Adventures Inc: Implementação de hibridização entre os gêeros Action RPG e Idle Game por meio de mecânicas de jogos assimétricas

Felipe Moraes Sérgio Nesteriuk

Universidade Anhembi Morumbi, Escola de Artes, Arquitetura, Design e Moda, Brasil

Resumo

A partir da análise do conceito de gênero em jogos digitais, mais especificamente de Action RPG e Idle Game, foi elaborado um projeto e implementado um jogo que procurou manter em seu game design aspectos essenciais destes dois gêneros ao mesmo tempo em que procurou proporcionar uma espécie de hibridização entre algumas de suas características. Por meio de estudo prévio realizado sobre as características intrínsecas de cada um destes gêneros e sobre o conceito de mecânicas de jogo assimétricas [Despain 2012], iniciou-se o desenvolvimento projetual e a implementação do jogo Adventures Inc, cujo processo é apresentado neste artigo.

Palavras-chave: *game design*, gêneros de jogos, mecânicas assimétricas.

Informações para Contato:
felipetm6@hotmail.com
nesteriuk@hotmail.com

1. Introdução

A partir da análise do conceito de gênero em jogos digitais, mais especificamente *Action RPG e Idle Game*, o projeto foi elaborado visando manter as regras essenciais de cada uma das mecânicas predominantes destes gêneros e corrigir pontos negativos, em um levantamento realizado com alguns jogadores, utilizando a partir combinação de elementos destas mecânicas em uma espécie de gênero hibrido.

Por meio dos estudos realizados sobre cada um destes dois gêneros [Moraes e Nesteriuk 2015] e o conceito de mecânicas assimétricas [Despain 2012], iniciou-se um processo de seleção do que poderia ou não ser mantido em cada uma das duas mecânicas. Elementos de ambas, que tornam o produto demasiadamente dependente de tempo livre por parte do usuário – como grinding em um ARPG para passar uma missão específica, farming para ter recursos suficientes para comprar um item desejado, ou a espera constante para ter recursos para seu próximo upgrade em um Idle Game – foram os principais alvos de alteração no projeto. Como objetivo geral, o jogo poderia conter qualquer um desses elementos, mas o jogador não necessitaria obrigatoriamente ter que optar por tais meios para conseguir progredir em determinada etapa do jogo.

Outro alvo no projeto foram as curvas geradas a partir da relação entre duração de sessão de jogo e etapa de progressão. Como pode verificar-se no Gráfico 1, a duração de uma sessão de um ARPG cresce a cada etapa de progressão do jogo. Um jogador focado em alcançar a melhor build possível tende a dedicar cada vez mais tempo em atividades repetitivas para alcançar sua meta. Se considerada a possibilidade de replay de um ARPG, a grande maioria das atividades feitas para alcançar a melhor build possível serão refeitas, aumentando ainda mais o tempo absoluto dedicado ao jogo [Moraes e Nesteriuk 2015].

Já em um Idle Game, como pode verificar-se no Gráfico 1, a duração de uma sessão aumenta quando alcança-se meados do mid game e torna a diminuir quanto mais aproxima-se do end game. O que eram antes sessões frequentes, com vários recursos a serem manejados, passam a ser sessões curtas de um ou dois recursos diferentes e o intervalo da unidade temporal entre elas muda de minutos ou horas para dias ou até semanas [Moraes e Nesteriuk 2015]. Como objetivo específico para o projeto, buscou-se não haver diferença entre cada etapa de progressão no jogo, ou seja, a variação de tempo gasta em cada sessão de jogo deveria ser constante.

Gráfico 1: comparação da duração de sessões entre os jogos de Action RPG e Idle Game

2. Projeto

Com os objetivos, geral e específico, em mente, as diversas regras existentes nos gêneros ARPG e Idle Game passaram a ser analisadas e abordadas. Com isso buscou-se permitir que o jogo se encaixasse nesse novo (sub)gênero planejado com o intuito de oferecer as experiências das mecânicas originais sem uma exigência de tempo elevada.

Considerando que uma mecânica assimétrica existe quando dois ou mais jogadores têm diferentes experiências propostas enquanto jogam um mesmo jogo juntos, competitiva ou cooperativamente [Despain 2012], há uma tentativa de implementar conjuntos de regras herdados de ambas as mecânicas sem que eles entrem em conflito, mas sim complementem uma a outra, em uma espécie de relação simbiótica, e assim proporcionar duas experiências diferentes para um único jogador.

Para isso, elaborou-se um conjunto de regras que permitem que a sessão de jogo do jogador nunca acabe. Quando parar de jogar ativamente a partida, o jogo utiliza suas mecânicas de funcionamento em Idle para que o usuário continue progredindo indiretamente. Sendo assim, quando ele iniciar sua próxima sessão ativa de jogo, este terá progredido - ainda que não tanto quanto progrediria caso ele es tivesse jogando ativamente.

