UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

JOÃO MARCOS ARISTIDES ALMEIDA

GERAÇÃO DE EVENTOS DINÂMICOS PARA JOGOS DIGITAIS: UMA ABORDAGEM SISTÊMICA

CAMPINA GRANDE - PB

2021

JOÃO MARCOS ARISTIDES ALMEIDA

GERAÇÃO DE EVENTOS DINÂMICOS PARA JOGOS DIGITAIS: UMA ABORDAGEM SISTÊMICA

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Professor Herman Martins Gomes.

CAMPINA GRANDE - PB

2021

A447g Almeida, João Marcos Aristides.
Geração de eventos dinâmicos para jogos digitais: uma abordagem sistêmica. / João Marcos Aristides Almeida. – 2021.

13 f.

Orientador: Prof. Dr. Herman Martins Gomes.
Trabalho de Conclusão de Curso - Artigo (Curso de Bacharelado em Ciência da Computação) - Universidade Federal de Campina Grande; Centro de Engenharia Elétrica e Informática.

1. Eventos dinâmicos. 2. Jogos digitais. 3. Desenvolvimento de jogos. 4. Design sistêmico. 5. Geração de conteúdo. 6. Abstração de jogo digital. I. Gomes, Herman Martins. II. Título.

CDU:004.338.4(045)

Elaboração da Ficha Catalográfica:

Johnny Rodrigues Barbosa
Bibliotecário-Documentalista
CRB-15/626

JOÃO MARCOS ARISTIDES ALMEIDA

GERAÇÃO DE EVENTOS DINÂMICOS PARA JOGOS DIGITAIS: UMA ABORDAGEM SISTÊMICA

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Professor Herman Martins Gomes

Orientador – UASC/CEEI/UFCG

Professor Marcelo Alves de Barros

Examinador – UASC/CEEI/UFCG

Professor Tiago Lima Massoni

Professor da Disciplina TCC – UASC/CEEI/UFCG

Trabalho aprovado em: 25 de MAIO de 2021.

CAMPINA GRANDE - PB

RESUMO (ABSTRACT)

The player experience is an important factor in game development and needs to be instigated at all times. Creating diversified activities in the game that reinforces this factor is essential to add value and consistency to the game, as well as to encourage players to try it again. However, the manual generation of this type of content is costly and may result in a divergence from the projected experience. Therefore, using dynamic events becomes an interesting strategy to allow coherent diversity, as these activities are constantly changing due to interaction with the game systems. In this context, the systemic design technique applied to games proves to be a promising approach, as it uses the interaction between the game's own systems to create the player's experience. Thus, this work proposes a model for dynamic events generation using the systemic design as a foundation. For this, an abstraction of a digital game was created to demonstrate such a model. The abstraction is divided into simple systems, which use the interaction of its properties and attributes to create varied and coherent activities. The model was tested and results show that the more events are generated, the lower the dissimilarity rate between them.

Key-words: Content Generation, Dynamic Events, Game Development, Systemic Design.

Geração de Eventos Dinâmicos para Jogos Digitais: Uma Abordagem Sistêmica.

João Marcos Aristides Almeida
Universidade Federal de Campina Grande
Campina Grande, Paraíba, brasil
joao.almeida@ccc.ufcg.edu.br

Herman Martins Gomes (Orientador)
Universidade Federal de Campina Grande
Campina Grande, Paraíba, brasil
hmg@computacao.ufcg.edu.br

Resumo

A experiência do jogador é um importante fator no desenvolvimento de jogos e precisa ser instigada sempre. Criar atividades diversificadas no jogo que reforcem esse fator é fundamental para agregar valor e coerência ao jogo, assim como fomentar que jogadores voltem a experimentá-lo. No entanto, a geração manual deste tipo de conteúdo é custosa e pode resultar em divergência da experiência projetada. Dessa forma, utilizar eventos dinâmicos se torna uma estratégia interessante para permitir uma diversidade coerente, pois são atividades em constante alteração pela interação com os sistemas do jogo. Neste contexto, a técnica de design sistêmico aplicada a jogos se mostra uma abordagem promissora, pois utiliza a interação entre os próprios sistemas do jogo para criar a experiência do jogador. Assim, este trabalho propõe um modelo para a geração de eventos dinâmicos utilizando o design sistêmico como fundamento. Para isso, a abstração de um jogo digital foi criada para demonstrar tal modelo. A abstração é dividida em sistemas simples, que utilizam a interação de suas propriedades e atributos para criar atividades variadas e coerentes. O modelo foi testado e resultados mostram que quanto mais eventos são gerados, menor é a taxa de dissimilaridade entre eles.

Palavras-chave: Geração de Conteúdo, Eventos Dinâmicos, Desenvolvimento de Jogos, Design Sistêmico.

1. INTRODUÇÃO

Todo jogo tem um propósito, a experiência que o jogador deve ter enquanto joga, sendo tal propósito um fator importante que guia seu desenvolvimento. Segundo Schell [1] um jogo não é a experiência em si, mas quem permite ela existir. Tudo que é adicionado a um jogo deve suportar essa experiência e isso o deixa coerente.

Além disso, jogos precisam diversificar suas atividades para manter o jogador em um constante ciclo de aprendizagem. Para Koster [2], aprender é o principal fator da diversão, pois "jogos que falham em exercitar o cérebro se tornam chatos" (Koster, 2013, p.39). Diversidade de conteúdo também influencia a rejogabilidade, valor do jogo em ser jogado mais de uma vez mantendo sua diversão. Segundo Frattesi et al. [3], a rejogabilidade aumenta a vida útil de um jogo, além de impactar em suas vendas.

