**CENTRO UNIVERSITÁRIO BELAS ARTES DE SÃO PAULO**

**ANÁLISE E DESENVOLVIMENTO DE SISTEMAS - EAD**

**MÓDULO 02**

**FELIPE SCHAITEL - 24112424**

**LAÍSE NEVES - 24112425**

**DESIGN DE SOFTWARE - ATIVIDADE 02 (UNIDADE 04)**

**SÃO PAULO**

**2024/2**

**SUMÁRIO**

**1 INTRODUÇÃO…………………………………………………………………………………….02**

**2 ANÁLISE…………………………………………………………………………………………..02**

**2.1 DEPURAÇÃO PARA JOGOS………………………………………………………….……..03**

2.1.1 Como é diagnosticado, com depurador, um problema de inconsistência visual e interação entre objetos?.........................................................................................................03

**2.2 DEPURAÇÃO EM ALGORITMOS DE INTELIGÊNCIA ARTIFICIAL……………………04**

2.2.1 Quais formas de se depurar as decisões erradas tomadas pelos algoritmos?............04

**2.3 DEPURAÇÃO EM COMPUTADORES VEICULARES…………………………………….05**

2.3.1 Como depurar um problema em um veículo cuja falha não é reproduzível em laboratório ou no ambiente de desenvolvimento, somente no ambiente de uso (andando nas ruas)?.....................................................................................................................................05

**2.4 ROTEIRO PERGUNTAS E RESPOSTAS…………………………………………………...06**

**3 RESULTADOS………………………………………………………………………………….…07**

**4 CONCLUSÃO……………………………………………………………………………………..09**

**5 REFERÊNCIAS BIBLIOGRÁFICAS……………………………………………………………10**

**1 INTRODUÇÃO**

Nesse relatório analisamos sobre o que é *debugger*, sua importância, o processo dos testes à implantação e plano de retorno. Também discorremos sobre como é feita a depuração de software em diferentes aplicações do mercado como de: Jogos, Algoritmos de Inteligência Artificial e Computadores veiculares. Desenvolvemos a partir de referências como o material didático da disciplina, artigo da EPIC Games e entre outras fontes disponíveis nas referências bibliográficas.

Também contém o registro da entrevista que realizamos sobre o assunto com Leonardo Veiga, Engenheiro de Desenvolvimento de Software em testes (SDET) na *Coherence Multiplayer Unlocked*.

Concluímos o relatório relacionando o conteúdo aprendido na disciplina com nosso o Projeto integrador multidisciplinar extensionista II (PIME).

**2 ANÁLISE**

Depurador ou *debuggers* são ferramentas importantes para o desenvolvimento de Software. O objetivo do debugger é testar e identificar *bugs* no código para correção, além de serem úteis para compreender o comportamento do programa e testar diversos cenários. Os *debuggers* também possuem grande capacidade de integração com outras ferramentas. Proporcionando qualidade, eficiência e confiabilidade da aplicação.

*Debugger* está presente nos Ambientes de Desenvolvimento Integrados (IDEs) como *Pycharm, Visual Studio, BlueJ,* Eclipse etc.

As funcionalidades mais comuns dos *debuggers* são alguns comandos comuns para controle e execução que permitem ao desenvolvedor acompanhar linha por linha da aplicação. *Breakpoint* (ponto de parada) é um conceito associado que envolve definir pontos de parada em locais específicos do código para encerrar a sua execução. Os *breakpoints* podem ser úteis, principalmente, para analisar códigos extensos. Na disciplina conhecemos o depurador da IDE *Pycharm* e os comandos *step* (executar linha por linha), *step into* (permite entrar em funções), *step over* (executa as instruções internas).

Testar e identificar *bugs* é crucial para o sucesso, funcionalidade e qualidade do *Software*, pois ao identificar o problema no início fica mais fácil corrigir além de ter menos gastos e mobilização de equipe. É importante testar, entender o comportamento do software e corrigir antes que afete o usuário final. Após a conclusão dos testes, em seguida é o momento de organizar a implantação gradual e controlada na produção. É importante selecionar um grupo de usuários. De acordo com o material didático:

Durante o processo de implantação, é essencial estabelecer formas de controle para avaliar o desempenho e a aceitação da nova versão do software por parte dos usuários. Isso pode ser feito por meio de diversas métricas, como a avaliação do comportamento dos servidores e infraestrutura onde estão as aplicações, a quantidade de reclamações registradas nas áreas de atendimento ao cliente, enquetes ou entrevistas rápidas (...) Essas métricas fornecem insights valiosos sobre a estabilidade e a usabilidade da nova versão do software.

