

**UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE EDUCAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM TECNOLOGIAS EDUCACIONAIS
EM REDE**

SUPER MARIO LOGIC: UM JOGO SÉRIO PARA LÓGICA DE PROGRAMAÇÃO

DISSERTAÇÃO DE MESTRADO

Felipe Schmitt Panegalli

Santa Maria, RS, Brasil

2016

Felipe Schmitt Panegalli

SUPER MARIO LOGIC: UM JOGO SÉRIO PARA LÓGICA DE PROGRAMAÇÃO

Dissertação apresentada ao Programa de Pós-Graduação em Tecnologias Educacionais em Rede – Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do título de Mestre em Tecnologias Educacionais em Rede.

Orientadora: Prof^a Dr^a Giliane Bernardi

Santa Maria - RS

2016

Felipe Schmitt Panegalli

SUPER MARIO LOGIC: UM JOGO SÉRIO PARA LÓGICA DE PROGRAMAÇÃO

Dissertação apresentado ao Programa de Pós-Graduação em Tecnologias Educacionais em Rede – Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para obtenção do título de Mestre em Tecnologias Educacionais em Rede.

Aprovado em 26 de setembro de 2016:

Giliane Bernardi, Dra. (UFSM)
(Presidente/Orientadora)

Leila Maria Araújo Santos, Dra. (UFSM)

Gilse Antoninha Morgental Falkembach, Dra (QI)

Santa Maria, 26 de Setembro de 2016.

AGRADECIMENTOS

À minha querida esposa, Diane, pela paciência, companheirismo, pelo carinho e pela força dada durante esses dois anos de caminhada no mestrado.

A minha orientadora Dr^a Giliane Bernardi, pela confiança, pela oportunidade de trabalhar ao seu lado e por incentivar a superação de meus limites.

Ao Dr Mairon Melo Machado, que me deu apoio e me ajudou a começar essa caminhada.

Ao meu grande amigo Nedisson Luis Gessi, que me colocou no meio profissional, e que, hoje onde estou se dá ao seu ato.

Ao meu amigo e colega, Alex Mazzuco, que me deu dicas para a elaboração desse trabalho e que me ajudou muitas vezes quando mais precisava, sendo na vida pessoal ou profissional.

Aos familiares e amigos que sempre me incentivaram a estudar e correr atrás de meus sonhos.

A todos os professores do mestrado que de alguma forma contribuíram para minha formação.

A banca que destinaram parte de seu precioso tempo para participarem desta pesquisa.

RESUMO

SUPER MARIO LOGIC: UM JOGO SÉRIO PARA LÓGICA DE PROGRAMAÇÃO

AUTOR: FELIPE SCHMITT PANEGALLI

ORIENTADORA: GILIANE BERNARDI

Esta dissertação faz parte do Programa de Pós-Graduação *Stricto Sensu* em Tecnologias Educacionais em Rede, linha de Desenvolvimento de Tecnologias Educacionais em Rede e tem como produto final um Jogo Sério chamado Super Mário Logic, para Lógica de Programação, disciplina esta essencial ao futuro profissional da área. Por meio dela, o estudante aprende a desenvolver algoritmos, que são sequências de passos para chegar a um determinado resultado. A partir do desenvolvimento de algoritmos tem-se a etapa da programação, na qual um significativo estudo da Lógica de Programação e de Algoritmos desempenham papel fundamental para o programador. Este trabalho foi elaborado com o objetivo de desenvolver um Jogo Sério para desafiar o aluno a resolver problemas de lógica de programação. Os Jogos Sérios estão se popularizando, pois ajudam na aprendizagem do aluno, envolvendo-o, fazendo-o interagir e gerar resultados de uma forma educativa e lúdica, saindo do padrão de ensino tradicional. Os Jogos Sérios têm por finalidade melhorar aspectos específicos da aprendizagem, com o intuito de reforçar um aprendizado específico. O jogo foi desenvolvido por meio da metodologia INTERAD, utilizada para desenvolvimento de Interfaces Educacionais.

Palavras-chave: Algoritmos, Lógica de programação, Jogos Sérios.

ABSTRACT

SUPER MARIO LOGIC: UM JOGO SÉRIO PARA LÓGICA DE PROGRAMAÇÃO

AUTOR: FELIPE SCHMITT PANEGALLI

ORIENTADORA: GILIANE BERNARDI

This project is part of the Graduate Program in Educational Technology, Development of Educational Technologies and it aims at developing, as a final product, a Serious Game named Super Mario Logic in order to help in the teaching and learning process of logic programming which is essential to the professional future of the area. It is through logic programming that students learn to develop algorithms, which are sequences of steps used to reach a specific result. From the development of algorithms there are the programming stages, in which a significant study of logic programming and algorithms plays a key role for the programmer. This project aims at developing a serious game that challenges students into solving problems in logic programming. Serious games are becoming popular as they help students learn, making them interact and learn in an educational and playful way beyond the traditional teaching pattern. The game is being developed through a methodology called INTERAD, which is used for the development of Educational Interfaces.

Keywords: Algorithms, Logic Programming, Serious Games.

LISTA DE FIGURAS

Figura 1 - Exemplo de fluxograma com estruturas sequenciais.....	15
Figura 2 - Exemplo de estrutura sequencial em forma de algoritmo.	16
Figura 3 - Estrutura de Decisão SE-ENTÃO.	18
Figura 4 - Estrutura de Decisão SE-ENTÃO-SENÃO.....	19
Figura 5 - Estrutura de Decisão CASO.	20
Figura 6 – Estrutura de repetição ENQUANTO – FAÇA.	22
Figura 7 - Estrutura de repetição REPITA.....	22
Figura 8 - Estrutura de repetição PARA-ATÉ-FAÇA.	23
Figura 9 - Estrutura de repetição PARA-ATÉ-FAÇA de forma interativa.....	24
Figura 10 - Jogo Castelo dos Enigmas.	36
Figura 11 - Jogo Light-Bot 2.0 desenvolvido pela Armor Games.	37
Figura 12 - <i>Computational Thinking Framework (CTF)</i>	38
Figura 13 - Tela do jogo <i>Program Your Robot</i>	40
Figura 14 - Diagrama da metodologia INTERAD.	44
Figura 15 – Desenho de conteúdo do Super Mario Logic.	51
Figura 16 – <i>Storyboard</i> Menu.....	52
Figura 17 - <i>Storyboard</i> Mapa.....	53
Figura 18 - <i>Storyboard</i> da escolha do personagem.	53
Figura 19 - <i>Storyboard</i> Fase.....	55
Figura 20 – Malha construtiva da fase.	61
Figura 21 – Malha estrutural da fase.....	62
Figura 22 – Logomarca (identidade visual da marca).	64
Figura 23 – Tela inicial do jogo.....	65
Figura 24 – Tela de criação do personagem.....	65
Figura 25 – Mapa inicial do jogo.....	66
Figura 26 – Tela do desafio.....	67
Figura 27 – Tela de parabéns.	67
Figura 28 – Tela de tente novamente.....	68
Figura 29 – Última fase do jogo Super Mário Logic com grau de dificuldade difícil...69	69

LISTA DE TABELAS

Tabela 1 - Operadores aritméticos básicos.....	17
Tabela 2 - Operadores relacionais	18
Tabela 3 - Classificação de jogos quanto ao gênero.....	30
Tabela 4 - Lista de requisitos.	57
Tabela 5 - Elementos do jogo Super Mario Logic.....	58

LISTA DE GRÁFICOS

Gráfico 1 - Levantamento de dados – Grupo Concentração.	71
Gráfico 2 - Levantamento de dados - Grupo Desafios.	72
Gráfico 3 - Levantamento de dados - Grupo Habilidade do Jogador.	73
Gráfico 4 - Levantamento de dados - Grupo Controle.....	74
Gráfico 5 - Levantamento de dados - Grupo Objetivos.	75
Gráfico 6 - Levantamento de dados - Grupo <i>Feedback</i>	76
Gráfico 7 - Levantamento de dados - Grupo Imersão.	77
Gráfico 8 - Levantamento de dados - Grupo Imersão vista de forma geral.....	78

LISTA DE ABREVIATURAS E SIGLAS

3 ^a	Abstração, Automação, Análise
C#	C Sharp (Linguagem de programação)
IA	Inteligência Artificial
MED	Material Educacional Didático
OA	Objeto de Aprendizagem
PC	Pensamento Computacional
RPG	<i>Role-Playing Game</i>
SO	Sistema Operacional
TIC	Tecnologias da Informação e Comunicação
INTERAD	Interface Interativas Digitais aplicadas à Educação
CFT	<i>Computational Thinking Framework</i>
RH	Recursos Humanos

SUMÁRIO

1.	INTRODUÇÃO	12
2.	O ENSINO DE LÓGICA DE PROGRAMAÇÃO	14
2.1.	ESTRUTURAS DE PROGRAMAÇÃO	14
2.1.1.	Estrutura Sequencial	14
2.1.2.	Estrutura De Decisão	17
2.1.3.	Estrutura De Repetição	21
2.2.	ENSINO DE LÓGICA POR MEIO DO PENSAMENTO COMPUTACIONAL	24
2.3.	CONSIDERAÇÕES DO CAPÍTULO	26
3.	JOGOS EDUCACIONAIS	28
3.1.	CONCEITOS GERAIS	28
3.2.	JOGOS SÉRIOS	33
3.3.	CONSIDERAÇÕES DO CAPÍTULO	34
4.	TRABALHOS CORRELATOS	36
4.1.	CONSIDERAÇÕES DO CAPÍTULO	40
5.	ASPECTOS METODOLÓGICOS	42
6.	MODELAGEM E DESENVOLVIMENTO DO JOGO SÉRIO <i>SUPER MARIO</i> <i>LOGIC</i>	47
6.1.	SUPER MARIO, UM <i>BLOCKBUSTER</i>	47
6.2.	COMPREENSÃO	48
6.2.1.	Tema	48
6.2.2.	Público-Alvo	48
6.2.3.	Objetivos Pedagógicos	48
6.2.4.	Contexto Educacional	49
6.2.5.	Necessidades do Aluno	49
6.2.6.	Subsídios Projetuais	49
6.3.	PREPARAÇÃO	50
6.3.1.	Desenho de Conteúdo	50
6.3.2.	Funcionalidades	55
6.3.3.	Lista de Requisitos	56
6.4.	EXPERIMENTAÇÃO	57
6.4.1.	Modelo Conceitual	58
6.4.2.	Arquitetura	60
6.5.	ELABORAÇÃO	60

6.5.1. Tipo de Interatividade	61
6.5.2. Malha Construtiva	61
6.5.3. Malha Estrutural	62
6.5.4. Desenho De Navegação	63
6.6. APRESENTAÇÃO	63
6.6.1. Identidade de Marca	63
6.6.2. Design Visual	64
6.7. APRESENTAÇÃO DO JOGO	65
7. AVALIAÇÃO E DISCUSSÃO DOS RESULTADOS	70
8. CONSIDERAÇÕES FINAIS	80
REFERÊNCIAS	83
APÊNDICE A - MODELO DE AVALIAÇÃO GAMEFLOW	86

1. INTRODUÇÃO

A tecnologia é uma realidade que faz parte da sala de aula. Isso significa que o processo de aprendizagem passa a ser influenciado pela integração dessas tecnologias e pode contribuir para um aprendizado mais dinâmico e contextualizado com a era da informação. Conforme Castro (2007), o saber se tornou intensamente trivial. O que antes chegava ao aluno pela fala do professor, agora chega de formas diversificadas e com intensas dimensões.

Portanto, a tecnologia está influenciando cada vez mais na aula, podendo ajudar no processo de aprendizagem. No entanto, é preciso que seja utilizada de forma correta, para que possam influenciar positivamente nas escolas.

Amaral (2010) afirma que “as tecnologias de informação e comunicação podem, em estratégias bem desenhadas, ajudar a escola a cumprir seu papel relevante na atualidade”, isto é, “fazer os indivíduos serem capazes de aprender de forma permanente e de aprender a viver em comunidade”.

A justificativa para elaboração deste estudo está na grande dificuldade dos alunos relacionada a compreensão de aspectos de lógica de programação, pois, para chegar na etapa de desenvolvimento de algoritmos, o bom entendimento e conhecimento da lógica é fundamental, conforme destacam Falkembach e Araújo (2013, p. 01),

A dificuldade está no reconhecimento dos procedimentos necessários para se chegar à solução do problema. Isso implica em ter que trabalhar de forma mais eficiente os processos cognitivos, em especial a abstração e a formalização, necessários à construção de um algoritmo, ou seja, à modelagem da solução do problema por meio da técnica de algoritmos pseudocódigos.

Dessa forma, existindo complexidade em compreender a lógica em si, o aluno pode perder o interesse pela programação ou até mesmo desistir de continuar o curso em que se encontra. Em vista disso, o uso das tecnologias pode facilitar a compreensão e entendimento da lógica de programação, por meio de desafios e práticas dentro de ambientes virtuais, como o que será abordado no decorrer desta dissertação.

Acredita-se que a aprendizagem de lógica de programação pode ser trabalhada de modo interativo e dinâmico, utilizando as tecnologias como mediadoras do processo. Para tal, foi desenvolvido um Jogo Sérió denominado Super Mario Logic que permitiu estabelecer uma relação prática com problemas relacionados à lógica de programação por meio de desafios envolvendo as estruturas básicas usualmente discutidas em disciplinas de lógica e programação.

Neste contexto, o objetivo principal desta dissertação é desenvolver um Jogo Sérió para auxiliar na Lógica de Programação, buscando fornecer uma alternativa dinâmica, interativa e motivacional.

Para que o objetivo geral fosse alcançado, alguns objetivos específicos foram necessários como: utilizar uma metodologia de desenvolvimento de *software* educacional, para formalizar as etapas do jogo desenvolvido; definir os desafios de lógica de programação para o jogo, com níveis de complexidade crescentes; desenvolver o Jogo Sérió para aprendizagem de lógica de programação; disponibilizar o jogo para a plataforma Windows e Linux; aplicar e avaliar os resultados obtidos na utilização do jogo em relação à lógica de programação, bem como em relação à sua usabilidade e jogabilidade.

O texto está estruturado nos seguintes capítulos: 2 – O Ensino de Lógica de Programação: apresenta uma base teórica sobre o assunto; 3 – Jogos Educacionais: nessa seção são apresentados os conceitos gerais dos jogos bem como os elementos essenciais para desenvolvimento com qualidade, a classificação conforme o gênero e, por fim, uma descrição de Jogos Sérios; 4 – Trabalhos Correlatos: aqui são apresentados trabalhos realizados por outros pesquisadores para mostrar o que foi desenvolvido, tecnologias utilizadas e aplicabilidades; 5- Aspectos Metodológicos: a abordagem, para o desenvolvimento do jogo, será discutida neste capítulo; 6 – Modelagem e Desenvolvimento: este capítulo traz a modelagem e o desenvolvimento do jogo passo a passo através da metodologia escolhida na seção anterior; 7 – Avaliação e Discussão dos Resultados: neste capítulo é apresentada a avaliação dos resultados obtidos com o uso do jogo desenvolvido; e, por fim, o capítulo 8 – Considerações Finais, apresenta o fechamento do trabalho desenvolvido.

2. O ENSINO DE LÓGICA DE PROGRAMAÇÃO

Inicialmente, é necessário entender o que é lógica para compreender melhor o processo do desenvolvimento de um algoritmo. Conforme Forbellone e Eberspächer (2005), a lógica é o estado de deixar em ordem nosso pensamento, raciocinar da maneira correta, sem *pular* passos ou misturá-los, sendo assim, está normalmente relacionada à coerência e à racionalidade, isto é, correção do pensamento.

A correção do pensamento pode ser trabalhada com o uso de um algoritmo, que consiste em uma sequência de passos coerentes e válidos (FORBELLONE e EBERSPÄCHER, 2005). Para Furlan et al. (2012), algoritmo representa um conjunto de regras para a solução de um problema. Um exemplo simples é a receita de um bolo, onde, para chegar ao produto final (o bolo), deve-se seguir o passo a passo da receita. Portanto, o estudo de lógica de programação e algoritmos é necessário para o processo de desenvolvimento de *softwares*.

Na seção 2.1 são explicados os conceitos de lógica que serão aplicados no jogo que será desenvolvido e na seção 2.2 será abordado o ensino de lógica pela metodologia do Pensamento Computacional.

2.1. ESTRUTURAS DE PROGRAMAÇÃO

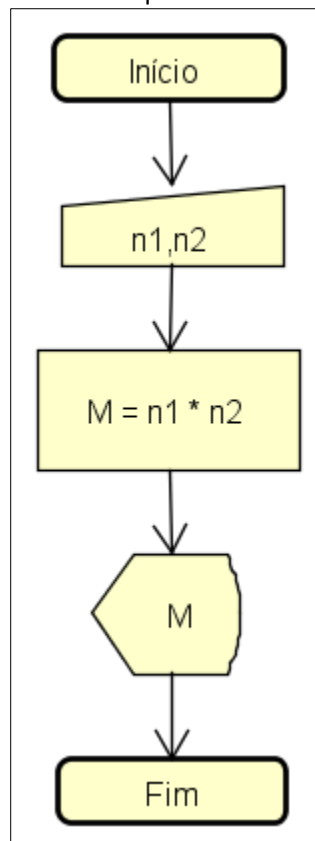
Conforme Furlan *et al.* (2012), as estruturas de programação permitem a descrição de qualquer algoritmo que seja computável, isto é, que pode ser executado em um computador. Dentre estas estruturas encontram-se: estrutura Sequencial, estrutura de Decisão e estrutura de Repetição, as quais serão abordadas nas próximas seções.

2.1.1. Estrutura sequencial

Para entender a lógica em geral, a estrutura sequencial é o ponto de partida para trabalhar as condições e os laços de repetição que serão abordados nesta seção. A lógica sequencial é trabalhada por meio de uma sequência lógica, isto é, um passo a passo para chegar a um objetivo, respeitando a sequência. Como exemplo, tem-se uma sequência de números de 1 até 10 [1,2,3...8,9,10], em que percebe-se que a sequência está correta pois começa no número 1 e vai até o número 10. No caso de [1,5,3,7,2,4,6,8,9,10], não há essa sequência.

Conforme Furlan *et al.* (2012), estruturas sequenciais de programação representam os comandos que são executados imperativamente, sem desvio de caminho, isto é, seguem uma sequência de início – meio – fim, sem que haja um desvio durante o processamento para se chegar a um objetivo. Na Figura 1 é apresentado um fluxograma que é uma representação gráfica da solução do problema, sendo este, exemplificado com um cálculo de multiplicação simples entre dois números. O **Início** representa o começo do fluxograma, **n1** e **n2** são as entradas (variáveis) dadas pelo primeiro e o segundo número. Na terceira etapa, tem-se **M** como alocador do resultado da multiplicação de **n1*n2**. Após, é mostrado o resultado **M** e **Fim** termina o fluxograma.

Figura 1 - Exemplo de fluxograma com estruturas sequenciais.



Fonte: Autor.

Da mesma maneira, porém utilizando um algoritmo escrito em português estruturado, o cálculo é demonstrado na Figura 2. Observa-se que o algoritmo é executado em uma sequência linear, começando da parte superior para a inferior e da esquerda para direita, da mesma maneira como o algoritmo foi escrito.

Figura 2 - Exemplo de estrutura sequencial em forma de algoritmo.

```

algoritmo "alg_multiplicacao"
// Função : Multiplicação de 2 números
var M, N1, N2: real

inicio
// Seção de Comandos
Escreva("Digite o primeiro número: ")
Leia(N1)
Escreva("Digite o segundo número: ")
Leia(N2)
M := N1 * N2
Escreva("O resultado é: ", M)
finalgoritmo

```

Fonte: Autor.

Observa-se que, na primeira linha, o cabeçalho **algoritmo** está identificando o algoritmo escrito; **var** trata da declaração das variáveis, **M** (resultado da multiplicação), **N1** e **N2** (entradas numéricas do tipo real); **Início** inicia a seção de comandos do algoritmo; **Escreva** é uma função utilizada para imprimir informações na tela, neste caso em específico a função está solicitando informar (digitar) e, ao final, exibindo o resultado da multiplicação; **Leia** é a função que recebe dados digitados pelo usuário. Neste caso, a função está aguardando os números (**N1** e **N2**) para a realização da multiplicação em **M := N1*N2** e, por fim, **finalgoritmo** que é utilizado para dizer ao computador que o algoritmo chegou ao final.

Ao longo do algoritmo percebe-se que aparece um asterisco (*) entre as variáveis (**N1** e **N2**), trata-se de um operador aritmético de multiplicação. Esses operadores aritméticos são utilizados para realização de uma conta matemática que conforme Manzano e Oliveira (2012) para elaborar uma expressão aritméticas, deve-se usar variáveis do tipo real ou inteira. Na Tabela 1 são apresentados os operadores aritméticos.