2.1 Vantagens da mistura assimétrica de mecânicas de jogo

Com o objetivo central definido, foi estabelecido quais elementos de cada uma das mecânicas deveriam permanecer no jogo e quais adaptações se mostravam fundamentais. Um dos principais contratempos relacionado à mistura assimétrica de mecânicas foi conseguir mesclar elementos de, neste caso, duas mecânicas que em alguns aspectos podem ser consideradas opostos garantindo ainda que o jogo ficasse balanceado e pudesse seguir o conceito de *flow* nos games [Chen 2007].

Uma das principais vantagens das duas mecânicas escolhidas para o projeto é que ambas têm em comum uma grande necessidade de gerenciar recursos. Seja distribuindo pontos de habilidade, atributos, ou pensando como seu dinheiro, biscoitos ou *mana* serão utilizados durante sua próxima sessão de jogo; a capacidade de administração do jogador sempre é testada. Sendo assim, optou-se por manter esse elemento por ser parte essencial da jogabilidade e da própria experiência do jogador.

Outra vantagem observada encontra-se justamente na discrepância entre os conceitos de cada um dos gêneros escolhidos. Enquanto ARPG demanda muito tempo de jogo, exploração e *farming*, podendo assim ser caracterizado como um jogo ativo; o Idle Game demanda pouquíssimo tempo do jogador dentro do jogo, uma vez que evolui o suficiente para deixar seus geradores de recursos automatizados, tornando-se cada vez mais passivo.

Por ser um estilo de jogo com progressão contínua e exponencial, a base do Idle Game pode ser utilizada para substituir o elemento que exige mais tempo do jogador com atividades repetitivas no ARPG, o *farming.* Com isso, essa atividade torna-se algo feito automaticamente por algoritmos que só precisa ser administrada periodicamente pelo jogador, conferindo a possibilidade de evitar a atividade repetitiva e progredir enquanto não joga ativamente. Assim, a sessão de jogo não somente é reduzida ou preenchida por atividades mais interessantes e menos repetitivas, como também se elimina um fator que pode potencialmente ser determinante para um usuário optar por não usufruir de determinado produto, seja por falta de tempo ou interesse em *farming*.

2.2 Desvantagens da mistura assimétrica de mecânicas de jogo

Embora a administração de recursos seja um ponto comum entre ambos os gêneros, a forma como ela existe em cada um destes separadamente acaba prejudicando o balanceamento e o *flow* da nova mecânica aqui proposta.

De um lado, o ARPG oferece grande diversidade de recursos para serem administrados, permitindo maior customização das personagens e estratégias por parte do jogador. Mas, essa superespecialização das personagens acaba levando à quebra do conceito do ARPG perfeito, descrito por Beyerl [2014], no qual todas as personagens devem vencer um combate contra um inimigo no mesmo intervalo de tempo, garantindo que não há desvantagem ao optar por uma *build* específica. Neste sentido, outro problema do ARPG apontado pelo autor é que, muitas vezes, essa diversidade acaba criando conjuntos específicos de atributos, habilidades e equipamentos que são muito mais eficazes que os demais, desviando o interesse do jogador de conjuntos menos eficazes. Desta forma, se diminui a requisição de habilidade do jogador, uma vez que esse conjunto específico torna, comparativamente, o jogo todo mais fácil.

Por outro lado, o Idle Game oferece grande diversidade de estruturas e *upgrades* para o jogador utilizar seus recursos. A quantidade exacerbada de elementos geradores de recurso faz com que os preços

dos *upgrades* e de novas estruturas evoluam exponencialmente, atingindo valores exorbitantes cujo objetivo é, única e exclusivamente, acompanhar a evolução de sua produção [Moraes e Nesteriuk 2015].

Com isso, ao utilizar o modelo do Idle Game como uma alternativa possível ao *farming*, a progressão do jogador ao longo do jogo fica sujeita a assumir a mesma identidade exponencial da mecânica original, podendo tanto fazer com que os novos equipamentos se tornem muito caros, requerendo dias sem jogar para que a progressão passiva permita ao usuário a geração de novos equipamentos para a participação na parte ativa; ou, ainda pior, fazendo com que o jogador alcance grandes quantidades de recursos e torne, a partir daí, todo o sistema de progressão muito fácil.

2.3 Soluções para a mistura assimétrica de mecânicas

Tendo em vista as vantagens e desvantagens da mistura de mecânicas, iniciou-se a busca por soluções para os problemas descritos anteriormente, tendo em mente sempre manter o balanceamento de jogo e seguir o conceito de *flow* nos games [Chen 2007]: proporcionar um fluxo de jogabilidade que possa se ajustar à habilidade do jogador e proporcionar uma experiência que consiga se manter em uma zona intermediária entre a ansiedade e o tédio.

Uma das primeiras decisões relacionadas à parte ativa cuja base é o ARPG foi simplificar o sistema de administração de recursos. Embora a ampla customização proporcionada ao jogador seja um elemento potencialmente interessante, ela dificulta o balanceamento e a manutenção do conceito d e que todas as classes devem fazer uma mesma tarefa no mesmo intervalo de tempo.

