IAs generativas no *modding* de plataformas de jogos digitais[1][2]

Hugo Pereira Andrade[3]
Universidade Federal de Minas Gerais, Minas Gerais, MG

Resumo

Jogos digitais podem ser entendidos como softwares apropriados pelo ato de jogar (Sicart, 2014, 2023). Neste artigo partimos deste pressuposto para analisar *Mantella*, um *mod* em desenvolvimento para *The Elder Scrolls V: Skyrim* que utiliza três inteligências artificiais para transformar a forma como o jogador interage com personagens não jogáveis. Esta análise parte da perspectiva do jogo como o resultado de uma série de estruturas tecnológicas, aqui observadas como plataformas e formas de uso. A partir desta observação percebemos que o ato de modificar um jogo e suas plataformas pode ser entendido como uma forma de *play*, deslocando a própria definição do que é o software do jogo e o local que ele ocupa entre outras plataformas, enquanto a aplicação de inteligências artificiais dá evidência à questões éticas e práticas que se relacionam diretamente com a infraestrutura dos jogos digitais.

Palavras-chave

Modding; Inteligência artificial; Jogos digitais; Game development; Plataformas.

Infraestruturas de jogo

Devido à multidisciplinariedade do tema de jogos digitais e a constante sobreposição de conceitos com nomes e traduções similares ou iguais entre as áreas, é necessário apresentar alguns termos e clarificar como eles serão utilizados neste artigo antes de iniciarmos a discussão. Estes termos são o que chamaremos de "jogo", "software" e "plataformas", que aqui tratam de separações de características específicas que compõem um jogo digital e que normalmente são invisibilizadas ou pressupostas durante o consumo cotidiano destas mídias.

Neste artigo escolhemos empregar o termo "jogo digital" – ao invés de seus vários quase sinônimos – para deixar claro uma relação direta de dependência que estes objetos aqui observados têm com os estudos de software. É claro que a interdisciplinaridade dos estudos de jogos não depende do uso de termos específicos, mas destacamos esta escolha para demarcar que adotamos uma perspectiva que entende estes objetos primeiro como um software – observando suas camadas materiais de código, hardware e outras plataformas – e depois como jogo digitalizado – que ocorre a partir do uso deste software, não de sua concepção. A partir

---

[1] Trabalho apresentado no GP Games, XXIII Encontro dos Grupos de Pesquisas em Comunicação, evento componente do 46º Congresso Brasileiro de Ciências da Comunicação.

[2] O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de financiamento 001.

[3] Mestrando em Comunicação Social pelo Programa de Pós-Graduação em Comunicação Social da Universidade Federal de Minas Gerais. Email: hugope.andrade@gmail.com.

disto, entendemos que camadas que são centrais para a definição de como o jogo ocorre, como a funcionalidade e o *gameplay* (Konzack, 2002) por exemplo, são efeitos da transformação do software em jogo, não do software em si. Como coloca Günzel:

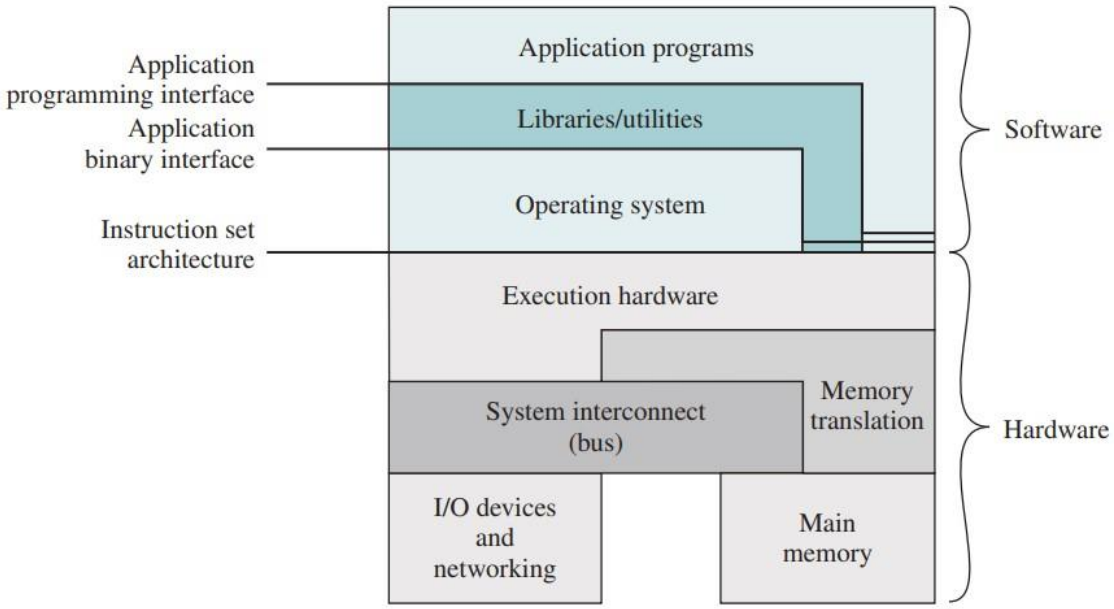
> [...] um jogo (programa) de computador não difere essencialmente de uma simulação ou um editor de texto no aspecto técnico. Mas um jogo de computador *difere* destes de acordo com como o programa é *usado*. (Günzel, 2012, p. 32 *apud* Freitas, 2017, p. 73).

Desta forma, entendemos o software do jogo digital apenas como um objeto onde o jogar se manifesta. Miguel Sicart (2014) entende este jogar ("*play*") como um meio termo entre a criação e a destruição, como força apropriadora e disruptiva – assim como as práticas de modificação que observaremos neste artigo – sendo ele uma forma de interagir com o mundo.