Entretanto, criar um conteúdo diverso e coerente, manualmente, é trabalhoso. Em uma postagem online de 2018, Koster[1] mostra que o custo de desenvolvimento de jogos está aumentando. Consequentemente, construir sistemas automáticos que facilitam a produção de conteúdo é uma estratégia viável para a redução de recursos, monetários e de tempo. Koster ainda cita o uso de conteúdo sistêmico para a redução de custos. O design sistêmico nos jogos se preocupa com a divisão do jogo em sistemas e a comunicação entre eles, para assim gerar inúmeras situações ao jogador. Por exemplo, havendo a interação entre os sistemas de terreno, movimento e clima, o jogador irá se mover de formas diferentes dependendo do estado do terreno, que é afetado pelo clima (como seco, escorregadio e movediço).

Eventos dinâmicos, nos jogos, não têm uma definição formal. No entanto, seu uso está relacionado a atividades com múltiplos resultados possíveis. Assim, atividades podem encadear outras atividades, ou mesmo aumentar seu escopo, dependendo de condições satisfeitas pelo jogador. Um exemplo disso é visto no jogo Guild Wars 2 [4], onde a interação dos jogadores altera o rumo do evento. Por sua vez, atividade é um termo amplo e se refere a qualquer tarefa dada ao jogador, desde combate com inimigos, coleta de itens, resolução de enigmas ou interação com algum elemento do jogo.

Para este trabalho, um evento dinâmico é uma atividade que, continuamente, se altera e evolui através das interações com o ambiente virtual do jogo. Dessa forma, o design sistêmico se mostra uma ótima alternativa para a geração desse tipo de conteúdo, o que pode acarretar atividades diversas e coerentes.

Este trabalho desenvolveu um modelo para a geração de eventos dinâmicos com uma abordagem sistêmica. Foi criada uma abstração de um jogo digital, com um mapa e várias regiões onde os eventos são criados. A abstração foi dividida em sistemas menores que foram submetidos à interação de suas propriedades e atributos. Assim, foi possível gerar atividades dinâmicas com características dos sistemas envolvidos.

Desta forma, os eventos gerados devem:

1. Ser formados pela interação dos sistemas do jogo;
2. Ser gerados continuamente;
3. Interagir com os sistemas do jogo;
4. Interagir com as ações do jogador;
5. Evoluir e se alterar.

[1] https://venturebeat.com/2018/01/23/the-cost-of-games/

2. FUNDAMENTAÇÃO TEÓRICA

A geração automática de conteúdo em jogos digitais tem sido estudada por várias décadas pela academia. Em um estudo recente, Soares et al. [5] combinou algoritmos genéticos com planejamento automatizado para gerar missões[2] baseadas em uma determinada estrutura de narrativa. Lee [6] utilizou redes de Petri para gerar o enredo de missões. Nestes casos, existia a preocupação da presença de coesão narrativa. Além disso, Doran [8] utilizou a classificação das estruturas de missões para gerar esse tipo de conteúdo, realizando uma análise de 3000 missões em quatro MMORPGs[3].

Hromada et al. [7] utilizou um sistema de blocos que se aproxima da ideia sistêmica apresentada neste trabalho. Cada bloco representa uma porção do conteúdo criado, eles são organizados como grafos acíclicos dirigidos onde o nó anterior carrega uma informação que o posterior irá utilizar.

No entanto, as abordagens utilizadas não englobam o design sistêmico, eventos dinâmicos, além de ter uma visão diferente para a atividade gerada.

3. CONCEITOS

Nas próximas subseções serão aprofundados os conceitos apresentados.

3.1 Geração de Conteúdo

Na indústria de desenvolvimento de jogos, a geração automática de conteúdo é dividida em duas principais categorias, aleatória e procedural[4]. Hendrikx [9] cita o uso de Geração de Números Pseudo-Aleatórios (GNPA) como uma das formas mais simples, e antigas, da geração procedural de conteúdo. Esse tipo de geração utiliza elementos já feitos e os organizam aleatoriamente para gerar conteúdo. Tal método pode ser visto em jogos como The Binding of Isaac [10] e Diablo 3 [11].

A geração procedural utiliza algoritmos formais para gerar o conteúdo do jogo [14]. Embora o uso de GNPA seja uma forma de geração procedural, a diferença entre os dois, na indústria, se apoia em duas características, escopo e complexidade. A geração aleatória tem elementos de maior escopo, sendo mais abrangentes, e utilizam algoritmos mais simples. Em contrapartida, o método procedural utiliza regras mais complexas de geração e possui elementos de menor escopo. Um exemplo disso pode ser visualizado comparando-se a geração de mapas nos jogos Minecraft e The Binding of Isaac. Neste último, salas inteiras já estão prontas e são organizadas para formar um mapa novo, já Minecraft possui regras para utilizar cada parte que compõe um território (como pedra, água e madeira, por exemplo) para formar um mapa. O método procedural é visto em jogos como Minecraft [12] e Dwarf Fortress [13].

3.2 Eventos Dinâmicos

Por sua definição, "dinâmico" é algo em constante mudança, incluindo evolução, movimentação e alteração. Atividades dinâmicas nos jogos apresentam uma, ou mais, dessas características. Assim, ao contrário de um conteúdo estático, eventos dinâmicos podem alterar seus objetivos, ampliar seu escopo e encadear diversos resultados dependendo da interação do jogador ou dos sistemas do jogo.

Embora o uso do termo seja recente, o conceito de um evento dinâmico pode ser visto desde o final da década de 1990 com jogos como Ultima Online [15] e EverQuest [16]. Tais jogos utilizavam eventos promovidos por membros da própria equipe de distribuição do jogo, um Game Master (GM). Por se tratar da criatividade humana e serem realizadas em tempo real, era possível entregar uma experiência única e imprevisível aos jogadores. No entanto, essas atividades não podiam ocorrer em qualquer horário, pois dependiam da presença de um GM, limitando a experiência a alguns jogadores.

O uso automático desse tipo de conteúdo se tornou popular com o jogo Guild Wars 2 [4]. Nesta instância, eventos previamente criados têm a capacidade de aumentar, ou diminuir, requisitos dependendo da quantidade de jogadores participantes. Além disso, a interação dos jogadores pode trazer impactos ao mundo virtual e desencadear outros eventos.

