AnaSintAsc: uma aplicação móvel para auxiliar alunos no processo de aprendizagem sobre Análise Sintática Ascendente na disciplina Compiladores

Júlio Miguel de Souza Costa[1], Alexandre Paes dos Santos[1], Jefferson Farias Moraes da Silva[1]

[1]Universidade Federal de Alagoas (UFAL)
Caixa Postal 57.309 – 005 – Arapiraca – AL – Brasil

{juliomiguelsouzacosta, alexandre.paes.santos, jefferson.fariasm}@gmail.com

Abstract. *Resources that assist the learning process can significantly contribute to retaining content. Currently, there is a huge variety in terms of tools to support studies. Mobile applications are examples of these tools, since they are part of our daily lives in such an intense way that it became natural to have applications installed for the most diverse daily activities. This article, therefore, proposes a mobile application as a teaching resource to assist students in Computer Science courses in the Compilers discipline with regard to the Ascending Syntactic Analysis process - a specific topic of the discipline.*

Resumo. *Recursos que auxiliam o processo de aprendizagem podem contribuir de forma significativa na retenção do conteúdo. Atualmente, presencia-se uma variedade enorme no que diz respeito as ferramentas de apoio aos estudos. Os aplicativos móveis são exemplos dessas ferramentas, já que fazem parte do nosso dia a dia de maneira tão intensa que tornou-se natural ter aplicações instaladas para as mais diversas atividades do cotidiano. Este artigo, portanto, propõe uma aplicação móvel como recurso didático para auxiliar os alunos dos cursos de Ciência da Computação na disciplina Compiladores no que diz respeito ao processo de Análise Sintática Ascendente - um tópico específico da disciplina.*

1. Introdução

Grande parte das disciplinas que compõem as graduações em Ciência da Computação são bem abstratas devido à natureza do curso, o qual fundamenta seu estudo em uso de modelos matemáticos, lógicos etc. que são imprescindíveis para a área. Sendo assim, os graduandos são introduzidos dentro do contexto do curso de maneira gradual e de modo que as disciplinas estejam dispostas nas configurações adequadas, para que todos possam evoluir durante a graduação. Há, obviamente, variações das grades – de acordo com cada universidade – dos cursos de Ciência da Computação, mas no geral, prezando pela didática, procura-se manter uma hierarquia lógica dos conteúdos, para que os alunos possam ascender no processo de aprendizagem. Todavia, mesmo com essa disposição gradativa dos conteúdos, que mantém uma hierarquia de dificuldade e lógica, observa-se

que muitos alunos acabam abandonando o curso por enfrentarem dificuldades em disciplinas de apelo mais abstrato [Mcgettrick et al. 2005].

Quando o aluno tem contato com a disciplina Compiladores, espera-se que ele possua conhecimentos de disciplinas como: Programação, Estrutura de Dados e Teoria da Computação, já que o não domínio desses conhecimentos poderão comprometer seu desenvolvimento na disciplina, uma vez que tais conhecimentos são suas características basilares. Nessa continuidade, pontua-se que é preciso que o aluno já possua uma bagagem teórica e prática razoável ou até mesmo consolidada, pois fatores como estes revelam que Compiladores é uma disciplina que requer um empreendimento maior de esforços durante o aprendizado.

O estudo e projeto de Compiladores consiste em uso de modelos, técnicas e ferramentas que possibilitam a construção de linguagens de programação e respectivamente sua execução. Para que isso tenha êxito é necessário na construção de um compilador lançar mãos de conhecimentos de áreas como a Arquitetura e Organização de Computadores e a Engenharia de Software [Aho. et. Al. 1995].

Em linhas gerais, um compilador consiste basicamente em um programa que recebe como entrada uma linguagem fonte e a traduz para outra semelhante, a linguagem alvo. Desta forma, a máquina com uso de um sistema operacional e processador específico é capaz de entender o código. É papel importante também que o compilador reporte ao usuário erros que possam conter na linguagem de entrada - programa fonte. Entre as etapas de ler uma linguagem fonte e traduzi-la para uma linguagem alvo há outros passos necessários para que se possa ocorrer a execução da linguagem.

Para compreensão da Teoria dos Compiladores, em geral, aborda-se os seguintes tópicos - que são as fases do processo de compilação (Figura 1): Análise Léxica, Análise Sintática, Análise semântica, Geração de Código Intermediário, Otimização de Código e Geração de Código final.

