Universidade Estatual de Campinas
Instituto de Computação

Documentação do Projeto
Iniciação Científica

Sistema Tutor Inteligente Mutimídia

Aluno: André Figueiredo de Almeida
Orientador: Fábio Nauras Akhras
Linha de Pesquisa: Informática aplicada à educação

Dezembro 2017

1 Sobre o projeto

1.1 Resumo

Nesse trabalho foi desenvolvido um sistema computacional tutor inteligente, que aceita diferentes tipos de mídia. Ele foi desenvolvido de maneira a ser executado em um navegador de internet. Os conteúdos, compostos por diferentes tipos de mídia, questões e opções de resposta, são previamente definidos pelo educador e organizados em uma estrutura de grafo de forma a permitir múltiplos caminhos no ensino.

1.2 Introdução

Sistemas tutores inteligentes têm como objetivo auxiliar professores e alunos no processo educativo, usando modelos computacionais para automatizar e personalizar a sequência e/ou conteúdo apresentado ao estudante sobre algum tema [1]. O uso de mídias, como filmes, na educação tem sido estudado [2] de forma a abordar temas da sala de aula e discutir interpretações diferentes, além de estudos que combinam filmes com interação [3]. Somado a isso, estudos abordam estratégias construtivistas de educação em sistemas tutores [4], que é a metodologia que esse trabalho segue. Este trabalho teve como objetivo estudar esses conceitos teóricos e desenvolver um sistema tutor que tenha suporte para multimídia e que possibilite se adaptar ao aluno de acordo com suas respostas, além de ser leve e portátil para ser executado em máquinas com sistemas antigos/defasados.

1.3 Discussão

Para que o sistema tutor seja compatível com o maior número de máquinas possível, sem depender de instalação de softwares adicionais (que podem não ser compatíveis por motivos de hardware e/ou software), o sistema foi arquitetado para rodar em cima de um navegador de internet e foi escrito em HTML5, CSS e JavaScript. As funcionalidades do HTML5 tais como suporte a vídeo e áudio, e do CSS são suportadas pela maioria dos navegadores [5][6]. O código desenvolvido em JavaScript é compatível com 97% dos navegadores atuais [7]. O HTML5 e o CSS são responsáveis pela interface gráfica (figura 1) e o JavaScript faz a parte computacional.

O educador deve, então, construir um conjunto de questões de múltipla escolha, associadas a conteúdos multimídia (vídeo, áudio, imagem). Ele deve então fazer uma relação entre as alternativas de resposta a cada questão (e conteúdo multimídia associado) e outras questões. A ideia é que, de acordo com a resposta do aluno, a questão seguinte tenha alguma relação com a resposta anterior, seguindo as teorias construtivistas [4]. Por exemplo, se o sistema tutor tem intenção de ensinar matemática, o educador pode criar respostas erradas que são geradas a partir de um erro de sinal na expressão. Na hora de construir a relação entre as respostas e outras questões, se o aluno optar por aquela resposta, o sistema leva a uma questão (e conteúdos multimídia associados) que reforça as regras de sinal. O sistema foi projetado de forma a ser genérico o suficiente para abordar inúmeras sequências, dando liberdade para vários cenários de adaptação. A responsabilidade da adaptabilidade fica, portanto, sobre a pessoa que projetar o sequenciamento de questões e conteúdos multimídia.

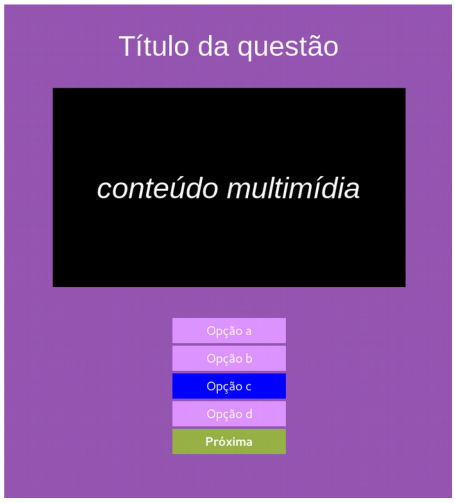

Figura 1. *Amostra da interface gráfica do sistema*

1.4 Referências

1. Brusilovsky, P. & Peylo, C. Adaptive and Intelligent Web-Based Educational Systems. International Journal of Artificial Intelligence in Education; 2003. 13:156-169.