Tabela 1 - Operadores aritméticos básicos.

Operador	Função
+	Adição
-	Subtração
*	Multiplicação
/	Divisão
**	Potenciação
Mod	Resto (Módulo)

Fonte: (MANZANO; OLIVEIRA, 2012, p.26).

É muito comum a utilização de expressões aritméticas dentro de um código, pois na maior parte das vezes, precisa-se realizar validações ou até mesmo, chegar a resultados utilizando cálculos matemáticos.

2.1.2. Estrutura de decisão

A estrutura de decisão, diferente da estrutura sequencial, precisa de uma condição para a escolha de uma ação. Conforme Forbellone e Eberspächer (2005), uma estrutura de decisão ou seleção permite a escolha de um grupo de ações a ser executado e quando tais condições, representadas por expressões lógicas ou relacionais, são ou não satisfeitas. Para um melhor entendimento, Manzano e Oliveira (2012) simplificam que condição é uma expressão booleana, cujo resultado é um valor verdadeiro lógico falso ou verdadeiro. Na tabela 02, são mostrados os Operadores Relacionais que são utilizados para comparação entre dois valores que são utilizados na maioria das estruturas de seleção conforme Manzano e Oliveira (2012).

Tabela 2 - Operadores relacionais

Operador	Função
=	Igual
>	Maior que
<	Menor que
>=	Maior ou igual a
<=	Menor ou igual a
<>	Diferente

Fonte: (MANZANO; OLIVEIRA, 2012, p.43).

As estruturas de decisão são divididas em três formas básicas, sendo elas: SE-ENTÃO, SE-ENTÃO-SENÃO e CASO. Essas três estruturas são utilizadas de diferentes modos:

Na estrutura SE-ENTÃO, tem por finalidade tomar uma única decisão, sendo ela verdadeira ou falsa, como por exemplo: SE idade > 18 ENTÃO ... {ação para idade verdadeira} conforme mostrado na Figura 3.

Figura 3 - Estrutura de Decisão SE-ENTÃO.

```

//Função : Verificar se a idade do usuário é maior que 18
//Identificação do Algoritmo
algoritmo "estrutura_se_entao"

var
    idade: inteiro

inicio
    Escreva ("Qual é a sua idade? ")
    Leia (idade)
    //Estrutura de condição, para verificar se a idade do usuário
    //é maior ou igual a 18
    Se(idade >= 18) entao
        Escreva ("Você é maior de idade")
    //Finaliza a estrutura de condição se-entao
    fimse
fimalgoritmo

```

Fonte: Autor.

Observa-se que quando a expressão for verdadeira, um comando (ou grupo de comandos) é executado; caso falso, o comando não é realizado. Por este motivo, para

executar outra ação, mediante o resultado falso, deve ser implementada a estrutura SE-ENTÃO-SENÃO.

Estrutura SE-ENTÃO-SENÃO é incluído um comando ou sequência de comandos caso uma expressão for satisfatória, sendo ela verdadeira ou falsa. A diferença em relação à estrutura SE-ENTÃO é a utilização de uma decisão tanto para falso quanto para verdadeiro. Considerando o exemplo anterior, onde a pergunta era *Qual é a sua idade*, ao executar o comando SE-ENTÃO, existia apenas uma possibilidade de ação mapeada, para a idade maior ou igual a 18. Em caso contrário, se não for igual ou maior que 18, nada era retornado pelo sistema. Para isto, é adicionado o comando SENÃO, em que se a idade não for igual ou maior que 18 uma resposta também é mapeada e exibida pelo sistema, conforme demonstrado na Figura 4.

Figura 4 - Estrutura de Decisão SE-ENTÃO-SENÃO.

```

//Função : Verificar se a idade do usuário é maior que 18
//caso contrário mostra a mensagem para o usuário que é menor de 18
//Identificação do Algoritmo
algoritmo "estrutura_se_senao"

var
    idade: inteiro

inicio
    Escreva ("Qual é a sua idade? ")
    Leia (idade)
//Estrutura de condição, para verificar se a idade do usuário
//é maior ou igual a 18
    Se(idade >= 18) entao
        Escreva ("Você é maior de idade")
//Caso a condição acima não seja atendida, é executado outro
//comando ou conjunto de comandos
    Senao
        Escreva ("Você é menor de idade")
//Finaliza a estrutura de condição se-entao
    fimse
fimalgoritmo

```

Fonte: Autor.

A estrutura CASO é utilizada quando é preciso escolher entre mais de um caminho como, por exemplo, um menu de opções de 1 a 4, em que é executada a ação correspondente à opção que o usuário deseja escolher. Observa-se que essa estrutura não possui limitação de escolha, porém deve ser definido um valor padrão

(*default*) caso o usuário não queira escolher nenhuma das opções mapeadas. Na Figura 5 tem-se um exemplo de como funciona a estrutura CASO.

Figura 5 - Estrutura de Decisão CASO.

```
//Função : Escolher um time de futebol
//Identificação do Algoritmo
algoritmo "estrutura_caso"

//Seção de declaração das variáveis
var
//Variável idade do tipo inteiro
    time : inteiro;

//Inicio da seção de comandos
inicio
//Função que mostra a mensagem para o usuário
    Escreval ("Qual é o seu time favorito? ")
    Escreval ("1 - Internacional")
    Escreval ("2 - Grêmio")
    Escreval ("3 - Juventude")
    Escreval ("4 - Brasil de Pelotas")
    Escreval ("0 - Outro")
//Função que mostra a mensagem para o usuário
    Escreva ("Digite a Opção: ")
//Função que aguarda o usuário digitar o numero da escolha
    Leia(time)
//Estrutura de condição CASO, verifica qual opção o usuário escolheu
    Escolha time
//Caso a escolha for 1 será mostrado a mensagem do Caso 1
        Caso 1
            Escreval ("Você escolheu Internacional.")
//Caso a escolha for 2 será mostrado a mensagem do Caso 2
        Caso 2
            Escreval ("Você escolheu Grêmio.")
//Caso a escolha for 3 será mostrado a mensagem do Caso 3
        Caso 3
            Escreval ("Você escolheu Juventude.")
//Caso a escolha for 4 será mostrado a mensagem do Caso 4
        Caso 4
            Escreval ("Você escolheu Brasil de Pelotas.")
//Caso a escolha for 0 ou qualquer outro numero ou letra,
// será mostrado a mensagem default em Outrocaso
        Outrocaso
            Escreval ("Você prefere outro time.")
//Finaliza a estrutura CASO
    Fimescolha
//Fim da seção de comandos
finalgoritmo
```

Fonte: Autor.

É importante destacar que a estrutura CASO pode ser implementada com o uso de sucessivos SE-ENTÃO-SENÃO. No entanto, a utilização de CASO torna o algoritmo mais claro e simplificado (código mais simples e não “poluído”).

2.1.3. Estrutura de repetição

A estrutura de repetição, também conhecida como laço de repetição (*looping*), tem por finalidade repetir um comando ou bloco de comandos por um número determinado de vezes (MANZANO e OLIVEIRA, 2012). A estrutura de repetição pode ser classificada de duas formas, estrutura de repetição interativa ou iterativa. A estrutura de repetição interativa necessita da intervenção do usuário para repetir a próxima ação, já a estrutura iterativa define repetições executadas de forma automática, pré-definidas pelo usuário.

Um exemplo básico de como funciona a estrutura de repetição é o ato de colocar água em um copo. No momento em que é colocada a água dentro do copo, a pessoa diz até onde ela gostaria de ser servida. A representação em um algoritmo seria algo como: adicione a água no copo até chegar na quantidade pedida, quando chegar na quantidade pedida, pare de colocar água. Existem três tipos de estruturas de repetições, são eles: repetição com teste no início, repetição com teste no final e repetição com variável de controle.

A repetição com teste no início faz com que seja verificada uma condição antes de executar o bloco de código. Ou seja, para executar o código relacionado à repetição, antes deve ser executado um teste condicional, o qual, se verdadeiro, executa a repetição; caso contrário, a repetição não será executada. Para realizar essa repetição, é utilizada uma estrutura denominada *enquanto* no início do fluxo de execução, conforme apresentado na Figura 6.

Figura 6 – Estrutura de repetição ENQUANTO – FAÇA.

```

// Função : Contar em ordem crescente de 1 até 10
//Identificação do Algoritmo
algoritmo "estrutura_repeticao_enquanto"

var
  n : inteiro

inicio
  n := 1 // := sinal de atribuição de dados
  enquanto n <= 10 faça //Enquanto n(1) for menor que 10 repita o comando escreva
    escreval (n)
  //Encrementação do número a cada repetição em que
  //n na primeira execução será 1, depois 2 até o n chegar a 10
  n := n+1
  //Finaliza a estrutura de repetição
  fimenquanto
finalgoritmo

```

Fonte: Autor.

Já a repetição com teste no final permite que pelo menos uma vez a estrutura de repetição seja executada, pois o teste condicional é realizado somente no final da repetição. Quando a execução chega ao final da repetição, será verificado se a condição foi atendida ou não. Em caso de ter sido atendida, a estrutura de repetição é interrompida; caso contrário, será executada até atender a condição. Para esta situação descrita, é utilizada uma estrutura de repetição denominada *repita*, conforme apresentado na Figura 7.

Figura 7 - Estrutura de repetição REPITA.

```

// Função : Contar em ordem crescente de 1 até 10
//Identificação do Algoritmo
algoritmo "estrutura_repeticao_repita"
var
  n : inteiro
inicio
  n := 1
  //Estrutura de repetição repita
  repita
    escreval (n)
    n := n+1
  //Enquanto n(1) for menor que 10 repita o(s) comando(s) acima
  //até satisfazer a condição ate n > 10 finalizando a estrutura de repetição
  ate n > 10
finalgoritmo

```

Fonte: Autor.

Por fim, tem-se a repetição com variável de controle, em que existe uma condição pré-definida para executar a repetição por um determinado número de vezes, algo que nem sempre existe nas repetições anteriores. Esta estrutura se chama *para*, sendo que nela é preciso que exista uma variável de controle (número inteiro), um valor inicial da variável de controle, um valor final (quantas vezes esse laço irá repetir) e, por fim, um valor de incremento ou decremento. Esta estrutura é apresentada no algoritmo da Figura 8.

Figura 8 - Estrutura de repetição PARA-ATÉ-FAÇA.

```
// Função : Contar em ordem crescente de 1 até 10

algoritmo "estrutura_repeticao_para"
var
  i : inteiro

inicio
  //Estrutura de repetição Faça
  Para i de 1 Ate 10 Faça
  //Função que mostra os valores de 1 até 10
    escreval (i)
  Fimpara
finalgoritmo
```

Fonte: Autor.

Os exemplos de estruturas de repetição citados nesta seção não se limitam apenas às estruturas sem a intervenção do usuário, isto é, estruturas de repetição pré-configuradas. É possível que o usuário realize esse controle conforme apresentado na Figura 9, em que o mesmo informa o dado inicial e o dado final.

Figura 9 - Estrutura de repetição PARA-ATÉ-FAÇA de forma interativa.

```

// Função : Contar em ordem crescente de um intervalo
//de dados informado pelo usuário
algoritmo "estrutura_repeticao_para"
var
    i, n1, n2 : inteiro

inicio
    Escreval("Digite o primeiro numero:")
    Leia(n1)
    Escreval("Digite o segundo numero:")
    Leia(n2)
//Estrutura de repetição Faça
    Para i de n1 Ate n2 Faça
//Função que mostra os valores dentro do intervalo
//informado pelo usuário
        escreval (i)
    Fimpara
//Fim da seção de Comandos
finalgoritmo

```

Fonte: Autor.

Como foi observado nos exemplos anteriores, as estruturas podem ser previamente configuradas ou de forma dinâmica, sendo que quem irá realizar o controle é o usuário. Além desta observação, as estruturas podem ser implementadas com uma ou mais estruturas, isto é, dentro de uma estrutura CASO, por exemplo, é possível implementar um PARA-ATE-FAÇA ou um SE, tudo vai depender da complexidade do problema cuja solução o usuário vai representar por meio de um algoritmo.

2.2. ENSINO DE LÓGICA POR MEIO DO PENSAMENTO COMPUTACIONAL

Conforme Wing (2006), a combinação do pensamento crítico com os fundamentos computacionais define uma metodologia para a resolução de problemas chamada de pensamento computacional. Esta metodologia faz com que seja possível tomar decisões mais lógicas, fazendo com que se raciocine antes do impulso, conseguindo, assim, tomar decisões mais coerentes nas mais diversas áreas do conhecimento.

Nesse contexto, Wing (2006) exemplifica que o pensamento computacional surge como um conjunto de técnicas para a resolução de problemas baseado no pensamento crítico e fundamentos conceituais da Ciência da Computação. A autora

também afirma que o pensamento computacional é uma habilidade fundamental para qualquer pessoa e não somente para cientistas da computação. Já para Blikstein (2008), o pensamento computacional envolve saber usar o computador como um instrumento de aumento do poder cognitivo e operacional humano, aumentando a produtividade, inventividade, e criatividade. Barr e Stephenson (2011) especificam seis características do Pensamento Computacional, descritas a seguir:

- a) formular problemas de forma a permitir que se utilize um computador e outras ferramentas para ajudar a resolver problemas;
- b) organizar e analisar os dados de forma lógica;
- c) representar dados através de abstrações, tais como modelos e simulações;
- d) automatizar soluções por meio do pensamento algorítmico (uma série de passos ordenados);
- e) identificar, analisar e implementar possíveis soluções com o objetivo de alcançar a combinação de passos e recursos mais eficiente e eficaz;
- f) generalizar e transferir o processo de resolução de problemas a uma ampla variedade de problemas.

Um exemplo básico de como funciona o conjunto de técnicas do pensamento computacional na resolução de problemas é a montagem de um quebra-cabeça no qual, para finalizar a montagem, cada pessoa terá uma forma (estratégia) para conseguir resolver o problema. Usando o pensamento computacional para resolver esse problema, se pode separar as peças por cor ou por extremidades, assim utiliza-se uma característica da ciência da computação que é a decomposição, em que há uma quebra de um problema maior em partes menores para tornar mais fácil a montagem. Neste momento, é utilizada a primeira característica que é a análise e organização dos dados de forma lógica.

A segunda etapa é a resolução desses problemas menores, em que é realizada a montagem das peças que foram separadas na primeira etapa, entrando a simulação com base em tentativa e erro e a abstração dos dados, uma vez que essa mesma lógica da montagem pode ser utilizada para resolver outros problemas. Nesta etapa, podem ser utilizadas as demais características como identificar e implementar uma possível solução, automatizar a solução via pensamento algorítmico (uma série de passos ordenados) por meio do pensamento crítico e do raciocínio lógico e

desenvolver e aplicar a solução para a finalização da montagem e utilizá-la para resolver os mais diversos problemas.

Andrade *et al.* (2013) citam em seu trabalho que existem três fundamentos básicos que compõem o pensamento computacional: abstração, automação e análise, os quais foram chamados de 3A. A *abstração* é um processo mental para separar um ou vários elementos de um problema maior com o objetivo de facilitar a compreensão do problema, isto é, remover os detalhes desnecessários realizando um filtro. A *automação* é o desenvolvimento de uma aplicação para automatizar a resolução do problema, fazendo com que haja a substituição do trabalho manual pelo eletrônico. Como os autores citaram, o computador é um ótimo exemplo dessa substituição. E, por fim, tem-se a *análise*, que consiste em avaliar o resultado gerado pela automação.

2.3. CONSIDERAÇÕES DO CAPÍTULO

Este capítulo teve como objetivo explicar o que é lógica de programação e como trabalhá-la por meio de algoritmos, exemplificando com situações do dia-a-dia. Além disso, foram abordadas as estruturas de programação, tais como estrutura sequencial, estrutura de decisão e estrutura de repetição. Para finalizar, foi discutido o ensino de lógica por meio do Pensamento Computacional, que apresenta uma explicação sobre o Pensamento Computacional e seus fundamentos.

As estruturas são a base para o desenvolvimento de qualquer algoritmo, pois necessitam uma ordem para seu desenvolvimento. Aqui tem-se a estrutura sequencial. Porém, muitas vezes faz-se necessária a tomada de decisão. Isto implica o uso de uma estrutura de decisão. No final, para que se tenha um algoritmo mais limpo, ou seja, organizado, o uso da estrutura repetição pode ajudar, pois auxilia na repetição de um determinado bloco de comandos. Considerando sua importância, é crucial que os estudantes consigam compreender adequadamente o objetivo de cada estrutura, e as particularidades inerentes de cada uma, para poder utilizar a mais adequada em cada situação, bem como integrar as mesmas quando necessário.

É fundamental destacar a complexidade envolvida em apreender tais conhecimentos, considerando que desenvolver o raciocínio lógico para resolução de problemas envolve, como descrito no capítulo, diversas questões, como organização e análise dos dados, interpretação e compreensão do problema, análise e escolha da melhor alternativa para intervenção e solução do problema e, principalmente, a

capacidade de generalizar e conseguir utilizar tais conhecimentos na resolução de outros problemas e intervenções similares.

Já que o objetivo principal desse trabalho é apoiar o processo de ensino aprendizagem de lógica de programação por meio de um Jogo Sérioso, no próximo capítulo serão abordados conceitos gerais de jogos digitais, descrevendo seus principais elementos, bem como os gêneros em que os mesmos podem ser enquadrados.

3. JOGOS EDUCACIONAIS

Conforme Pinto *et al.* (2008), os jogos educacionais são atividades lúdicas que integram objetivos pedagógicos para desenvolver o raciocínio e aprendizagem tanto em crianças quanto em adultos. Os jogos educacionais podem facilitar a aprendizagem por meio de diversão e da motivação criada. Considera-se que os jogos, se bem planejados, podem ser um recurso pedagógico eficaz para a construção do conhecimento.

Para a elaboração dos jogos digitais, algumas características ou elementos devem ser considerados para que o desenvolvimento de um jogo seja um sucesso. Os jogos possuem gêneros ou *estilo*, que são um conjunto de características como jogabilidade, interação e apresentação gráfica, que serão apresentados na seção 3.1. Além de destacar os elementos e os gêneros dos jogos, na seção 3.2 serão abordados os Jogos Sérios e seu papel na educação, foco desta dissertação.

3.1. CONCEITOS GERAIS

Autores como Salen e Zimmerman (2004) definem os jogos digitais como um sistema no qual os jogadores se engajam em um conflito artificial, definido por regras, cujo resultado é quantificável. Desta maneira, Huizinga (2007) elenca cinco elementos fundamentais para o sucesso de qualquer jogo: 1) Liberdade; 2) Delimitação; 3) Imprevisibilidade; 4) Regulamentos e normas, e 5) Ficção (ou distância da realidade).

Um dos elementos mais importantes, porém pouco conhecido, de acordo com Huizinga (2007), é a *Liberdade* do jogador perante o jogo, pois nem sempre o desenvolvedor sabe como ele pode reter a atenção do usuário. Outra questão elencada pelo autor e que busca a liberdade do jogador é a escolha de com quem e onde jogar, isto é, a liberdade do usuário em poder escolher com qual personagem ele se identifica e para quais lugares ele pode ir com o personagem, independente do gênero de jogo, seja ele RPG, luta ou corrida, entre outros.

O elemento *Delimitação* do jogo diz respeito ao espaço em que o jogo será realizado, delimitando o tema e local em que serão realizadas as partidas. Pode ser em ambientes como uma cidade fictícia, futurística e, até mesmo, um local existente no espaço real como, por exemplo, uma sala de aula. Huizinga (2007) utiliza como exemplo o jogo Batman Arkhan City, um jogo de ação-aventura baseado na série de

quadrinhos Batman, no qual existe uma cidade chamada Arkhan City onde Batman, protagonizado por Bruce Wayne, tenta escapar de uma “super-prisão” (Arkhan City), enfrentando, assim, vários vilões. Ou seja, foi criado um ambiente que possui prédios, casas, vilões e desafios, o que define um enredo voltado para um público que busca ação e diversão, corroborando a busca pelo elemento liberdade supracitado.

A *Imprevisibilidade* é uma expectativa gerada automaticamente por uma força maior, sorte ou Inteligência Artificial (IA), em que o jogador acaba recebendo um resultado diferente do que esperava. Um exemplo simples de imprevisibilidade é um jogo de futebol, no qual o histórico dos dois times pode favorecer um ou outro e no final da partida o time que parecia não ter chances consegue vencer.

Outro elemento importante são os *Regulamentos e Normas*, pois dificilmente um jogo funciona de forma efetiva se não tiver um regulamento ou norma, isto é, sem regras um jogo pode se tornar inútil e sem objetivo. Conforme Arruda (2014, p21):

As regras existem para que todos possam ter as mesmas condições iniciais de se desenvolverem e, a partir daí, obterem sucesso e se sobressaírem, vencendo o jogo por seu esforço, conhecimento e habilidade. Assim, as normas e regras servem para nivelar os jogadores e fazer a vitória ser justa.

