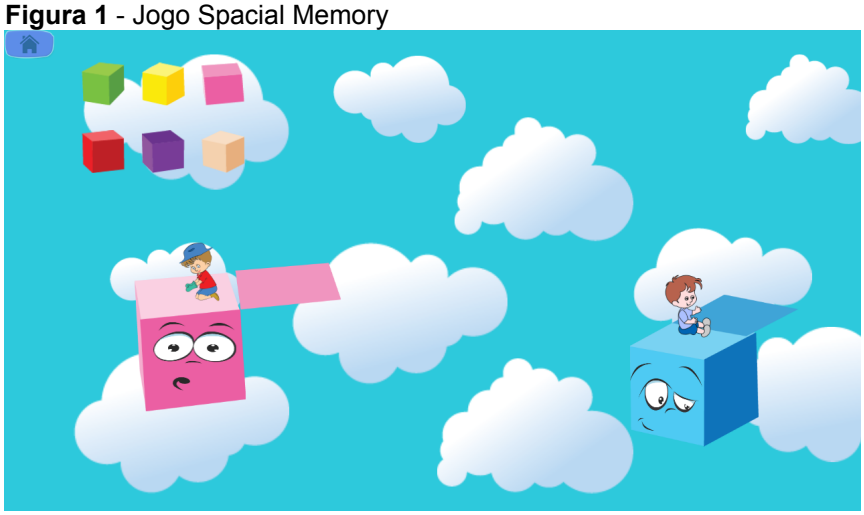
a) Memory Games for kids

Este é um jogo[1] em inglês que foi criado pela desenvolvedora de jogos conhecida como Shubi e está disponível para download gratuito na loja de aplicativos da Google. Ele é um jogo de memória que possui quatro tipos diferentes de modalidades: a) ***Spacial Memory***: dois personagens animados aparecem na tela; desaparecem e aparecem novamente, um deles é modificado e a criança precisa identificar qual deles mudou (Figura 1); b) ***Short-term memory***: novamente personagens animados aparecem em blocos de notas, na tela, e a criança precisa identificar qual deles aparece sozinho no bloco de notas maior (Figura 2); c) ***Working Memory***: a criança deve encontrar três diferenças entre as duas imagens que aparecem na tela do jogo (Figura 3); d) ***Auditory Memory***: cada um dos personagens que aparecem na tela emite um som e, em seguida, o jogo emitirá um

1 O jogo pode ser baixado no link https://play.google.com/store/apps/details?id=air.com.shubi.memoryEnglish

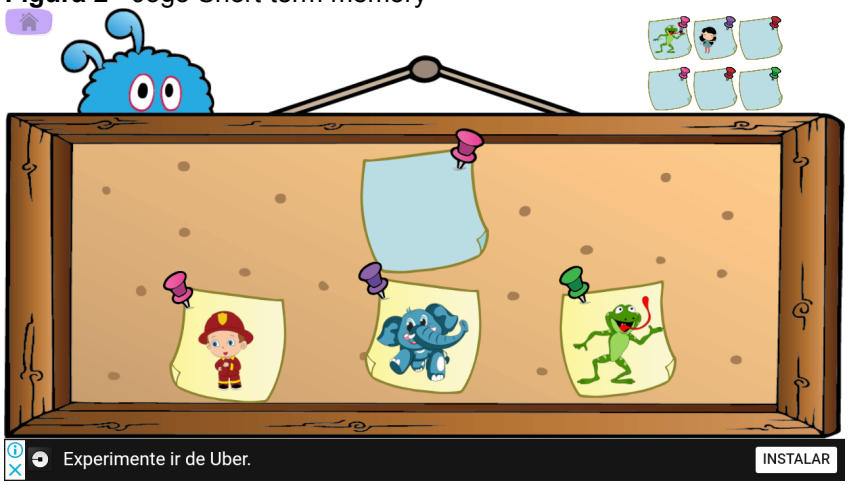
desses sons e solicitará que o jogador clique no personagem que emitiu esse som (Figura 4).

Figura 1 - Jogo Spacial Memory



Fonte: *print screen* do jogo feito no sistema operacional Android (2018)

Figura 2 - Jogo Short-term memory

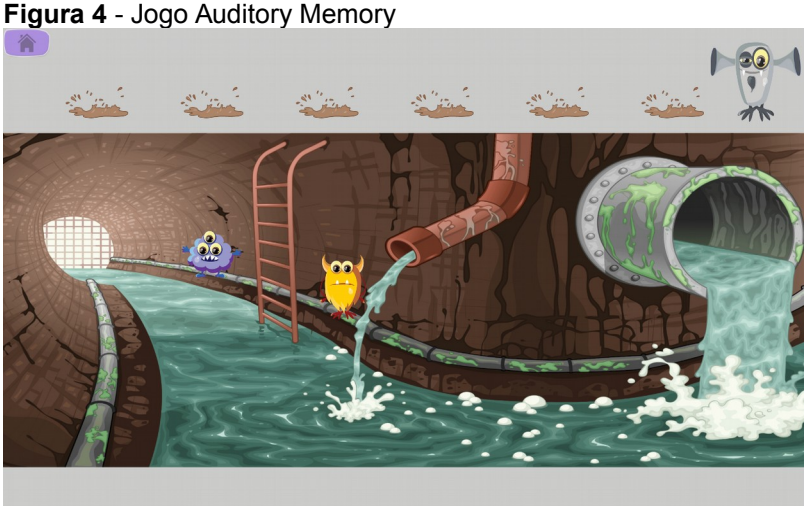


Fonte: *print screen* do jogo feito no sistema operacional Android (2018)

Figura 3 - Jogo Working Memory

Fonte: *print screen* do jogo feito no sistema operacional Android (2018).

Figura 4 - Jogo Auditory Memory



Fonte: *print screen* do jogo feito no sistema operacional Android (2018).

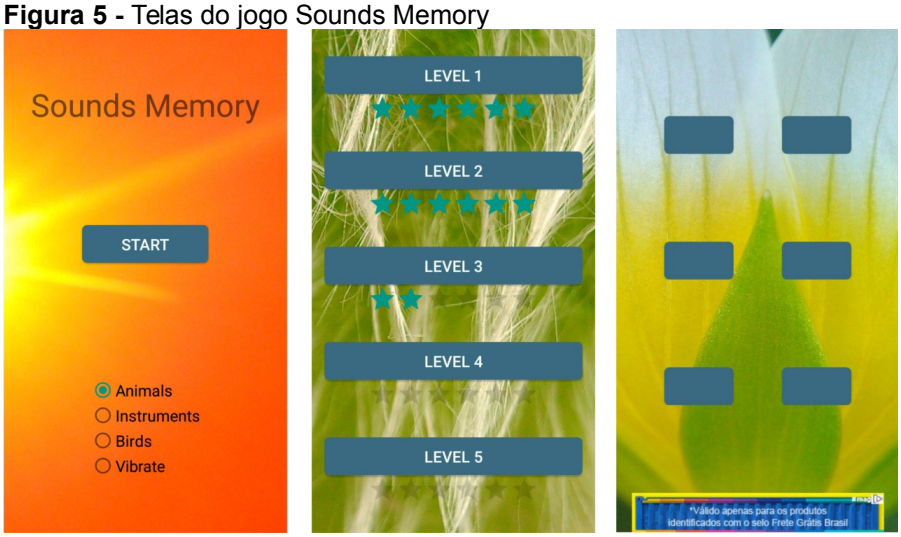
Em todas as modalidades do jogo nenhum tipo de pontuação é apresentado ao jogador, independentemente de quantas vezes o jogador acerte. Como foi citado acima, os jogos possuem algumas características que prendem a atenção dos jogadores e a falta de pontuação influencia diretamente no interesse do usuário pelo jogo. Para evitar esse tipo de problema, no Jogo de Memória Auditiva construiu-se um sistema de pontuação com o objetivo de motivar o jogador para que ele permaneça jogando.

Além disso, uma desvantagem do *Memory Games for Kids* é que ele não está disponível em Língua Portuguesa, embora possua uma versão para diversas outras línguas na loja de aplicativos do Google.

b) Sounds Memory

Esse jogo também está disponível gratuitamente da loja de aplicativos do Google[2]. Segundo a descrição do jogo na loja, ele é indicado para crianças de 5 a 12 anos e apresenta quatro tipos diferentes de jogos de memória, com 5 níveis de dificuldade em cada tipo. Os quatro tipos de jogos são: *Animals*, *Instruments*, *Birds* e *Vibrations*, como pode ser visto na (Figura 5). Essa classificação feita pelo desenvolvedor ficou um tanto quanto confusa, tendo em vista que a categoria *Birds* (pássaros, em português) é uma categoria diferente de *Animals* (Animais, em português), deixando a entender que pássaros não são animais.

Figura 5 - Telas do jogo Sounds Memory



Fonte: *print screen* do jogo feito no sistema operacional Android (2018).

O objetivo do jogo é formar pares dos sons que são ouvidos ao se clicar em um dos diversos botões que aparecem na tela. Os níveis de dificuldade são caracterizados pela quantidade de pares que o jogador necessita fazer para terminar o nível. Toda vez que o jogador escolhe um nível e o finaliza, é redirecionado para a tela inicial do jogo, o que torna a jogabilidade estranha e confusa. Além disso, todos os níveis que já foram jogados podem ser identificados pelas estrelas em cor azul, indicando, indiretamente, qual a pontuação obtida naquele nível.

Uma variação interessante do jogo é a *Vibrations*. Escolhendo essa variação, ao invés de sons, o jogador deve formar pares de vibrações que o jogo irá produzir

2 O jogo pode ser baixado através do endereço <https://play.google.com/store/apps/details?id=jursoft.mind.game.memory.vibrate.vibratememory>

no celular. Como informado pelo desenvolvedor, essa variação permite que o jogo de memória seja jogado por crianças com dificuldades auditivas, pois ela perceberá as vibrações através do toque no celular.

Um pouco diferente do jogo anterior, este apresenta a pontuação ao usuário, entretanto a forma como essa pontuação é ganha e como é possível melhorá-la não é intuitiva, o que o torna ainda mais confuso.

c) Sounds Essentials

Desenvolvido pela MeshTech Solutions[3], esse jogo está no catálogo de jogos na loja de aplicativos da *Google*. É um jogo de memória auditiva, indicado para para crianças entre 2 e 7 anos, que tem o objetivo principal de ajudar a melhorar as habilidades de discriminação auditiva e memória auditiva nos jogadores. Segundo a descrição do aplicativo na *Google Play*, o jogo foi desenvolvido em parceria com fonoaudiólogos, pais e professores.

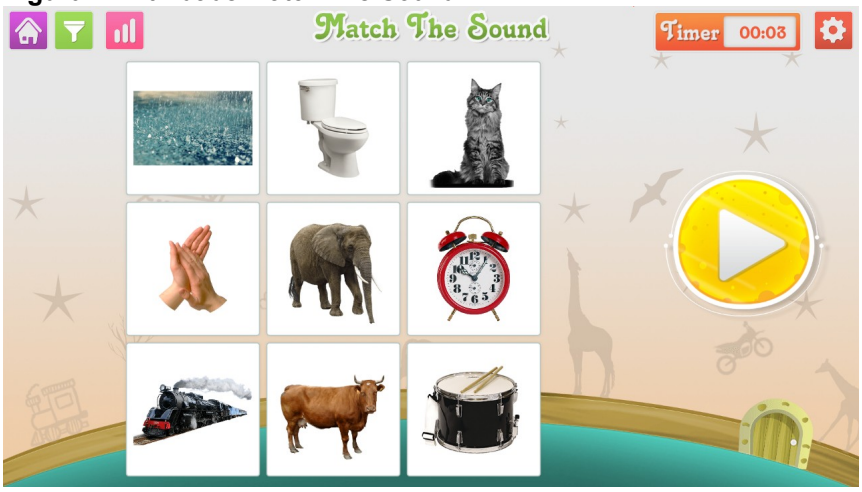
Ele está dividido em quatro atividades diferentes: a) ***Know The sound***: uma sequência de imagens é apresentada ao jogador para que ele reconheça o som que representa aquela imagem (Figura 6); b) ***Match The Sound***: a criança ouve um som e precisa clicar em uma das nove imagens correspondente àquele som. Quando a criança acerta, uma mensagem em inglês é executada (*"that's right"*). Do contrário, um som curto é executado, indicando que a criança não acertou (Figura 7);c) ***Sound Hunt***: nessa atividade a criança precisa arrastar a imagem que aparece para um dos três espaços (*nature*, *transport*, e *animals*) que aparecem na tela. A desvantagem dessa atividade em relação às outras é que a criança precisa saber ler (em inglês) para realizar algumas das atividades (Figura 8); d) ***Find The Sound***: é similar à atividade anterior, com a diferença de que aparecem apenas 3 imagens (Figura 9).