3.3 Design sistêmico

Design sistêmico se caracteriza pela divisão do jogo em sistemas e a comunicação entre eles. Entretanto, sistemas simples estão, e sempre estiveram, presentes em jogos digitais, o foco da abordagem sistêmica está em sistemas complexos.

Segundo Adams [17], um sistema complexo é aquele que possui muitas partes que, mesmo simples individualmente, ao serem postas em conjunto mostram um comportamento imprevisível e difícil de explicar, olhando cada parte separadamente (Adams, 2012 p.45). Não apenas isso, Salen et al. [18] ainda acrescenta que esses sistemas precisam ter conexões suficientes entre eles de modo a permitir comportamentos não aleatórios.

Sellers [19] utiliza a visão de sistemas nos jogos dividindo-os em "partes", "laços" e "todo". Dessa forma, jogos são feitos por partes, que juntas formam laços de interação, criando um todo persistente e emergente. A principal finalidade desse tipo de design é a própria emergência. Segundo Salen, emergência é gerada por sistemas complexos, onde o todo apresenta propriedades que suas partes não possuem.

Dessa forma, emergência de jogabilidade fornece ao jogo uma alta variedade consistente de interação. Esse tipo de design é utilizado em jogos como Dwarf Fortress e The Legend of Zelda: Breath of the Wild [20].

4. SOLUÇÃO

Esta seção irá aprofundar a solução desenvolvida neste trabalho. Para isso, foram criadas subseções, onde 4.1 explica o conceito da solução, 4.2 detalha os sistemas utilizados, 4.3 relata o funcionamento e, por fim, 4.4 exemplifica o uso do modelo em jogos reais.

4.1 Visão Geral

Para criar eventos dinâmicos, o modelo adotado nesta pesquisa separa as informações necessárias entre os sistemas que compõem o jogo. Dessa forma, a comunicação entre tais sistemas é utilizada para a criação das atividades, onde todo sistema tem uma propriedade ou atributo que estará presente em um evento. Além

[2] Conjunto de atividades associadas a uma narrativa.

[3] *Massive Multiplayer Online Role-Playing Game*.

[4] https://www.gamasutra.com/blogs/JoshBycer/20150807/250760/Procedural_vs_Randomly_Generated_Content_in_Game_Design.php

disso, a parte dinâmica dos eventos também é criada por meio da comunicação dos sistemas, onde cada componente contribui para a alteração e evolução de eventos.

Para demonstrar esse processo, a abstração de um jogo digital foi desenvolvida através da ferramenta Godot Engine[5]. Tal ferramenta permite o uso de várias linguagens de programação, para este trabalho foi utilizada a GDScript, uma linguagem nativa do software com uma sintaxe semelhante a de Python.

4.2 Sistemas

A abstração consiste de uma tela de 1280 x 720 pixels dividida em quatro regiões. Os sistemas mais básicos para que este modelo funcione são separados em:

- Região: diversidade do mundo virtual;
- Vegetação: cenário do mundo virtual;
- População: quem habita o mundo virtual;
- Dificuldade: controle de experiência do jogador;
- Objetivo: diversidade de atividades;
- Evento: construção de atividades.

Tais sistemas serão aprofundados nas subseções a seguir.

4.2.1 Região

Representa um espaço no mapa, exercendo controle sobre a quantidade de populações e objetivos gerados. Possui uma dificuldade, que é variável, uma vegetação e dois tipos de populações fixas e únicas para cada região.

Figura 1: Mapa e Regiões

A dificuldade é representada pela cor em cada região. Além disso, cada região tem um limite de quatro eventos antes que novos sejam gerados.

Região	Informações	Máximo:	
		População	Objetivo
Região 1	População 1	2	3
	População 2	3	
	Vegetação 1	-	
Região 2	População 3	3	3
	População 4	4	
	Vegetação 2	-	
Região 3	População 5	4	4
	População 6	5	
	Vegetação 3	-	
Região 4	População 7	5	4
	População 8	5	
	Vegetação 4	-	

Tabela 1: Distribuição de Populações e Vegetações

Nota-se que o hífen utilizado para a vegetação representa que esta não é gerada virtualmente, mas sim um atributo pertencente à região.

4.2.2 Dificuldade

Representa o nível de desafio ao jogador e exerce controle sobre todas as métricas para conclusão de eventos. Existem ao todo cinco níveis de dificuldade, onde quatro deles são utilizados inicialmente nas regiões e uma dificuldade especial, somente aplicada através da evolução de eventos. Cada dificuldade também controla a velocidade de evolução dos eventos e seu crescimento por união de objetivos (ver Subseção 4.2.6).

Nível de Dificuldade	Máximo de Uniões	Região de Origem
1	1	Região 1
2	1	Região 2
3	2	Região 3
4	3	Região 4
5	4	-

Tabela 2: Níveis de Dificuldade e Máximo de Uniões

Nota-se que o hífen utilizado para a dificuldade 5 demonstra que ela não é aplicada a nenhuma região, inicialmente.

4.2.3 Objetivo

Representa um tipo de atividade a ser executada pelo jogador. Existem cinco tipos de objetivos, que foram escolhidos por representar atividades comuns encontradas em jogos eletrônicos, tais quais Minecraft e Guild Wars 2. Eles utilizam informações fornecidas por diferentes sistemas, além das métricas de conclusão, controladas pela dificuldade da região. Todas as regiões podem gerar qualquer tipo de objetivo, aleatoriamente por meio do uso de GNPA.

Quando um objetivo colide com uma população um evento pode ser criado, a depender da população (ver Subseção 4.2.5). Ao serem criados, objetivos escolhem a localização de uma das populações ou eventos existentes, também por GNPA, e se movem nessa direção até formarem um evento ou serem apagados ao sair da região.

