UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ

CÂMPUS CORNÉLIO PROCÓPIO

DIRETORIA DE GRADUAÇÃO E EDUCAÇÃO PROFISSIONAL

TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

ELIAS DE MORAES FERNANDES

**NONDA: SERIOUS GAME NA EDUCAÇÃO DE RESÍDUOS SÓLIDOS URBANOS**

TRABALHO DE CONCLUSÃO DE CURSO

CORNÉLIO PROCÓPIO

2016

ELIAS DE MORAES FERNANDES

**NONDA: SERIOUS GAME NA EDUCAÇÃO DE RESÍDUOS SÓLIDOS URBANOS**

Trabalho de Conclusão de Curso de graduação, do curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas da Coordenação de Informática – TADS – da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para a obtenção do título de Tecnólogo.

Orientador: Prof. Dr. Paulo Augusto Nardi

CORNÉLIO PROCÓPIO

2016

Dedico este trabalho à minha família.

**AGRADECIMENTOS**

Agradeço primeiramente à Deus, por ter me dado inteligência e conhecimento para realização desse TCC.

Ao meu orientador Prof. Dr. Paulo Augusto Nardi, pela sabedoria com que me guiou nesta trajetória.

Aos meus colegas de sala.

A Secretaria do Curso, pela cooperação.

Gostaria de deixar registrado também, o meu reconhecimento à minha família, pois acredito que sem o apoio deles seria muito difícil vencer esse desafio.

Agradeço também à minha noiva, pela ajuda e companherismo em todas etapas da minha graduação.

Enfim, a todos os que por algum motivo contribuíram para a realização desta pesquisa.

Epígrafe – elemento opcional

Espaço destinado à epígrafe (elemento opcional). Nesta folha, o autor usa uma citação, seguida de indicação de autoria e ano, relacionada com a matéria tratada no corpo do trabalho.

**LISTA DE GRÁFICOS**

Elemento opcional

|  |  |
| --- | --- |
| GRÁFICO 1 – CULTURA DA INFORMAÇÃO NAS INSTITUIÇÕES................... | 80 |
| GRÁFICO 2 – OBTENÇÃO E COLETA DE INFORMAÇÕES............................. | 82 |
| GRÁFICO 3 – UTILIZAÇÃO DE SISTEMAS EM TEMPO REAL E A DISPONI-BILIDADE DE COMPUTADORES................................................ | 86 |
| GRÁFICO 4 – FLUXO E COMPARTILHAMENTO DE INFORMAÇÕES E CO-NHECIMENTO.............................................................................. | 92 |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**LISTA DE FIGURAS**

|  |  |
| --- | --- |
|  |  |
| **Figura 1** – Tradução de: Variáveis Tecnológicas influenciando telepresença 1212  **Figura 4** – Fluxograma de camadas do MVC 1919  **Figura 7** –Personagem Nonda: Spritesheet com animação de Pular 2525  **Figura 9** – Jogo Nonda: Inimigo Pássaros 2626  **Figura 11** – Jogo Nonda: Inimigo Formiga 2626  **Figura 13** – Diagrama de Classe 3029  **Figura 14** – Fluxograma Animação do personagem Nonda (Player) 3130  **Figura 15** - Código usando Coroutines 3332  **Figura 16** - Trecho de código da classe ItemSpawner 3433 |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**LISTA DE ABREVIATURAS**

Elemento opcional

|  |  |
| --- | --- |
| acep. | Acepção |
| Biol. | Biologia |
| flex. | Flexão |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**LISTA DE SIGLAS**

|  |  |
| --- | --- |
| ABRALPE | Associação Brasileira das Empresas de Limpeza Pública e Resíduos Especiais |
| TIC | Tecnologia da Informação e Comunicação |
| PGRS | Programa de Gestão de Resíduos Sólidos |
| REA | Recursos Educationais Abertos |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**LISTA DE ACRÔNIMOS**

Elemento opcional

|  |  |
| --- | --- |
| CAE | Computer Aided Engineering |
| FORTRAN | Formula Translation |
| IPPUC | Instituto de Pesquisa e Planejamento Urbano de Curitiba |
| NASA | National Aeronautics and Space Administration |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**LISTA DE SÍMBOLOS**

Elemento opcional

|  |  |
| --- | --- |
| Υ | Coeficiente de atenuação da função da onda do elétron no óxido |
| λ | Parâmetro de ajuste para modulação do comprimento do canal |
| σ | Efeito DIBL |
| Ө | Parâmetro de entrada para ajuste de mobilidade |
| έ | Permissividade elétrica do silício |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

**RESUMO**

De M. Fernandes, Elias. **Nonda: Serious Game na Educação de Resíduos Sólidos Urbanos.** 2016. 15f. Trabalho de Conclusão de Curso (Graduação) – Tecnologia em Análise e Desenvolvimento de Sistemas. Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2016.

A tecnologia da informação aplicada à educação tem permitido o uso de inúmeras ferramentas com finalidade de difundir o conhecimento nas diversas áreas de ensino tais como Humanas, Exatas e Biológicas. Da fusão entre educação e o entretenimento tem nascido os jogos sérios educacionais, que demandam conteúdo sucinto e de suma importância. Pode-se elaborar jogos educacionais para temáticas como a Vermicompostagem - sistema de compostagem que trabalha com a bioxidação de resíduos sólidos orgânicos no envolvimento de minhocas na fauna microbiana para surgimento de húmus. Nesse contexto, o presente trabalho objetiva criar um jogo educacional como material didático de apoio sobre a Vermitecnologia. Desse modo, é apresentado uma proposta de um Serious Game nessa temática, detalhes de sua construção e desenvolvimento.

**Palavras-chave**: Vermitecnologia, Vermicompostagem, Educação, Serious Games, Mobile, Compostagem.

**ABSTRACT**

De M. Fernandes, Elias. **Nonda: Serious Game in Urban Solid Waste Education.** 2016. 15f. Trabalho de Conclusão de Curso (Graduação) – Tecnologia em Análise e Desenvolvimento de Sistemas. Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2016.

Information technology applied in education has allowed the use of numerous tools with the purpose of disseminating knowledge in different educational areas such as Humanities, Physical and Biological. The fusion of education and entertainment was born the educational games that require succinct content and of critical importance. Educational games can be created using themes as Vermicomposting - composting system that works with the Bio-oxidation of organic solid waste in the involvement of earthworms on microbial fauna to emergence of humus. In this context, the present work aims to create n educational game as a didactic material support about Vermitechnology. Thus, it is presented a proposal of a Serious Game in that subject, details of contruction and development.

**Keywords:** Vermitechnology, Vermicomposting, Education, Serious Game, Mobile, Composting.

**SUMÁRIO**

1. **INTRODUÇÃO** 6
2. **OBJETIVOS 9**
   1. Objetivo Geral 9
   2. Objetivos Específicos 9
3. **FUNDAMENTAÇÃO TEÓRICA 10**
   1. *Serious games* para interagir e envolver 10
   2. Vermicompostagem 12
4. **MATERIAIS E MÉTODOS 17**
   1. Processo de Software *Scrum Solo* 18
   2. Design Pattern 18
   3. MVC 18
   4. Proposta 19
   5. Gênero 19
   6. Enredo 19
   7. *Storyboard* 20
   8. *Gameplay* – Perspectiva Centrada no Jogador 22
   9. Mecânica do Jogo 23
   10. *Game Design* 23
   11. Personagem 23
   12. Controle 24
   13. Interface 25
   14. Inimigos 25
       1. Pássaros 25
       2. Sanguessuga 26
       3. Formiga 26
   15. *Level Design* 26
   16. Arte 26
5. Cronograma 28

**REFERÊNCIAS** 30

**1 INTRODUÇÃO**

É notório tanto o aumento da população, como dos resíduos sólidos urbanos no Brasil. Isso acarreta a preocupação relativa ao alto índice de destinação irregular desses resíduos bem como a falta de mecanismos para auxiliar na decomposição ecologicamente corretas desses resíduos. A vermicompostagem - técnica que usa minhoca para produzir húmus e adubar a terra - (NDEGWA, THOMPSON, 2001) é uma forma correta de destinação dos resíduos.

Segundo Nuernberg (2014), atualmente o país carece por serviços básicos (coleta e destinação adequada) e orientação para população de procedimento com finalidade de reduzir a contaminação do meio ambiente, diminuir o impacto na saúde pública entre outros fatores. Uma das formas que já se tem é a separação do lixo ecotóxico e coleta seletiva. Porém, a gestão para um sistema de tratamento de resíduos sólidos urbanos usando vermicompostagem ainda necessita de orientação e educação das pessoas.

Com o advento das novas Tecnologias da Informação e Comunicação (TIC), diversas tecnologias além de computadores têm sido empregadas no processo de ensino-aprendizado, como o caso dos dispositivos móveis. Esses têm como vantagem a mobilidade e podem ser acessados em qualquer lugar, diferentemente do computador pessoal. De acordo com Tarouco (2004), a importância do uso dos computadores e das novas tecnologias na educação deve-se hoje não somente ao impacto dessas novas tecnologias (ferramentas) na nossa sociedade e às novas exigências sociais e culturais que se impõe, mas também ao surgimento da Tecnologia Educativa.

