UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CÂMPUS FRANCISCO BELTRÃO
LICENCIATURA EM INFORMÁTICA

ROSAINE FIORIO

UM SOFTWARE DE APOIO À APRENDIZAGEM DE GRAMÁTICA E ESTILO LITERÁRIO DA LÍNGUA PORTUGUESA BRASILEIRA

TRABALHO DE CONCLUSÃO DE CURSO

FRANCISCO BELTRÃO
2015

ROSAINE FIORIO

UM SOFTWARE DE APOIO À APRENDIZAGEM DE GRAMÁTICA E ESTILO LITERÁRIO DA LÍNGUA PORTUGUESA BRASILEIRA

Trabalho de Conclusão de Curso, apresentado ao Curso de Licenciatura em Informática, da Universidade Tecnológica Federal do Paraná, campus Francisco Beltrão, como requisito parcial para obtenção do título de Licenciado.

Orientador: Profª. Drª. Maici Duarte Leite

FRANCISCO BELTRÃO
2015

ROSAINE FIORIO

UM SOFTWARE DE APOIO À APRENDIZAGEM DE GRAMÁTICA E ESTILO LITERÁRIO DA LÍNGUA PORTUGUESA BRASILEIRA

Trabalho de Conclusão de Curso, apresentado a Universidade Tecnológica Federal – Campus Francisco Beltrão, como parte das exigências para a obtenção do título de Licenciado em Informática.

UTFPR-FB, 01 de dezembro de 2015

BANCA EXAMINADORA

_____

Profª. Maici Duarte Leite (Orientador)
Doutora em Informática

_____

Prof. Ademir Roberto Freddo (Convidado)
Doutor em Informática

_____

Prof. Francisco A. F. Reinaldo (Presidente da Banca)
Doutor em Engenharia Electrotécnica e de Computadores

"A Folha de Aprovação assinada encontra-se na Coordenação do Curso (ou Programa)".

DEDICATÓRIA

Para minha família: meu marido Jhonnatan e meu filho Rafael.

AGRADECIMENTOS

Se você está lendo esta página é porque eu consegui e lhe confesso que não foi fácil chegar até aqui, pois foi um longo caminho percorrido. Nada foi fácil nem tampouco tranquilo e como diz um provérbio africano “a sola do pé conhece toda sujeira da estrada”. Por isso deixo aqui registrado aqueles que fizeram parte dessa caminhada:

Primeiramente a Deus e a Nossa Senhora Aparecida que me guiaram pelos caminhos da vida, me sustentaram quando me desequilibrei ou mesmo me carregaram quando já havia desistido.

A minha “grande família”, aos meus pais, Osmar e Alzira que me ensinaram a ser nobre, persistente, valente, uma verdadeira guerreira diante dos empecilhos da vida. Aos meus irmãos, Jhonatan e Thomas, que me incentivaram em muitos momentos. Ao meu marido, Jhonnatan, que muitas vezes renunciou de seus sonhos para que eu pudesse realizar o meu, por ter me incentivado, me apoiado e por estar sempre ao meu lado na alegria e na tristeza, na saúde e na doença, e na riqueza e na pobreza. Ao meu filho, Rafael, que ainda não sabe ler, mas que desde que estava no meu ventre frequentou aulas, fez provas, trabalhos, e depois que esteve nesse mundo suportou minhas ausências como mãe e mesmo assim entendeu que tudo isso foi necessário.

Aos professores da COLIN que fizeram parte de todo caminho percorrido na aventura do conhecimento.

Ao Prof. Me. Paulo Junior Varela, meu mentor, exemplo profissional e amigo, por ter me guiado em assuntos até então obscuros, por ter me propiciado muitas oportunidades e pela confiança. Quando ‘crescer’ eu quero ser como você.

A Profª. Drª. Maici Duarte Leite por ter aceitado ser minha orientadora, por ter me ajudado a percorrer o caminho do conhecimento, pela dedicação, pela amizade e confiança em meu trabalho.

Ao Sr. Willian Daniel Colen de Moura por ter gentilmente me auxiliado durante as pesquisas sobre o Cogroo e me enviado sua dissertação além de outras informações.

Aos meus colegas e amigos, Adair, Emanuelle e Rauany pela paciência e companheirismo.

Enfim, com vocês divido essa experiência e alegria.

RESUMO

FIORIO, Rosaine. UM SOFTWARE DE APOIO À APRENDIZAGEM DE GRAMÁTICA E ESTILO LITERÁRIO DA LÍNGUA PORTUGUESA BRASILEIRA. 85f. Trabalho de Conclusão de Curso (Licenciatura em Informática) - Universidade Tecnológica Federal do Paraná. Francisco Beltrão, 2015.

O desenvolvimento e aplicação de novas tecnologias tem permitido a implantação de serviços computacionais que possibilitam a substituição de determinados procedimentos ora antes manuais e onerosos, por um processo automatizado, intuitivo e didático. Esta inserção da automatização dos ambientes possibilita a aceleração do processo de aprendizado, e tende a reduzir ocorrência de erros na execução das atividades. Em consonância a área educacional, é uma das áreas que carece de softwares e ferramentas que possibilitem uma maior agilidade em seu processo, principalmente pelo fator lúdico que o meio computacional produz. Uma das áreas da Língua Portuguesa Brasileira que necessita de atenção é o ensino da gramática, que perfaz um dos mais importantes estudos da língua. Umas das formas mais utilizadas para a aprendizagem da Gramática é a leitura e a produção de textos. Sendo assim, este processo também envolve a Linguística e a Literatura. Isto abre uma lacuna para o desenvolvimento de um software que consiga fazer a extração e a manipulação de informações sintáticas gramaticais a partir de amostras de textos que se deseja estudar, bem como, identificar o estilo literário através das observações das estruturas textuais. A forma do uso da língua dentro das aulas de Língua Portuguesa é caracterizado como artificial, o que acaba se repetindo nas três unidades de ensino da Língua Portuguesa quais sejam: a leitura, produção textual e a análise linguística. Com isso, se torna difícil fazer com que o aluno adquira domínio da língua em suas diferentes instâncias – oral e escrita- pois continua-se utilizando em sala de aula a língua como algo estático e artificial que não depende da sociedade em que está inserida e suas evoluções e os falantes que dela fazem uso. Assim, pode-se justificar o desenvolvimento de um software que vise auxiliar o professor a desenvolver um ensino focado nesses aspectos, ou seja, um software que possa ser aplicado como apoio ao ensino da língua portuguesa e aos estilos literários. Um software que possibilite a identificação da função que cada palavra exerce dentro de um texto, pode ser aplicado como uma ferramenta de aprendizado adicional, cuja aplicação pode dar-se tanto em sala de aula como em atividades extraclasse, como complemento de estudos. Para tanto, o software oferece três níveis de extração de características sintáticas da língua portuguesa. O primeiro nível demonstra a função das palavras dentro de um texto, tais como: sujeito e predicado. Já o segundo nível de gramática dos sintagmas, demonstram a relação entre as palavras em uma frase, tais como: sintagmas nominais e verbais. E, por fim, o terceiro nível possibilita a extração das classes morfológicas das palavras, tais como: verbo, adjetivo e advérbio. Além disso, com a disponibilização do corpus de textos que abrange todos os estilos literários brasileiros o professor pode possibilitar a formação do aluno de forma que esse possa ter contato com a parte escrita da língua e sua análise sintática e, também, da produção de textos e sua respectiva análise.

Palavras-chave: Java. Ensino. Língua Portuguesa. Estilos Literários. Sintática.

ABSTRACT

FIORIO, Rosaine. UM SOFTWARE DE APOIO À APRENDIZAGEM DE GRAMÁTICA E ESTILO LITERÁRIO DA LÍNGUA PORTUGUESA BRASILEIRA. 85f. Trabalho de Conclusão de Curso (Licenciatura em Informática) - Universidade Tecnológica Federal do Paraná. Francisco Beltrão, 2015.

He development and application of new technologies has allowed the deployment of computing services that enable the replacement of certain procedures before now manuals and costly, an automated, intuitive and didactic process. This insertion automation environment enables the acceleration of the learning process, and tends to reduce the occurrence of errors in the execution of activities. Consistent educational area is one area that lacks software and tools that enable greater flexibility in the process, especially the playful factor that produces the computational means. One of the areas of the Brazilian Portuguese language that needs attention is the teaching of grammar, which makes one of the most important studies of the language. One of the most used forms for learning grammar is read and the production of texts. Thus, this process also involves the language and literature. This opens a gap to develop software that can do the extraction and manipulation of grammatical syntactic information from samples of texts under study, as well as identify the literary style through the observations of the textual structures. The form of language use within the Portuguese classes is characterized as artificial what ends up repeating the three teaching units of the Portuguese language such as: reading, text production and linguistic analysis. Thus, it becomes difficult to make students acquire language proficiency in their different instances - speak and write- it continues up using classroom language as something static and artificial that does not depend on the society in which it operates and its evolutions and speakers that employ them. Thus, one can justify the development of a software that aims to help teachers to develop a teaching focused on these aspects, that is, software that can be applied to support the teaching of the Portuguese language and literary styles. A software that enables the function of identification that each word carries within a text, can be applied as an additional learning tool, the application can take place both in the classroom and in extracurricular activities, as a complement to studies. For this, the software offers three levels of extraction of syntactic features of the Portuguese language. The first level shows the function of words in a text, such as subject and predicate. The second level of grammar of phrases, show the relationship between words in a sentence such as: nominal and verbal phrases. And finally, the third level enables the extraction of morphological classes of words, such as verb, adjective and adverb. Moreover, with the availability of the corpus of texts covering all Brazilian literary styles the teacher can enable the formation of the student in order that this may have contact with the writing of the language and its syntax analysis and also the production of texts and their respective analysis.

Keywords: Java. Education. Portuguese language. Literary styles. Syntactic.

LISTA DE SIGLAS

API	*Application Programming Interface*
CPLP	Comunidade dos Países de Língua Portuguesa
IDE	*Integrated Development Environment*
J2SE	*Java Standard Edition*
JPA	*Java Persistence API*
MVC	*Model-View-Controller*
OO	Orientação a Objetos
ORM	*Object Relational Mapping*
PLN	Processamento de Linguagem Natural
POO	Programação Orientada a Objetos
QBE	*Query By Example*
SGBDs	Sistema Gerenciadores de Banco de Dados
SQL	*Structured Query Language*
SVM	*Support Vector Machine*
UML	*Unified Modeling Language*
XML	*Extensible Markup Language*

LISTA DE FIGURAS

LISTA DE QUADROS

LISTAGENS DE CÓDIGO

SUMÁRIO

1. INTRODUÇÃO

Este capítulo contém as considerações iniciais, apresentando o assunto de pesquisa e o trabalho realizado, os objetivos e a sua justificativa. O capítulo é finalizado com a organização do texto por meio dos seus capítulos.

1.1 CONSIDERAÇÕES INICIAIS

O desenvolvimento e aplicação de novas tecnologias tem permitido a implantação de serviços computacionais, que possibilitam a substituição de determinados procedimentos ora antes manuais e onerosos, por um processo automatizado, intuitivo e didático. Esta inserção da automatização dos ambientes possibilita a aceleração do processo de aprendizado, e tende a reduzir ocorrência de erros na execução das atividades. Em consonância a área educacional, é uma das áreas que carece de softwares e ferramentas, que possibilitem uma maior agilidade em seu processo, principalmente pelo fator lúdico que o meio computacional produz.