Sendo assim, as regras são essenciais para a manutenção do jogo, tornando-o desafiador, fazendo com que os jogadores possam competir e ganhar suas recompensas pelo esforço realizado durante o jogo.

Por fim, tem-se o elemento *Ficção*, no qual o grande ponto encontra-se em situar o jogo em um ambiente irreal, utilizando equipamentos que seriam muitas vezes impossíveis na vida real. Para Arruda (2014), um jogo de corrida, por exemplo, apresenta uma noção de como se dirige um carro, mas na vida real isso não seria possível por um simples motivo, o reinício. Isso quer dizer, podem ser realizadas manobras complexas em uma pista de corrida em um jogo digital, tais como bater, ultrapassar de forma arriscada, capotar, enfim, atividades que não são possíveis na vida real. O capotamento de um carro, na vida real, pode ser fatal, sendo que não há reinício na vida real. A ideia de criar simulações, ficção científica e construções exageradas é característica da ficção.

Todos os elementos citados são importantes para a elaboração de um jogo, sendo que cada gênero de jogo pode possuir um ou mais pontos fortes em relação

aos elementos. Após apresentado os elementos de um jogo eletrônico, faz-se necessária a apresentação quanto ao seu gênero ou estilo.

Escolher um gênero para um jogo pode se tornar complexo, pois o mesmo jogo pode fazer parte de um ou mais gêneros. Conforme Arruda (2014, p46):

Gênero de um jogo, é o mesmo que o tipo de jogo. Pode ser considerado um termo que engloba todas as características de um conjunto de jogos e que permite organizar e separar os jogos de acordo com as semelhanças entre eles.

Belli e Raventós (2008) complementam que gênero pode ser classificado pela sua apresentação gráfica, o tipo de interação entre o jogador e o computador, e, por fim, o seu ambiente e sistema de jogabilidade.

Na tabela 3, é apresentada a classificação de um jogo quanto ao gênero, conforme pesquisa da *Entertainment Software Association (2015)*, uma empresa especializada em jogos digitais que realiza pesquisas anualmente nos Estados Unidos.

Tabela 3 - Classificação de jogos quanto ao gênero.

(Continua)

Gênero	Significado
Ação	São jogos nos quais é preciso agir conforme a situação, movimento e atividade. Em um jogo de ação, o personagem assume um papel e, por meio deste, realiza tarefas e desafios ao longo da história. Ação é o mais amplo dos gêneros conforme Arruda (2014) e Belli e Raventós (2008).
Ação em 1ª Pessoa	Também conhecida como FPS (<i>First Person Shooter</i>), o gênero de ação em 1ª pessoa é quando a câmera mostra o campo visual como se o jogador estivesse no jogo. Bons exemplos para este gênero são jogos como: Doom, Quake e Counter Strike

Tabela 3 - Classificação de jogos quanto ao gênero.

(Continua)

Gênero	Significado
Ação em 3ª Pessoa	Segue o mesmo estilo do gênero anterior, a grande diferença é a total liberdade do personagem e a sua visão isométrica que podem ser vistas todas as ações do personagem. Destacam-se jogos com essas características: Tomb Raider, GTA e Max Payne.
Arcade	Os jogos de Arcade se destacam pela simplicidade e ação rápida. São jogos com apenas alguns comandos de ação como, por exemplo, direcional de um controle ou um/dois botões de ação a mais. Isto é, possui jogabilidade simples. São exemplos desse gênero: Pac-Man, Asteroids e Space Invaders.
Aventura	Nos jogos de aventura o jogador assume o papel principal da história, realizando desafios e recebendo recompensas. Alguns exemplos são: Monkey Island e Myst.
<i>Beat them up</i>	É um gênero de briga de rua, isto é, um jogo de luta com estágio, em que ao final de cada estágio o jogador deve lutar contra um inimigo mais forte. Bons exemplos para este gênero são: Capitão Comando, Final Fight e Streets of Rage.
Corrida	Como o próprio nome diz, são jogos que trabalham a disputa de corridas, sendo de carro, bicicleta, moto, entre outros. Dentre os jogos que se destacam estão: Need for Speed, Forza e Grand Turismo.
Educacional	Os jogos educacionais possuem uma finalidade específica que é o cunho pedagógico, memorização de conceitos e aprendizagem. Exemplos: Perguntados, Sílabas e Soletrando.

Tabela 3 - Classificação de jogos quanto ao gênero.

(Continua)

Gênero	Significado
Esporte	Os jogos de Esporte simulam esportes reais, como futebol, basquete, skate entre outros. Alguns exemplos de clássicos do esporte: FIFA, Pro Evolution Soccer, NBA e Tony Hawk's Pro Skater.
Estratégia	Considerado o gênero mais complexo, pois se trata de um jogo em que o jogador deve refletir para realizar a melhor estratégia para ganhar um jogo. Entre as habilidades estão: construir casas, carros, tropas e planetas. Alguns jogos são: Age of Empires, Age of Mithology e Warcraft.
Luta	Jogos de luta são caracterizados pelo combate corpo a corpo com dois ou mais componentes. Exemplos deste gênero são: UFC, Mortal Kombat e Street Fighter.
Musicais	É um gênero mais atual que tem como tema, a música e trabalha habilidades de audição e agilidade. Destacam-se: Guitar Hero, Dj Hero e Singer.
Plataforma	No jogo de plataforma, o jogador assume o papel de um personagem que avança pelo cenário (normalmente da esquerda para direita, vice-versa ou ambos), pulando objetos, correndo, pegando recompensas, vencendo inimigos e, ao fim, lutando contra um inimigo superior aos demais e único, sendo o principal inimigo do jogador. Os mais famosos deste gênero são: Super Mario, Sonic e Megaman.

Tabela 3 - Classificação de jogos quanto ao gênero.

		(Conclusão)
Gênero	Significado	
Simulação	São jogos voltados para o realismo, que é determinado por apresentar atos que dificilmente o jogador poderia realizar fora da tela sem consequências para si ou para os outros. Na simulação, existem diversos subgêneros, tais como simulação de guerra como America's Army, simulador de trânsito para obtenção da carteira de motorista do DETRAN, simulador de vida real como Second Life, entre outros.	

Fonte: (BELLI; RAVENTÓS, 2008, p.167).

É importante destacar que um jogo pode assumir características de mais de um gênero descrito na tabela 2. É possível encontrar jogos que misturam elementos de ação, estratégia, plataforma, simulação entre outros. Um exemplo é o jogo Super Mario World, em que o mesmo assume mais de um gênero, sendo eles, plataforma e ação.

Os jogos, quando utilizados em um contexto não apenas de diversão/entretenimento, mas possuindo um cunho educacional, empresarial ou até mesmo organizacional, são denominados Jogos Sérios ou *Serious Games*. Como o foco deste trabalho é desenvolver um jogo educacional, integrando os gêneros ação e plataforma, a próxima seção destacará Jogos Sérios.

3.2. JOGOS SÉRIOS

Os Jogos Sérios ou *Serious Games* são um tipo de jogo em que o foco principal é a aprendizagem e treinamento e não somente a diversão. Para Freitas e Liarokapis (2011), os Jogos Sérios possuem um grande potencial para complementar a educação tradicional, sendo que podem oferecer um novo paradigma para complementar a educação de forma lúdica e divertida. Para Scaico *et al.* (2012), é possível aprender com jogos porque um aprendiz está mais suscetível a aprender quando a aprendizagem não é forçada.

Um dos problemas mais encontrados em artigos e publicações é o termo Jogos Sérios como Gamificação. De acordo com Fadel *et al.* (2014), a Gamificação abrange

a utilização de mecanismos de jogos para a resolução de problemas para a motivação e o engajamento de um determinado público, ou seja, a Gamificação utiliza as mecânicas mais eficientes de um jogo, utilizando assim, para a elaboração não somente de recompensas, mas também de um estímulo para que o usuário tente alcançar os objetivos e ganhar os desafios.

Susi *et al.* (2007) definem Jogos Sérios como uma aplicação das tecnologias de jogos, processos e *design* para soluções de problemas enfrentados pelos negócios e outras organizações. Os Jogos Sérios promovem a inter-relação do conhecimento de desenvolvimento de jogos e técnicas do mercado e educação, como *marketing*, *design* de produtos, vendas, aprendizagem de conteúdos educacionais entre outros.

As áreas de aplicação de Jogos Sérios são diversas: Militar - são jogos voltados para área militar como, por exemplo, America's Army¹ em que o objetivo é simular uma guerra e aplicar estratégias para sobreviver e modos de atacar, entre outras atividades relacionadas à área militar; governamental - jogos voltados para simulação governamental, como tarefas e situações específicas de gestão; educacional - voltado para o ensino e aprendizagem de conteúdos educacionais, simulações de exercícios, entre outros; corporativos - são jogos voltados para a indústria, são muitas vezes simulações de equipamentos como montagem e uso, treinamentos e habilidades organizacionais; e assistência médica - jogos voltados para a saúde como simulação de cuidados médicos, simulação de cirurgia, treinamentos, diagnósticos e tratamentos.

3.3. CONSIDERAÇÕES DO CAPÍTULO

Neste capítulo foram abordados os conceitos necessários para que se obtenha sucesso no desenvolvimento de um jogo, de acordo com Huizinga (2007). Além desses conceitos, foram descritos os gêneros (tipo de interação, apresentação gráfica e o ambiente do jogo) em que se apresentam os jogos, sendo que cada jogo pode possuir mais de um tipo de gênero e, por fim, a apresentação de Jogos Sérios assim como suas características, áreas de aplicação e exemplos práticos desses jogos.

Sendo assim, não basta somente ter a ideia de um jogo e iniciar seu desenvolvimento. Há a necessidade de definir um roteiro para seguir, elencar as

¹ America's Army – Disponível em www.americasarmy.com

características marcantes do jogo e escolher em qual gênero o jogo pretendido se encaixa. No próximo capítulo, serão abordados trabalhos correlatos ao que está proposto neste projeto, com o objetivo de analisar os mesmos e buscar extrair suas principais potencialidades que possam ser inseridas no jogo proposto, além de discutir aspectos importantes destes trabalhos.

4. TRABALHOS CORRELATOS

Para o desenvolvimento do jogo proposto neste trabalho, foi necessária a realização de um levantamento bibliográfico sobre Jogos Sérios desenvolvidos para a área de lógica de programação excluindo aplicativos e ambientes por não serem o foco deste projeto. Esse levantamento teve por finalidade verificar como foram desenvolvidos os jogos citados neste capítulo, o tipo e a plataforma desenvolvida e os conceitos aplicados obtendo, assim, subsídios para elaboração do jogo proposto neste projeto.

Um exemplo destes trabalhos é o *Castelo dos enigmas*, proposto por Scaico *et al.* (2012), jogo de RPG (*Role-Playing Game*) desenvolvido para o Sistema Operacional Android. A história do jogo envolve um aluno que realizará uma avaliação de programação no dia seguinte e que só começa a estudar à noite. Após o personagem pegar no sono, ele começa a sonhar que está em um castelo onde o objetivo é sair do mesmo para ir até a escola e realizar a prova. O jogo possui níveis de dificuldades graduais, sendo que a cada desafio conquistado no *Castelo dos enigmas* um assunto proposto é estudado.

No entanto, o trabalho de Scaico *et al.*(2012) não apresenta a aplicação do jogo com possíveis contribuições e resultados encontrados, limitando-se a apresentar apenas a proposta do jogo. Após a análise do jogo em seu âmbito real, observou-se que o jogo possui apenas um nível implementado, o condicional, conforme representado pela interface do jogo (Figura 10).

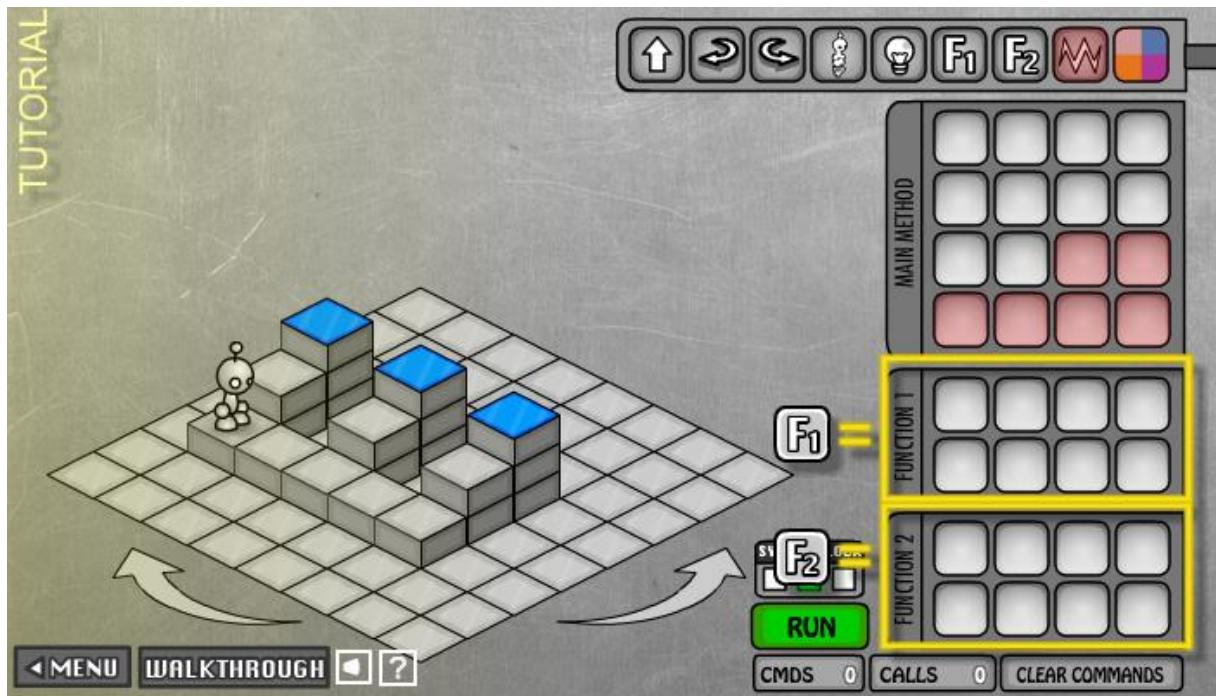
Figura 10 - Jogo Castelo dos Enigmas.



Fonte: (SCAICO *et al.*, 2012).

Outro trabalho interessante é o jogo *Light-Bot 2.0* (Figura 11). Este é um jogo que utiliza a tecnologia Adobe Flash³ e pode ser jogado em qualquer navegador que possua o *plugin* instalado. O jogo aborda a jornada de um robô que deve chegar ao seu destino utilizando blocos de *passo à frente*, *vire a esquerda*, *vire a direita*, *pular*, *acender luz*, *função 1* e *função 2*.

Figura 11 - Jogo Light-Bot 2.0 desenvolvido pela Armor Games.



Fonte: (ARMOR GAMES¹).

O bloco de *passo à frente* faz o robô dar um passo à frente para onde ele estiver direcionado; *vire à esquerda* não dá nenhum passo, apenas rotaciona uma vez o robô para à esquerda; *vire à direita* rotaciona o robô para a direita; *pular* faz com que o robô suba obstáculos, porém, apenas com um passo, *função 1* e *função 2* correspondem a um bloco de repetição de vários passos previamente criados.

Após o robô chegar ao seu destino representado por um quadrado azul, ele deve acender uma luz para concluir o objetivo. O jogo começa com um tutorial bem intuitivo, porém os ícones não possuem uma facilidade para a compreensão da lógica

² Light-Bot 2.0 – Criado pela Armor Games no link <http://armorgames.com/play/6061/light-bot-20>

³ Adobe Flash – Criado pela Macromedia adquirida pela Adobe disponível em <http://www.adobe.com/br/products/animate.html>

de programação. Um dos problemas encontrados no *Light-Bot* foi a falta de documentação acadêmica, pelo fato do jogo ser comercial.

No entanto, alguns trabalhos relatam análises do uso do jogo no contexto educacional, como Gouws, Bradshaw e Wentworth (2013) que, em seu trabalho, desenvolveram uma tabela (Figura 12) para avaliar o jogo, conforme as habilidades do Pensamento Computacional, bem como os diferentes níveis de habilidade praticados, sendo que, para os autores, o jogo se adequa às habilidades exigidas pelo Pensamento Computacional.

Figura 12 - *Computational Thinking Framework (CTF)*.

		Identificar	Compreender	Aplicar	Assimilar
Processos Transformações	e				
Modelos Abstrações	e	2			
Padrões Algoritmos	e				
Ferramentas Recursos	e				
Dedução e Lógica					
Avaliação Progresso	e				

Fonte: (GOUWS; BRADSHAW; WENTWORTH, 2013, p.11)

Gouws, Bradshaw e Wentworth (2013) aplicaram um estudo com estudantes, utilizando a escala de Likert para avaliar cada uma das habilidades contidas no *Computational Thinking Framework (CTF)* e, como resultados, obtiveram uma média de 74%, sendo destes, 75% para o Processo e Transformações, em que foi observado que o jogo reforça a habilidade, porém não há a possibilidade de manipular dados de entrada e saída como adição de valores e exibição de mensagem para o usuário; no modelo e abstração, obteve a maior avaliação, 92%, pela questão de possuir *signos* de fácil compreensão para o aluno.

As habilidades de Padrões e Algoritmos obteve 58%, por mais que o jogo traga algumas noções de programação, falta estruturas básicas como repetições e principalmente estruturas de decisões, para que o aluno possa resolver o problema por mais de um caminho; Ferramentas e Recursos, aqui os alunos elencam que poderia existir uma implementação para salvar algumas sequências de códigos e não

se limitar a duas funções, mesmo assim em um aspecto geral esta habilidade recebeu 83%.

Na Dedução e Lógica, embora o aluno deva pensar para resolver os desafios, o jogo não oferece muito espaço para práticas além do que é pedido, mais uma vez é reforçado pelos autores a falta de conceitos randômicos e habilidades condicionais, sendo que esta habilidade obteve nota 50%; e, por fim, as Avaliações e Progresso, esta habilidade atingiu 83%, sendo que aqui é abordado que se por algum motivo o aluno consegue resolver o desafio em menor número de comandos, o mesmo não recebe nenhuma recompensa a não ser a *satisfação pessoal*.

Além disso, algumas melhorias seriam necessárias, como a aplicação de estruturas de repetições e estruturas condicionais, além de correção de erros como o comportamento inesperado do robô.

No trabalho de Borges et al. (2015) é apresentado o KidCoder, um Jogo Sério em forma de um ambiente de prática, contendo desafios graduais. Esses desafios tratam de combates de RPG e resolução de quebra-cabeças que devem ser solucionados por meio de escrita de códigos-fontes. De acordo com os autores, o jogo utiliza os elementos de fantasia, pois trata da imersão em um mundo fictício; narrativa, em que criada uma história em um contexto maior, particionado em pequenas histórias, as quais seriam os níveis graduais do jogo e, por fim, desafio por meio de combates entre o personagem principal e os inimigos, e a resolução de quebra-cabeças. Como o jogo trata-se de um protótipo, o trabalho não apresenta quaisquer aplicações do mesmo com possíveis contribuições e resultados e nem uma demonstração para avaliá-lo.

Kazimoglu et al. (2012) apresentam, em seu trabalho, o Jogo Sério *Program Your Robot* (Figura 13), similar ao *Light-Bot*, que tem como objetivo chegar a um destino para se teletransportar para o próximo nível. O jogo proposto pelos autores se diferencia pelo fato de ser desenvolvido não para fins de diversão e sim para fins de aprendizagem, para o desenvolvimento de competências do Pensamento Computacional. O jogo se destaca em relação ao *Light-Bot* pelas correções das funções e melhorias que os autores Gouws, Bradshaw e Wentworth (2013) abordaram, como a criação de estruturas condicionais e de repetição, obtenção de prêmios em forma de pontuação, caso realizado em menor número de passos para

se atingir o objetivo da fase, e também a inserção de um depurador⁴ de passos, fazendo com que o aluno possa depurar cada comando implementado.

Figura 13 - Tela do jogo Program Your Robot.



Fonte: (KAZIMOGLU *et al.* 2012, p. 1995)

Kazimoglu *et al.* (2012) avaliaram o jogo em uma disciplina de programação do curso de Ciências da Computação na Universidade de Greenwich em Londres, para identificar pontos positivos e negativos do mesmo. A avaliação foi realizada por meio de *feedback* dos desafios aplicados, com vinte e cinco alunos de diferentes programas de graduação e com diferentes níveis de conhecimento. De acordo com os autores, os vinte e cinco alunos conseguiram concluir os exercícios e por meio dos *feedbacks* observou-se que o jogo é adequado para ajudar os alunos a compreender de forma introdutória a lógica de programação.