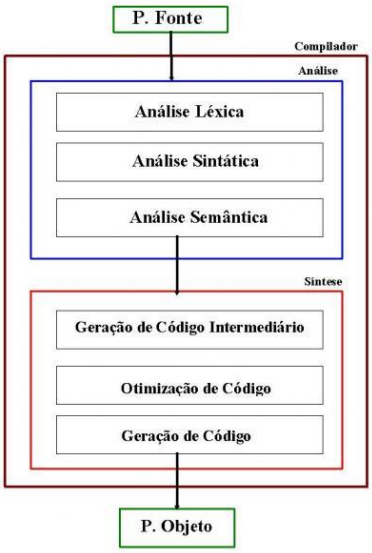

Figura 1. Fases da Compilação.

Quanto à Análise Sintática, um dos papeis que o Analisador Sintático desempenhará, consistirá em verificar estruturas de sentenças para confirmar se ela está correta ou se há presença de erros em sua formação, produzindo uma árvore sintática correspondente à linguagem que está sendo compilada. Sendo que a árvore é intrinsicamente importante para as fases posteriores da compilação. Podemos comparar a Análise Sintática com o processo de Análise Sintática de línguas naturais. Em outras palavras, não é suficiente conhecer o significado isolado de cada palavra, mas também é necessário entender o papel que ela desempenhará na sentença, para que a mensagem em uma frase venha ter sentido [Ricarte 2008]. Dito de uma maneira técnica, a Análise Sintática - também conhecida por Análise Gramatical - realiza o processo para determinar se uma certa cadeia de símbolos léxicos podem ser gerados por uma gramática [Alfred V. Aho 1995].

Duas abordagens para processo de Análise Sintática são utilizadas: Descendente (TopDown) e Ascendente (Buttom-up). Uma das diferenças entre as abordagens consiste na estratégia de analisar a árvore sintática – a descendente começa no nó raiz da árvore enquanto a ascendente começa pelos nós folhas. Além disso, enquanto a primeira preocupa-se com qual regra de produção usará para construir a cadeia (string); A segunda preocupa-se em reduzir cadeias de caracteres para chegar no símbolo inicial ($). Logo, a análise ascendente destaca-se pelo seu desempenho, porém é mais difícil de ser implementada.

1.1. A Aplicação Móvel como ferramenta de apoio à disciplina

Os Dispositivos Móveis trouxeram portabilidade e flexibilização na utilização de softwares que antes estavam restritos aos computadores pessoais. Com processadores cada vez menores e mais poderosos, atualmente estes dispositivos executam desde aplicações simples as mais complexas. Os aplicativos vieram e ocuparam muitos espaços, possibilitando a criação de novos paradigmas para realização de coisas do dia a dia.

Como processo natural, vários âmbitos sociais tiveram que adaptar-se e, até mesmo, reinventar-se para o uso das novas tecnologias. Com a educação esse processo não foi diferente, uma vez que diversas abordagens foram e estão sendo criadas para que esse campo tão necessário da sociedade possa usufruir o melhor das TICs (Tecnologias da Informação e Comunicação).

Como ressalta Rios et. al. (2011, p.8), as TICs contribuem de maneira positiva para que estudantes possam desenvolver suas habilidades e competências de maneira crítica e autônoma para atuação pertinente no mundo em constantes mudanças.

1.2. Justificativa

Apresentada em linhas gerais a disciplina Compiladores e algumas características das demais disciplinas que compõem os cursos de Computação, motivou-nos a desenvolver uma aplicação móvel para um tópico específico. Visto que, devido à escassez de materiais relacionados que apoiem o aluno durante seus estudos e visando também contribuir de maneira expressiva para uma formação consistente, na qual o graduando seja capaz de sair com uma formação que o permita aplicar seus conhecimentos adquiridos no curso para resolver demandas reais em sociedade. A aplicação, portanto, tem como objetivo minimizar as dificuldades diante de um tópico bastante abstrato da disciplina, contando com recursos que auxiliarão durante todas as etapas necessárias para compreensão do conteúdo.

1.3. Objetivo

O objetivo do presente trabalho consiste em mostrar o processo de desenvolvimento do Aplicativo Móvel AnaSintAsc, utilizando-se dos conceitos de processo de software para que ele possa ser disponibilizado, utilizado e melhorado com feedbacks adicionais dos alunos e professores.

2. Trabalhos correlatos

Nesta seção serão apresentados alguns softwares didáticos voltados para a disciplina compiladores que exploram algumas das etapas do processo de compilação. Verificou-se que não há aplicativos móveis semelhantes aos citados nesta seção e nem do presente trabalho.