Hoje no Brasil, ao decorrer do ensino básico, fundamental e médio são no mínimo 12 anos de aprendizagem da Língua Portuguesa. Entretanto, ainda se percebe a dificuldade que os alunos e professores enfrentam quanto a língua escrita e falada. Percebe-se que através do método tradicional de ensino os alunos aprendem, porém não é o ideal para redigir e falar de forma culta. No entanto, formas lúdicas de aprendizagem de língua portuguesa também podem ser aplicadas. Neste caso, o uso de meios computacionais para produzir softwares que auxiliem alunos e professores no aprendizado da língua portuguesa é de grande valia.

Uma das áreas da Língua Portuguesa Brasileira que carece de atenção é o ensino da Gramática, que perfaz um dos mais importantes estudos da língua. Umas das formas mais utilizadas para a aprendizagem da Gramática é a leitura e a produção de textos. Sendo assim, este processo também envolve a linguística e a literatura. Por conseguinte, a linguística é dada através da investigação da linguagem por meio de observações empíricas e controladas, que possam ser verificadas com base na estrutura geral da linguagem, ou seja, da gramática (LYONS, 1968). Já o meio literário é passível de aplicação pelo fator estético da linguagem escrita, que denota o estilo de escrita de cada autor ou de cada período literário. Diante deste fato, surge a linguística computacional que por sua vez, é uma área multidisciplinar que tem por finalidade envolver processos computacionais para manipular a linguagem humana. Envolve em suas estruturas os conceitos da computação, da estatística e da linguística. No entanto, ainda são ínfimas as soluções de software que possam auxiliar linguistas,

pesquisadores, e estudantes em geral no que concerne a extração e análise das características em língua portuguesa. Isto abre uma lacuna para o desenvolvimento de um software que consiga fazer a extração e a manipulação de informações sintáticas gramaticais a partir de amostras de textos que se deseja estudar, bem como, identificar o estilo literário através das observações das estruturas textuais.

1.2 OBJETIVOS

O objetivo geral apresenta o resultado principal do trabalho realizado e os objetivos específicos o complementam, no sentido de valores agregados.

1.2.1 Objetivo Geral

Evidenciar a importância das estruturas gramaticais e da literatura no aprendizado da língua portuguesa brasileira através dos níveis morfológico, sintagma e sintático.

1.2.2 Objetivos Específicos

- Identificar, e demonstrar as formas, os mecanismos e o controle de extração de características textuais da Língua Portuguesa Brasileira que podem auxiliar linguistas, pesquisadores, alunos e professores em casos que envolvam o ensino da gramática da língua portuguesa e estilo literário;
- Criar e disponibilizar um *corpus* de textos literários, contendo amostras de textos de autores da Literatura Brasileira que compreenda todos os períodos literários praticados no Brasil;
- Desenvolver um software de apoio a aprendizagem da estrutura gramatical e estilo literário aplicada à língua portuguesa brasileira;
- Reutilizar ferramentas e bibliotecas de análise da Língua Portuguesa já existentes e testadas;

1.3 JUSTIFICATIVA

A forma do uso da língua dentro das aulas de Língua Portuguesa é caracterizado como artificial o que acaba se repetindo nas três unidades de ensino da Língua Portuguesa quais sejam: a leitura que é feita de forma superficial pelos alunos, pois são orientados somente a realizar exercícios de interpretação de texto; quanto a produção textual a artificialidade também

está presente, visto que, além do professor somente corrigir o texto e deixar de lado a leitura em si, o aluno não considera os possíveis leitores do texto e acaba por não se esforçar na criação de estratégias discursivas o que acarreta na formulação de um texto que fala "exatamente o que a escola e o professor querem ouvir" (GERALDI, 2006); e quanto a análise linguística o aluno somente reproduz as análises existentes que lhe foram apresentadas pelo professor o que acaba se refletindo na escrita, pois os alunos não conseguem raciocinar gramaticalmente quando estão escrevendo e até mesmo lendo(GERALDI, 2006).

Com isso, se torna difícil fazer com que o aluno adquira domínio da língua em suas diferentes instâncias (oral e escrita), pois continua-se utilizando em sala de aula a língua como algo estático e artificial que não depende da sociedade em que está inserida e suas evoluções e os falantes que dela fazem uso. Para isso Geraldi (2006) propõe que o ensino da língua seja focado em três pontos: leitura de textos de diferentes gêneros, a produção textual tanto na forma escrita como falada e a análise linguística tanto nos textos lidos como nos produzidos. Com isso o objetivo maior e o de fazer com que o aluno reflita sobre os diferentes aspectos da linguagem auxiliando no desenvolvimento da sua comunicação de forma que ele se torne um sujeito crítico tanto da sociedade quanto de sua língua materna.

Assim, pode-se justificar o desenvolvimento de um software, que vise auxiliar o professor a desenvolver um ensino focado nos aspectos citados por Geraldi (2006), ou seja, um software que possa ser aplicado como apoio ao ensino da língua portuguesa e estilos literários.

Um software que possibilite a identificação da função que cada palavra exerce dentro de um texto, pode ser aplicado como uma ferramenta de aprendizado adicional, cuja aplicação pode dar-se tanto em sala de aula como em atividades extraclasse, como complemento de estudos. Para tanto, o software oferece três níveis de extração de características sintáticas da língua portuguesa. O primeiro nível demonstra a função das palavras dentro de um texto, tais como: sujeito e predicado. Já o segundo nível de gramática dos sintagmas, demonstram a relação entre as palavras em uma frase, tais como: sintagmas nominais e verbais. E, por fim, o terceiro nível possibilita a extração das classes morfológicas das palavras, tais como: verbo, adjetivo e advérbio. Além disso, com a disponibilização do corpus de textos que abrange todos os estilos literários brasileiros o professor pode possibilitar a formação do aluno de forma que esse possa ter contato com a parte escrita da língua e sua análise sintática e, também, da produção de textos e sua respectiva análise.

Então, um software que auxilie uma área que carece de soluções e ferramentas, tal como o ensino de Língua Portuguesa e Literatura, preenche uma lacuna ainda em aberta na língua portuguesa, além da possibilidade de aplicação deste nos estudos que envolvam outros

tipos de análise sintática pelos estudiosos da linguística. Sendo assim, o desenvolvimento do software pode auxiliar a comunidade cientifica e acadêmica para o desenvolvimento de suas pesquisas e atividades.

1.4 ESTRUTURA DO TRABALHO

Este texto está organizado em capítulos. O Capítulo 2 apresenta a fundamentação teórica que está centrada na Língua Portuguesa e Literatura, seu ensino e a utilização de softwares como forma de apoio. No Capítulo 3 estão os materiais e o método utilizado. No Capítulo 4 é apresentado o software desenvolvido. No Capítulo 5 exibe a conclusão, e, por fim, as referências utilizadas para fundamentar o trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados os conteúdos necessários para o bom entendimento deste trabalho. São detalhados assuntos tais como: a língua portuguesa e suas características, bem como definições de linguagem, linguística, o ensino de língua portuguesa no Brasil e o uso de softwares como apoio ao aprendizado da Língua Portuguesa.

2.1 A LÍNGUA PORTUGUESA

Entre os países que falam português, o Brasil é o que possui o maior número de nativos que falam o idioma, sendo que atualmente sua população ultrapassa 200 milhões de habitantes. O número de nativos da língua portuguesa chega a cerca de 241 milhões, tornando assim o português a quinta língua mais falada no mundo e a terceira entre as línguas ocidentais, somente atrás do inglês e do espanhol (COSTA, 2013).

Os oito países que falam português formaram a comunidade dos Países de Língua Portuguesa - CPLP - que tem por finalidade estreitar os laços entre esses países e diminuir as diferenças. Pensando nisso que em 2009 foi promovido um acordo ortográfico realizando algumas alterações nas grafias para aproximar as línguas e diminuir as diferenças e variedades da língua portuguesa.

2.1.1 Variedade Linguística da Língua Portuguesa no Brasil

No início da colonização portuguesa no Brasil, a língua tupi e o idioma português viviam lado a lado. No entanto, os padres jesuítas, que eram responsáveis por difundir o cristianismo, também difundiram o idioma português. Mesmo assim, algumas palavras do tupi foram incorporadas ao idioma, e com a chegada dos escravos algumas expressões utilizadas por eles também foram adotadas. Após a independência e com a grande imigração dos países afetados pela 2º guerra mundial o português agregou mais algumas expressões.

Todo esse processo explica as variedades linguísticas e as mudanças lexicais que há entre as várias regiões do Brasil que podem estar associadas aos seguintes fatores, de acordo com Pavalec (2007):

- **Geográficos**: faz parte desse processo as diferentes correntes imigratórias que ocorreram na região dando origem as variações que constituem os dialetos ou

sotaques, por exemplo. Os maiores acentos são: caipira, cearense, baiano, fluminense, gaúcho, mineiro, nordestino, paulistano, sertanejo e sulista.

- **Sociais**: a possibilidade de o indivíduo frequentar a escola ou não, traz diferenças a língua falada o que, muitas vezes, gera um meio de dominação e segregação social.

- **Profissionais**: é a linguagem técnica utilizada pelas profissões que exigem essa forma de comunicação.

- **Situacionais**: a adaptação do ser humano o leva a desenvolver linguagens diferentes para as diferentes situações e aqui destaca-se o fenômeno tecnológico, e com ele as redes sociais que levaram a formação de novas palavras ou mesmo abreviações de palavras já existentes.

- **Literária**: surge quando o autor acrescenta língua preocupações estéticas, criando assim elementos linguísticos. Um autor que se ateve e usou desses artifícios foi Guimarães Rosa, conhecido entre os literários como Mago da Linguagem.

Esses fatores possuem peculiaridades que influenciam nas modalidades escritas e faladas da língua e a linguística auxilia no entendimento destas peculiaridades, contudo para entender-se a função da linguística no contexto do ensino da língua portuguesa brasileira necessita-se entender a importância da linguagem e suas combinações.

2.2 LINGUAGEM

A linguagem humana é a combinação de sons significativos para produzir a linguagem natural. Cada pessoa possui preferencias e peculiaridades ao se expressar pelas diversas formas de linguagem, onde temos como principais: a escrita e a fala; e são essas diferenças que compõem o estilo de cada autor.

