FACULDADE DE TECNOLOGIA DE AMERICANA
Curso Superior de Tecnologia em Jogos Digitais

Leonardo Millan de Souza Pinto

GRIDALIS

Americana, SP
2017

FACULDADE DE TECNOLOGIA DE AMERICANA
Curso Superior de Tecnologia em Jogos Digitais

Leonardo Millan de Souza Pinto

GRIDALIS

Relatório técnico desenvolvido em cumprimento à exigência curricular do Curso Superior de Tecnologia em Jogos Digitais sob a orientação do Prof. Jonas Bodê

Americana, SP
2017

FICHA CATALOGRÁFICA – Biblioteca Fatec Americana - CEETEPS
Dados Internacionais de Catalogação-na-fonte

P728g PINTO, Leonardo Millan de Souza

Gridalis./ Leonardo Millan de Souza Pinto. – Americana: 2017.

27f.

Monografia (Curso de Tecnologia em Jogos Digitais) - - Faculdade de Tecnologia de Americana – Centro Estadual de Educação Tecnológica Paula Souza

Orientador: Prof. Esp.Jonas Bodê

1. Jogos eletrônicos 2.Realidade aumentada 3. Android - aplicativos I. BODÊ, Jonas II. Centro Estadual de Educação Tecnológica Paula Souza – Faculdade de Tecnologia de Americana

CDU: 681.6
681.519

Leonardo Millan de Souza Pinto

GRIDALIS

Trabalho de graduação apresentado como exigência parcial para obtenção do título de Tecnólogo em Jogos Digitais, pelo CEETEPS/Faculdade de Tecnologia – Fatec/ Americana.
Área de concentração: Desenvolvimento de Jogos

Americana, 26 de junho de 2017.

Banca Examinadora:

Jonas Bodê (Presidente)
Especialista
Fatec Americana

Rodrigo Nogueira Tofani (Membro)
Especialista
Fatec Americana

Benedito Aparecido Cruz (Membro)
Graduado
Fatec Americana

RESUMO

Um relatório sobre o processo de desenvolvimento de um jogo RPG procedural para a plataforma Android utilizando localização GPS para movimento do jogador. Descrevemos o processo usado na criação e a implementação dos principais sistemas e mecânicas de jogabilidade do jogo, e também analisamos o resultado final do projeto e recepção do jogo pelos jogadores após o lançamento no mercado.

Palavras Chave: RPG; Realidade aumentada; Android

ABSTRACT

A report regarding the process of development of a procedural RPG game for the Android platform using GPS location for player movement. We described the process used on the creation and implementation of the game main systems and gameplay mechanics, and also we analyzed the project's final results and reception of the game by the players after the release on the market.

Keywords: *RPG; Augmented reality; Android*

LISTA DE FIGURAS

Figura 1 - Geração do input para o algoritmo procedural de criação do mapa 14
Figura 2 - Diagrama do *shader* de sprites de monstros 15
Figura 3 - Diagrama de sequência da execução do sistema de batalha.....17
Figura 4 - Gráfico de instalações por usuários ao longo de 30 dias.....20
Figura 5 - Gráfico de avaliações por usuários na Play Store 21
Figura 6 - Listagem dos rankings da categoria RPG na Play Store 21
Figura 7 - Screenshot da criação de personagem.....22
Figura 8 - Screenshot da abertura do jogo (tutorial).....22
Figura 9 - Screenshot da tela principal do jogo 23
Figura 10 - Screenshot da tela de forja de equipamentos.....23
Figura 11 - Screenshot da tela de batalha 24

LISTA DE QUADROS

Quadro 1 - Estados do fluxo de batalha 16
Quadro 2 - Fatores de decisão para AI de inimigos em batalha 18

SUMÁRIO

1 INTRODUÇÃO 9
2 METODOLOGIA 10
2.1 Ferramentas de Desenvolvimento 10
2.2 Equipe do Projeto 11
2.3 Cronograma 11
2.4 Conceito do Jogo 11
3 IMPLEMENTAÇÃO 13
3.1 Mapa do Jogo 13
3.2 Sistema de UI 14
3.3 Sistema de Batalha 15
3.4 Inteligência Artificial 17
3.5 Multiplayer 18
3.6 Plugins 19
4 RESULTADOS 20
5 CONSIDERAÇÕES FINAIS 25
REFERÊNCIAS BIBLIOGRÁFICAS 26
APÊNDICE 1 – CÓDIGO FONTE 27

1 INTRODUÇÃO

Nesse relatório será documentado todo o processo de desenvolvimento do jogo Gridalis, um RPG de realidade aumentada baseado em GPS, inspirado na popularidade do jogo Pokémon GO da empresa Niantic.

Pokémon GO foi lançado em Julho de 2016 e rapidamente se tornou um marco dos jogos *mobile*, estabelecendo cinco recordes no Guinness World Records e atingido um status de fenômeno cultural. Apesar de seu lançamento restrito a um número limitado de países, ele se tornou o jogo com maior lucro do mercado de jogos mobile, rendendo um total de $206.5 milhões apenas no primeiro mês de lançamento. Desde então, o jogo foi baixado 130 milhões de vezes, e em pouco mais de seis meses atingiu um faturamento de $1 bilhão (SWATMAN, 2016).

Apesar do seu sucesso, Pokémon GO foi inicialmente recebido com diversas críticas, devido a falhas de infraestrutura que impediram o acesso de jogadores nas primeiras semanas do lançamento e a falta de funcionalidades que haviam sido prometidas antes do lançamento. A Niantic anunciou a inclusão de algumas destas funções apenas em Junho de 2017, quase um ano após o lançamento original do jogo (RODGER, 2017).

