UNIVERSIDADE DO ESTADO DE SANTA CATARINA - UDESC

CENTRO DE EDUCAÇÃO DO PLANALTO NORTE - CEPLAN

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

INTERPRETADOR DE ALGORITMOS PORTUGOL EM HTML5

São Bento do Sul, SC

2016

GILBERTO EDMUNDO TAVARES

INTERPRETADOR DE ALGORITMOS PORTUGOL EM HTML5

Trabalho de Conclusão apresentado ao Curso de Bacharelado em Sistemas de Informação, da Universidade do Estado de Santa Catarina, como requisito para a obtenção do grau de Bacharel em Sistemas de Informação

Orientador: Dr. Luiz Cláudio Dalmolin

São Bento do Sul, SC

2016

GILBERTO EDMUNDO TAVARES

INTERPRETADOR DE ALGORITMOS PORTUGOL EM HTML5

Trabalho de Conclusão apresentado ao Curso de Bacharelado em Sistemas de Informação, da Universidade do Estado de Santa Catarina, como requisito para a obtenção do grau de Bacharel em Sistemas de Informação

Banca Examinadora

Orientador:

Dr. Luiz Cláudio Dalmolin
Universidade do Vale do Itajaí

Membros:

Dr. Nilson Ribeiro Modro
Universidade Federal de Santa Catarina

Ma. Nelcimar Ribeiro Modro
Universidade Federal do Paraná

São Bento do Sul SC, 29/11/2016

Dedico este aos meus familiares falecidos:
Meu irmão que sempre apoiou minha formação superior e carreira até oferecendo certa colaboração financeira.
E ainda mais recente meu pai que financiou meu pré ingresso e não pude compartilhar com ele esta conquista.

AGRADECIMENTOS

A minha mãe me apoiou em minha decisão de cursar faculdade em outra cidade. E sempre que possível, mesmo com dificuldade, ajudou como pode com móveis para meu primeiro quarto, algumas compras e mercado e algum dinheiro quando ficava mais crítico. Inclusive isso tudo pode ter sido um dos motivos da decisão dela em vender o imóvel em morávamos.

A toda a família Lischka que me acolheu inicialmente como hóspede na residência do casal Arnaldo e Jacira (prima da minha mãe). Foi também na empresa familiar deles juntos com seus filhos que tive meu primeiro emprego na cidade e me cedido para moradia o quartinho e dependências junto à empresa.

Ao meu irmão que me empregou as sábados e minhas cunhadas que me hospedavam, além de amigos. A todos os motoristas que me deram carona, econominanzo assim a parassagem para que eu pudesse voltar com uma grana a mais do final de semana.

A Joseli que sua experiência passando pelo mesma situação de fim de curso foi de grande ajuda. Também seu apoio e motivação, bem como dos meu amigos de infância José e Rafael que fizeram o mesmo.

A toda a experiência social, esportiva e de conheciento que as confraternizações com outros academicos, Jiudesc e os vários Lantinoware proporcionaram.

Sem esquecer é claro da minha garrafa térmic,a que manteve tanto café quentinho para embalar as noites dedicadas a esta realização.

“A maioria dos bons programadores programa não por esperar ser pago ou adulado pelo público, mas porque é divertido programar.” (tradução nossa)

Linus Torvalds

RESUMO

Trata-se do estudo das opções para prática computacional no processo de ensino e aprendizagem, sugerindo um conceito de ferramenta que diferente das existentes permite acesso via navegador de internet sem instalação ou download mesmo que apenas de complementos ou já disponíveis no computador. Vários conceitos são apresentados sobre algoritmos sua estrutura e seus "transformadores", sejam eles compiladores, tradutores ou interpretadores. Mas para o foco do trabalho apenas instruções simples de escrita em tela sem utilização de variáveis foi utilizada, permitindo demonstrar seu diferencial que além de acesso via navegador, nativamente (diferindo das demais soluções similares), o acesso mesmo sem conexão com a internet.

Palavras-chaves: Lógica da programação. Algoritmos. Portugol. Interpretador. HTML5.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

HTML	*HyperText Markup Language*
UAL	UNESA *Algorithmic Language*
UNESA	Universidade Estácio de Sá

SUMÁRIO

1 INTRODUÇÃO

A base da programação é ensinar máquina a realizar determinada tarefa, para que isso seja possível é necessário que o usuário comunique-se com a máquina, numa linguagem que ela entenda, a linguagem de programação. A construção da linguagem de programação envolve ações definidas em passos que se deseja que a máquina execute, a estes passos damos o nome de algoritmos. Para isso pode-se dizer que independente do paradigma utilizado no processo de desenvolvimento ele envolve algoritmos (MEDEIROS, 2015).