4.1. CONSIDERAÇÕES DO CAPÍTULO

Uma das grandes dificuldades enfrentadas em relação aos jogos desenvolvidos para auxiliar na aprendizagem de lógica de programação foi encontrar jogos de cunho acadêmico com resultados e metodologias. Mesmo assim, foi possível utilizar o que se julgou melhor de cada jogo, como mecânicas de funcionamento e novas ideias baseadas nos jogos analisados, como por exemplo: a utilização de blocos de

⁴ Depurador - é um programa de computador usado para testar outros programas e fazer sua depuração, que consiste em encontrar os defeitos do programa.

comandos para ajudar o usuário a entender o comando que será usado no desenvolvimento do algoritmo, na execução via linha de comando; e implementar as estruturas de repetição e de decisão, assim, trazer novas possibilidades de estruturas para que o aluno possa trabalhar outras habilidades relacionadas aos conteúdos.

Portanto, o próximo passo para a continuação do projeto é apresentar a abordagem metodológica que foi utilizada para o desenvolvimento do projeto que será abordada no próximo capítulo.

5. ASPECTOS METODOLÓGICOS

Inicialmente, foi realizada uma pesquisa bibliográfica para contemplar a aprendizagem de lógica de programação por estudantes de informática e o uso das tecnologias. De acordo com Gil (1991), a pesquisa bibliográfica se dá a partir da elaboração de material já publicado, constituído principalmente de livros, artigos, periódicos e, atualmente, com materiais disponibilizados na Internet.

Após a realização desta primeira etapa, passou-se ao levantamento de trabalhos correlatos para entender o desenvolvimento do jogo, uma vez que é importante, além de um embasamento teórico, analisar jogos ou aplicativos desenvolvidos com o mesmo objetivo. Em seguida, na etapa de planejamento do *software*, foram definidas as fases de desenvolvimento. Conforme Rezende (2005, p108):

Estudo preliminar, que é a visão global e genérica do projeto, sistema ou *software*; análise do sistema atual, visão global do atual sistema; projeto lógico, confecção de macro propostas de soluções, definição dos requisitos funcionais reais; projeto físico, execução, confecção de programas e seus respectivos testes; projeto de implantação, disponibilização, execução do planejamento de implantação.

Para o desenvolvimento do jogo proposto, optou-se por utilizar a metodologia INTERAD (PASSOS, 2011), desenvolvida para a elaboração de Objetos de Aprendizagem (OA), a qual foi considerada adequada para a elaboração do Jogo Sérió, pois contém etapas para o planejamento e desenvolvimento de *softwares* educacionais com foco no desenvolvimento das interfaces e interações do sistema, características fundamentais em um jogo. A INTERAD é composta pelas seguintes fases de desenvolvimento:

- a) *Compreensão*: esta primeira fase consiste no levantamento inicial das necessidades do OA, que começará a criar forma a partir da terceira etapa. Nesta fase, estão elencados os itens: Definição do tema; o público-alvo para o qual o OA será desenvolvido; os objetivos pedagógicos; a pesquisa institucional, local em que atuam os usuários, onde o OA será disponibilizado; o contexto educacional, ou seja, a modalidade educacional; necessidade do aluno (esse pode determinar

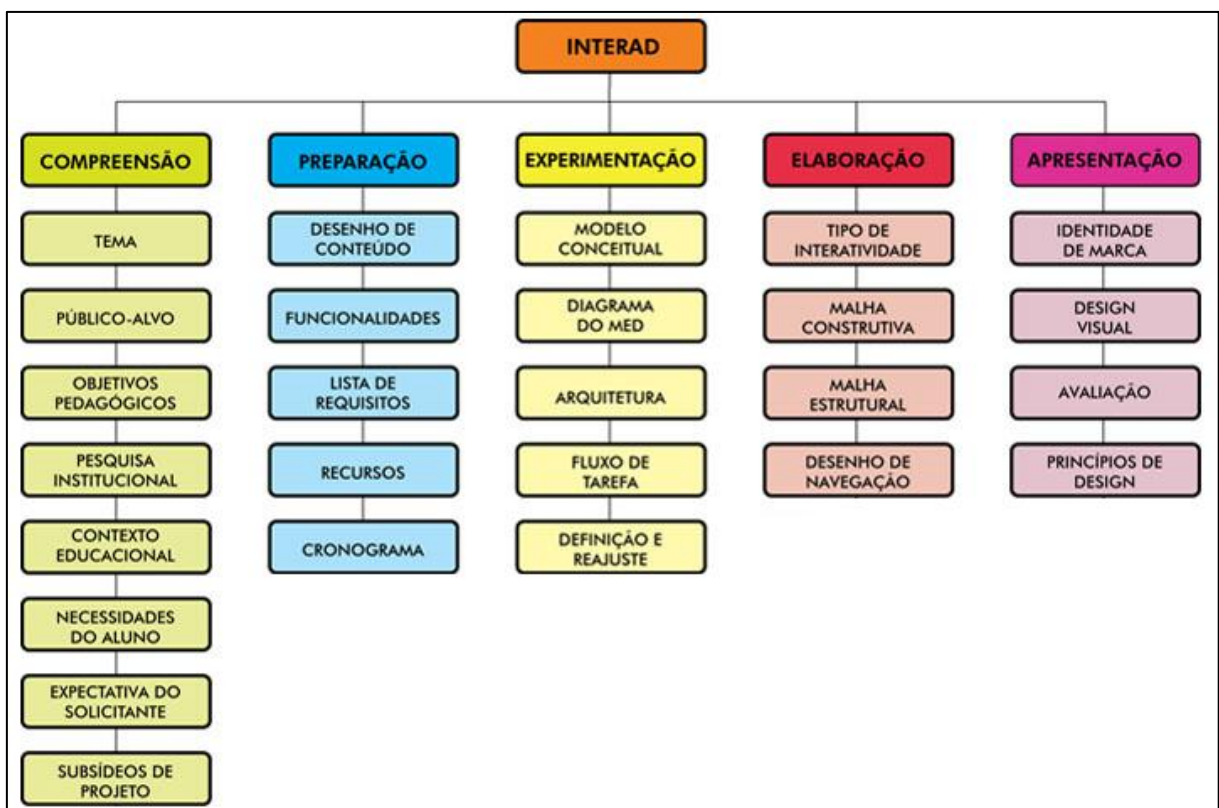
o sucesso ou o fracasso no desenvolvimento do OA); expectativa do solicitante e, por fim, os subsídios projetuais (o levantamento bibliográfico relevante ao projeto).

- b) *Preparação*: após a etapa da compreensão, é necessária a realização da preparação do material para a realização da próxima etapa como desenho de conteúdo, o qual trata da identificação completa do conteúdo a ser apresentado pelo OA; funcionalidades, são as ações que serão realizadas no OA; lista de requisitos, inclui a listagem das funcionalidades, metas de *design* de interação e a intenção de identidade visual do OA (deve estar em formato textual). Ainda, tem-se a descrição dos recursos, o que envolve a questão da viabilidade do projeto, sendo financeiro, RH (recursos humanos) ou limitações tecnológicas; e, para finalizar esta fase, tem-se o cronograma, sendo que é muito importante respeitar esse requisito, pois um mau planejamento pode acarretar na não conclusão do projeto ou falta de tempo para correção de problemas.
- c) *Experimentação*: a partir desta fase o OA passa a ser detalhado, pois trata da elaboração do modelo conceitual de interface, o qual é composto por: 1) modelo conceitual: modelo de todos os conceitos, dos signos que compreendem a interface; 2) diagrama do MED: mapa de forma hierárquica em que é apresentada a relação entre o conteúdo e a funcionalidade; 3) arquitetura: pode ser apresentada de quatro formas distintas (GARRET, 2003) sendo elas, estrutura sequencial, hierárquica, matricial e orgânica; 4) fluxo de tarefa: descrição das ações ou atividades que o usuário deve executar em determinadas tarefas do OA; e, 5) a definição e reajuste: consiste em refazer as etapas, caso necessário.
- d) *Elaboração*: com o modelo conceitual realizado, a fase da elaboração trata da escolha das interatividades no OA. Nesta fase são apresentados os seguintes itens: tipo de interatividade; malha construtiva; malha estrutural e desenho de navegação.
- e) *Apresentação*: essa fase pode ser considerada uma das essenciais, pois é durante a apresentação que ocorre o desenvolvimento e finalização do

projeto, em que são apresentados os seguintes itens: 1) a identidade visual, que trata da marca do OA desenvolvido; 2) o *design* visual, que é a conclusão do projeto de interface, o seu desenvolvimento e 3) avaliação, que busca analisar o objeto desenvolvido de acordo com critérios definidos.

A Figura 14 apresenta a metodologia INTERAD com suas fases e etapas, conforme apresentado por Passos (2011).

Figura 14 - Diagrama da metodologia INTERAD.



Fonte: (PASSOS, 2011).

Com relação às tecnologias escolhidas para o desenvolvimento do OA, foi utilizada a linguagem de programação C# e a *engine* de jogos Unity. A escolha da linguagem de programação C# se deve ao fato de ser flexível, pois utiliza, em sua codificação, os princípios de orientação a objetos (SEBESTA, 2009). Esses princípios possibilitam reutilizar o código de programação, permitindo uma redução do tempo de desenvolvimento, além de otimizar o código, deixando-o mais dinâmico.

Juntamente com a linguagem de programação C# foi utilizada a *engine* Unity, pois há grande facilidade no desenvolvimento de jogos em duas ou três dimensões

(2D e 3D), além de utilizar C# como linguagem principal de elaboração de *scripts*. Outra vantagem da Unity é o aspecto de portabilidade, o que possibilita que o jogo possa ser otimizado para ser executado em diversos dispositivos tecnológicos. Desta forma, existe a possibilidade de executar o jogo, por exemplo, em Sistemas Operacionais como Windows, Linux e Mac, bem como em plataformas móveis com Sistemas Operacionais como Android e iOS (UNITY, 2015).

Após concluído o desenvolvimento, foi realizada a etapa de avaliação do jogo por meio do método *GameFlow*, (Apêndice A) o qual aborda oito categorias de avaliação, a saber (NEVES *et al.*, 2014):

- a) **Concentração** – é a atenção do jogador retido no jogo;
- b) **Desafio** – utilizado para desafiar o jogador. Deve ser levada em conta a dificuldade gradual para não frustrar o jogador;
- c) **Habilidade** – do jogador, são habilidades que devem ser ofertadas e desenvolvidas pelo próprio jogo;
- d) **Controle** – trata-se do controle que o jogador exerce sobre o jogo e o andamento do jogo, como interromper, salvar e retomar o jogo, assim como preferências do jogador e customizações;
- e) **Objetivos claros** – são as instruções do jogo, para que o jogador consiga realizar as tarefas os objetivos devem ser claros e precisos;
- f) **Feedback** – trata-se do andamento do jogo, pois o jogador deve receber informações de sua evolução, conquistas, pontuação, *status*;
- g) **Imersão** – serve para trazer o jogador para dentro do jogo, tornando ele (o jogador) parte do mundo virtual;
- h) **Interação Social** – refere-se à conexão entre os jogadores, como cooperação, bate-papos e competições entre os jogadores.

A avaliação foi realizada por meio da disponibilização do jogo a estudantes das áreas de Computação e Engenharia Civil, os quais possuem em sua grade curricular disciplinas associadas à lógica e programação. Ainda, o jogo foi disponibilizado, também, a estudantes de mestrado na área de Tecnologias Educacionais, envolvendo

perfis da área de Computação e também de Educação, importante para avaliar os aspectos pedagógicos relacionados ao jogo. A seção 6.2.2 detalha o perfil dos usuários que utilizaram o jogo e responderam ao questionário, bem como destaca os resultados obtidos.

Com os aspectos metodológicos definidos, passou-se para as etapas de modelagem e desenvolvimento do jogo proposto, as quais são detalhadas no próximo capítulo.

6. MODELAGEM E DESENVOLVIMENTO DO JOGO SÉRIO *SUPER MARIO LOGIC*

Conforme descrito no capítulo anterior, para o desenvolvimento do Jogo Sério proposto foi utilizada a metodologia INTERAD. As etapas consideradas pertinentes de cada fase foram executadas e são apresentadas neste capítulo. Ainda, a seção 6.1 descreve a história e enredo do jogo.

6.1. SUPER MARIO, UM BLOCKBUSTER⁵

A franquia de jogos Super Mario⁶ é a mais popular e vendida de todos os tempos (DIZON, 2015), com 287.88 milhões desde 13 de Setembro de 1985, sendo que 40.24 milhões destes referem-se ao jogo Super Mario Bros de 1985 (GRINTERS, 2008). Os números dizem respeito aos jogos dos gêneros de Ação/Plataforma da franquia, excluindo os demais gêneros, como corrida, *puzzle*, aventura, entre outros.

O principal personagem é Mario, um encanador, que junto com seu irmão Luigi precisa resgatar a princesa Peach das mãos do Bowser, seu principal inimigo, o qual pretende dominar o Reino dos Cogumelos. Mario passa por diversos desafios e precisa derrotar seus inimigos pulando em cima deles. No jogo, ele aumenta suas habilidades por meio de *powerups* (poderes), tais como um cogumelo que o deixa maior e mais forte, uma flor que faz com que ele atire uma bola de fogo nos inimigos e uma estrela que deixa Mario invencível por um período de tempo.

O jogo proposto nesta dissertação tem como base o jogo Super Mario pela sua popularidade entre os jogadores e fácil reconhecimento do personagem. Suas ações são mediadas por meio de comandos, sendo que cada comando é específico como: andar para frente, pular, abaixar, pular no inimigo, subir obstáculos, entre outros. Tais comandos são executados por meio de utilização das estruturas de programação descritas na seção 2.1, sendo que o estudante (jogador) deverá escolher a(s) mais adequada(s) para cada situação/objetivo.

Com relação ao objetivo do jogo, este é avançar de fase até resgatar a princesa (como na história original). Ao final de cada fase, o usuário irá aprender conceitos básicos ensinados na fase em questão, e a cada fase que vai avançando, novos

⁵ *Blockbuster* - Indica algo popular e de elevado sucesso financeiro.

⁶ Nintendo's Official Home for Mario: <http://mario.nintendo.com/>

desafios serão aplicados de forma gradual com as estruturas sequenciais, estruturas de decisão e, por fim, a estrutura de repetição. Após um resumo sobre o jogo desenvolvido, são apresentadas as etapas da INTERAD.

6.2. COMPREENSÃO

A compreensão, conforme abordado no capítulo 5, diz respeito à primeira etapa do levantamento de dados. O foco é o tema do projeto, público-alvo, objetivos pedagógicos, pesquisa institucional, contexto educacional, necessidades do aluno, expectativa do solicitante e subsídios projetuais.

6.2.1. Tema

O tema do projeto é o desenvolvimento de um Jogo Sérioso para trabalhar, de forma dinâmica e interativa, a Lógica de Programação nos níveis de estrutura sequencial, estrutura de decisão e estrutura de repetição para que o aluno se sinta mais determinado a alcançar os objetivos auxiliando na aprendizagem do desenvolvimento de algoritmos.

6.2.2. Público-alvo

Estudantes de cursos da área de computação ou afins, que tenham a disciplina de lógica de programação ou algoritmos e que podem ter no jogo um auxílio no entendimento de estruturas de algoritmos.

6.2.3. Objetivos Pedagógicos

Os objetivos que devem ser alcançados com o Jogo Sérioso são:

- a) promover o uso de Jogos Sérios no processo de ensino e aprendizagem;
- b) facilitar o ensino e aprendizagem de lógica de programação no nível sequencial, repetição e decisão;
- c) estimular o aluno, de modo que ele perceba que o algoritmo é uma importante ferramenta para resolver problemas;
- d) estimular a aprendizagem de uma forma mais ativa, dinâmica e motivadora;
- e) operar com a noção de algoritmos, aplicando-a, de modo apropriado, às situações e problemas apresentados;
- f) possibilitar uma aprendizagem através da tentativa e erro.

6.2.4. Contexto educacional

O Jogo Sérió é voltado para a área da computação e pode ser utilizado de várias formas: durante as aulas, em sala de aula ou em laboratório, ou fora do âmbito das aulas (em casa, por exemplo). Pode, também, ser utilizado como apoio às aulas presenciais, à distância e sem a presença do professor.

6.2.5. Necessidades do aluno

Uma relevante dificuldade no ensino na área de informática é a lógica de programação. Por meio dela aprende-se a desenvolver algoritmos, que são sequências de passos para chegar a um determinado resultado. O ensino e aprendizagem de lógica e algoritmos desempenha um papel fundamental para o programador, pois, geralmente, são disciplinas ofertadas no primeiro semestre com alunos ingressantes que estão se deparando pela primeira vez com o ensino superior. Muitas vezes, estes alunos estão modificando suas formas de aprendizagem.

É possível que uma série de dificuldades possa surgir até que os alunos se apropriem deste conhecimento, pois, ainda são disciplinas que exigem muito raciocínio lógico e Pensamento Computacional e nem todos os estudantes chegam ao ensino superior com estas habilidades cognitivas. Portanto, o jogo será desenvolvido de forma agradável, divertida e de fácil compreensão para o usuário para que se desfaça essa impressão negativa sobre essas disciplinas que possuem grau elevado de dificuldade.

6.2.6. Subsídios projetuais

Para o desenvolvimento do jogo, foi realizada uma revisão bibliográfica com o tema de lógica de programação, descrita no capítulo 2, destacando autores como Furlan, Forbellone e Eberspäche, Manzano e Oliveira. Para definir conceitos de jogo assim como elementos, gêneros e Jogos Sérios, foram utilizados autores como Salen e Zimmerman, Huizinga, Belli e Raventós, Freitas e Liarokapis e demais autores. Ainda, foram realizadas análises dos trabalhos correlatos como: Castelo dos Enigmas de Scaico; Light-Bot da empresa Armor Games; o trabalho de Borges apresentando o KidCoder; e, por fim, o Jogo Sérió *Program Your Robot* uma versão melhorada de Kazimoglu para procurar um diferencial que contemplasse o jogo proposto, que melhor se adapte ao usuário.

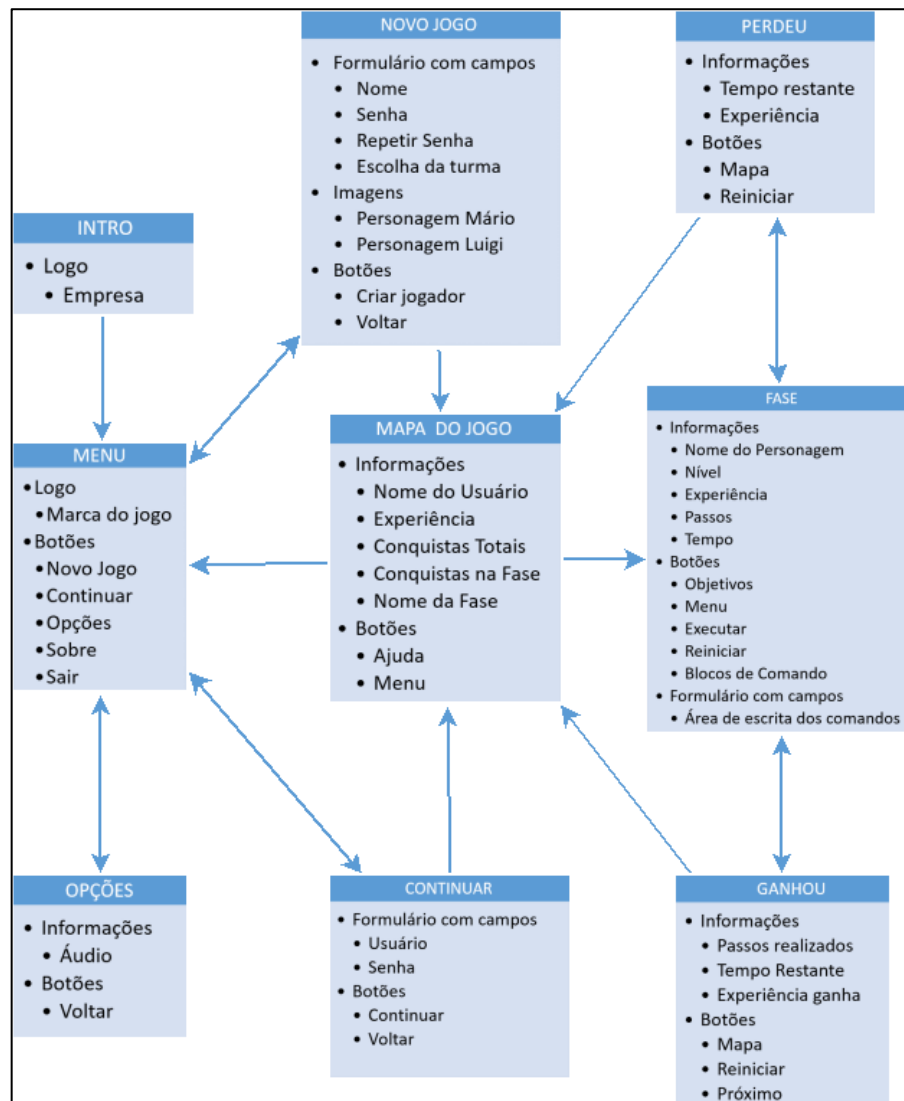
6.3. PREPARAÇÃO

A preparação consiste no levantamento de dados a partir de pesquisas bibliográficas para responder questões como “Por que fazer esse material” e “Como fazer o material”. Esta fase contempla o desenho de conteúdo, funcionalidades, lista de requisitos, recursos e o cronograma.