Para resolver esse problema específico, a quantidade de atributos, equipamentos e habilidades das personagens foi reduzida e padronizada. Ao invés de seis ou sete atributos diferentes, em que cada um tem influência em mais dois ou três atributos secundários, por exemplo, somente quatro atributos (Vitalidade, Força, Destreza e Sabedoria - ilustradas na Figura 1) são utilizados, sendo que cada um destes tem influência em somente três dos atributos secundários (Vida, Dano e Resistência). Além disso, ideias como habilidades passivas e diversos atributos em equipamentos foram descartadas, dando espaço para equipamentos simplificados com influência aplicada a estes dois tipos de atributos.

Figura 1: Os quatro atributos de “Adventurers Inc”.

Ao invés de diversas árvores de habilidades diferentes, passivas e ativas, cada uma com seu objetivo e excentricidade, foram instituídas três habilidades para cada personagem (habilidade direta, habilidade em área e habilidade defensiva), que, embora tenham um elemento visual e mecânico diferente, permitem que todos as personagens tenham a mesma eficácia em um confronto direto com o inimigo ou adversário.

Por meio dessas alterações, várias questões problemáticas no conceito da nova mecânica puderam ser resolvidas. Com a delimitação de atributos e habilidades oferecidas ao jogador, menos informações precisam ser administradas antes de começar uma partida e entrar na parte “ativa” de fato, colocando todo o foco dessa parte da mecânica na ação em si, e não no planejamento – o que também implica em uma otimização do tempo deste jogador. Além disso, a simplificação das possibilidades de customização contribui para que a possibilidade de uma combinação de equipamentos seja mais vantajosa que a de outra personagem não exista.

Consequentemente, em termos de desenvolvimento e implementação, a simplificação na quantidade de itens a serem controlados durante o balanceamento do jogo tornou o processo inteiro mais intuitivo e amigável, uma vez que existem menos variáveis para serem controladas – questão previamente apontada como problemática pelo público-alvo.

Outro ponto importante a ser revisto relacionado à dinâmica projetual foi a forma de progressão existente no Idle Game. A parte passiva do jogo, moldada nas bases do gênero estudado, corria o risco de adquirir um nível de evolução exponencial de grandes valores poderia tornar o jogo extremamente fácil e passivo. As fórmulas utilizadas para a geração de recursos e cálculo de custos foram pensadas de forma simples e direta,

focadas em manter o aspecto exponencial da progressão sem permitir que valores muito grandes pudessem ser atingidos.

Por conta dessas alterações, tornou-se mais fácil o controle de valores e de como a progressão do jogador se desenvolvia. Ainda assim, permanecia a possibilidade de o jogador se beneficiar demasiadamente do tempo fora do jogo. Por exemplo: um mês sem jogar poderia fazê-lo acumular muito mais recursos do que o planejado (esperado) para a etapa do jogo em que se encontra, tornando a sua volta ao produto muito mais fácil. Embora o objetivo da parte passiva seja justamente substituir uma atividade repetitiva e demorada que o jogador deveria fazer, idealmente isso seria feito sem influenciar demais a curva de progressão de dificuldade do jogo. Para resolver tal problema, algumas soluções projetuais foram tomadas.

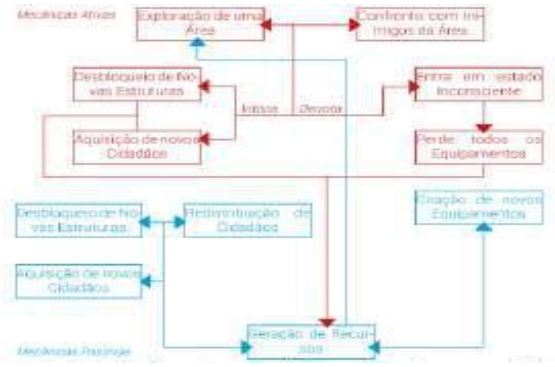
A primeira delas foi uma limitação para o acúmulo de recursos obtidos pelo jogador. A criação de um sistema de armazenamento destes itens permitiu estabelecer um "teto" (valor máximo), fixo ou variável de acordo com a progressão do jogador para quantos itens pode-se ter em determinado momento do estado do jogo. Embora essa solução sozinha resolva grande parte do problema e garanta que o jogador pare de progredir depois de determinada quantidade de tempo sem jogar, outra medida foi tomada para adicionar um pouco mais de dificuldade ao jogo. Toda vez que o jogador falhar em um nível, tendo sua vida reduzida a zero, todos os equipamentos utilizados para aquela tentativa são perdidos – tendo, portanto, que ser reconquistados no jogo.

Dessa forma, ainda que com muitos recursos acumulados, o jogador precisa definir uma estratégia com um mínimo de planejamento antes de entrar em um novo nível, tendo em mente qual a melhor combinação de equipamentos que apresenta um custo-benefício apropriado para o ambiente a ser explorado.

Caso continue tentando entrar em níveis mais difíceis sem a habilidade necessária na parte ativa e nenhum conjunto de equipamentos específico como facilitador, o jogador continuará perdendo recursos até chegar ao ponto de ter que esperar sua produção acumular uma quantidade determinada para que tenha condições mínimas de voltar a jogar.