A partir do uso das Tecnologias voltadas para a educação, o objetivo dessa proposta é conscientizar as crianças sobre a importância da vermicompostagem na gestão de resíduos por meio do desenvolvimento de um jogo mobile, material didático de apoio para o Departamento Acadêmico de Química e Biologia (DAQBI) da Universidade Tecnológica Federal do Paraná (UTFPR), câmpus Curitiba (UTFPR-CT).

**JUSTIFICATIVA**

Segundo a ABRELPE – Associação Brasileira de Empresas de Limpeza Pública e Resíduos Especiais, em pesquisa realizada em 2013, foi gerada no Brasil mais de 76 milhões de toneladas de resíduos sólidos urbanos (ABRELP, 2014) e, no Brasil a produção de lixo (21%) mais que dobrou em relação ao número de população (9,65%) nos últimos 10 anos.

Ainda nesses dados, o problema se encontra nos destinos finais dos Resíduos Sólidos Urbanos (RSU) que tem 58,4% destinados adequadamente e 41,6% inadequadamente. Esses números parecem promissores se comparado com a quantidade de lixo que foi produzido nos últimos 10 anos, porém tem-se uma longa jornada a fim de destinar corretamente esses resíduos (TRIGUEIRO, 2013).

Em algumas regiões do país, a iniciativa de Coleta Seletiva parece desfavorável ao número de casos que tentam apoiar o mesmo. Por exemplo, na região Nordeste, 57,2% da população não tem apoio para fazer a coleta; e, no Centro-oeste esse número chega a 62,5%. Ainda mais, essas duas regiões somam 30.3% das participações do total de RSU coletados (ABRELP, 2014), o que deixa uma lacuna que precisa ser preenchida.

Segundo o Ministério do Meio Ambiente, (2016):

A Lei nº 12.305/10, meta a prevenção e a redução na geração de resíduos, tendo como proposta a prática de hábitos de consumo sustentável e um conjunto de instrumentos para propiciar o aumento da reciclagem e da reutilização dos resíduos sólidos (aquilo que tem valor econômico e pode ser reciclado ou reaproveitado) e a destinação ambientalmente adequada dos rejeitos (aquilo que não pode ser reciclado ou reutilizado).

O índice de abrangência da Coleta de RSU no Brasil em 2014 foi de 90,68% (ABRELPE, 2014). Isso significa que o Brasil está crescendo na quantidade de RSU gerado, assim como a população brasileira. Algumas regiões se sobressaem, devido ao maior número de grandes centros urbanos.

Sabe-se que a incidência de Coleta Seletiva em municípios pequenos (cerca de 50 mil habitantes) é menor que nos grandes centros. É necessário levar a esses pequenos municípios o incentivo a cultivar essas boas práticas de redirecionamento adequado do lixo e da vermicompostagem, sem deixar de atender as metrópoles.

Um passo para tornar a coleta de resíduos sólidos eficiente é conscientizar o papel da população na participação da Coleta Seletiva, que pode começar domesticamente e crescer para uma coleta de nível industrial, do tamanho de uma grande empresa ou Universidade, que é o caso da UTFPR Curitiba, por exemplo.

Uma alternativa promissora para conscientizar um grande número de pessoas são os jogos digitais educativos pois pode ser inserido dentro das salas de aulas que por sua vez são levados até membro familiares, fazendo a ciclo completo de divulgação do problema. Embora os serious game seja um segmento recente no Brasil, começou a ganhar espaço assim como aconteceu com os games casuais em meados de 2008 (SAMPAIO, 2008). Comparativamente, a indústria de jogos digitais educativos tem aumentado 26% a cada ano (Innovation House Rio, 2015), desempenhando o papel de dramatizar os problemas, contribuírem para desenvolvimento de estratégias e rápidas tomadas de decisões, levando à um rápido processo de feedback.

Com o apoio do DIRGRAD na Produção de Recursos Educacionais Abertos (REA), a construção do material de apoio para ser usado dentro da sala de aula junto com cartilha ensinando sobre vermicompostagem.

**OBJETIVOS**

Com base no que foi exposto sobre a situação do Brasil no panorama da coleta de Resíduos Sólidos Urbanos e a possibilidade do uso da tecnologia aplicada em jogos digitais para incentivar formas de reutilização desses resíduos orgânicos produzido pelo próprio gerador, abaixo estão relacionados os objetivos gerais e específicos.

Objetivo Geral

* Criar um jogo educativo sobre vermicompostagem para conscientização e a importância de tratamento de resíduos sólidos urbanos (separação correta do lixo orgânico que pode ser reaproveitado daquele que não pode) como demanda do Departamento Acadêmico de Química e Biologia (DAQBI) da UTFPR-CT. Trata-se de traduzir para linguagem de jogo usando dispositivos móveis.
* Criar uma metodologia de ensino diferenciada para aplicação em salas de aula, usando a tecnologia como forma de absorção de conteúdo.
* Estimular e sensibilizar alunos possibilitando o contato com elementos envolvidos no processo correto de vermicompostagem apresentados de forma lúdica.
* Ambientalizar os alunos no tema vermicompostagem e cultivar boas práticas para melhor proveito de materiais sólidos orgânicos
* Desenvolver um jogo mobile como material de apoio para educadores aplicar em sala de aula, quando conteúdo é sobre vermicompostagem.

Objetivos Específicos

Desenvolver jogo sério com temática de vermicompostagem para ensinar quais alimentos podem ser reaproveitados para vermicompostagem, seguindo os critérios:

* Sistema de menus intuitivos, contendo botões Jogar, Quiz, Opções e Sair.
* Fases progressivas, aumentando nível de dificuldade gradualmente a cada nova fase.
* Sistema de toque sensitivos para movimentar o personagem principal.
* Sistema para coletar itens e pontuação.
* Implementar predadores (inimigos) com habilidades específicas.
* Sistema para informação de item.
* Sistema para salvar pontuação mais alta (funcionalidade do Quiz).
* Gerenciador de som de fase e efeitos sonoros.
* Desenvolver e distribuir o jogo móvel para celulares com sistema Android superiores a 4.3 (Jelly Bean) e iOS superiores a 6.1.6.
* Arquitetar o jogo para operar em dispositivos móveis de hardware com baixa memória, no mínimo 512MB e processadores 1.0 GHZ ou superiores, como por exemplo celular com processadores Qualcomm Snapdragon MSM8255 (SPECOUT BY GRAPHIC, 2014) .

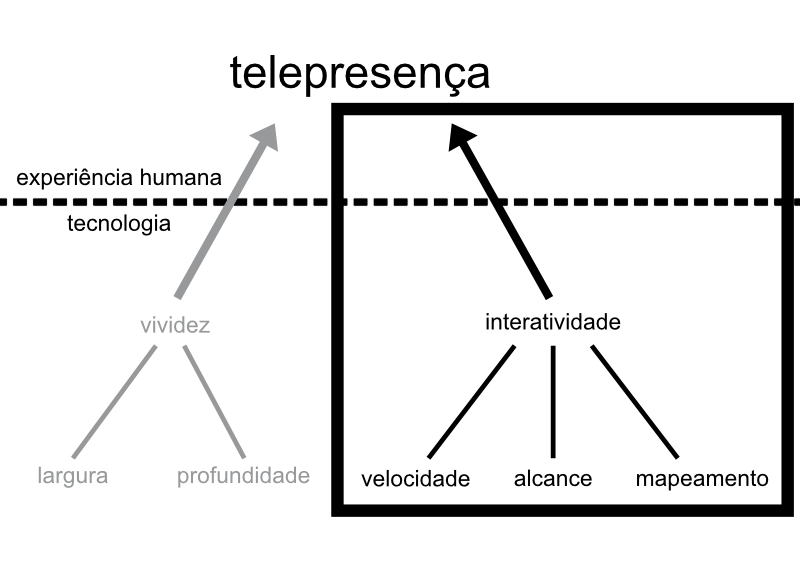
**3 FUNDAMENTAÇÃO TEÓRICA**

Nesse capítulo, são apresentados conceitos sobre as duas principais abordagens para o desenvolvimento do jogo, a vermicompostagem e *serious games*, com foco maior na elaboração do *serious games*. Ainda sobre esse último tópico é apresentada a perspectiva do jogo como ferramenta de aprendizagem por meio de plataforma e fixação do conteúdo citados no capítulo anterior. Por fim, é apresentada uma noção básica do tema que tem base nos estudos desenvolvidos pela Ana Cláudia Nuernberg (2014) visando o desenvolvimento de um jogo mobile a partir da extração da técnica explorada por ela e transformando em linguagem de jogo para interação com o usuário.