De acordo com Varela (2010), conforme demonstrado na Figura 1, os linguistas dividem o estudo da linguagem em diversas áreas como:

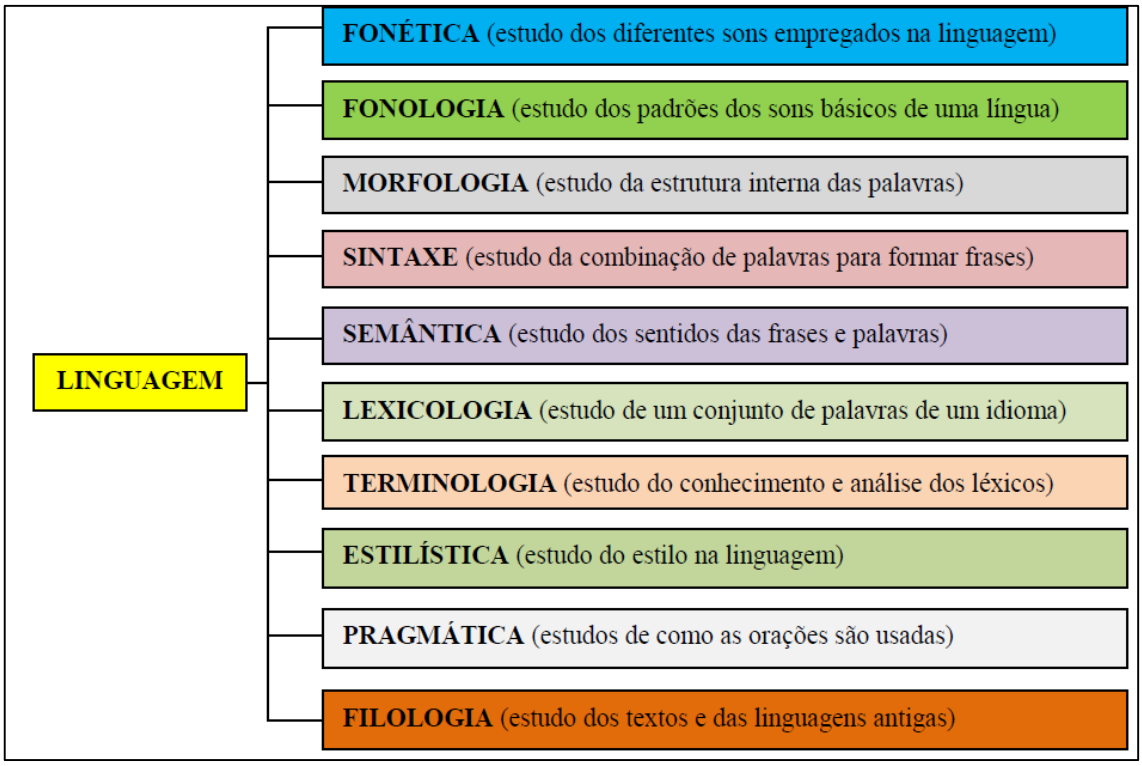


Figura 1 - Áreas de estudo da linguagem
Fonte: Varela (2010)

Dentre as principais classificações da linguagem, se encontra a classe sintática, que tem por função estudar a combinação das palavras na formação de frases e orações. Sendo assim, este trabalho se aprofunda na classe de linguagem sintática. Para tanto, também se faz necessário compreender outros conceitos que são apresentados nas próximas seções.

2.2.1 Linguística

Historicamente, a linguística do ocidente tem origem na filosofia grega que foi a primeira a estabelecer estudos sobre a linguagem, no entanto os pensadores gregos interessavam-se sobre as condições da linguagem, ficando, então, as categorias como nome, verbo e gênero gramatical para a lógica e a filosofia. No entanto, o funcionamento da língua, isto é, da análise da língua e suas particularidades não era essencial para os estudiosos da época (FUZER, 2004).

Com isso, somente em 1660, a língua latina foi analisada, pois Claude Lancelot com a colaboração de Antoine Arnauld escreveram um livro designado “Gramática de PortRoyal” que tinha por objetivo relatar os princípios comuns a todas as línguas (DUCROT & TODOROV, 1982, p. 19 apud FUZER, 2004). Para tanto, o que observa-se é que para os autores as línguas

tem princípios que são compartilhados e são responsáveis pela estruturação do pensamento, por exemplo, numerais que quantificam algo, artigos que se referem a algo ou alguém, substantivos que são o nome de algo entre outras. Então, sobre esse ponto de vista as demais línguas passaram a ser observadas e analisadas de forma a deduzir regras gerais (FUZER, 2004).

Sintetizando, linguística refere-se ao estudo da linguagem humana em seus aspectos práticos, onde utiliza-se dos conhecimentos adquiridos para melhorar a comunicação através da língua; e teóricos, porque estuda as características presentes em uma determinada língua (CRYSTAL, 2000).

A análise de características da língua portuguesa brasileira é complexa e quando fala-se no ensino dessas características observa-se dificuldades no processo aprendizado e por isso um sistema de apoio ao ensino seria interessante. No entanto, é preciso discutir conceitos associados a linguística e ao computador.

2.2.2 Linguística computacional

A linguística computacional começou a ser desenvolvida em 1950, mas o impulso dessa área veio através do desenvolvimento da Inteligência Artificial e de programas de tradução automática (OTHERO, 2006). Observa-se, então, que com os avanços da informática aplicados ao estudo da língua tornou possível encontrar abordagens diferenciadas para problemas descritivos e práticos da língua que anteriormente não poderiam ser resolvidos, sendo que uma dessa abordagens constitui-se na linguística de corpus que utiliza-se de computadores para armazenar e acessar textos escritos ou falados que sendo legíveis por máquina podem ser analisados e gerar informações a respeito das particularidades e construções de determinada língua (OTHERO, MENUZZI, 2009).

Outro foco da linguística computacional é o desenvolvimento de programas capazes de interpretar ou gerar informação a partir da linguagem e são chamados de processadores de linguagem natural. Dentro desse contexto, alguns softwares propõem-se a reconhecer a categoria que as palavras pertencem constituem-se em etiquetadores de categorias gramaticais, já sistemas que realizam a análise estrutural e constituição das frases são denominados analisadores sintáticos. O primeiro coloca uma etiqueta na palavra identificando a sua classe gramatical, já o segundo realiza o processamento da linguagem natural propriamente falando, visto que analisam os constituintes da frase (OTHERO, MENUZZI, 2008, OTHERO, 2006).

Para tanto, para se construir um software de apoio ao processo de ensino aprendizado é necessário entender o processamento de linguagem natural e suas implicações para a análise sintática da língua portuguesa brasileira de forma informatizada.

2.3 PROCESSAMENTO DE LINGUAGEM NATURAL E AS CAR

As aplicações que falam sobre a Linguagem Natural constituem-se em aplicações baseadas em textos que são sistemas que buscam por documentos específicos em uma determinada base de ados, tradutores e sistemas que resumem textos ou aplicações baseadas em diálogos que são sistemas que se referem a linguagem natural para bancos de dados, sistemas tutores e interpretadores de comandos em língua escrita ou falada (OLIVEIRA, 2002). No entanto, conforme Silva e Lima (2007), as primeiras pesquisas que falavam sobre a informatização do tratamento da língua natural datam do início da década de 50.

Com os avanços das tecnologias da informação e comunicação e a crescente necessidade de aplicações da linguística computacional ficou claro a necessidade de implementação de sistemas de processamento de linguagem natural que possam trabalhar com diferentes recursos linguísticos e lexicais, no entanto, em particular na língua portuguesa, a grande variação morfológica e sintática e a ambiguidade caracterizam-se como obstáculos no processamento de linguagem natural. Contudo, a análise linguística possui três níveis – morfológico, sintático e semântico- que podem auxiliar na solução desse obstáculo, pois a partir da correta análise e descrição do léxico o processamento será satisfatório (SMARSARO, 2004). Conforme descrito nos objetivos, vamos nos concentrar na análise sintática que é a utilizada na construção do sistema.

2.4 ANÁLISE SINTÁTICA

2.4.1 Análise sintática computacional

Na análise sintática computacional o Processamento de Linguagem Natural (PLN) faz o tratamento de sons, palavras, sentenças e discursos, ou seja, dos aspectos relacionados a comunicação oral ou escrita. Além disso, o PLN propõem-se a tornar possível que um computador comunique-se em linguagem humana dentro de alguns dos níveis da análise do

discurso, quais sejam: fonético e fonológico; morfológico, sintático, semântico e pragmático (ALLEN, 1995).

Com relação a gramática computacional, essa é constituída por regras e sentenças que compõem uma língua e essa regras de acordo com Bouillon (1998) tem dupla função: a função normativa que define regras para a combinação de palavras com o objetivo de gerar sentenças corretas e a função representativa que faz a associação entre as frases e a representação sintática. Para tanto, a gramática deve ter um forma que a represente computacionalmente e uma dessas formas é a de "*phrase-structure gramar*" que é definida a partir de uma quadrupla <T, N, P, S> onde o T representa as palavras, o N as categorias funcionais e lexicais, o P as regras de produção e S é um símbolo inicial que pertence a N (NUNES, 1999 apud GONZALEZ E LIMA, 2003).

Dentro da literatura há diferentes formas de representar uma gramática, sendo que não há uma distinção entre as melhores e as piores. Com isso, em relação a PLN é imprescindível que critérios formais sejam utilizados para construir as regras gramaticais, pois estes serão aliados ao léxico (VIEIRA, LIMA, 2001) que se constitui como o universo das relações existentes entre as palavras, a gramática e os seus significados (SCAPINI, 1995).

Ainda, o objetivo dos léxicos é ter um conjunto estruturado de informações sobre as palavras de maneira a prover informações sobre os seus sentidos, a linguagem e o mundo em que aquela linguagem está inserida, além disso, os léxicos que constituem a processamento de linguagem natural deve ser legíveis e tratáveis por máquina de forma que se possam fazer diferentes processamentos com o conteúdo (GUTHRIE, 1996). Os processamentos possíveis são o morfossintático, que trabalha com a construção das palavras e os seus grupos de elementos; a morfológica, que lida com a construção e classificação das palavras; e a sintática que analisa a constituição da frase e suas relações.

A análise sintática faz a análise de como as regras gramaticais são combinadas de forma a gerar uma árvore, que represente a estrutura sintática que constitui a sentença avaliada, além disso, esse estilo de análise tem lugar de destaque sendo considerada uma "interpretação da sintaxe" e, também, posição secundária tendo em vista os estudiosos "semântico gerativistas", pois consideram a sintaxe uma "projeção da semântica" (NUNES, 1999 apud GONZALEZ E LIMA, 2003). Neste estudo, optou-se pela análise sintática, pois em relação as lacunas existentes para pesquisa identificou-se, também, que para a análise forense de documentos havia uma lacuna que permitia a construção de um software capaz de extrair e quantificar as

características sintáticas de um texto e montasse o vetor necessário para as demais análises. Já o ponto de vista educacional, objetivo desse trabalho, tratar-se-á na secção seguinte.

2.5 O ENSINO DA LINGUA PORTUGUESA NO BRASIL

No Brasil Colonial não havia uma língua padrão, pois havia convivência de três línguas: o português trazido pelo colonizador, as línguas indígenas que se uniram para formar a chamada língua geral e o latim que era utilizado para questões de ensino (SOARES, 2002b). Assim, observa-se que o português, além de não existir não fazia parte do currículo escolar, pois os poucos que frequentavam a escola faziam parte da elite dominante e seguiam o modelo educacional vigente na Europa e era uma língua que não tinha valor cultural (HOUAISS, 1985)

No entanto, com as reformas desencadeadas pelo Marques de Pombal o uso da língua portuguesa passou a ser obrigatório além de sua inclusão e valorização nas escolas, com isso os alunos passaram a estudar as estruturas e particularidades da língua portuguesa (PESSANHA; DANIEL; MENEGAZZO, 2003/2004). Contudo, o ensino baseava-se na gramática e na retórica, principalmente, porque a escola até meados dos anos 40 do século XX destinava-se ao público restrito que constitui-se na elite que buscava preservar o purismo linguístico e o bom gosto literário sendo que cabia ao professor aprofundar os estudos a partir do material didático, exercícios e análises da língua (SOARES, 2002b).