O software, em um entendimento simplificado, é apenas uma série de linhas de código que organiza atividades de hardware. Porém, o que realmente compõe e viabiliza o software de um jogo digital é uma infraestrutura complexa de camadas e plataformas interdependentes, tanto em hardware quanto em software, que permitem a execução de softwares jogáveis.

Uma das principais características que os jogos digitais herdam do software é esta dependência de uma série de plataformas e suas conexões. Não apenas o software depende de estruturas de hardware – geralmente organizadas no formato de plataformas – para seu funcionamento, mas também os próprios softwares são capazes de compor estruturas para outros softwares. Estas estruturas podem ser divididas em diversas categorias baseadas não apenas no funcionamento destes objetos de hardware e software, mas também no uso aplicado à eles. Podemos tomar como exemplo a infraestrutura apresentada por William Stallings (2018):

Figura 1 – Estruturas de software e hardware



Fonte: Stallings, 2018, p. 70.

> Se alguém fosse desenvolver um programa de aplicação como um grupo de instruções de máquina responsável por controlar o hardware do computador completamente, enfrentaria uma tarefa extremamente complexa. Para facilitar esta atividade, é fornecido um conjunto de programas de sistema. Alguns destes programas são chamados de utilitários ou programas de biblioteca.[4] (Stallings, 2018, p. 70, tradução nossa).

Cada ponto desta infraestrutura é necessário para tornar possível o funcionamento dos programas de aplicação, categoria na qual os jogos digitais se encontram – pelo menos na maior parte do tempo. Stallings adota o termo "camadas" para se referir a estas divisões na infraestrutura, porém, neste artigo utilizamos o termo "plataformas", tanto para fazer alusão ao formato de dependências estacadas no qual estas estruturas funcionam quanto pelo alinhamento com o entendimento de Clara Fernández-Vara para o termo quando observando jogos digitais, se referindo à todas as tecnologias necessárias para o funcionamento do jogo, "não apenas o hardware [...], mas também motores de jogos [...] ou até plataformas de desenvolvimento"[5] (Fernández-Vara, 2015, p. 70, tradução nossa).

É importante destacar que mesmo dentro dos estudos dos jogos digitais o entendimento do termo "plataformas" varia bastante. Chia *et al*. (2020), por exemplo, partem da perspectiva da plataformização para analisar objetos similares ao que observamos neste artigo, em especial

[4] No original: "If one were to develop an application program as a set of machine instructions that is completely responsible for controlling the computer hardware, one would be faced with an overwhelmingly complex undertaking. To ease this chore, a set of system programs is provided. Some of these programs are referred to as utilities, or library programs".

[5] No original: "not only the hardware (consoles vs. PC; portable gaming consoles vs. smartphones), but also game engines (Unreal, Aurora Toolset, Source), or even development platforms (Unity, Flash)".

o motor de jogo *Unity*. Porém, eles o classificam como plataforma não pela forma como o motor estrutura o jogo digital, e sim pelo modo no qual ele "claramente opera como uma plataforma pelo modelo de negócio focado em expandir efeitos de rede e coleta de dados [...]"[6] (Chia *et al*., 2020, p. 22, tradução nossa). Ou seja, um mesmo objeto é capaz de carregar múltiplas facetas que representam modelos de plataformização, sejam estas facetas relacionadas entre si ou não, muitas das quais não serão observadas neste artigo.

Além das plataformas estruturais externas, jogos digitais ainda trazem plataformas "internas", que fazem parte da composição do software finalizado, mas que ainda realizam funções similares às bibliotecas e utilitários, como os próprios motores citados por Fernández-Vara. Porém, ao contrário das plataformas externas, que possuem portas de contato para múltiplos programas de aplicação e interconexões, as plataformas internas são fechadas, servindo ao propósito único de tornar possível e funcional o software do que vem a ser um jogo digital.

> [...] a forma atual e tradicional de se desenvolver jogos digitais é modular (devido às suas complexidades extremas), mas no nível do código fonte. Após a compilação é criado um software monolítico, no qual apenas um desenvolvedor experiente pode substituir módulos internos.[7] (Hassam; Kara; Belqasmi, 2014, p. 109, tradução nossa).

Apesar deste modelo, é cada vez mais comum que jogos digitais incluam portas de acesso que facilitem o processo de modificação deste software (Filho, 2018). Este tipo de estrutura também pode ser interna – dentro da própria construção e organização dos módulos que compõem o jogo –, externa – como na disponibilização de outros softwares que garantem acesso e permissão aos arquivos inicialmente percebidos como "monolíticos" –, ou uma união dos dois.

Playing software

> Softwares usam regras e processos para permitir que computadores performem operações em representações de dados e jogos digitais usam regras e processos para permitir que usuários interajam com modelos computacionais

[6] No original: "[…] clearly operates as a platform through a business model focused on expanding network effects and data collection […]".

[7] No original: "[...] the current and traditional way to develop video games is modular (due to their extreme complexity), but at the source code level. After compilation, a monolithic software is created, in which only an experienced developer can replace the internal modules".

> desenvolvidos para criar o *play*. [...] Neste sentido, jogos digitais são motores de mundos[8]. (Sicart, 2023, p. 39, tradução e destaque nossos).

Sicart aponta para a aproximação entre software e jogo digital a partir do uso que ambos fazem de regras e processos para alcançar objetivos diferentes, mas complementares. Ao mesmo tempo, regras de jogo em jogos digitais sempre se alinham às regras do software, seja este alinhamento feito intencionalmente no design do jogo ou simplesmente por limitações práticas impostas pelas plataformas utilizadas pelo software.