[5] https://godotengine.org/

Tipo (ID)	Foco do Objetivo	Sistema Fornecedor	Métrica
Combate (1)	Inimigo Normal	População	Quantidade
	Inimigo Poderoso		
Defesa (2)	Defesa Móvel		Tempo
	Defesa Imóvel	Vegetação	
Coleta (3)	Item		Quantidade
Movimento (4)	Ponto Destino		Tempo
Interação (5)	Alvo de interação		Quantidade
		População	

Tabela 3: Tipos de Objetivos e Informações

Alguns dos tipos de objetivos podem receber informações para seu foco de mais de um sistema ou possuem mais de um foco, como é o caso de “Combate”, “Defesa” e “Interação” (Tabela 3). Nestes casos, um dos sistemas, ou focos, será escolhido por GNPA.

Objetivo	Foco	Métrica
Combate	População 1	Dificuldade 1
Inimigo Normal	Normal-1	2
Condição de vitória: “Derrotar 2 Normal-1”		

Tabela 4: Exemplo de objetivo

Objetivos são representados por figuras quadradas e um número de identificação (ID) de tipo, sua cor indica qual população aceita o tipo de objetivo (ver Subseção 4.2.5). Dessa forma, quando as duas populações de uma região aceitam um mesmo tipo de objetivo, este apresentará cores de ambas as populações.

4.2.4 Vegetação

Representa a natureza de uma região e guarda informações para a formação de eventos, além de oferecer um bônus à sua evolução. Existem quatro vegetações, uma para cada região, sendo imutáveis.

4.2.5 População

Representa os habitantes de uma região, guarda informações para a formação de eventos, além de determinar que tipos de objetivos podem ser utilizados em um evento. As populações são representadas por figuras circulares, contendo um código para identificação e uma cor única.

População	Objetivos Aceitos		
	1º	2º	3º
População 1	Combate	Coleta	Interação
População 2	Combate	Defesa	Movimento
População 3	Combate	Defesa	Interação
População 4	Coleta	Movimento	Interação
População 5	Combate	Defesa	Coleta
População 6	Defesa	Movimento	Interação
População 7	Combate	Coleta	Movimento
População 8	Defesa	Coleta	Interação

Tabela 5: Populações e Objetivos Aceitos

Dessa forma, a "População 2" só se unirá a objetivos dos tipos “Combate”, “Defesa” e “Movimento”, qualquer evento que a utilize também só aceitará esses tipos.

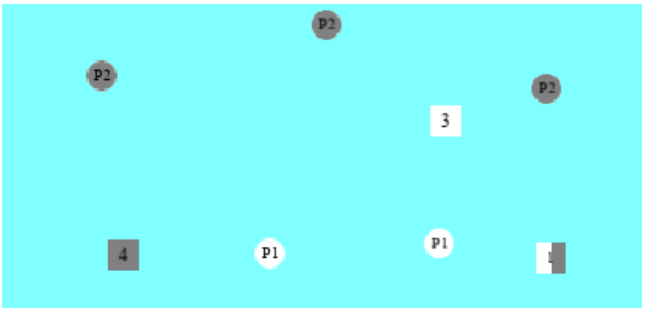

Figura 2: População 1 e 2 e Objetivos (Região 1)

4.2.6 Evento

Representa um evento dinâmico e se forma pela interação entre os sistemas. Possui uma população, vegetação, dificuldade e uma lista de objetivos. São representados por uma figura circular de cor cinza. Possuem um identificador de evento (representado pela letra “E”), os IDs dos objetivos aceitos pela população presente e um indicador do nível de evolução.

Ao ser criado, cada evento recebe informações dos outros sistemas, preenchendo todas as informações necessárias para serem uma atividade, conforme a Tabela 3 e 4 ilustram.

Um evento pode sofrer alteração aumentando seu escopo através da união de objetivos, a depender da população presente. Neste processo, um evento pode adicionar outros objetivos e se tornar uma atividade de múltiplas condições de vitória, sendo indicado pelo aumento no tamanho do evento. Além disso, um evento também pode evoluir com o passar do tempo, utilizando informações da dificuldade e vegetação presentes para determinar a velocidade. São três níveis de evolução, indicados pela tonalidade da cor do evento (Figura 3), que aumentam as métricas para conclusão do evento.

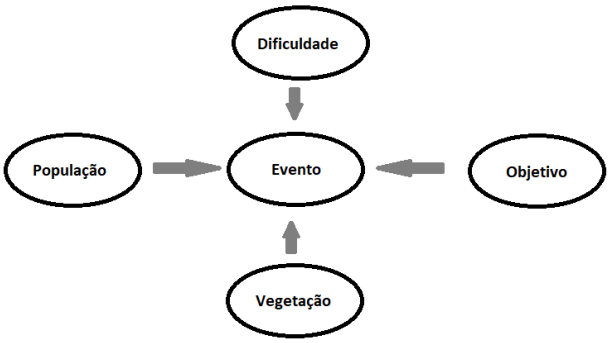

Figura 3: Interação entre Sistemas para criação de Evento

Ao atingir o nível máximo de evolução e de uniões de objetivos, um evento altera a dificuldade da região em que está inserido para um nível superior. Quando esse evento é desfeito, a dificuldade da região volta ao seu nível original.

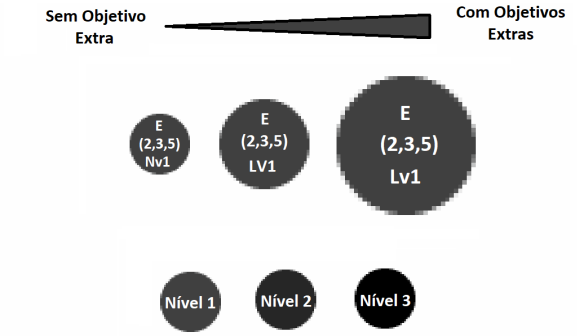

Figura 3: Alteração e Evolução de Objetivos

Além dos sistemas mencionados, existe uma entidade externa ao jogo, um jogador humano que interage com a abstração desenvolvida. Vale notar que esta abstração não aceita múltiplos jogadores. A interação com o jogador é representada por dois estados, vitória e derrota. Ao ser completado, um evento é desfeito, caso contrário ele recebe um bônus de evolução, proporcional a seu nível.