Com as reivindicações das camadas populares pelo direito a escolarização houve uma abertura das escolas nas décadas de 50 e 60 o que levou a reformulação dos objetivos e currículos aplicados nas escolas principalmente na disciplina de Língua Portuguesa, visto que a nova clientela trazia uma variedade linguística que não fazia parte da escola. Além disso, houve necessidade mais professores e com isso uma mudança no público acadêmico de Letras que passou a receber pessoas pouco letradas (SOARES, 2001). Apesar dessas mudanças, o ensino continuou sendo o tradicional que somente sofreu alterações após a chegada da linguística nos cursos de Letras, visto que focava no reconhecimento das diferentes variedades da língua que agora faziam parte das salas de aula (SOARES, 1997).

Na década de 70 com a Lei de Diretrizes e Bases da Educação a disciplina de Português sofre mudanças, pois a língua passa a ser concebida, então, como instrumento colocando como importante a comunicação. Com isso os objetivos da disciplina tornam-se práticos e utilitários.

Já na década de 80 com a abertura política novas teorias da área das ciências linguísticas começam a penetrar na escola e trazem discussões sobre os rumos da disciplina. Nesse contexto, a linguística através de estudos acerca da língua portuguesa escrita e falada traz novas concepções da gramática, porque há o reconhecimento da gramática tanto na língua escrita como falada e a linguística textual amplia a concepção da função e natureza a gramática em fins didáticos, pois fica evidente a necessidade e a conveniência de que a gramática não fique limitada somente as estruturas fonológicas e morfossintáticas, mas que se torne viva e possa chegar aos textos (SOARES, 2002).

2.6 A GRAMÁTICA E OS ESTILOS LITERÁRIOS COMO COMPONENTES CURRICULARES

Com a linguística pode-se vislumbrar novas perspectivas para o ensino da língua portuguesa tendo como objetivo a incorporação na pratica dos professores da concepção interativa da linguagem, e da valorização do sujeito e das variedades linguísticas, entretanto o que se observa é que a realidade do ensino da língua se mantém baseado na literatura e na gramática na forma escrita da língua (SOARES, 2001)

A crítica a esse modelo tradicionalista de ensino da língua portuguesa, que encontra-se arraigado na disciplina, está justamente no fato de que a gramática é tratada como normativa que constitui-se no conjunto de regras que o usuário deve saber para falar e escrever de forma correta (CAMARA JUNIOR, 1986). Nesse contexto a língua é tida como autônoma e homogênea e usos fora das regras da norma-padrão são considerados erros (BAGNO, 1999; CAGLIARI, 1990). Além disso, os conteúdos que são ensinados são paradigmas e classificações que pouco colaboram com a formação de um aluno competente no uso da língua principalmente em sua modalidade escrita. E, ainda, em qualquer fase do ensino da Língua Portuguesa é importante que os alunos possam refletir sobre as funções que os elementos linguísticos exercem no texto escrito e falado.

Segundo Cook (1996), Germain e Séguin (1999) e o Conselho da Europa (2001) o conhecimento do locutor sobre os recursos que compõem a língua praticada e a sua compreensão são designados como competência gramatical e dentro desta estão os domínios da morfologia e sintaxe. Sendo que a primeira trabalha com o discurso, ou seja, substantivas, verbos, adjetivos, advérbios, pronomes, preposições, conjunções e interjeições, além da

formação das palavras, já a segunda é a que discute a forma como as palavras são utilizadas em uma frase, suas combinações e como as proposições se entrelaçam.

Com isso observa-se que o conhecimento, que cada locutor dispõe acerca da gramática e a sua habilidade na produção de discursos utilizando esses conhecimentos dependem de fatores como o ambiente em que está inserido, seu acesso a informação, a forma como essa informação é internalizada para geração de conhecimento e como foi seu contato no ambiente escolar com a gramática. Portanto, o ensino da gramática vai além da explicação de regras e padrões e normas linguísticos, a didática atualmente pede que a língua seja apresentada ao aluno em um contexto comunicativo e relevante, a estrutura gramatical que o professor deseja alcançar deve estar em consonância aos objetivos comunicativos do aluno, a atividades prática promovam a aquisição de itens linguísticos e da linguística pelos alunos e a escolha por uma abordagem onde o estudo da gramática seja interessante e motivador (BROWN, 2007).

Além disso, na literatura encontra-se que os alunos do ensino médio tem dificuldades no conteúdo que aborda a literatura, pois de acordo com essas pesquisas o conteúdo, a forma de abordagem e a metodologia utilizada não atendem as expectativas da clientela, além de haver um distanciamento entre a proposta de ensino e a realidade dos atores envolvidos no processo ensino aprendizagem, porém apesar dessas constatações esses estudos também identificam que para o alunado o aprendizado da literatura, principalmente, com relação a leitura é muito importante. Assim unir o ensino da literatura com a gramática pode ser um fator estimulante e que melhore a educação, pois o professor poderá estimular a pesquisa, leitura e observação das características sintáticas dentro de livros da literatura brasileira e o aluno se torna um sujeito pesquisador. Atitudes estas que rompem com os paradigmas vigentes e abrem espaço para construção de sujeitos que são capazes de inferir e modificar o meio em que vivem (ZINANI, 1991)

Como Cereja (2005, p.54) deixa claro,

> Depois de anos de estudo de literatura, os jovens brasileiros deixam o ensino médio sem terem desenvolvido suficientemente certas habilidades básicas de análise e interpretação de textos literários, tais como levantamento de hipóteses interpretativas; rastreamento de pistas ou marcas textuais; reconhecimento de recursos estilísticos e de sua função semântico-expressiva; relações entre a forma e o conteúdo do texto; relações entre os elementos internos e os elementos externos (do contexto sócio-histórico) do texto; relações entre o texto e outros textos, no âmbito da tradição; relações entre o texto verbal e texto não verbal, etc.

Para tanto, tendo em vista o exposto se faz relevante a construção de um software que possibilite ao aluno observar de maneira dinâmica a análise gramatical, nesse caso mostrando

para o aluno as funções das palavras dentro da frase, e, também, que ele possa realizar comparações entre as formas de escritas dos autores de diferentes estilos literários da literatura brasileira, de maneira que possa participar da construção do seu conhecimento se tornado agente atuante no processo educacional deixando, assim, de ser agente passivo de forma que possa passar a refletir e ser crítico de sua língua materna.

2.7 SOFTWARES NO ENSINO DE PORTUGUÊS

A forma de ensino da Língua Portuguesa já teve inúmeras transformações passando pela educação jesuítica até a Lei de Diretrizes e Base da Educação Nacional nº 9394/96 – LDB (BRASIL, 1996) e dentro dessas formas de ensino estão a linguagem como forma de expressão do pensamento, forma de comunicação e de interação (GERALDI, 1990). Contudo, com as tecnologias e suas crescentes inovações o desafio apresentado aos atores envolvidos no processo educacional principalmente professores, e, nesse caso, os da disciplina de Língua Portuguesa é de que maneira utilizar-se das tecnologias no processo de ensino aprendizagem especificamente de sua disciplina.

Além de realizar intervenções que tem como foco o desenvolvimento da leitura, oralidade e escrita através de textos, o professor necessita atuar como facilitador e garantir que o aluno esteja preparado para lidar com o acesso as diferentes culturas da mídias e o uso desses recursos tecnológicos, pois, atualmente, uma das formas de comunicação mais comum é através da internet, onde, normalmente, o padrão culto da escrita não prevalece. Um exemplo do estilo de comunicação a que os alunos estão expostos quando usuário da tecnologia e que faz parte da cultura das mídias é o gênero multimodal, que é constituído por todas as formas infográficas de linguagem criadas pelo homem, as práticas sociais do discurso e suas formas de composição nos suportes tecnológicos que lhe garantam significado (FARACO, 2002).

> [...] desde o aparelho fonador até as redes digitais atuais, embora, efetivamente não passem de meros canais para a transmissão de informação, os tipos de signos que por eles circulam, os tipos de mensagens que engendram e os tipos de comunicação que possibilitam são capazes não só de moldar o pensamento e a sensibilidade dos seres humanos, mas também de propiciar o surgimento de novos ambientes socioculturais (SANTAELLA, 2003, p.13).

Assim, os espaços tecnologias, como por exemplo a Internet, estão em evidência no cotidiano, tanto como a sociedade das conexões, da hibridização de culturas e de gêneros discursivos. Com isso, cabe ao professor de Língua Portuguesa ter uma nova postura frente a esse desafio imposto pela sociedade que se construiu e se constrói em torno das tecnologias da

informação e comunicação de maneira a ser um mediador na construção do conhecimento, principalmente, da Língua Portuguesa.

De acordo com Libâneo (2011, p. 40):

> A escola continuará durante muito tempo dependendo da sala de aula, do quadro-negro, dos cadernos. Mas as mudanças tecnológicas terão um impacto cada vez maior na educação escolar e na vida cotidiana. Os professores não podem mais ignorar a televisão, o vídeo, o cinema, o computador, o telefone, o fax, que são veículos de informação, de comunicação, de aprendizagem, de lazer, porque há tempos o professor e o livro didático deixou de serem as únicas fontes do conhecimento. Ou seja, professores, alunos, pais, todos precisamos aprender a ler sons, imagens, movimentos e a lidar com eles.

Além disso, tanto se faz necessário, que o professor ensine os seus alunos a buscar informação e fazer a triagem dela, quanto, o aluno precisa envolver-se no processo de aprendizagem, pois quanto mais ativa for sua participação mais ela irá integrar e reter o que está sendo exposto (RIBEIRO, 2006; LÉVY, 1993). Para tanto, o auxílio de um software no ensino da Língua Português, em especial aplicado a gramática, favorece a exploração e a construção saber.

2.8 CONSIDERAÇÕES FINAIS

A aplicação dos conhecimentos obtidos através dos estudos da linguística, que posteriormente foram utilizados e aplicados na área computacional gerando métodos para identificação de características sintáticas em textos, possibilitam a aplicação destes em softwares com capacidade de analisar e produzir informações cuja aplicação, por exemplo, pode ser a apresentação de características sintáticas de textos de forma a simplificar o entendimento destas.

Além disso, apesar das evoluções tecnológicas crescentes e das pressões para que a escola e consequentemente ensino agreguem e explorem as tecnologias informação e comunicação, especificamente, no caso da disciplina de Língua Portuguesa houve pouca evolução no que concerne as práticas de ensino e a inclusão de tecnologias como método enriquecedor da didática cotidiana.

Com isso, encontrou-se como lacuna para pesquisa científica o desenvolvimento de um software, que possa vir a suprir essa demanda, em especial que esteja relacionado e facilite o entendimento do conteúdo de gramática da Língua Portuguesa Brasileira e Literatura.

3 MATERIAIS E MÉTODOS E TÉCNICAS

Este capítulo apresenta as ferramentas e as tecnologias utilizadas para o desenvolvimento do software apresentado neste trabalho, descreve-se, também, o método, como uma sequência de passos para modelagem e implementação e as técnicas onde são apresentados os diagramas da UML, bem como algumas especificações do código fonte desenvolvido.

