Assembly Quest: Plataforma Educacional Interativa para Ensino de Arquitetura e Organização de Computadores

Ana Beatriz R. Garcia[1] **Bruno de A. Correia**[2] **Leonardo A. Araújo**[3]

[1] Instituto de Ciência e Tecnologia – Universidade Federal de São Paulo (UNIFESP)
CEP 12247-014 – São José dos Campos – SP – Brasil
ana.garcia29@unifesp.br, bcorreia@unifesp.br,
arazo.leonardo@unifesp.br

Resumo. *Este trabalho apresenta o Assembly Quest, uma plataforma educacional web interativa desenvolvida para o ensino de conceitos fundamentais de Arquitetura e Organização de Computadores. A plataforma possui três simuladores principais: um simulador de CPU e Assembly com sistema de fases progressivas, um simulador de hierarquia de memória e operações de disco, e um simulador de dispositivos de entrada e saída (I/O) com visualização 3D. A plataforma utiliza tecnologias web modernas (HTML5, CSS3, JavaScript ES6+, Three.js) e integra um assistente de IA baseado em RAG (Retrieval-Augmented Generation) para fornecer orientação contextual aos estudantes. Os resultados demonstram que a abordagem gamificada e visual facilita a compreensão de conceitos complexos de arquitetura de computadores.*

1. Introdução

O ensino de Arquitetura e Organização de Computadores enfrenta desafios relativamente complexos devido aos conceitos abstratos envolvidos. Estudantes frequentemente têm dificuldade em visualizar e compreender como componentes de hardware interagem, como instruções são executadas em nível de máquina, e como diferentes níveis da hierarquia de memória operam. Tradicionalmente, o ensino desses conceitos depende fortemente de diagramas estáticos e descrições textuais, o que limita a compreensão prática dos estudantes.

Este projeto abrange o Assembly Quest, uma plataforma web interativa desenvolvida para superar essas limitações através de simulações visuais e interativas. A plataforma combina três simuladores especializados que abordam diferentes aspectos da arquitetura de computadores: execução de instruções Assembly, hierarquia de memória e operações de entrada/saída.

A principal contribuição deste trabalho é a integração de múltiplos simuladores em uma única plataforma, com visualizações 3D, sistema de gamificação através de fases progressivas, e assistente de IA contextual para orientação dos estudantes. A plataforma foi desenvolvida utilizando tecnologias web padrão, garantindo acessibilidade e facilidade de uso sem necessidade de instalação de software adicional.

2. Arquitetura da Plataforma

2.1. Visão Geral

O Assembly Quest é uma aplicação web *single-page* que integra três módulos principais de simulação:

- **Simulador de CPU e Assembly:** Executa código Assembly em uma CPU virtual com visualização 3D dos componentes, sistema de fases progressivas e feedback em tempo real sobre o estado dos registradores e memória.
- **Simulador de Memória e Disco:** Simula a hierarquia completa de memória (registradores, cache L1, RAM e disco), operações de leitura/escrita em disco, e gerenciamento de cache com visualizações interativas.
- **Simulador de I/O:** Simula oito dispositivos de entrada e saída (teclado, monitor, impressora, disco, mouse, scanner, webcam e alto-falante) com modelos 3D animados e simulação de operações reais.

2.2. Tecnologias Utilizadas

A plataforma foi desenvolvida utilizando uma *stack* baseada em padrões web:

Frontend: *HTML5*, *CSS3* e *JavaScript ES6+* para a estrutura e lógica da aplicação. O *CSS* utiliza *Grid* e *Flexbox* para layouts responsivos, e animações CSS para feedback visual.

Visualização 3D: *Three.js* para renderização de modelos 3D, animações de componentes e efeitos visuais. *OrbitControls* permite manipulação interativa da câmera em todos os simuladores.

Assistente de IA: Integração com *Google Gemini 2.5 Flash* através de *API REST*. O assistente utiliza *RAG (Retrieval-Augmented Generation)* para fornecer respostas contextuais baseadas no simulador ativo, fase atual e ações do usuário.

Web APIs: *Canvas API* para renderização de texto dinâmico em monitores e impressoras, *Web Audio API* para geração de som real no simulador de alto-falante, e *LocalStorage* para persistência de configurações.

2.3. Arquitetura de Componentes

A arquitetura do projeto segue um padrão modular, onde cada simulador está implementado como uma classe JavaScript independente:

- `AssemblyGame`: Gerencia a execução de código Assembly, ciclo de instruções, e visualização 3D da CPU.
- `MemorySimulator`: Implementa a hierarquia de memória, operações de cache e disco, com sistema de fases.
- `IO3DVisualizer`: Gerencia a visualização 3D e simulação de dispositivos I/O (*Input/Output*), incluindo animações e *feedback* visual.
- `AIAssistant`: Implementa o sistema RAG e comunicação com a API do Gemini.
- `AIChatUI`: Interface de chat arrastável e minimizável para interação com o assistente de IA.