O objetivo de Gridalis foi criar um RPG que utilizasse da premissa básica de realidade aumentada do Pokémon GO e fornecesse, em cima dessa estrutura, um *gameplay* novo e diferenciado, com mecânicas que remetessem a RPGs tradicionais e que fornecessem aspectos não existentes no original, como batalhas cooperativas entre os jogadores. Um objetivo adicional do projeto a foi de fornecer essa experiência sem que fosse necessário conexão com a internet, permitindo que qualquer usuário possa jogar desde que tenha um dispositivo móvel equipado com GPS.

2 METODOLOGIA

2.1 Ferramentas de Desenvolvimento

Para o desenvolvimento deste projeto foi escolhido a ferramenta GameMaker: Studio, uma *engine* de jogos criada pela empresa escocesa YoYoGames voltada para jogos 2D, por já possuir vasta experiência com a mesma e pela possibilidade de exportar para a plataforma Android. A ferramenta utiliza a sua própria linguagem de programação baseada em C chamada GML (*Game Maker Language*), e permite o uso e a criação de *plugins* escritos nativamente para a plataforma a qual o jogo será compilado. Neste projeto utilizou-se a edição Professional da ferramenta na versão v1.99.548, com a extensão de exportação para Android.

A programação dos *plugins* da *engine* que foram utilizados no jogo foi feita com o *software* Sublime Text 3, um editor de códigos que possui suporte a uma vasta gama de linguagens de programação. O software também foi usado para a criação dos arquivos JSON que definem os bancos de dados e elementos de interface do jogo.

Foi utilizado o software Google Sheets para o *game design* e criação das planilhas de balanceamento dos parâmetros e mecânicas do jogo, que faz parte do serviço Google Drive.

Também foi usado Adobe Photoshop para criação de *concept arts*, *mockups* de interfaces e ilustração de personagens, e Audacity e FL Studio para engenharia de som.

Para testes na plataforma Android foi utilizado um *smartphone* Samsung Galaxy S5, um *smartphone* Sony Xperia M2 e um *tablet* QBEX TXM785i.

2.2 Equipe do Projeto

O projeto foi inteiramente desenvolvido por apenas uma pessoa, sendo esta responsável pelo conceito, planejamento e criação de todos os aspectos do jogo.

2.3 Cronograma

O planejamento do jogo foi iniciado no dia 4 de Abril. Nesta etapa foi criado o conceito e escopo do jogo, definindo as mecânicas e regras principais que formariam o *gameplay*.

A partir do dia 11 de Abril foi iniciado a etapa de produção do projeto. Durante esse período, alternava-se entre atividades de programação, arte, *game design* e teste, a medida que fosse necessário.

O desenvolvimento foi encerrado no dia 27 de Maio, quando também foi realizado a publicação do jogo na Google Play Store.

2.4 Conceito do Jogo

Em Gridalis, o jogador poderá navegar por um mapa virtual composto de uma grade de regiões geradas proceduralmente, sendo que cada região corresponde a uma área mapeada ao mundo real. Dentro de cada região ele pode encontrar uma *dungeon* onde é possível enfrentar inimigos e desafios para adquirir ouro, itens e experiência.

As mecânicas de progressão do jogador e de batalha são baseadas nas mecânicas usadas em RPGs tradicionais, como Dragon Quest e Final Fantasy. O jogador pode escolher uma classe que determina suas propriedades de batalha e pode adquirir experiência que permite subir de nível e ficar mais poderoso.

O jogador pode customizar seu avatar no jogo por meio da criação de itens e equipamentos. Para isso, ele usa materiais encontrados nas regiões do mapa ou conquistados nas *dungeons* ao derrotar inimigos. Com estes materiais, ele produz itens por meio de um sistema de alquimia ou forja armas e armaduras através de um sistema de forja.

O jogo também permite a interação entre jogadores em partidas *multiplayer*, permitindo batalhas cooperativas contra os inimigos do jogo.

Por fim, o jogo possui o diferencial de poder ser jogado sem conexão com a internet, desde que o jogador possua um dispositivo com suporte a GPS. Esse fator foi essencial para delimitar o escopo e a implementação das mecânicas do jogo, tornando necessárias soluções não-triviais para prover a experiência que foi esperada para o jogador.

3 IMPLEMENTAÇÃO

3.1 Mapa do Jogo

O mapa do jogo é o sistema mais importante de Gridalis, uma vez que o jogo se baseia na navegação do mapa por meio do GPS, e o conteúdo encontrado pelo jogador depende da funcionalidade deste sistema.