3.1 FERRAMENTAS E TECNOLOGIAS

O Quadro 1 apresenta as ferramentas e as tecnologias utilizadas para o desenvolvimento do *software*. As ferramentas e tecnologias incluem as utilizadas durante o ciclo de vida da aplicação, da definição dos requisitos aos testes.

NOME	VERSÃO	LOCALIZAÇÃO (endereço*web*)	DESCRICÃO DE USO
Astah* Community	6.6.3	http://astah.net/download#community	Documentação do sistema baseada em UML
Netbeans	7.2.1	https://netbeans.org/	IDE de desenvolvimento
MySQL Server	5.5.27	http://www.mysql.com/downloads/mysql/	Sistema Gerenciador de Banco de dados
MySQL GUI Tools	5.0	http://dev.mysql.com/downloads/gui-tools/5.0.html	Ferramentas de administração do sistema gerenciador de banco de dados MySQL
MySQL Workbench	5.2.40	http://dev.mysql.com/downloads/workbench/5.2.html	Ferramenta de administração e modelagem de banco de dados MySQL
Linguagem Java	7	http://www.oracle.com/technetwork/java/javaee/downloads/index.html	Linguagem de programação
Hibernate	4.3.1	http://www.hibernate.org/hibernate	Biblioteca de mapeamento Objeto-Relacional
CoGrOO	4.0.0	www.cogroo.sourceforge.net	Biblioteca de análise sintática

Quadro 1 - Ferramentas e tecnologias utilizadas
Fonte: Elaborado pelo Autor.

3.2 TECNOLOGIAS APLICADAS AO DESENVOLVIMENTO DE *SOFTWARE*

Para o desenvolvimento do *software* estabelecido neste projeto foi necessário o uso de algumas tecnologias e métodos essenciais para o processo. Nas próximas seções serão abordados detalhes, tais como: o paradigma de orientação a objetos, a linguagem de modelagem UML (*Unified Modeling Language*), a linguagem de programação Java, o banco de dados MySQL, a biblioteca CoGrOO e a ferramenta de persistência de dados Hibernate.

3.2.1 Programação orientada a objetos

Um dos primeiros paradigmas de desenvolvimento a se popularizar foi o procedimental ou estruturado. A maneira como se codificava era linear, muito próximo ao *hardware* e utilizava conceitos matemáticos para definir soluções. "Um algoritmo é uma sequência ordenada e finita de etapas, cuja execução passo a passo resolve um determinado problema" (VILARIM, 2004, p.7).

O desenvolvimento de aplicações baseadas neste paradigma era de uma complexidade muito alta, levando a descontinuação de vários projetos de *software*. Para tentar reverter essa realidade foi criada uma nova abordagem de desenvolvimento, o padrão de programação orientada a objetos. Este, por sua vez, permitia a criação de um código de melhor legibilidade, rotinas podiam ser mais facilmente reutilizadas e processos complexos podiam ser escritos de forma mais compreensível, o que implica diretamente em uma melhoria no tocante a manutenção.

Segundo Deitel (2010), a Orientação a Objetos (OO) é um paradigma que aproxima o programador do mundo real, no qual tudo pode ser visto como objetos, como por exemplo, livro, aluno, faculdade, etc. Antes da orientação a objetos, o desenvolvimento se preocupava com as ações desses objetos, a programação se baseava nos verbos, como por exemplo: alugarLivro, cadastrarAluno, realizarMatricula, etc. O programador recebia os problemas em objetos e codificava em verbos. Com o advento da orientação a objetos, o programador passou a codificar exatamente o que via. A modelagem passou a se basear nos substantivos, como, por exemplo, o livro, o aluno, etc.

3.2.2 Linguagem de Modelagem Unificada - UML

A UML, *Unified Modeling Language*, ou Linguagem de Modelagem Unificada surgiu da união de três métodos de modelagem: o método de Booch, o método OMT (*Object Modeling Technique*) de Rumbaugh e o método OOSE (*Object-oriented software engineering*) de Jacobson. Estas eram, até meados da década de 90, os três métodos de modelagem orientada a objetos mais populares entre os profissionais da área de desenvolvimento de *software*. A união dessas metodologias contou com o amplo apoio da Rational Software, que incentivou e financiou a união das três metodologias (MEDEIROS 2006).

A UML é uma linguagem visual utilizada para modelar sistemas computacionais por meio do paradigma de Orientação a Objetos. O objetivo da UML é ajudar a definir as características do *software*, tais como seus requisitos, seu comportamento, sua estrutura lógica, a dinâmica de seus processos e suas necessidades físicas em relação ao equipamento sobre o qual o sistema deverá ser implantado. Todas essas características são definidas por meio da UML, antes do *software* começar a ser realmente desenvolvido (GUEDES, 2004).

A UML é composta por diferentes tipos de diagrama, cada um representando o sistema sob uma determinada ótica. A utilização de diversos diagramas permite que falhas sejam descobertas, diminuindo a possibilidade da ocorrência de erros futuros.

3.2.3 Linguagem de Programação Java

Java é uma linguagem de programação orientada a objetos que foi desenvolvida pela Sun Microsystems. É antes de tudo uma linguagem simples, fortemente tipada, independente de arquitetura, robusta, segura, extensível, bem estruturada, distribuída, *multithreaded* e com coletor de lixo. Além disso, é uma linguagem de alto nível muito parecida com C++, porém, mais simples. Java não possui sobrecarga de operadores, *structs*, *unions*, aritmética de ponteiros, herança múltipla, arquivos .h, diretivas de pré-processamento e a memória alocada dinamicamente é gerenciada pela própria linguagem, que usa algoritmos de coletor de lixo para liberar regiões de memória que não estão mais em uso. Paralelamente, é uma linguagem independente de plataforma. Isto quer dizer que um programa em Java pode funcionar em qualquer sistema operacional. Isto é possível porque existe uma Máquina Virtual para cada sistema operacional e que se encarrega de executar o programa de Java (DEITEL, 2010)

Atualmente Java é utilizado em um amplo leque de aplicações, sendo possível seu uso para o desenvolvimento de sistemas para internet, desktop, aplicativos móveis, cartões inteligentes, entre outras plataformas. Além disso, há disponível uma ampla quantidade de *frameworks* e algoritmos que visam melhorar a organização e desempenho do *software* desenvolvido e também podem proporcionar ganhos significativos na produtividade do processo de desenvolvimento.

3.2.4 Banco de Dados MySQL

O MySQL é um sistema de gerenciamento de banco de dados relacional muito utilizado em sistemas *web*, principalmente por sua performance, velocidade e facilidade de uso em sistemas deste tipo.

O MySQL se tornou o mais popular banco de dados *open source* do mundo porque possui consistência, alta performance, confiabilidade e facilidade de uso. Atualmente é utilizado em mais de 70 milhões de instalações em todos os continentes (inclusive na Antártica), que vão desde instalações em grandes corporações até aplicações embarcadas. Além disso, o MySQL se tornou a escolha de uma nova geração de aplicações que utilizam o modelo LAMP (Linux, Apache, MySQL, PHP) e WEB 2.0 (MYSQL AB, 2009).

> *"O servidor de MySQL controla o acesso aos dados para assegurar que vários usuários possam trabalhar com os dados, ao mesmo tempo fornece acesso rápido aos dados e assegura que somente usuários autorizados obtenham acesso." (WELLING; THOMSON, 2005).*

Comparando o MySQL com seus principais concorrentes, como o Oracle, PostgreSQL e Microsoft SQL Server, é possível listar algumas vantagens, dentre elas (WELLING; THOMSON, 2005):

- Alto desempenho;
- Baixo custo;
- Fácil configuração e aprendizado;
- Portabilidade;
- Disponibilidade de código-fonte;
- Disponibilidade de suporte ao usuário.

3.2.4.1 Ferramentas ORM

Object Relational Mapping (ORM) são ferramentas ou *frameworks* complexos que realizam o mapeamento do objeto no modelo relacional de forma automatizada e transparente (BAUER, KING, 2005). Algumas ferramentas reduzem significantemente o trabalho dos desenvolvedores para persistir os objetos. Em Bauer e King (2005) os autores fazem uma análise sobre a importância das ferramentas ORM. Eles afirmam que as ferramentas trazem um ganho considerável em termos de produtividade para o projeto. Afirmam ainda que, embora

ocorra certa penalidade no desempenho da aplicação, isso se torna válido visto o ganho em produtividade. Alegam também que, desenvolver uma aplicação com bom desempenho para diversos tipos de banco de dados é uma tarefa árdua e nem sempre se escolhem os comandos mais performáticos, o que não ocorre em uma ferramenta ORM com anos no mercado. Esta possui conhecimento suficiente para decidir qual o melhor comando para cada banco e, assim, compensar a questão de desempenho perdido.

3.2.4.2 Hibernate

Hibernate é uma das ferramentas ORM mais utilizadas no ambiente corporativo. Atende a todos aos requisitos de uma ORM. Implementada no ambiente Java com código aberto (*OpenSource*), o Hibernate provê uma arquitetura flexível e configurável (BAUER; KING, 2005).

Essa ferramenta, por exemplo, disponibiliza algumas maneiras de obtenção de dados, através de uma linguagem de consulta própria (*Hibernate Query Language* ou HQL), sendo esta parecida com SQL e uma API (*Application Programming Interface*) de consulta por critérios (QBE – *Query By Example*), um modo seguro para expressar consultas. Assim, o desenvolvedor não precisa se preocupar com uma linguagem diferente para cada tipo de banco de dados. A ferramenta também possui métodos para execução de comandos SQL, ideal para soluções complexas ou que necessitem de um melhor desempenho.

Segundo Bauer e King (2005), um dos objetivos do Hibernate é automatizar 95% do trabalho de persistência realizado pelo desenvolvimento sem uma ORM, resultando em um ganho de produtividade, um requisito importante em grandes projetos.

3.2.5 Biblioteca CoGrOO

O sistema CoGrOO foi iniciado em 2004, motivado pela falta de um corretor gramatical nos *softwares* da família *LibreOffice* o que os tornava menos competitivos em relação as ferramentas não *open source*. Trata-se de um projeto de código aberto capaz de identificar erros de português como erros de concordância nominal e verbal, crase, regência nominal e verbal e outros erros comuns na escrita em português brasileiro.

Para realizar a correção gramatical, o sistema CoGrOO analisa a entrada de texto do usuário e traz como resultado uma lista de possíveis erros, para realizar esse processo é realizada

uma análise superficial seguida de uma verificação baseado em regras. Na análise inicial do texto é utilizada a técnica de Processamento de Linguagem Natural cuja função, segundo Abrahão (1997), "*visa simular computacionalmente as capacidades humanas de comunicação e interpretação, ditas Linguagens Naturais, utilizando técnicas de representação de conhecimento*".

A utilização do CoGrOO junto ao sistema desenvolvido visou a utilização de suas funcionalidades de etiquetagem, que são utilizadas na identificação das funções sintáticas.

3.3 MÉTODO

Realizou-se inicialmente um levantamento de requisitos junto aos *stakeholders* buscando identificar as funcionalidades e definir o escopo do software. De posse dos requisitos desenvolveram-se os diagramas necessários para entendimento da estrutura e funcionalidades do software.

Para modelagem UML do software foi selecionada a ferramenta Astah Community, que é uma ferramenta gratuita com funcionalidades de criação de diagramas de classes, casos de uso, máquina de estados, atividades, sequência, comunicação, entre outros.