3.1 *Serious games* para interagir e envolver

Segundo Clark (1987), um jogo é um contexto em que jogadores se enfrentam tentando alcançar objetivos a partir de regras propostas; Clark evidencia que essas características não são suficientes para definir jogos, sendo que jogadores podem cooperar para contrapor uma situação natural do jogo que não fazem deles jogadores, pois esses não possuem objetivos. Para estruturar um jogo a um desses significados acima é necessário criar subgêneros narrativos, que transpassa os modelos narrativos habituais, onde coloca-se o jogador como principal tomador de decisões e mudar o rumo do jogo, como propõe os jogos digitais (MURRAY, 2003). A aprendizagem baseada nessas tomadas de decisões necessita o constante raciocínio do jogador, podendo esse raciocínio ser para situações estratégicas envolvendo tempo ou pontuais e específicas (JOHNSON, 2005), propósito engajado nos serious games.

sem oferecernem , mas que se beneficia igualmente da interatividade ambiente/jogador

Interatividade descreve medições mecânicas responsáveis pela ação do usuário e resposta da plataforma. São analisadas três particularidades nessa perspectiva: velocidade, mapeamento e alcance como destaca a Figura 1 (STEUER, 1993). A primeira, também conhecida como tempo de resposta, é uma característica importante no sistema de mídia interativa que dita a velocidade em que a plataforma responde às ações do jogador (STEUER, 1993). A segunda constitui a habilidade do sistema mapear os controles e mudar em tempo real de acordo com o ambiente de forma natural e previsível (STEUER, 1993). A última, o alcance, refere-se ao número de possibilidades por ação em um tempo determinado, ou a capacidade de telas toques sensitives na tela, o que significa a liberdade do jogador poder executar ações dentro do ambiente. O jogador precisa ter a sensação de liberdade em executar várias tarefas, mesmo que seja um número limitado de opções (STEUER, 1993).

**Figura 1** – Tradução de: Variáveis Tecnológicas influenciando telepresença

(STEUER, 1993)

É necessário uma plataforma de desenvolvimento e integração que atenda os requisitos da interatividade citados acima. Kanstensmidt (2010) define a plataforma como um sistema capaz de executar jogos desenvolvidos especificamente para aquele ambiente. Cada plataforma disponibiliza recursos tecnológicos de entrada*,* processamento, exibição e, em alguns casos, transferência de dados. A plataforma tem função importante tanto quanto a narrativa e mecânica do jogo. Como exemplo disso cita-se a possibilidade de processar ações comandadas pelo jogador, processá-las de acordo com o comportamento de cada objeto e do sistema de regras e instantaneamente apresentar esse resultado para o jogador. Esse ambiente, juntamente com outros aspectos pode definir o grau de imersão do jogo, pois leva-se em consideração características do game design e do processo de produção, onde foca-se nas limitações técnicas dessa determinada plataforma como por exemplo, adaptações para um gameplay mais atrativo ou simplificado (MENDES T., 2012).

Como apresentado no tópico anterior, a objetividade nas tarefas estabelece uma relação entre o *game* e o jogador, em que o foco nas atividades são extremamente altos (PRENSKY, 2001). Isso ocorre porque existe um potencial imersivo através de desafio e recompensas, caracaterísticas da gamificação. Essas características são utilizadas na mecânica dos *games* em cenários *non games*, criando espaço de aprendizados mediado pelo desafio, prazer e entretenimento (ALVES, L. R. G. et al, 2014). Dessa maneira, o jogador sente-se motivado e engajado para continuar a realizar atividades mesmo sem dicas ou tutoriais. O envolvimento de tarefas ou uma série de tarefas mais parecidas com afazeres do que com divertimento faz com que o jogador se habitue a continuar jogando pelo motivo de ainda existir recompensa, a cada avanço dentro do contexto do jogo. Esses jogos contêm, sistematicamente, uma grande quantidade de objetos, que transmitem, de forma clara, recompensas articuladas (PRENSKY, 2001).

Com isso, a presente proposta foca em utilizar da técnica de vermicompostagem (compostagem com a ação de minhocas e microorganismos que misturam os resíduos sólidos para produzir húmus) para criar um jogo lúdico utilizando os conceitos de jogos apresentados anteriormente.

**3.2** **Vermicompostagem**

A vermicompostagem é um processo bioxidativo (de resíduos sólidos), em que algumas espécies de minhocas detritívoras interagem, afetando positivamente e significativamente as taxas de degradação de uma matéria orgânica, na sua maioria devido às alterações ocorridas nas suas propriedades químicas, físicas ou microbiológicas (LOURENCO, 2015). Essas minhocas se alimentam de folhas mortas, gramíneas e outros resíduos orgânicos em diversos estágios de decomposição que são depositados no solo. Dentre esses resíduos, os principais são: esterco de animais, bagaço de cana-de-açúcar, frutas, verduras, resíduos industriais orgânicos, restos de podas, borras de café entre outros. O resultado dessa alimentação é a produção de húmus, ou vermicomposto, um excremento das minhocas, um produto natural, estável de coloração escura, rica em matéria orgânica, tendo nutrientes facilmente absorvidas pelas plantas (NUERNBERG, 2014). Dentre muitos benefícios, fica aqui esclarecido os principais que de acordo com Sindifrutas (2014) melhoram a porosidade e a aeração do solo, aumenta a vida biológica do solo, com o desenvolvimento de fungos fixadores do oxigênio e bactérias, além da proliferação de microrganismos, aumenta a capacidade de *captação de nutrientes* pelas plantas e pode ser utilizado em todos tipos de culturas.

1 TECNOLOGIAS E FERRAMENTAS

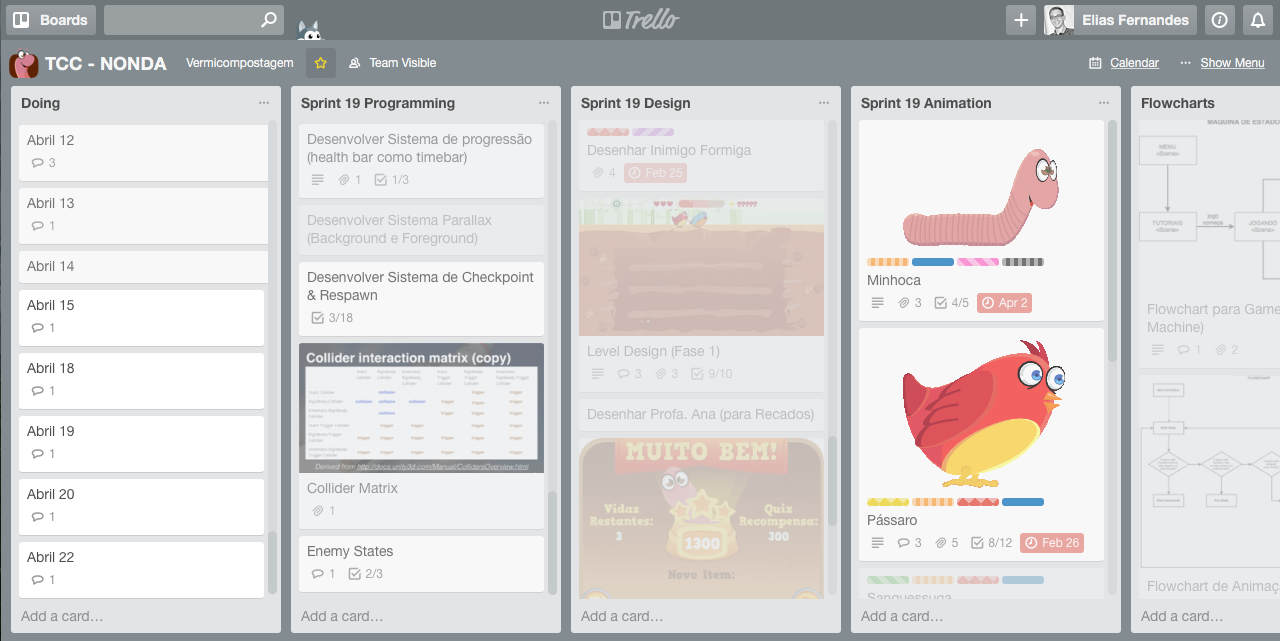
No que diz respeito à tecnologia, foi utilizado a *game engine Unity*, produzida pela *Unity Technologies*. Com *Unity* é possível desenvolver jogos para celulares e sistemas operacionais como Apple e Windows, jogos de consoles para Playstation, Xbox, Wii U, jogos baseados na Web usando Web Player e Web GL, para tecnologia de realidade virtual aumentada como Oculus Rift, Gear VR (da Samsung) além de suporte para TV usando Android TV e Samsung Smart TV. *Unity* tem diversos licenças (também chamado de produtos): *Unity Personal, Unity Plus, Unity Pro e Unity Enterprise*. Para o desenvolvimento do jogo Nonda, foi utilizado o *Unity Personal*, usando projeto em modo 2D. Como mencionado acima, a Unity pode distribuir um jogo em multiplataforma, característica usada para o jogo Nonda. Um fator importante para escolha dessa *engine* é a possibilidade de contar com assets gratuitos e atualizações compatíveis (atualmente na versão 5.4.1) com a necessidade do projeto além dos tutoriais disponíveis em caso de dúvidas ou até mesmo o uso da comunidade para dúvidas e interação.