Algoritmos podem não necessariamente serem computacionais, já que são um conjunto de passos finitos. Analogias simplistas são os ingredientes e modo de preparo de uma receita ou instruções para uma tarefa do cotidiano como a troca de um pneu furado em um automóvel (MEDINA; FERTIG, 2006). Porém se tratando de computação deve ter maior formalidade, seguindo uma linguagem pois diferente do ser humano uma máquina não é tão interpretativa. Desconsideradas aqui inteligências artificiais que são desenvolvidas por algoritmos para terem comportamento de compreensão humana. Isto pode ser analogamente algo como não entender idioma estrangeiro e as instruções estarem nesta linguagem. Porém em informática isso tem que ser mais especificado com palavras ou caracteres predeterminados de início e fim, de blocos, repetições, etc (DERSHEM; JIPPING, 1990).

Sendo o berço mais popular da informática os Estados Unidos da América e seus criadores em muitos casos dessa origem ou de países com mesmo idioma as primeiras linguagens de computação e a maioria delas são em inglês (SEBESTA, 2009).

Isso muda quando falamos do Portugol que possui variações, mas resumidamente sua origem vem como uma tradução de uma linguagem de programação simples e comum, com seus termos no idioma nativo do Brasil.

Tendo como foco o aprendizado de algoritmos para uma boa fundamentação de futuros programadores ou profissionais da área, e muitas vezes a facilidade com o inglês não é algo comum em nosso país, a linguagem Portugol se torna um artifício vantajoso (JESUS et al., 2004).

1.1 OBJETIVOS DO TRABALHO

1.1.1 Objetivo Geral

Apresentar o conceito de ferramenta computadorizada para se tornar referência em criação, edição, visualização e interpretação de algoritmos em pseudo linguagem, como alternativa às existentes. Sendo seu diferencial o acesso nativo a partir de navegadores *Web* modernos. Inclusive acessível quando uma conexão com a internet esteja indisponível.

1.1.2 Objetivos Específicos

Visando atingir o objetivo geral em sua totalidade lista-se a seguir objetivos específicos de modo sucinto:

- Desenvolver o protótipo, sendo seu código fonte aberto e disponível para que outros possam manter, ampliar e evoluir a aplicação;
- Fazê-lo compatível com a sintaxe desenvolvida na Universidade Estácio de Sá (UNESA) a UNESA *Algorithmic Language* (UAL). E mínimo produto viável para a primeira lição de ensino aprendizagem;
- Permitir o acesso via navegadores *Web* modernos, sem necessidade de instalações complementares;
- Aplicar aprimoramentos introduzidos na versão 5 da *HyperText Markup Language* (HTML) para uso sem conexão com a internet.

1.2 JUSTIFICATIVA DO TRABALHO

Praticar em *software* à abstração em papel auxilia e muito no processo de ensino e aprendizagem, em algoritmos além do pensamento lógico por etapas já pode ser desenvolvido e descrito para o repasse a um computador. “O Aprendizado de algoritmos não é uma tarefa muito fácil, só se consegue através de muitos exercícios” (LOPES; GARCIA, 2002, p. 1).

Para esse ensino é essencial conhecer dois conceitos principais: lógica de programação e algoritmos. Primeiramente é apresentado um paralelo de situações do cotidiano com algoritmos, ao efetuar comparações de instruções passo a passo realizadas diariamente, como fritar um ovo ou trocar um pneu de automóvel. A seguir inicia-se a aplicação de linguagem estruturada em português para descrição dos algoritmos. Em alguns casos, utiliza-se de linguagens em inglês, como por exemplo, Pascal. Há professores que preferem usar pseudo linguagem em português somente “em papel”, iniciando a codificação diretamente em linguagem C. Tem-se inclusive, também “em papel”, de analisar e interpretar os passos do algoritmo para saber quais serão os valores em memória e a saída na tela.

Simultaneamente, com a prática escrita, sem software específico, pode-se simular os mesmos algoritmos em interpretador ou compilador. Sendo em um interpretador sua construção dividida em partes, cada uma com uma função específica. Geralmente identifica-se: o analisador léxico, o analisador sintático, o analisador semântico, e o resultado de saída. Caso seja um compilador há, ao invés de resultado de saída, o gerador de código, que transforma o programa em linguagem de máquina composta de 0s e 1s (DELAMARO, 2004).

Diante do exposto, este trabalho justifica-se pelo empenho em facilitar o processo de aprendizagem de algoritmos, desenvolvendo um programa em navegador via Internet, fazendo com que seja desnecessária a instalação do programa, podendo ser acessado a partir de qualquer computador, com o objetivo que os exercícios sejam resolvidos com maior praticidade.

1.3 LIMITAÇÕES DO TRABALHO

Limitou-se o trabalho numa imposição que vem preservar que o tempo disposto seja viável para sua execução.

Por seguir um processo bem diferenciado de construção optou-se por sua mínima execução. Permitindo apenas algorítmos muito simples pois um dos focos é a disponibilidade independente de sistema operacional ou instalações.