Para a criação do diagrama de entidades e relacionamentos optou-se pela utilização do MySQL Workbench, que é uma ferramenta com as funcionalidades de modelagem de banco de dados, desenvolvimento SQL e administração de bases de dados.

Para o software em questão foram desenvolvidos os diagramas de caso de uso, diagrama de classes e diagrama de entidade e relacionamento utilizando as ferramentas mencionadas.

Com a estrutura do software planejada, ilustrada e documentada por meio dos diagramas, foi então realizado o levantamento das tecnologias que atendem às propostas de desenvolvimento. Levando em consideração de que não haveria investimento na aquisição de soluções comerciais, optou-se pela utilização de tecnologias de livre utilização, como a linguagem de programação Java, a IDE (*Integrated Development Environment*) NetBeans, e demais ferramentas componentes aplicadas no desenvolvimento do *software*.

Os fatores que exerceram influência na escolha da linguagem de programação Java foi o domínio desta linguagem por parte da responsável pelo desenvolvimento, o que afeta diretamente o tempo empregado em uma eventual curva de aprendizagem no caso de escolha de uma linguagem até então desconhecida, além obviamente, da ciência de que a linguagem dispunha de todos os recursos necessários para o desenvolvimento do *software* em questão.

De forma a agilizar o processo de desenvolvimento foi realizado um breve levantamento sobre as opções de IDEs compatíveis com a linguagem Java de livre utilização (sem necessidade de investimentos em licenciamento), por mais que isso pudesse resultar em variações de produtividade por não contar, eventualmente, com recursos existentes em soluções proprietárias. A IDE NetBeans oferece um ambiente de desenvolvimento que abstrai a necessidade do estudo de comandos de compilador para tarefas de compilação do projeto, além de permitir a execução e depuração do projeto, entre outras funcionalidades. Um fator preponderante na escolha da IDE NetBeans frente às concorrentes, como por exemplo a IDE Eclipse, foi que a mesma possui, sem necessidade de instalação e configuração de complementos, uma interface gráfica para desenvolvimento para aplicações *desktop* que permite a criação de telas utilizando, por exemplo, funcionalidade de "arrastar e soltar" (*Drag-and-Drop*), além de abstrair a necessidade de conhecimentos de utilização de classes de gerenciamento de *layout*, e codificação manual do posicionamento dos elementos gráficos de interface, o que implicaria, certamente, em uma queda brusca de produtividade.

Para a persistência de dados do *software* houve a necessidade da escolha de um sistema de gerenciamento de banco de dados. A opção pelo MySQL se deu levando em consideração o desempenho satisfatório da solução e a compatibilidade entre as tecnologias utilizadas.

A análise sintática de textos pode ser considerada a parte principal do *software* desenvolvido, sendo assim, foi realizado levantamento sobre *frameworks* que dispusessem de funções de análise, de forma que o escolhido pudesse ser parte integrante do *software*. Por meio do levantamento realizado, logo chegou-se à biblioteca CoGrOO, a qual apresenta as funções etiquetagens necessárias para identificação das funções sintáticas, viabilizando assim o desenvolvimento do sistema.

Após a realização do desenvolvimento, o software desenvolvido entrou em fase de testes. Nesta fase, um corpus de textos foi submetido ao processo de extração de características sintáticas com a finalidade de avaliar se as extrações estavam ocorrendo de forma correta. Nesta fase, também foi possível efetuar algumas alterações e correções no software, antes de entrar em operação.

A Figura 2 tem por objetivo ilustrar as fases compreendidas dentro do método apresentado.

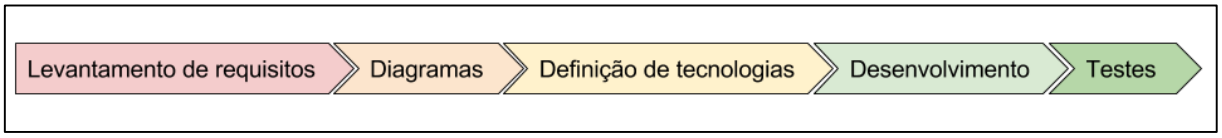


Figura 2 - Fases compreendidas dentro do método aplicado

Fonte: Elaborado pelo Autor

3.4 PROCESSO DE EXTRAÇÃO DE CARACTERÍSTICAS

O processo de extração de características engloba uma série de processos aninhados que são necessários para geração dos vetores de características. Este processo pode ser melhor visualizado através da Figura 3.

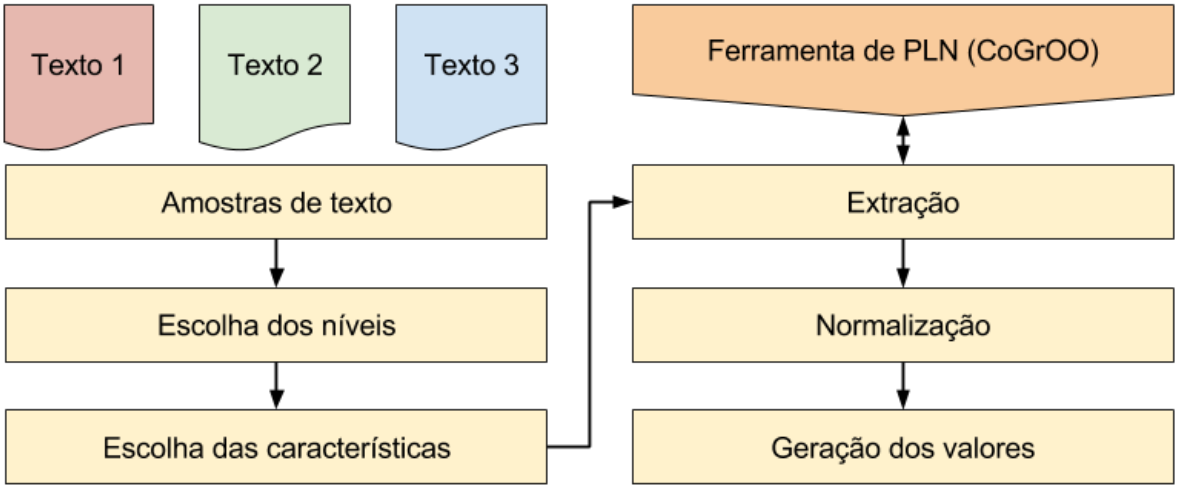


Figura 3 - Processo de extração de características
Fonte: Elaborado pelo Autor

O primeiro passo é indicar qual amostra de texto ou conjunto de amostras será submetido ao processo de extração de características. Quando submetido ao processo de extração o texto é atribuído a um determinado autor conhecido ou não. Posteriormente à escolha do usuário são realizadas as configurações para a extração das características conforme os seus níveis. Para auxiliar no processo de extração das informações sintáticas, uma ferramenta de Processamento de Linguagem Natural (PLN) é utilizada para realizar a rotulagem das palavras. De acordo com a seleção das características efetuadas são gerados os vetores com os valores correspondentes de forma normalizada.

Para detalhar de forma mais aprofundada a etapa de extração de características, são descritas as próximas subseções sobre as ferramentas utilizadas e a geração dos vetores.

3.4.1 Ferramenta de Extração de Características

Para que o processo de extração seja customizado foi desenvolvido uma interface gráfica (Figura 4), que permite a extração baseada nas características sintáticas da língua portuguesa desejadas.

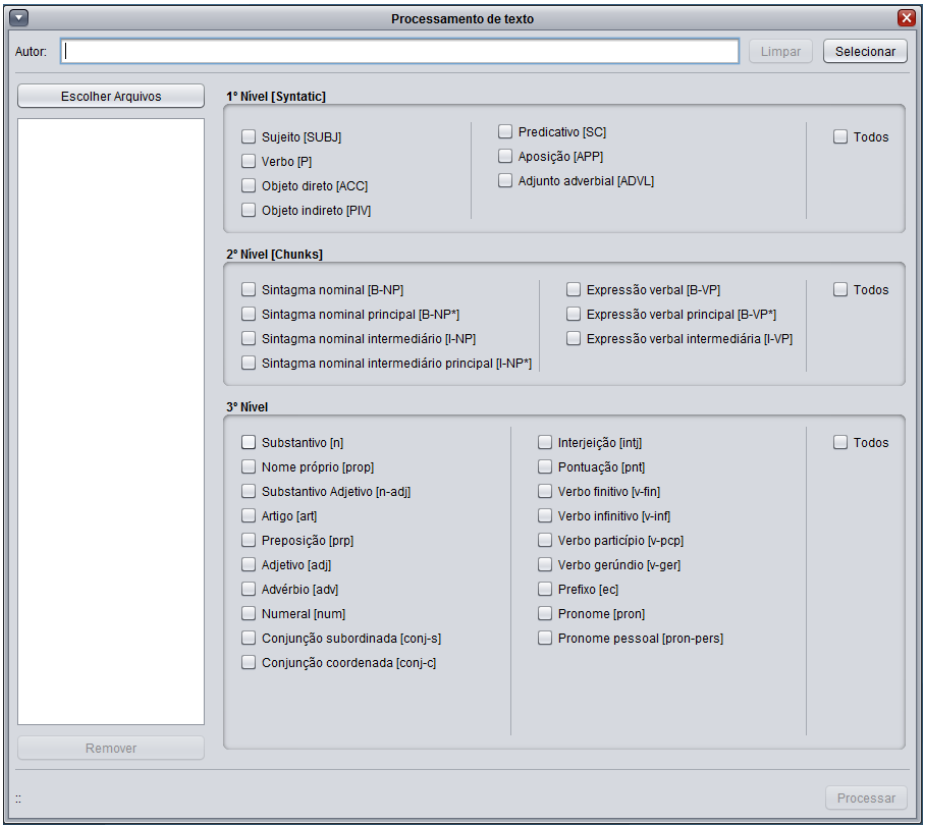


Figura 4 - Tela de seleção e extração de características.
Fonte: Elaborado pelo Autor

Para extrair as informações de estrutura sintática das amostras dos textos, o *software* faz uso da biblioteca de correção gramatical CoGrOO. Esta biblioteca possui uma série de funções que possibilitam a realização de análise e a obtenção de estruturas de dados contendo informações estruturais sobre o texto analisado.

As informações resultantes das análises de textos, além dos registros de processamento de textos e autores são guardados pelo sistema utilizando uma estrutura de banco de dados desenvolvida utilizando o MySQL.

O *software* foi desenvolvido de forma a possibilitar a extração de 3 níveis de características sintáticas em separado ou em conjunto (Tabela 1). Maiores informações a respeito do *software* são encontradas no Capítulo 4.

Tabela 1 - Níveis de características sintáticas

Nível de Extração	Características
1º	Sujeito, verbo, objeto indireto, objeto direto, predicativo, aposição, adjunto adverbial e agentes de voz ativa e passiva.
2º	Sintagma nominais (parte da oração que contém um nome ou pronome) e verbais (parte da oração que contém o verbo).

3º	Substantivos, artigos, preposições, adjetivos, advérbios, numerais, conjunções, interjeições e sinais de pontuação.