3 Outros jogos educativos do desenvolvedor podem ser acessados no endereço https://play.google.com/store/apps/dev?id=7432344024775399416

Figura 6 - Atividade Know The Sound

Fonte: *print screen* do jogo feito no sistema operacional Android (2018)

Figura 7 - Atividade Match The Sound

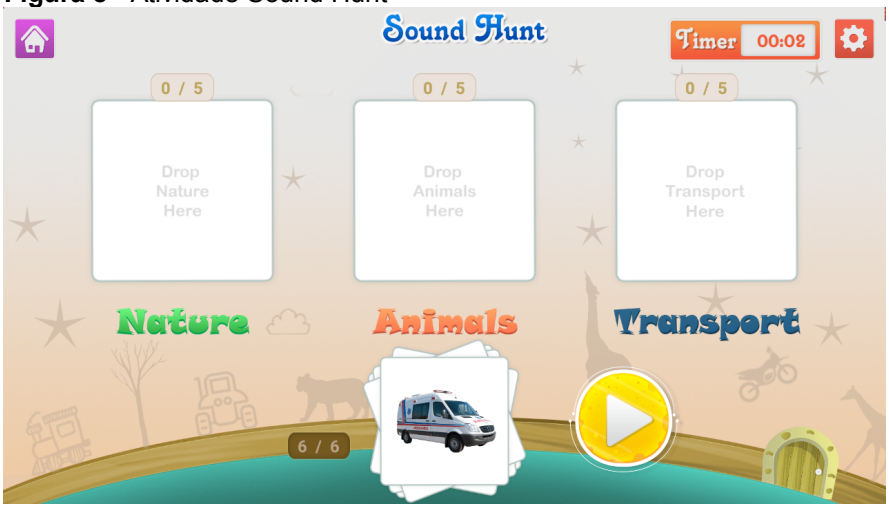


Fonte: *print screen* do jogo feito no sistema operacional Android (2018).

Em todas as atividades do jogo (exceto a *Know The Sound*, em que não há acertos nem erros) quando uma partida é finalizada uma pequena janela é mostrada na tela, com a pontuação do jogador e as opções de salvar ou não a pontuação. Caso o jogador escolha salvar, sua pontuação aparecerá num *ranking* de pontuações juntamente com as pontuações de outros jogadores.

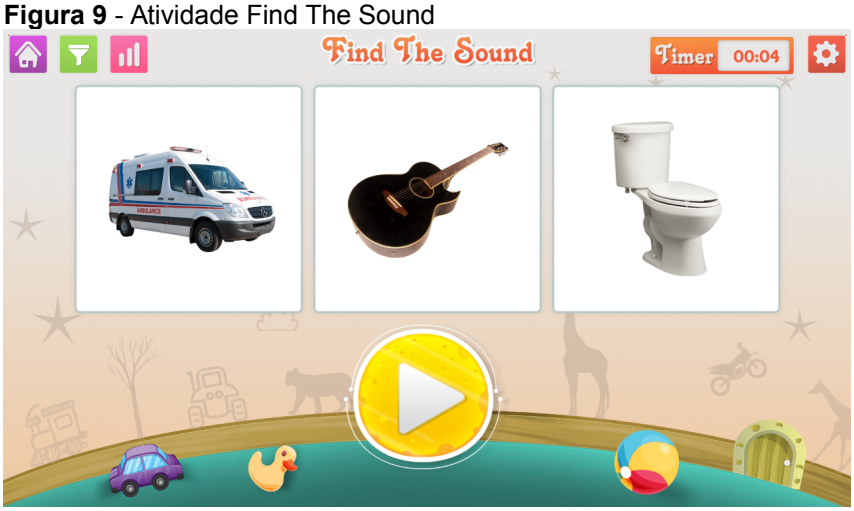
Embora exista seis tipos diferentes de sons (animais, máquinas, natureza, ações, transporte e instrumentos) a quantidade desses sons que aparecem na versão grátis do jogo é limitada, deixando-o um pouco repetitivo. Além disso, alguns sons, como por exemplo o que representa a sirene da ambulância, são muito longos.

Figura 8 - Atividade Sound Hunt



Fonte: *print screen* do jogo feito no sistema operacional Android (2018)

Figura 9 - Atividade Find The Sound



Fonte: *print screen* do jogo feito no sistema operacional Android (2018).

d) Training with Phonak

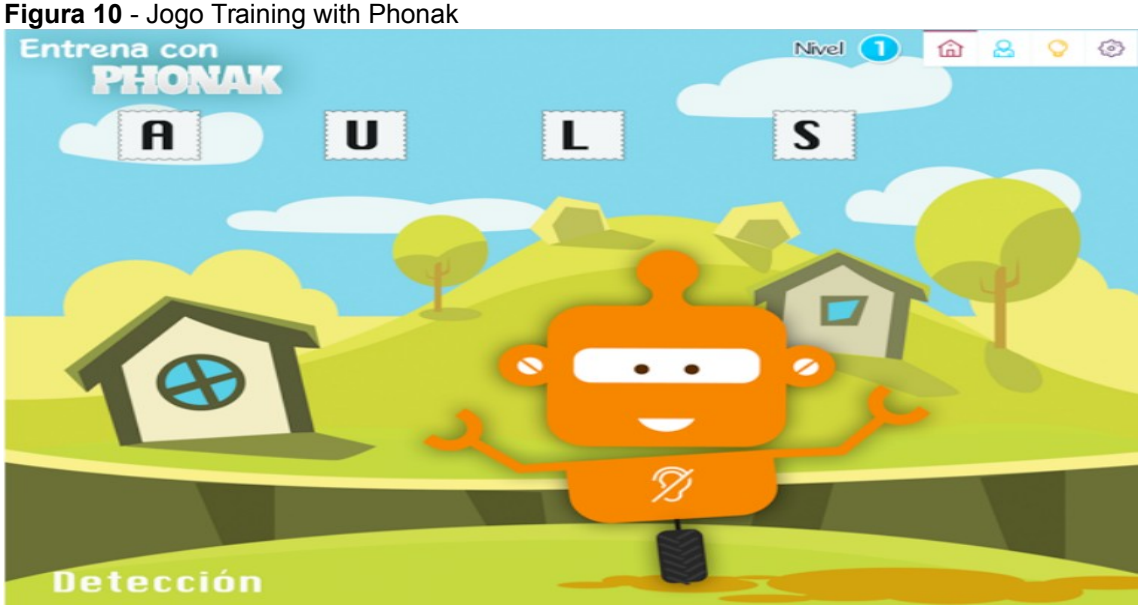
Training With Phonak é um jogo para dispositivos *tablets,* que possuem o sistema operacional Android, apresentado por Cano *et al*. (2015), que foi desenvolvido para crianças com implantes cocleares. Segundo Cano *et al*. (2015), crianças com esse tipo de implante precisam aprender a ouvir, por isso, o jogo foi desenvolvido tendo como base os cinco estágios no processo de aprendizado e reabilitação: detecção, discriminação, identificação, reconhecimento e compreensão.

O enredo conta a história do personagem Phonak, que está perdido no planeta *Sounds* e precisa aprender a ouvir para reencontrar sua família. Para que o

personagem possa progredir no jogo e consiga seu objetivo, ele necessita viajar para outros planetas, representados pelos estágios citados acima, e desenvolver habilidades que o ajudarão a adquirir o poder de ouvir.

Durante os cenários a criança deve ouvir vários sons e reproduzir esses sons através de um microfone, como pode ser visto na Figura 10, que representa o primeiro cenário do jogo e também o primeiro estágio (detecção) no processo de reabilitação.

Figura 10 - Jogo Training with Phonak



Fonte: Cano *et al*. (2015).

e) GraphoGame

Como informado pelo próprio site do jogo[4], o *GraphoGame* foi criado inicialmente para ser uma ferramenta de intervenção ao tratamento de crianças com dislexia, entretanto percebeu-se que poderia também beneficiar outras crianças auxiliando-as no desenvolvimento de suas habilidades básicas de leitura.

Nesse sentido, Ojanen *et al*. (2015) afirmaram em seu estudo que a relação entre a linguagem falada e o sistema de escrita tem um efeito considerável na taxa de aprendizagem da alfabetização. Além disso, a quantidade de fonemas do sistema de escrita bem como a quantidade de combinações possíveis estão intimamente

4 Mais informações sobre o jogo podem ser encontradas no site: <https://graphogame.com/>

relacionadas com a taxa de aprendizado.

O alfabeto da Finlândia, país de origem do *GraphoGame*, é, segundo Ojanen *et al*. (2015) completamente transparente, ou seja, as associações entre as letras e os sons produzem sempre uma pronúncia correta. Em outros idiomas, entretanto, uma mesma letra pode ter várias pronúncias diferentes, dependendo de onde a letra se encontra na palavra. Desse modo, Ojanen *et al*. (2015) realizaram um estudo em alguns países da África, a fim de verificar se o *GraphoGame* poderia ser utilizado em outras línguas com uma transparência similar à da Finlândia. O estudo mostrou resultados promissores acerca da utilização do jogo nesses países, apesar de os desafios encontrados, como a falta de motivação de pais e professores, prejudicarem a eficácia do jogo.

Uma versão na língua Finlandesa do *GraphoGame* está disponível na *Google Play*, com o nome de Ekapeli Alku[5]. Na primeira tela do jogo é possível realizar *login*, para que o progresso seja salvo no sistema e o jogador possa ter seus dados atualizados em qualquer dispositivo em que esteja jogando, ou jogar sem fazer *login*, em que o progresso é salvo apenas no dispositivo que está executando o aplicativo.

Ao iniciar, o jogador escolhe um avatar e dá um nome a ele. Na próxima tela, o avatar encontra-se em uma ilha com caminhos que levam a diferentes lugares, em que é possível ver várias estrelas (Figura 11).

Figura 11 - Primeiro cenário do jogo GraphoGame

Fonte: *print screen* do jogo feito no sistema operacional Android (2018).

5 O jogo pode ser baixado no link <https://play.google.com/store/apps/details?id=fi.ekapeli.EkapeliAlku>

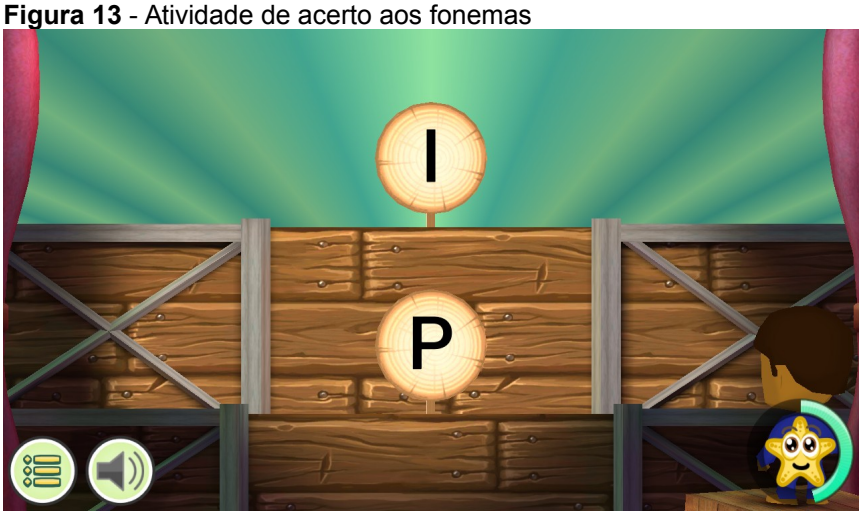
Algumas dessas atividades consistem em ouvir os sons das palavras (Figura 12) ou acertar as letras que aparecem na tela (Figura 13). Essas atividades podem se repetir ou não em determinadas estrelas durante o caminho percorrido pelo jogador.

Figura 12 - Atividade de reconhecimento dos fonemas

Fonte: *print screen* do jogo feito no sistema operacional Android (2018).

No final de cada atividade uma tela contendo as letras do alfabeto aparece para o jogador. Nela é possível ver quais letras o jogador já aprendeu e quais já apareceram em alguma atividade anterior mas, por conta das dificuldades do jogador, ele precisa reforçar (Figura 14). Ainda nessa tela é possível clicar nas letras e ouvir a forma como elas são pronunciadas.

Figura 13 - Atividade de acerto aos fonemas



Fonte: *print screen* do jogo feito no sistema operacional Android (2018).