6.3.1. Desenho de conteúdo

O jogo Super Mario Logic foi desenvolvido para computadores, para as plataformas Windows, Linux e OS X. Na Figura 15, é apresentado o desenho de conteúdo do jogo, já nas Figuras 16, 17 e 18 são apresentados seus *storyboards*.

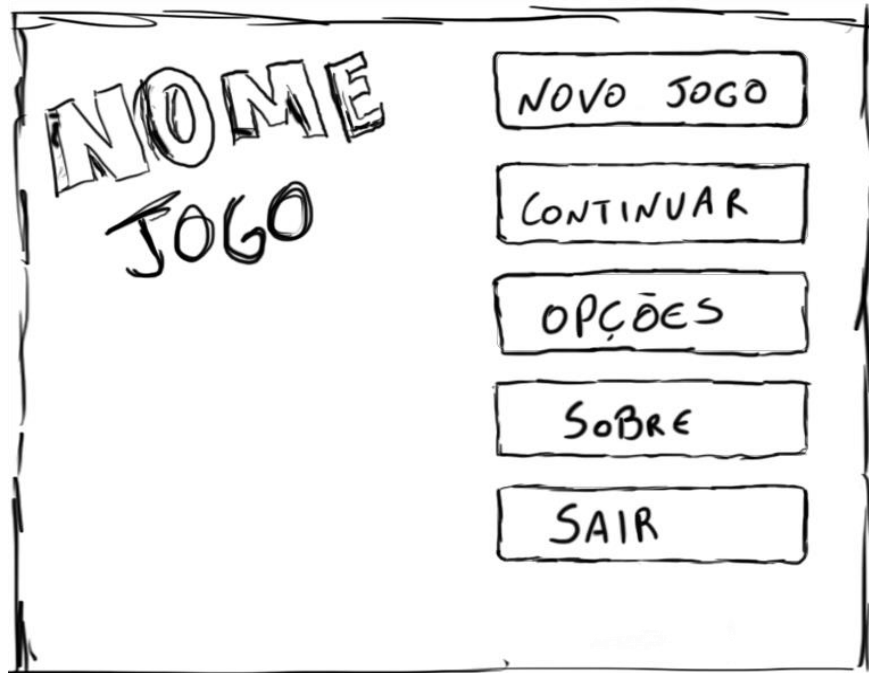
Figura 15 – Desenho de conteúdo do Super Mario Logic.



Fonte: Autor.

Na Figura 16, é exibido o *storyboard* do menu inicial do jogo, sendo que no lado esquerdo da figura tem-se o logo do jogo, ou seja a identidade do mesmo; ao lado direito, tem-se as opções **Novo**, que inicia um novo jogo; **Continuar**, o usuário deve digitar o nome de usuário que ele criou e a senha para continuar de onde parou; **Opções**, são as opções do jogo como, aumentar e diminuir a música de fundo e efeitos sonoros; **Sobre**, dados relacionados às imagens e músicas do jogo; e, por fim, **Sair**, utilizado para sair do jogo.

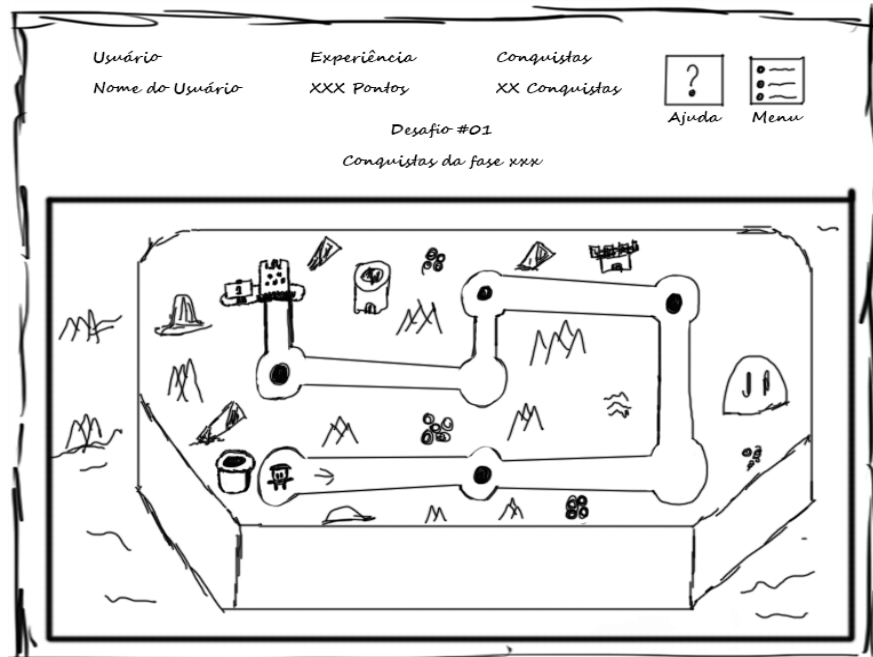
Figura 16 – Storyboard Menu.



Fonte: Autor.

Ao acessar o menu ou **Novo Jogo** o usuário irá para a tela de escolha do personagem e informações. Ao clicar em **Continuar**, o usuário será direcionado ao **Mapa** de desafios (Figura 17), no mapa é possível navegar entre os desafios, sendo que a condição para acessar o próximo desafio se dá somente se vencer o anterior. No *storyboard* do mapa é possível observar informações como: nome do desafio, conquistas adquiridas naquele desafio, nome de usuário, pontuação total no jogo, conquistas gerais, que se dá na soma das conquistas adquiridas em cada fase, o botão de menu e, por fim, o botão de ajuda, que informa ao usuário o funcionamento do jogo.

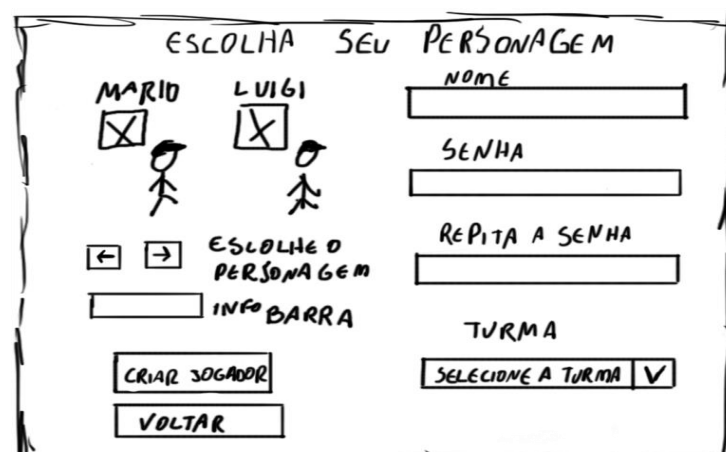
Figura 17 - Storyboard Mapa.



Fonte: Autor.

O próximo *storyboard* denominado **Novo Jogo** (Figura 18), está dividido em duas partes: ao lado direito é possível escolher entre os personagens do jogo; abaixo dos personagens, é possível ver algumas informações de como escolher o personagem; no lado direito, existem três campos: digitar seu nome, digitar uma senha e repetir a senha escolhida anteriormente; e, por fim, o selecionar turma que, por sua vez, tem a função de escolher a turma em que está sendo aplicada a avaliação. Para finalizar, tem-se o botão criar jogador, que cria o usuário no banco de dados e inicia o jogo.

Figura 18 - Storyboard da escolha do personagem.

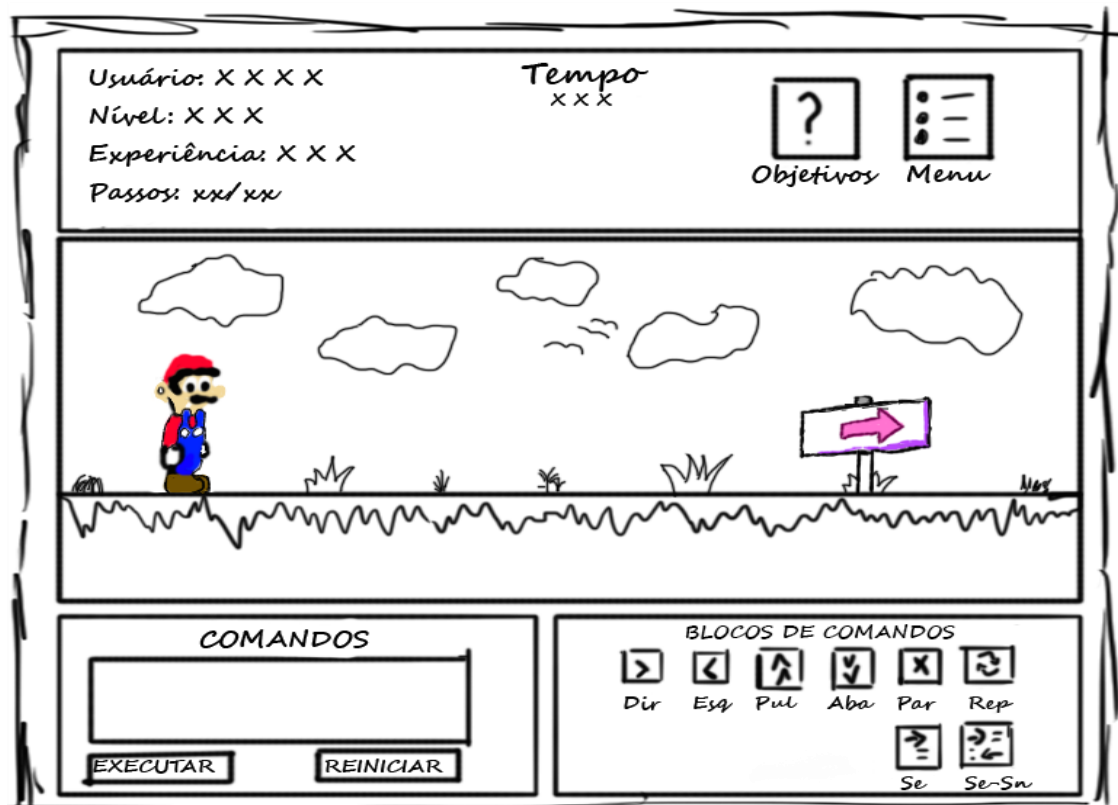


Fonte: Autor.

O *storyboard* **Fase** (Figura 18) está dividido em quatro seções, sendo elas:

- a) **informações do jogo**, como nome do usuário; nível do personagem em que o usuário se encontra; experiência, que é a quantidade de pontos que o jogador adquiriu até o momento; número de passos para alcançar o(s) objetivo(s) da fase; o tempo necessário para concluir o desafio, sendo que quando chegar a metade o contador muda para a cor vermelha para informar ao jogador que já passou metade do tempo; objetivos, como é apresentado o(s) objetivo(s) quando inicia a fase, é interessante disponibilizar essa informação também enquanto o usuário estiver jogando essa fase; menu, para reiniciar a fase, sair e acessar o mapa, ver seu progresso no jogo através das estatísticas; e, por fim, as opções de jogo, como a possibilidade de alteração no volume do som;
- b) **cenário do jogo**, onde se encontram o personagem, objetivos e inimigos;
- c) **área da edição dos comandos**, seção onde são digitados e executados os comandos, para que o personagem realize as ações dos comandos digitados;
- d) **blocos de comando**, que são blocos com informações (ajuda) de como digitá-los na seção de comando, sendo que os mesmos só são liberados conforme a evolução do jogador no jogo.

Figura 19 - Storyboard Fase.



Fonte: Autor.

6.3.2. Funcionalidades

As funcionalidades não se limitam somente a uma fase do jogo, por esse motivo as mesmas foram divididas de forma que contemplem todo o jogo, destacando as mesmas associadas às telas, conforme o desenho de conteúdo. São as seguintes:

- Menu*: as funcionalidades nessa tela são somente de transição para as demais telas, tais como: *novo jogo*, *continuar*, *opções*, *sobre* e *sair*;
- Opções*: aqui se encontram as opções do jogo como *áudio*, função para aumentar ou diminuir o volume da música de fundo e efeitos sonoros do jogo; e a função *salvar*, que salva as configurações do usuário;
- Novo jogo*: aqui existem três funções: *escolha do personagem* (alternando entre Mário e Luigi); a função *criar*, que irá gravar no banco de dados o usuário e senha, além do personagem escolhido junto com as atribuições; e a opção *cancelar*, que retornará ao menu anterior sem salvar nenhuma configuração;
- Continuar*: após o usuário e senha serem adicionados corretamente, é habilitada a opção de *continuar*, que verificará se o usuário e senha estão

- corretos e levará o jogador à fase de escolha junto com todas as configurações pessoais, e a opção *cancelar* que retorna à tela de menu;
- e) *Mapa do jogo*: no mapa do jogo, o usuário poderá escolher qual fase ele deseja jogar (desde que a fase esteja liberada, o que significa que ele a venceu); após a escolha, ele será direcionado à fase escolhida. Além da escolha pela fase, o mapa possui um botão de *informações*, que são informações iniciais de como funciona o jogo, assim como algumas regras que devem ser tomadas para um melhor aproveitamento do jogo. E, por fim, o botão de *menu*, que possui as opções de *estatísticas* relacionadas à evolução do usuário no jogo, *opções* com as mesmas funcionalidades citadas anteriormente: *retornar* que faz com que o usuário saia do menu e volte ao mapa, e *sair* que retorna a tela de menu do jogo;
 - f) *Estatísticas*: essa tela só pode ser acessada por meio do botão de menu que se encontra tanto no mapa quanto na tela de fases. Sua funcionalidade é mostrar ao usuário o progresso do jogo trazendo como informações o nome de usuário, experiência total, o nível em que se encontra, o tempo total de jogo e as conquistas por mundo e total;
 - g) *Fase*: na tela principal do jogo existem três opções, sendo elas: *menu*, que abre um menu contendo as opções *retornar*, *reiniciar*, *estatísticas*, *opções* e *sair*; *objetivos*, que mostra ao usuário os objetivos e informações da fase em questão e, por fim, o *bloco de comandos* que trata de uma caixa de texto onde são digitados os códigos para as ações do personagem na fase.

6.3.3. Lista de requisitos

A lista de requisitos consiste em mostrar um escopo dos requisitos do projeto como apresentado na Tabela 04. A lista deve ser direta, objetiva e em forma textual contemplando os requisitos de qualidade do usuário (não funcionais), identidade visual (PASSOS, 2011).

Tabela 4 - Lista de requisitos.

PRINCIPAIS REQUISITOS DO USUÁRIO

Eficiência

Tem como objetivo fazer com que o usuário realize poucas ações para chegar onde deseja;

Usabilidade

Deve possuir um visual limpo e agradável, com cores apropriadas para o jogo e nível gradual de dificuldade para uma melhor experiência com o jogo indo do nível fácil até o difícil;

Segurança

Permite que o usuário salve seu personagem assim como prêmios e conquistas ganhas, para que a cada início de jogo não tenha que começar do zero;

Elementos gráficos priorizados

Menu, ícones e área de navegação. Esses elementos são indispensáveis para uma boa navegação no jogo;

Identidade Visual

Limpeza, simplicidade, objetividade, consistência visual, interação, clareza, conhecimento, lógica, programação.

Fonte: Autor.

6.4. EXPERIMENTAÇÃO

A etapa da experimentação tem como objetivo desenvolver o modelo conceitual para a interface, ou seja, o rascunho das telas e objetos que vão constar no Jogo Sério.

6.4.1. Modelo Conceitual

O modelo conceitual consiste em trabalhar os signos, os elementos que irão compor as interfaces, em termos de botões, ícones, etc. A tabela 5 mostra os elementos do jogo e o objetivo de cada um.

Tabela 5 - Elementos do jogo Super Mario Logic.

ELEMENTO	OBJETIVO
	Inicia um novo jogo.
	Continua o jogo salvo.
	Utilizado para alterar as preferências do usuário como o som.
	Informações pertinentes ao jogo como direitos autorais e objetivo do jogo.
	Botão utilizado para sair do jogo.
	Responsável por mostrar o tempo regressivo de cada fase.
	Trata-se do nível do personagem.
	Nome criado pelo jogador.
	Quantidade de pontos obtidos pelo jogador.
	Quantidade de passos que o usuário pode executar.
	Blocos de comandos, contendo as informações para sua utilização.
	Botão responsável por executar os comandos digitados.
	Botão responsável por reiniciar os comandos da fase.

Tabela 5 - Elementos do jogo Super Mario Logic.

(Continuação)
















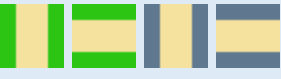

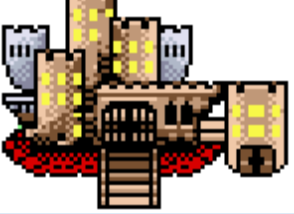
ELEMENTO	OBJETIVO
	Botão responsável para retornar ao mapa.
	Botão responsável por reiniciar a fase desde o início.
	Após o fim de uma fase, ao clicar nesse botão, será enviado para próxima fase sem precisar acessar o mapa.
	Signo do Mapa, responsável por identificar o usuário.
	Signo do Mapa, responsável por exibir a experiência total do usuário.
	Signo do Mapa, responsável por exibir a quantidade total de conquistas (estrelas adquiridas por fase) do usuário.
	Fase aberta para jogar.
	Fase fechada. Só abre no momento que a anterior é ganha.
	Placas que indicam o usuário em que continente ele se encontra.
	Cano de tele transporte. Transporta o jogador para outro continente.
	Desafio final de cada continente.
	<i>Moedas</i> , ajudam na pontuação do usuário e também são necessárias para alguns objetivos.
	<i>Switch P</i> , utilizado para abrir caminhos bloqueados.

Tabela 5 - Elementos do jogo Super Mario Logic.

ELEMENTO	OBJETIVO (Conclusão)
	<p><i>Placa de continuação</i>, ao chegar até a placa o jogo é finalizado.</p>
	<p><i>Placa de Cuidado</i>, diz ao usuário que um inimigo irá surgir.</p>
	<p>Caminhos no mapa em que o personagem pode prosseguir.</p>
	<p>Caminhos no mapa fechados, o personagem não pode prosseguir.</p>
	<p>Último desafio do jogo.</p>

Fonte: Autor.

6.4.2. Arquitetura

Como se trata de um jogo, a estrutura de fases remete, em sua maior parte, ao fluxo da informação no formato linear, pois o próximo desafio somente poderá ser acessado se o desafio atual for completado. No entanto, após concluir uma fase, o jogador pode acessá-la novamente na tela de mapa, a qual irá conter todas as fases desbloqueadas após completar os desafios e, neste caso, a arquitetura passa a ser a hierárquica ou estrutura em árvore.

6.5. ELABORAÇÃO

Na etapa da elaboração, devem ser escolhidos quais são os tipos de interatividade, o desenvolvimento da malha construtiva e estrutural e, por fim, o desenho de navegação.

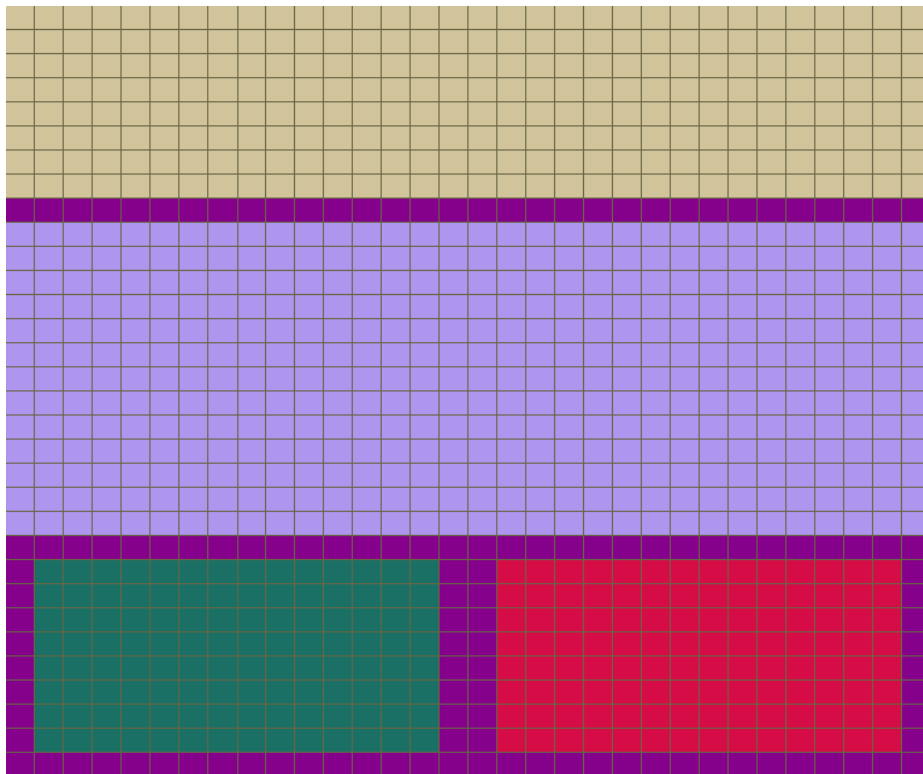
6.5.1. Tipo de interatividade

Os tipos de interatividade que o jogo proporciona são: objetiva, pois os alunos utilizam o *mouse* para acessar cada fase, instruções e menu; contextual não-imersiva, este conceito se estende aos vários níveis interativos em um completo ambiente de treinamento virtual capaz de explorar o contexto significativo relacionado com o conteúdo; e de suporte, pois para que o usuário consiga escrever os comandos, existem vários botões com dicas de como aplicá-la ao jogo.