(CALVACANTE, p.140)

Também é importante estabelecer um plano de retorno caso tenha algum problema imprevisível e/ou desagradar os usuários. Geralmente essa fase é gerenciada pelo setor de Gestão de Mudanças (GMUD).

Portanto, *Debuggers* são ferramentas essenciais para desenvolvimento de programas, entre eles, *Softwares* de *Games*, Algoritmos de Inteligência Artificial, Computadores veiculares etc.

**2.1 DEPURAÇÃO PARA JOGOS**

**2.1.1 Como é diagnosticado, com depurador, um problema de inconsistência visual e interação entre objetos?**

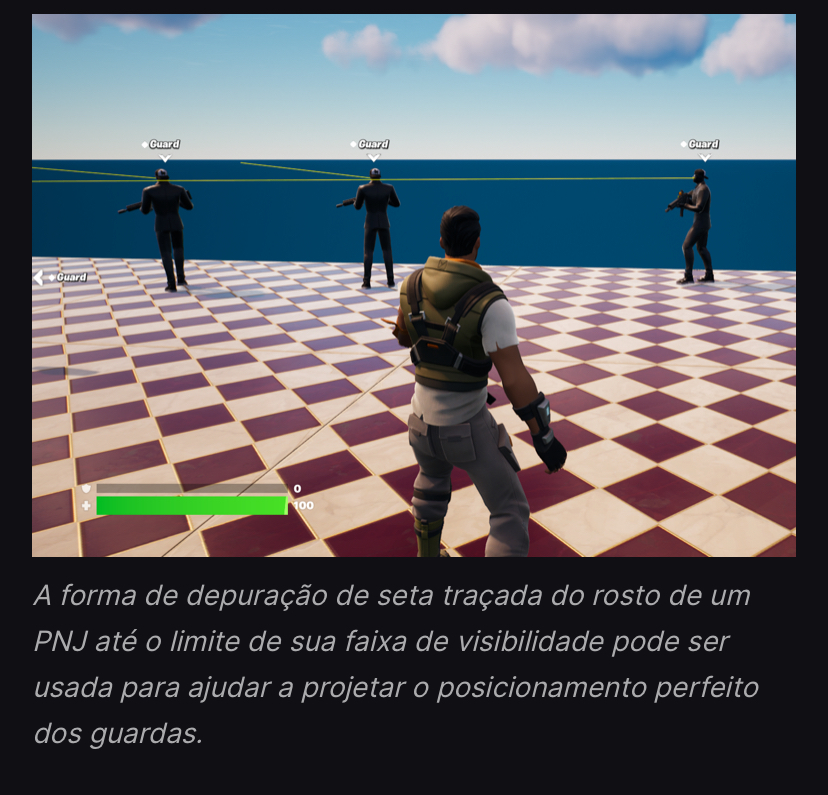
De acordo com a *EPIC Games*, empresa multinacional de jogos como Fortnite, uma forma de *debuggar* jogos é através do *Debug Draw*

Use Verse para traçar formas e ajudar a depurar seu jogo[…] Para ajudar na [depuração](https://dev.epicgames.com/documentation/pt-br/uefn/verse-glossary#debugging) do seu jogo, você pode exibir determinados dados do jogo para fins de teste enquanto oculta essas informações do jogador. Uma forma de fazer isso é usar registros em log, mas você pode exibir alguns dados do jogo de maneira mais visual com a **API Debug Draw**. Usando a API Debug Draw, você pode traçar formas básicas e definir seu local, tamanho, cor e o tempo que aparecem na tela. Por exemplo, você pode querer visualizar a faixa de visibilidade de um PNJ ou a distância a partir da qual o áudio pode ser ouvido do seu ponto de origem. Com a forma de depuração da caixa, você pode traçar uma caixa cujas dimensões e local correspondam a qualquer volume, mesmo que esse volume esteja invisível no seu jogo. O que pode ajudar a descobrir o posicionamento de volumes com certos efeitos.