Independentemente da atividade que está sendo feita, o jogador apenas consegue finalizá-la depois de 5 acertos. Os erros durante a atividade não influenciam nos acertos, apenas na pontuação final do jogador. Entretanto, percebe-se que quanto mais acertos, mais difícil se torna o jogo, pois o número de possibilidades é maior. Da mesma forma, quando ocorre um erro, o nível do jogo diminui, com menos possibilidades aparecendo.

Figura 14 - Tela final com as letras aprendidas

Fonte: *print screen* do jogo feito no sistema operacional Android (2018).

Dentre os jogos citados este foi mais completo que se pode analisar, porém, assim como os outros, não está disponível em Língua Portuguesa, o que impossibilita o seu uso em escolas brasileiras. Sucena *et al*. (2016), entretanto, empreenderam um estudo que busca a adaptação do *GraphoGame* para a nossa língua.

2.2.3 Design de software para crianças

Uma das questões analisadas durante o desenvolvimento do Jogo de Memória Auditiva foi como construir uma interface intuitiva o bastante para que crianças que não soubessem ler pudessem entender, em pouco tempo, o funcionamento do jogo. Esta questão surgiu durante a concepção da ideia geral do jogo, em que ficou decidido que ele seria destinado a crianças entre 6 e 7 anos de idade.

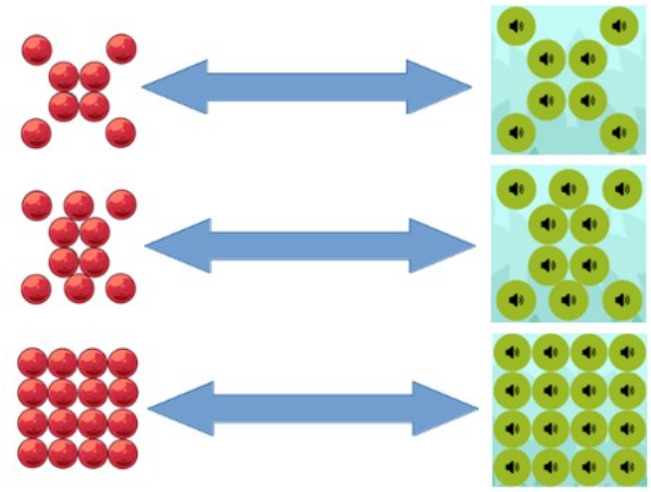
Segundo Hanna, Risden e Alexander (1997), embora o design de interfaces destinada a adultos esteja bem definido na literatura desde a década de 1990, a pesquisa na área de criação de interfaces para crianças teve seu início apenas nas duas últimas décadas.

De acordo com Druin, Bederson e Hourcade (2001), o uso de textos em *softwares* deve ser minimizado, particularmente para crianças que não sabem ler ou que começaram a ler recentemente. Por isso, tentou-se construir uma interface para o Jogo de Memória Auditiva com a menor quantidade de informação textual possível, pois os usuários principais do jogo estão em processo de alfabetização e supõe-se que eles não entenderiam (pelo menos não nas primeiras vezes que jogassem) os textos adicionados na interface.

A partir dos três anos de idade, “a maioria das crianças entendem que um símbolo representa outra coisa, que algo pode ser tanto um objeto e um símbolo, e que um símbolo pode representar algo no mundo real” (DeLouche, 1999, *apud*, HOURCADE, p. 21). Em um jogo para jogadores mais velhos e alfabetizados, simplesmente nomear os níveis pelo seu grau de dificuldade seria o bastante para que o jogador percebesse que o “Nível 10” é mais difícil que o “Nível 2”. Entretanto, para uma criança em processo de alfabetização isso não é tão trivial. A partir disso, tentou-se criar um ambiente em que a criança pudesse associar os ícones à algum significado dentro do jogo. Por exemplo, a fim de representar a diferença de dificuldade entre os três níveis presentes no jogo em desenvolvimento, preferiu-se criar três ícones diferentes para cada um dos níveis, de forma que cada ícone estivesse associado à quantidade de botões clicáveis do jogo de memória, como mostra a Figura 15.

Os ícones dos grupos foram representados por “casas” em formato de cogumelo e os grupos já finalizadas foram representadas pela mesma casa com um “X” na porta (Figura 16), indicando que o jogador não pode mais acessar esse grupo. Além disso, para reforçar a ideia de que o grupo ou nível não pode ser mais acessado, o clique com o mouse nos grupos e níveis finalizadas é impedido, para que a criança tenha a oportunidade e a motivação de conhecer novos fonemas.

Figura 15 - Ícones dos níveis e a quantidade de botões a eles relacionados



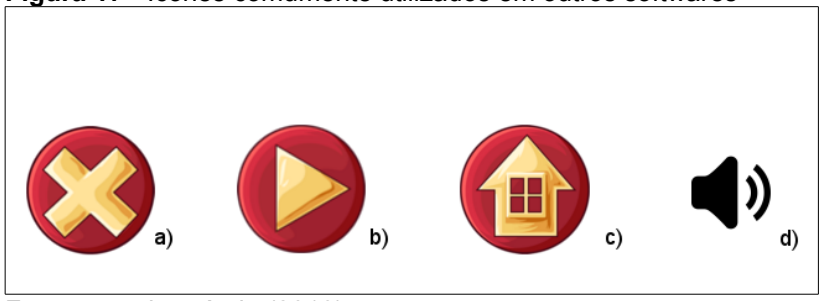
Fonte: autoria própria (2018).

Figura 16 - Ícones dos grupos

Fonte: autoria própria (2018).

Em relação aos outros ícones presentes no jogo, preferiu-se utilizar aqueles comumente utilizados em outros jogos e *softwares* de maneira geral, como, por exemplo, o botão de fechar, o botão de iniciar, o botão que leva ao menu inicial e ícones que representam um som (Figura 17).

Figura 17 - Ícones comumente utilizados em outros softwares



Fonte: autoria própria (2018).

. Uma outra preocupação durante o desenvolvimento do Jogo de Memória Auditiva estava relacionada ao fato de que as crianças precisam saber manipular um mouse de computador para poder jogar. Segundo Hourcade (2008) alguns movimentos das mãos tem seu início de desenvolvimento entre três e sete anos de idade. Em outro estudo Hourcade *et al*. (2004), descobriram que a maioria das crianças entre quatro e cinco anos utiliza o botão esquerdo do mouse para a realização de tarefas (normalmente esse é o botão padrão para a realização da maioria das tarefas em qualquer tipo *software* de qualquer sistema operacional). Dessa forma, supõe-se que as crianças ao utilizarem o jogo, podem ter uma dificuldade a princípio, mas se adaptariam enquanto estivessem jogando.

Ainda sobre a utilização do mouse, Hourcade (2008) afirma que crianças entre sete e doze anos podem ter dificuldades ao clicar em botões ou ícones muito pequenos, e, por isso, esses elementos precisam ter um tamanho consideravelmente maior do que um ícone de uma interface para um adulto.

Com relação à forma como o jogo responde às ações do jogador, Hourcade (2008, p. 316) afirma que ações rápidas são muito importantes em interfaces para crianças. Segundo o autor, "crianças precisam de *feedbacks* rápidos, e se elas não o tiverem, provavelmente irão fazer outra atividade."

Como foi dito anteriormente, é importante, durante o desenvolvimento de jogos educativos, ter-se o cuidado para que a diversão presente no jogo não desviar o foco da criança dos conteúdos que se deseja trabalhar. De fato Fisch (2005) afirma, em relação a *softwares* educativos, que o ato de jogar deve ser baseado diretamente no conhecimento ou habilidades que o jogo foi projetado para desenvolver em seus usuários. No Jogo de Memória Auditiva, a criança conseguirá obter maiores pontuações apenas se ouvir atentamente os fonemas que são executados, ou seja, ela precisa realizar corretamente a formação dos pares de fonemas caso queira manter uma boa pontuação no jogo e, consequentemente, uma boa posição no *ranking*.

3 DELINEAMENTO DO ESTUDO

Este capítulo consiste da descrição dos objetivos gerais e específicos do presente estudo, das etapas executadas para o entendimento do jogo, da metodologia de desenvolvimento e das ferramentas que foram utilizadas no seu desenvolvimento, e dos testes que foram realizados.

3.1 OBJETIVOS

3.1.1 Objetivo geral

Este trabalho tem por objetivo a construção de um jogo educativo, denominado Jogo de Memória Auditiva, com enfoque na melhora da memória de trabalho, especialmente a do processamento auditivo, que possa ser utilizado como uma ferramenta complementar para a alfabetização de crianças no anos iniciais da alfabetização.

3.1.2 Objetivos específicos

A seguir tem-se os objetivos específicos que visamos alcançar com este trabalho:

- Revisar a literatura científica acerca do processamento auditivo, em especial a memória de trabalho e seu subcomponente fonológico e sua relação com a aquisição e aprendizado inicial da leitura e escrita;
- Revisar a literatura científica acerca do desenvolvimento de softwares para crianças;
- Planejar a lógica e a interface do Jogo de Memória Auditiva;
- Escolher as tecnologias a serem utilizadas na implementação do jogo;
- Testar a funcionalidade do jogo

3.2 MATERIAIS E MÉTODOS

Para o entendimento da ideia inicial do Jogo de Memória Auditiva, algumas etapas foram necessárias. Inicialmente foram realizadas reuniões com o Coorientador da pesquisa e idealizador do projeto, a fim de se entender quais os requisitos básicos do jogo e a viabilidade de sua execução no período de tempo que foi estipulado (cerca de 6 meses).

Em seguida analisou-se alguns jogos educativos disponíveis no mercado, mais especificamente os de memória ou que de alguma forma trabalhassem a memória auditiva do jogador. Essas análises possibilitaram uma melhor definição dos requisitos do jogo que haviam sido descritos, a adição de alguns outros (também importantes na construção de um *software* educativo), melhorias que poderiam ser feitas tendo como base os *softwares* analisados e a definição da estrutura do jogo aqui desenvolvido. O design inicial do jogo pode ser visto nas figuras do Apêndice A.

Na terceira etapa realizou-se uma análise da literatura acerca do desenvolvimento de *softwares* e jogos destinados à crianças, principalmente com relação aos requisitos a serem observados durante construção das interfaces e de como os jogadores iriam interagir com elas.

Na última etapa analisou-se as metodologias de desenvolvimento de *software* que poderiam ser utilizadas e escolheu-se a que mais se adequaria à nossa realidade. Tendo em vista o prazo estimado para o desenvolvimento do jogo, o nível de complexidade do projeto, os requisitos elicitados e o número de desenvolvedores (nesse caso, apenas um), definiu-se a metodologia e as ferramentas a serem utilizadas no desenvolvimento do jogo, que serão descritas a seguir.

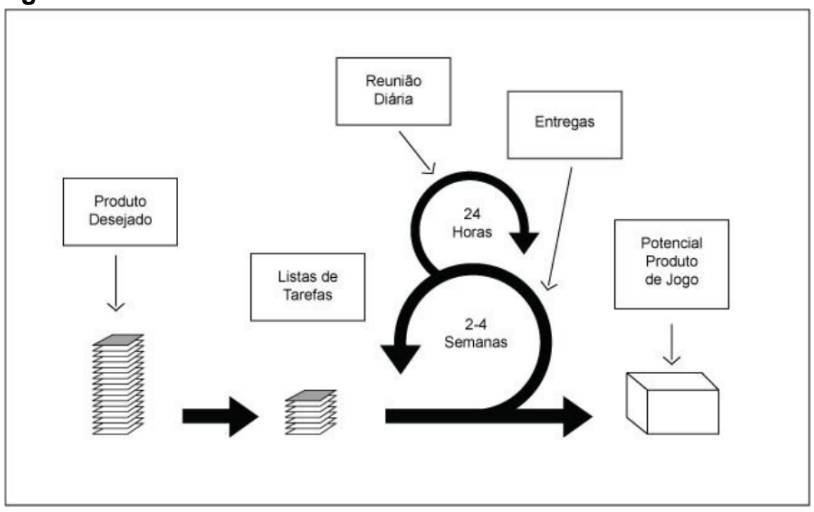
3.2.1 Metodologia de desenvolvimento

Como metodologia de desenvolvimento, tendo em vista a possibilidade de que, mesmo com o projeto inicial, os requisitos do jogo pudessem ser alterados, decidiu-se adotar a metodologia conhecida como Scrum, que foi adaptada para o desenvolvimento de jogos por Keith (2010). Essa metodologia é indicada tanto para grandes projetos quanto aos menores, pois possibilita a adaptação às mudanças de necessidades e novos requisitos que possam surgir durante a implementação do *software* (BISSI, 2007).