6.5.2. Malha construtiva

Por se tratar de um jogo, a malha construtiva para o Super Mario Logic não possui um padrão em todas as telas. Assim, a Figura 20 apresenta a malha predominante do jogo dividida em quatro seções, sendo elas: *superior*, contendo informações gerais do jogo; *meio*, que contém o cenário principal de execução do jogo; *inferior esquerdo*, que apresenta a seção dos comandos; e *inferior direito*, onde são apresentados os blocos com informações de cada comando.

Figura 20 – Malha construtiva da fase.

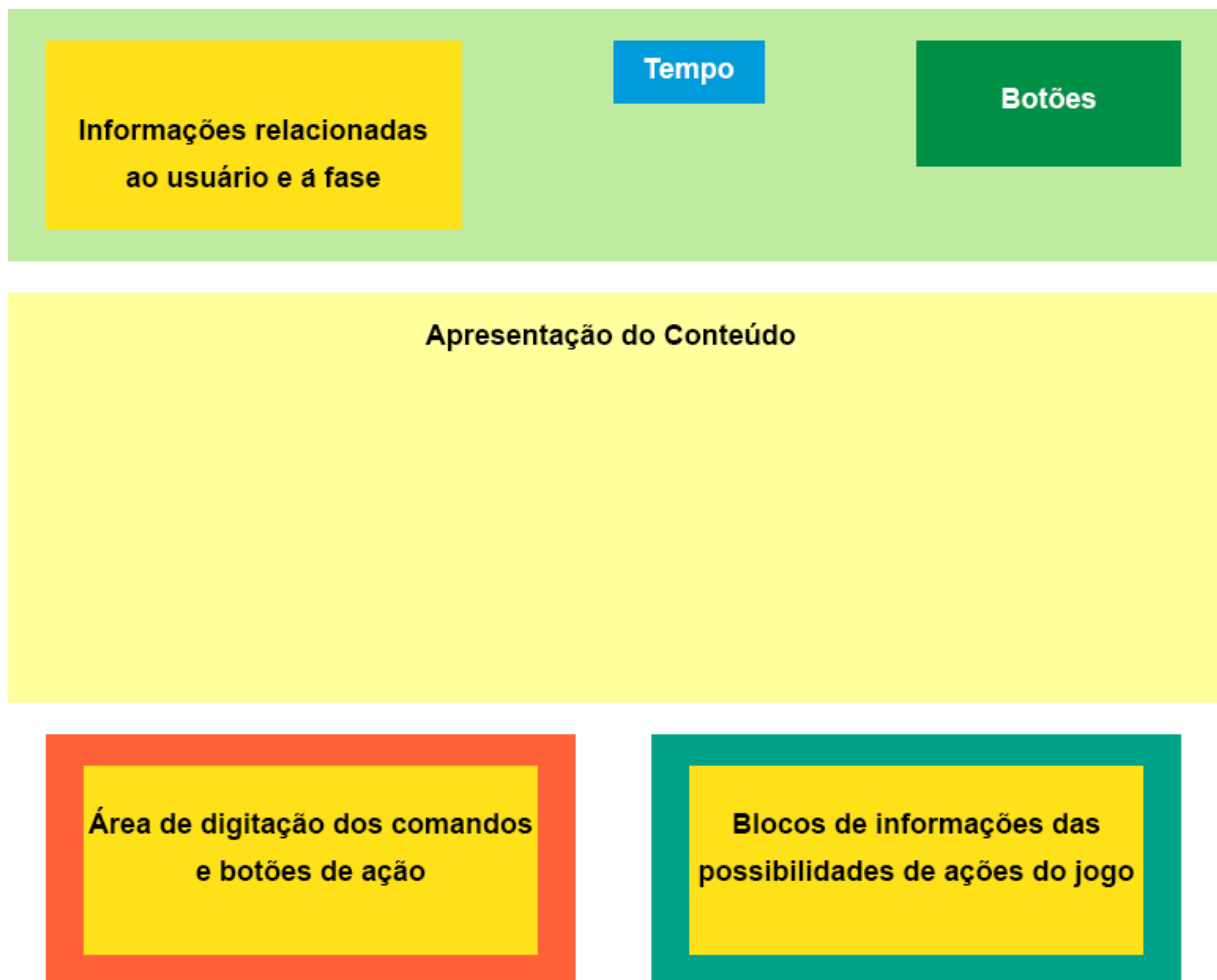


Fonte: Autor

6.5.3. Malha Estrutural

Na Figura 22, é apresentada a malha estrutural mostrando a disposição dos elementos no leiaute da fase. A seção superior é dividida entre as informações do usuário e também da fase, o tempo regressivo e a área de botões objetivos e menu; na seção central apresenta-se o conteúdo que trata dos objetos do jogo como personagem, inimigos e cenário; no canto inferior direito tem-se a área de digitação das ações que o personagem irá realizar e, por fim, no canto inferior direito se encontram os blocos de informações das ações do jogo.

Figura 21 – Malha estrutural da fase.



Fonte: Autor;

6.5.4. Desenho de navegação

A navegação do Super Mario Logic é baseada em botões, sendo eles com ícones e sem ícones. Os botões são autoexplicativos, ou seja, já induz o usuário a clicar e realizar a ação que ele espera. Por exemplo, quando o usuário está no mapa, existe um botão com um símbolo de interrogação remetendo ao usuário que clicando naquela área ele irá ver informações relacionadas com o jogo.

6.6. APRESENTAÇÃO

Esta é a última etapa da metodologia INTERAD. Nesta etapa, é apresentado o projeto final contendo a identidade visual, isto é, a marca do jogo e também a interface já com seus elementos. Esta etapa é importante, pois mostra todo o estudo desenvolvido até aqui e apresenta a estética do jogo.

6.6.1. Identidade de marca

A logomarca do jogo é apresentada na Figura 23 e segue as características do conceito de leiaute (designer visual), bem como em sua logotipia. As cores vermelho, azul, amarelo e verde estão associadas às cores das letras do jogo Super Mario World. A cor vermelha está associada à emoção do usuário e, juntamente com a cor azul, desperta a paz e produtividade. A cor verde remete a harmonia no geral e a cor amarela remete a uma cor enérgica, estimulante e de criatividade, sendo que seu uso dilucida a mente conforme Freitas (2007).

Além da escolha das cores, a logomarca possui as frases MOSTRAR.LOGO(“SUPERMARIOLOGIC”); e FINALIZAR (); que remete ao usuário que a identidade está diretamente ligada à Lógica de Programação.

Figura 22 – Logomarca (identidade visual da marca).



Fonte: Autor;

6.6.2. Design visual

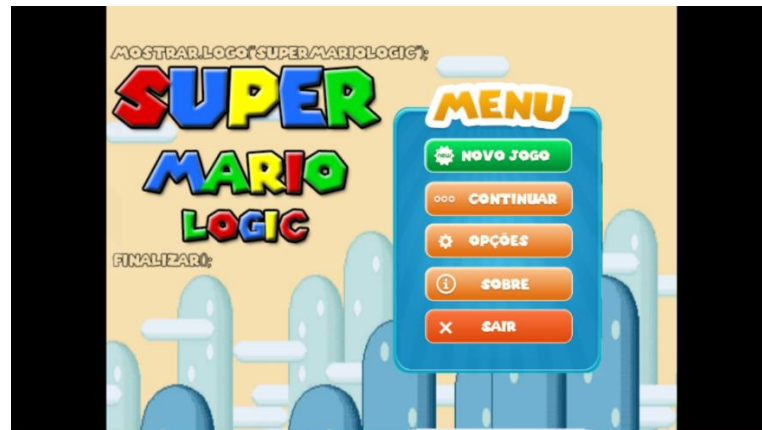
Todos os componentes foram planejados seguindo uma harmonia no *design*, sendo especificado por Garrett (2002), conforme abaixo:

- a) foram utilizados elementos equilibrados, dando destaque a ícones de forma minimalistas, desenhados como forma de manter um padrão entre a interface e personagens de interação;
- b) com relação ao equilíbrio, o contraste e uniformidade da composição, foi respeitada a profundidade dos objetos, aperfeiçoando a sensação de usabilidade, com os elementos bem claros e definidos no escopo do objeto;
- c) as cores e a tipografia deram prioridade a cores suaves, fazendo uma clara e bem-sucedida união entre o azul, verde, amarelo, laranja e branco. As cores de objetos mais fortes também foram suavizadas para dar uma melhor composição, unido com harmonia aos tons de azul. Na tipografia foram utilizadas fontes “sem serifa”, de leitura clara e leve, unindo perfeitamente o conjunto de elementos às cores;
- d) os elementos foram alinhados e dispostos de forma que o usuário se localize de forma rápida e sem dispersão.

6.7. APRESENTAÇÃO DO JOGO

O jogo foi desenvolvido de forma intuitiva, ou seja, de fácil compreensão para o usuário. Ao iniciar o jogo, o estudante visualiza um menu contendo as seguintes opções: *novo jogo*, *continuar*, *opções*, *sobre* e *sair* conforme apresentado na Figura 23.

Figura 23 – Tela inicial do jogo.



Fonte: Autor;

Se o jogador já possui uma conta, deve acessar a opção continuar para dar segmento ao jogo em andamento; caso contrário, vai acessar a opção *novo jogo* para escolher um personagem e criar uma conta adicionando dados como: *nome de usuário*, *senha* e a *escolha da turma* (a escolha da turma só é possível se ela já foi criada previamente pelo professor ou administrador do jogo). Após ter adicionado os dados e escolhido o personagem, o usuário deverá clicar em *criar jogador* (Figura 24).

Figura 24 – Tela de criação do personagem.



Fonte: Autor;

Após criado o personagem, o usuário será redirecionado para o mapa do jogo (Figura 25). No topo do mapa, são exibidas informações como o *nome do usuário*, *experiência* e *conquistas*. Ao lado das conquistas estão localizados os botões de *ajuda* e *menu*, onde a *ajuda* serve para mostrar as informações da mecânica do jogo, e o botão *menu* irá exibir opções para o usuário como *retornar* ao mapa, *estatísticas* do usuário, *opções* e *sair*. Para escolher um desafio (fase), o mesmo deve estar desbloqueado, caso não esteja, o usuário precisa desbloqueá-lo completando os objetivos do desafio anterior.

Figura 25 – Mapa inicial do jogo.



Fonte: Autor;

Ao acessar um desafio, inicialmente será exibida uma tela com as informações sobre o mesmo, tais como: *o que deve ser feito*, *o tempo máximo para cumprir o objetivo* e *a quantidade de passos para chegar até o destino*. Como explicado na seção 6.3.1, na parte superior se encontra o nome do usuário, o nível em que está no momento, a quantidade de experiência adquirida até o momento, e a quantidade de passos necessários para completar o objetivo. Observando o exemplo apresentado na Figura 26, é possível perceber que esta fase deve ser completada com 5 passos e que, por ser início de fase, nenhum passo foi digitado até o momento (Passos 0/5). O número de passos não é limitado na execução, porém, se o usuário cumprir o objetivo utilizando mais passos do que o necessário, o jogador irá receber um decréscimo na pontuação final.

Figura 26 – Tela do desafio.



Fonte: Autor;

O jogador pode atingir todos os objetivos definidos pelas regras sendo que cada objetivo vale uma conquista (estrela), bem como, atingi-los parcialmente desde que vença o desafio (quando o personagem chega a placa de continuação) antes de terminar o tempo. Quando o jogador termina o desafio, uma tela de parabéns é exibida (Figura 27) contendo a *experiência* adquirida, a *quantidade de conquistas* (estrelas) que podem variar de acordo com os objetivos completados e mostrar o tempo em que foi concluído o desafio. Também são apresentados nesta tela, três botões: *mapa* que ao ser acessado redireciona o jogador ao mapa do jogo; *reiniciar*, que reinicia o desafio (fase) caso o jogador queira ganhar mais pontos e/ou conquistas; e, por fim, *próximo*, que ao ser clicado conduz o usuário para o próximo desafio, sem a necessidade de ir ao mapa e escolher o desafio.

Figura 27 – Tela de parabéns.



Fonte: Autor;

O jogador pode fracassar em sua tentativa de passar de fase mediante algumas situações: (a) se não conseguir atingir o objetivo, ou seja, não informar os passos adequados para que o personagem chegue ao destino; (b) se o tempo estipulado para execução da fase terminar; ou, (c) utilizando a terminologia de jogos, caso o personagem “morra” em determinado ponto da execução ao se deparar com algumas situações, tais como cair em buracos, colidir com uma bomba ou inimigo ou cair em um rio. Nestes casos, é necessário jogar o desafio novamente para poder avançar no jogo, situação que pode ser visualizada na Figura 28, que apresenta a tela de *tente novamente*. O usuário deve escolher o botão *reiniciar* para executar a fase novamente ou pode retornar ao mapa do jogo, por meio do botão *mapa*.

Figura 28 – Tela de tente novamente.



Fonte: Autor;

O nível de complexidade vai aumentando conforme o usuário vai avançando nos desafios. Ao total são 13 desafios, cinco com dificuldade fácil, cinco na dificuldade média e três desafios difíceis. Nos desafios de dificuldade fácil, é possível utilizar os comandos de *mover para direita*, *mover para esquerda*, *pular*, *abaixar* e *parar*. Na dificuldade média, o comando *repita* é adicionado sendo possível realizar várias ações com um comando somente; e, por fim, na dificuldade difícil, os comandos de decisão *se* e *se senão* são acrescentados. Além do acréscimo de comandos ao avançar no jogo, a complexidade também aumenta fazendo com que o jogador pense em estratégias de como alcançar os objetivos do jogo no menor tempo possível trabalhando, assim, a agilidade, percepção e raciocínio lógico. Para ter uma noção do grau de complexidade de um nível avançado, é apresentada na Figura 29 a última fase, onde

o objetivo é resgatar a princesa Peach. Para que isto aconteça, 16 passos são necessários em 500 segundos, além da disponibilidade de todos os comandos para concluir o objetivo.

Figura 29 – Última fase do jogo Super Mário Logic com grau de dificuldade difícil.



Fonte: Autor;

Após a implementação do jogo Super Mario Logic, passou-se à etapa de aplicação e avaliação dos resultados, que são discutidas no próximo capítulo.

7. AVALIAÇÃO E DISCUSSÃO DOS RESULTADOS

Este capítulo tem como objetivo apresentar os resultados obtidos por meio da aplicação e avaliação do jogo Super Mário Logic com grupos de estudantes que possuem em sua grade curricular, disciplinas associadas a Lógica de Programação. Para isto, a avaliação foi dividida em dois momentos, a disponibilização do jogo Super Mario Logic para os alunos testarem e, após, a aplicação do questionário. Considerando esta estratégia, é importante ressaltar que este estudo foi avaliado apenas sob o ponto de vista do estudante, pois os mesmos responderam ao questionário fornecendo explicitamente suas impressões sobre o jogo.

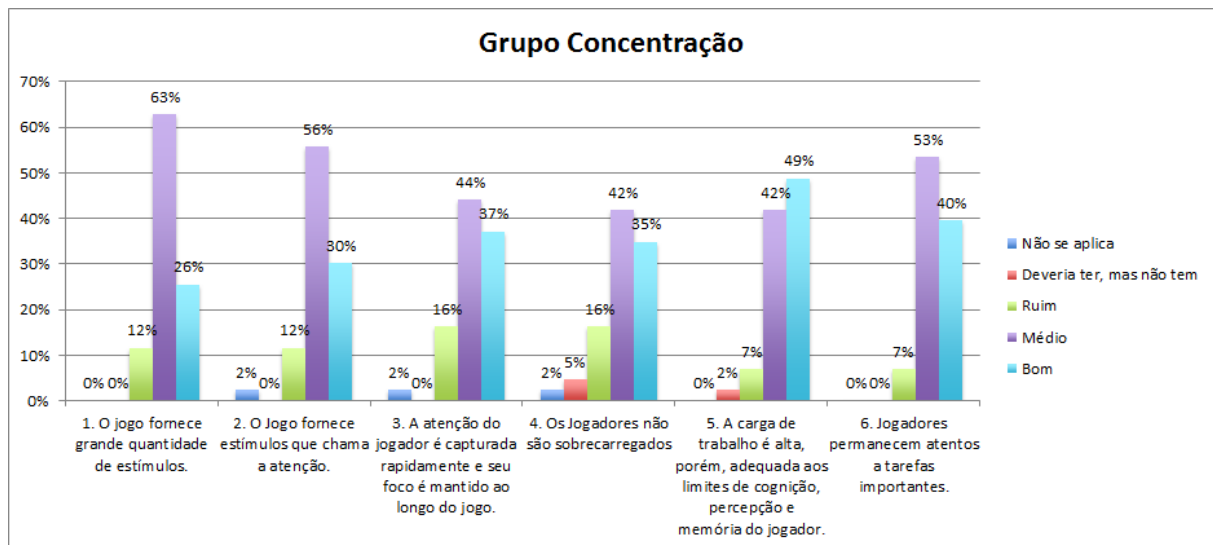
No total, quarenta e três usuários testaram o jogo e responderam o questionário, sendo estes divididos em quatro turmas, assim identificadas: (1) treze alunos de Engenharia Civil da Universidade Federal de Santa Maria (UFSM); (2) vinte e seis alunos do curso de Ciência da Computação da Universidade Federal do Pampa (UNIPAMPA), campus Bagé, cursando a disciplina de Algoritmos e Programação; e (4) quatro alunos do Grupo de Redes e Computação Aplicada (GRECA) da UFSM, considerando que o autor da dissertação e a sua orientadora fazem parte do mesmo.

Conforme descrito no capítulo 5, o questionário utilizado para a realização da avaliação, foi o *GameFlow*, (NEVES *et al.*, 2014) adaptado para a realidade deste estudo. Neste caso, a categoria de interação social foi retirada, pois no jogo Super Mario Logic não há interação *multiplayer*⁷ entre os usuários.

A avaliação do *GameFlow*, é dividida em 5 alternativas sendo: 0 – Não se aplica; 1 - Deveria ter, mas não tem; 2 – Ruim; 3 – Médio e 4 – Bom;

⁷ Multiplayer ou multijogador, trata-se de jogos online onde há interação com mais de um usuário no mesmo jogo ao mesmo tempo.

Gráfico 1 - Levantamento de dados – Grupo Concentração.