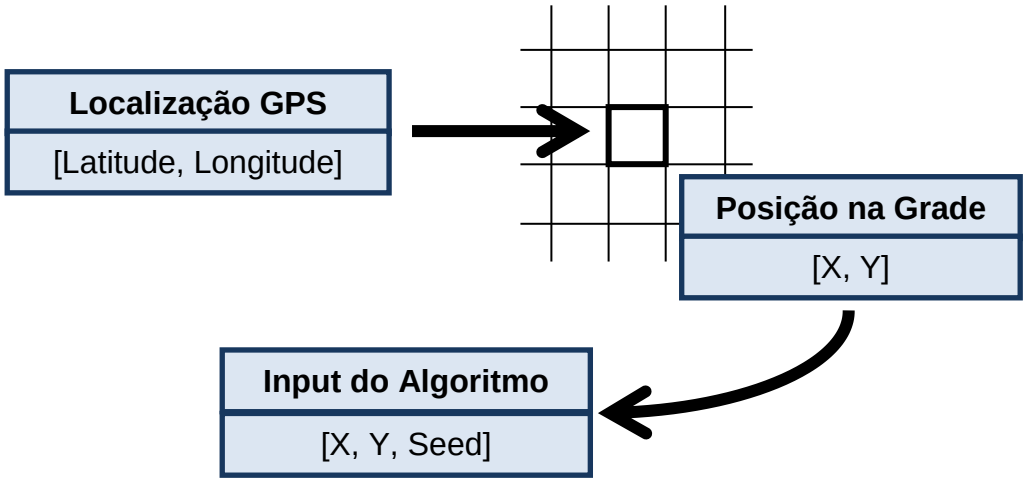
Uma das mecânicas planejadas para o jogo foi a de partidas *multiplayer*. Por isso, se dois jogadores estiverem em um mesmo local no mundo real, âmbos devem encontrar a mesma região no jogo, com os mesmos elementos e mesmos inimigos.

Em Pokémon GO, o conteúdo do mapa é enviado por um servidor para os jogadores, de acordo com a sua posição geográfica. Mas, devido a restrição de não utilizar conexão com a internet neste projeto, foi necessário desenvolver uma outra forma de garantir que todos os jogadores visualizassem o mesmo mapa virtual do jogo.

Para delimitar as regiões do mapa do jogo, o mapa do mundo foi dividido em uma grade de 36 mil por 18 mil células, resultando em um total de 648 milhões de regiões, cada um mapeado a uma área de aproximadamente 1km². Criar individualmente cada uma destas regiões seria uma tarefa impossível e resultaria em um pacote de mais de 1GB de informação, o que inviabiliza o armazenamento destes dados dentro no jogo.

Para solucionar esse problema, foi utilizado algoritmos procedurais que geram as regiões do mapa do jogo dinâmicamente, a medida que o jogador encontra as regiões. Os algoritmos recebem a longitude e a latitude do jogador, providas pelo GPS do dispositivo móvel, e determina qual célula corresponde a esta localização. A posição é então transformada em um input para o algoritmo procedural, que determina todas características desta região e os elementos que a compõem.

Figura 1 - Geração do input para o algoritmo procedural de criação do mapa



Fonte: Próprio autor

Diversas planilhas foram criadas para realizar o balanceamento dos parâmetros usados pelo algoritmo procedural na geração das regiões do mapa do jogo. Isso permitiu realizar experimentos e testes com o algoritmo que seriam impossíveis de outra forma devido a imensa quantidade de regiões existentes no jogo.

3.2 Sistema de UI

A *engine* GameMaker: Studio não possui um sistema de *User Interface* padrão. O seu sistema de entidades é projetado com as necessidades de jogos com interações em tempo real em mente, o que não é ideal para um jogo com uso extensivo de interfaces e batalhas por turno. Por isso, foi projetado um sistema de UI para o jogo, que fosse mais intuitivo e compatível com as necessidades previstas nos *mockups* de interfaces do jogo.

O sistema de UI esquematizado para Gridalis foi baseado em componentes e eventos, permitindo elaborar e carregar os componentes definidos em arquivos JSON, assim sendo possível uma composição modular e intuitiva da interface. Nos arquivos JSON, é possível determinar eventos a serem disparados pela interação do jogador, que são posteriormente tratados por *scripts* declarados no código do jogo.

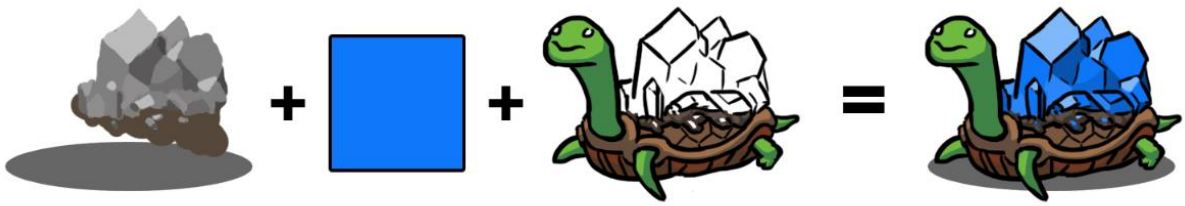
O sistema suporta aninhamento de componentes. Os componentes podem disparar eventos que são propagados pela hierarquia de componentes a qual pertence, até que um *handler* processe o evento e termine a sua propagação.

3.3 Sistema de Batalha

O sistema de batalha do jogo é baseado nas batalhas de turno encontrados em RPGs tradicionais. Na batalha, o jogador enfrenta um grupo de monstros usando ataques, magias e itens. O jogador deve eliminar os inimigos, reduzindo sua vida a 0, para vencer a batalha. Se a sua própria vida chega a 0, o jogador é derrotado.

Cada unidade de batalha, seja o jogador ou um inimigo, possui seis atributos de batalha. Os inimigos também possuem uma posição, seja na linha de frente ou na retaguarda, e um elemento que representa a sua força e fraqueza mágica, representado como uma tonalidade aplicada a parte do sprite do monstro. Para desenhar o sprite final dos monstros, foi criado um *shader* que combina um *sprite* base com a tonalidade do elemento aplicada com um *sprite* sobreposto.