(EPIC, 2024)

A imagem 01 mostra como fica na tela as formas e linhas que o *verse* vai traçar através do *Debug Draw.*

Imagem 01



fonte: *EPIC Games Fortinite*

**2.2 DEPURAÇÃO EM ALGORITMOS DE INTELIGÊNCIA ARTIFICIAL**

**2.2.1 Quais formas de se depurar as decisões erradas tomadas pelos algoritmos?**

Durante a pesquisa constatamos que as Inteligências artificiais estão sendo utilizadas como ferramentas para *debuggar*. Esses algoritmos utilizam aprendizagem de máquina para prever, identificar e corrigir *bugs.* Mas quais são as formas de depurar as decisões erradas? Procuramos bastante e a maior parte dos artigos e sites que encontramos trouxe esse problema para uma perspectiva mais focada em explicabilidade, auditabilidade e ética.

No artigo “Como você pode depurar um sistema de IA de forma mais eficaz?” escrito pelo *Machine Learning Enginer* Maximilian Jalea, uma forma de *debuggar* IA é através de alguns passos como:

### 1 [Usar registro em log e monitoramento](https://www.linkedin.com/advice/1/how-can-you-debug-ai-system-more-effectively-5tbef?lang=pt&originalSubdomain=pt#usar-registro-em-log-e-monitoramento);

### 2 [Usar teste e validação](https://www.linkedin.com/advice/1/how-can-you-debug-ai-system-more-effectively-5tbef?lang=pt&originalSubdomain=pt#usar-teste-e-valida%C3%A7%C3%A3o);

### 3 [Usar depuração e rastreamento](https://www.linkedin.com/advice/1/how-can-you-debug-ai-system-more-effectively-5tbef?lang=pt&originalSubdomain=pt#usar-depura%C3%A7%C3%A3o-e-rastreamento);

### 4 [Use explicabilidade e interpretabilidade](https://www.linkedin.com/advice/1/how-can-you-debug-ai-system-more-effectively-5tbef?lang=pt&originalSubdomain=pt#use-explicabilidade-e-interpretabilidade);

### 5 [Use a colaboração e a comunicação](https://www.linkedin.com/advice/1/how-can-you-debug-ai-system-more-effectively-5tbef?lang=pt&originalSubdomain=pt#use-a-colabora%C3%A7%C3%A3o-e-a-comunica%C3%A7%C3%A3o);

**2.3 DEPURAÇÃO EM COMPUTADORES VEICULARES**

**2.3.1 Como depurar um problema em um veículo cuja falha não é reproduzível em laboratório ou no ambiente de desenvolvimento, somente no ambiente de uso (andando nas ruas)?**

Vimos na disciplina sobre a importância de *debuggar*, principalmente, nos softwares críticos em que um erro pode ocasionar acidentes e/ou perdas de vida.

Muitas pessoas utilizam essa tecnologia dos computadores veiculares, por isso, é importante ter algum tipo de mecanismo para registrar e documentar os dados em tempo real, como por exemplo, sensor de freio etc. De acordo com a empresa *Goodyear*:

Imagem 02:

### **4. Diagnóstico de falhas**

O computador do carro também é responsável por realizar o autodiagnóstico contínuo do veículo para detectar e registrar possíveis problemas ou falhas em sistemas e componentes.

Quando uma anormalidade é detectada, o computador pode ativar uma luz de advertência no painel para alertar o motorista e armazenar códigos de erro que os técnicos podem usar para diagnosticar e reparar o problema.

Fonte: Goodyear

Para que qualquer problema que apareça seja identificado e corrigido rapidamente. Pode-se analisar dados como o clima, a temperatura, relato dos motoristas etc. Para ajudar a identificar erros e na tomada de decisão sobre a correção. Por isso, uma forma de depurar é através de atualização regular de software para corrigir problemas ou adicionar novos recursos. Também é importante simular o ambiente controlado em que o computador veicular irá transitar.