Outra limitação foi a não possibilidade de gravar e manter executáveis da saída gerada do algoritmo. Devido ao entendimento que é ponto não essencial para a prática do conhecimento e que uma possível solução no modelo pretendido seria dispendioso e demorado.

Devido a depender de estrutura adicional como servidores e hospedagem além de programação adicional desta interface também não há a gravação automática dos códigos para acesso irrestrito. Cabendo ao salvamento e abertura pela máquina. Levou-se em conta que há vários serviços para tal atividade com no caso cabendo ao usuário.

O sistema de tratamento de erros foi explorado até onde complexidade dos algoritmos possíveis dentro do proposto permitiu. Realizados assim neles os testes de erros possíveis.

1.4 ESTRUTURA DO TRABALHO

O presente trabalho está estruturado em cinco capítulos que abordam o Processo de Ensino e Aprendizagem comum na atualidade e as Ferramentas Utilizadas, detalhes sobre o ensino com a utilização de Processamento de Linguagem Natural em Algoritmos.

Após o primeiro o anterior introdutório o capítulo dois descreve principais termos tem sua teoria descrita além de apresentar as principais ferramentas utilizadas no ensino e prática de lógica de programação.

Os capítulos seguintes apresentam no três a metodologia utilizada e no quatro aplicação em seu desenvolvimento até o seu resultado, apresentando alguns temos pertinentes somente ao processo.

Por fim o capítulo cinco conclui-se com as considerações finais, recomendações, sugestões e trabalhos futuros, podendo ser pelo autor ou outros interessados.

2 FUNDAMENTAÇÃO TEÓRICA

Para o entendimento e desenvolvimento da proposta conforme explícita nos objetivos gerais e específicos, alguns conceitos teóricos devem ser dispostos. Algoritmos do conceito ao modelo utilizado, variações de modelos algorítmicos e outros ambientes de estudo, ensino ou prática.

2.1 ALGORITMOS

A última das notas de Ada Lovelace, que foram republicadas em 1953, apresenta os primeiros conceitos sobre programação, descrevendo um algoritmo. Desde então estes foram difundidos, sendo utilizados até a atualidade (SANTIAGO; DAZZI, 2003).

Conforme Medina e Fertig (2006) algoritmo é um termo mais amplo, sendo inclusive destinado a outras áreas e finalidades além da programação. As encontradas na literatura são interpretativas porém todas chegam ao mesmo resultado, um **conjunto de instruções para a resolução de uma situação**.

- Estruturas de Controle: como desvio condicional, laços de repetição;
- Tipos de Dados: o tipo de valor que será inserido na variável;
- Atribuições: inserir um valor em uma variável.
- Operações aritméticas: utilizadas para cálculos entre números e variáveis;
- Operações relacionais: para estabelecerem relação entre duas comparações;
- Variáveis: para armazenar algo na memória principal;
- Vetores: conjunto de variáveis com o mesmo nome, contendo um índice para diferenciá-las;

2.2 LINGUAGEM NATURAL

Com menor rigidez do que se faz necessário para uma máquina, esses conjuntos também podem ser descritos em uma linguagem humanamente mais natural. Numa conversa ou em uma explicação de rota de um destino ou uma receita culinária o entendimento ocorrerá. Por depender de formalidade e buscando a aproximação com a rigidez das linguagens de programação utiliza-se o idioma estruturado. Em nosso caso o português estruturado, mas há também o inglês estruturado e certamente outras (MEDINA; FERTIG, 2006).

2.3 PROGRAMAS DE COMPUTADOR

Sua estrutura é E → P → S (entrada-programa-saída), podendo não haver argumentos de entrada. Exemplo, um programa após compilado pode receber em sua execução como argumento um número inteiro para resultar em sua raiz quadrada, ou outro sem receber argumento já tenha a raiz quadrada de um número pré definido (MEDINA; FERTIG, 2006).

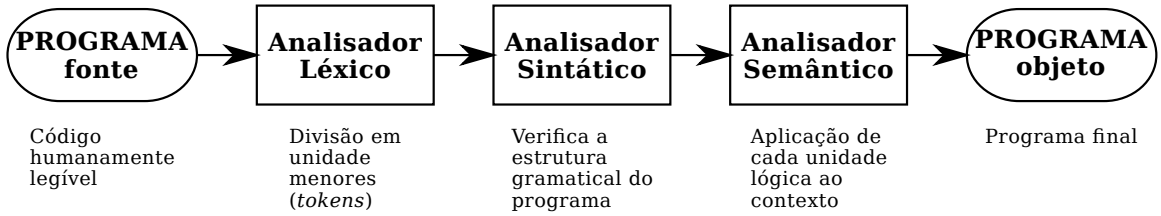
2.4 PSEUDO CÓDIGO OU LINGUAGEM

Visando ser uma ponte entre a linguagem natural e as linguagens de programação temos as pseudo-linguagem. Notadamente Portugol é evidenciada como a solução, pois desde Saliba (1992) é confirmada sua ampla utilização. Porém não há uma padronização existindo inúmeras variações, no entanto com muitas semelhanças entre si (MEDINA; FERTIG, 2006).