Figura 2 - Diagrama do *shader* de sprites de monstros



Fonte: Próprio autor

Os atributos de batalha são:

- **Health Points (HP):** pontos de vida. Quando chega a zero, a unidade é incapacidade e não pode mais agir.
- **Attack (ATK):** poder de ataque físico. Usado no cálculo de dano de ataques físicos.
- **Defense (DEF):** resistência a ataques físicos. Reduz a quantidade de dano físico que a unidade recebe ao ser atacada.

- **Intelligence (INT):** poder de ataque mágico. Usado no cálculo de dano de ataques mágicos. É combinado com um fator multiplicador referente a efetividade do elemento da magia utilizada contra o alvo.
- **Resistance (RES):** resistência a ataques mágicos. Reduz a quantidade de dano mágico que a unidade recebe ao ser atacada.
- **Speed (SPD):** determina a ordem em que as unidades executam os seus comandos em batalha.

A posição dos inimigos determina quais alvos o jogador pode atacar. Apenas os inimigos da linha de frente podem ser atingidos com ataques físicos. O jogador pode usar certas magias e itens que permitem atingir inimigos na retaguarda. Quando todos os inimigos da linha de frente são derrotados, os inimigos restantes se aproximam, e passam a ser considerados inimigos da linha de frente.

O sistema de batalha foi projetado para suportar um modo *multiplayer* com até quatro jogadores. Para isso, o controlador do sistema de batalha foi separado em dois objetos: “servidor” e “cliente”.

O sistema de batalha possui quatro estados, nos quais o servidor e os clientes realizam uma operação determinada, podendo compartilhar dados através do evento (quando do servidor) ou sinal (quando do cliente).

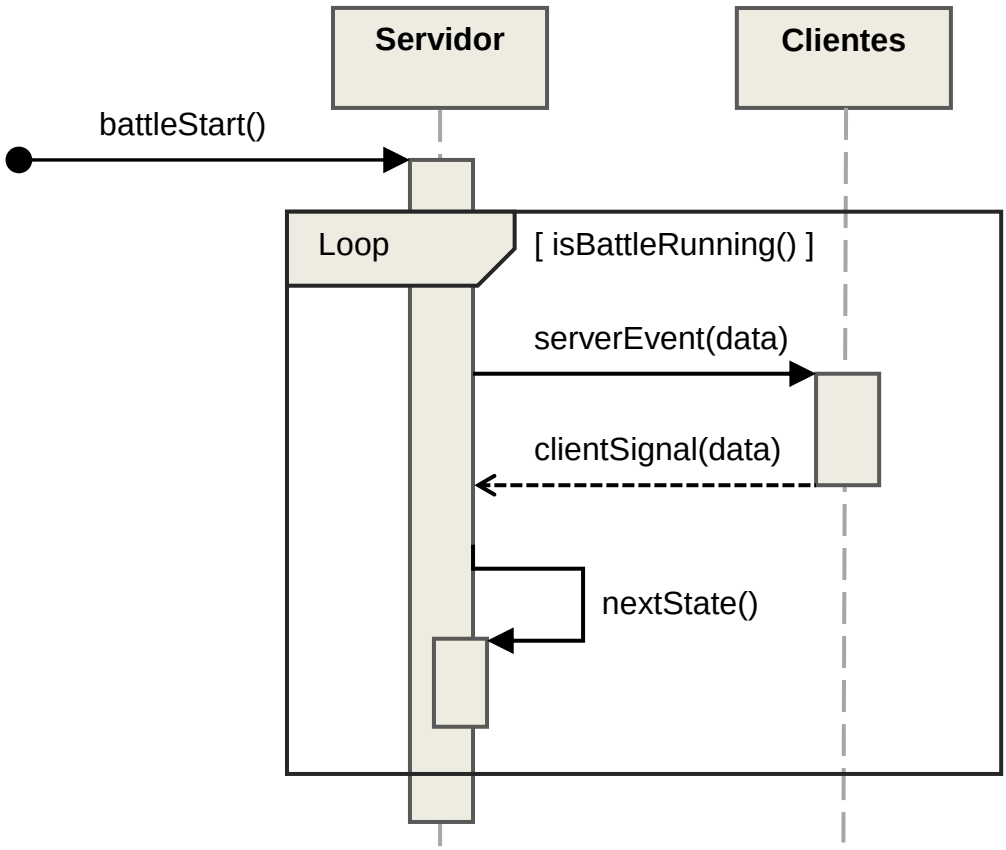
Quadro 1 - Estados do fluxo de batalha

ID	Nome	Operação	Dados do Evento	Dados do Sinal
0	STARTING	Inicia os clientes; realiza animação do começo da batalha	-	-
1	COMMAND	Recebe os comandos de cada jogador	-	Comando do jogador
2	EXECUTION	Avalia a execução do turno e envia os passos para exibição ao jogador nos clientes	Lista de passos da execução do turno	-
3	ENDING	Verifica condição de vitória ou derrota; exibe resultado ou retorna ao estado COMMAND	Resultado	-

Fonte: Próprio autor

A cada estado do sistema, o objeto servidor envia um evento para todos os objetos clientes, que realizam as ações pertinentes ao estado referente ao evento recebido. Quando um cliente termina a sua ação, um sinal é enviado de volta ao servidor, que prossegue para o próximo estado após receber um sinal de todos os objetos clientes.

Figura 3 - Diagrama de sequência da execução do sistema de batalha



Fonte: Próprio autor

3.4 Inteligência Artificial

No sistema de batalha, a inteligência artificial determina qual ação será tomada pelos inimigos em um determinado turno de batalha. O sistema usa lógica difusa para determinar o comando escolhido pelo inimigo.