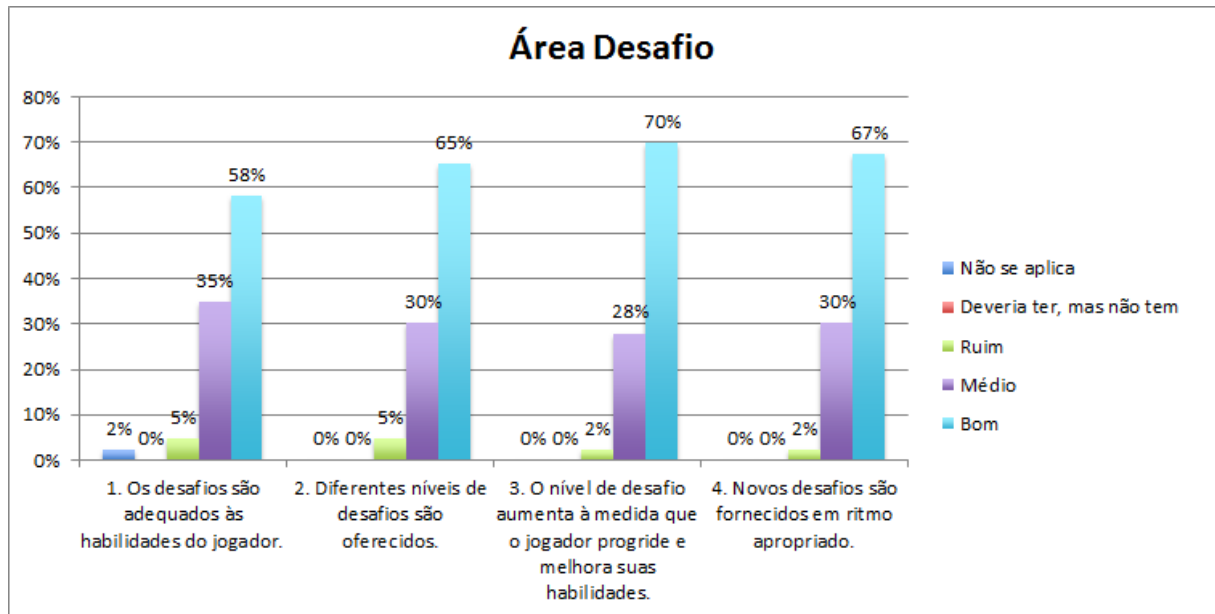
Fonte: Autor;

O critério de *Concentração*, de modo geral, obteve um índice médio (Gráfico 1). O ponto positivo mais alto foi o item 5 que diz “a carga de trabalho alta, porém, adequada aos limites de cognição, percepção e memória do jogador”, alcançando 91% de aprovação (médio e bom). Este critério vem ao encontro de um elemento de Huizinga (2007), a *Liberdade* do jogador, ou seja, encontrar maneiras de reter a atenção do usuário com o que interessa para ele, para que consiga jogar de forma adequada e concentrado no que está fazendo. Assim, a carga de trabalho não deve ser exagerada e também fraca, achando um meio termo para o mesmo.

A avaliação mais baixa no critério *Concentração* foi evidenciada no item 4 que questiona se “os jogadores não são sobrecarregados”, atingindo 16% de avaliações negativas (ruim). Considerando-se o ponto de vista dos estudantes, o que pode ter contribuído para esta percepção, é o número de passos a ser realizado em algumas fases, dando-se uma sobrecarga de até oito passos de um desafio para o outro.

O jogo possui três níveis de dificuldades, fácil, médio e difícil, sendo a fácil para o mundo 1, desafios médios para o mundo 2 e, por fim, difícil para o mundo 3. Os desafios foram pensados de forma que a dificuldade vai aumentando assim que o jogador vai avançando no jogo.

Gráfico 2 - Levantamento de dados - Grupo Desafios.

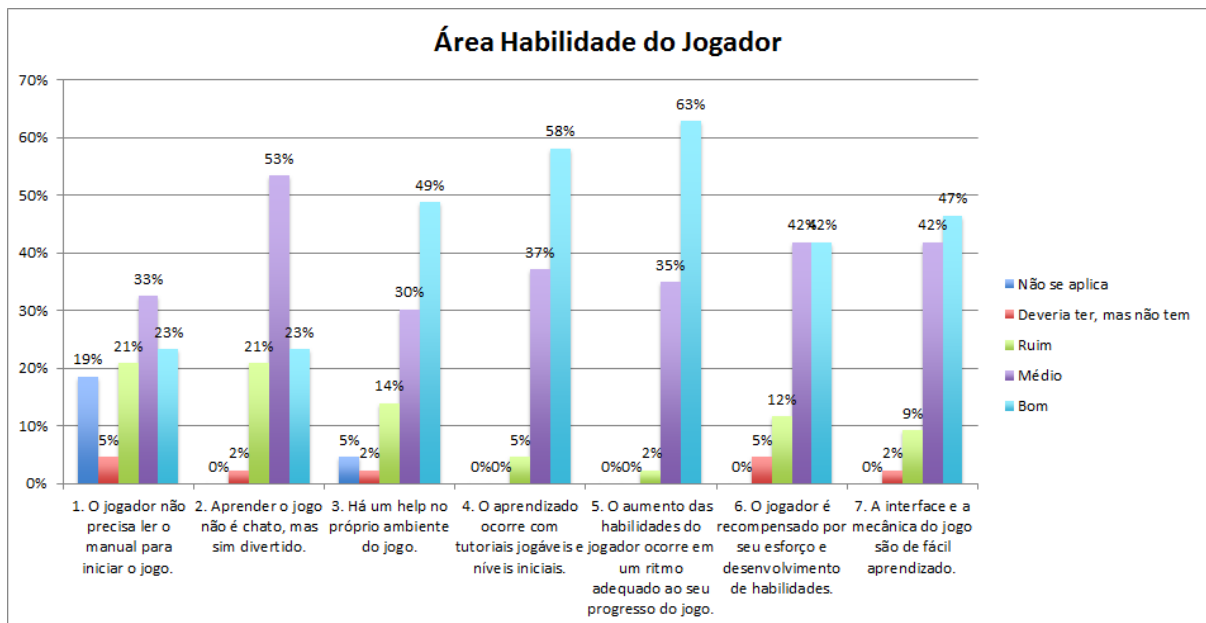


Fonte: Autor;

Pode ser observado no Gráfico 2, referente ao critério de *Desafio*, que houve uma boa avaliação, mostrando que os desafios aplicados no jogo Super Mario Logic foram bem elaborados e adequados às habilidades dos jogadores.

Observa-se que dois itens desde critério foram bem avaliados como no caso em que quando o jogador vai avançando nos desafios, o nível de dificuldade também vai aumentando, fornecendo, assim, uma jogabilidade com o nível de dificuldade apropriado para cada jogador.

Gráfico 3 - Levantamento de dados - Grupo Habilidade do Jogador.



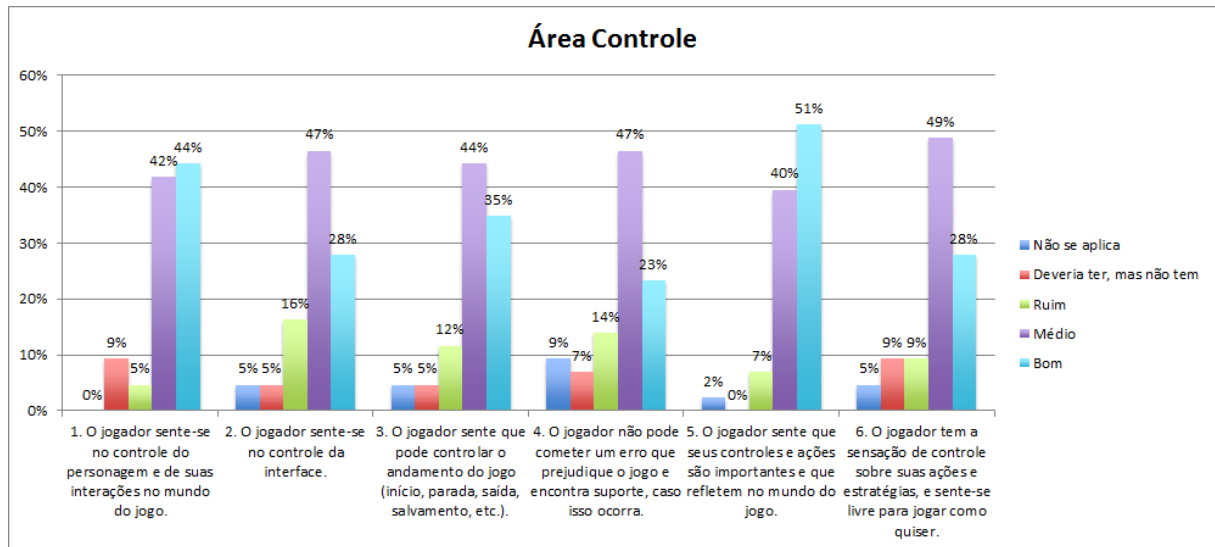
Fonte: Autor;

Na avaliação do grupo de *Habilidades do Jogador*, pode ser visualizado no Gráfico 3 que o item que não obteve um desempenho muito bom foi o item 1 – Os desafios são adequados às habilidades do jogador. Isto demonstra que o jogo, mesmo sendo intuitivo, não elimina a necessidade do usuário ler as instruções de como jogar. A cada início de desafio, é mostrado o objetivo e as regras a serem cumpridas, sendo que, caso o usuário não se lembre dos objetivos iniciais, há um botão com tais informações fazendo com que ele não necessite reiniciar o desafio para ver quais os objetivos que ele deve alcançar. Isto vem ao encontro do item 3, onde 21 alunos confirmaram a existência de um *help* no ambiente do jogo.

A pontuação mais alta destaca-se no item 5, onde, sob ponto de vista dos estudantes, o aumento das habilidades do jogador ocorre em um ritmo adequado ao seu progresso no jogo (63% de avaliações positivas). Isto é, o jogo não muda de nível drasticamente do fácil para o difícil e sim de forma gradual para que o usuário progrida conforme as experiências adquiridas nos desafios anteriores.

Ter o poder de realizar ações a partir de tentativas e erros traz ao estudante a sensação de controle que segundo Neves (2014), é um aspecto importante para levar o jogador a experimentar o estado de controle sobre o(s) elemento(s) do jogo.

Gráfico 4 - Levantamento de dados - Grupo Controle.



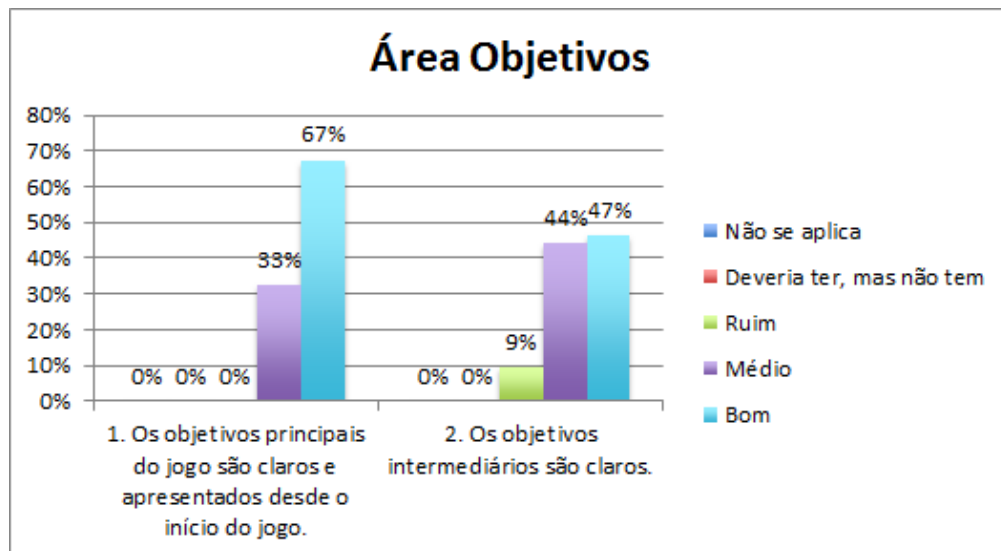
Fonte: Autor;

Essa colocação, reflete-se no item 5 do Gráfico 4, em que 22 alunos que responderam bom e 17 médio, ou seja, 91% dos alunos sentem que seus controles e ações são importantes e que refletem no mundo do jogo. Isso vem ao encontro do elemento de Liberdade de Huizinga (2007) e também da análise que Arruda (2014) discute em seu livro sobre o jogador tomar suas próprias decisões para alcançar seus objetivos.

Outra questão importante que Neves (2014) aborda é em relação à customização no jogo que no caso do Super Mario Logic, destaca-se na escolha do personagem, nome de usuário, senha, escolha das fases, visualização do progresso do jogo, entre outros. Ainda, o fato de poder salvar o progresso do jogo permite ao usuário poder continuar de onde parou a qualquer momento e em qualquer lugar em que o jogo esteja instalado, reforçando o requisito de controle.

Quando se trata de objetivos, os mesmos devem ser claros para que o jogador entenda o que deve ser realizado para concluir os desafios. No jogo do Super Mario Logic isso fica evidente no resultado da pesquisa no grupo de *Objetivos* (Gráfico 5).

Gráfico 5 - Levantamento de dados - Grupo Objetivos.



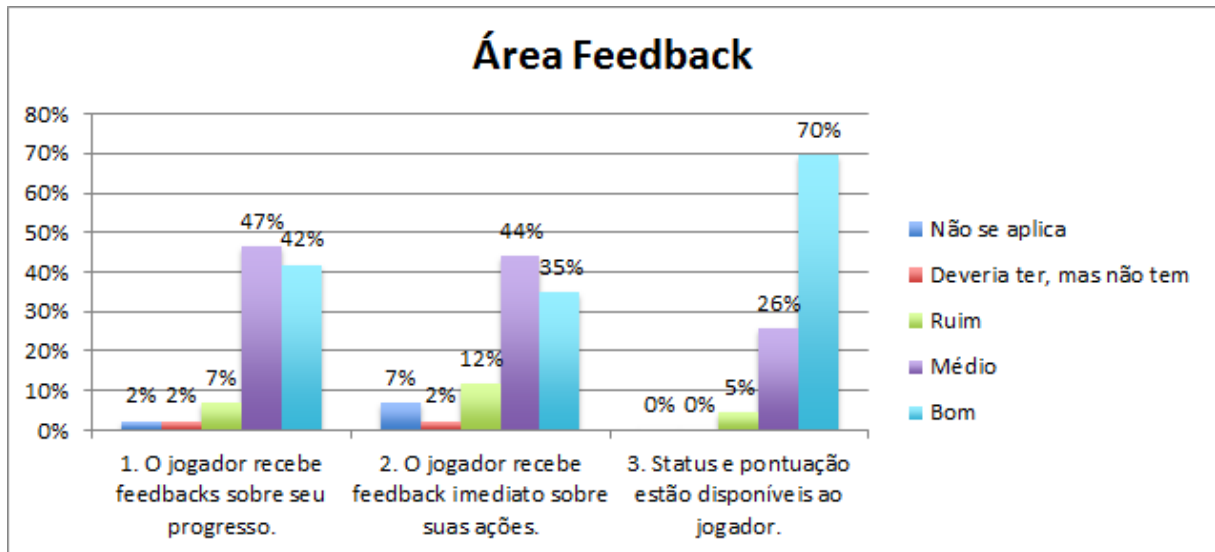
Fonte: Autor;

Segundo os estudantes questionados, 100% (bom e médio) acreditam que os objetivos principais do jogo são claros e 91% (bom e médio) acreditam que os objetivos intermediários também se apresentam de forma clara.

No jogo se encontram dois tipos de objetivos: objetivo principal, em que o personagem Mario ou Luigi, comandado pelo usuário, deve resgatar a princesa Peach; e, também, os objetivos de desafios, que são objetivos intermediários que devem ser alcançados para vencer um desafio e passar para a próxima fase.

Isso mostra que este critério estabelece uma ligação direta com o que é exposto pelos autores Salen e Zimmerman (2004) e também Huizinga (2007), em que objetivos claros e regras bem definidas fazem com que o jogador consiga se orientar dentro do jogo para realizar o que é solicitado obtendo assim, uma satisfação pessoal por vencer o desafio aplicando seu esforço, habilidades e conhecimentos adquiridos através dos diversos níveis de dificuldades enfrentados em cada fase.

Gráfico 6 - Levantamento de dados - Grupo Feedback.



Fonte: Autor;

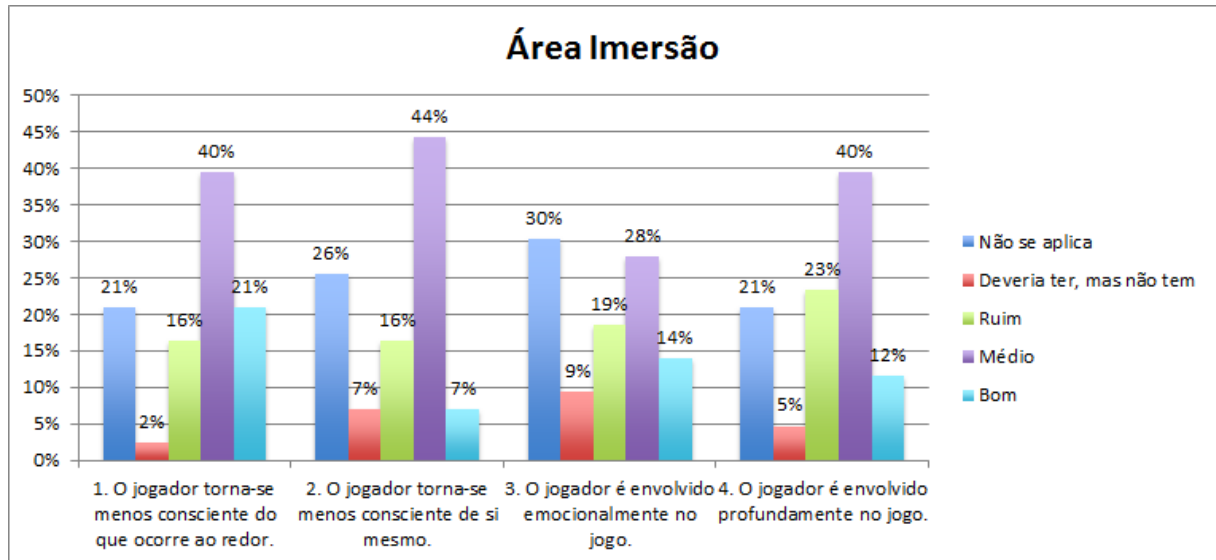
Com relação ao *Feedback* do jogo, este se dá em dois momentos: em tempo real, onde é informado o nome do usuário, conquistas, nível, número de passos dados e em qual linha está o erro de digitação de um comando, ou seja, são *feedbacks* em tempo de jogo, que ficam permanentemente sendo exibidos na interface. A outra forma de *feedback* do jogo é chamada de estatísticas, onde é informada a experiência total adquirida, o nível em que o jogador se encontra, tempo total de jogo e a porcentagem de conquista por mundo e geral. Este tipo de *feedback* é exibido ao jogador caso ele acesse a opção de Estatísticas, presente na interface, não ficando permanentemente exposto.

Aplicado o questionário aos alunos, no que se refere ao *feedback* no jogo obteve-se uma grande aprovação pelos alunos no item 3 do Gráfico 6 – *status e pontuação estão disponíveis ao jogador*, com 95% de respostas do tipo Bom e Médio. Este resultado pode ser decorrente da disponibilização aos dados em tempo real.

Finalmente, o gráfico 7 apresenta os resultados obtidos com relação ao grupo *Imersão*. Apesar da imersão estar mais comumente associada à interações em três dimensões (NEVES et al., 2014), a intenção foi avaliar em qual grau o estudante, no papel de jogador, se considerou envolvido emocionalmente com o mesmo. De acordo com Mendonça e Mustaro (2011), “a imersão permite que o estudante se engaje no universo do *serious game*, tornando-o crível, o que facilita o fluxo de informações e

auxilia na compreensão dos objetivos, bem como o caminho a ser percorrido para atingi-los”. Assim, pretendia se explorar estas possibilidades.

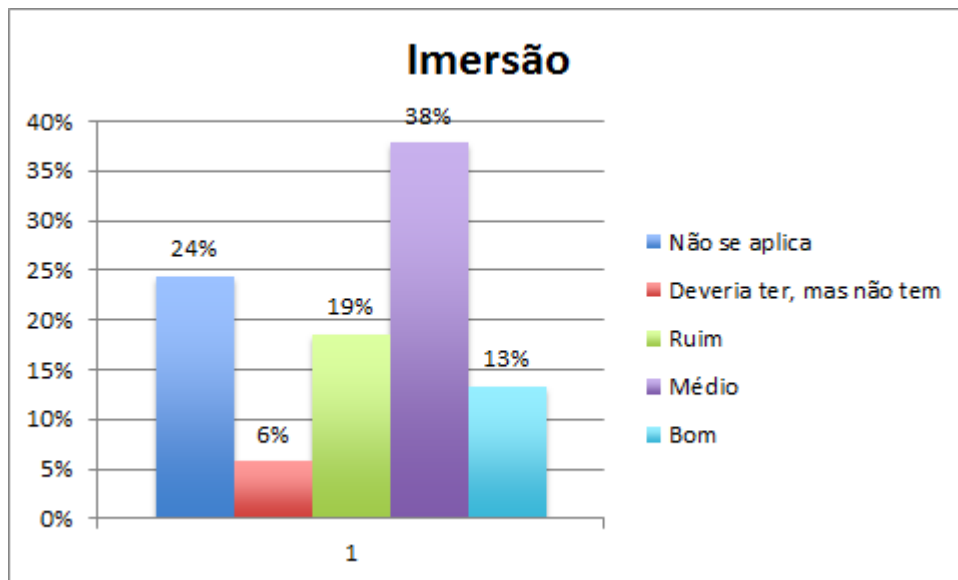
Gráfico 7 - Levantamento de dados - Grupo Imersão.