3. Simuladores Implementados

3.1. Simulador de CPU e Assembly

O simulador de CPU implementa um conjunto de instruções Assembly simplificado, incluindo operações aritméticas (`MOV`, `ADD`, `SUB`, `MUL`, `DIV`, `MOD`), operações de

comparação (`CMP`), saltos condicionais (`JMP, JE, JNE`), e operações de entrada/saída (`IN, OUT`). O simulador mantém o estado de registradores (R1 até R4), memória principal (M0-M8), e contadores de programa (essas quantidades são simbólicas para simulação).

O sistema de fases progressivas apresenta desafios sequenciais, começando com operações básicas de movimentação de dados e evoluindo para operações aritméticas, saltos condicionais e entrada/saída. Cada fase possui objetivos diretos e sistema de pontuação baseado na estrutura da solução.

A visualização 3D mostra componentes da CPU com animações que indicam o fluxo de dados durante a execução de instruções. O log de execução fornece feedback detalhado sobre cada ciclo, bem como a operação realizada.

3.2. Simulador de Memória e Disco

Este simulador foca na hierarquia de memória e operações de armazenamento secundário. Implementa quatro níveis de memória com suas respectivas características:

- **Registradores:** 32 *bytes*, tempo de acesso de 1ns
- **Cache L1:** 64 KB, tempo de acesso de 1ns
- **RAM:** 8 GB, tempo de acesso de 100ns
- **Disco:** 1 TB, tempo de acesso de 10ms

O simulador inclui operações específicas para cada nível: `LOAD` e `STORE` para RAM, `CACHE` para cache, e `DISK_READ`, `DISK_WRITE`, `DISK_SEEK` para operações de disco. O sistema simula latência, incluindo tempo de busca (*seek time*) para operações de disco.

A visualização inclui um *grid* interativo de setores do disco, mostrando setores ocupados, operações de leitura/escrita em tempo real, e estatísticas de desempenho. O sistema de fases ensina progressivamente desde operações básicas de disco até integração completa da hierarquia.

3.3. Simulador de I/O (*Input/Output*)

O simulador de I/O consiste de oito dispositivos com modelos 3D detalhados e com animações específicas:

- **Teclado:** Simula buffer de entrada, geração de interrupções e animação de teclas pressionadas.
- **Mouse:** Simula cliques, movimento e scroll com feedback visual.
- **Scanner:** Simula varredura de documentos com barra de progresso visual.
- **Webcam:** Simula captura de fotos com efeito de flash.
- **Monitor:** Renderiza texto real em tempo real usando *Canvas API*, com animação de brilho
- **Impressora:** Simula impressão de documentos com texto renderizado em papel virtual
- **Alto-falante:** Gera som real usando *Web Audio API* com controle de volume
- **Disco:** Simula controlador de disco com DMA, rotação de pratos e movimento de cabeça

Cada dispositivo possui controles específicos na interface, estatísticas de operação, e animações que ilustram o funcionamento interno. Um componente central 3D visualiza o fluxo de dados entre a CPU e os dispositivos selecionados através de partículas animadas.

4. Assistente de IA Integrado

4.1. Arquitetura *RAG*

O Assembly Quest integra um assistente de IA baseado em *RAG (Retrieval-Augmented Generation)* que utiliza o modelo *Gemini 2.5 Flash* da Google. O assistente mantém uma base de conhecimento estruturada contendo:

- Conceitos teóricos sobre Assembly, hierarquia de memória e I/O
- Descrições detalhadas de cada fase educacional
- Comandos e instruções disponíveis em cada simulador
- Exemplos práticos e soluções sugeridas

4.2. Funcionalidades

O assistente adapta suas respostas com base no contexto atual:

Contexto de Simulador: Identifica qual simulador está ativo (CPU, Memória ou I/O) e ajusta a base de conhecimento utilizada.

Contexto de Fase: Quando o usuário está em uma fase específica, o assistente fornece informações sobre o objetivo, comandos necessários e dicas relevantes.

Histórico de Ações: Rastreia as últimas ações do usuário para fornecer orientação mais precisa e contextualizada.

Interface de Chat: Interface arrastável e minimizável que permite interação natural em português, com histórico de conversação e persistência de contexto durante a sessão.