Esta interação é representada através do uso de um periférico Mouse, onde o botão esquerdo sinaliza a vitória do jogador sobre o evento, e este é concluído. Já o botão direito sinaliza a derrota do jogador e o evento persiste.

4.3 Funcionamento

Ao ser inicializada a abstração, todas as regiões irão gerar suas respectivas populações e objetivos. Tais objetivos começarão a se mover pela região (Figura 4).

Ao encontrar os devidos objetivos, cada população inicia a geração de um evento, utilizando dados de ambos e da vegetação e dificuldade que estão inseridos. Então, os eventos criados passarão a evoluir e se alterar, aumentando seu escopo (Figura 5).

Em seguida, alguns eventos atingem seu limite de crescimento e alteram a dificuldade da região (Figura 6).

Finalmente, quando tais eventos são completados, as regiões voltam a seus estados iniciais de dificuldade e o ciclo se repete (Figura 7).

A abstração continuará a ser executada até ser finalizada por intervenção do jogador.

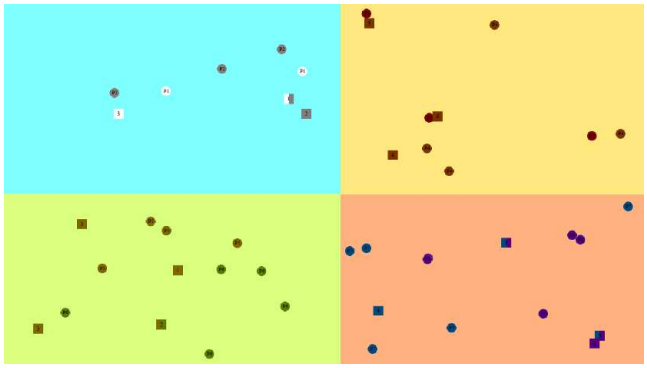

Figura 4: Criação de Populações e Objetivos

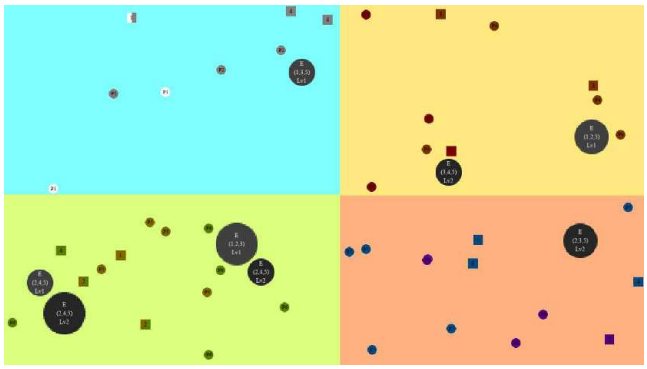

Figura 5: Formação e Crescimento de Eventos

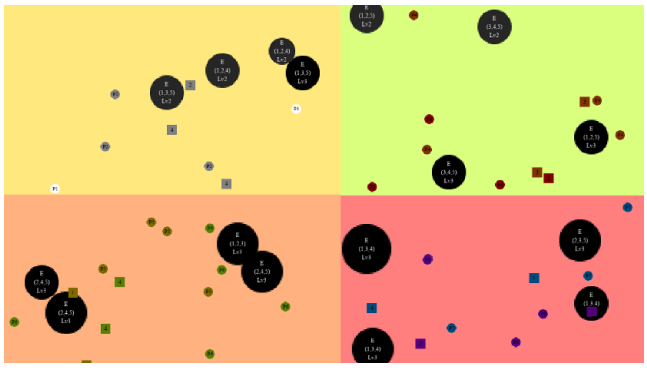

Figura 6: Mudança de Dificuldades

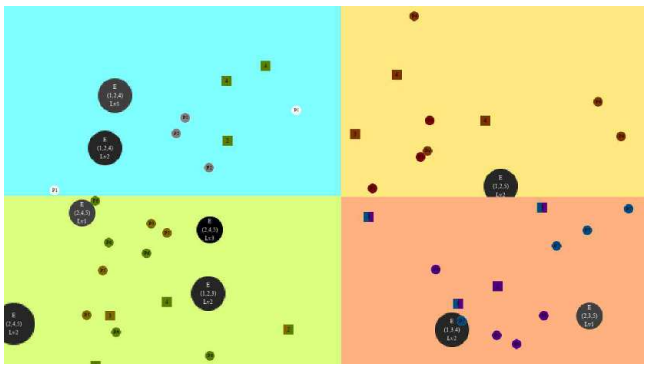

Figura 7: Níveis de Dificuldade Restaurados

4.4 Utilidade em Jogos

A aplicação desenvolvida neste trabalho serve exclusivamente para demonstrar como o modelo proposto funciona, deixando os sistemas bastante simples. No entanto, esta solução pode ser utilizada em qualquer jogo digital que se utilize o design sistêmico. Isso ocorre pois este modelo se baseia puramente na distribuição de informações entre os sistemas que compõem o jogo.

Por exemplo, o modelo proposto neste trabalho poderia ser utilizado no gênero RPG[6] eletrônico para criar atividades complementares à progressão no jogo. Neste caso, todas as populações do jogo (e.g. facções em um jogo típico de RPG), sejam inimigas ou aliadas ao jogador, formariam o sistema de população do modelo. A vegetação implementada na aplicação poderia ser vista como biomas e ambientes simulados em jogos digitais. Os territórios de cada população em um jogo digital típico, como The Legend of Zelda: Breath of the Wild e Guild Wars 2, seriam as regiões deste modelo. O sistema de dificuldade da aplicação desenvolvida seria adaptado à dificuldade desejada do próprio jogo. Por fim, o sistema de objetivos da aplicação desenvolvida seria composto pelas atividades principais do jogo em questão.

5. AVALIAÇÃO

Esta seção detalha o processo de avaliação, começando por como os testes foram feitos (Seção 5.1) e concluindo com os resultados encontrados (Seção 5.2).