Resumidamente, o Scrum é um método para gerenciamento e controle de *software* e desenvolvimento de produtos em ambientes com mudanças repentinas em que os processos são realizados de maneira iterativa e incremental (CERVONE, 2011).

Na Figura 18 é possível visualizar as principais etapas que compõem o ciclo de vida de desenvolvimento de jogos com Scrum. Deve-se ressaltar que dada a situação de desenvolvimento do jogo (em relação o número de desenvolvedores) algumas tarefas da metodologia Scrum, como por exemplo, a reunião diária, que tinham o objetivo de atualizar todos os membros da equipe sobre o andamento do desenvolvimento do *software*, não foram realizadas.

Figura 18 - Ciclo de vida do Scrum



Fonte: Aymone (2015).

Em seu estudo, Keith (2010) listou as principais etapas do Scrum:

- **Product Backlog** (Produto desejado)**:** é uma lista de requisitos e funcionalidades, ordenadas pela prioridade, que um jogo deve possuir;
- **Sprint Backlog** (Lista de tarefas)**:** tarefas a serem realizadas para que um requisito, da lista de requisitos do produto desejado, seja entregue;
- **Daily Scrum** (Reunião diária)**:** reunião diária de 15 minutos feita com todos os membros da equipe a fim de identificar problemas e garantir a sincronia entre a equipe.
- **Sprint** (Entregas)**:** entregas de atualizações do *software* que são feitas a cada 2 ou 4 semanas;
- **Potentially Product Game** (Potencial Produto de Jogo): no final de cada entrega tem-se uma versão do jogo que já pode ser utilizada.

Assim, a primeira etapa do desenvolvimento do jogo após o entendimento do seu funcionamento geral foi a criação do *Product Backlog* inicial do produto:

- O jogo deve ter 7 grupos e 3 tipos de níveis possíveis para cada grupo;
- O jogo deve oferecer um tempo diferente para alguns grupos e para cada nível
- O jogador pode iniciar uma partida;
- O jogador pode escolher o avatar;
- O jogador pode atribuir nome ao avatar;
- O jogador pode trocar de avatar;
- O jogador pode editar o nome do avatar;
- O jogador pode iniciar jogo a partir do *ranking*;
- O jogador pode sair do jogo;
- O jogador pode iniciar o próximo grupo;
- O jogador pode reiniciar grupo finalizado;
- O jogador pode reiniciar o jogo finalizado;
- O jogador não pode acessar um grupo finalizado;
- O jogador não pode acessar um nível finalizado;
- O jogador pode ver a pontuação de cada fase;
- A cada acerto, as cores dos dois botões clicados serão modificadas;
- O jogador receberá um *feedback* por cada acerto;

- O jogador receberá um *feedback* de término de grupo;
- O jogador receberá um *feedback* à cada erro;
- O jogador receberá um *feedback* quando terminar um nível;
- A pontuação total do jogador deverá ser salva;
- O jogador verá o *ranking* atualizado caso termine o jogo ou quando perder;
- O jogador pode ir para o menu inicial;
- O jogador pode pausar o tempo;
- O jogador pode executar o vídeo demonstrativo do jogo;
- O jogador pode pular o vídeo demonstrativo

A segunda etapa consistiu de analisar o *Product Backlog* e criar os *Sprints Backlogs* a serem desenvolvidos até a entrega final do jogo. Como já informado acima, a etapa de *Daily Scrum* não era realizada, entretanto as entregas de novas atualizações do Jogo de Memória Auditiva eram feitas a cada 2 ou 3 semanas. Deve-se ressaltar que durante o desenvolvimento do jogo, diversos *Sprints Backlogs* foram feitos, e que para uma melhor organização deste trabalho, preferiu-se apresentar apenas as tarefas consideradas mais importantes, listadas no primeiro *Sprint Backlog* definido, que pode ser visto no Quadro 1 abaixo.

Quadro 1 - Primeiro *Sprint Backlog* do jogo.

Tarefa	Prioridade
O jogo deve ter 7 grupos e 3 tipos de níveis possíveis para cada grupo	Urgente
O jogador pode iniciar uma partida	Urgente
O jogador pode iniciar uma partida	Desejável
O jogador pode atribuir nome ao avatar	Desejável
O jogador pode ir para o menu inicial	Desejável
O jogador pode sair do jogo	Urgente

Fonte: autoria própria (2018).

A terceira etapa foi a implementação do jogo em si que, dada a característica cíclica da metodologia utilizada, se repetiu até a sua finalização. No Quadro 2 é possível ver as principais tarefas que foram elencadas para serem desenvolvidas e

quais delas foram entregues ou não durante o desenvolvimento do jogo.

Quadro 2 - Tarefas e estado da implementação

Tarefa	**Implementada**
O jogo deve ter 7 grupos e 3 tipos de níveis possíveis para cada grupo	Sim
O jogo deve oferecer um tempo diferente para alguns grupos e para cada nível	Sim
O jogador pode iniciar uma partida	Sim
O jogador pode escolher o avatar	Sim
O jogador pode atribuir nome ao avatar	Sim
O jogador pode trocar de avatar	Não
O jogador pode editar o nome do avatar	Não
O jogador pode iniciar jogo a partir do *ranking*	Sim
O jogador pode sair do jogo	Sim
O jogador pode iniciar o próximo grupo	Sim
O jogador pode reiniciar grupo finalizado	Sim
O jogador pode reiniciar o jogo finalizado	Sim
O jogador não pode acessar um grupo finalizado	Sim
O jogador não pode acessar um nível finalizado	Sim
O jogador pode ver a pontuação de cada grupo	Sim
A cada acerto, as cores dos dois botões clicados serão modificadas	Sim
O jogador receberá um *feedback* por cada acerto	Sim
O jogador receberá um *feedback* de término de grupo	Sim
O jogador receberá um *feedback* à cada erro	Sim
O jogador receberá um *feedback* quando terminar um nível	Sim
A pontuação total do jogador deverá ser salva	Sim
O jogador verá o *ranking* atualizado caso termine o jogo ou quando perder	Sim
O jogador pode ir para o menu inicial	Sim
O jogador pode pausar o tempo	Sim
O jogador pode executar o vídeo demonstrativo do jogo	Não
O jogador pode pular o vídeo demonstrativo	Não

Fonte: autoria própria (2018).

3.2.2 Tecnologias utilizadas

Todas as tecnologias utilizadas foram escolhidas com base em alguns requisitos definidos nas primeiras reuniões. O primeiro deles refere-se à restrição de utilização de *softwares* que possuíssem algum tipo de custo de utilização.

Ainda, o prazo estipulado para a entrega final do jogo motivou a tentativa de se maximizar o tempo disponível para o desenvolvimento, de modo que evitou-se a utilização de ferramentas desconhecidas, que iriam exigir um tempo de aprendizado e adaptação.

As tecnologias escolhidas para o desenvolvimento do jogo estão descritas a seguir.

a) Java e JavaFX

Tendo em vista a possibilidade do jogo precisar ser executado em diferentes sistemas operacionais, optou-se pela utilização da linguagem de programação Java. Um dos motivos para esta escolha é o fato de que programas criados com esta linguagem podem ser executados em diversos sistemas operacionais, desde que eles possuam o java instalado[6].

Dessa forma, o principal requisito para a execução do jogo seria ter-se a versão 8 ou superior do Java instalado no computador (a maioria dos computadores atuais possui o Java instalado para a execução de outros serviços).

Além disso, utilizou-se o JavaFX, uma biblioteca de código aberto escrita em Java que é usada para o desenvolvimento de interfaces gráficas, dos elementos gráficos, tais como botões, e multimídias do jogo. Além de fácil entendimento, essa biblioteca é baseada na linguagem de marcação XML[7] para a criação da interface, possibilitando a utilização de tecnologias como o *CSS*[8], que permite a criação de interfaces gráficas com uma aparência mais moderna e atrativa ao usuário.

---

6 Mais informações sobre a linguagem Java podem ser encontradas no site https://docs.oracle.com/javase/8/docs/technotes/guides/language/index.html

7 XML é uma linguagem de marcação similar à HTML, que foi desenvolvida para gravar e transportar dados

8 CSS é um linguagem utilizada para descrever a forma como os elementos serão exibidos para o usuário

b) Scene Builder

Assim como outras bibliotecas de interface gráfica, os elementos disponibilizados pela JavaFX poderiam ser definidos diretamente nas classes da linguagem java. Entretanto, a fim de se agilizar a construção e testes das interfaces criadas, optou-se por utilizar o *Scene Builder*.

O *Scene Builder* é um *framework*[9] que foi construído para o desenvolvimento de aplicações baseadas em JavaFX, e permite a criação de interfaces de maneira rápida e que podem ser facilmente integradas com as classes Java.

Com ele é possível definir, por exemplo, a posição, cores e tamanho de elementos na interface, além de que ele permite a adição de eventos a elementos tais como clique em botões e imagens e o reconhecimento dos movimentos do mouse.

Uma das vantagens de se utilizar o *Scene Builder* é que as propriedades dos elementos que são utilizados em cada interface ficam em arquivos (no formato .fxml) separados das classes onde esses elementos são utilizados, deixando o código mais claro, fácil de entender e modificar.

c) Audacity e GIMP

Como o jogo utiliza diferentes áudios na sua execução, foi necessário a utilização de uma ferramenta de edição de áudio para o tratamento dos áudios usados. Assim, escolheu-se o *software Audacity*, pois ele é um editor de áudio gratuito com todas as funcionalidades necessárias para as edições de áudios que precisaríamos realizar.

Uma das principais edições feitas com este *software* foi a redução dos tamanhos dos áudios. Isso fez-se necessário devido ao fato de que, durante a execução do jogo, quando um dos botões com os sons dos fonemas era clicado, ele bloqueava a execução do áudio de qualquer outro botão até que o primeiro áudio

9 Um *framework* é um software, normalmente com interface gráfica, que facilita a criação e reuso de códigos por meio da utilização de bibliotecas internas já testadas

terminasse de ser executado. Dessa forma, durante os testes iniciais do jogo percebeu-se que alguns sons não eram executados e, ao analisar-se a duração do tempo dos áudios, notou-se que em boa parte deles havia cerca de 1 segundo de duração em que nenhum som era emitido (esse tempo “em silêncio” foi causado durante a gravação dos áudios).

Além disso, utilizou-se o *Audacity* para edição dos áudios de *feedback* como acerto, fase finalizada, escolha de avatares e clique na botão de iniciar o jogo.

Quanto às imagens usadas, optou-se pelo *software* de edição de imagens GIMP que, assim como o *Audacity*, é uma ferramenta grátis e com todas as funcionalidades necessárias para o desenvolvimento do jogo aqui proposto. Com essa ferramenta editou-se a imagem de fundo do jogo, as imagens internas dos botões e as imagens presentes dentro dos *pop-ups*[10] apresentados durante o jogo.

c) *Netbeans* e Github

Como IDE[11] de desenvolvimento do jogo, escolheu-se o Netbeans, pois além de uma boa apresentação do código em desenvolvimento (que o tornava fácil de entender), o *Netbeans* possui uma boa integração com a linguagem Java e organiza os pacotes contendo as classes Java e pastas do jogo de uma maneira hierárquica que facilita a manutenção do código.

A fim de evitar problemas durante a codificação do jogo, utilizou-se a plataforma de versionamento conhecida como Github, que fornece uma variedade de funcionalidades, como por exemplo, a possibilidade de verificar todas alterações feitas no código desde o início do seu desenvolvimento e de reverter modificações feitas. Essa plataforma é interessante pois evita que eventuais erros que venham a acontecer no *software* atrasem a continuidade do seu desenvolvimento.

10 Pop-ups são janelas, que se abrem por cima da janela principal de um *software* ou site e normalmente são de um tamanho menor.
11 IDE (*Integrated Development Environment*) é um editor de texto que possui algumas funcionalidades que facilitam a implementação de programas de computador

e) Padrão *Model-View-Controller*

Com relação à organização de *software*, optou-se por utilizar o padrão *Model-View-Controller* (MVC). Segundo Sharan (2015) esse é o padrão mais antigo e utilizado para a modelagem de aplicações com interface gráfica, que é composto de três componentes: *model*, *view* e *controller*.