JISON: a ferramenta escrita em JavaScript é um gerador online de analisador que recebe uma gramática LR para análise e apresenta uma descrição, representação das regras de produção e a tabela SRL. A gramática a ser submetida precisa estar escrita no formalismo de Backus-Naur que não é imediatamente intuitivo e não há uma representação visual organizada.

Grammophone: esta ferramenta realiza além da análise de gramáticas LR, transformações para outras modalidades de gramática. Semelhantemente ao JISON, ela apresenta a tabela SRL, autômato e exemplos de sentenças que podem ser reconhecidas. A representação da gramática para análise dá-se de maneira simples e a disposição das informações são razoáveis junto com a navegação. Nesta ferramenta não há como testar sentenças e ver o processo de reconhecimento da gramática de forma que auxilie o aluno.

3. Validação e metodologia

A escassez de materiais de apoio, carência de aplicações voltadas ao ensino de Análise Sintática - principalmente no detalhamento que dá-se nesta etapa - e a relativa complexidade do tema foram algumas das razões que motivaram a criação de uma aplicação móvel para um tópico específico da disciplina Compiladores, justificando seu desenvolvimento [Juliano Henrique Foleiss 2009]. Além disso, o contato do aluno com o aplicativo reduzirá de maneira significativa a curva de aprendizagem do conteúdo, pois a aplicação se encarregará de mostrar todo passo a passo da Análise Sintática Ascendente para qualquer gramática válida inserida, contando com recursos como: um Quiz para testar sua aprendizagem e uma seção de revisão teórica do conteúdo.

Considerando também contribuir, mesmo que de maneira pontual, com a redução dos índices de evasão nos Cursos de Ciência da Computação, uma vez que foi constatada numa matéria do jornal virtual Poder360 (2019), que se baseou em estudos do MEC (Ministério da Educação). Os cursos de Computação ocupam o Ranking de 6º lugar no quesito evasão. Contudo, é obvio que são vária as causas de evasão, mas como exposto neste trabalho, a natureza abstrata das disciplinas é um dos fatores que mais contribuem [Mcgettrick et al. 2005].

Seguindo o processo de software para o desenvolvimento da aplicação que compõe-se de atividades necessárias para se chegar ao produto software [Eduardo Bezerra 2006]. Verificou-se que a viabilidade do projeto é positiva, ou seja, é possível prosseguir com sua elaboração, pois não demanda tempo e recursos consideráveis, além de haver tecnologias que atendem os requisitos da aplicação. Além disso, o desenvolvimento do

aplicativo contribui com a solução de um problema constatado, consequentemente permitindo acrescentar à educação novas abordagens pedagógicas com uso da tecnologia.

4. Especificação

O modelo de processo de software adotado para o desenvolvimento da aplicação foi o Iterativo e Incremental, pois partes funcionais do aplicativo podem ir sendo entregues e respectivamente receber feedbacks por parte dos usuários para que se possa refinar as funcionalidades e acrescentar novas funções. A Figura 2 mostra através de um diagrama o esquema deste modelo de processo de software.

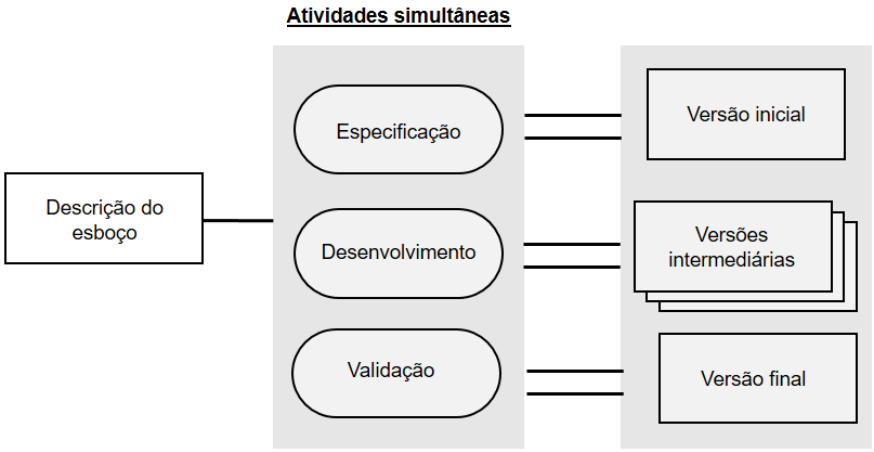

Figura 2. Modelo Iterativo e Incremental

A fim de capturar as funcionalidades do sistema foi esboçado um Diagrama de Casos de Uso para facilitar a modelagem dos requisitos funcionais da aplicação (Figura 3). As funções principais do sistema serão descritas logo mais junto com os protótipos de tela.