**2.4 ROTEIRO DAS PERGUNTAS E RESPOSTAS TRANSCRITAS**

**0. Podemos fazer algumas perguntas sobre depurador?**

Resposta: Sim, *Debugger* é basicamente uma ferramenta que proporciona debugar (se vocês não tem essa palavra no seu vocabulário agora vocês tem). Ninguém fala depurar, todo mundo fala *debugar*, é importante aprender isso. Se não, as pessoas não vão saber do que vocês estão falando. Mas, é basicamente assim, sendo bem técnico, *debugger* é uma ferramenta que captura o processo da execução do código que você está fazendo e você coloca *breakpoints* que são pontos de paradas no seu código, quando a ferramenta chega nesse ponto para a execução. E nesse ponto você consegue inspecionar os valores que estão nas variáveis naquele momento naquele ponto do código. E você consegue avançar de instrução para instrução. Essas ferramentas estão todas embutidas nas *IDEs* de desenvolvimento. Não sei qual vocês estão aprendendo mas, se for *Python* provavelmente o seu professor falou do *Pycharm*. *Pycharm* tem um depurador dentro da *IDE*. Toda ferramenta de desenvolvimento, todo editor de texto de código vai ter um algo assim para vocês usarem.

Eu particularmente acho muito massante. O que eu mais gostava era o do Eclipse para fazer em JAVA. Mas enfim, hoje o meu depurador é simplesmente escrever “console.log estou aqui / console.log valor da variável” e mandar executar porque isso resolve o meu problema. Eu coloco mensagens de *print* na tela dos valores que quero olhar enquanto estou executando. Esse é o jeito que eu *debuggo* código mas, com depurador é muito mais eficiente porque você volta, passa para frente, para trás, vê as variáveis etc do *breakpoint*.

**1. Em qual área da Tecnologia da Informação você atua?**

Resposta: Desenvolvimento de software, QA

**2. Você utiliza depuradores no seu trabalho diário?**

Resposta: Sim

**3. Se sim, qual depurador você utiliza? Poderia descrever como o utiliza e qual o tipo (de software, ferramenta, etc.)?**

Resposta: Geralmente o famoso print(“passei aqui”). Às vezes, ao invés da mensagem, coloco o valor da variável que quero olhar. Pra códigos mais complexos com muitas variáveis, o depurador que está embutido na IDE do momento é utilizado.

**4. Na sua opinião, quais são as principais vantagens de usar um depurador? Você acredita que seria possível trabalhar sem ele?**

Resposta: Agilidade e testabilidade. Você consegue acompanhar em uma velocidade humana a execução do código e os valores quadrados nas variáveis/argumentos em tempo de execução. Sim, é possível, porém dependendo do cenário, mais trabalhoso.

**5. Você teria alguma dica ou curiosidade sobre o uso de depuradores que gostaria de compartilhar conosco?**

Resposta: Depurador é ferramenta. Usem quando necessário mas não dependam dele. Saber se virar sem ele é fundamental.

**3 RESULTADOS**

Foi interessante que durante a pesquisa achamos diversos sites, entre eles do UFMG sobre o livro premiado, que traz em seu conteúdo sobre algoritmo ser uma instituição social. Nunca tínhamos pensado nisso, a pesquisa da atividade nos encaminhou para diversas leituras formativas e aprendizados novos.

Durante a pesquisa sobre depuração nos Jogos, encontramos exemplos de códigos no site da *EPIC Games* além da explicação do *Debug Draw*.

Exemplo do código de como traçar a primeira forma de depuração:

using { /UnrealEngine.com/Temporary/Diagnostics }

my\_debug\_draw := class(debug\_draw\_channel) {}

DebugDraw:debug\_draw = debug\_draw{Channel := my\_debug\_draw}

//forma esfera

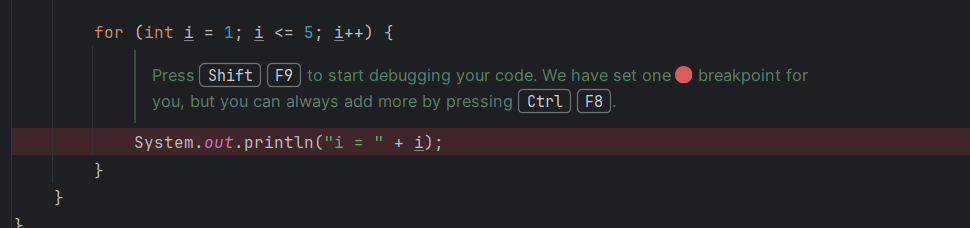
DebugDraw.DrawSphere(Center := vector3{Z:=150.0},?DrawDurationPolicy := debug\_draw\_duration\_policy.Persistent)

// forma caixa:

DebugDraw.DrawBox(vector3{Z:=150.0}, rotation{}, ?DrawDurationPolicy := debug\_draw\_duration\_policy.Persistent)

Na disciplina de Programação Orientada a Objetos estamos programando em *Java* e ao utilizar a *IDE intelliJ* foi possível notar o *debugger* do ambiente e principalmente, entender agora sua utilidade, funcionalidade e auxílio no desenvolvimento de software.