A implementação da lógica usa um conjunto de fatores como input. Cada fator representa a probabilidade da unidade selecionar um comando específico. Alguns fatores podem ser omitidos, no caso da unidade não possuir magias de cura ou ataque.

O cálculo do fator de cada comando é onde a inteligência artificial determina o perfil do inimigo. Por exemplo, uma unidade com alto poder físico possui maior probabilidade de usar ataques físicos, enquanto uma unidade com poder mágico possui maior chance de usar magias. O quadro abaixo representa os fatores para cada comando e os seus influenciadores de probabilidade:

Quadro 2 - Fatores de decisão para AI de inimigos em batalha

Comando	Alvo	Influenciadores	Condição
Ataque	Herói	ATK da unidade	
Defesa	n/a	HP, DEF e posição da unidade	
Magia: Ataque	Herói	INT da unidade	
Magia: Cura	Qualquer	Dano causado a todos os monstros vivos, posição da unidade	Dano > 0
Magia: Suporte	Qualquer	Número de monstros, RES e posição da unidade	
Magia: Perigo	Qualquer	HP e INT da unidade	HP < 50%

Fonte: Próprio autor

3.5 Multiplayer

O jogo possui um modo *multiplayer* cooperativo, permitindo que até quatro jogadores entrem em uma batalha em rede local via Wi-Fi Direct. Essa tecnologia permite que dispositivos Android conectem-se diretamente, sem a necessidade de roteador ou outra rede local existente.

A arquitetura do sistema de batalha foi projetada com a implementação do *multiplayer* em mente e, por isso, o sistema *multiplayer* funciona de forma transparente dentro do sistema de batalha. Na batalha *multiplayer*, cada jogador possui um objeto cliente. O jogador que inicia a partida *multiplayer* possui o objeto

servidor do sistema de batalha, e cada jogador conectado possui um objeto “servidor remoto” que recebe as mensagens do servidor através da interface de rede e a repassam para o objeto cliente local.

A *engine* possui funções para codificar e decodificar as estruturas de dados usadas no jogo para JSON. Estas funções foram usadas no envio dos pacotes de dados através dos *sockets* entre cliente e servidor.

O GameMaker: Studio não suporta conexão com Wi-Fi Direct nativamente. Um plugin foi desenvolvido exclusivamente para a utilização neste projeto, adicionando as funcionalidades existentes na API WifiP2P do Android ao jogo.

3.6 Plugins

Neste projeto foram desenvolvidos e usados dois *plugins* para a *engine* GameMaker: Studio. Os *plugins* são programados em código Java e rodam no ambiente nativo da plataforma Android:

- AndroidGPS: é o *plugin* responsável por receber a localização GPS do dispositivo e transmiti-la para a *engine*.

- WifiDirect: expõe as funções de Wi-Fi Direct do Android para comunicação em rede.

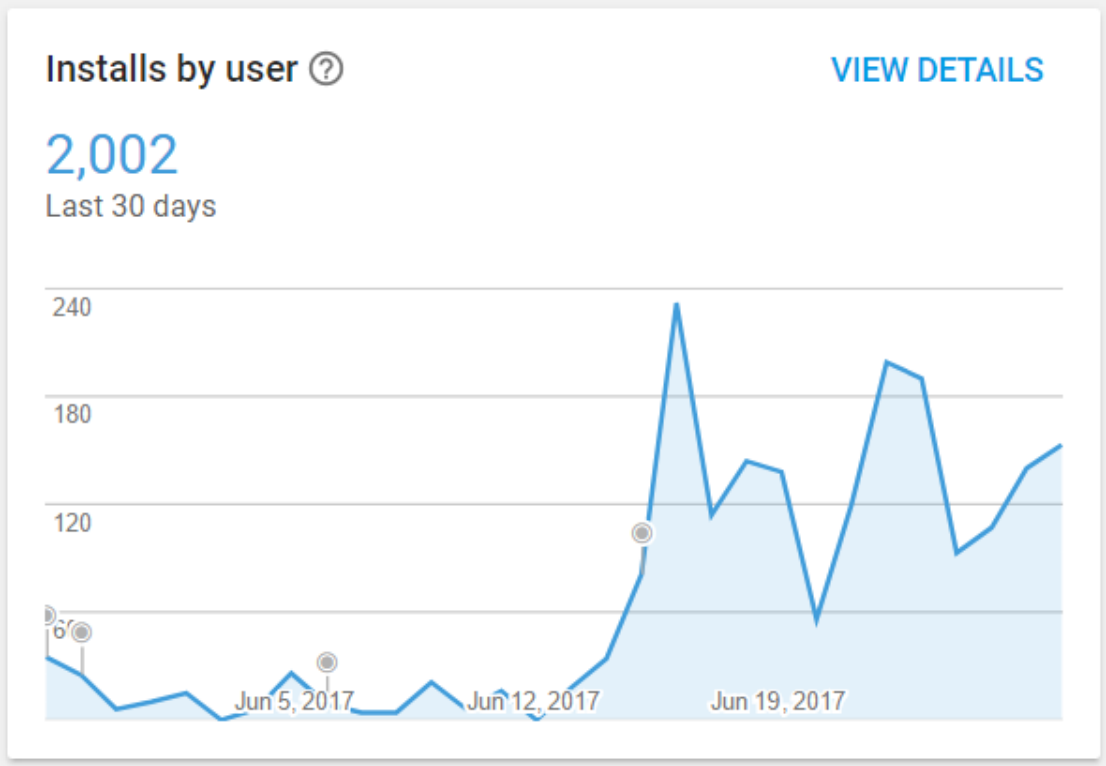
4 RESULTADOS

Todos os sistemas e mecânicas principais planejados no início do desenvolvimento foram implementados na versão final do jogo. Gridalis foi publicado na Google Play Store no dia 27 de Abril de 2017. Uma atualização foi lançada no dia seguinte após receber diversos feedbacks de jogadores pedindo as alterações, ajustando diversos elementos do gameplay.