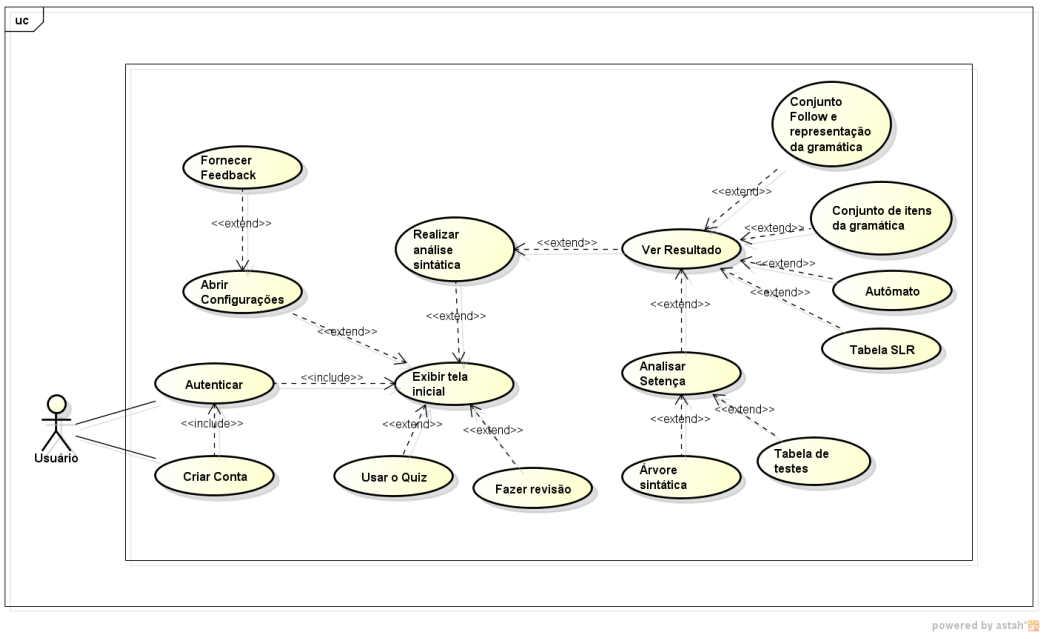

Figura 3. Diagrama de Casos de Uso

O aplicativo será desenvolvido com a utilização da linguagem JavaScript e com o framework front-end React Native. A adoção do React Native traz consigo as vantagens de escrever um único código que seja possível rodar tanto em dispositivos Android como iOS, permitindo também migrar o código para react-web, já que os códigos em JavaScript são altamente escaláveis. A Aplicação fará uso do Firebase como back-end que agilizará de maneira expressiva o desenvolvimento, pois esta tecnologia integra diversos serviços como: banco de dados NoSQL em tempo real na nuvem, notificações push, Analytics, que possibilita obter um relatório em tempo real sobre o uso do aplicativo etc.

4. Estado atual do desenvolvimento

As ferramentas necessárias para o desenvolvimento da aplicação já estão prontas, além de que também os protótipos das telas já foram criados. A seguir é apresentado os protótipos de tela da aplicação e suas funcionalidades.

A figura 4 mostra quatro telas dispostas em sequência. A primeira tela concentra as funcionalidades principais da aplicação, resumindo é a tela inicial da App. As demais telas que seguem, relacionam-se à Análise Sintática LR. Na segunda tela percebe-se que há um campo para inserir uma gramática para a análise. Quando a gramática for válida e for submetida, automaticamente abre-se a tela de resultados onde há seções detalhadas sobre todo o processo que ocorreu na Análise Sintática LR. Estas seções podem ser expandidas para mostrarem os resultados e contraídas para ocultá-los. Na terceira tela há outro campo logo abaixo onde o usuário pode submeter sentenças a serem reconhecidas e ver o resultado na quarta tela através da tabela de testes, na qual ocorre todo o passo a passo de reconhecimento da sentença. Por fim, ainda na quarta tela, apresenta-se uma das ferramentas mais importantes deste processo: a árvore sintática.