Imagem 03:



Fonte: IDE IntelliJ

Foi bastante difícil achar informações específicas sobre *debugger* em computadores veiculares e *debuggar* decisões erradas de IA., tivemos que ampliar a nossa busca para inglês.

Entretanto, com o que achamos sobre computador veicular foi possível fazer relações entre o conteúdo com a nossa primeira área de formação acadêmica: Licenciatura Artes - Teatro. No teatro é extremamente importante ensaiar, simular e repetir diversas vezes as cenas para chegar onde se espera na apresentação. Ensaio significa: “*substantivo masculino (...) avaliação crítica sobre as propriedades, a qualidade ou a maneira de usar algo; teste, experimento”* por Dicionário *Oxford Languages.*

Diversas outras áreas utilizam o artifício teatral de ensaiar, por exemplo, advogados ensaiam situações para usar no tribunal, estudantes de medicina durante a faculdade interagem com atores simulando doenças e entre outros cenários e cursos. Portanto, ensaiar tem caráter de *debuggar* e pedagógico, pois a partir da experiência simulada pode-se refletir e aprender fazendo com segurança. Assim como na depuração de computadores veiculares é importante estar experimentando antes em ambientes controlados para que no momento que for para valer ter uma performance excelente.

**4 CONCLUSÃO**

Durante o ano letivo aprendemos nas disciplinas que o sucesso, qualidade e eficiência de um software depende de alguns fatores entre eles:

1- Análise e documentação dos requisitos bem estruturado influência na modelagem e funcionalidades;

2- Confeccionar a identidade visual e Interface a partir dos princípios da Usabilidade. Pois a interface é o primeiro meio de contato que o usuário tem com a aplicação;

3 - Codificar em módulos, nomear as variáveis com nomes claros e objetivos. Facilita o trabalho em equipe e a manutenção;

4 - *Debuggar*: Testar diferentes cenários e entender o comportamento do algoritmo;

5 - Implantação na produção;

6 - Plano de retorno (se houver necessidade de voltar);

7 - divulgação e lançamento.

Na disciplina do PIME temos a oportunidade de experienciar e aplicar os conteúdos vistos de maneira prática e multidisciplinar. Nosso projeto é desenvolver um *chatbot* e um aplicativo para gerir o processo agendamento do projeto social Paz e Harmonia no Futebol coordenado pelo Avai F.C (Florianópolis- SC).

Através da disciplina de design de software aprendemos sobre fluxograma, pseudocódigo, teste de mesa, programar em módulos, *debuggar* etc. Estamos aplicando tudo em nosso PIME. Atualmente estamos finalizando a Interface do aplicativo. Iremos desenvolver e codificar a aplicação em módulos mas antes será feito o pseudocódigo para ser transcrito em Python ou JavaScript. O agente *chatbot* está concluído agora só falta integrar com o canal de atendimento do projeto. Também pretendemos realizar os testes de qualidade aprendidos.

Seguir os conceitos aprendidos ajudou o desenvolvimento do trabalho e principalmente na consistência. Portanto, o espaço de experimentação e de interdisciplinaridade é de suma importância para fixar, concretizar na prática e entender o que foi visto, é essencial para o processo de ensino-aprendizagem.

**5 REFERÊNCIAS BIBLIOGRÁFICAS**

Material didático da disciplina slide e e-book elaborado pelo Professor Márcio Calvacante;

Epic Games. **Debug Draw in Verse.** Disponível em:<https://dev.epicgames.com/documentation/pt-br/uefn/debug-draw-in-verse>. Acesso em: 29 nov. 2024.  
  