Fonte: Elaborado pelo Autor

Nível 1

A análise sintática pode ser definida como o estudo das palavras a parti da função que desempenham dentro de um contexto linguístico (DUARTE, 2012). Sendo que, para que esta função seja identificada é preciso ter conhecimento de todos os elementos sintáticos que são denominados: sujeito, predicado, complemento nominal e verbal, adjunto adnominal, vocativo, aposto, agente da passiva, entre outros. Um exemplo deste tipo de análise é dado por Duarte (2012) através da análise da frase: "Acreditamos sempre em dias melhores", onde o sujeito oculto é revelado a partir do número apresentado pelo verbo acreditar (acreditamos); o predicado verbal é designado por meio da ação que fica implícita no sujeito (acreditamos sempre em dias melhores); é identificado um adjunto adverbial de tempo expresso pela palavra sempre; o objeto direto, que é o complemento do verbo acreditar (em dias melhores); e um predicativo do objetos dada pela palavra "melhores", que confere uma característica ao objeto direto. Na Figura 5 pode ser vista uma árvore sintática da análise da frase.

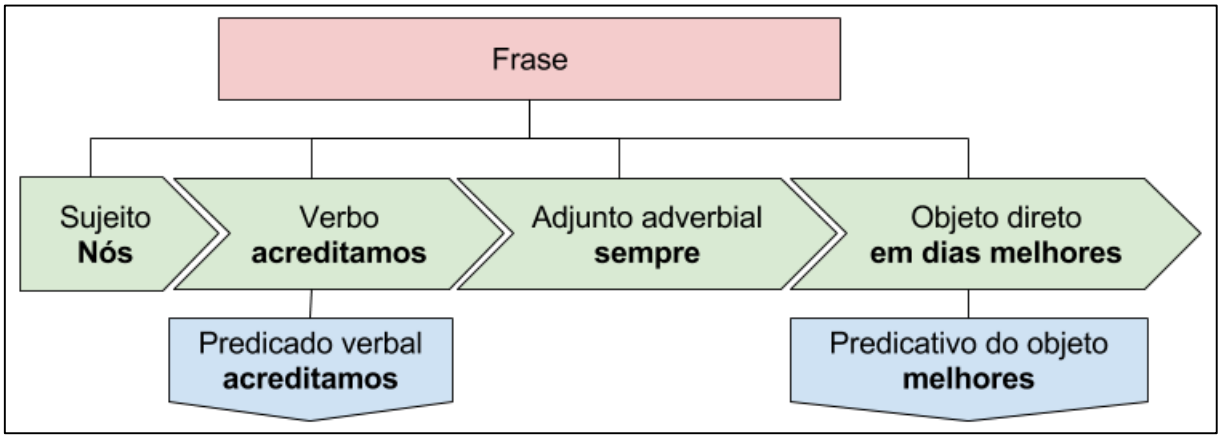


Figura 5 - Árvore sintática
Fonte: Adaptado de (Duarte, 2012)

Quando se opta pelo 1º nível é possível fazer a extração de características baseadas em informações de sujeito, verbo, predicado e complemento. Tais características também podem ser extraídas em separado ou em conjunto. Este nível de extração perfaz o conjunto de novas características propostas pela abordagem deste trabalho, onde é possível extrair os termos essenciais, integrantes e acessórios de cada oração.

Nível 2

Já no 2º nível de extração de características são possíveis de extrair características da gramática sintagmática. Nestas classes estão incluídos os sintagmas nominais e verbais, que apresentam meios mais simplificados para a descrição da estrutura das orações.

Os sintagmas são identificados a partir dos elementos que compõem a oração e, também, são uma unidade importante, pois devido a sua organização em torno do núcleo da oração mantem a dependência e ordem entre si, sendo que o núcleo sozinho pode constituir o sintagma (DUARTE, 2012).

Para melhor identificar um sintagma em uma frase, recorre-se a um exemplo: "A virtude é uma característica humana".

Quando o núcleo é um nome ou pronome ("a virtude"), tem-se um sintagma nominal. Os demais elementos da oração possuem como elemento fundamental o verbo ("é uma característica humana"), razão pela qual são denominados de sintagmas verbais (DUARTE, 2012).

Na Figura 6é possível observar a estrutura dos elementos dos sintagmas através da representação da árvore sintática sintagmática.

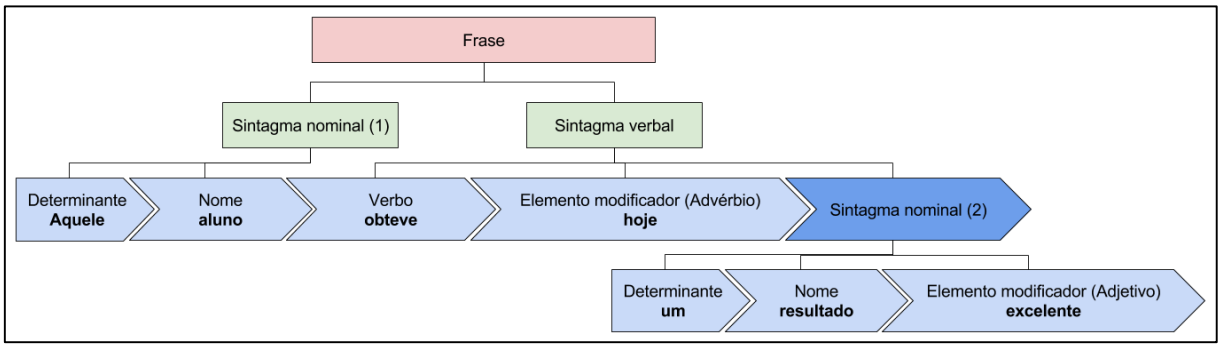


Figura 6 - Árvore sintática sintagmática
Fonte: Adaptado de (Duarte, 2012)

Percebe-se que existem alguns elementos constituintes da frase, que são essenciais para a análise sintática sintagmática, tais como o determinante que é representado geralmente pelos artigos, numerais e pronomes. Existe também o elemento modificador que muitas vezes são representados por advérbios e adjetivos (DUARTE, 2012).

Nível 3

No 3º nível podem ser selecionadas as classes mais comuns das palavras, tais como: substantivos, adjetivos e advérbios. Neste caso, são extraídas cada uma das classes de palavras em separado, combinadas ou no conjunto completo. Ao selecionar uma ou mais classes de palavras, o *software* realizará a extração das características baseadas nestes parâmetros.

Esta análise é denominada morfológica, pois estuda as palavras de acordo com a classe gramatical a que elas pertencem. Como exemplo, na Figura 7 classificam-se as palavras de acordo com a sua classe.

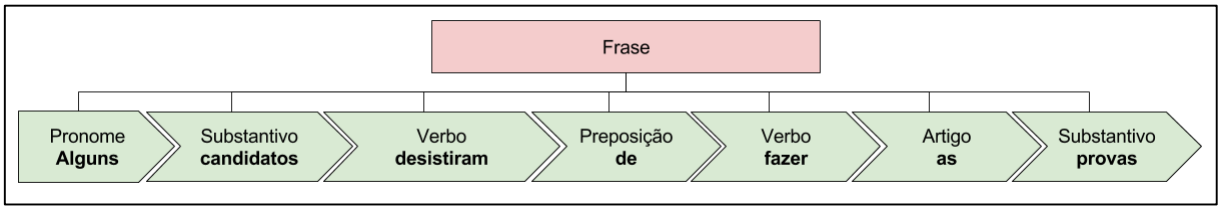


Figura 7 - Árvore sintática morfológica
Fonte: Adaptado de (Duarte, 2012)

Ao final o *software* gera os vetores de acordo com as caraterísticas selecionadas pelo usuário.

3.4.2 Ferramenta de Processamento de Linguagem Natural - PLN

Partindo de uma premissa que já existem diversos Processadores de Linguagem Natural em língua portuguesa, optou-se em não desenvolver um método próprio de análise sintática, e sim fazer uso de uma ferramenta de PLN.

As ferramentas disponíveis atualmente são sete softwares e três bibliotecas: (CORREA, 2015):

- GATE: é um software de código aberto para processamento e análise de textos;
- StandfordNLP: é um conjunto de softwares para PLN, sendo que processam textos em inglês, francês, alemão, chinês e arabico;
- LX-Center: criado pela Universidade de Lisboa tem um conjunto de ferramentas para PLN de textos em português de Portugal;
- CoGroo: é um corretor gramatical que atua como PLN para o português do Brasil;
- WordNET: banco de dados do léxico inglês;
- MXPOST: etiquetador morfossintático com suporte para português brasileiro;
- TreeTAgger: etiquetador morfossintático com suporte para português brasileiro;
- UIMA: biblioteca para processamento de textos não-estruturados;
- LingPipe: biblioteca Java para processamento de textos;
- OpenNLP: biblioteca Java para PLN baseada em aprendizado de máquina

A ferramenta de PLN escolhida foi a biblioteca de correção gramatical utilizada nos editores de textos do Open, Libre e Br Office – CoGrOO (SILVA, 2013). Tal biblioteca tem por função gerar as árvores sintáticas com as subclassificações de cada palavra encontrada no texto, e é derivado do Projeto JSpell (ALMEIDA E SIMÕES, 2007). O CoGrOO é um sistema hibrido de PLN, pois utiliza a análise estatística para segmentação e categorização do texto (SILVA, 2013).

O processo do CoGrOO quando recebe um texto para processamento consiste de algumas etapas, que são (SILVA, 2013):

(i) recebe um texto e efetua a divisão em frases;

(ii) recebe uma frase e efetua a divisão em palavras e sinais de pontuação;

(iii) recebe a quantidade de sentenças e identifica os potenciais nomes próprios;

(iv) recebe a quantidade de sentenças, a localização dos nomes próprios e transforma em um único símbolo;

(v) recebe uma frase e atribui um rótulo morfológico, de acordo com seu contexto;

(vi) recebe um texto marcado e verifica se é uma frase nominal (NP) ou uma frase verbal (VP);

(vii) recebe uma sentença marcada com NPs e VPs, e efetua a procura pela sujeito. Se encontrar o sujeito, marca o NP como sujeito de uma VP.

Na Figura 8 é possível observar um exemplo de uma árvore sintática construída pela biblioteca CoGrOO. Nela é possível observar os níveis de aprofundamento da árvore.

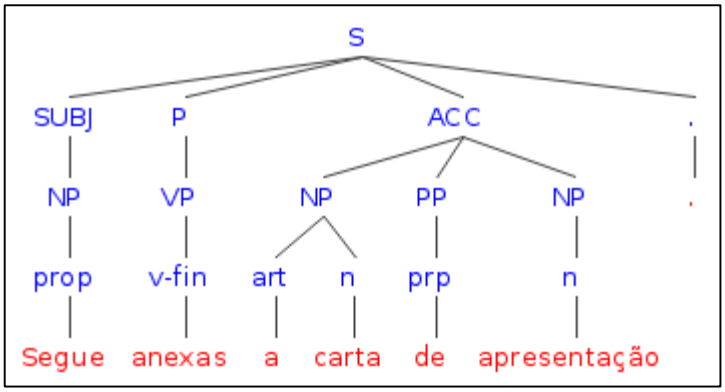


Figura 8 - Exemplo de árvore de estrutura sintática gerada pelo CoGrOO
Fonte: Elaborado pelo Autor

Primeiramente a frase (S) é identificada, gerando assim, uma espécie de tronco da árvore sintática. Posteriormente, uma análise rasa da frase é efetuada pelo CoGrOO, determinando assim os primeiros rótulos de cada parte da frase (ver Tabela 2 para descrição dos rótulos). Neste primeiro nível de análise são identificadas as cláusulas da frase, tais como: sujeito, predicado e complemento.