5.1 Planejamento

A avaliação do modelo se preocupou com a variedade dos eventos gerados e como a alteração de tais eventos impacta nessa variedade em uma região. Desta forma, os eventos analisados então divididos em dois estados, "Alterado" e "Inicial". Este último é composto por eventos recém-criados, quando só apresentam um único objetivo. Já o estado "Alterado" é composto por eventos em seu nível máximo de crescimento, possuindo mais de um objetivo (indicado pelo máximo de objetivos em cada gráfico).

Foram realizados quatro testes, incrementando-se o número de eventos criados em cada região.

- Teste 1: 4 eventos por região
- Teste 2: 8 eventos por região
- Teste 3: 12 eventos por região
- Teste 4: 16 eventos por região

A medida de tempo utilizada é chamada de "geração" e representa o período de tempo desde que os eventos são criados até atingirem seu limite de crescimento. Foram usadas 20 gerações para cada teste. A medida usada para classificar a variedade é uma porcentagem que representa a dissimilaridade entre os eventos de uma região. Tal medida é calculada para cada região em cada geração, sendo definida por:

$$Di = \frac{QOD}{QOT}$$

Em que:

- *Di* = Dissimilaridade
- *QOD* = Quantidade de Objetivos Diferentes
- *QOT* = Quantidade de Objetivos Totais

Para isso, o tipo do objetivo e seu foco são os parâmetros principais para determinar a semelhança, por determinarem a natureza da atividade. Assim, um exemplo de comparação seria:

	Objetivo 1	**Objetivo 2**
Evento 1:	"Coleta","Item-2"	"Combate","Inimigo-Normal-2"
Evento 2:	"Coleta","Item-2"	"Defesa","Defesa-Movel-1"
***Di*:**	$\frac{3}{4} = 75\%$	

Tabela 6: Exemplo de Cálculo de Dissimilaridade entre Eventos. Os 3 objetivos diferentes são "Coleta,Item-2", "Combate,Inimigo-Normal-2" e "Defesa,Defesa-Movel-1".

5.2 Resultados

Nesta seção são apresentados os gráficos para as regiões 1 e 4 em todos os testes. Entretanto, essas regiões destacam bem o impacto das alterações nos eventos. Nos dois primeiros testes (Figuras 8 e 9) é possível perceber quedas nas porcentagens *Di* ao se observar o estado "Inicial" em cada região. Essa diminuição se acentua ainda mais no estado "Alterado", principalmente na região 4.