A linguagem de programação em código aberto escolhida para desenvolver foi o C#. O *Unity* também permite o desenvolvimento nas linguagem de programação *Javascript* e *Boo*. C# é uma linguagem de alto nível que permite desenvolvedores entrar facilmente no processo de desenvolvimento do jogo, aproveitando o máximo dos elementos e técnicas que a linguagem já possui. Por ser uma ramificação do C e C++, tem uma curva de aprendizado menor, e possui programação orientada a objetos como filosofia de design. Todas essa vantagens contribui para criação de códigos fáceis de executar e depurar. Na questão de IDE, o C# é uma das linguagens mais versáteis já existente, aceitando IDEs como Visual Studio, Visual Basic e MonoDeveloper, para plataforma Mac.

Para fazer o *design* e criação de animação serão utilizados o *Affinity Designer* e *Adobe Illustrator CC*. O *Affinity Designer* é um software pago para plataforma Mac e Windows que permite criação de elementos gráficos vetorizados. A escolha deve-se ao fato do mesmo permitir o feature "*Export Persona*" para exportação de *spritesheet* e pixel art pronta para usar no Unity. Para realizar algumas operações inexistentes no *Affinity Designer*, foi utilizado o *Adobe Illustrator*, que é da mesma categoria do *Affinity*. Em alguns momentos foi necessário o uso do Adobe Photoshop CC para redimensionamento de imagens, tamanho de arquivos e outros.

Para versionamento e colaboração do projeto foi utilizado o *GitHub*, que possui versão gratuita para estudante. O *GitHub* é um sistema de controle de versão para software baseado na web, podendo ser manipulado via terminal. O mesmo consiste em repositórios, onde são armazenadas as informações atualizadas de cada projeto. Através de um *link*, qualquer usuário (ou uma equipe) pode baixar, colaborar, atualizar, enviar novas atualizações sem depender de trabalho extra. Isso tudo torna o *GitHub* totalmente flexível.

Para armazenar todas as atividades a serem feitas, foi utilizado o *Trello*, que é uma ferramenta colaborativa de tarefas com versão grátis e paga. Como a versão grátis do Trello é suficiente para listar os requisitos do jogo, as atividades foram divididas em cartões (*Cards*) e listas (*Lists*). Na Figura 2, é mostrado o *Lists* contendo tarefas para um dia (*Doing*) e divisões entre diferentes áreas (Desenhos, Animação, Programação, Testes e Erros etc), marcando a *Sprint* em questão, sendo que terá um List contendo o total de *Sprints* do jogo. Na Figura 12 (ver Seção 6), contém a lista completa de atividades.



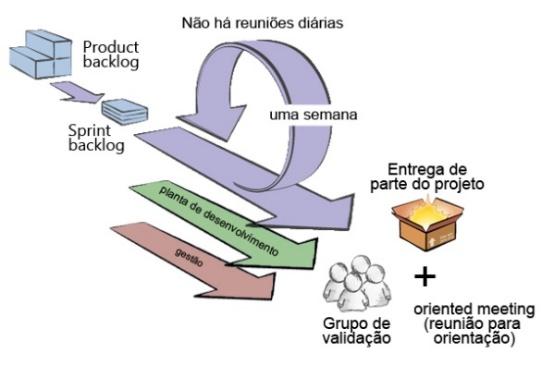
**Figura 2** - Estrutura do Trello usando Scrum Solo

4 MATERIAIS E MÉTODOS

Nessa seção serão descritos e discutidos os materiais e métodos a serem utilizados nesse trabalho. A primeira subseção tem como objetivo descrever a metodologia para dar suporte no processo do jogo. A segunda subseção apresenta a arquitetura *Model View Controller* (MVC)*,* que será usada no desenvolvimento do projeto.

4.1 Processo de Software *Scrum Solo*

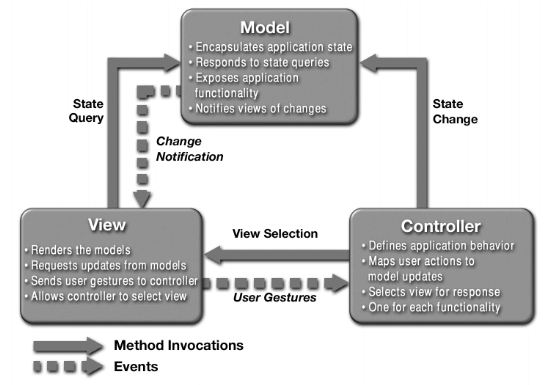
Essa proposta será desenvolvida de acordo com o *framework* S*crum solo*, que tem base no *Scrum* e é uma metodologia Ágil. Dentre as fases do processo estão: detalhes dos requisitos, análise & design, Implementação & teste, aprovação do teste, reavaliar / priorizar novas tarefas. Segundo JAMES, M. (2010), no desenvolvimento de jogos, o *Scrum* tem início a partir do levantamento dos requisitos passando pelo análise e design do UI/UX, ambiente, personagens e animação passando pela implementação da lógica do jogo, até a fase de teste onde finaliza a primeira iteração (*sprint*), sendo que nessa iteração o foco principal é estabelecer quais métodos, instâncias serão reaproveitados para reutilizar nas demais fases. O *Scrum solo*, desenvolvido por professores da UTFPR-CP, beneficia o desenvolvedor solo e tem as boas práticas do *Scrum* e *Personal Software Process* (PSP), processo que auxilia o desenvolvedor a entender a própria performance, desenvolvendo melhorias em suas práticas de trabalho. O *Scrum solo* divide o projeto em várias *sprints* (iterações nos ciclos de desenvolvimento) que duram 1 semana e tem incrementos cada vez que um *product owner* identifica uma nova prioridade no *product backlog* (funcionalidades que o *scrum team* deve desenvolver no software). Durante as iterações, o desenvolvedor codifica e testa pequenas partes do projeto. O diferencial é que entrega uma fatia do software no prazo dito acima, e se necessário existe reunião para orientação entre o grupo de validação (cliente final) e o desenvolvedor e não há reuniões diárias, como acontece no Scrum (FABRI, J. A. et al, 2016, no prelo).



**Figura 3** - Fluxo de uma sprint específica de uma entrega parcial do projeto

4.2 PADRÃO DE PROJETO MVC

Entre os engenheiros de software e arquitetos foi amplamente aceito que a concepção de aplicações em conformidade a esses *designs pattern* facilitariam a reutilização da experiência e conhecimento adquiridos por *experts* ao longo de exaustivos esforços em desenvolver um software de alto nível no mundo real (MASOVER, 2004). A arquitetura *Model* *View* C*ontroller* – MVC – justifica a afirmação do Masover (2004) e o fluxograma contido na Figura 4 possibilita a compreensão do reuso da experiência através da divisão das três camadas de aplicação Modelo de Negócio, Interface de Usuário e Lógica de programação. Em geral, o modelo gerencia os dados do software, a Interface é a camada de apresentação para o usuário e a Lógica trata os eventos para a interface (YONGLEI, 2002).



**Figura 4** – Fluxograma de camadas do MVC

(MASOVER, 2014)

Trazendo o MVC para o desenvolvimento de games, é notório dizer que o fluxo de requisições está sempre na espera de uma ação do usuário ou em uma condição de disparo, seguido do envio da notificação desses eventos para a lógica do jogo que responde na conformidade dos eventos relacionados à ação disparada no início. Essa metodologia introduz outra camada de abstração que ajuda no planejamento do software. Quando divide em dados, interface e decisões há uma redução de numero de arquivos que consequentemente reduzem a complexidade de adicionar funcionalidades ou corrigir problemas (COSTA, 2015).

**4.4** **Proposta**

Nessa seção são descritos a história, fases, mecânica do jogo, característica dos personagens e outros aspectos cruciais para entender o jogo Nonda.

4.5 Gênero

O jogo é uma combinação de dois estilos: *puzzle* e *non-stop running*. O primeiro estilo é apropriado pela arte de forçar o raciocínio do jogador antes de efetuar uma ação que resulte uma reação da parte lógica do game. O segundo, implica que o jogador não terá domínio sobre a velocidade do personagem, ou seja, o mesmo está sempre em movimento. Para mudar de direção do personagem, o jogador precisará deslizar na parte esquerda da tela (ver Seção 5.2).

4.6 Enredo

A história da minhoca Nonda acontece no minhocário da UTFPR, que fica sob a responsabilidade da Professora Ana.

Ana educa a todos através de palestras como deve ser feito a vermicompostagem corretamente dentro da sala de aula ou fora, como no pátio da UTFPR ou no minhocário, onde Nonda trabalha.

O minhocário (*level design*) é composto por plataformas, que caracteriza o labirinto, e o objetivo de Nonda é cuidar para que nenhum inimigo tome posse e reproduza ou infecte a caixa de terra com agente ecotóxico. Para que Nonda continue produzindo biofertilizantes de qualidade (criando húmus ao defecar) e continue sempre forte, a Professora Ana sempre abastece a caixa de Nonda com resíduos orgânicos.

O jogo conta com 3 inimigos, que são: o pássaro, o sanguessuga e a formiga. Cada um destes possui poderes diferentes, podendo ser mais lento, porém o dano é maior, como por exemplo o sanguessuga, ou rápido e invasor, como o caso do pássaro. A formiga terá uma colônia (carreira) de formigas que poderão atacar Nonda.

4.7 Storyboard

Antes de iniciar o jogo, é mostrado uma história sobre os personagens envolvidos (principal e inimigos).