2. Hobbs, R. Non-optimal uses of video in the classroom. Learning, Media and Technology; 2006 31, pp. 35-50.

3. Zahn, C., Krauskopf, K., Hesse, F.W., & Pea, R. Digital video tools in the classroom: Empirical studies on constructivist learning with audio-visual media in the domain of history. In: International Conference of the Learning Sciences. Chicago, IL; 2010.

4. Akhras, F. N. & Self, J. A. System Intelligence in Constructivist Learning. International Journal of Artificial Intelligence in Education; 2000 11(4):344- 376.

5. `https://www.w3schools.com/tags/tag_video.asp`

6. `https://www.w3schools.com/cssref/css3_browsersupport.asp`

7. `http://jscc.info/`

2 Arquitetura do sistema

O sistema é divido em dois módulos: controle e exibição. A parte de controle é responsável por exibir as perguntas salvas no banco de dados na ordem correta de acordo com o que foi planejado pelo instrutor do ambiente de aprendizado e de como o que o estudante está respondendo as perguntas. Já a parte da exibição é responsável por exibir para o aluno as perguntas corretas, juntamente com as mídias relacionadas aquela pergunta. Basicamente, o módulo de exibição é responsável pela interface gráfica do sistema. No começo, o estado do sistema corresponde a caixa *Pegar pergunta do banco de dados* da figura 2, onde ele vai obter a primeira pergunta. O sistema para quando *Verificar se é a última pergunta* for verdade.

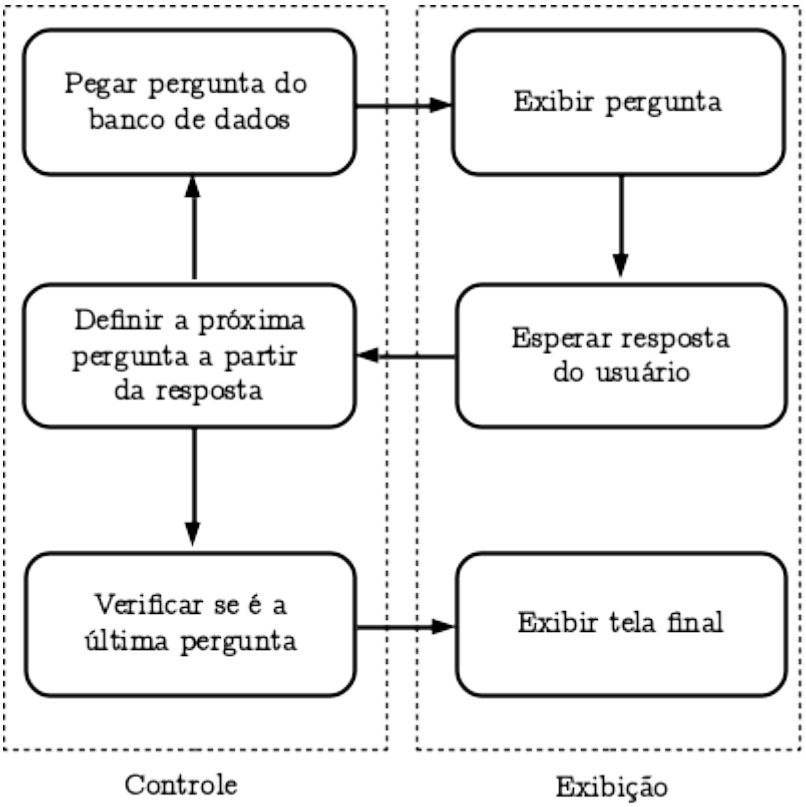

Figura 2. *Fluxograma de interação entre os módulos do sistema*

O módulo de **controle** corresponde ao arquivo:

- `js/quiz_engine.js`: programa que realiza as ações de controle.