Porém, para os propósitos desta pesquisa é necessário distinguir o jogar ("*play*") o jogo do brincar ("*play*") com o software. Enquanto a perspectiva dada ao *play* não se descaracteriza quando tratamos de facetas diferentes deste mesmo objeto (o jogo digital), devemos ressaltar a forma como brincar com o software tem o potencial de alterar o próprio software, suas plataformas e, por consequência, todas as regras que constituem o jogo.

"*Play é disruptivo*, por consequência de ser apropriativo. [...] Ele quebra o estado das coisas"[9] (Sicart, 2014, p. 14, tradução nossa). Esta característica se torna bastante visível quando olhamos para as modificações realizadas em jogos digitais – aqui incluindo *mods*, *hacks*, *cracks*, gambiarras e outros modelos de apropriação – como mais uma forma de jogar/brincar o jogo e com o jogo.

Ao mesmo tempo, a modificação direta não é a única forma na qual o *play* altera a infraestrutura dos jogos digitais. Podemos observar este efeito também na simples apropriação que é feita do uso destes jogos e suas plataformas. A função de um software e de suas plataformas não é definida apenas por suas características internas, mas também pelo seu uso. As formas do *play* são um caminho que define a funcionalidade dos softwares aqui observados. Quando observamos um jogo digital, temos um software que se transforma em um ambiente de jogo – ou *playground* (Sicart, 2014) – a partir da apropriação de um *play*, o jogar o jogo, utilizando as regras definidas a partir do design do software. Ao mesmo tempo, se este *play* ocorre de forma modificadora, no brincar com o software, sua posição não é mais a de *playground*, e sim a de *plaything* (Sicart, 2023), um produto gerado pelo jogar.

Mantella

Para ilustrar os efeitos do brincar com o software utilizaremos o *mod Mantella*. *Mantella* é uma modificação ainda em desenvolvimento para *The Elder Scrolls V: Skyrim VR*, criado pelo

[8] No original: "Software uses rules and processes to allow computers to perform operations on data representations, and video games use rules and processes to allow users to interact with computational models designed to create play. […] In this sense, video games are world engines".

[9] No original: "*Play is disruptive* as a consequence of being appropriate. […] it breaks the state of affairs".

usuário "Art from the machine" e divulgado no Youtube[10] e no Reddit[11]. O *mod* tem uma premissa simples com uma execução complexa: Criar e substituir diálogos com personagens não jogáveis utilizando inteligências artificiais generativas – em especial o ChatGPT. Para esta análise observaremos como o *mod* funciona a partir dos processos que ele gera por meio de comandos e requisições e como estes processos interagem com as plataformas que compõem a infraestrutura do jogo.

Desenvolvido para a versão de realidade virtural (*VR*) de *Skyrim* – mas também funcional nas versões *Special* e *Anniversary*, segundo o autor –, o *mod* aproveita da interface de áudio disponibilizada nos aparelhos de *VR* para capturar a voz do jogador em meio ao jogo. Esta não é uma funcionalidade exclusiva deste *mod* nem da versão do jogo. *Mods* como *ThuuMic* (DeadlyAzuril; PsychoHampster, 2012) e *SkyVoice SSE* (Terreden, 2017) também utilizam interfaces de áudio – que neste caso não necessariamente vêm da plataforma de *VR*, mas de qualquer interface reconhecida pelo sistema operacional – para capturar a voz do jogador e realizar comandos de jogo a partir desta informação. Porém, enquanto estes dois *mods* processam a informação de áudio em uma plataforma que já faz parte da infraestrutura utilizada pelo jogo, o sistema de reconhecimento que faz parte do sistema operacional do computador, *Mantella* insere novas plataformas que não são externas apenas ao software do jogo, mas também são externas à própria infraestrutura do computador utilizado.

Por ainda estar em fase de desenvolvimento, muito do que compõe o *mod* está em constante mudança e atualização. Para este artigo utilizamos informações disponibilizadas pelo autor em redes sociais abertas, como o Reddit, YouTube e Discord, e análise direta dos arquivos disponibilizados pelo autor da versão 0.3 do *mod*. Porém, ainda é necessário destacar que esta análise não será completa devido à falta de acesso tanto à versão mais atualizada do *mod*, que ainda está indisponível, quando à documentação interna das plataformas de inteligência artificial utilizadas, devido ao modelo de negócios "encaixapretado" adotado por elas.

Whisper, a primeira plataforma inserida pelo *mod*, é uma ferramenta de transcrição de áudio baseada em uma infraestrutura de inteligência artificial treinada em uma enorme quantidade de dados não especializados e pouco supervisionados, fazendo com que a ferramenta seja funcional em diversos contextos de uso, sem a necessidade de direcionamentos e adaptações para usos específicos (Radford *et al.*, 2022). Na versão observada do *mod*, *Whisper*

[10] Disponível em: https://www.youtube.com/watch?v=fJPY6sD527A. Acesso em: 26/07/2023.
[11] Disponível em: https://www.reddit.com/r/skyrimvr/comments/14cp2sp/mantella_chatgpt_in_skyrim_vr_improved_voices/. Acesso em: 26/07/2023.

é chamado a partir da API[12] do programa na nuvem, não sendo necessária a instalação da versão local do programa. Porém, utilizar a versão local é possível e pode afetar a velocidade de resposta do sistema (Art From The Machine, 2023a) – computadores com grande capacidade de processamento podem ser mais rápidos do que a versão online e vice-versa.