O primeiro diz respeito o domínio dos objetos que modelam os problemas do mundo real, ou seja, ele manipula as informações armazenadas em um banco de dados ou em arquivos do sistema. O segundo componente é responsável por exibir os dados gerenciados pelo *Model* e atualizar as suas informações sempre que algum dado for modificado. Ao componente *Controller* cabe a função de receber entradas, saídas e interações do usuário com os elementos da interface e decidir o que fazer com elas.

No presente trabalho, o *View* é representado pelos arquivos criados pelo JavaFX contendo os elementos da interface que serão apresentados na tela. Quando esses arquivos são criados, o *Netbeans* automaticamente cria os arquivos que representam o componente *Controller* do padrão MVC. Os componentes *Model*, por sua vez, precisam ser criado manualmente.

3.3 DESENVOLVIMENTO DO JOGO

3.3.1 Descrição geral

O *software* aqui desenvolvido, que foi denominado de “Jogo de Memória Auditiva”, é um jogo de memória para desktops indicado para crianças que estão nas fases iniciais de alfabetização (com idades entre 6 e 7 anos). Ele é composto de três níveis de dificuldade com sete grupos de fonema em cada nível, em que o jogador precisa formar pares desses fonemas, através de cliques com o mouse em botões contendo os fonemas. para poder obter pontuação e melhorar sua posição no *ranking* de jogadores.

3.3.2 Funcionamento do jogo

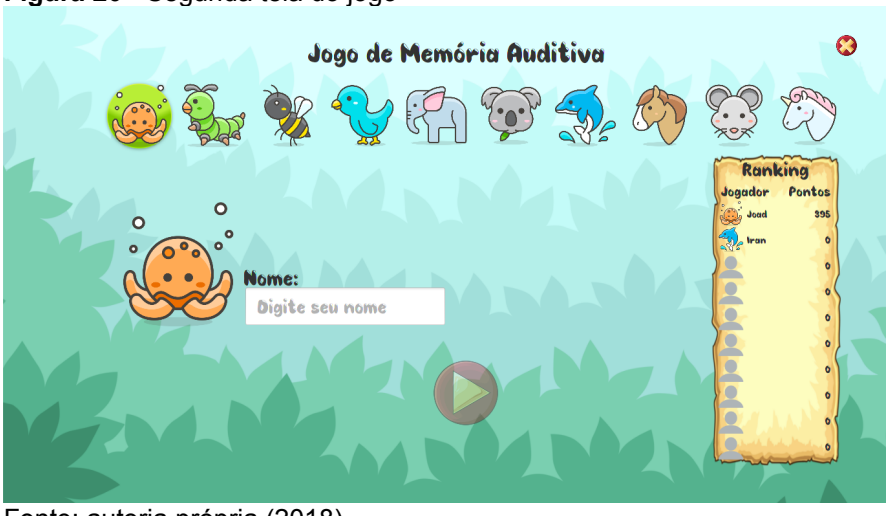
Na primeira tela do jogo (Figura 19), o jogador pode ir para a próxima tela, sair do jogo ou clicar no botão “Sobre” para ver as informações referentes aos direitos autorais e equipe de desenvolvimento do jogo.

Figura 19 - Tela inicial do jogo

Fonte: autoria própria (2018).

Na próxima tela (Figura 20) é possível escolher o avatar que representará o jogador, inserir o nome do avatar escolhido e visualizar a classificação atual no *ranking*.

Figura 20 - Segunda tela do jogo



Fonte: autoria própria (2018).

Para iniciar uma partida, um jogador pode seguir dois passos: no primeiro ele pode inserir um nome diferente dos nomes que já estão presentes no *ranking* e clicar no botão de *play* (que fica desabilitado até que um nome seja inserido). E no segundo, caso tenha jogado alguma vez, ele pode clicar na posição do *ranking* referente ao seu avatar.

Assim que inicia-se a partida é possível visualizar a tela principal do jogo (Figura 21). Inicialmente o jogador se encontrará no nível 1 e no grupo 1 (vogais) de fonemas. Cada grupo de fonemas possui um conjunto de fonemas diferentes, compostos pelas seguintes estruturas: a) Grupo 1: vogais; b) Grupo 2: sílabas simples (VC e CV); c) Grupo 3: palavras simples; d) Grupo 4: sílabas complexas I (CVC); e) Grupo 5: sílabas complexas II (CCV) ;f) Grupo 6: sílabas complexas III (CCVC[12]); g) Grupo 7: palavras complexas.

Figura 21 - Tela principal do jogo

Fonte: autoria própria (2018)

A dificuldade de cada nível é representada pelo número de pares corretos que o jogador precisa formar em cada nível, de modo que o jogador tem de formar 4 pares no primeiro, 5 no segundo e 8 no terceiro. As outras configurações de dificuldade do jogo, referentes à quantidade de botões, podem ser vistas no Apêndice B.

A pontuação total do jogador em cada grupo de fonemas será exibida acima do respectivo botão do grupo e é calculada utilizando-se as variáveis tempo, quantidades de cliques e pontuação obtida no grupo, como pode ser visto na Figura

12 Caso o leitor queira mais informações acerca da classificação das sílabas em Língua Portuguesa, sugerimos o livro de Linguagem Escrita e Alfabetização de Carlos Alberto Faraco, publicado em 2016 pela editora Contexto

22. Embora o jogador pontue em cada acerto que faça, sua pontuação final não será salva no *ranking* até que ele finalize o grupo atual (faça todos os pares possíveis) ou até que o tempo acabe. Ainda, a sua pontuação no *ranking* será atualizada apenas se a pontuação atual for maior que a anterior.

Como cada grupo apresenta quantidades de fonemas com um tempo diferente de duração e essa diferença influencia no tempo que o jogador terá disponível para finalizar o grupo, foi necessário atualizar-se a função *setPontuacaoJogador*() do *Model* PopUpModel.java (Figura 22) para que nos grupos com os fonemas maiores, o tempo gasto pelo jogador não prejudicasse sua pontuação.

Figura 22 - Função que calcula a pontuação final do jogador no grupo

```
public void setPontuacaoJogador(int pontuacao, Double tempo, int cliques, int grupo, int nivel) {
    double ajusteGrupo = 1;
    bloquearBotaoContinuar(grupo);//bloqueia o botão continuar que aparece no popup
    if(grupo==3 || grupo == 4){
        ajusteGrupo = 1.5;
    }
    Double pontosRanking = (((pontuacao * (tempo) + pontuacao) / cliques) * 100) * ajusteGrupo;
    pontuacaoFase = Math.round(pontosRanking);
    atualizarPontuacao();
    setQuantidadeEstrelas(Math.round(pontosRanking), nivel);
}
```

Fonte: autoria própria (2018).

A cada grupo finalizado é apresentado ao jogador um *pop-up* (Figura 23) de *feedback* mostrando a quantidade de pontos obtidos no grupo em que o jogador pode escolher continuar para o próximo grupo, reiniciar o grupo ou sair do jogo. Caso ele clique em reiniciar, a sua pontuação no respectivo grupo não será salva (isso impede que o jogador repita o mesmo grupo várias vezes a fim de melhorar sua posição no *ranking*).

Figura 23 - Pop-up exibido quando um grupo é finalizado

Fonte: autoria própria (2018).

Quando o jogador finaliza todos os grupos de um nível, um *pop-up* (Figura 24) é exibido, permitindo que ele escolha qual dos outros níveis disponíveis deseja jogar. Caso o jogador não consiga terminar o grupo em que está até que o tempo acabe, outro *pop-up*, contendo o *ranking* atualizado com as informações da partida jogada, será apresentado à ele. Ainda, caso o mude de nível antes de finalizá-lo, a pontuação do jogador em cada grupo daquele determinado nível será salva até que ele retorne. Afim de garantir que o jogador percorra todas os níveis e grupos, assim que finaliza um deles, o jogador é automaticamente levado ao próximo nível (caso tenha finalizado todos os grupos) ou ao próximo grupo (caso ainda exista um grupo não finalizado no nível em que ele se encontra).

Figura 24 - Pop-up exibido quando um nível é finalizado

Fonte: autoria própria (2018).

O principal aspecto do Jogo de Memória Auditiva com relação à memória de trabalho auditiva, é que o jogador precisa ouvir os fonemas (que serão armazenados em sua memória de trabalho auditiva) e procurar outro fonema correspondente ao que foi ouvido. O número de grupos e de níveis possibilita ao jogador um vasto conjunto de fonemas que, como foi explicitado acima, o jogo, automaticamente, apresentará ao jogador. Ou seja, a cada grupo ou nível que terminar, o jogador estará treinando sua memória de trabalho auditiva com fonemas que ainda não conhece e reforçando os que já conhece.

3.3.3 Implementação

Toda a codificação do jogo aqui desenvolvido foi feita utilizando-se a IDE *NetBeans,* de maneira tal que os arquivos eram salvos tanto na máquina que foi utilizada para a implementação como no repositório do Github.

Como visto anteriormente, a interface do jogo aqui desenvolvido precisaria ser o mais amigável e atrativa possível, por isso optou-se por utilizar alguns elementos de interface gráfica disponibilizados gratuitamente no site <https://craftpix.net>, e adaptá-los de acordo às nossas necessidades.

Dessa forma, o plano de fundo do jogo, a aparência dos *pop-ups* e as imagens presentes nos botões foram baseadas em um pacote de interface disponibilizada pelo site, mais especificamente o pacote *Kids Fantasy Game GUI.* Já as imagens dos avatares foram obtidas de uma pacote de imagens disponibilizados, também gratuitamente, no site <https://pixabay.com/pt/>.

Os elementos sonoros referentes aos fonemas foram gravados por uma integrante do Grupo de Pesquisa, Ensino e Aprendizado Típico e Atípico da Leitura e da Escrita[13] de onde a ideia inicial deste jogo surgiu. Já os sons de *feedback* foram adquiridos gratuitamente, no site https://freesound.org/browse/tags/sound-effects/.

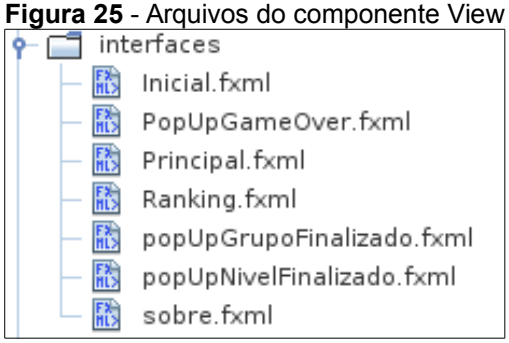
Seguindo o padrão MVC que foi utilizado para a modelagem do *software*, precisou-se criar três diretórios principais que conteriam os arquivos de cada um dos componentes do referido padrão.

O primeiro diretório criado foi referente ao componente *View*. A criação de cada arquivo se deu após a verificação de quais telas que o jogo iria possuir. Por questões de organização, preferiu-se salvar os arquivos referentes ao *View* em um diretório com o nome de “interface”, como pode ser visto na Figura 25.

Os primeiros *Views* criados referiam-se à estrutura da interface básica do jogo, mostrada na Figura 32 do Apêndice A. Embora essa estrutura tenha sido um protótipo que foi alterado ao longo do desenvolvimento da ferramenta, ela possibilitou a implementação da funcionalidade essencial do jogo (permitir que o jogador formasse pares do mesmo som ouvido) que era indispensável para a realização de futuros testes e para a adição das novas funcionalidades

13 O cadastro do grupo no CNPQ pode ser acessado no site http://dgp.cnpq.br/dgp/espelhogrupo/0978642031757397

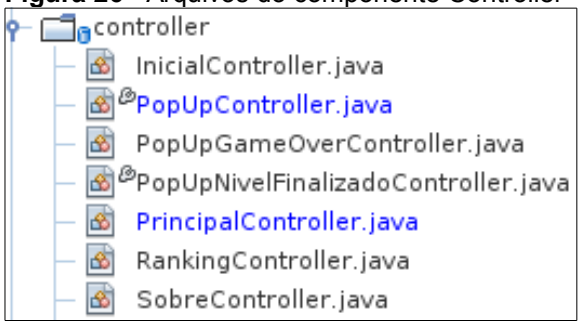
Figura 25 - Arquivos do componente View



Fonte: autoria própria (2018).

A cada arquivo do *View* que era criado, o *Netbeans* automaticamente criava um arquivo *Controller* para receber as interações com o usuário daquele *View*. Algumas interações com as interfaces, como o clique nos botões, foram adicionados diretamente durante a criação das telas do jogo, para posteriormente suas funções serem implementadas. Dessa forma, é possível ver na Figura 26 os arquivos do *Controller* referentes a esses *Views*.