2.5 COMPILADORES E INTERPRETADORES

Passando por um processo de entendimento de elementos mínimos no código, ignorando outros e agrupando em símbolos especificados, a compilação se inicia. Além dessa análise também é necessária a verificação se esses grupos estão em ordem ou outros erros possíveis. Citar (0000) descreve que código fonte de entrada termina tendo como saída o programa objeto, o compilador converte a linguagem simples em uma linguagem entendida pela máquina

Figura 1 – Etapas de um compilador

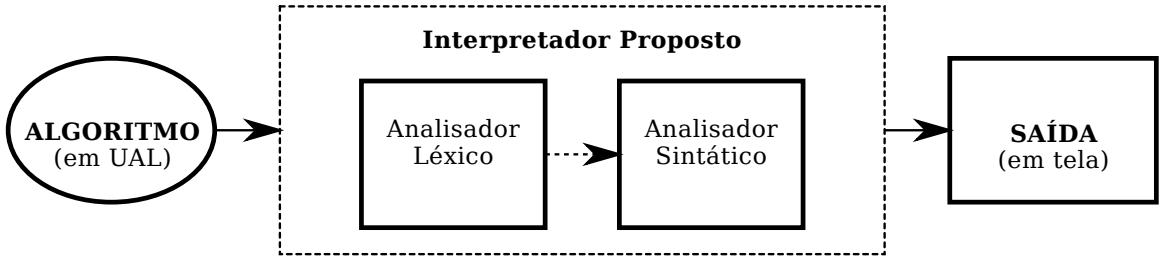


Fonte: Autor, adaptado de Ferrandin e Stephani (2005).

Explicado o processo do compilador de maneira simples e objetiva é preciso ainda nomear tecnicamente além de expor sua etapas. Os elementos mínimos são os tokens, numa instrução cada letra, número e outros caracteres (como pontuação e demais símbolos) além do espaço são identificados, essa etapa é denominada **Análise Léxica**.

Esse elementos mínimos definidos na etapa anterior são então agrupados, em: palavras reservadas, comandos, conjunto de texto, nome de variáveis, inícios e fins do algoritmos ou de blocos nele, que juntamente com a análise anterior, denomina-se **Análise Morfológica**. Por fim esse grupos devem estar numa ordem que tenha sentido para sua execução correta, sendo então a **Análise Sintática**.

Figura 2 – Interpretador proposto



Fonte: Autor, adaptado de Esmin (1998).

Tendo as mesmas primeiras etapas análoga ao compilador o interpretador não necessita a saída compilada, sendo imediatamente executado após a formação da árvore gramatical (CITAR, 0000)

2.6 UAL - ENSINO COM LIVRO TEXTO

O livro "Introdução à programação: 500 algoritmos resolvidos" tem um grande apelo por seu elevado número resoluções como seu título deixa claro. Isso motiva os educadores utilizarem como livro texto base de disciplinas de ensino de Lógica da Programação (CITAR, 0000).

Neste livro é relatado o desenvolvimento da Linguagem UAL em um trabalho desenvolvido na UNESA. Era basicamente compilador via linha de comando, desenvolvido em Haskel para linux. Ele foi posteriormente portado para sistema operacional proprietário mais popular, e para este recebeu um editor próprio Editor UAL (CITAR, 0000).

Todos os algoritmos propostos para execução computadorizada são impressos nesta sintaxe variante do Portugol. Porém acompanha uma mídia ótica que contém os exercícios nesta e também para a desenvolvida por Evaristo e Crespo (2000) o Interpretador de Linguagem Algorítmica (ILA) (CITAR, 0000).

Quadro 1 – Comparativo UAL e ILA

Em UAL	Em ILA

`prog algoritmo11`
`  imprima "Aprendendo prog!!!";`
`fimprog`
	`//prog algoritmo11`
	`inicio`
	`  limpar`
	`  escrever "Aprendendo inicio!!!"`
	`fim`

Fonte: Autor, a partir dos exemplos de Lopes e Garcia (2002).

Todo o projeto está pautado neste algoritmo simplista, porém se nota a diferença nos blocos de início e fim do programa, além de somente em UAL ter o nome do mesmo (solucionado com uso de linha comentada) e o comando de limpeza somente necessário em ILA. As chamadas

de escrita em tela também têm sua palavra reservada diferente. Nenhuma delas delimita o argumento por parênteses e UAL exige ponto e vírgula ao final. As duas não são sensíveis no uso de letras maiúsculas ou minúsculas nas palavras chaves (CITAR, 0000).