O *ChatGPT* é a segunda plataforma inserida pelo *mod* e é onde ocorre o processamento de resposta dos personagens não jogáveis. Após a transcrição realizada pelo *Whisper*, o software do *mod* é responsável por transferir o texto transcrito para o *ChatGPT* por meio da API da plataforma, que já recebeu *prompts*[13] prévios indicando o contexto e o personagem que deve interpretar em sua resposta. Na versão atual, o *mod* traz alguns *prompts* já escritos que são enviados automaticamente ao *ChatGPT*, mas todos eles podem ser editados pelo usuário. A inteligência artificial então utiliza as informações contextuais para interpretar uma resposta do personagem à pergunta feita pelo usuário. Destacamos que o *mod* atualmente funciona apenas com as versões 4, 4-32K, 3.5-turbo e 3.5-turbo-16K do *ChatGPT*, sendo todas elas pagas. Portanto, como a plataforma é acessada do lado do usuário, o usuário deve ter um código de acesso para uma destas versões para utilizar o *mod*.

Figura 2 - *Prompts* de personagem

```
[Prompt]
; prompt:
;       The starting prompt sent to the LLM when an NPC is selected
;       If you would like to edit this, please ensure that the below dynamic variables are contained in curly brackets {}:
;               name = the NPC's name
;               bio = the NPC's background description
;               trust = how well the NPC knows the player (eg "a stranger", "a friend")
;               location = the current location
;               time = the time of day as a number (eg 1, 22)
;               time_group = the time of day in words (eg "in the morning", "at night")
;               language = the selected language
;               conversation_summary = reads the latest conversation summaries for the NPC stored in data/conversations/NPC_Name/NPC_Name_summary_X.txt
```

Listas de *prompts* enviados ao *ChatGPT* indicando à inteligência artificial quem é o personagem a ser interpretado, seu histórico, sua relação com o personagem jogador e outras informações contextuais, como horário no jogo e língua falada.
Note que o autor do *mod* utiliza "LLM" (*Large Language Model*) e não "ChatGPT", indicando que o sistema é construído de forma modular, podendo ser adaptado para outras plataformas de inteligência aritifical.
Fonte: Art from the machine, 2023b.

A resposta gerada pelo *ChatGPT* é então transferida pelo *mod* ao *xVASynth*, uma ferramenta também baseada em inteligência artificial capaz de converter o texto escrito em arquivos de áudio que imitam as vozes originais dos personagens do jogo. Este áudio é reproduzido dentro do jogo como uma fala do personagem.

Por fim, a última plataforma utilizada pelo *mod* é o software desenvolvido pelo próprio autor do *mod*, responsável por fazer as conexões entre as APIs das outras plataformas e do jogo,

[12] Interface de programação de aplicação (*Application Programming Interface*). É uma interface aberta para a comunicação entre softwares, especialmente entre aplicações e bibliotecas e utilitários (Stallings, 2018).
[13] Comandos ou indicações contextuais enviados ao software pelo usuário ou por meio de API.

enviar os *prompts* pré-definidos, indicar locais de armazenamento de arquivos de histórico e temporários a serem utilizados pela plataforma e pelo software do jogo, etc.

Ou seja, o *mod* não apenas conecta outras plataformas à infraestrutura de *Skyrim*, mas ele próprio é uma plataforma necessária para instrumentalizar a conexão entre estas plataformas. É importante notar que este não é o modelo padrão do *modding* em *Skyrim*.

Os arquivos de *mods* em *Skyrim* são divididos em formatos de arquivos diferentes de acordo com o local que a modificação ocupa na hierarquia das plataformas internas – .esm para modificações "*master*", que são prioridade na ordem de carregamento por servirem de base para os arquivos de hierarquia menor, .esp para modificações de "*plugins*", que são carregadas individualmente sobre as plataformas .esm, e .esl para modificações "*light*", que são menores e mais dependentes, sendo carregadas todas juntas e em último lugar na lista de prioridade de arquivos. Ou seja, o formato modular escondido atrás do método "monolítico" de distribuição do jogo é revelado em alguns níveis a partir destas aberturas em diferentes níveis hierárquicos, algo que só é possível devido à modularidade na qual o jogo foi desenvolvido e na permissão garantida pelos desenvolvedores ao abrir este tipo de acesso.

Enquanto *Mantella* ainda tem um arquivo padrão de *mod*, um .esp, este arquivo é usado apenas para inserir uma nova magia no jogo, responsável unicamente por ativar um *script* que inicia o modo de diálogo do *mod*. Ou seja, este arquivo existe apenas para que o sistema possa ser ativado de dentro do jogo, enquanto o sistema em si funciona de forma externalizada por meio de plataformas e pontes criadas entre elas.

Ao mesmo tempo, para que o *mod* funcione, o usuário deve abrir primeiro a plataforma *xVASynth* em seu computador, seguido do arquivo executável do software próprio do *mod* e só depois abrir o jogo. Portanto, por mais que o *mod* seja ativado dentro do jogo, ainda existe um processo que deve ser realizado manualmente pelo usuário fora do jogo.

Também devemos destacar que o processo de diálogo do *mod* não se encaixa de forma imperceptível no contexto do jogo como um todo. O *mod* funciona apenas com um personagem não jogável por vez, já que os *prompts* são enviados ao *ChatGPT* previamente, existe um atraso não natural entre a pergunta do usuário e a resposta do personagem devido ao tempo necessário para que as plataformas cumpram suas funções, e, por mais que o *xVASynth* imite as vozes dos dubladores originais dos personagens, a versão produzida pela IA não soa tão natural quanto a atuação dos dubladores, fazendo com que a entonação dos personagens soe "robótica".

Inteligências artificiais

Entender *Mantella* e *mods* similares não se resume a compreender as mudança estruturais realizadas no software, mas também entender o que efetivamente muda na constituição completa do jogo digital.