O banco de dados que alimento o **controle**:

- `js/database.js`: base de dados de perguntas que alimentam o `quiz_engine`;
- e aos arquivos dispostos em `/media`, onde ficam as mídias das perguntas.

O módulo de **exibição** corresponde aos arquivos:

- `html/index.html`: estrutura da tela de perguntas do sistema;
- `html/end.html`: tela final do sistema;
- `css/style.css`: regras de estilo para os arquivos HTML.

3 Código fonte

Todo o código fonte se encontra em `src/`

3.1 `js/`

Em `js/` ficam os arquivos JavaScript, responsáveis pela parte lógica do sistema.

3.1.1 `database.js`

Aqui ficam registrados as perguntas e suas respectivas respostas do sistema. Esse é um arquivo de modelo e seu banco de dados pode ter outros nomes, mas é importante atualizar o nome no resto do sistema. Por padrão, o nome do arquivo é `database` e o nome da variável do banco é `db`. O banco é um grande array JSON chamado `questions`, onde cada elemento é uma pergunta com os seguintes atributos:

- `id`: o identificador único daquela pergunta, um número inteiro maior igual que 0.
- `title`: o título da questão.
- `text`: o texto da questão, a pergunta em si.
- `media`: se a pergunta conter algum arquivo de mídia associado, é necessário especificar:
  - `type`: o tipo de arquivo (text, image, audio ou video).
  - `file`: o arquivo da mídia, dentro da pasta `media`.

  se não houver mídia, o valor deve ser `null`
- `answers`: vetor de respostas de uma determinada pergunta. Cada resposta é composta por:
  - `t`: texto da resposta.
  - `p`: próxima pergunta que deve ser chamada caso o usuário opte por essa alternativa. Caso não exista próxima pergunta e o programa deve ir para a tela final, colocar o valor -1.

3.1.2 `quiz_engine.js`

O motor do sistema é o arquivo `js/quiz_engine.js`. É ele quem faz a correspondência entre a resposta do aluno e a próxima pergunta. As variáveis globais do programa são:

- `var database`: quando atribuída ao banco em `loadDatabase(name)` é o objeto onde é feita as buscas no banco de dados. Você pode acessar as perguntas usando `database.question[`*n*`]`, onde *n* é o número a questão que você quer acessar. De forma análoga, você consegue pegar as respostas `database.question[`*n*`].answers[`*m*`]` e seus respectivos atributos.
- `var clickedAnswer = 0`: é a variável que guarda a resposta que o usuário escolheu na pergunta atual, variando de 0 até o n - 1, onde n é o número de respostas disponíveis (no padrão, 4).
- `var actualQuestion = 0`: é a variável que guarda a pergunta atual, começando na primeira pergunta (pergunta número 0).
- `var questionsPath = []`: é o caminho que o usuário está seguindo, guardando os *id*s das perguntas por quais ele vai passando.

Quanto as funções, elas são:

- `function loadDatabase(name)`: carrega o banco de dados de nome `name` que está no arquivo `database.js` e coloca na variável global `database`. O banco é um objeto JSON.
- `function begin()`: é a primeira função a ser chamada para iniciar o funcionamento do sistema. Inicia o banco de dados e prepara a primeira pergunta (pergunta 0).
- `function setQuestion(num)`: verifica se o quiz acabou (pergunta -1). Se acabou, se direciona para a tela final e passa na URL o `questionsPath`. Se não é a última questão, vai no banco e busca o texto, a mídia e as alternativas. Coloca o texto no componente HTML adequado. Se a midia existir chama a função `setMedia(type, file)`, e, senão exitir, só remove a anterior. Pega as alternativas e escreve os textos delas no componente de respostas do HTML.
- `function selectAnswer(num)`: pega a resposta escolhida pelo usuário, coloca em `clickedAnswer` e muda o estilo dos botões, dando destaque para a resposta escolhida.
- `function getNextQuestion(answer)`: quando o usuário confirma sua escolha, essa função pega a próxima questão conforme a alternativa escolhida.
- `function insertMedia(html)`: insere o trecho HTML passado por parâmetro contendo a mídia na `div` HTML que comporta as mídias.
- `function setMedia(type, file)`: a partir do tipo e do arquivo de mídia passados por parâmetro, gera um código HTML para ser inserido na página.
- `function printEnd()`: a partir da URL, pega o caminho das respostas do usuário e coloca no HTML para ser exibido na tela.
- `function getUrlVars()`: função auxiliar de `printEnd()`, que pega os dados da URL.

3.2 `html/`

Nesta pasta se encontram os arquivos HTML responsáveis pela estrutura visual da página web.

3.2.1 `index.html`

- `head`: metadados da página, como localização dos arquivos JavaScript e CSS e o `charset`, definido como *UTF-8* possibilita a utilização de caracteres do alfabeto latino (acentos e cedilha). Caso o nome de alguns dos arquivos de `js/` e `css/` forem alterados, devem ser alterados aqui também para permitir que a página encontre os arquivos.
- `body`: aqui se encontram as estruturas da página. O atributo `onload`, com o valor `begin()` significa que essa função será chamada quando a página estiver carregando, fazendo com que a primeira pergunta seja carregada.
  - `<h3 id="question_text">`: aqui fica o texto da pergunta;
  - `<div id="media_div">`: divisão para o elemento multimídia;
  - `<table class="answer_list">`: tabela com as alternativas de respostas possíveis, onde cada célula é um botão.
    - `<button class="answer_button"...`: Note que cada botão deve ter um *id* único, começando no 0 e indo de forma crescente, da mesma forma que seu parâmetro em `onclick="selectAnswer(`*id*`)"` deve ter o mesmo número.
    - `<button class="control_button"...`: Quando este botão é clicado, ele chama a função definida em `onclick` que, a partir da alternativa selecionada, busca qual a proxima pergunta e prepara a tela com ela.

3.2.2 `end.html`

Esta página exibe a sequência utilizada pelo usuário no sistema, no elemento `<p id='end'>`.

3.3 `css/`

Nesta pasta fica o arquivo CSS, responsáveis pelas regras de estilo do HTML.

3.3.1 `style.css`

Esse arquivo define como será a aparência da página, em termos de cor, tamanhos e espaçamentos.

- `body`: define a fonte, cor de fundo, cor da fonte, margem e alinhamento do texto.
- `.control_button`: definem as características do botão de confirmação de resposta. Define cor de fundo, cor do texto, tamanho da fonte e tamanho do botão.
- `.control_button:hover`: definem as características deste botão quando o mouse está sobre ele, tornando-o com fundo preto.
- `.answer_button`: definem as características do botão de alternativa de resposta. Define cor de fundo, cor do texto, tamanho da fonte e tamanho do botão.
- `.answer_button:hover`: definem as características deste botão quando o mouse está sobre ele, tornando-o com fundo preto.
- `.answer_button_clicked`: muda a cor do botão de resposta quando este foi clicado.
- `.answer_list`: retira as margens da lista de alternativas.
- `#question_text`: define o tamanho do texto da pergunta.

- `#media_div`: define uma margem para a divisão de mídia.
- `.text-body`: remove a margem para o corpo da mídia, quando esta for um texto.
- `.text`: define o tamanho da fonte e o alinhamento da mídia, quando esta for um texto.

3.4 `media/`

Nesta pasta devem ser salvas as mídias que serão utilizadas pelo quiz. É importante verificar quais tipos de arquivo são compatíveis com o navegador.