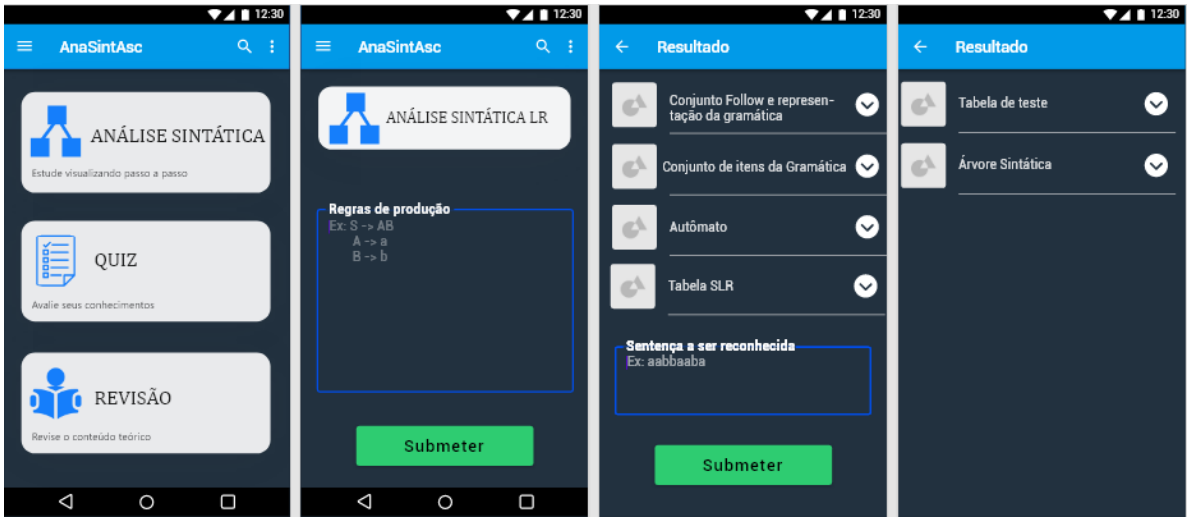

Figura 4. Protótipo de telas

Como funcionalidades adicionais da aplicação a figura 5 exibe respectivamente a tela de Quiz e a de Revisão dos conteúdos. A tela de Quiz contará com diversas perguntas relacionadas aos conteúdos. Logo, quando o usuário responde uma questão, a correção é retornada imediatamente, permitindo assim validar seus conhecimentos. A tela de revisão consta como um recurso que possibilita rápidas consultas no conteúdo.

Figura 5. Protótipo das telas de Quiz e Revisão

5. Etapas posteriores

Após o desenvolvimento do produto mínimo viável – uma primeira versão da aplicação. Serão realizados estudos de como dispor melhor as informações na tela. Portanto, o feedback de alunos e professores será de suma importância, para que a ferramenta possa ser entregue de maneira totalmente funcional.

E por fim, após atingir uma versão totalmente operante, a ferramenta ficará disponível nas lojas de aplicativos e hospedada no site GitHub, para que alunos e professores possam colaborar com sua evolução e acrescentar novos recursos.

6. Bibliografia

AHO, A. V.; SETHI, R.; ULLMAN, J. D. **Compiladores: princípios, técnicas e ferramentas**. Rio de Janeiro: LTC. 1995.

BEZERRA, Eduardo. **Princípios de Análise e Projetos de Sistemas com UML**. Rio de Janeiro: Editora Elsevier. 2006. 19 p.

RIOS, Clitien Alice Meira. DOS SANTOS, Dulce Pereira. **Mídias na educação: formação continuada do professor, privilégio para o aluno**. Unimontes. Montes Claros, 2011.

RICARTE, Ivan. (2008). **Introdução à compilação**. Rio de Janeiro: Elsevier, 2008. 97 p.

7. Referências

CRISTIANE. **Fases da Compilação**. 2009. Disponível em: <https://lifo-shadow.blogspot.com/2009/04/fases-da-compilacao.html>. Acesso em: 28 fev. 2020.

FOLEISS, J. H., Guilherme Puglia Assunção, CRUZ, E. H. M., GONÇALVES, R.A.L. e FELTRIN, V.D (2009). Scc: **Um compilador C como ferramenta de ensino de**

compiladores. WEAC2009 - Workshop Educação em Arquitetura de Computadores, pages 15–22.

JFLEX. **JFlex 2020**. Disponível em: <https://www.jflex.de/ >. Acesso em: 28 fev. 2020

MCGETTRICK, A., BOYLE, R., IBBETT, R., LLOYD, J., LOVEGROVE, G. e MANDER, K. (2005). **Grand challenges in computing: Education - a summary**. The Computer Journal, 48(1):42–48.

SCHNEIDER, C.S., PASSERINO, L.M. e OLIVEIRA, R. F. (2005). **Compilador educativo verto: ambiente para aprendizagem de compiladores**. RENOTE.