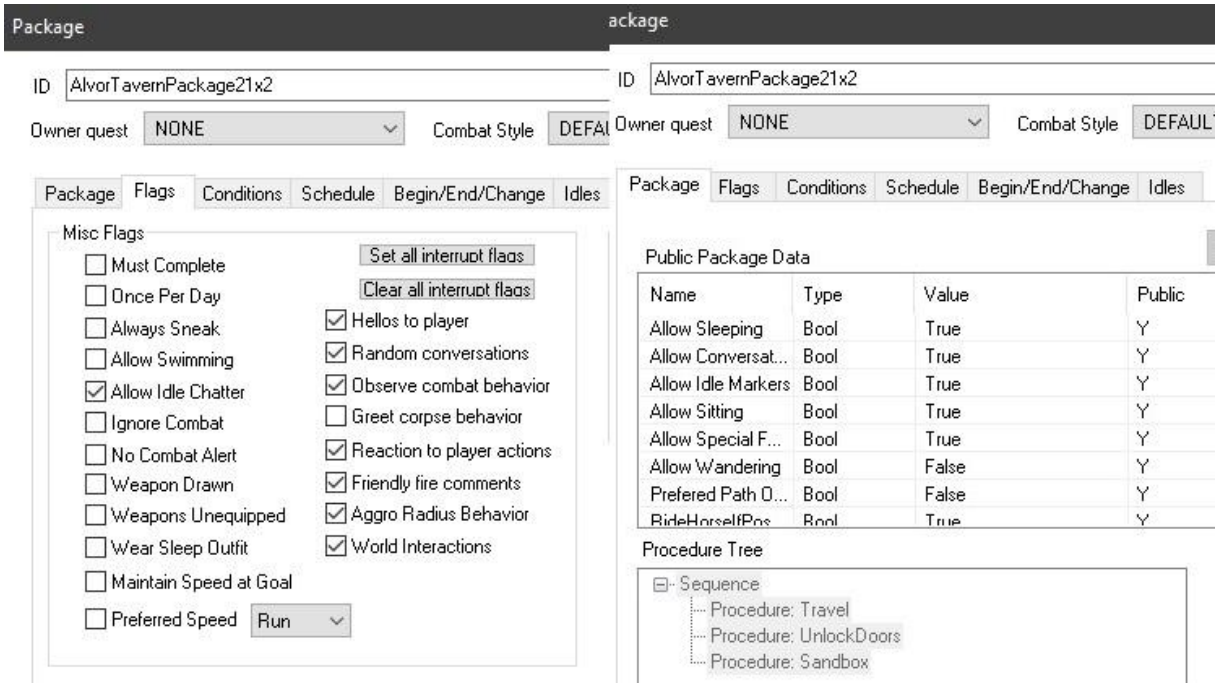
O tópico de inteligências artificiais (ou IAs/*AI*s) não é novo para os jogos digitais, porém, as características as quais este termo se refere em um senso comum criado em torno destes jogos variam bastante de acordo com o contexto ou com o jogo sobre o qual estamos falando, e se diferem ainda mais dos significados encontrados fora do contexto dos jogos. Neste senso comum dos jogos digitais, é comum atribuir à "*AI*" qualquer aparente decisão tomada pelo computador durante o jogo, normalmente fundamentadas em informações prévias, como o contexto e o ator em questão, e incluindo mas não limitado à ações muito simples, como o movimento e o *pathing* de atores (personagens, criaturas, etc) controlados pela máquina e suas ações que interagem com o mundo e com o jogador, por exemplo. Estas decisões, por mais que possam ser tomadas por IAs, muitas vezes apenas aparentam serem decisões reais.

Tomemos como exemplo *Skyrim*, que é um jogo de mundo aberto, com centenas de diferentes atores controlados pela máquina com comportamentos variados e que muitas vezes respondem ao contexto no qual estes atores estão inseridos. A aparente capacidade de "compreensão" de contexto destes atores e suas respostas, também aparentemente lógicas, à estes contextos criam a impressão de que existem processos de tomada de decisão em tempo real pela máquina. Porém, quando observamos o funcionamento do jogo a partir de seu código, não a partir de seus resultados, podemos notar que estas ações são em sua maioria pré-definidas, mas que passam a impressão de tempo real a partir de um sistema complexo de condicionamentos, estruturas de *behavior trees* e aleatoriedade que organizam o ordenamento destas ações.

Neste caso, quando falamos da IA implementada em *Skyrim*, nos referimos à um sistema que envolve muito mais uma lista de gatilhos e ações relacionadas definidas pelos desenvolvedores humanos do que decisões tomadas por uma inteligência artificial – porém, isto não exclui as IAs deste processo, pois elas ainda são responsáveis por transformar estas decisões em ações através dos processos de troca entre as estruturas que fazem o jogo rodar.

Yoones Sekhavat (2017) descreve como as chamadas *behavior trees* (ou *BT*s) funcionam como mecanismos de controle para IAs abrangentes que são utilizadas para definir o comportamento de atores não-jogadores nos jogos digitais. Com o uso de *BT*s, desenvolvedores são capazes não apenas de predefinir ações para seus atores, mas também hierarquizá-las dentro de agrupamentos de condições.

Figura 3 - *AI Package* em *Skyrim*



Listas de condicionamentos predefinidos de um *AI Package* composto para um personagem específico em *Skyrim*.
Fonte: Captura de tela própria dos arquivos de *Skyrim Special Edition* (Bethesda Softworks, 2016) pelo *Skyrim Special Edition: Creation Kit* (Bethesda Softworks, 2022).

Na figura 3 podemos notar como *Skyrim* organiza suas permissões e condições dentro de pacotes ("*packages*") de ações, que posteriormente são hierarquizadas e condicionadas à determinados contextos. Enquanto o próprio pacote traz definições detalhadas de ação, ele não é o único pacote presente no código deste personagem, tornando este processo de hierarquização complexo, envolvendo organização não só de condições, mas de pacotes inteiros de condições, que cria a impressão de dinamicidade e intencionalidade nas "escolhas" realizadas pela máquina – que no fim das contas apenas segue instruções claramente definidas por humanos.