3. O jogo "Adventurers Inc"

Considerando a essência presente em uma mecânica assimétrica [Despain 2012], o objetivo principal da implementação do jogo foi estabelecer um conjunto de regras herdadas de dois gêneros diferentes que sejam complementares. Após analisar as vantagens e desvantagens de tais regras, e estabelecer soluções projetuais que compensassem os possíveis problemas de cada um desses elementos, o seguinte fluxograma de mecânica de jogo (fluxograma 1) foi estabelecido, representando a dinâmica na qual o jogo se baseia



Fluxograma 1: relações entre mecânicas ativas (com variações entre as condições de vitória e de derrota), em vermelho, e mecânicas passivas, em azul.

Como se pode observar, a mecânica consiste em um *looping* que é formado por dois pontos chave: a etapa de exploração de uma área, correspondente à parte ativa do jogo (em vermelho) e a etapa de geração de recursos, correspondente à parte passiva do jogo (em azul). O jogador sempre inicia sua sessão na etapa de geração de recursos, na qual pode optar por acessar diferentes controles para administrar seu reino. Por meio dele, o jogador pode decidir acelerar ou desacelerar produções, criar novos itens, e organizar suas posses. Uma vez pronto para jogar a parte ativa, o jogador parte para a exploração de área. O caminho no fluxograma a partir desse momento depende única e exclusivamente da habilidade do jogador, e suas consequências serão definidas a partir do sucesso ou fracasso em superar os desafios estabelecidos no *level design*.

Gráfico 2: Comparação da duração das sessões entre os jogos de Action RPG, Idle Game e do novo (sub)gênero proposto em Adventures Inc.

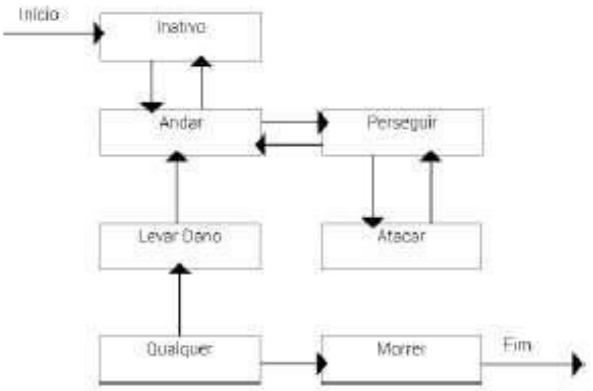
Entretanto, independente do resultado, a próxima etapa ao sair da parte ativa do jogo é a gestão de recursos, em que o *looping* recomeça. Caso o jogador opte por sair de sua sessão de jogo, o momento de sua saída é registrado. Ao iniciar uma nova sessão, o jogo dá ao jogador na etapa de gestão de recursos o valor que ele teria obtido caso o jogo estivesse funcionando de maneira contínua, assim como termina qualquer outra ordem (como a criação de itens, por exemplo) que acabaria nesse mesmo intervalo de tempo. Sendo assim, o jogo progride como se só houvesse uma grande sessão (conforme representado no Gráfico 2), e não nas sessões de jogo cortadas que demandam tempo e dedicação variados dos jogadores.

3.1 Soluções para o projeto

Com as decisões tomadas sobre balanceamento e compensação de elementos ao misturar as duas mecânicas de jogo, iniciou-se então o projeto de desenvolvimento do jogo. Durante o planejamento de personagens, inimigos, cenários e níveis, entretanto, verificou-se uma discrepância na gestão de projetos formada pela tríade qualidade, tempo e recursos. Sendo assim, iniciou-se um planejamento de otimização de todas as tarefas envolvidas, buscando garantir que cada problema resolvido em nível projetual pudesse ter sua solução reproduzida em diversos pontos diferentes do jogo em nível de implementação efetiva.

Uma das áreas em que esta proposta foi melhor aplicada foi a programação. Personagens jogáveis, inimigos e personagens não jogáveis (NPCs – *Non Player Characters*) passaram por um procedimento de elaboração de um código base universal. Esse processo, embora mais complexo, por envolver uma abstração maior durante a programação, facilitou extremamente o processo de implementação de novos itens.

Isso acontece, por exemplo, com a máquina de estados dos inimigos. Por terem estados comuns (representados no fluxograma 2), como inativo, andar, perseguir, levar dano, atacar e morrer, todos podem utilizar um código pai universal como base. Cada inimigo tem um código próprio gerado a partir desse código pai, em que somente os estados que sofrem alguma alteração comum são atualizados. No caso de um chefe (*boss*), por exemplo, o estado inativo é substituído pelos usuais cinco segundos de espera no lugar por uma transformação que altera as propriedades de tamanho do inimigo.



Fluxograma 2: Máquina de Estado base

Esse processo todo, apesar de mais complexo ao programar o código pai, facilita a implementação de todos os inimigos, uma vez que o programador precisa atuar somente em estados chave nele, ao invés de reescrever todo o código. Entretanto, esse tipo de código também acaba por replicar qualquer tipo de problema comportamental do código, uma vez que uma ação indesejada é replicada por todos os filhos ao longo do produto. Por outro lado, a abordagem de tais problemas é mais fácil uma vez que o programador só precisa lidar com eles uma vez, em um único código, e qualquer tipo de solução atingida é automaticamente espalhada por todos os códigos filho do jogo.