2.7 FERRAMENTAS EXISTENTES

- **Portugol/Plus**: Desenvolvida por Esmin (1998) quando vinculado a UNOESC, é a mais antiga opção encontrada no Brasil. Utiliza ainda em plataforma Disk Operating System (DOS), inclusive com interface gráfica. Comumente referenciada por outros autores inclusive quando são sobre desenvolvimento de novas soluções.
- **Construtor**: Desenvolvido pelo Centro Educacional de Informática Aplicada do SENAC (Rio de Janeiro), é a primeira ferramenta encontrada para uso em plataforma Windows e tem a vantagem acompanhar o livro "Construção de Algoritmos" de Fernandes e Botini (1999).
- **VisuAlg**: Desenvolvida por Claudio Morgado de Souza profissional programador e analista bem como professor universitário, é utilizada tanto em cursos técnicos quanto em meio universitário. A facilidade de obtenção do seu executável para Windows (não necessária instalação), documentação e simplicidade, sem perder funções úteis para iniciantes, são grandes atrativos (SOUZA; FRANÇA, 2013).
- **Web Portugol**: Desenvolvida por pesquisadores na UNIVALI permite o acesso via navegadores que tenham integração com Java, desde que o mesmo também esteja instalado no computadors (SOUZA; FRANÇA, 2013).

Abaixo um comparativo de pontos que ressaltam como fatores determinantes em escolha da opção ao uso.

Quadro 2 – Comparativo das ferramentas

	VisuAlg	Web Portugol	Portugol/Plus	Contrutor
Plataforma	Windows	Web (Java Applet)	DOS	Windows
Licença	*freeware*	*open source*	*open source*	*open source*
Instituição		UNIVALLI	UNOESC	SENAC
Linguagem		Java		

Fonte: Produção do autor.

Ao delimitar a quantidade de itens algumas não foram listadas mas também merecem menção: **AMBAP**, pelo Departamento de Tecnologia da Informação da Universidade Federal de Alagoas (TCI/UFAL); **TBC-AED**, por pesquisadores da Universidade Federal de Lavras/Departamento de Ciências da Computação em Minas Gerais; PascalX, por Athur Vargas Lopes da ULBRA; **Web-UNERJOL**, por acadêmico na UNERJ.

Outras mais não foram consideradas por fugirem do escopo ao utilizar robótica, jogos ou similares: Guido VanRobot, Robot Prog, Kids Ruby, Fut Code. Eram pretendidas somente ferramentas com pseudo-linguagem algorítmica em português, preferencialmente Portugol, partindo novamente do Editor UAL como referencial.

3 MÉTODOS DE PESQUISA

Segundo Prodanov e Freitas (2013) a pesquisa aplicada tem como propósito produzir compreensão sobre determinada questão objetivando aplicação prática, assim esta pesquisa classifica-se como aplicada, pois busca resolver o problema de acessibilidade ao softwares de prática no aprendizado de algoritmo.

3.1 CLASSIFICAÇÃO DAS PESQUISA

Quanto aos objetivos a pesquisa qualifica-se do tipo exploratória, procurando conhecer as ferramentas existentes para aprendizagem de algoritmo.

A fase exploratória da pesquisa tem como função promover mais conhecimentos sobre o assunto (PRODANOV; FREITAS, 2013).

3.2 MÉTODO CIENTÍFICO

Quanto aos procedimentos, a pesquisa realizou-se através de levantamento bibliográfico, realizada através de material já publicado (PRODANOV; FREITAS, 2013). Apurando artigos científicos, teses e monografias disponíveis publicamente na internet, principalmente via Google Acadêmico.

Inicialmente a pesquisa bibliográfica foi instituída tendo como meta os últimos 10 anos, mas como foram encontradas publicações relevantes vários anos anteriores esse valor foi alterado para o dobro, 20 anos. Nenhuma relevância encontrada no Brasil anterior à 1998 passando esse a ser o limiar, com dois trabalhos neste ano.

3.3 UNIVERSO DA PESQUISA

Segundo Prodanov e Freitas (2013) o universo da pesquisa caracteriza-se pela similaridade de características em uma população. Desta forma esta pesquisa está inserida no universo de desenvolvimento de software.

3.4 POPULAÇÃO E AMOSTRA

Amostra da pesquisa define-se pela parcela de individuos inseridas no universo da pesquisa. (PRODANOV; FREITAS, 2013). Assim sendo, a amostra da pesquisa tem como alvo os softwares desenvolvidos para prática de exercícios durante o aprendizado de algoritmos.