Em relação às ações de conversa, que é a área modificada por *Mantella*, estes pacotes definem o que e quando estes personagens vão falar quando relacionados à ações contextuais e ações do jogador não relacionadas diretamente ao ato de conversar com o personagem – por exemplo o tipo de equipamento utilizado pelo personagem jogador, o horário em jogo em que a conversa se inicia, a ocasião onde a conversa ocorre, entre outros.

A partir do momento em que o jogador inicia um diálogo direto com o personagem, as ocorrências conversacionais não são mais definidas por inteligências artificiais ou *BTs*. O diálogo ocorre em uma estrutura fechada, onde o jogador tem apenas algumas opções de falas e os personagens têm respostas fixas para cada opção aberta ao jogador. Ou seja, é um processo

linear, apesar de distribuído em formato de árvore, pré-determinado pelos desenvolvedores, sem a necessidade de adaptações específicas por parte do software. Com *Mantella*, o diálogo sai da estrutura de *BTs* ou linear para adotar o caráter generativo do *ChatGPT*.

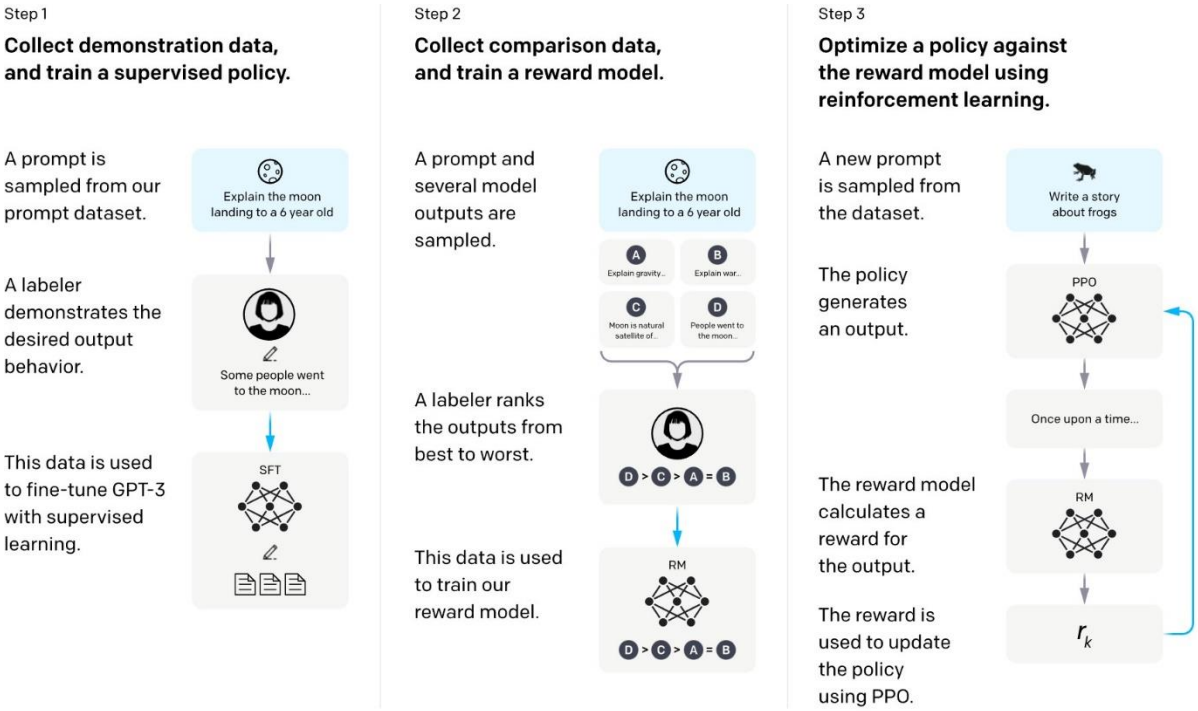
> IA generativa refere à inteligências artificiais que podem gerar conteúdo original, ao invés de simplesmente analisar ou agir sobre dados existentes, como *expert systems*.[14] (Gonzalo-Brizuela; Garrido-Merchán, 2023, p. 1, tradução e destaque nossos).

Gonzalo-Brizuela e Garrido-Merchán (2023) comparam o modelo de IAs com sistemas *expert* devido ao modo no qual os sistemas processam os dados aos quais têm acesso. Enquanto os sistemas *expert* utilizam modelos baseados em *if-else*, ou seja, programações baseadas em regras condicionais, assim como os *BTs*, inteligências artificiais modernas utilizam modelos *discriminator* ou *transformer*. O *ChatGPT* funciona em um modelo generativo *transformer* pré-treinado (*Generative Pre-trained Transformer*, GPT), o que significa que ao invés de trabalhar num sistema de regras pré-definidas como os *BTs*, a inteligência artificial do *ChatGPT* cria respostas a partir da análise e reconhecimento de padrões de repetição linguísticos, mesmo no caso de suas versões multimodais (CAO *et al*., 2018), que são utilizadas no *Mantella*, apesar de o *mod* utilizar apenas funções unimodais.

Para a geração de respostas, o *ChatGPT* foi treinado em um sistema de três passos, onde os dois primeiros envolvem ação humana direta – indicando formatos de resposta esperados no primeiro e ranqueando respostas obtidas no segundo – e um terceiro retroalimentado, onde as próprias respostas geradas pela IA servem de parâmetro para a decisão para respostas subsequentes (Ouyang *et al.*, 2022). A forma na qual estas respostas são entregues ao usuário, no caso unimodal de texto, é gerada em um modelo de linguagem "autorregressivo", onde o próximo resultado (ou a próxima palavra) é baseado nos resultados anteriores dentro da mesma resposta, tornando a IA capaz de criar frases coesas a partir apenas da estrutura (CAO *et al.*, 2018).