A utilização de códigos globais para itens de mesma categoria também permite a fácil implementação de novos itens no jogo. Por necessitar somente de alterações pontuais em funções específicas, por exemplo, um único inimigo demora pouco tempo para ser adicionado ao arquivo do projeto de forma funcional, seguindo os padrões de comportamento de outros dessa mesma categoria. Com isso, ainda que necessitando de alguns refinamentos e da programação de alguns diferenciais para tornar o comportamento de cada inimigo no jogo único, a prototipagem e teste puderam ser feitos muito mais rapidamente pela equipe de desenvolvimento.

Esta ideia também pôde ser aplicada no desenvolvimento do conteúdo tridimensional (3D) do jogo. Utilizando o sistema de animação da *engine* Unity conhecido como *Mecanim*, foi possível estabelecer um padrão na área de modelagem e animação que permitisse a aceleração na produção de novos modelos e a reutilização de bibliotecas de animações em personagens diferentes para acelerar a implementação e teste dos elementos *ingame*.

Um dos procedimentos adotados na modelagem e animação em 3D foi a padronização de modelos com

características similares, como, por exemplo, bípedes com movimentação humanoide. Todos os modelos feitos para personagens, jogáveis ou não, utilizam um mesmo corpo de base (figura 2), duplicado para cada modelo novo a ser desenvolvido. Isso garantiu não somente que haverá um padrão entre seres da mesma "espécie", mas também facilitou a elaboração de indumentárias e acessórios com um corpo base já estabelecido (figura 3).

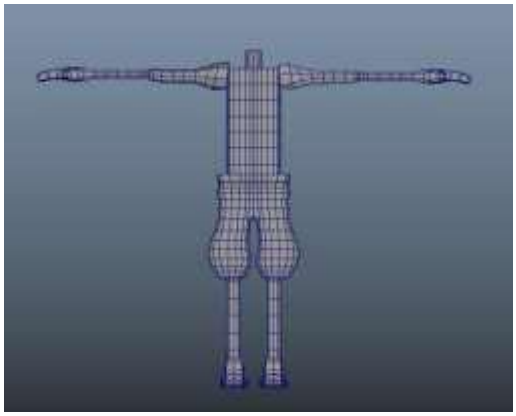


Figura 2: Wireframe de modelo base 3D (biblioteca).

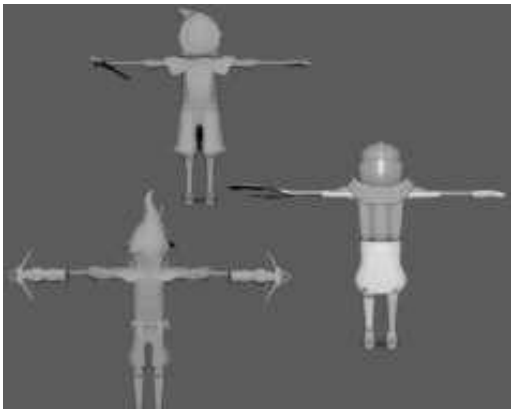


Figura 3: Silhuetas dos modelos de personagens elaborados a partir do mesmo modelo base.

Esse procedimento também duplicou as hierarquias de esqueleto (*rigging*) utilizadas para seres humanoides ou antropomorfizados. Uma das principais vantagens observadas da padronização do esqueleto foi a possibilidade da reutilização de animações em diferentes personagens através do sistema *Mecanim*, possibilitando uma implementação antecipada de modelos sem animação finalizada já dentro do jogo. Com isso, foi possível a implementação de novas personagens, habilidades, interfaces e mecânicas de jogos de forma antecipada, estabelecendo uma produção paralela de conteúdo por parte do modelador e do programador - antes o trabalho de um só poderia acontecer após a finalização do trabalho do outro.

O sistema de criação de equipamentos também foi outro ponto influenciado por essas dinâmicas, em que os equipamentos foram modelados utilizando como base sua versão original. Esse processo permitiu uma implementação mais fácil pelo programador, uma vez que todos os equipamentos tinham o mesmo centro, rotação e propriedades, simplificando o código de criação e seleção dos mesmos.

Outro aspecto que recebeu atenção especial na área de programação e implementação na Unity foi a otimização de todo o conteúdo produzido. Por ser um jogo que tem uma parte jogável disponibilizada para plataformas móveis, o Adventures Inc devia contar com uma versão mais otimizada do jogo possível. Para alcançar tal objetivo, pesquisas e testes sobre otimização dentro da plataforma utilizada foram realizados, buscando formas de melhorar por meio da programação o desempenho geral do jogo mesmo em *hardwares* com configurações mais simples.