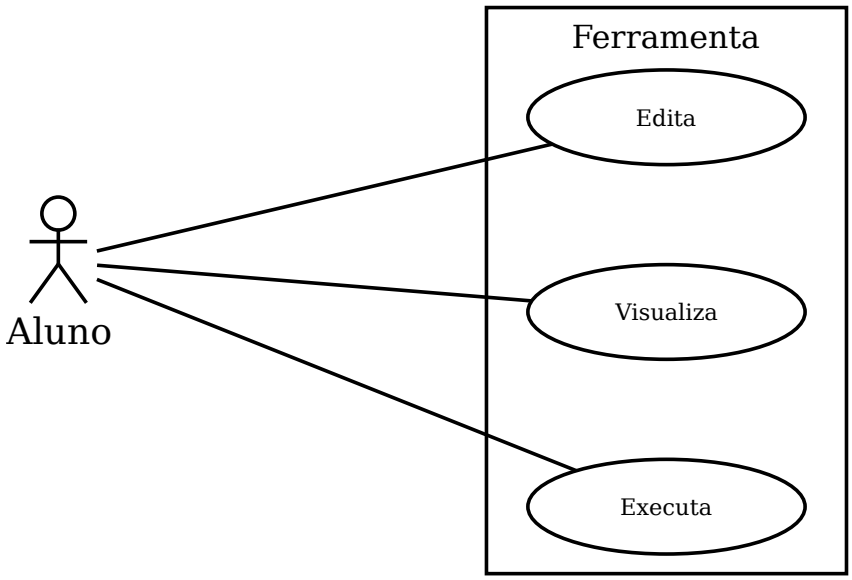
4 DESENVOLVIMENTO DA APLICAÇÃO

A criação do protótipo desse sistema foi o ponto principal desse trabalho, pois os objetivo foi oferecer opção para substituir *softwares* que vêm sendo utilizados atualmente por um prático, atual, acessível e com o código disponível para que possa ser corrigido e melhorado.

Nesse protótipo buscou-se eliminar algumas deficiências básica que existem em ferramentas similares, por exemplo funcionalidade de desfazer durante a edição, pois essas podem fazer com que o seu uso seja desgastante e desanimador. Como mencionado nas limitações a inclusão de funcionalidades que poderiam estar disponíveis ficarão como sugestões, algumas delas são encontradas por exemplo em Ambientes Integrados de Programação, comumente utilizada em outras linguagens.

Inclusive há caso em que a ferramenta inicialmente utilizada não é capaz de executar alguns algoritmos mais avançados. E isto leva a necessidade do uso de outro software na mesma disciplina até com possivelmente uma nova linguagem (por exemplo a linguagem de programação C ou C++), devido a está necessidade alguns educadores optam por esta sendo a única ferramenta utilizada, ignorando linguagem Portugol. Espera-se que diferente o UAL essa limitação possa realmente ser concluída em trabalho futuros, conforme proposto nas conclusões.

Figura 3 – Caso de uso Aluno

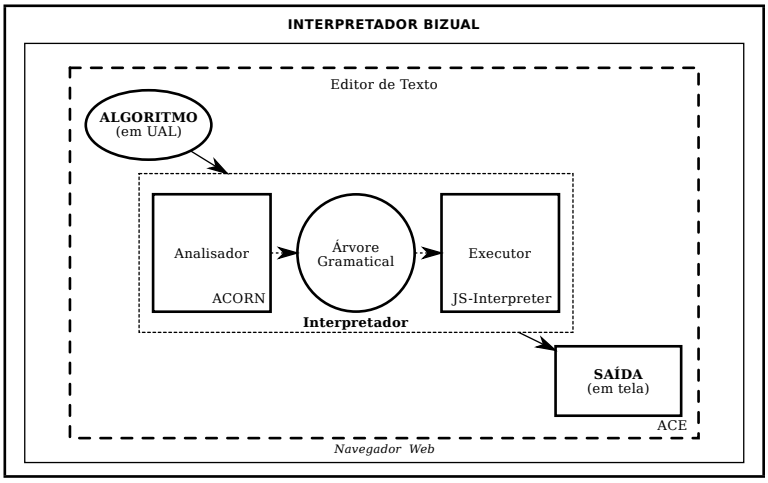


Fonte: Produção do autor.

Essencialmente a aplicação foi desenvolvida em linguagens Web, o JavaScript sendo a linguagem de programação, HTML como linguagem de marcação e CSS como estilização. Nesse

modelo já existem editores de código fonte de linguagens de programação,dos mais utilizados temos ACE e o CodeMirror, optado pelo primeiro pois em análise entendeu-se como sendo mais prático para o uso e criação do modelo da linguagem. Considerou-se a criação do modelo gramatical para destaque da sintaxe como dificuldade relativamente baixa.