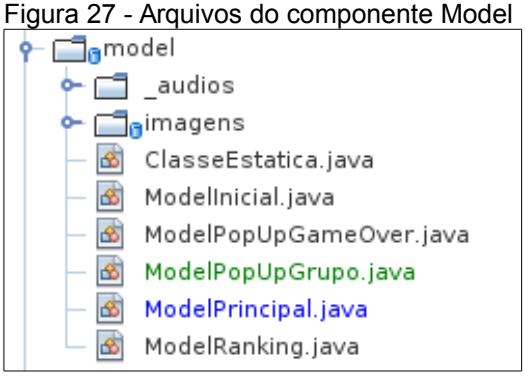
Figura 26 - Arquivos do componente Controller



Fonte: autoria própria (2018).

O terceiro diretório criado foi o do *Model* (Figura 27). Nele é possível verificar que nem todos os *controllers* estão associados a um *model*. Isso se deve ao fato de que um *Model* pode gerenciar as requisições de mais de um *Controller* e este pode estar associado a mais de um *View*. Durante a construção de alguns *Views*, como os da interface principal, percebeu-se que sua estrutura era parecida e que era possível então utilizar o mesmo *Controller* para eles.

Figura 27 - Arquivos do componente Model



Fonte: autoria própria (2018).

A partir da figura é possível ver, ainda, a existência dos diretórios "_audios" e "imagens". Eles foram alocados como um subdiretório do *Model* porque, ainda que o Java permita a utilização de uma pasta *resources* para armazenamento de todo tipo de recurso necessário na aplicação, a utilização desta pasta gerava erros quando se executava o arquivo .jar[14] do jogo.

Embora embora seja inviável a descrição de todas as funções que foram codificadas, consideramos importante comentar sobre as duas mais importantes. A primeira é a função *gerarOpcoes()* que pode ser vista no Apêndice C, que é responsável por gerar aleatoriamente os fonemas que serão ouvidos durante o jogo. A cada vez que o jogador inicia em um grupo/nível novos fonemas serão gerados. Isso impede que o jogador decore a posição dos fonemas caso mude de um nível/grupo para um nível/grupo que estava antes e ainda não tenha terminado.

A segunda função é a *verificarClique()* que verifica os pares escolhidos pelo jogador. Além da referida verificação, ela garante que, quando um jogador clicar em um botão, o fonema executado por ele seja reproduzido completamente para que o jogador não tenha dificuldade em entendê-lo. Essa função pode ser vista no Apêndice D.

3.3.4 Testes

Como adotamos a metodologia Scrum, a fase de testes do jogo acontecia periodicamente a cada nova entrega de funcionalidades do sistema, quando

14 Essa é a extensão utilizada quando se gera arquivos executáveis em Java

realizava-se três tipos de testes: testes de interface, testes de funções e testes de experiência do usuário.

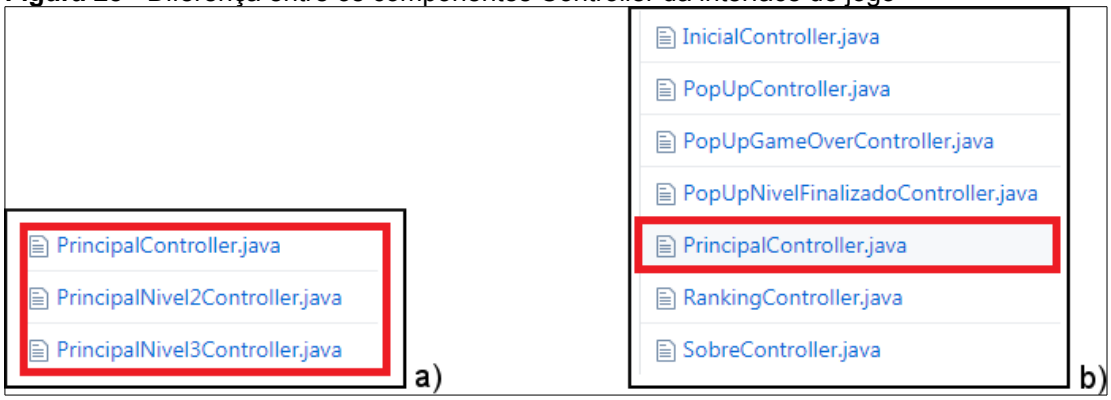
Os testes de interface consistiram de tentar-se encontrar erros nas transições e nos cliques dos botões das interfaces que foram construídas. A boa definição dos elementos e estrutura da interface garantiu que nenhum erro de execução fosse encontrado nos primeiros testes do Jogo de Memória Auditiva.

A realização desse teste nas fases iniciais do desenvolvimento foi fundamental para a verificação de uma melhoria na performance do jogo: cada vez que se clicava em um dos três níveis a interface era atualizada, entretanto, essa atualização acontecia de forma lenta. Isso ocorria porque cada uma das interfaces estava salva em um arquivo .fxml diferente e estava associada a um *Controller* e a um *Model* diferentes. Dessa forma, ao se clicar no botão do nível II, por exemplo, todo o arquivo .fxml desse nível deveria ser carregado na tela.

Como visto anteriormente, no padrão MVC diferentes *Views* podem ter o mesmo *Controller* e *Model*. E como o único elemento da interface que era alterado ao se clicar nos níveis eram os botões centrais do jogo, decidiu-se utilizar um único *Controller* para realizar a atualização da interface. Na Figura 28 é possível ver como ficou o *Controller* dos *Views* da interface principal antes e após a alteração.

Além de melhorar a performance do jogo, a modificação feita otimizou o tempo gasto nas futuras atualizações do Jogo de Memória Auditiva. Caso a versão antiga (Figura 28 (a)) fosse mantida, cada novo elemento adicionado à interface principal deveria ser adicionado nos outros dois arquivos .fxml que foram removidos e os *Controllers* de cada uma delas deveria ser atualizado. Isso deixaria a manutenção do código mais demorada e trabalhosa.

Figura 28 - Diferença entre os componentes Controller da interface do jogo



Fonte: autoria própria (2018).

Nos testes de função verificou-se as funcionalidades essenciais do jogo (a função que gerava as novas opções de áudio e a função que verifica se o par escolhido pelo jogador) estavam corretas. Durante essa fase foram encontrados erros referentes à geração dos novos áudios, pois como utilizou-se *arrays* para mapear os nomes do arquivos, a busca de alguns arquivos nesses *arrays* gerava a exceção de *IndexOutOfBoundException*[15], que foi corrigida com uma simples atualização na faixa de valores aleatórios que a função poderia acessar no *array*.

A terceira etapa de testes foi a mais longa entre as três. Depois de implementada a estrutura da interface e as funções básicas citadas acima, o próximo passo no processo de desenvolvimento consistia na entrega das funcionalidades referentes à melhoria na experiência do usuário, principalmente os *feedbacks*. Assim, a cada novo elemento de *feedback* adicionado, testava-se a sua integração com as funcionalidades já existentes, e novamente se testava a interface e as funções básicas, de modo que esse ciclo se repetiu até a finalização do Jogo de Memória Auditiva.

A maioria dos erros encontrados nessa etapa estava relacionado à usabilidade e a jogabilidade do jogo, pois apesar de não apresentar mais erros de execução, percebeu-se que algumas alterações poderiam ser feitas, e que elas poderiam melhorar a experiência do usuário.

Embora o jogo tenha sido testado nos sistemas operacionais GNU/Linux e Windows, poucas diferenças foram percebidas durante as execuções. A principal delas diz respeito à qualidade do som ouvido, que no Windows, era superior.

15 Na linguagem Java esse erro ocorre quando tenta-se acessar um índice que não existe em um determinado vetor ou *array*

4 RESULTADOS E CONCLUSÃO

Softwares educativos, mais especificamente jogos, têm-se mostrado uma ótima alternativa como ferramentas complementares na alfabetização de crianças. Embora exista uma vasta gama de jogos dos mais variados tipos, poucos deles são específicos no treinamento da memória de trabalho auditiva das crianças e, dos que existem, menos ainda estão disponíveis na Língua Portuguesa.

A utilização de metodologias de desenvolvimento como o Scrum para o planejamento dos requisitos a serem implementados e de padrões de organização de *software* como o MVC mostraram-se fundamentais para a realização do projeto aqui proposto, principalmente no que diz respeito às primeiras etapas a serem realizadas no processo de implementação e à organização dos arquivos criados.

Nesse sentido, a utilização da plataforma de versionamento Github foi imprescindível, pois a máquina em que toda codificação estava sendo feita apresentou um defeito no HD interno, impossibilitando a continuidade da implementação. Entretanto, como uma versão atualizada do código estava disponível na referida plataforma, foi possível continuar o desenvolvimento do jogo em outra máquina até que o problema na primeira fosse resolvido.

Além disso, a revisão da literatura a cerca do desenvolvimento de interfaces de *software* para crianças e o conhecimento acerca do desenvolvimento de *softwares* que foi adquirido durante a graduação foram imprescindíveis para o projeto e a implementação do Jogo de Memória Auditiva.

A principal dificuldade encontrada durante a fase de desenvolvimento foi a implementação dos *pop-ups* de *feedbacks* que são apresentados aos usuários. Embora o Java possua uma biblioteca que facilita a criação de *pop-ups*, ela não supria completamente nossas necessidades (reconhecer cliques nos botões que seriam adicionados ao *pop-up*). Numa tentativa de contornar o problema encontramos uma biblioteca externa desenvolvida para o Java, entretanto ela utilizava outras bibliotecas que já não eram mais suportadas. Assim, a solução encontrada foi carregar o arquivo .fxml dentro da janela do *pop-up* gerado pela biblioteca do Java, como pode ser visto na Figura 29, em que tem-se o trecho do código que gera o *pop-up* de grupo finalizado, exemplificado anteriormente, pela Figura 23.

Figura 29 - Carregamento da interface dentro do pop-up

```
setMostrandoPopUp(true);
setDesabilitado(true);
fxmlPopUp = new FXMLLoader(getClass().getResource("/interfaces/popUpGrupoFinalizado.fxml"));

try {
    ClasseEstatica.popup.getContent().add((Parent) fxmlPopUp.load());
} catch (IOException ex) {
    Logger.getLogger(ModelInicial.class.getName()).log(Level.SEVERE, null, ex);
}
PopUpController popUpController = fxmlPopUp.getController();
ClasseEstatica.popup.show(janela);
popUpController.setPontuacaoJogador(pontos, getTempo(), cliquesTotais, getFase(), getNivel());
ClasseEstatica.popup.addEventHandler(ActionEvent.ANY, new WeakEventHandler<>(evento -> {
    ClasseEstatica.popup.hide();
}));
```

Fonte: autoria própria (2018).

Como dito acima, a ideia inicial do jogo surgiu de um grupo de pesquisa e, durante o seu desenvolvimento, percebemos que um software de melhor qualidade poderia ter sido produzido caso dispuséssemos de alguns recursos. Por isso consideramos válido sugerir que grupos de pesquisa que desejem produzir algum tipo de *software* educativo tenham acesso a esses recursos. O primeiro refere-se à recursos humanos, mais especificamente designers gráficos e ilustradores, que podem produzir interfaces específicas para o tipo de *software* que se deseja desenvolver. O segundo refere-se à ferramentas necessárias para produção ou edição das mídias que serão utilizadas no *software*, como, por exemplo, gravadores de áudio/vídeo.

A realização dos testes mostrou que o Jogo de Memória Auditiva atende de maneira satisfatória aos objetivos que nos propusemos atingir, de maneira que produzimos um jogo esteticamente atrativo, com uma interface simples e divertida, que ocupa pouco espaço na memória do computador e de fácil instalação.

Assim, considerando-se os conceitos teóricos acerca do desenvolvimento de *software* para crianças e as pesquisas referentes à utilização de *softwares* educativos como ferramentas interventivas para a alfabetização, conclui-se que o jogo aqui criado pode ser usado como uma ferramenta complementar no ensino da leitura e da escrita nos anos iniciais da alfabetização em escolas brasileiras.

5 TRABALHOS FUTUROS

Visando a melhoria do trabalho aqui realizado, alguns novos estudos poderiam ser feitos. Na área da Ciência da Computação, o jogo poderia ser validado com crianças em que se verificaria a sua usabilidade e se o *design* desenvolvido está realmente adequado às crianças. Na área de aquisição da linguagem, o jogo poderia ser testado com crianças, em que se verificaria sua eficiência em ajudá-la a aprender a ler e a escrever.

Com relação ao desenvolvimento do jogo, algumas melhorias podem ser adicionadas a fim de torná-lo mais divertido e instigador para os jogadores. Uma delas é manter o *ranking* fixo na tela, de modo que que a criança possa verificar sua posição de acordo fosse ganhando mais pontos.