Neste sentido, uma das medidas tomadas foi a abolição completa do sistema de criação e destruição de itens ao longo dos níveis. Por ser uma função de grande custo, a implementação de muitos inimigos e personagens que disparassem uma grande quantidade de projéteis ou criassem objetos que emitissem partículas constantemente mostraram-se de grande impacto na velocidade de processamento do jogo. Para compensar esse problema, foi adotada a técnica de *object pooling*, em que tudo o que aparecerá em jogo é criado em seu início do mesmo e administrado ao longo da partida por meio de técnicas como ativar e desativar o funcionamento de objetos, e reaproveitar algo que já foi utilizado e está fora da tela, isto é, do campo de visão do jogador.

Por exemplo: a personagem que utiliza bestas no jogo não consegue atirar mais do que oito flechas antes da primeira que foi disparada sair do campo de visão. Sendo assim, ao início da partida, o jogo cria oito flechas que ficam desativadas dentro do objeto "jogador". A cada disparo, uma das flechas é ativada e colocada no local de lançamento e seu código é executado normalmente. No nono disparo, a primeira flecha disparada já está há muito fora da visão do jogador. Para evitar a criação infinita de projéteis, a nona flecha é, na verdade, a primeira flecha da lista, que foi desativada, posicionada no local de início novamente e teve seu código executado do zero. A décima flecha será a segunda da lista, e a décima primeira a terceira da lista, e assim sucessivamente, de maneira que esse *looping* ocorre durante toda a partida.

Por meio da implementação dessa técnica substituindo todo processo de criação/destruição no jogo, o impacto de processamento acontece somente quando uma cena é iniciada no programa, e não durante o *gameplay*, consequentemente não impactando na fluidez do jogo ou na própria percepção do jogador.

A introdução do *object pooling* facilitou o processo de implementação de novos equipamentos também, uma vez que todas as peças que iriam ter alguma variação precisavam ser criadas logo no início da partida. Isso permitiu o posicionamento de cada uma delas individualmente dentro do próprio programa, o que é mais fácil de ser feito via código - para que depois a peça correta fosse ativada e todas as outras desativadas conforme fosse desejado. Esse processo somado aos procedimentos de criação de conteúdo estabelecidos na área do 3D acabou facilitando a implementação de todo o sistema de criação de itens, uma vez que as novas peças viriam todas com o centro, a posição e a rotação configuradas direto para sua implementação.

3.2 Balanceamento e playtest

Uma vez estabelecidos os métodos facilitadores durante o projeto e as premissas de balanceamento e compensação na mistura de duas mecânicas, iniciou -se a fase de desenvolvimento e teste. Todo o balanceamento de Adventures Inc. foi baseado em duas frentes de teste. Na primeira delas, os desenvolvedores jogavam os níveis pós -implementação de novas mecânicas. Por meio desse método próprio foi possível verificar se as premissas assumidas durante a etapa de planejamento e *level design* eram válidas e principalmente, funcionais. Nesta etapa já foi possível testar o jogo e suas características principais, avaliar a progressão estabelecida e ver como os elementos técnicos produzidos pela equipe funcionavam em conjunto, alavancando pontos a serem alterados e apontando novos padrões que deveriam ter sido estabelecidos, mas que foram previstos durante a fase de planejamento.

A outra frente de testes foi composta por usuários. O produto foi disponibilizado para jogadores de ambos os sexos a partir de 14 anos para teste, tanto presencialmente quanto *online*, divididos em três etapas. Na primeira etapa, os usuários foram introduzidos ao projeto. Durante um breve discurso inicial, apresentou-se a hibridização de mecânicas, justificou-se as decisões tomadas durante o desenvolvimento do produto e apresentou -se o projeto em si, qual era sua etapa de desenvolvimento e o que estava efetivamente disponível para ser jogado. Também procurou-se deixar claro que o que estava sendo testado não era a habilidade do jogador em si, mas o próprio jogo. Essa etapa foi fundamental para a comprovação da hipótese elaborada para a criação do Adventures Inc, uma vez que todos os grupos de teste abordados pela equipe de produção eram compostos unicamente por usuários que se enquadravam no perfil estabelecido como público-alvo. Já nesse ponto foi possível receber *feedback* relacionado à recepção do produto por parte dos usuários, o que a hibridização de mecânicas significou para eles e quão desejada era a proposta por parte deles.

Na segunda etapa, os jogadores jogaram com a menor interferência externa possível, interagindo com os aplicadores somente em caso de dúvidas durante a sessão de jogo. Todo jogador que permitiu teve sua sessão de jogo gravada, junto com uma câmera e microfone que captavam as reações dos jogadores durante o *gameplay*. Esse material qualitativo permitiu ao grupo de desenvolvedores a coleta de informações de naturezas diversas, proporcionando um retorno necessário para a equipe prosseguir o desenvolvimento.

Com as informações coletadas com a gravação, foi possível ainda avaliar o nível de engajamento do público em determinados momentos do jogo, identificar pontos a serem adicionados, removidos ou alterados nas interfaces, avaliar o balanceamento e se o *level design* caminhavam na direção esperada do projeto: oferecendo três tipos de personagem que buscassem refletir unicamente o estilo de jogo do jogador, e não uma vantagem ou desvantagem causada pelo planejamento de cada personagem.