Fonte: Autor;

Ao observar os resultados do grupo *Imersão* de forma geral (Gráfico 8), destaca-se que 51% consideraram a imersão média (38%) ou boa (13%). Considerando os demais resultados, este pode ser considerado o avaliado de forma menos positiva, sendo 19% de respostas na faixa *ruim* e 6% na faixa *Deveria ter, mas não tem*. No entanto, 24% dos estudantes questionados informaram que as questões de imersão *não se aplicavam* ao jogo avaliado. Observa-se pelas respostas que, talvez, os estudantes não tenham compreendido de forma adequada o que se desejava com as questões relacionadas com imersão. Como não houve um momento posterior, onde os alunos pudessem ser indagados sobre suas respostas, não é possível inferir e analisar de forma adequada os resultados obtidos.

Gráfico 8 - Levantamento de dados - Grupo Imersão vista de forma geral.



Fonte: Autor;

Para finalizar o questionário, de forma descritiva, o estudante era convidado a deixar sua opinião e sugestão para melhorias do jogo avaliado. Destas, foram retiradas as mais interessantes para a contribuição deste trabalho conforme abaixo.

- a) Aluno 1: *“Certa vez vi um jogo para introdução à programação onde o usuário “arrastava” os comandos para uma linha de programação e depois executava as atividades. Creio que uma abordagem neste estilo seria interessante. O jogo ainda está um pouco “travado”, mas creio que isso será solucionado nas próximas versões. Num geral, gostei muito do jogo! Acho que todo jogador órfão do Super Nes sentirá uma grande nostalgia ao jogar o jogo. Parabéns!”;*
- b) Aluno 2: *“Uma pequeno problema está na diferenciação entre letras minúsculas e maiúsculas (Mover(Direita)) sendo que na programação real isso pode causar dificuldades de quem joga e pega um cood ou dev para programar. O demais tudo 100%”;*
- c) Aluno 3: *“Acredito que ele deveria informar o quanto ele se move para a direita com cada comando mover(direita); (e para a esquerda também), porque às vezes falta só um para terminar e é preciso repetir tudo de novo”;*
- d) Aluno 4: *“Colocar mais detalhes nas explicações de cada fase (dos objetivos secundários da própria fase e dos comandos que podem ajudar o jogador a completá-la)”.*

Algumas decisões do jogo questionadas pelos alunos foram inseridas de forma proposital tal como a forma de movimento mencionado pelo aluno 1, “*onde o usuário “arrastava” os comandos para uma linha de programação e depois executava as atividades*”. No jogo proposto, a ideia era fazer com que o aluno escrevesse o código em vez de arrastar para que pudesse trabalhar o raciocínio e as habilidades de resolver um problema de lógica.

O aluno 2 menciona a diferenciação das maiúsculas e minúsculas na escrita dos comandos no jogo. Muitas linguagens de programação utilizam um tipo de análise tipográfica onde um comando *Mover* é diferente de *mover*. Assim, no jogo proposto, não foi utilizada essa regra para que o jogo fluísse de uma forma mais interativa, fazendo com que o aluno não precise se preocupar com esse detalhe.

Após a análise da questão descritiva, algumas propostas foram apontadas para um trabalho futuro como:

- a) escrever um manual de forma mais elaborada para que sejam esclarecidas todas as dúvidas iniciais do aluno;
- b) revisar e corrigir pequenos bugs encontrados pelos alunos;
- c) elaborar um sistema de métrica para que o aluno saiba quantos passos são necessários para chegar a um local;
- d) disponibilizar o jogo via web, para que rode em qualquer plataforma independente do sistema operacional.

De forma geral, os resultados demonstram que o jogo foi bem aceito pelos alunos, apesar do curto período de tempo em que estes utilizaram o mesmo.

8. CONSIDERAÇÕES FINAIS

Esta dissertação teve como objetivo principal apresentar o desenvolvimento de um Jogo Sério denominado Super Mario Logic para apoiar o processo de ensino aprendizagem de lógica de programação, buscando fornecer uma alternativa dinâmica, interativa e motivacional. Para isto, foi realizada uma sequência de passos resgatando vários conceitos, entre eles de Jogos Sérios, Pensamento Computacional, estruturas de programação, trabalhos correlatos e os aspectos metodológicos escolhidos para a execução do projeto.

O estudo relativo ao ensino de Lógica de Programação trouxe um embasamento sobre as estruturas de programação que foram utilizadas no jogo Super Mario Logic além dos conceitos do ensino de lógica por meio do Pensamento Computacional. Tendo em vista este segundo item, as seis características do Pensamento Computacional de Barr e Stephenson (2011), descritas na seção 2.2, foram analisadas e integradas no jogo e em cada desafio.

O aluno foi desafiado a utilizar o computador como ferramenta, montar uma estratégia lógica para alcançar o desafio primário e secundário, resolver o problema através de abstração de código, implementar a solução com o objetivo de alcançar a combinação de passos e recursos mais eficiente e eficaz e, por fim, transferir o processo de resolução de problemas a uma ampla variedade de problemas através da experiência adquirida em cada problema. Também foi necessário abordar sobre jogos educacionais e Jogos Sérios para que fosse possível desmistificar que jogos têm como única meta a diversão sem cunho de aprendizagem, e sim que jogos também podem facilitar a aprendizagem por meio de diversão e da motivação criada.

Além disso, foram abordados os elementos fundamentais para o sucesso de qualquer jogo, conforme mencionado por Huizinga (2007), e que foram necessários para definir os requisitos mínimos a serem utilizados no jogo desenvolvido, tais como a *liberdade*, em que o jogador pode escolher o personagem com o qual ele mais se identifica, a *delimitação*, que foi baseada na história original do *Super Mario World*, a *imprevisibilidade*, necessária em certos momentos, por meio do surgimento de inimigos (Thwomps e Piranha Plants), em que o aluno deveria achar o momento certo para não tocá-los e acabar perdendo o jogo; no que diz respeito aos *Regulamentos e Normas*, estes foram necessários para que todos os jogadores agissem da mesma forma

dentro do jogo; e, por fim, a *Ficção*, pois o jogo se passa no Reino dos Cogumelos, um mundo fictício criado pelo desenvolvedor da franquia Super Mario.

A análise dos trabalhos correlatos tornou possível a verificação dos pontos positivos e mecânicas dos jogos apresentados e também os pontos negativos e ideias para aprimorar e desenvolver um Jogo Sérió proposto nessa dissertação. Foi importante esse levantamento, pois trouxe um enriquecimento no desenvolvimento de ideias para o jogo tais como: conceitos aplicados nos jogos e, também, a plataforma para o qual seria desenvolvido.

Finalizadas essas questões, foi necessário pensar em uma metodologia que pudesse ajudar a desenvolver o Jogo Sérió. Para isto, a melhor solução para esse trabalho foi a de utilizar a metodologia INTERAD de Passos (2011), a qual possui um conjunto detalhado de etapas para planejar e desenvolver *softwares educacionais*, com foco no desenvolvimento de interfaces e interações do sistema que, no caso do jogo desenvolvido, foi imprescindível. A adoção de uma metodologia formal de desenvolvimento tornou possível resolver questões como: a) a organização no desenvolvimento do jogo; b) formalização das etapas do desenvolvimento; c) geração de uma documentação completa do sistema, que muitas vezes não é feita no decorrer do desenvolvimento de software; e, também, d) a possibilidade de integrar em uma mesma metodologia aspectos pedagógicos e técnicos inerentes ao tipo de *software* que estava sendo desenvolvido.

Resgatando essas abordagens e trazendo isso em forma de um produto, levantou-se o problema de pesquisa apresentado nesta dissertação: *É possível apoiar o processo de ensino aprendizagem de lógica de programação por meio de um Jogo Sérió, buscando fornecer uma alternativa dinâmica, interativa e motivacional?*

Pensando nesta questão, foi necessário considerar como coletar dados que pudessem responder à mesma. Para esse fim, foi adaptado e aplicado o método GameFlow, que abordou sete categorias de questões relacionadas à experiência do usuário em relação a jogos. De uma forma geral, o jogo Super Mario Logic obteve uma avaliação positiva, e, assim, pode-se concluir que o objetivo proposto foi alcançado, considerando que, do ponto de vista dos estudantes, o jogo desenvolvido foi considerado atrativo e desafiador.

Os resultados obtidos vão ao encontro do que destacam os autores Freitas e Liarokapis (2011), de que os Jogos Sérios possuem um grande potencial para complementar a educação tradicional, bem como ao afirmado por Scaico *et al.* (2012), que descrevem que o uso de jogos na educação favorece o desenvolvimento da lógica, estratégia, análise e algumas vezes da memória, partindo-se da tentativa e erro no vencer das fases do jogo.

Como sugestões de trabalhos futuros destacam-se as já mencionadas no capítulo de análise e também: a) mostrar ao final de cada desafio os comandos em outras linguagens de programação como Java, Pascal e C; b) criar um sistema de *ranking*, onde seja possível visualizar os melhores jogadores para promover o desafio entre os alunos; c) desenvolver um sistema de tempo inicial para cada desafio, fazendo com que o aluno analise o desafio e pense em soluções para o problema proposto; e, por fim, inserir o jogo em sala de aula durante um semestre em turmas de algoritmos para buscar avaliar sob o ponto de vista pedagógico de aprendizagem de algoritmos e lógica computacional, avaliação esta que não foi realizada neste trabalho.

REFERÊNCIAS

- AMARAL, A. V. G. Novas Competências em Educação: o papel das novas tecnologias de informação e comunicação na escola atual. **Revista Iberoamericana de Educación / Revista Ibero-americana de Educação**, v. 53/3, n. 1681-5653, p. 1–5, 2010.
- ANDRADE, D. et al. Proposta de Atividades para o Desenvolvimento do Pensamento Computacional no Ensino Fundamental. **Anais do Workshop de Informática na Escola**, v. 1, n. Cbie, p. 169–178, 2013.
- ARRUDA, E. P. **Fundamentos para o Desenvolvimento de Jogos Digitais: S{é}rie Tekne**. [s.l.] Bookman Editora, 2014.
- ASSOCIATION, E. S. **2015 Essential Facts About the Computer and Video Game Industry**. Disponível em: <<http://www.theesa.com/wp-content/uploads/2015/04/ESA-Essential-Facts-2015.pdf>>. Acesso em: 28 out. 2015.
- BARR, V.; STEPHENSON, C. Bringing Computational Thinking to K-12: What is Involved and What is the Role of the Computer Science Education Community? **ACM Inroads**, v. 2, n. 1, p. 48–54, 2011.
- BELLI, S.; RAVENTÓS, C. L. Breve historia de los videojuegos. **Athenea Digital. Revista de Pensamiento e Investigación Social**, n. 14, p. 159–179, 2008.
- BLIKSTEIN, P. **O pensamento computacional e a reinvenção do computador na educação**. Disponível em: <http://www.blikstein.com/paulo/documents/online/ol_pensamento_computacional.html>. Acesso em: 14 jun. 2015.
- BORGES, C. et al. KidCoder: Uma Proposta de Ensino de Programação de forma Lúdica. **Anais do XXVI Simpósio Brasileiro de Informática na Educação**, v. 1, n. 02, p. 687–691, 2015.
- CASTRO, R. P. Das formas de ensinar e aprender e os desafios do século XXI. **Revista Educação Temática Digital**, v. 9, n. 1676-2592, p. 115–135, 2007.
- DIZON, J. **15 Best-Selling Video Games Of All Time**. Disponível em: <<http://www.techtimes.com/articles/32614/20150213/15-best-selling-video-games-of-all-time.htm>>. Acesso em: 11 nov. 2015.
- FADEL, L. M. et al. **Gamificação na Educação**. São Paulo: Pimenta Cultural, 2014.
- FALKEMBACH, G. A. M.; ARAUJO, F. V. DE. **Aprendizagem de Algoritmos: Dificuldades na Resolução de Problemas**. Anais SULCOMP. **Anais...2013** Disponível em: <<http://periodicos.unesc.net/index.php/sulcomp/article/view/916/909>>
- FORBELLONE, A. L. V.; EBERSPÄCHER, H. FREDERICO. **Lógica de Programação - A construção de algoritmos e estruturas de dados**. 3 Ed ed. São Paulo: Prentice Hall, 2005.
- FREITAS, A. K. M. DE. Psicodinâmica das Cores em Comunicação. **Nucom Nucleo de Comunicação**, v. 12, p. 18, 2007.

- FREITAS, S. DE; LIAROKAPIS, F. Serious Games: A New Paradigm for Education? In: **Serious Games and Edutainment Applications**. [s.l: s.n.]. p. 9–23.
- FURLAN, M. A. DE S. et al. **Algoritmos e Lógica de Programação**. 2 Ed ed. São Paulo: Cengage Learning, 2012.
- GOUWS, L.; BRADSHAW, K.; WENTWORTH, P. **Computational Thinking in Educational Activities An evaluation of the educational game Light-Bot**. p. 10–15, 2013.
- GRINTERS, R. **Super Mario Sales Data: Historical Units Sold Numbers for Mario Bros on NES, SNES, N64...** Disponível em: <http://www.gamecubicle.com/features-mario-units_sold_sales.htm>. Acesso em: 14 nov. 2015.
- HUIZINGA, J. **Homo ludens: o jogo como elemento da cultura**. 5. ed. São Paulo: Perspectiva, 2007.
- KAZIMOGLU, C. et al. A serious game for developing computational thinking and learning introductory computer programming. **Procedia - Social and Behavioral Sciences**, v. 47, p. 1991–1999, 2012.
- MANZANO, J. A. N. G.; OLIVEIRA, J. F. DE. **Estudo dirigido de Algoritmos**. 15^a. ed. São Paulo: Editora Érica Ltda, 2012.
- MENDONÇA, R. L.; MUSTARO, P. N. Elementos imersivos e de narrativa como fatores motivacionais em serious games. **SB Games**, v. 10, p. 10, 2011.
- NEVES, D. E. et al. Avaliação de jogos sérios casuais usando o método GameFlow. **Revista Brasileira de Computação Aplicada**, v. 6, n. 1, p. 45–59, 2014.
- PASSOS, P. C. S. J. **Interad: uma metodologia para design de interface de materiais educacionais digitais**. [s.l.] UFRGS, 2011.
- PINTO, I. M. et al. **Saberlândia: Plataforma Lúdica Integrando Robótica e Multimídia para Educação**. IV Seminário Jogos Eletrônicos, Educação e Comunicação – construindo novas trilhas. **Anais...** Salvador: 2008 Disponível em: <<http://www.comunidadesvirtuais.pro.br/seminario4/trab/imp.pdf>>. Acesso em: 28 set. 2015
- SALEN, K.; ZIMMERMAN, E. Rules of Play: Game Design Fundamentals. **The MIT Press Cambridge**, v. 1, n. 0, p. 672, 2004.
- SCAICO, P. D. et al. **Implementação de um Jogo Sérioso para o Ensino de Programação para Alunos do Ensino Médio Baseado em m-learning**. XX Workshop sobre Educação em Informática. **Anais...** Curitiba: Sociedade Brasileira de Computação, 2012
- SEBESTA, R. W. **Conceitos de Linguagens de Programação**. [s.l: s.n.].
- SUSI, T.; JOHANNESSON, M.; BACKLUND, P. Serious Games – An Overview. **Elearning**, v. 73, n. 10, p. 28, 2007.
- UNITY, T. **Unity - Manual: Platform Specific**. Disponível em: <<http://docs.unity3d.com/Manual/PlatformSpecific.html>>. Acesso em: 19 set. 2015.
- WING, J. M.; WING, J. M. Computational thinking. **Communications of the ACM**, v. 49, n. 3, p. 33–35, 2006.

PDI. Plano de Desenvolvimento Institucional do Instituto Federal de Educação, Ciência e Tecnologia Farroupilha. 2014. Disponível em: <http://www.iffarroupilha.edu.br/site/midias/arquivos/2014816145120955pdi_2014_2018.pdf>. Acesso em: 17 maio, 2015.

APÊNDICE A - MODELO DE AVALIAÇÃO GAMEFLOW

Item	Critério	Avaliador 1			Avaliador 2			Avaliador 3		
		M	L	RL	M	L	RL	M	L	RL
Concentração	Jogo fornece grande quantidade de estímulos.	2	2	2	2	2	3	2	2	2
	Jogo fornece estímulos que chamem atenção.	2	1	2	2	2	3	2	2	3
	A atenção do jogador é capturada rapidamente e seu foco é mantido ao longo do jogo.	3	3	3	2	3	3	2	2	3
	Jogadores não são sobrecarregados.	4	4	4	3	3	3	4	4	4
	A carga de trabalho é alta, porém adequada aos limites de cognição, percepção e memória do jogador.	3	2	4	3	2	3	3	2	3
	Jogadores permanecem atentos a tarefas importantes.	3	4	4	2	3	3	4	4	4
		2,8	2,7	3,2	2,3	2,5	3,0	2,8	2,7	3,2
Desafio	Os desafios são adequados às habilidades do jogador.	4	2	4	3	2	4	3	3	4
	Diferentes níveis de desafio são oferecidos.	4	2	3	4	3	4	3	3	3
	O nível de desafio aumenta à medida que o jogador progride e melhora suas habilidades.	3	1	1	3	2	3	3	2	3
	Novos desafios são fornecidos em ritmo apropriado.	3	1	1	3	2	3	2	2	3
		3,5	1,5	2,3	3,3	2,3	3,5	2,8	2,5	3,3
Habilidades do jogador	O jogador não precisa ler o manual para iniciar o jogo.	3	4	4	3	3	3	4	4	4
	Aprender o jogo não é chato, mas sim divertido.	4	3	4	2	2	3	3	2	3
	Há um help no próprio ambiente do jogo.	4	2	4	4	4	4	4	4	4
	O aprendizado ocorre com tutoriais jogáveis e níveis iniciais.	4	3	4	2	2	3	1	2	3
	O aumento das habilidades do jogador ocorre em um ritmo adequado ao seu progresso no jogo.	4	1	3	3	1	3	2	1	4
	O jogador é recompensado por seu esforço e desenvolvimento de habilidades.	1	1	1	1	1	1	1	1	1
	A interface e a mecânica do jogo são de fácil aprendizado.	4	3	4	4	4	4	4	4	4
		3,4	2,4	3,4	2,7	2,4	3,0	2,7	2,6	3,3
Controle	O jogador sente-se no controle de personagens e de suas interações no mundo do jogo.	0	0	0	0	0	0	0	0	0
	O jogador sente-se no controle da interface.	2	1	1	1	1	1	1	1	1
	O jogador sente que pode controlar o andamento do jogo (início, parada, saída, salvamento, etc.).	2	2	2	2	2	2	2	2	2
	O jogador não pode cometer um erro que prejudique o jogo e encontra suporte, caso isso ocorra.	4	3	4	4	4	4	4	4	4
	O jogador sente que seus controles e ações são importantes e que refletem no mundo do jogo.	0	0	0	0	0	0	0	0	0
	O jogador tem a sensação de controle sobre suas ações e estratégias, e sente-se livre para jogar como quiser.	1	1	1	1	1	1	1	1	1
			2,3	1,8	2	2	2	2	2	2
Objetivos	Os objetivos principais do jogo são claros e apresentados desde o início do jogo.	4	4	4	4	4	4	4	4	4
	Os objetivos intermediários são claros.	4	4	4	4	4	4	4	4	4
		4	4	4	4	4	4	4	4	4
Feedback	O jogador recebe <i>feedbacks</i> sobre seu progresso.	2	2	2	2	2	2	2	2	2
	O jogador recebe <i>feedback</i> imediato sobre suas ações.	2	1	3	3	2	4	1	1	3
	Status e pontuação estão disponíveis ao jogador.	1	2	1	1	1	1	1	1	1
		1,7	1,7	2,0	2,0	1,7	2,3	1,3	1,3	2,0
Imersão	O jogador torna-se menos consciente do que ocorre ao redor.	3	2	2	2	2	3	2	2	2
	O jogador torna-se menos consciente de si mesmo.	1	1	1	1	2	3	2	2	2
	O jogador é envolvido emocionalmente no jogo.	0	0	0	0	0	0	0	0	0
	O jogador é envolvido visceralmente no jogo.	0	0	0	0	0	0	0	0	0
		2,0	1,5	1,5	1,5	2,0	3,0	2,0	2,0	2,0
Interação	Jogo dá suporte à competição e à cooperação entre jogadores.	1	1	1	1	1	1	1	1	1
	Jogo dá suporte à interação social (chat, etc.).	1	1	1	1	1	1	1	1	1
	Há suporte a comunidades, dentro e fora do jogo.	1	1	1	1	1	1	1	1	1
		1,0	1,0	1,0	1,0	1,0	1,0	1,0	1,0	1,0

Legenda:

M = jogo de memória, L = jogo de linguagem, RL = jogo de raciocínio lógico
 0 = não se aplica, 1 = deveria ter, mas não tem, 2 = ruim, 3 = médio, 4 = bom