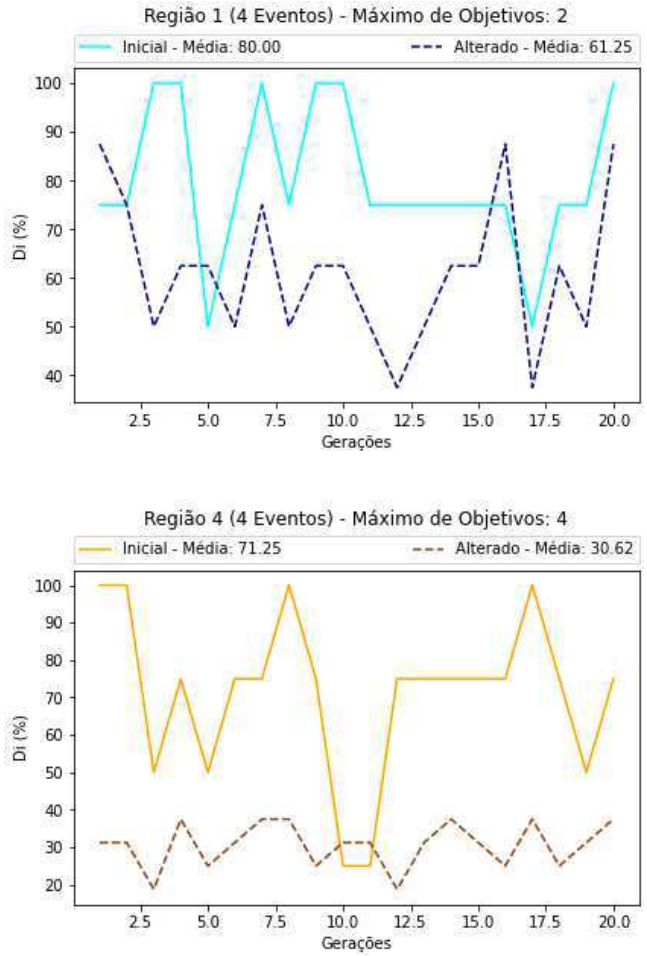

Figura 8: Teste 1 - 4 Eventos Gerados por Região

[6] sigla para *Role-Playing Game*

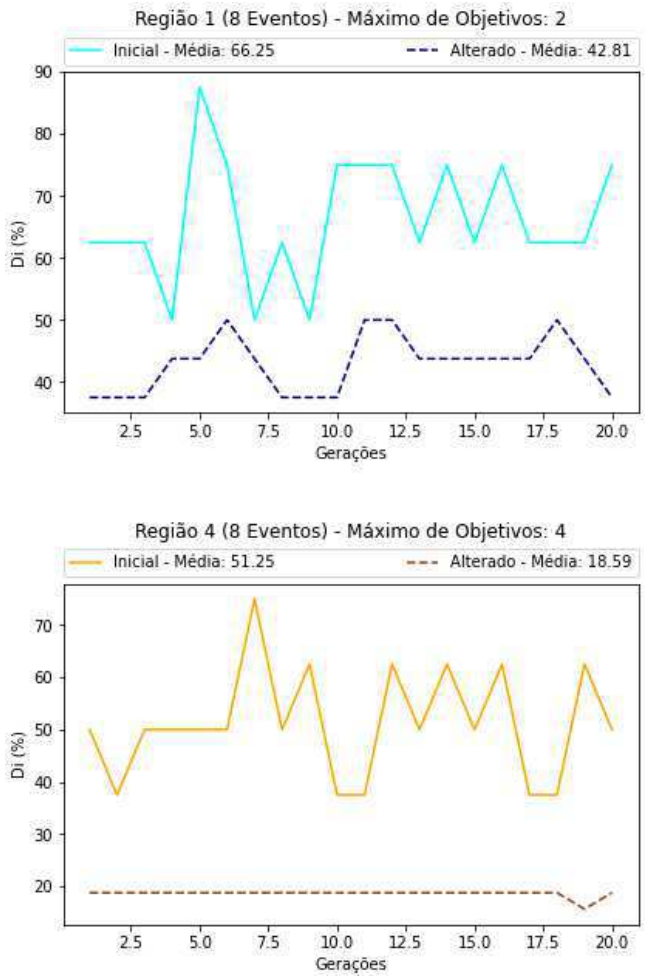

Figura 9: Teste 2 - 8 Eventos Gerados por Região

Isso ocorre por causa do número de uniões da região 4 ser maior que o da região 1. Dessa forma, enquanto cada evento da região 1 têm dois objetivos no estado "Alterado", os da região 4 têm quatro, aumentando as chances de um objetivo repetido ser adicionado.

Outro aspecto a mencionar são os tipos de objetivos que podem ser adicionados a cada evento, sendo três dos cinco totais (Subseção 4.2.5, Tabela 5). Esse fator também influencia a queda da taxa de dissimilaridade. Tendo em vista o número de objetivos nos eventos alterados da região 4, sempre existirá um objetivo repetido. Essa foi uma decisão tomada para favorecer a dissimilaridade entre as populações existentes, mas pode ser alterada para melhorar a variedade de eventos.

Além disso, quanto maior o número de eventos gerados, a taxa de dissimilaridade entre os eventos alterados cai ainda mais (Figuras 10 e 11). A diferença entre os testes 1 e 4 (Figuras 8 e 11) é enorme, havendo uma queda de mais da metade nas médias de dissimilaridade.

Dessa forma, pode-se concluir que as alterações têm um impacto negativo na diversidade de eventos, sendo os seguintes elementos responsáveis por isso:

- Quantidade de eventos gerados;
- Quantidade de objetivos em cada evento;
- Quantidade de tipos de objetivos a serem utilizados pelas populações.

Observando-se os resultados dos gráficos, o uso de quatro eventos têm melhores taxas de dissimilaridade para eventos alterados. Para eventos iniciais, no entanto, o uso de oito eventos seria mais interessante, pois o custo para se dobrar o valor anterior não é tão alto, ficando ainda acima dos 50% em ambas regiões 1 e 4.

Entretanto, é preciso notar que seria necessário uma aplicação do modelo em um jogo real, juntamente com a realização de testes com jogadores, para avaliar o real impacto dessas alterações, já que podem existir pesos diversos na classificação de eventos diferentes.