[14] No original: "Generative AI refers to artificial intelligence that can generate novel content, rather than simply analyzing or acting on existing data like expert systems".

Figura 4 - Passos de treinamento do *ChatGPT*



Fonte: Ouyang *et al*., 2022.

Estas diferenças no modelo de "inteligências" envolvidas nos diálogos do jogo faz com que, por mais que tanto os modelos lineares em árvore e *BTs* quanto o modelo do *ChatGPT* funcionem com sistemas lógicos sob estruturas definidas por humanos, este diálogo saia de uma posição pré-definida e previsível para adotar o modelo generativo, que não é previsível devido à sua fundamentação em um volume de dados muito maior do que um humano conseguiria analisar.

Conclusões

Quando observamos o ato de modificar um jogo digital a partir da perspectiva da brincadeira ("*play*") podemos tratar as estruturas modificadas por usuários como objetos do jogar, *playthings*. Esta perspectiva nos ajuda a entender a forma como o modificar é capaz de mover as estruturas que sustentam os softwares do jogos digitais, assim como o jogar transforma o software em jogo.

Enquanto os *mods* são normalmente tratados apenas como adições ou alterações menores em um software de jogo, que *a priori* se mantém "monolítico", com estas modificações sendo realizadas ao seu redor, quando falamos de *mods* que alteram diretamente não apenas características internas deste software, mas também as plataformas das quais ele depende, podemos notar uma transformação maior no que compõe o jogo. As possibilidades do jogar em

Skyrim são muito diferentes das possibilidades do jogar no *Skyrim* modificado por *Mantella*. O jogar muda porque o *playground* muda.

Se entendemos o próprio ato de dialogar com personagens fictícios como uma ação de jogo, devemos entender também que *Mantella* se "eleva" ao espaço de *playground*, enquanto *Skyrim* se torna mais uma plataforma que sustenta este *playground*. Esta discussão nos leva de volta à percepção dos objetos a partir de seus usos, não de suas características materiais. Enquanto *Skyrim* é uma aplicação desenvolvida com a função de ser um *playground*, ele também pode se tornar uma plataforma ou um *plaything* de acordo com o uso que é feito de seu software. Da mesma forma, este software transita entre vários espaços da infraestrutura de uma aplicação. Isto acontece não apenas em *mods* que externalizam os processos infraestruturais do jogo, como *Mantella*, mas também em modificações que utilizam o software do jogo como uma estrutura de base para a criação de outros jogos, como é o caso de *Enderal* (SureAI, 2019), um jogo desenvolvido a partir de modificações em *Skyrim*, transformando o que era uma aplicação monolítica em uma estrutura de utilidades e bibliotecas.

Ao mesmo tempo, *mods* como *Mantella* abrem espaços para discussões mais amplas que surgem a partir desta desestruturalização do jogo. Podemos pensar por exemplo em iniciativas como a plataforma *Xbox Cloud Gaming*, onde a infraestrutura de centenas de jogos – incluíndo o próprio *Skyrim* – é removida da plataforma utilizada pelo jogador, facilitando processos de compatibilidade entre hardware e software, mas impedindo quase todas as possibilidades abertas pelo brincar com o software devido à alienação da materilidade do software do jogador e ao modelo comercial "encaixapretado" da plataforma. Essencialmente, estes objetos são capazes de perder possibilidades de *play* por meio da externalização não do *playground*, mas da infraestrutura que o suporta.

Mantella também nos leva à uma série de discussões que envolvem a utilização de inteligências artificiais tanto durante o jogo quanto no desenvolvimento e estruturação de seu software. Podemos levantar como exemplo as questões éticas envolvidas na substiuição de trabalhadores humanos e na apropriação de seus trabalhos quando as funções de roteirista e dublador são substituídas por uma inteligência artificial que se baseia não apenas no uso não autorizado de suas produções, mas que foram fundamentadas no uso não autorizado de milhares de outras produções as quais não temos acesso atualmente.

Por enquanto estas questões ainda surgem como uma consequência do jogar, entendendo o jogo modificado como um *plaything*, porém, já nos indicam para problemas que podem surgir em grande escala caso a prática seja adotada no desenvolvimento de jogos digitais em um futuro não tão distante. Atualmente o trabalho de desenvolvimento de jogos digitais e

de suas estruturas internas já é precarizado (Woodcock, 2020), o que torna ainda mais necessário manter sob vigia o surgimento e popularização de novas formas de se estruturar estes produtos que sejam problemáticas tanto na forma como se estruturam como na forma como são utilizadas e aplicadas nestes contexto.

Referências

ART FROM THE MACHINE. **Mantella**, [*S. l*], 18 jun., 2023a. Disponível em: https://discord.gg/Q4BJAdtGUE. Acesso em: 28/07/2023.

ART FROM THE MACHINE. **Mantella**. Versão 0.3, 16 jul., 2023b. Disponível em: https://discord.com/channels/1114128157554004011/1114128960113098854/1130166036126253226. Acesso em: 28/07/2023.

BETHESDA SOFTWORKS. **The Elder Scrolls V: Skyrim:** Special Edition. Versão 1.6.640.0, 27 out., 2016. Disponível em: https://store.steampowered.com/app/489830/The_Elder_Scrolls_V_Skyrim_Special_Edition/. Acesso em: 17/07/2023.