A concepção geral do jogo, como mostra a Figura 5 é fazer o jogador entender o que pode e não alimentar a minhoca através da exploração do minhocário, quais os empecilhos e inimigos que ela pode ter para produzir húmus de qualidade e posteriormente indicar o desempenho alcançado durante a fase.

Na primeira fase o objetivo é proteger a comida distribuída aleatoriamente dentro do minhocário (ambiente do jogo). O jogador precisa coletar esses alimentos e, como desafio, evitar que o inimigo ataque a minhoca. Além disso, o jogador deve atingir uma pontuação especificada a fim de ir para próxima fase.

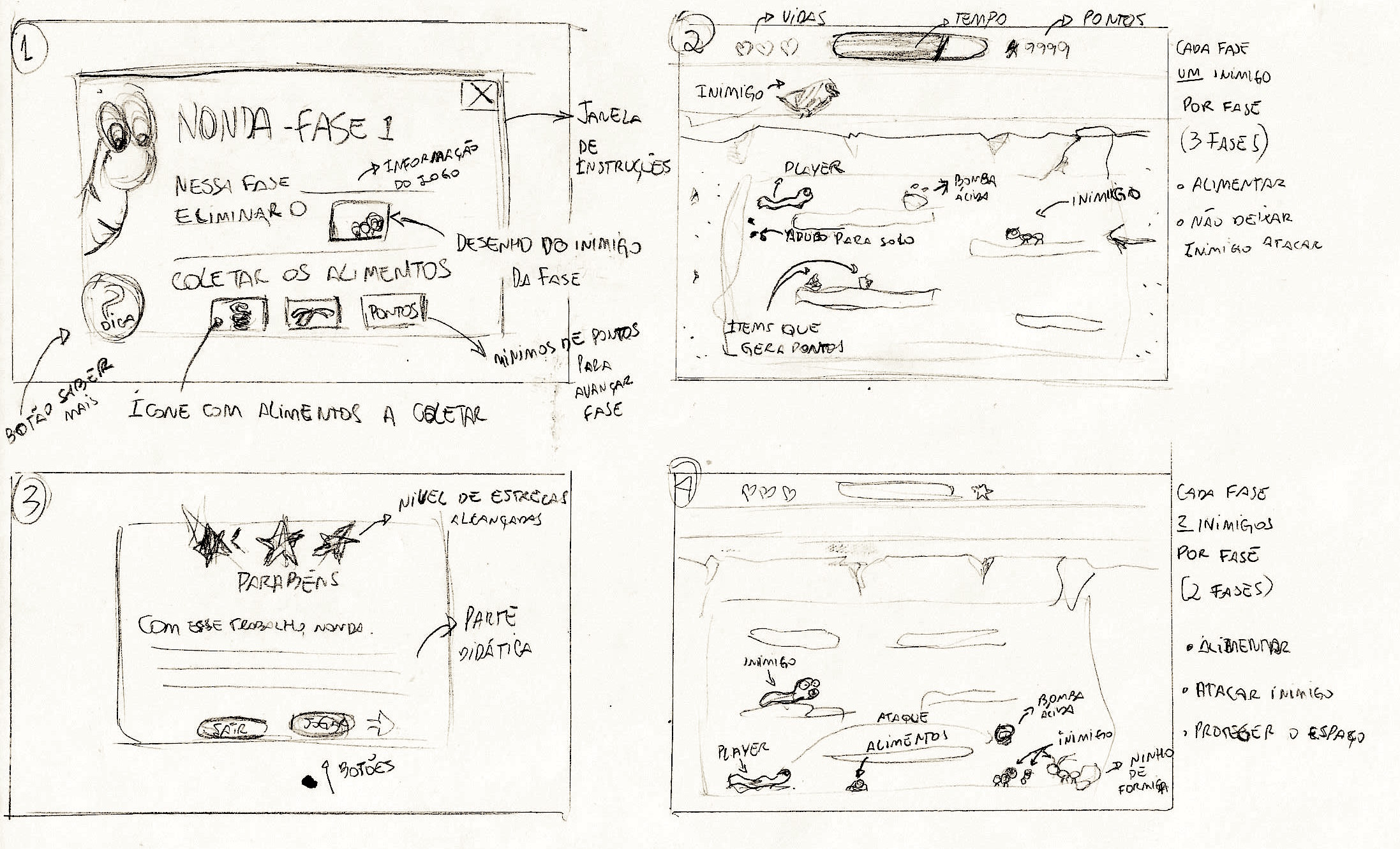
Na segunda fase, são jogados alimentos que pode e não pode comer, fazendo o jogador decidir qual é o correto. Novamente, precisa da pontuação mínima para avançar.

Da terceira fase à quinta fase, começa a fase dos inimigos. O personagem precisa se alimentar e proteger sua vida e o minhocário. A profª. Ana deposita alimentos na caixa. A partir da segunda fase, as Bombas Ácidas apresentam perigo ao inimigo que podem ser arremessadas, caso o jogador tenha adquirido. Outro perigo é o ataque dos inimigos. Os ataques entendem-se pelo fato do inimigo tocar no personagem. Cada toque, o personagem perde energia. Para repulsar os inimigos, precisa tocar seguidamente para afastá-los. É preciso defecar em uma área indicada na fase para poder adubar a terra, a fim de contabilizar os pontos.

A sexta fase é somente o personagem fugindo das bombas ácidas, durante um tempo pré-determinado. Para ajudar o personagem, o jogador precisa tocar na Bomba Ácida, quando estiver longe para fazer ela explodir.

Da sétima até a nona fase são apresentados o Ataque dos Inimigos. Eles vêm em *waves,* porém um inimigo para cada fase. Vai apresentar muitas Bombas Ácidas para auxiliar o jogador a vencer. Também terá o auxílio de *power-ups[[1]](#footnote-1)* como o Borro de Café. O jogador pode coletar esse item para restaurar 100% da sua vida.

A décima fase é composta por dicas sobre vermicompostagem, no formato múltipla escolha. Antes de cada fase ser concluída, o *Level Clear*[[2]](#footnote-2) contará também com uma questão múltipla escolha sobre o cenário em questão. Caso o jogador acerte, terá uma recompensa.



**Figura 5** – Nonda: Sketch do Storyboard representado através de telas

4.8 Gameplay – Perspectiva centrada no jogador

*Gameplay* é o elemento central do jogo, que para muitos jogadores determina a qualidade do jogo (DJAOUTI, D. et al). Em outras palavras é a relação de conjunto de atividades que o jogador pode fazer e como é a experiência do mesmo enquanto joga.

A Figura 6 ilustra o gameplay, que induzir o jogador a responder com ações aos elementos que interagem na tela, partindo da necessidade do personagem se alimentar, depois defendendo o ambiente dos predadores, além de defecar em uma área específica (para algumas fases), como forma de adubar a terra. Conta também com o elemento tempo, indicando a prioridade em determinadas ações para terminar determinadas fases.



**Figura 6** Tela de gameplay

4.9 Mecânica do Jogo

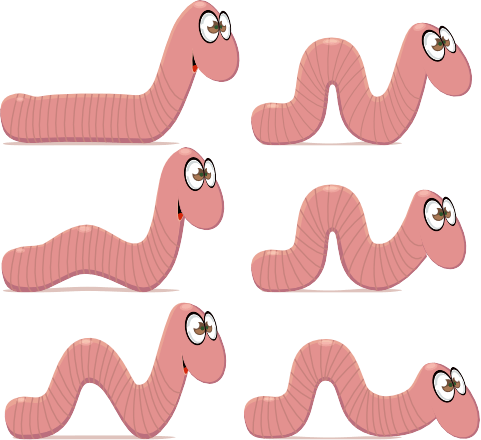
Atividade e Interatividade são dois conceitos que fundamenta o tópico anterior. De acordo com (BRUNER, 1972), qualquer atividade lúdica envolve a interação com objetos concretos ou abstratos. É cada parte do *gameplay* individualmente. A mecânica do jogo Nonda é a seguinte: evitar que a comida seja devorada pelos inimigos, usar toques para escapar de ataques dos inimigos e atirar bomba ácida, quando disponível. Uma outra que terá no jogo é a capacidade de chegar até uma comida antes que ela desapareça da tela.

4.10 Game Design

É o processo do jogo onde informa toda caraterística do jogo, incluindo controles, jogabilidade, interfaces, personagens, inimigos, fases e outros.

4.11 Personagem

Nonda tem 1 ano de idade, é uma minhoca (anelídeo), tem um tamanho padrão para uma minhoca. Nonda sempre sofreu com a invasão dos inimigos dentro da sua casa. Muitas vezes o solo estava quase pronto quando algum inimigo chegava e destruía tudo o que ela tinha umidificado. Tem personalidade calma, trabalhadeira e protetora e não possui poderes especiais. As ações dela dentro do jogo limitam-se em: andar, correr, pular, comer e arremessar bombas ácidas. Nonda defeca para adubar a terra. Abaixo, na Figura 7 mostra a personagem.