Gridalis foi bem recebido pelos jogadores. Segundo as estatísticas providas pela Google Play Store, o jogo foi instalado mais de 2000 vezes em um mês após o lançamento, com uma avaliação média de 4.5 estrelas por mais de 100 usuários.

Gridalis encontra-se disponível para download em:
https://play.google.com/store/apps/details?id=com.leonmillan.Gridalis

Figura 4 - Gráfico de instalações por usuários ao longo de 30 dias



Fonte: Google Play Developer Console

Figura 5 - Gráfico de avaliações por usuários na Play Store

Fonte: Google Play Store

Nos 30 primeiros dias após o lançamento, Gridalis esteve listado na categoria Top RPGs Novos em diversos países pela Play Store, frequentemente aparecendo entre os 50 maiores da lista.

Figura 6 - Listagem dos rankings da categoria RPG na Play Store

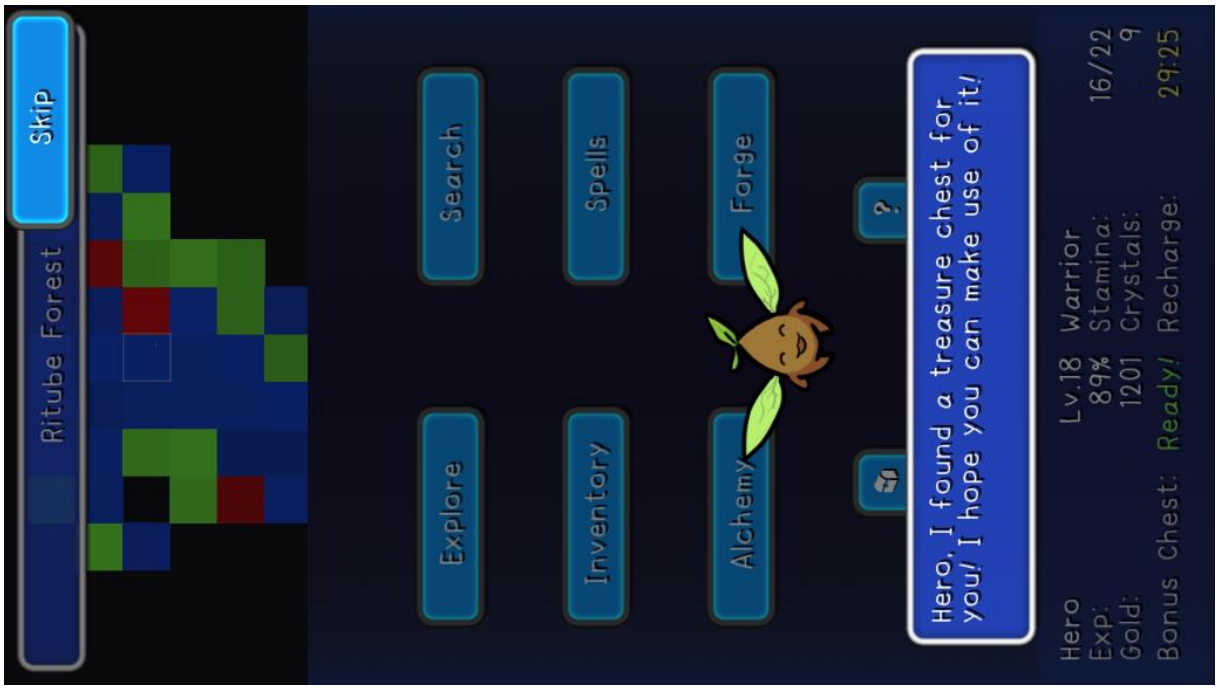
Rank	Country	Category	List
34	United States	Role Playing	Top New Free
28	Germany	Role Playing	Top New Free
34	France	Role Playing	Top New Free
59	Japan	Role Playing	Top New Free
36	United Kingdom	Role Playing	Top New Free
39	Brazil	Role Playing	Top New Free
39	Spain	Role Playing	Top New Free
50	Indonesia	Role Playing	Top New Free
57	Netherlands	Role Playing	Top New Free
58	Switzerland	Role Playing	Top New Free