BETHESDA SOFTWORKS. **Skyrim Special Edition:** Creation Kit. Versão 1.6.438.0, 25 abr., 2022. Disponível em: https://store.steampowered.com/app/1946180/Skyrim_Special_Edition_Creation_Kit/. Acesso em: 17/07/2023.

DEADLYAZURIL; PSYCHOHAMPSTER. **Thuumic – Use your mic for dragon shouts**. Versão 1.94.2, 20 ago., 2012. Disponível em: https://www.nexusmods.com/skyrim/mods/5626/. Acesso em: 26/07/2023.

CAO, Yihan; LI, Siyu; YAN, Zhiling; DAI, Yutong; YU, Philip S.; SUN, Lichao. A Comprehensive Survey of AI-Generated Content (AIGC): A History of Generative AI from GAN to ChatGPT. **Association for Computing Machinery Journal**, [*S. l.*], v. 37, n. 4, p.1-44, ago., 2018. Disponível em: https://arxiv.org/abs/2303.04226. DOI: https://doi.org/10.48550/arXiv.2303.04226. Acesso em: 23/06/2023.

CHIA, Aleena; KEOGH, Brendan; LEORKE, Dale; NICOLL, Benjamin. Platformisation in game development. **Internet Policy Review**, [*S. l.*], v. 9, n. 4, p. 1-28, out., 2020. DOI: 10.14763/2020.4.1515. Disponível em: https://policyreview.info/articles/analysis/platformisation-game-development. Acesso em: 27/07/2023.

FERNÁNDEZ-VARA, Clara. **Introduction to game analysis**. Nova Iorque: Routledge, 2015.

FILHO, Edu Fernandes lima Jacques. **(Re)faça você mesmo**: Práticas de Modding e a circulação midiática na série de jogos ARMA. Tese (Doutorado em Ciências da Comunicação) – Unidade Acadêmica de Pesquisa e Pós-Graduação, Universidade do Vale do Rio dos Sinos. São Leopoldo, p. 291. 2018. Disponível em: http://www.repositorio.jesuita.org.br/handle/UNISINOS/7311. Acesso em: 21/05/2022.

FREITAS, Filipe Alves de. **Video game como comunicação**: Perspectivas sobre a produção de sentido a partir de jogos digitais casuais. Belo Horizonte: PPGCOM UFMG, 2017. Disponível em: https://seloppgcomufmg.com.br/publicacao/video-game-como-comunicacao/. Acesso em: 07/10/2022.

GONZALO-BRIZUELA, Roberto; GARRIDO-MERCHÁN, Eduardo. **ChatGPT is not all you need**. A state of the art review of large generative AI models. arXiv preprint arXiv:2301.04655, p. 1-22, 2023. DOI: https://doi.org/10.48550/arXiv.2301.04655.

HASSAM, Mickael; KARA, Nadjia; BELQASMI, Fatma. Virtualized infrastructure for video game applications in cloud environments. *In*: MobiWac'14: ACM international symposium on mobility management and wireless access, 12., 2014, Montreal. **Proceedings** […] Montreal: ACM Digital Library, 2014. p. 109-114. DOI: https://doi.org/10.1145/2642668.2642679.

KONZACK, Lars. Computer Game Criticism: A Method for Computer Game Analysis. *In*: MÄYRÄ, Frans (ed.). **Proceedings of Computer Games and Digital Cultured Conference**. Tampere: Tampere University Press, 2002. p. 89-100.

OUYANG, Long; WU, Jeff; JIANG, Xu; ALMEIDA, Diogo; WAINWRIGHT, Carroll; MISHKIN, Pamela; ZHANG, Chong; AGARWAL, Sandhini; SLAMA, Katarina; RAY, Alex; SCHULMAN, John; HILTON, Jacob; KELTON, Fraser; MILLER, Luke; SIMENS, Maddie; ASKELL, Amanda; WELINDER, Peter; CHRISTIANO, Paul; LEIKE, Jan; LOWE, Ryan. Training language models to follow instructions with human feedback. *In*: Advances in Neural Information Processing Systems, 35., 2022, New Orleans. **Proceedings** […] New Orleans: Neural Information Processing Systems Foundation (NeurIPS), 2022. p. 1-15. Disponível em: https://proceedings.neurips.cc/paper_files/paper/2022. Acesso em: 28/07/2023.

RADFORD, Alec; KIM, Jong Wook; XU, Tao; BROCKMAN, Greg; MCLEAVEY, Christine; SUTSKEVER, Ilya. Robust speech recognition via large-scale weak supervision. **OpenAI**, 2022. Disponível em: https://openai.com/research/whisper. Acesso em: 27/07/2023.

SEKHAVAT, Yoones. Behavior trees for computer games. **International Journal on Artificial Intelligence Tools**, [*S. l.*] v. 26, n. 2, p.1730001-1, 1730001-28. abr. 2017. DOI: https://doi.org/10.1142/S0218213017300010.

SICART, Miguel. **Play Matters**. Cambridge: The MIT Press, 2014.

SICART, Miguel. **Playing Software**: Homo ludens in computational culture. Cambridge: The MIT Press, 2023.

STALLINGS, William. **Operating Systems**: Internals and Design Principles. 9ª ed. Harlow: Pearson Education, 2018.

SURE AI. **Enderal**: Forgotten Stories (Special Edition). Versão 11586694, 29 jun., 2023. Disponível em: https://store.steampowered.com/app/976620/Enderal_Forgotten_Stories_Special_Edition/. Acesso em: 27/07/2023.

TERREDEN. **SkyVoice SSE – Become the Dragonborn**. Versão 1.5, 10 out., 2017. Disponível em: https://www.nexusmods.com/skyrimspecialedition/mods/12835. Acesso em: 26/07/2023.