Outra atualização possível seria tornar a dificuldade do jogo adaptável ao nível do jogador, de modo que ele não perdesse o interesse em jogar, caso sinta dificuldades durante as partidas.

Uma outra, que está presente no *Product Backlog* do jogo, mas que não foi possível entregar, é a adição de um vídeo demonstrativo do jogo, ensinando ao jogador como ele funciona.

REFERÊNCIAS

ANDRADE, A. N. D. *et al*. Avaliação comportamental do processamento auditivo em indivíduos gagos. **Pró-Fono Revista de Atualização Científica**, 2008.

BADDELEY, A.; HITCH, G. Working memory. **Psychology of learning and motivation**, v. 8, p. 47-89, 1974.

BISSI, W. Metodologia de desenvolvimento ágil. **Campo Digital**, v. 2, n. 1, 2007.

CAPELLINI, S. A.; GERMANO, G. D.; CARDOSO, A. C. V. Relação entre habilidades auditivas e fonológicas em crianças com dislexia do desenvolvimento. **Psicologia Escolar e Educacional**, p. 235-251, 2008.

CANO, S. et al. Training with Phonak: Serious Game as support in Auditory--Verbal Therapy for Children with Cochlear Implants. In:**Proceedings of the 3rd 2015 Workshop on ICTs for improving Patients Rehabilitation Research Techniques**. ACM, 2015. p. 22-25.

CERVONE, H. F. Understanding agile project management methods using Scrum. **OCLC Systems & Services: International digital library perspectives**, v. 27, n. 1, p. 18-22, 2011.

CHAVES, E. O que é Software Educacional. **Rio de Janeiro: Janeiro**, 1987.

CHANDRASEKARAN, B.; KRAUS, N. Music, noise-exclusion, and learning. **Music Perception: An Interdisciplinary Journal**, v. 27, n. 4, p. 297-306, 2010.

COSTA, C. L. **Dificuldades de Leitura e memória de trabalho: um estudo correlacional**. (2010).

DEHAENE, S. Os neurônios da leitura. **Porto Alegre: Penso**, 2012.

DRUIN, A. et al. Designing a digital library for young children. In:**Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries**. ACM, 2001. p. 398-405.

ENGELMANN, L.; FERREIRA, M. I. D. C. Avaliação do processamento auditivo em crianças com dificuldades de aprendizagem. **Rev Soc Bras Fonoaudiol**, v. 14, n. 1,

p. 69-74, 2009.

EUGÊNIO, M. L.; ESCALDA, J.; LEMOS, S. M. A. Desenvolvimento cognitivo, auditivo e linguístico em crianças expostas à música: produção de conhecimento nacional e internacional. **Revista CEFAC**, São Paulo, v. 14, n. 5, p. 992-1003, set./out. 2012.

FALKEMBACH, G. A. M. **O lúdico e os jogos educacionais**. Centro Interdisciplinar de Novas Tecnologias na Educação, UFRGS. 2006. Disponível em: <http://penta3.ufrgs.br/midiasedu/modulo13/etapa1/leituras/arquivos/Leitura_1.pdf>. Acesso em: 22 Mai. 2018

FISCH, S. M. Making educational computer games educational. In:**Proceedings of the 2005 conference on Interaction design and children**. ACM, 2005. p. 56-61.

GIL, H; MENEZES, H. **Software educativo e a importância de uma «métrica»**. 6º Simpósio Internacional de Informática Educativa, 2004. Disponível em: <https://repositorio.ipcb.pt/bitstream/10400.11/922/1/Software%20educativo %20e...pdf>. Acesso em: 2 Mai. 2018.

GROS, B. The design of learning environments using videogames in formal education. In:**Digital Game and Intelligent Toy Enhanced Learning, 2007. DIGITEL'07. The First IEEE International Workshop on**. IEEE, 2007. p. 19-24.

GUARESI, R. **Alfabetização e letramento: é possível qualificar o ensino de língua materna no Brasil?** Curitiba, PR: CRV, 2017.

HONDA, A. et al. Transfer effects on sound localization performances from playing a virtual three-dimensional auditory game. **Applied Acoustics**, v. 68, n. 8, p. 885-896, 2007.

HOURCADE, J. P. et al. Interaction design and children. **Foundations and Trends® in Human–Computer Interaction**, v. 1, n. 4, p. 277-392, 2008.

HOURCADE, J.P.; BEDERSON, B. B.; DRUIN, A. **Preschool children's use of mouse buttons**, in Extended Abstracts of Human Factors in Computing Systems (CHI 2004), pp. 1411–1412, ACM Press, 2004.

JUCÁ, S. C. S. **A relevância dos softwares educativos na educação profissional**. Ciências & Cognição, v. 8, p. 22-28, 2006.

JUUL, J. The game, the player, the world: Looking for a heart of gameness.**PLURAIS-Revista Multidisciplinar**, v. 1, n. 2, 2010.

KEITH, C. **Agile game development with Scrum**. Pearson Education, 2010.

KRUG, F. S. A importância da leitura na formação do leitor. **REI–Revista de educação do IDEAU**, v. 10, n. 22, 2015.

LEMOS, G. O Processamento Auditivo Central nos Distúrbios Articulatórios. **Monografia (Especialização em Motricidade Oral). CEFAC, Fortaleza**, 1999.

LIMA, A.; RIBEIRO, V. M.; CATELLI JR, R. **Indicador de Alfabetismo Funcional – INAF**: Estudo especial sobre alfabetismo e mundo do trabalho. São Paulo, 2016. Disponível em: <http://acaoeducativa.org.br/wp-content/uploads/2016/09/INAFEstudosEspeciais_2016_Letramento_e_Mundo_do_Trabalho.pdf>. Acesso em: 19 jul. 2018.

LIMA, A. **Ambiente virtual para auxiliar a aprendizagem de meninas com dificuldade de leitura**. Mogi das Cruzes, SP. Tese de Doutorado em Engenharia Biomédica. Universidade de Mogi das Cruzes, 2010.

MACHADO, C. S. S. et al. Caracterização do processamento auditivo das crianças com distúrbio de leitura e escrita de 8 a 12 anos em tratamento no Centro Clínico de Fonoaudiologia da Pontifícia Universidade Católica de Minas Gerais. **Revista CEFAC**, v. 13, n. 3, p. 504-512, 2011.

MORAIS, R. X. T. **Software educacional**: a importância de sua avaliação e do seu uso nas salas de aula. Monografia (Bacharel em ciências da computação) - Faculdade Lourenço Filho. 51p. Fortaleza, 2003.

NETO, J. F. B. **Uma Metodologia de Desenvolvimento de Jogos Educativos em Dispositivos Móveis para Ambientes Virtuais de Aprendizagem**. 2012.

NETO, J. F. B.; FONSECA, F. S. Jogos educativos em dispositivos móveis como auxílio ao ensino da matemática. **RENOTE**, v. 11, n. 1, 2013.

NEVES, I. F.; SCHOCHAT, E. Maturação do processamento auditivo em crianças com e sem dificuldades escolares. **Pró-Fono Revista de Atualização Científica**, v.

17, n. 3, p. 311-20, 2005.

NUNES, C.; FROTA, S.; MOUSINHO, R. Consciência fonológica e o processo de aprendizagem de leitura e escrita: implicações teóricas para o embasamento da prática fonoaudiológica. **Cefac**. 2009;11(2):207-12.

OJANEN, E. et al. GraphoGame–a catalyst for multi-level promotion of literacy in diverse contexts. **Frontiers in psychology**, v. 6, p. 671, 2015.

PINKERING, S. J. **The development of visuo-spatial working memory**. *Memory*, 9, 423-32, 2001.

PIPER, F. K. A importância da memória de trabalho para a aprendizagem. ***Semana de letras***, v. 13, p. 1-6, 2013.

RAMOS, D. K. Jogos eletrônicos e aprendizagem: aspectos motivacionais na percepção de jovens jogadores. **Revista NUPEM**, v. 7, n. 12, p. 209-225, 2015.

SAVI, R.; ULBRICHT, V. R. Jogos digitais educacionais: benefícios e desafios. **RENOTE**, v. 6, n. 1, 2008.

SIMON, L. F; ROSSI, A.G. **Triagem do processamento auditivo em escolares de 8 a 10 anos**. Psicol. esc. Educ .2006;10(2):293-304.

SHARAN, K. **Learn JavaFX 8: Building User Experience and Interfaces with Java 8**. Apress, 2015.

SHEILA A.; MASSIGNANI, R.; SCHILLO, R. Aplicabilidade do software Fast Forword na reabilitação dos distúrbios do processamento auditivo: resultados iniciais. **Revista CEFAC**, v. 10, n. 4, 2008.

SOARES, M. Alfabetização e letramento: caminhos e descaminhos. **Revista Pátio**, v. 29, p. 19-22, 2004.

STEINER, LUCIANE. Processamento auditivo central. **Trabalho de Conclusão de Curso (Especialização em Audiologia Clínica)–CEFAC. Porto Alegre**, 1999.

SUCENA, A. et al. **Graphogame Português Alicerce**: Software de apoio a crianças

disléxicas. 2015.

TALLAL, Paula; PIERCY, Malcolm. Developmental aphasia: Rate of auditory processing and selective impairment of consonant perception. **Neuropsychologia**, v. 12, n. 1, p. 83-93, 1974.

TAROUCO, L. M. R. et al. Jogos educacionais. **RENOTE**. Porto Alegre, RS, 2004.

TEIXEIRA, A. C; BRANDÃO, E. J. R. Software educacional: o difícil começo. **RENOTE**, v. 1, n. 1, 2003.

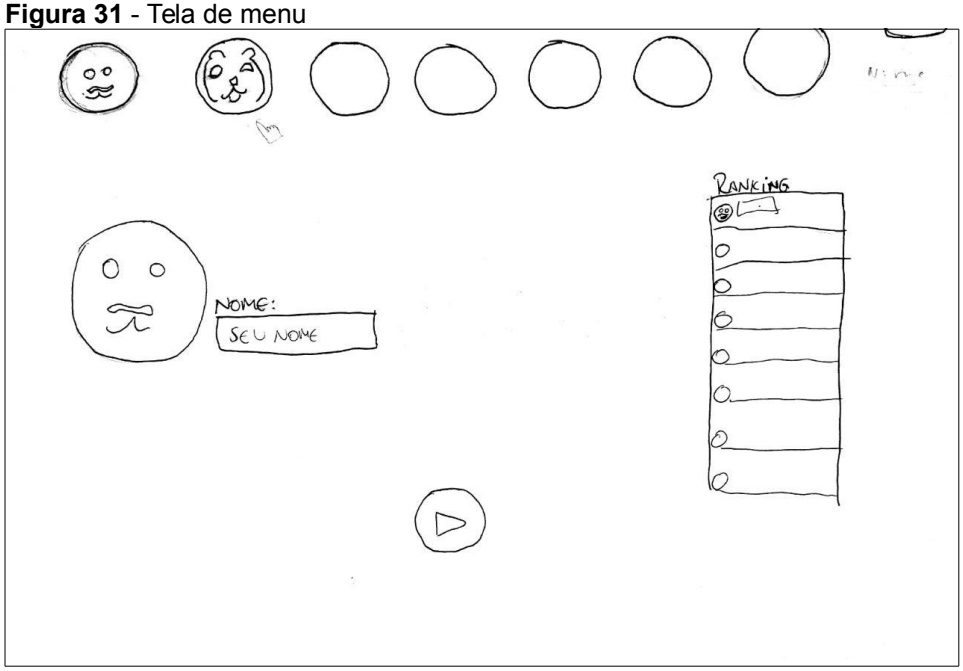
UEHARA, E.; LANDEIRA-FERNANDEZ, J. Um panorama sobre o desenvolvimento da memória de trabalho e seus prejuízos no aprendizado escolar. **Ciências & Cognição**. *cogn*., Rio de Janeiro , v. 15, n. 2, p. 31-41, ago. 2010.