Fonte: AppBrain

Figura 7 - Screenshot da criação de personagem

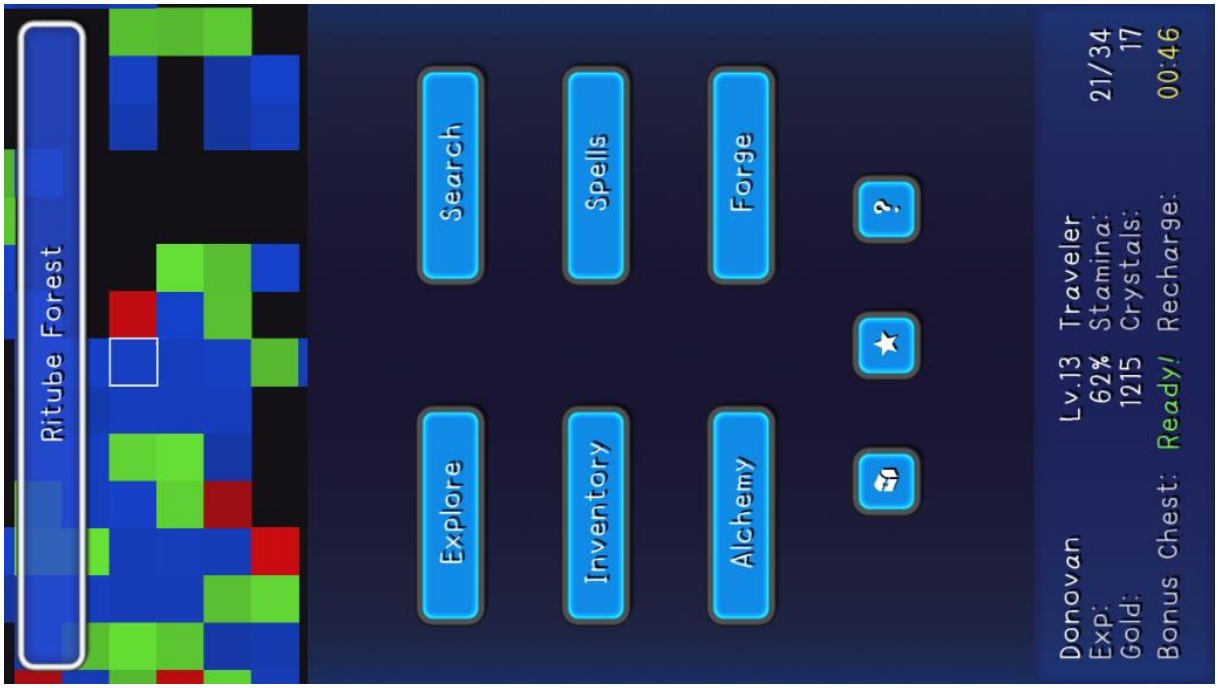
Fonte: Próprio autor

Figura 8 - Screenshot da abertura do jogo (tutorial)