**Figura 7** –Personagem Nonda: Spritesheet com animação de Pular

4.12 Controle

Do personagem:

O personagem estará sempre se movendo, nunca parado. Para controlar, é necessário deslizar o dedo sobre tela (*slide*). Os controles disponíveis são:

Direcionais

• Deslizar horizontalmente no lado esquerdo da tela muda a direção do personagem

• Deslizar verticamente (de baixo para cima) o lado direito da tela, ativa o commando pular. Se deslizar 2x seguidas, o personagem ativará a ação de pulo duplo.

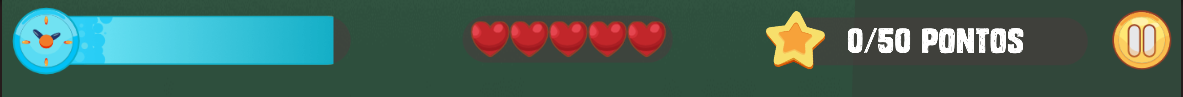
• Toque (várias vezes) em cima do inimigo para dar dano.

Do inimigo:

• Serão randômicos conforme a especialidade de cada um.

4.13 Interface

Na interface, o foco é no HUD – *head-up display*, termo utilizado para visualizar todos os elementos de interesse do jogador, tais como barra de progresso, tempo restante, quantidade de vidas, pontuações, indicações de localidade para atacar etc – do jogo, conforme ilustra a Figura 8 abaixo.



**Figura 8** – Nonda: HUD – Posicionamento do UI

4.14 Predadores

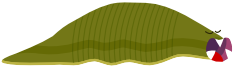
O predador no contexto do jogo são espécies distintas que aparecem para atrapalhar o trabalho de Nonda no decorrer do jogo.

4.14.1 Pássaros



**Figura 9** – Jogo Nonda: Inimigo Pássaros

4.14.2Sanguessuga



**Figura 10** – Jogo Nonda: Inimigo Sanguessuga

Fonte: Autoria Própria

4.14.3Formiga



**Figura 11** – Jogo Nonda: Inimigo Formiga

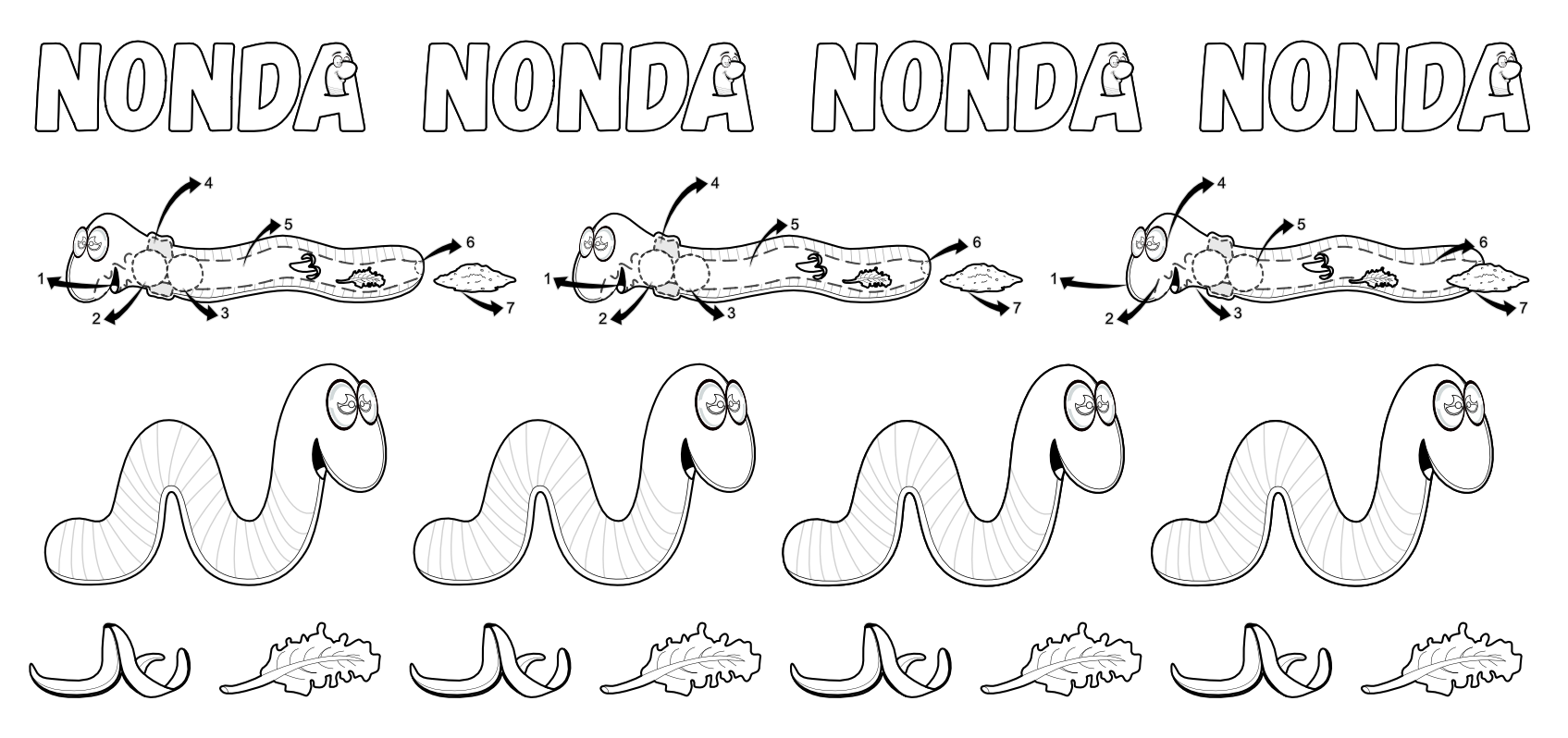
4.15 Level Design

O jogo vai se passar somente em um cenário, pois se trata de um ambiente de cativeiro da minhoca, que é característica básica da vermicompostagem. Tem a possibilidade desse cativeiro mudar de cor, conforme a qualidade do adubo.

4.16 Arte

A arte é baseada no minhocário da UTFPR, tendo apenas como plano de fundo a mudança de plataformas, conforme a fase. Foi escolhida o estilo *cartoon* para dar maior diversão ao jogador, pois não se sente na obrigação de estar em um simulador ou um jogo de primeira pessoa, por exemplo. Outro fator para escolha é pela habilidade artística adquirida com *cartoons* ao longo dos anos.

Com esse estilo de arte, foi possível expandir o design do jogo, transferindo para o papel, no formato de colorir como mostra a Figura 12. Essa expansão foi fundamental para ensinar as crianças do ensino fundamental durante as atividades sobre o tema.



**Figura 12** - Desenho para Colorir

1. DESENVOLVIMENTO

Até o tópico 5, este trabalho concentrou-se em mostrar a fundamentação teórica seguida do anexo *Game Design Document* para entender aspectos de criação do jogo Nonda e as ferramentas que faz parte do processo de desenvolvimento. Nesse capítulo será abordado o processo de construção, incluindo diagramas, trechos de códigos e validações.

Primeiro, é necessário mostrar o storyboard que resulta na sequência em que o jogo é jogado.