APÊNDICE A – Design inicial do jogo

Figura 30 - Tela inicial

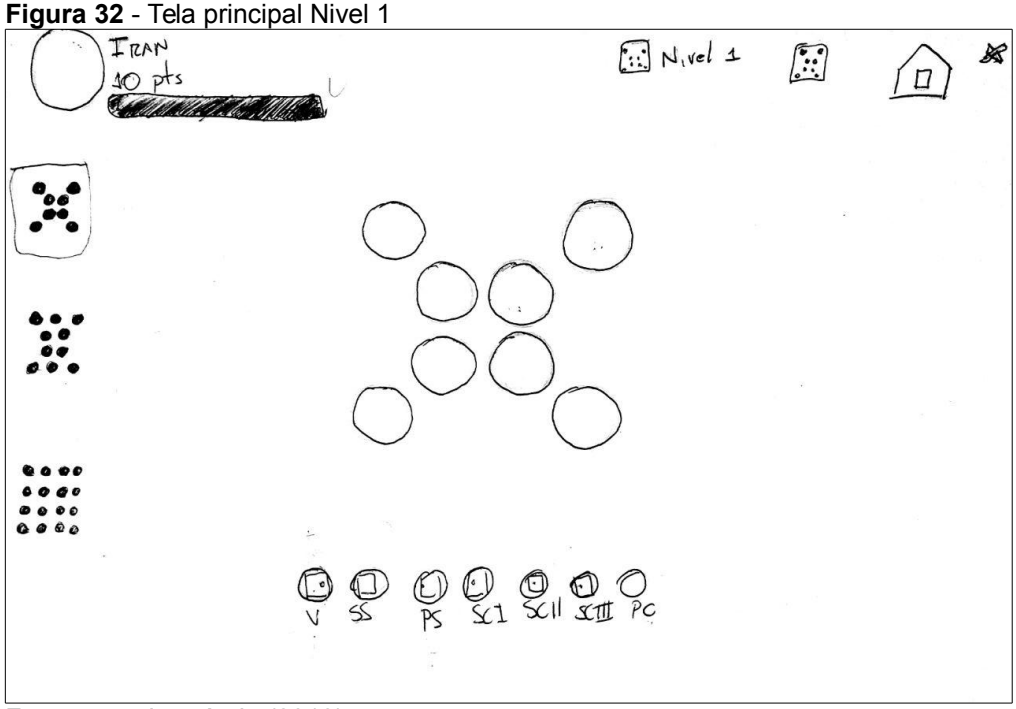
Fonte: autoria própria (2018).

Figura 31 - Tela de menu



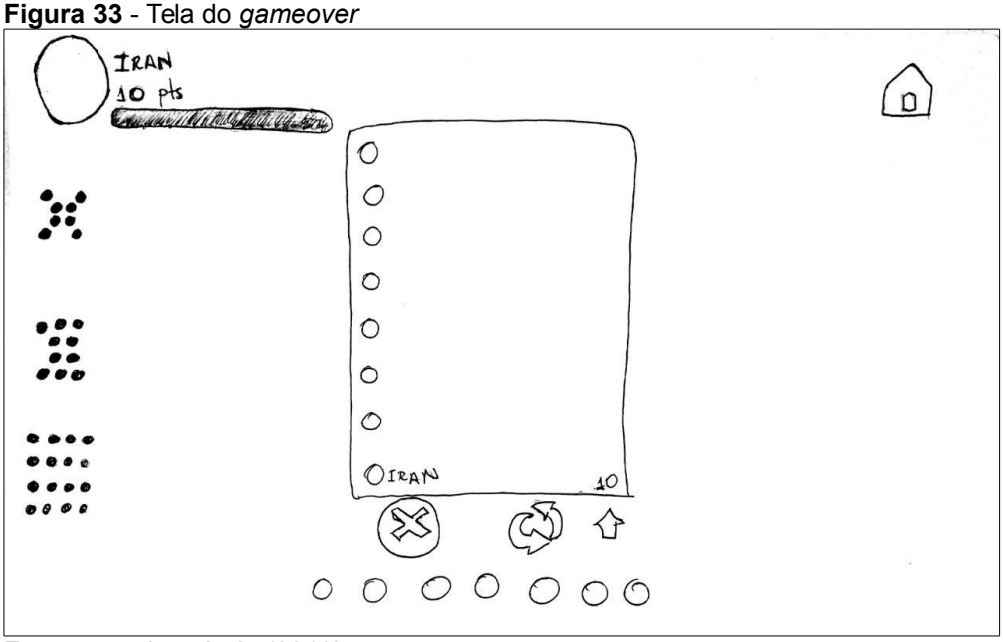
Fonte: autoria própria (2018).

Figura 32 - Tela principal Nivel 1



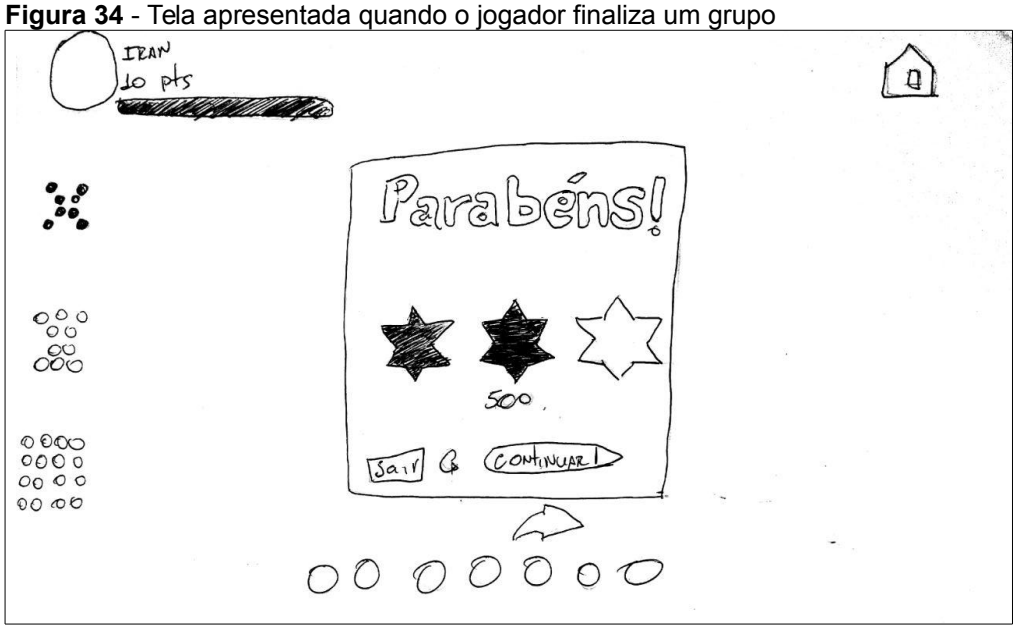
Fonte: autoria própria (2018).

Figura 33 - Tela do *gameover*



Fonte: autoria própria (2018).

Figura 34 - Tela apresentada quando o jogador finaliza um grupo



Fonte: autoria própria (2018).

APÊNDICE B – Níveis 2 e 3 do jogo

Figura 35 - Nível 2 do jogo

Fonte: autoria própria (2018).

Figura 36 - Nível 3 do jogo

Fonte: autoria própria (2018).

APÊNDICE C – Função que escolhe os fonemas aleatoriamente

```
public void gerarOpcoes(int grupo) {
      int i = 0;
      int proxValor = 0;
      int numeroFonemas = 0;
      int contador = 0;
      int numeroFonemasVetores = 0;
      novasOpcoes = new ArrayList();
      ArrayList indicesUtilizados = new ArrayList();
      ArrayList indicesFonemasUtilizados = new ArrayList();
      Random indice = new Random();
      switch (getNivel()) {
         case 1: numeroFonemas = 4; break;
         case 2: numeroFonemas = 5; break;
         case 3: numeroFonemas = 8; break; }
      switch (getGrupo()) {
         case 1: numeroFonemasVetores = 5; break;
         case 2: numeroFonemasVetores = 94; break;
         case 3: numeroFonemasVetores = 99; break;
         case 4: numeroFonemasVetores = 244; break;
         case 5: numeroFonemasVetores = 57; break;
         case 6: numeroFonemasVetores = 30; break;
         case 7: numeroFonemasVetores = 101; break;
}
      while (i < numeroFonemas) {
         proxValor = indice.nextInt(numeroFonemasVetores);
         if (!indicesUtilizados.contains(proxValor)) {
            novasOpcoes.add(proxValor);
            indicesUtilizados.add(proxValor);
            i++;
         } else {
            if (getNivel() == 3 && getGrupo() == 1) {
               novasOpcoes.add(proxValor);
               i++;
            }
         }
      }
      while (contador < numeroFonemas * 2) {
         for (int j = 0; j < novasOpcoes.size(); j++) {
            int posicao1 = indice.nextInt(numeroFonemas * 2);
            int posicao2 = indice.nextInt(numeroFonemas * 2);
            while ((posicao1 == posicao2)
                  || ((indicesFonemasUtilizados.contains(posicao1)
                  || indicesFonemasUtilizados.contains(posicao2)))) {
               posicao1 = indice.nextInt(numeroFonemas * 2);
               posicao2 = indice.nextInt(numeroFonemas * 2);
            }
            indicesFonemasUtilizados.add(posicao1);
            indicesFonemasUtilizados.add(posicao2);
            ArrayNivel1[posicao1] = getFonemaArray(grupo, j);
            ArrayNivel1[posicao2] = getFonemaArray(grupo, j);
            contador = contador + 2;
         }
```

```
    }
}
```

APÊNDICE D – Função que verifica os pares escolhidos

```
public void verificarOpcao(ActionEvent evento) {
      if (primeiroClique() && !getTimerIniciado()) {
         iniciarTimer();
      }
      String nomeBotao = ((Button) evento.getSource()).getId();
      cliques++;
      if (cliques == 1) {
         tocarAudioBotao(evento);
         btemp1 = ((Button) evento.getSource());//grava qual botao foi clicado
         botao1 = ArrayNivel1[Integer.parseInt(nomeBotao.substring(1)) - 1];
         setCorBotao(btemp1, "#ffff00");//botao fica amarelo

      } else if (cliques == 2) { //verifica se é o segundo clique
         btemp2 = ((Button) evento.getSource());
         botao2 = ArrayNivel1[Integer.parseInt(nomeBotao.substring(1)) - 1];
         if (botao1.equals(botao2) && (!btemp1.equals(btemp2))) {
            incrementarAcerto();
            evento1Botao = (ActionEvent event) -> {
               setCorBotao(btemp2, "#00EE00");
               setCorBotao(btemp1, "#00EE00");
               tocarAudioBotao(evento);
            };
            evento2Botao = (ActionEvent event) -> {
               btemp1.setDisable(true);
               btemp2.setDisable(true);
               setCorBotao(btemp1, "#00000");
               setCorBotao(btemp2, "#00000");
               listaBotoes.add(btemp1);
               listaBotoes.add(btemp2);
            };
            eventoSomAcerto = (ActionEvent event) -> {
               tocarEfeitoAcerto();
               try {
                  verificarTerminoNivel();
               } catch (IOException ex) {

Logger.getLogger(ModelPrincipal.class.getName()).log(Level.SEVERE, null, ex);
               }
            };
            new Timeline(new KeyFrame(Duration.seconds(0), evento1Botao),
                  new KeyFrame(Duration.seconds(0.4), evento2Botao),
                  new KeyFrame(Duration.seconds(0.2), eventoSomAcerto)).play();
         } else {//os dois botões ficam vermelhos
            evento1Botao = (ActionEvent event) -> {
               setCorBotao(btemp2, "#ff0000");
               setCorBotao(btemp1, "#ff0000");
               tocarAudioBotao(evento);
            };
            evento2Botao = (ActionEvent event) -> {
               setCorBotao(btemp1, "#00000");
               setCorBotao(btemp2, "#00000");
            };
```

```
            new Timeline(
                    new KeyFrame(Duration.seconds(0), evento1Botao),
                    new KeyFrame(Duration.seconds(0.5), evento2Botao)).play();
            incrementarErro();
        }
        this.cliquesTotais = cliquesTotais + cliques;
        cliques = 0;
    }
}
```

APÊNDICE E – Trecho da função que exibe o *pop-up*

```
setMostrandoPopUp(true);
setDesabilitado(true);
fxmlPopUp = new
FXMLLoader(getClass().getResource("/interfaces/popUpGrupoFinalizado.fxml"));
try {
   ClasseEstatica.popup.getContent().add((Parent) fxmlPopUp.load());
} catch (IOException ex) {
   Logger.getLogger(ModelInicial.class.getName()).log(Level.SEVERE, null, ex);
}
PopUpController popUpController = fxmlPopUp.getController();
ClasseEstatica.popup.show(janela);
popUpController.setPontuacaoJogador(pontos, getTempo(), cliquesTotais,
getFase(), getNivel());
ClasseEstatica.popup.addEventHandler(ActionEvent.ANY, new
WeakEventHandler<>(evento -> {
   ClasseEstatica.popup.hide();
}));
```