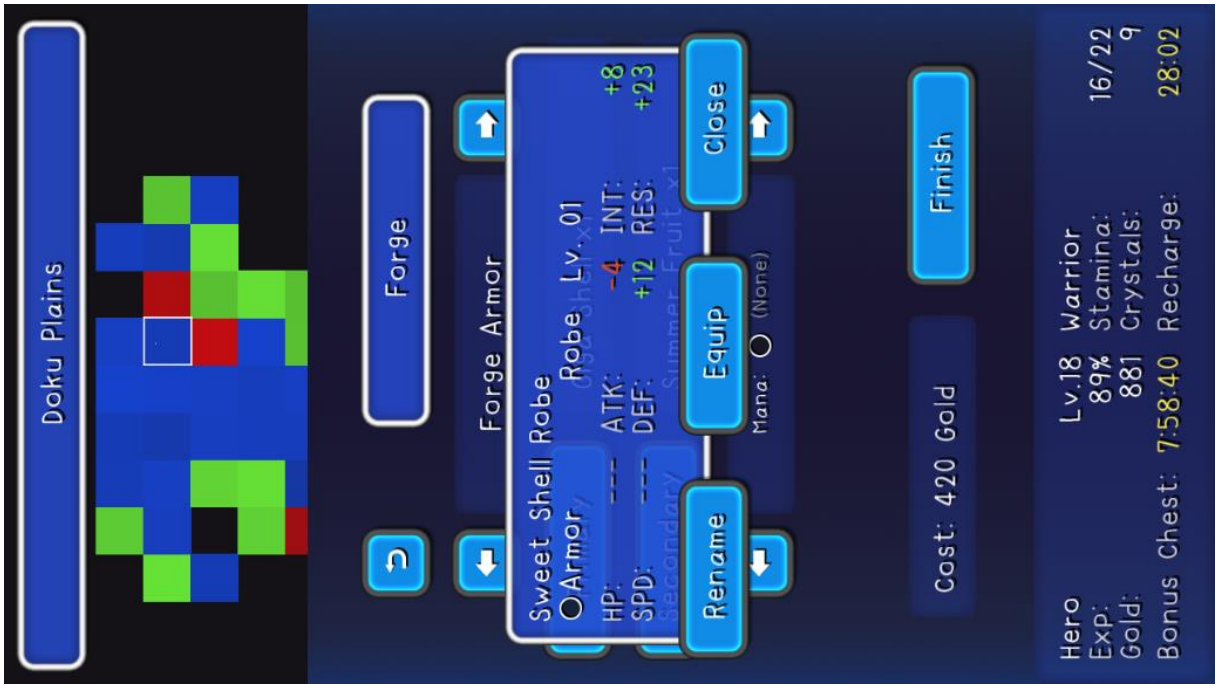
Fonte: Próprio autor

Figura 9 - Screenshot da tela principal do jogo



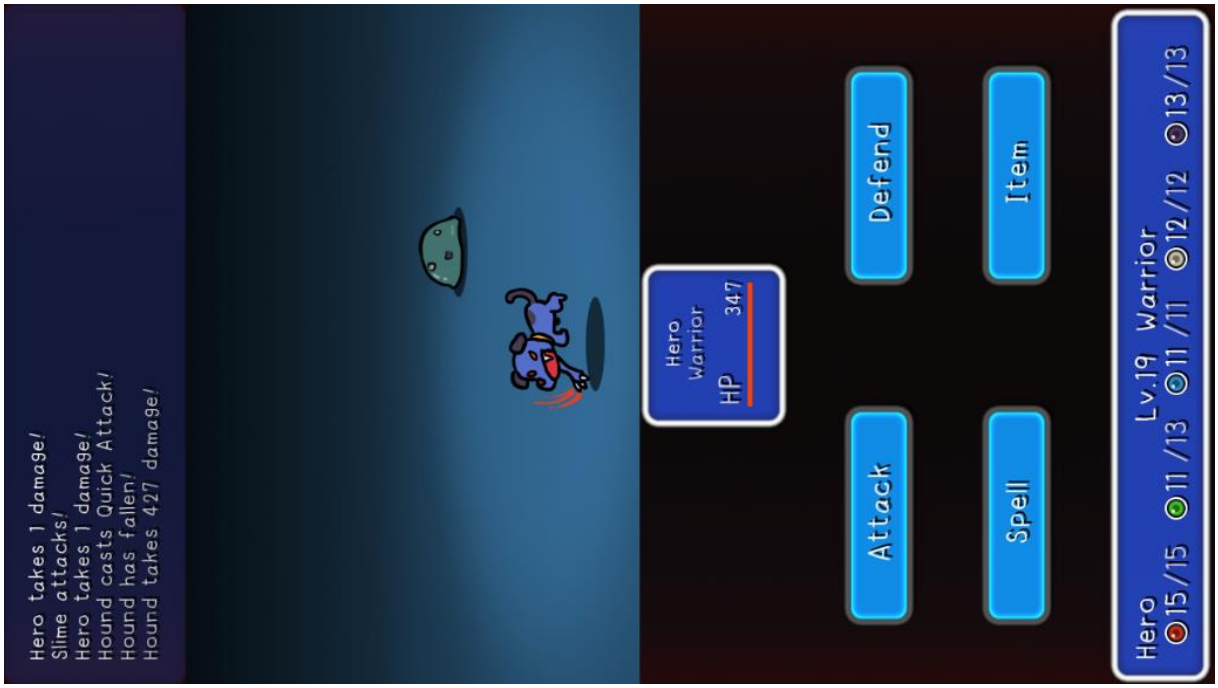
Fonte: Próprio autor

Figura 10 - Screenshot da tela de forja de equipamentos



Fonte: Próprio autor

Figura 11 - Screenshot da tela de batalha



Fonte: Próprio autor

5 CONSIDERAÇÕES FINAIS

O jogo superou as expectativas iniciais, tendo uma avaliação positiva de diversos jogadores. O sucesso provou que a fórmula possui grande potencial no mercado, mas ainda há muitos pontos que necessitam de melhorias.

Com os comentários recebidos por jogadores, ficou evidente a importância de balancear os parâmetros do jogo, de forma que o jogador sempre possa progredir e não perca o interesse em jogar. O balanceamento do jogo é extremamente sensível, e a alteração de um parâmetro pode alterar dramaticamente a dinâmica do *gameplay*. Com a metodologia usada neste projeto, o balanceamento dos sistemas foi feito individualmente, com planilhas simples definidas para cada sistema. Futuramente será necessário implementar planilhas mais sofisticadas, que usem os dados de todos os sistemas do jogo simultâneamente para simular a interação dos parâmetros.

Na versão atual, os jogadores que não podem jogar em *multiplayer* ou que não têm facilidade de locomoção possuem uma enorme desvantagem comparados aos outros jogadores. Será necessário redefinir algumas mecânicas para permitir que estes jogadores consigam progredir no jogo.

Uma crítica comum encontrada nos feedbacks dos jogadores é a falta de um sistema de loja para compra e venda de itens do jogo. Também foi sugerido mais opções de customização, adição de gráficos para refletir as características do jogador e do mapa do jogo, e a inclusão de opções de idiomas para a acessibilidade de jogadores sem domínio do inglês.

REFERÊNCIAS BIBLIOGRÁFICAS

ANDROID. **Wi-Fi Peer-to-Peer**. Disponível em <https://developer.android.com/guide/topics/connectivity/wifip2p.html>. Acesso em: 02/06/2017.

APPBRAIN. **AppBrain**. Disponível em <https://www.appbrain.com/>. Acesso em: 26/06/2017.

GOOGLE. **Google Drive**. Disponível em <https://www.google.com/intl/en/drive/>. Acesso em: 02/06/2017.

GOOGLE. **Google Play Store**. Disponível em <https://play.google.com/store/>. Acesso em: 02/06/2017.

NIANTIC. **Pokémon GO**. Disponível em <http://pokemongo.nianticlabs.com/>. Acesso em: 02/06/2017.

PINTO, L. M. S.. **Gridalis**. Disponível em <https://play.google.com/store/apps/details?id=com.leonmillan.Gridalis>. Acesso em: 26/06/2017.

RODGER, J. **Pokémon Go update from Niantic promises Legendary characters and other long-awaited features**. Disponível em <http://www.birminghammail.co.uk/news/midlands-news/pokemon-go-update-niantic-legendary-12425104>. Acesso em: 02/06/2017.

SUBLIME TEXT. **Sublime Text 3**. Disponível em <https://www.sublimetext.com/>. Acesso em: 02/06/2017.

SWATMAN, R. **Pokémon Go catches five new world records**. Disponível em <http://www.guinnessworldrecords.com/news/2016/8/pokemon-go-catches-five-world-records-439327>. Acesso em: 02/06/2017.

YOYOGAMES. **GameMaker Studio**. Disponível em <https://www.yoyogames.com/gamemaker>. Acesso em: 02/06/2017.

APÊNDICE 1 – CÓDIGO FONTE

O código-fonte do projeto encontra-se em CD em anexo.