Além disso, as gravações (registros dos testes) provaram-se uma fonte extensa de pontos a serem corrigidos no jogo, apontando tanto por problemas já conhecidos pelos desenvolvedores que ainda não haviam sido solucionados, como uma série de novos problemas de origens diversas que não haviam sido identificados previamente. Esses problemas foram abordados pela equipe em paralelo a produção de novo conteúdo, e adicionado todo tipo de solução possível, desde correções em problemas de lógica na programação de determinado item até revisões de abordagem por conta de certas limitações de software, incompatibilidades entre programas e comportamentos indesejados causados por circunstâncias alheias ao processo de desenvolvimento em si, sobretudo *bugs*.

Por fim, pedimos para todos os voluntários completarem um questionário (quantitativo) sobre sua experiência no jogo. Essas questões têm como objetivo abordar pontos chave elencados pelos desenvolvedores como essenciais na experiência do jogador, avaliando

questões como, por exemplo, dificuldade dos níveis, balanceamento, clareza dos objetivos, clareza na HUD (Heads-Up Display) e possíveis problemas técnicos e de linguagem ao longo da partida.

Com os dados tabulados, tornou-se possível avaliar a experiência do jogador (e não somente a de seus desenvolvedores) ao longo do jogo e implementar soluções para os (novos) problemas identificados. Alguns dos problemas apontados a partir da análise dos dados tabulados e suas respectivas soluções foram:

A ausência de eventos (como lutas contra inimigos ou puzzles) por longos períodos de tempo em alguns mapas. A solução tomada foi ampliar as rotas de inimigos já posicionados previamente nas áreas em questão e aumentar a quantidade e a periodicidade de ocorrência de diferentes classes de inimigos.

A dificuldade de usar o sistema de *lock* da câmera, que oferece ao jogador uma navegação orbital ao redor do inimigo. A solução tomada foi ampliar a área de alcance do lock, aumentando assim as chances de o jogador travar a câmera ao pressionar o botão enquanto estiver olhando para um inimigo.

A dificuldade de acertar um inimigo sem o sistema de *lock* da câmera, uma vez que a personagem possui maior altura que alguns inimigos e sem olhar diretamente para eles seus golpes não o acertariam. Para resolver esse problema, foi desenvolvido outro sistema de lock que quando o jogador ataca próximo suficiente de um inimigo, a personagem ou projétil é direcionado automaticamente para o inimigo, garantindo que ele conseguirá acertar qualquer ataque corretamente direcionado em cena.

A dificuldade de finalizar um *combo*, uma vez que na versão testada todo golpe causava *knockback* no inimigo. Para esta solução foi necessário reestruturar o sistema de dano dos inimigos, instituindo uma dinâmica na qual o inimigo só sofre *knockback* quando acertado por três golpes seguidos ou por alguma uma habilidade especial.

A dificuldade em entender o objetivo de cada área. Esse foi um dos pontos recorrentes citados nas avaliações dos jogadores, que não conseguiam entender o que poderia ou deveria ser feito por eles em cada área do jogo. Duas das soluções pensadas e implementadas foram a introdução de um "guia" durante as partes ativas que indicam quando você alcança seu objetivo ou está próximo de alcança-lo, e uma introdução mais amigável do seu objetivo antes de entrar em cada área, com uma janela gráfica indicando qual o desafio da área e o que deve ser feito.

A vantagem que usuários que optam por classes de ataque de longo alcance tem em relação aos personagens de ataque corpo -a-corpo. Este foi um ponto trabalhado constantemente pelo time de desenvolvimento, cujas soluções envolveram alterações tanto no balanceamento das personagens utilizados pelo jogador como alterações no comportamento dos inimigos sobre como lidar com cada tipo de personagem.

3.3 Premissas durante o desenvolvimento

Um dos principais pontos de preocupação durante o processo de desenvolvimento foi o controle das fórmulas utilizadas nos cálculos de atributos, recursos e aprimoramentos. Embora a redução destes elementos tenha eliminado um trabalho considerável, ainda era primordial buscar melhores curvas de aprendizagem e progressão possíveis ao jogador, garantindo que não houvesse formas de corrupção de nenhum dos elementos da mecânica e que, ao mesmo tempo, o jogador não tivesse que gastar muito tempo até poder explorar o próximo nível ou área.

Sendo assim, durante o processo de desenvolvimento diversas reformulações foram feitas, baseando-se na complexidade dos cálculos e dos resultados obtidos em versões anteriores e de como as curvas de aprendizado e progressão foram formadas a partir destes. No decorrer do projeto ficou claro que uma das melhores opções levantada era simplificar todas as fórmulas, estabelecendo constantes entre coisas comuns (como a geração de recursos, por exemplo) e deixando somente uma ou duas variáveis de controle para serem alteradas dentre itens da mesma categoria. Dessa forma, por meio de projeções das curvas de aprendizagem e de progressão e repetidos testes com usuários (jogadores), tornou-se possível equilibrar todos os valores para atingir o conceito de ARPG perfeito [Beyerl 2014]. Ao mesmo tempo, por meio da hibridização dos gêneros, buscou -se evitar que o fator tempo pudesse causar um grande impacto na progressão do usuário (jogador).