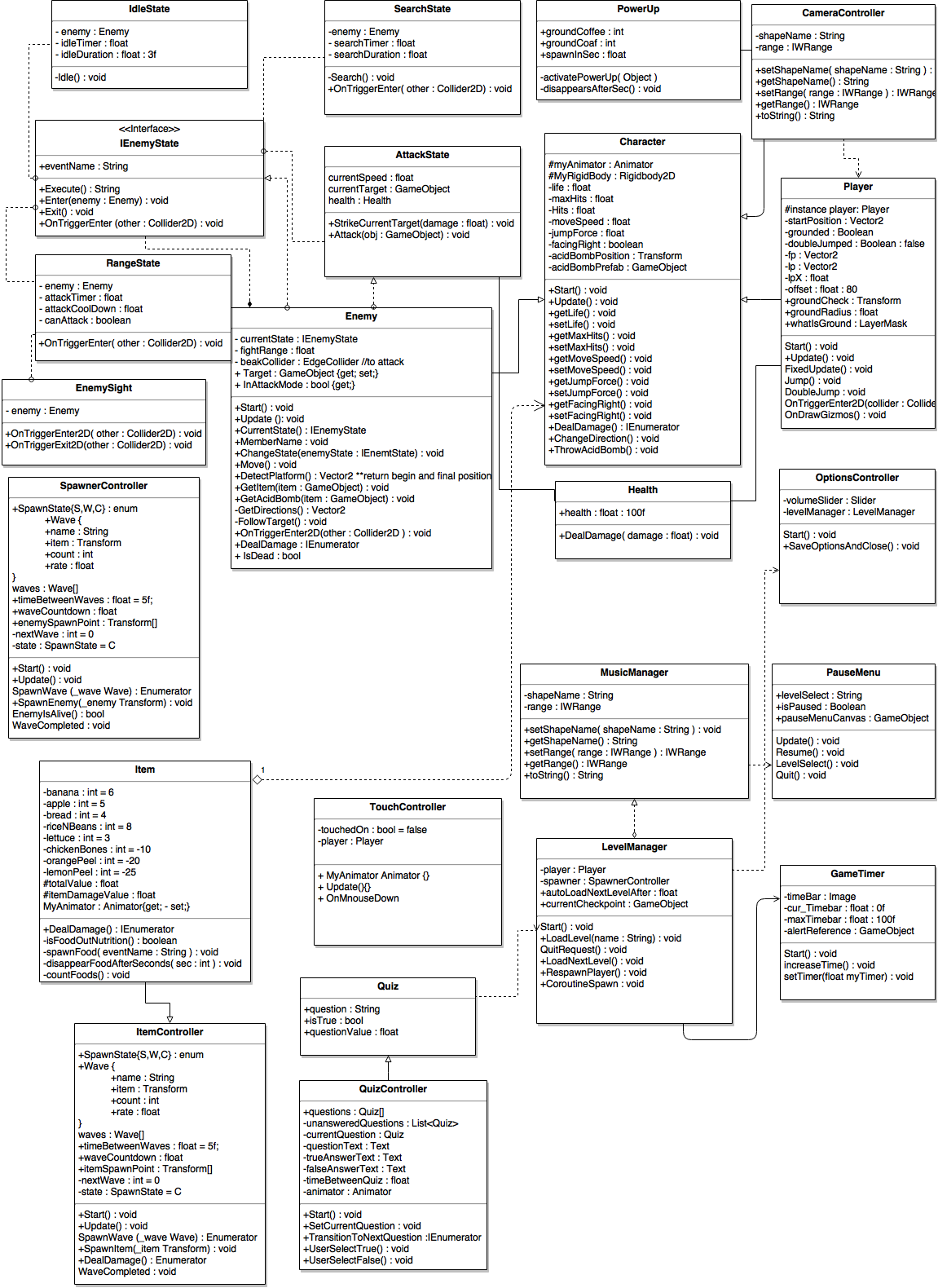
* 1. **Diagrama de Sequência**

No contexto do jogo Nonda, o diagrama de sequência narra o fluxo do jogo, deixando claro quais cenas necessitarão de quais elementos e em qual ordem. Na Figura *13*, primeiro é mostrado ao usuário a tela principal do jogo, seguido da tela de tutorial (que se repete a cada fase). Logo após surge a tela do jogo para o usuário interagir, pontuar e avançar de fase. É importante mencionar que existem telas que podem ser acessadas durante o jogo como o Tutorial ou o Menu de Opções. Seguindo o fluxo do jogo, a próxima tela informa a performance do jogador na fase específica (também conhecido como *End Screen*) que dá acesso a uma próxima fase, jogar novamente a fase, caso não pontuar suficiente ou ao menu principal. Essa sequência do storyboard faz o *game design* fluir organicamente. [TROCAR FIGURA DO DIAGRAMA DE SEQUENCIA]

**Figura 13** - Diagrama de Sequência

* 1. **Diagrama de Classe**

Para atingir esse resultado, foi desenvolvido o diagrama de Classe segundo o padrão de projeto MVC como mostrado na Figura 14. Pode ser observado que a Classe *Character* é a base tanto para implementar o jogador quanto para o predador. Outra classe importante a citar é o *Spawner* *Controller*, que foi implementada para dar base às outras como spawnar [[3]](#footnote-3)itens coletados e *spawnar* predadores.



**Figura 14** – Diagrama de Classe

* 1. **Fluxograma de Animação**

A personagem Nonda possui diversos estados de animação, de acordo com um comportamento gerado pelo *gameplay* ou pela intervenção do usuário através do toque na tela. São atribuídos cinco estados diferentes que Nonda: Andar, Correr, Pular, Cair, Pousar. Esses estados de animação são definidos abaixo, na Figura 15. Para mudar de estado, algum outro estado precisa intervir em tempo de execução do jogo, no método Update(), que é chamado a cada *frame*[[4]](#footnote-4). O estado de animação padrão da Nonda é Andar. Isso significa que sempre a velocidade é maior do que 0 e, se o jogador executar ações do tipo pular ou mudar de direção, implicará mudança de estado de animação resultando em conjunto de sprites totalmente diferente para que represente tal estado de animação.

/Users/Elias/Downloads/Animation_flowchart_Player.png

**Figura 15** – Fluxograma Animação do personagem Nonda (*Player*)

[Escrever baseado na imagem ]

* 1. **Codificação**

No presente capítulo, foca-se na prática de desenvolvimento, usando a linguagem C#, tendo subtópicos para códigos que implementaram função descritas no Diagrama de Classe usando o padrão de projeto MVC.

* + 1. **Coroutines**

Coroutine (Co-rotina, em português) é uma função da própria *game engine* que fornece execução de uma tarefa sequencial (sem divisão de processador). Coroutine funciona da seguinte forma: a função salva o último ponto que parou, e no próximo frame executa o restante do código. O ganho em relação às *threads* (*threads* ou *multithreads* são funções que usam sincronia de várias funções distintas para executar animações, colisões, entradas fornecidas pelo usuário, entre outros) é que não exige alto processamento computacional, especialmente em caso de dispositivo móvel com baixa memória. O Unity 3D usa a interface IEnumerator para implementar Coroutine, isso significa que une duas interfaces (IEnumerator e IEnumerable) para que possa numerar, através de um cursor interno para o índice atual e fazer o gerenciamento para o ponto real eliminando verificações que espendem memória. É usado a declaração *yield* como marcador para que continue após essa declaração na próxima vez que for chamado. Semelhantemente ocorre com a declaração *return*. Abaixo, na Figura 16 pode-se observar o código extraído da classe ItemController.cs que tem adiciona itens na tela em forma de *waves*[[5]](#footnote-5). No método *SpawnWave*(Wave *wave) dentro do um laço de repetição for encontra-se uma chamada para o método SpawnItem(\_wave.item*) que adiciona na jogo a quantidade de items definidos pelo tamanho do laço de repetição. Logo após, é chamada a declaração *yield* que retorna uma função *WaitForSeconds(1f/rate),* esperando um tempo passado entre parâmetros para dar continuidade na execução do método. Logo após sair do laço, o Enum muda de estado da função para *Waiting* e passa a executar o que vem após a declaração *yield*.



**Figura 16** - Código usando Coroutines

Do ponto de vista de programação, Coroutines encurtou o passo de criar várias classes dependentes, solidificou essas dependências apenas criando métodos específicos. Outra contribuição importante foi a capacidade de eliminar processamento de memória (dito anteriormente) com a habilidade de pausar a execução e retornar o controle para o Unity, e quando volta a execução continua a partir do ponto onde foi deixado.

Coroutines foi a base para implementar todas classes que envolve criação de objetos em tempo de jogo como as classes *ItemController.cs*, *SpawnerController.cs* que representa os Itens e Predadores, respectivamente. Porém, existe uma classe que é exceção: ItemSpawner.cs. Essa classe deposita os itens não saudáveis em tempos aleatórios e faz desaparecer independente se todos foi tocado pelo jogador ou não. A classe traz um conceito simples de inteligência artificial para jogo, embora o desenvolvimento não tem foco nessa área. Na Figura *17*, o trecho do código da classe em questão verifica se é hora de spawnar novos objetos na cena, caso seja positivo é chamado o método *isTimeToSpawn()*. Esse método verifica se existe algum item na cena e se ainda pode ser depositado item. Caso seja positivo, é executado o método *SpawnWave()* com parâmetros necessários.



**Figura 17** - Trecho de código da classe ItemSpawner

* + 1. Modelo e Controle para Predadores

Como o jogo possui várias fases e é alimentado por diferentes predadores, é necessário implementar uma classe modelo para predadores que seja dinâmico e fácil de controlar. O modelo desenvolvido apresenta a classe *Character* que é base tanto para o implementar a personagem Nonda quanto os predadores. Através da herança, foi criado classes de controller para implementar os estados de animação que incluem: *Idle, Range, Search e Attack*. O estado *Idle* deixa o predador sem movimento (inativo) por alguns segundos. O estado Search busca pelo jogador baseado no estado *Range* que verifica se existe o jogador próximo dele para que possa habilitar o estado Attack, que efetiva o estado de atacar o jogador.

O Controller dos predadores tem estrutura para construir inúmeras *waves* de predadores diferentes, isso através do trecho de código representando na Figura 18.



**Figura 18** - Código que estrutura criação de Waves

A classe Wave possui quatro parâmetros que são setados dentro do Unity Editor e são armazenados dentro de cada posição do array *Wave[] waves*. Com isso a criação e adição de novos predadores fica flexível além de facilitar na manutenção e refatoração do código em versões futuras.

* + 1. Sound Manager com array dinâmico

CRONOGRAMA

Para a execução do projeto proposto, serão realizadas as atividades relacionadas conforme a Figura 19. As atividades foram desenvolvidas por semanas, conforme a Metodologia Scrum solo. Para maior compreensão e separação das atividades, será criado 4 fases no processo de desenvolvimento. A primeira fase endo a primeira a criação de um Game Design Document (GDD), seguido do estudo e compreensão do tema – teorias e metodologias a serem aplicadas – e, a terceira a implementação, junto com o design. Completa todas etapas, o jogo será testado pela equipe da UTFPR para validação e então retornar para a última etapa, que é o refinamento e alterações/correções de erros de código.