Figura 4 – Esquemático do Interpretador

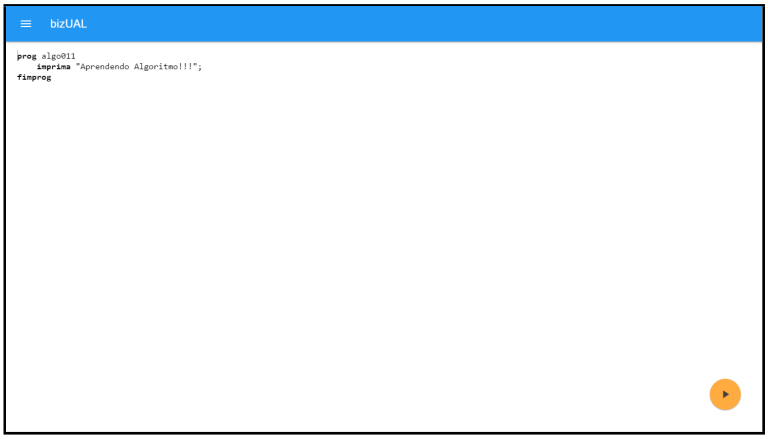


Fonte: Autor

Para resultar na árvore gramatical encontrou-se o ACORN um parser já escrito em JavaScript. De modo trabalhoso foi necessário adaptar o código fonte para compreender a estrutura do algorítmos proposto, principalmente devido à não utilização de parênteses para o comando de impressão em tela na linguagem UAL.

Como saída o interpretada o JS-Interpreter foi encontrado como a única opção. Ele já se utiliza do mesmo ACORN mencionado acima e interpreta a própria linguagem JavaScript. Pelo mesmo motivo citado acima, falta de parênteses no comando, se tornou uma tarefa um tanto dispendiosa fazer com que ele se adaptasse ao modelo.

Figura 5 – Interface Desktop

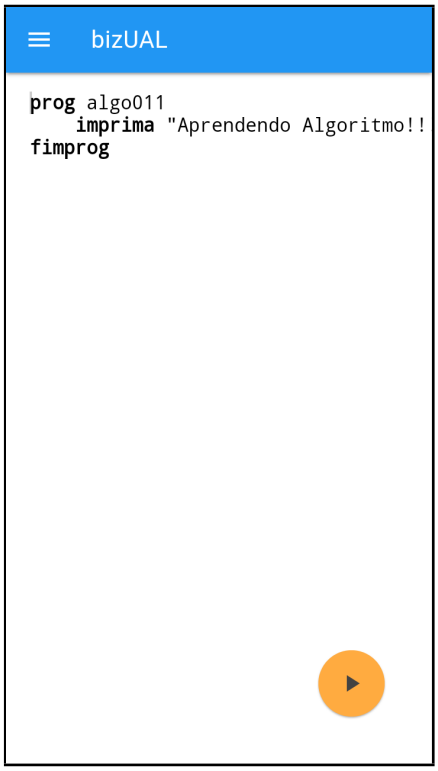


Fonte: Produção do autor.

Como interface gráfica foi mantido o mínimo necessário. Somente o editor de texto e um botão para executar. Foram visualizadas renderizações e tamanhos de telas variados, computadores (Figura 5), tablets, smartphones (Figura 6), etc. Seguindo conceitos do Google Material Design foi adicionada uma barra no topo como botão para menu o nome da aplicação, denominada Bizu, e um botão flutuante no canto inferior direito como ícone "play" para a execução do algoritmo.

De acordo com o dicionarioinformal.com.br (2016) a gíria "bizu" teve origem em quartéis militares, onde os experientes sussurravam dicas para os novatos. Os superiores de longe ouviam aquela onomatopéia característica de cochicho "bzbzbzu", dando origem ao termo.

Figura 6 – Interface Smartphone



Fonte: Autor

Facilitando a tarefa de construção da interface foi utilizada a biblioteca Material Design Lite (MDL) disponibilizada oficialmente pelo Google. As cores foram escolhidas pela funcionalidade disponível no site da biblioteca.

A escolha do nome se deve a partir da palavra bizu adicionada a sigla da sintaxe.

Foi utilizado a funcionalidade *app cache* a partir de um arquivo manifesto listando o conteúdo que o navegador deve guardar cópia local. Podendo assim ter a internet desconectada após o primeiro acesso que a aplicação continuará acessível.

Todo o código foi desenvolvido em repositório disponível no github e quando concluído em sua versão mínima movido para seu destino final e podendo ser acessado.

4.1 FERRAMENTAS UTILIZADAS

4.1.1 Editor de Código ACE

4.1.2 *Parser* Acorn

4.1.3 JS-Interpreter

4.2 PROCESSO CRIACIONAL

4.2.1 Git e Github

4.2.2 Node.js e Grunt

4.2.3 SASS

4.2.4 TDD

4.3 INTERFACE GRÁFICA

4.3.1 *Material Design*

MDL

4.3.2 Inspirações

5 CONSIDERAÇÕES FINAIS

5.1 RELAÇÃO ENTRE OS OBJETIVOS E OS RESULTADOS OBTIDOS

5.2 LIMITAÇÕES DA PESQUISA

- Auto salvamento do progresso;
- Ampliar a abrangência de conceitos nos algoritmos aceitos;
- Acesso via login e senha com repositório dos códigos;
- Desenvolver linguagem que una e venha a substituir a UAL buscando similaridade com Java.
- Permitir flexibilidade para definir as variações de Portugol para que possa ser utilizado por usuários de outros ambientes.