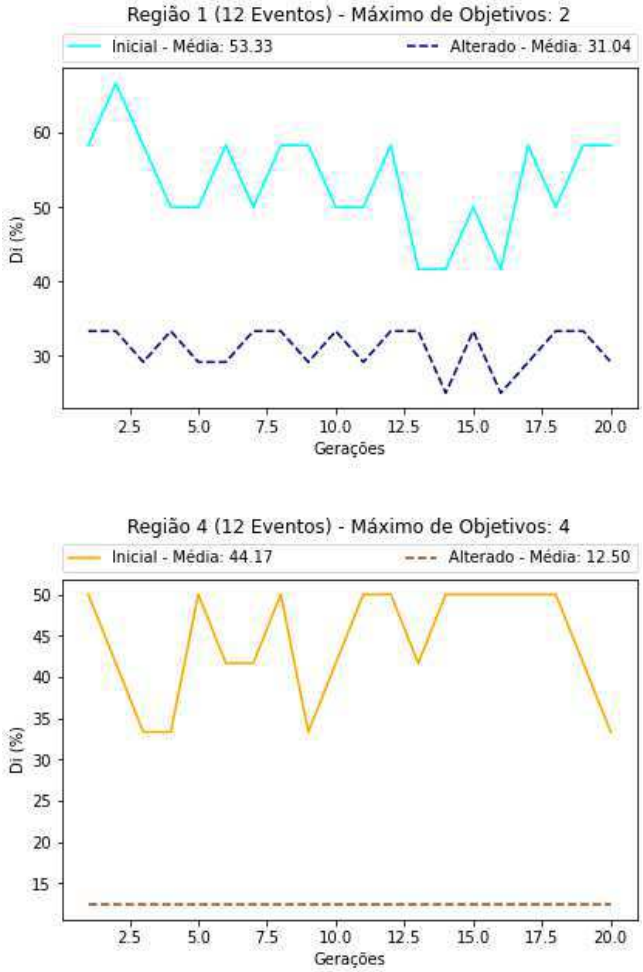

Figura 10: Teste 3 - 12 Eventos Gerados por Região

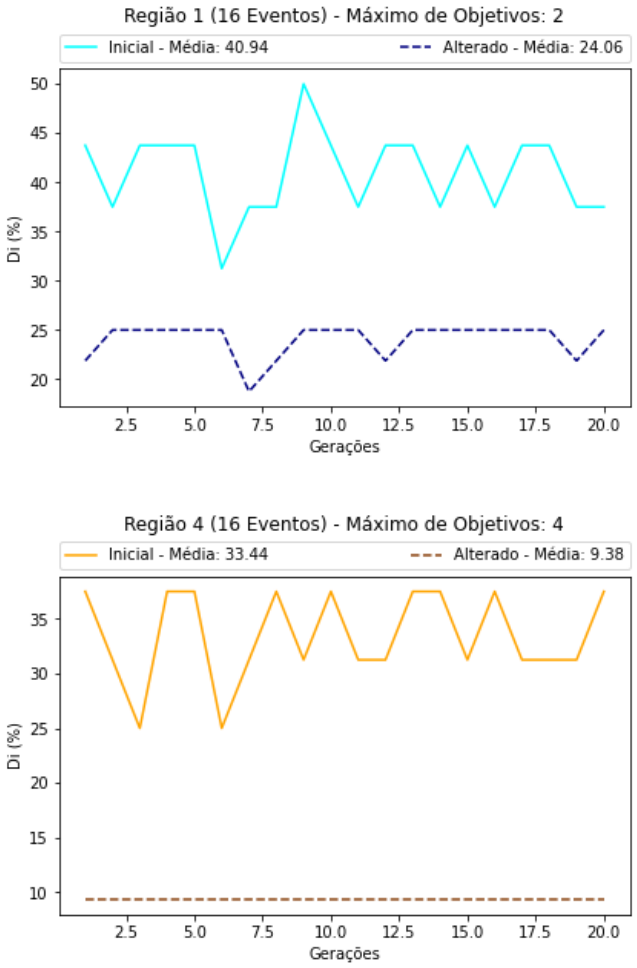

Figura 11: Teste 4 - 16 Eventos Gerados por Região

6. CONCLUSÃO

Os resultados reportados neste trabalho mostram que o modelo proposto consegue criar eventos dinâmicos, no entanto, as alterações sofridas por estes eventos reduzem drasticamente a taxa de dissimilaridade entre eles. Além disso, uma validação mais completa deveria envolver a aplicação do modelo proposto em um jogo real, já que a pura dissimilaridade entre os eventos pode não refletir na experiência do jogador.

Existem várias possibilidades para se melhorar o modelo proposto, além do simples aumento no número de objetivos possíveis. Uma adição simples seria variar o número máximo de objetivos em cada evento, utilizando a quantidade de eventos gerados como um fator para determinar essa variação. Assim, em instâncias onde houvesse um menor número de eventos gerados, existiria um maior número de objetivos utilizados em cada evento.

Além disso, aumentar o impacto que a interação do jogador traz aos eventos pode beneficiar o modelo, já que seria uma camada de alteração a mais, e ligada às ações do jogador. Também, substituir o uso de números pseudo-aleatórios por algum algoritmo mais robusto, que favoreça a diversidade de eventos, é uma estratégia viável.

Outra opção seria a implementação de um nível de evolução de maior hierarquia, isso traz a possibilidade de uma mudança radical na parte dinâmica dos eventos. A existência de “Super Eventos” foi uma possibilidade projetada no início deste trabalho, através da movimentação de eventos pelo mapa, mas isso foi descartado ao decorrer do desenvolvimento da abstração em função do tempo disponível para desenvolvimento.

Uma última estratégia seria o aumento da interação entre os sistemas existentes, isso criaria ainda mais conexões entre as partes que formam o jogo, melhorando o fator de emergência do mesmo, consequentemente, aumentando a taxa de dissimilaridade.

No mais, este modelo pode ser implementado em outros jogos que usem o design sistêmico. Por se tratar de um modelo mais relacionado ao design do jogo, ele pode facilmente ser alterado para atender as características de outros jogos.

7. REFERÊNCIAS

[1] J. Schell, “The Art of Game Design: A Book of Lenses, Second Edition”, 2ª Edição, CRC Press, 2014, página 11.

[2] R. Koster, “Theory of Fun for Game Design”, 2ª Edição, O'Reilly Media, 2013.

[3] T. Frattesi, D. Griesbach, J. Leith, T. Shaffer, J. DeWinter, “Replayability of Video Games”, IQP, Worcester Polytechnic Institute, Worcester, 2011.

[4] ArenaNet, “Guild Wars 2”,https://www.guildwars2.com/en/, Último acesso: 16/05/2021.

[5] E. Soares de Lima, B. Feijó e A. L. Furtado, "Procedural Generation of Quests for Games Using Genetic Algorithms and Automated Planning," 2019 18th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames), 2019, pp. 144-153.

[6] Y. Lee and S. Cho, "Context-Aware Petri Net for Dynamic Procedural Content Generation in Role-Playing Game," in IEEE Computational Intelligence Magazine, vol. 6, no. 2, pp. 16-25, May 2011.

[7] T. Hromada, M. Cerný, M. Bída e C. Brom, “Generating Side Quests from Building Blocks”, International Conference on Interactive Digital Storytelling, 2015.

[8] J. Doran e I. Parberry, “A Prototype Quest Generator Based on a Structural Analysis of Quests from Four MMORPGs”, In Proceedings of the 2nd International Workshop on Procedural Content Generation in Games, pp1-8, New York, 2011.

[9] M. Hendrikx, S. Meijer, J. V. D. Velden e A. Iosup, “Procedural Content Generation for Games: A Survey”, ACM Transactions on Multimedia Computing Communications and Applications, Volume 9, Artigo 1, 2013.

[10] E. McMillen e F. Himsl, “The Binding of Isaac”, https://bindingofisaac.com/, Último acesso: 16/05/2021.

[11] E. Blizzard, “Diablo 3.”, https://us.diablo3.com, Último acesso: 16/05/2021.

[12] Mojang, “Minecraft,” https://minecraft.net, Último acesso: 16/05/2021..

[13] Bay 12 Games, “Dwarf Fortress”, http://www.bay12games.com/dwarves/, Último acesso: 16/05/2021.

[14] G. Smith, “An Analog History of Procedural Content Generation.” FDG, 2015.

[15] Origin Systems, “Ultima Online”, https://uo.com/, Último acesso: 16/05/2021.

[16] Verant Interactive e 989 Studios, “EverQuest”, https://www.everquest.com/home, Último acesso: 16/05/2021.

[17] E. Adams, J. Dormans, “Game Mechanics: Advanced Game Design (Voices That Matter)”, 1ª Edição, Pearson Education, 2012.

[18] K. Salen, E. Zimmerman, “Rules of Play: Game Design Fundamentals”, 1ª Edição, The MIT Press, 2003.

[19] M. Sellers“Advanced Game Design: A Systems Approach”, 1ª Edição, Addison-Wesley Professional, 2017.

[20] E. P. D. Nintendo, “The Legend of Zelda: Breath of the Wild”,https://www.nintendo.com/pt_BR/games/detail/the-legend-of-zelda-breath-of-the-wild-switch/, Último acesso: 16/05/2021.