Outra preocupação durante o balanceamento foi não prejudicar o jogador ao escolher uma classe específica de personagem em detrimento de outra. Para tanto, foram feitos ajustes constantes em variáveis chave para esse equilíbrio, como dano, quantidade de vida (HP – *health point*) e velocidade de ataque, para que todos as possibilidades proporcionassem ao fim da sessão de jogo uma experiência equivalente em termos de balanceamento das mecânicas assimétricas

4. Conclusão

Baseado no referencial teórico da pesquisa, é possível afirmar que a hipótese de aplicação da ideia de uma jogabilidade assimétrica [Despain 2012] ao combinar, de forma harmônica e cooperativa, duas mecânicas que

podem ser consideradas em alguns elementos divergentes, é um processo viável. Além disso, os elementos que possibilitam os gêneros escolhidos serem considerados opostos demonstraram-se elementos chave para a solução de conflitos decorrentes da hibridização proposta.

Em sua dimensão conceitual, este documento buscou analisar os pontos relevantes para a fundamentação de um novo (sub)gênero a partir de outros dois gêneros que podem ser considerados divergentes (Action RPG e Idle Game). A partir de estudo prévio sobre estes gêneros e suas mecânicas de jogo assimétricas [AVALIAÇÃO CEGA, 2015], buscou também estabelecer a fundamentação para a elaboração de uma *game design* necessário para a instituição do (sub)gênero em questão.

Enquanto projeto de desenvolvimento, foi possível elaborar um produto (jogo) que utiliza elementos de dois gêneros originais para compensar pontos considerados negativos na experiência do usuário, como objetivo de implementar um jogo com conceitos do ARPG perfeito [Despain 2012] sem sua requisição de demasiado tempo livre por parte do usuário.

Após uma breve retomada dos conceitos abordados em [Moraes e Nesteriuk 2015], uma análise mais aprofundada de elementos que apresentavam uma vantagem ou desvantagem para o projeto e quais as possíveis soluções a serem utilizadas para resolver conflitos ao misturar o Action RPG e o Idle Game, foi possível estabelecer um conjunto de regras básicas a ser utilizado no novo (sub)gênero aqui proposto e melhor apresentadas no *Game Design Document* (GDD) deste projeto.

Um dos principais focos na elaboração desse novo (sub)gênero foi garantir que o jogador pudesse optar muito tempo do seu dia ao jogo somente por opção própria, e não por uma imposição de mecânica de jogo. Desta forma, buscou-se torna-lo mais atrativo tanto para jogadores que gostam dos dois gêneros originais quanto aqueles que não gostam ou não possuem disponibilidade em jogar por conta da demanda excessiva de tempo disponível e necessário.

Um dos elementos que possibilitou a elaboração desse (sub)gênero foi a ideia de que a sessão de jogo do usuário, a rigor, nunca termina, uma vez que o progresso acontece a qualquer momento do dia, independente do jogador estar administrando a parte passiva, jogando a parte ativa ou mesmo fora da sessão de jogo. Sendo assim, não há penalidade ao jogador por não jogar, considerando que a evolução ou progressão do estado de jogo não está única e exclusivamente relacionada ao ato de jogar.

Entretanto, esse elemento também foi um dos principais obstáculos ao longo da implementação projeto. Uma vez que a curva de progressão do jogador é constante, abrimos a possibilidade de mantê-lo interessado no produto (jogo) por mais tempo que os gêneros originais. Sendo assim, faz-se necessário maior quantidade de itens e conteúdos de jogo disponibilizado ao jogador, a qual foi difícil alcançar no tempo planejado, ainda que utilizando metodologias facilitadoras durante o trabalho de desenvolvimento.

Referências

BEYERL, J. J., 2014. Achieving balance in the ARPG genre of videogames with asymmetrical choices. In: Congressus Numerantium, 220: 227-232. Disponível em: <http://www.faculty.uca.edu/jbeyerl/papers/arp_balance.pdf> [Acesso em: 12 maio 2016].

CHEN, J., 2007. Flow in Games (and everything else). Communications of the ACM. New York, vol. 50, n.4, 31-34. Disponível em: <http://www.ccs.neu.edu/home/lieber/courses/cs4500/sp09/resources/p31-chen-flow-in-games.pdf> [Acesso em: 05 maio 2016].

DESPAIN, W., 2012. 100 Principles of Game Design. San Francisco: New Riders, 2012.

MORAES, F. NESTERIUK, S., 2015. Action RPG e Idle Games: possíveis combinações assimétricas de mecânicas de jogo divergentes. In: SBC – Proceedings of SBGames 2015, 11-13 Novembro 2015 Teresina. Curitiba: SBC, 641-647. Disponível em: <http://www.sbgames.org/sbgames2015/anaispdf/artesedesign-full/147636.pdf> [Acesso em: 02 maio 2016].