5.3 SUGESTÕES PARA TRABALHOS FUTUROS

REFERÊNCIAS

AJAX.ORG. **Ace - The High Performance Code Editor for the Web**. 2016. <https://ace.c9.io/>. Acesso em: 24 agosto 2016.

CITAR, C. **Citar**. 0000.

DELAMARO, Márcio Eduardo. **Como Construir um Compilador Utilizando Ferramentas Java**. 1. ed. São Paulo: Novatec, 2004. v. 1. ISBN 8575220551.

DERSHEM, Herbert L; JIPPING, Michael J. **Programming languages: structures and models**. [S.l.]: Wadsworth Publ. Co., 1990.

DICIONARIOINFORMAL.COM.BR. **Significado de Bizu**. 2016. Acesso em: 27 setembro 2016. Disponível em: <http://www.dicionarioinformal.com.br/bizu/>.

ESMIN, Ahmed Ali Abdalla. Portugol/plus: uma ferramenta de apoio ao ensino de lógica de programação baseado no portugol. In: **IV Congresso RIBIE**. Brasília: Anais do IV Congresso RIBIE, 1998.

EVARISTO, Jaime; CRESPO, Sérgio. **Aprendendo a Programar - Programando numa Linguagem Algorítmica Executável (ILA)**. [S.l.]: Book Express, 2000. ISBN 8586846473.

FERNANDES, Antonio Luiz Bogado; BOTINI, Joana. **Construção de algoritmos**. Rio de Janeiro: SENAC, 1999. ISBN 8585746564.

FERRANDIN, Mauri; STEPHANI, Simone Lilian. Ferramenta para o ensino de programação via internet. In: **Congresso Sul Catarinense de Computação**. Criciúma: Anais do I Congresso Brasileiro de Computação, 2005. p. 102–110.

FRASER, Neil. **JS-Interpreter - A sandboxed JavaScript interpreter in JavaScript**. 2016. <https://github.com/NeilFraser/JS-Interpreter>. Acesso em: 29 agosto 2016.

FUEGI, J.; FRANCIS, J. Lovelace babbage and the creation of the 1843 'notes'. **IEEE Annals of the History of Computing**, v. 25, n. 4, p. 16–26, Oct 2003. ISSN 1058-6180.

GOOGLE. **Material Design Lite**. 2016. <https://getmdl.io/>. Acesso em: 19 julho 2016.

HAVERBEKE, Marijn. **Acorn - A small, fast, JavaScript-based JavaScript parser**. 2016. <https://github.com/ternjs/acorn>. Acesso em: 29 agosto 2016.

HAVERBEKE, Marijn. **CodeMirror - In-browser code editor**. 2016. <https://codemirror.net/>. Acesso em: 24 agosto 2016.

JESUS, E. A. de; SANTIAGO; DAZZI, R. L. S. R. de. Ferramenta para criação e teste de algoritmos utilizando fluxogramas ou portugol. In: **Congresso Brasileiro de Computação, 2004, Itajaí**. Itajaí: Congresso Brasileiro de Computação, 2004.

LOPES, Anita; GARCIA, Guto. **Introdução à programação: 500 algoritmos resolvidos**. Rio de Janeiro: Campus, 2002. ISBN 9788535210194.

MEDEIROS, Antônio Vinícius Menezes. Um interpretador online para a linguagem portugol. 2015.

MEDINA, Marco; FERTIG, Cristina. **Algoritmos e programação**. São Paulo: Novatec, 2006.

PRODANOV, Cleber Cristiano; FREITAS, Ernani Cesar de. **Metodologia do Trabalho Científico: Métodos e Técnicas da Pesquisa e do Trabalho Acadêmico**. 2. ed. Novo Hamburgo: Editora Feevale, 2013.

SALIBA, Walter Luiz Caram. **Técnicas de Programação: Uma Abordagem Estruturada**. São Paulo: Markon, 1992. ISBN 0074607316.

SANTIAGO, Rafael de; DAZZI, Rudimar Luís Scaranto. Ferramentas que auxiliam o desenvolvimento da lógica de programação. In: **XII SEMINCO - Seminário de Computação, 2003, Blumenau. Anais do XII SEMINCO**. Blumenau: Furb, 2003. p. 113–120.

SEBESTA, Robert W. **Conceitos de linguagens de programação**. Porto Alegre: Bookman, 2009. ISBN 8577808629.

SOUZA, Márcia Valéria Rocha de; FRANÇA, A. César C. Ferramentas de auxílio ao aprendizado de programação: Um estudo comparativo. In: **XIII ERBASE - WEIBASE**. Aracaju: Anais da XIII ERBASE - WEIBASE, 2013. p. 41–50.

TAVARES, Gilberto E. **BizUAL - Portugol interpreter in HTML5**. 2016. <https://camaleaun.github.io/bizual/>. Acesso em: 4 outubro 2016.