5. Interface e Experiência do Usuário

5.1. Design Visual

A interface utiliza um tema escuro com componentes neon (verde, azul, magenta, amarelo) que cria uma estética futurista e tecnológica. O menu principal apresenta um fundo 3D animado com partículas, geometrias flutuantes e efeitos de iluminação dinâmicos, criados com *Three.js*.

Cada simulador possui uma interface composta por:

- Painéis informativos mostrando estado atual do sistema
- Editores de código com numeração de linhas (CPU e Memória)
- Visualizações 3D interativas com controles de câmera
- Logs de execução em tempo real

5.2. Feedback Visual

O sistema fornece feedback visual imediato através de:

- Animações de componentes 3D durante operações
- Destaque visual de elementos ativos (registradores, setores de disco, dispositivos)
- Efeitos de partículas para fluxo de dados
- Mudanças de cor e brilho indicando estados
- Mensagens de sucesso/erro claramente visíveis

6. Resultados e Avaliação

6.1. Funcionalidades Implementadas

A plataforma Assembly Quest foi desenvolvida com sucesso, implementando todos os componentes planejados:

- Três simuladores funcionais e integrados
- Sistema de fases progressivas com fases no simulador de CPU e no simulador de memória
- Visualizações 3D interativas para todos os componentes
- Assistente de IA com *RAG* funcional
- Interface responsiva e intuitiva
- Animações e *feedback* visual em tempo real

6.2. Características Técnicas

A implementação demonstra várias características técnicas, dentre elas:

Performance: Utilização eficiente de Three.js com otimizações de renderização, permitindo animações suaves mesmo com múltiplos objetos 3D simultâneos.

Modularidade: Arquitetura modular facilita manutenção e extensão. Cada simulador é independente e pode ser desenvolvido separadamente.

Extensibilidade: A estrutura permite adicionar facilmente novos dispositivos I/O, novas fases educacionais, e novos comandos Assembly.

Acessibilidade: Plataforma web pura não requer instalação, funcionando em qualquer navegador moderno com suporte a WebGL.

6.3. Contribuições Educacionais

A plataforma oferece várias vantagens educacionais:

Visualização Concreta: Conceitos abstratos são visualizados através de modelos 3D e animações, facilitando a compreensão.

Aprendizado Ativo: Estudantes interagem diretamente com os sistemas, escrevendo código e observando resultados em tempo real.

Feedback Imediato: O sistema fornece *feedback* instantâneo sobre erros e sucessos, facilitando o aprendizado por tentativa e erro.

Gamificação: Sistema de fases e pontuação motiva os estudantes a progredir e completar desafios.

Orientação Contextual: O assistente de IA fornece ajuda específica baseada no contexto atual do estudante.

7. Conclusão

Este trabalho se compôs pelo Assembly Quest, uma plataforma educacional web interativa para ensino de Arquitetura e Organização de Computadores. A plataforma integra três simuladores especializados com visualizações 3D, sistema de gamificação e assistente de IA contextual.

Os resultados demonstram que a abordagem de simulação visual e interativa facilita a compreensão de conceitos complexos de arquitetura de computadores. A integração de múltiplos simuladores em uma única plataforma oferece uma experiência de aprendizado relevante, enquanto o assistente de IA fornece suporte contextualizado que adapta-se às necessidades individuais dos estudantes.

A plataforma foi desenvolvida utilizando tecnologias web padrão, garantindo acessibilidade e facilidade de uso. A arquitetura modular permite futuras implementações, como novos dispositivos I/O, algoritmos de cache mais sofisticados, simulação de diferentes tipos de memória (SSD, NVMe), e mudança do modelo do assistente (melhoria em treinamento, modelos, entre outros parâmetros).

Trabalhos futuros incluem: avaliação da eficácia educacional da plataforma, implementação de sistema de avaliação automatizada, suporte para múltiplos usuários e colaboração (adaptações na IA), e integração com sistemas de gerenciamento de aprendizado (LMS).

Referências

[1] TANENBAUM, A. S.; BOS, H. *Modern Operating Systems*. 4th ed. Boston: Pearson, 2016.

[2] HENNESSY, J. L.; PATTERSON, D. A. *Computer Architecture: A Quantitative Approach*. 6th ed. San Francisco: Morgan Kaufmann, 2017.

[3] THREE.JS Documentation. *Three.js - JavaScript 3D Library*. Disponível em: `https://threejs.org/`. Acesso em: 2024.

[4] GOOGLE AI. *Gemini API Documentation*. Disponível em: `https://ai.google.dev/`. Acesso em: 2024.

[5] KHRONOS GROUP. *WebGL Specification*. Disponível em: `https://www.khronos.org/webgl/`. Acesso em: 2024.