**Figura 19** - Cronograma completo de atividades semanais

1. CONSIDERAÇÕES FINAIS

Espera-se que esse trabalho contribuía de forma positiva para o Programa de Recursos Educacionais Digitais e fortifique o laço com Acordo de Cooperação Técnicas entre a UTFPR Câmpus Curitiba e as secretarias de Educação dos municípios de Piraquara, São José dos Pinhais, Pinhais e Curitiba para promover o jogo desenvolvido juntamente com a criação de cartilhas e conteúdos digitais disponibilizados gratuitamente, em específico, às instituições de ensino. Espera-se também que esse trabalho traga uma reflexão na quantidade de resíduos sólidos descartados de forma irregular que pode ter um destino adequado se usado a tecnologia de compostagem. O resultado desse projeto também será incorporado pelo Programa Jogada Certa – coleta Seletiva da UTFPR, como recurso de capacitação de funcionários dos serviços gerais no processo de gestão de resíduos sólidos.

O propósito do presente trabalho foi distribuir um jogo de qualidade que integrasse a abordagem do conteúdo aprendido com a tecnologia de ensino para que além de familiarizar os alunos no meio de recursos educacionais digitais instigasse uma progressão lógica de aprendizado e um autocontrole.

Embora o trabalho tenha demonstrado que é importante saber quais resíduos sólidos orgânicos separar e reaproveitá-los dentro do minhocário, existem limitações em termos técnicos como não mostrar o processo de construção de um vermicomposteira, a escolha de qual espécie de minhoca é apropriada para a criação de adubos, os cuidados para manter o minhocário livre de mal odor e de predadores. Em termos tecnológicos a limitação reside no fato de o aplicativo final ser distribuído apenas para dispositivo Android.

**REFERÊNCIAS**

Abt, Clark C. *Serious games*. University Press of America, 1987.

ALVES, LYNN R. G, MINHO, MARCELLE R. S, DINIZ, MARCELO V. C. Pimenta Cultural 2014. **Gamificação: diálogo com a educação**. Disponível em: http://www2.dbd.puc-rio.br/pergamum/docdigital/PimentaCultural/gamificacao\_na\_educacao.pdf Acesso em: 23 fev. 2016 4:30

Bigg Shark. Why Using C# with Unity is Better Than Using Boo or JS for Your Mobile Game, 2015.Disponível em: http://biggshark.com/why-using-c-with-unity-is-better-than-boo-and-js-for-your-next-mobile-game/ Acesso em: 03 fev.2016 16:35

Bruner, J. S. (1972), “Nature and uses of immaturity”, American Psychologist, Vol. 27, No. 8,. In Bruner, J. S., Jolly, A. and Sylva, K. (eds.) (1976), Play. Its role in development and evolution. Penguin Books, New York.

Corti, K. (2006) Games-based Learning; a serious business application. PIXELearning Limited. [www.pixelearning.com/docs/games\_basedlearning\_pixelearning.pdf] Acesso em: 04 mar.2016 09:35

COSTA, E. DIAS. Toptal Developers. **Unity with MVC: How to Level Up Your Game Development** Disponível em: http://www.toptal.com/unity-unity3d/unity-with-mvc-how-to-level-up-your-game-development Acesso em: 24 fev. 2016 14:40

DJAOUTI, D. et al. “A Gameplay Definition through Videogame Classification,” International Journal of Computer Games Technology, vol. 2008, Article ID 470350, 7 pages, 2008. doi:10.1155/2008/470350

FABRI, J. A. et al. “SCRUM SOLO”, disponível em: <https://www.scrumsolo.wordpress.com/>, Acesso em: 03 mar 2016.

FABRICATORE, C. **Gameplay and Game Mechanics Design**. Gameplay and Game Mechanics Design: A Key to Quality in Videogames. Disponível em: http://www.oecd.org/edu/ceri/39414829.pdf Acesso em: 04 mar. 2016 17:41

Innovation House Rio (IHR). Disponível em: https://innovationhouserio.wordpress.com/2015/09/03/brazilian-gaming-studios/ Acesso em: 22 jan. 2016 16:05.

JAMES, M. **Scrum Reference Card**. Disponível em: http://scrumreferencecard.com/scrum-reference-card/ Acesso em: 24 fev. 2016 15:20

KANSTENSMIDT, C. Revistas eletrônicas. **Os impactos das Tecnologias dos Jogos Digitais Multijogadores na Jogabilidade Social**. Disponível em: http://revistaseletronicas.pucrs.br/ojs/index.php/famecos/article/view/7789/5531 Acesso em: 23 fev. 2016 5:45

LEMES, David de Oliveira. ABRELIVROS. **Artigo: Serious Games – Jogos e Educação**. Disponível em: http://www.abrelivros.org.br/home/index.php/bienal-2014/resumos-e-fotos/5647-primeiro-resumo Acesso em: 23 fev. 2016 15:11

Lévy, P. (1999) “Cibercultura”, São Paulo SP ed.34.

LOURENCO, NELSON. **Manual de Vermicompostagem e Vermicultura para Agricultura Orgânica**, 2014. Disponível em: https://goo.gl/9sNeiW Acesso em: 04 fev. 2016 13:15

MASOVER, S. IST-SIS. 2014. **Model-View-Controller:  
A Design Pattern for Software** Disponível em: https://ist.berkeley.edu/as-ag/pub/pdf/mvc-seminar.pdf Acesso em: 24 fev. 2016 12:00

MURRAY, J. Hamlet no Holodeck: o futuro da narrativa no ciberespaço. São Paulo: Itaú Cultural, 2003.

NDEGWA, P. M., and S. A. THOMPSON. "Integrating composting and vermicomposting in the treatment and bioconversion of biosolids." *Bioresource technology* 76.2 (2001): 107-112.

NUERNBERG, ANA CLAUDIA**.** Vermicompostagem: estudo de caso utilizando resíduo orgânico do restaurante universitário da UTFPR Câmpus Curitiba/Sede Ecoville. Disponível em: http://repositorio.roca.utfpr.edu.br/jspui/handle/1/3911 Acesso em: 23 jan.2016 11:30.

Panorama dos Resíduos Sólidos no Brasil, 2015. Disponível em: http://www.abrelpe.org.br/Panorama/panorama2014.pdf Acesso em: 21 jan.2016 17:15.

PRENSKY, M. Digital Game-Based Learning. St. Paul: Paragon House, 2001.

PRENSKY, M[. Digital Game-Based Learning, McGraw-Hill Pub. Co., 2004](http://dl.acm.org/citation.cfm?id=1209307&CFID=852500841&CFTOKEN=24195620)

SAMPAIO, H. **Jogos casuais tomam conta da Indústria**. Disponível em: http://jogos.uol.com.br/reportagens/ultnot/2008/06/24/ult2240u131.jhtm Acesso em: 02 mai.2016 17:31

SCRUM SOLO. Visão Geral, 2015. Disponível em: https://www.scrumsolo.wordpress.com/ Acesso: 18 abril, 2016 12:10

SPECOUT BY GRAPHIQ, 2014. QUALCOMM SNAPDRAGON MSM 8255. Disponível em: http://system-on-a-chip.specout.com/l/264/Qualcomm-Snapdragon-MSM8255 Acesso: 10 fev. 2016 22: 02

STEUER, J. Department of Communication, Stanford University. 1993. **Defining Virtual Reality: Dimension Determining Telepresence**. Disponível em: http://www.cybertherapy.info/pages/telepresence.pdf Acesso em: 23 fev. 2016 23:10

TAROUCO, L. M. R. ; FABRE, Marie-Christine Julie Mascarenhas ; ROLAND, Letícia Coelho; KONRATH, Mary Lúcia Pedroso . **Jogos educacionais**. RENOTE. Revista Novas Tecnologias na Educação, Porto Alegre, v. 2, n. 1, p. 1-7, 2004.

**The Best Development Platform for Creating Games**. Disponível em: https://unity3d.com/unity Acesso em: 03 fev.2016 16:00.

UNITY (Game Engine). Disponível em: http://research.omicsgroup.org/index.php/Unity\_(game\_engine). Acesso em: 12 setembro, 2016. 21:40.

YONGLEI Tao, "Component- vs. application-level MVC architecture," Frontiers in Education, 2002. FIE 2002. 32nd Annual, 2002, pp. T2G-7-T2G-10 vol.1.

1. Como mecânica de jogo, são Itens que beneficiam ou adicionam habilidades extras instantânea ao jogador [↑](#footnote-ref-1)
2. Tela que disponibiliza os pontos e outros recursos coletados no jogo, ressaltando a vitória ou derrota na fase. [↑](#footnote-ref-2)
3. spawn - [↑](#footnote-ref-3)
4. frame: É a capacidade máxima que o processador consegue reproduzir imagens dentro de 1 segundo. A essa denominação, chama se *frames per second* ou em português frames por segundo. [↑](#footnote-ref-4)
5. *waves* - é o termo usado para caracterizar um grupo de inimigos, itens, power-ups que vem em um certo tempo em uma quantidade pré-determinada durante o gameplay de um jogo. [↑](#footnote-ref-5)