ABC Pneus. **Como proteger o computador do carro?** Disponível em:<https://www.abcpneus.net/como-proteger-o-computador-do-carro?srsltid=AfmBOoozCQknmFXTvvgO9-uceJ60W4m0i_X3FcEKC3oQy6qLDrIyX6Rm>. Acesso em: 29 nov. 2024.;

Total Energies. **O que é computador de bordo: vantagens e desvantagens.** Disponível em:<https://totalenergies.com.br/o-que-e-computador-de-bordo-vantagens-e-desvantagens>. Acesso em: 29 nov. 2024.

Cobli. **Computador de bordo: o que é e como funciona.** Disponível em:<https://www.cobli.co/blog/computador-bordo-carro/>. Acesso em: 29 nov. 2024.

Garetti. **O que são os computadores de bordo?** Disponível em:<https://www.garettiloja.com.br/o-que-sao-os-computadores-de-bordo/>. Acesso em: 29 nov. 2024.

Localiza Frotas. **Computador de bordo: o que é e como funciona.** Disponível em:<https://frotas.localiza.com/blog/computador-de-bordo>. Acesso em: 29 nov. 2024.

LINKEDIN. **How can you debug an AI system more effectively?** Disponível em:<https://www.linkedin.com/advice/1/how-can-you-debug-ai-system-more-effectively-5tbef?lang=pt&originalSubdomain=pt>. Acesso em: 29 nov. 2024.

LINKEDIN. **How do you debug AI errors to improve user experience?** Disponível em:<https://www.linkedin.com/advice/3/how-do-you-debug-ai-errors-improve-user?lang=pt&originalSubdomain=pt>. Acesso em: 29 nov. 2024.

Microsoft. **Train, debug, and deploy responsible AI models with Azure Machine Learning – Part 1.** Disponível em:<https://learn.microsoft.com/pt-pt/shows/make-azure-ai-real/train-debug-and-deploy-responsible-ai-models-with-azure-machine-learning-part-1>. Acesso em: 29 nov. 2024.

CompraCo. **Estratégias de teste de software de IA para controle de qualidade e depuração aprimorados.** Disponível em:<https://compraco.com.br/ru/blogs/tecnologia-e-desenvolvimento/estrategias-de-teste-de-software-de-ia-para-controle-de-qualidade-e-depuracao-aprimorados?srsltid=AfmBOoqPwKaMc1NO69zPAKZH-X-GtekJdPW9tCQ2UQcNtyqCoZealkKn>. Acesso em: 29 nov. 2024.

Knauf Automotive. **Carros com um pacote de software definido.** Disponível em:<https://knaufautomotive.com/pt-br/carros-com-um-pacote-de-software-definido/>. Acesso em: 4 dez. 2024.

MundoGeo. **Computadores veiculares da Advantech facilitam solução abrangente de gerenciamento de frota e entretenimento.** Disponível em:<https://mundogeo.com/2022/11/22/computadores-veiculares-da-advantech-facilitam-solucao-abrangente-de-gerenciamento-de-frota-e-entretenimento/>. Acesso em: 4 dez. 2024.

Alfatest. **Diag PC-2 completo.** Disponível em:<https://www.alfatest.com.br/software/diag-pc-2-completo/>. Acesso em: 4 dez. 2024.

LINKEDIN. **How can you use a software debugger for electronic systems?** Disponível em:<https://www.linkedin.com/advice/0/how-can-you-use-software-debugger-electronic-wi3oc>. Acesso em: 4 dez. 2024.

Oliver Wyman. **Debugging the car hardware-software disconnect.** Disponível em:<https://www.oliverwyman.com/our-expertise/insights/2020/jun/automotive-industry-at-the-crossroads/rd-and-production/debugging-the-car-hardware-software-disconnect.html>. Acesso em: 4 dez. 2024.

LINKEDIN. **What is the best approach for debugging and troubleshooting?** Disponível em:<https://www.linkedin.com/advice/0/what-best-approach-debugging-troubleshooting>. Acesso em: 4 dez. 2024.

UFMG. **Em livro premiado, professores da Ciência Política e da Computação propõem tratar algoritmos como instituições.** Disponível em:<https://ufmg.br/comunicacao/noticias/em-livro-premiado-professores-da-ciencia-politica-e-da-computacao-propoem-tratar-algoritmos-como-instituicoes#:~:text=Os%20algoritmos%20cumprem%20papel%20crucial,%E2%80%93%20ou%20seja%2C%20influenciam%20comportamentos>. Acesso em: 5 dez. 2024.