Tabela 2 - Níveis de cláusulas do CoGrOO

Rótulo/TAG	Descrição
ACC	Objeto direto
ADVL	Adjunto adverbial
APP	Aposição Adnominal
DAT	Objeto Dativo
OC	Complemento do objeto
P	Predicado
PIV	Objeto Proposicional
SA	Complemento do advérbio
SC	Complemento do sujeito
SUBJ	Sujeito

Fonte: Adaptado de (SILVA, 2013)

Posteriormente, em um segundo nível da árvore sintática, as palavras são novamente rotuladas com marcadores de frase, também conhecidos como *chunks*, tais como: frase nominal e verbal (ver Tabela 3).

Tabela 3 - Marcadores de *chunks* do CoGrOO

Rótulo/TAG	Descrição
NP	Frase Nominal
VP	Frase Verbal
PP	Frase Proposicional
ADVP	Frase Adverbial

Fonte: Adaptado de (SILVA, 2013)

E em um terceiro nível, cada palavra recebe mais uma classificação conforme a classe gramatical à qual pertence (ver Tabela 4).

Tabela 4 - Classes de palavras do CoGrOO

Rótulo/TAG	Descrição	Rótulo/TAG	Descrição
N	Nome	**V**	Verbo
PROP	Nomes próprios	**V-FIN**	Verbo Finito
SPEC	Especificadores	**V-INF**	Verbo no Infinitivo
DET	Determinantes	**V-PCP**	Verbo no Particípio
ART	Artigo	**V-GER**	Verbo no gerúndio
PRON-PERS	Pronome Pessoal	**NUM**	Numeral (Cardinais)
PRON-DET	Pronome Determinativo	**PRP**	Preposição
PRON-INDP	Pronome Independente	**CONJ-S**	Conjunções Subordinadas
ADJ	Adjetivo	**CONJ-C**	Conjunções Coordenadas
N-ADJ	Nome ou adjetivo	**INTJ**	Interjeições
ADV	Advérbio	**EC**	Prefixos por hífen

Fonte: Adaptado de (SILVA, 2013)

E por fim, também é possível extrair informações das categorias das palavras, tais como: modo, tempo e caso (ver Tabela 5).

Tabela 5 - Grupos de categorias de palavras do CoGrOO

Grupo	Rótulo/TAG	Descrição	Grupo	Rótulo/TAG	Descrição
Gênero	M	Masculino	Tempo	PR	Presente
	F	Feminino		IMPF	Futuro Imperfeito
Número	S	Singular		OS	Perfeito Simples
	P	Plural		MQP	Mais que perfeito
Caso	NOM	Nominativa		FUT	Futuro
	ACC	Acusativa		COND	Condicional
	DAT	Dativa	Modo	IND	Indicativo
	PIV	Prepositiva		SUBJ	Subjuntivo
Pessoa	1	Primeira		IMP	Imperativo
	2	Segunda	Finitude	VFIN	Verbo finito
	3	Terceira		INF	Infinitivo
				PCP	Particípio
				GER	Gerúndio

Fonte: Adaptado de (SILVA, 2013)

De acordo com a seleção das características efetuadas o *software* fará a extração das características, realiza a rotulagem conforme as classes e irá gerar o vetor com os atributos escolhidos.

3.5 TÉCNICAS

3.6 MODELAGEM DO SOFTWARE

Inicialmente foi realizado um levantamento de requisitos, em que foram levantados os seguintes requisitos funcionais:

- O *software* deve permitir a seleção de um diretório contendo arquivos do tipo *.txt;*
- O *software* deve possuir funcionalidade de cadastramento de autores;
- O *software* deve possibilitar a seleção de características sintáticas que serão extraídas do texto processado;
- Deve possibilitar a consulta geral das características extraídas por meio de interface gráfica disponibilizada no próprio *software*;

Na sequência observou-se que o software também poderia ser usado para análise de documentos por pesquisadores da área de linguística o que gerou o seguinte requisito:

- Após a extração, o *software* deverá gerar um arquivo do tipo ARFF abrangendo as características selecionadas pelo usuário;

Como requisitos não funcionais do sistema foram definidos:

- Utilização de tecnologias não proprietárias;

Posteriormente, de posse destes requisitos, foi desenvolvida uma análise de viabilidade de desenvolvimento do *software* levando em consideração as necessidades que as tecnologias e ferramentas envolvidas deveriam satisfazer.

Tendo em vista a compatibilidade entre as tecnologias, não foi identificado nenhum fator que limitasse ou impedisse o desenvolvimento do sistema, tendo sido definido inicialmente o escopo do sistema, bem como suas funcionalidades, foi possível determinar uma estimativa de tempo de desenvolvimento. De posse destas informações se concluiu por ser viável o desenvolvimento do *software* em questão.

Os requisitos foram organizados sob a forma de um diagrama de Caso de Uso conforme exibido na Figura 9:

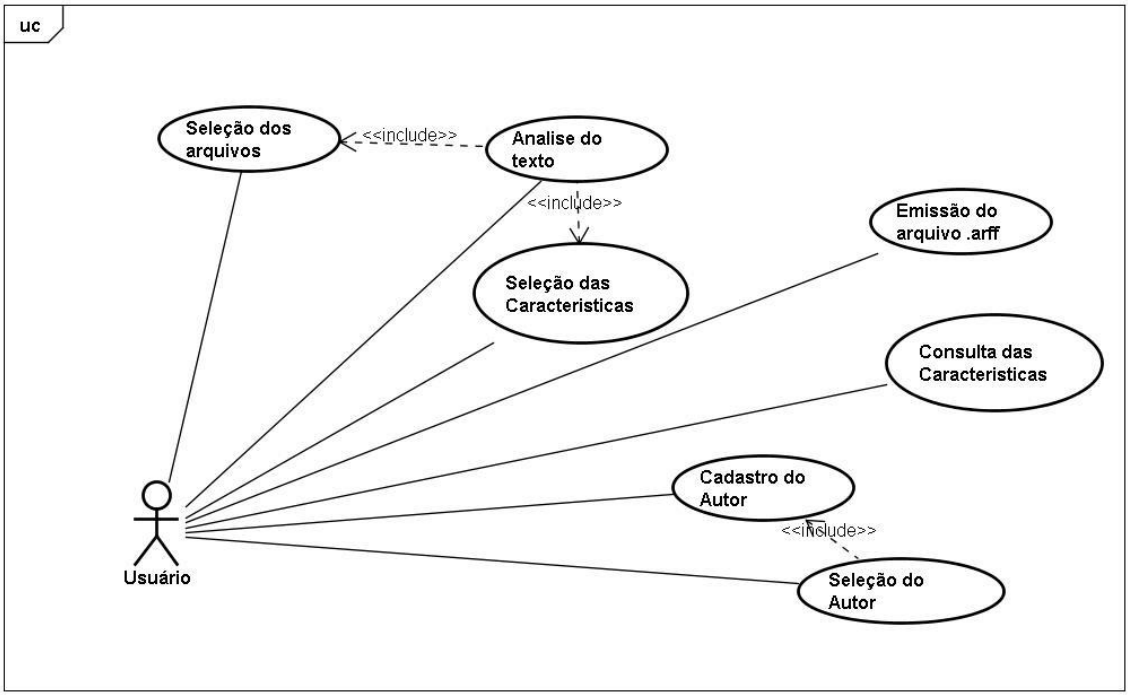


Figura 9 - Diagrama de casos de uso do sistema
Fonte: Elaborado pelo Autor

O Quadro 2 apresenta o caso de uso de alto-nível relacionado ao caso de uso “Cadastramento de autores”.

Caso de uso: Cadastramento de autores
O usuário acessa a tela do *software* e clica no respectivo item do menu principal "Cadastros" e seleciona o item "Autor". Na tela de cadastro do autor deve se direcionar para "Novo", digitar o nome do autor e, em seguida, clicar no botão “Salvar” ou "Salvar e Fechar.

Quadro 2 - Caso de uso de alto-nível: Cadastramento de autores
Fonte: Elaborado pelo Autor.

No Quadro 3 é apresentado o caso de uso de alto-nível para o caso de uso “Seleção de autor”.

Caso de uso: Seleção de autor
O usuário acessa a tela do *software* e clica no respectivo item do menu principal "Cadastros" e seleciona o item "Processamento". Na tela de processamento de texto deve se clicar no botão "Selecionar", o qual exibirá a listagem de autores cadastrados. O usuário clica sobre o nome do autor desejado e então no botão "Selecionar".

Quadro 3 - Caso de uso de alto-nível: Seleção de autor.
Fonte: Elaborado pelo Autor.

O Quadro 4 exibe o caso de uso de alto-nível para o caso de uso “Consulta de características”.

Caso de uso: Consulta de características
O usuário acessa a tela do *software* e clica no respectivo item do menu principal "Cadastros" e seleciona o item "Texto", onde será exibida uma listagem dos textos processados, o usuário deve selecionar o texto a ser verificado e clicar no botão "Detalhes", que mostrará em forma de abas divididas pelos níveis estabelecidos as características que foram extraídas a partir das previamente selecionadas de maneira que o usuário possa verificar a palavra e a quantidade e ainda caso seja necessário clicar no botão "Excluir" para eliminar alguma característica.

Quadro 4 - Caso de uso de alto-nível: Consulta de características.
Fonte: Elaborado pelo Autor.

No Quadro 5 é apresentado o caso de uso de alto-nível para o caso de uso “Geração do arquivo ARFF”.

Caso de uso: Geração do arquivo ARFF
O usuário acessa a tela do *software* e clica no respectivo item do menu principal "Arquivos" e seleciona o item "Exportar". Será exibida a tela para gerar o arquivo onde deve-se selecionar as características a serem exportadas, o tipo de frequência - relativa ou absoluta - e os autores que comporão o arquivo e em seguida clicar no botão "Exportar" que abrirá a caixa de diálogo solicitando onde salvar o arquivo.

Quadro 5 - Caso de uso de alto-nível: Geração do arquivo ARFF.
Fonte: Elaborado pelo Autor.

No Quadro 6 é exibido o caso de uso de alto nível representando o caso de uso “Seleção das características sintáticas”.

Caso de uso: Seleção das características sintáticas
O usuário acessa a tela do *software* e clica no respectivo item do menu principal "Cadastros" e seleciona o item "Processamento" onde abrirá a tela na qual é efetuado o processamento do texto onde aparecem as características que podem ser extraídas dos textos e são divididas em 3 níveis dentro dos quais temos as características sintáticas que podem ser identificadas nos textos.

Quadro 6 - Caso de uso de alto-nível: Seleção das características sintáticas.
Fonte: Elaborado pelo Autor.

O caso de uso de alto-nível apresentado no Quadro 7 representa o caso de uso “Seleção dos arquivos”.

Caso de uso: Seleção dos arquivos
O usuário acessa a tela do *software* e clica no respectivo item do menu principal "Cadastros" e seleciona o item "Processamento". Nessa tela, o usuário clica no botão "Escolher Arquivos", onde será apresentada a caixa de diálogo, na qual o usuário deve escolher os arquivos em seguida clicar no botão "OK". Os arquivos selecionados são apresentados na área localizada abaixo do botão.

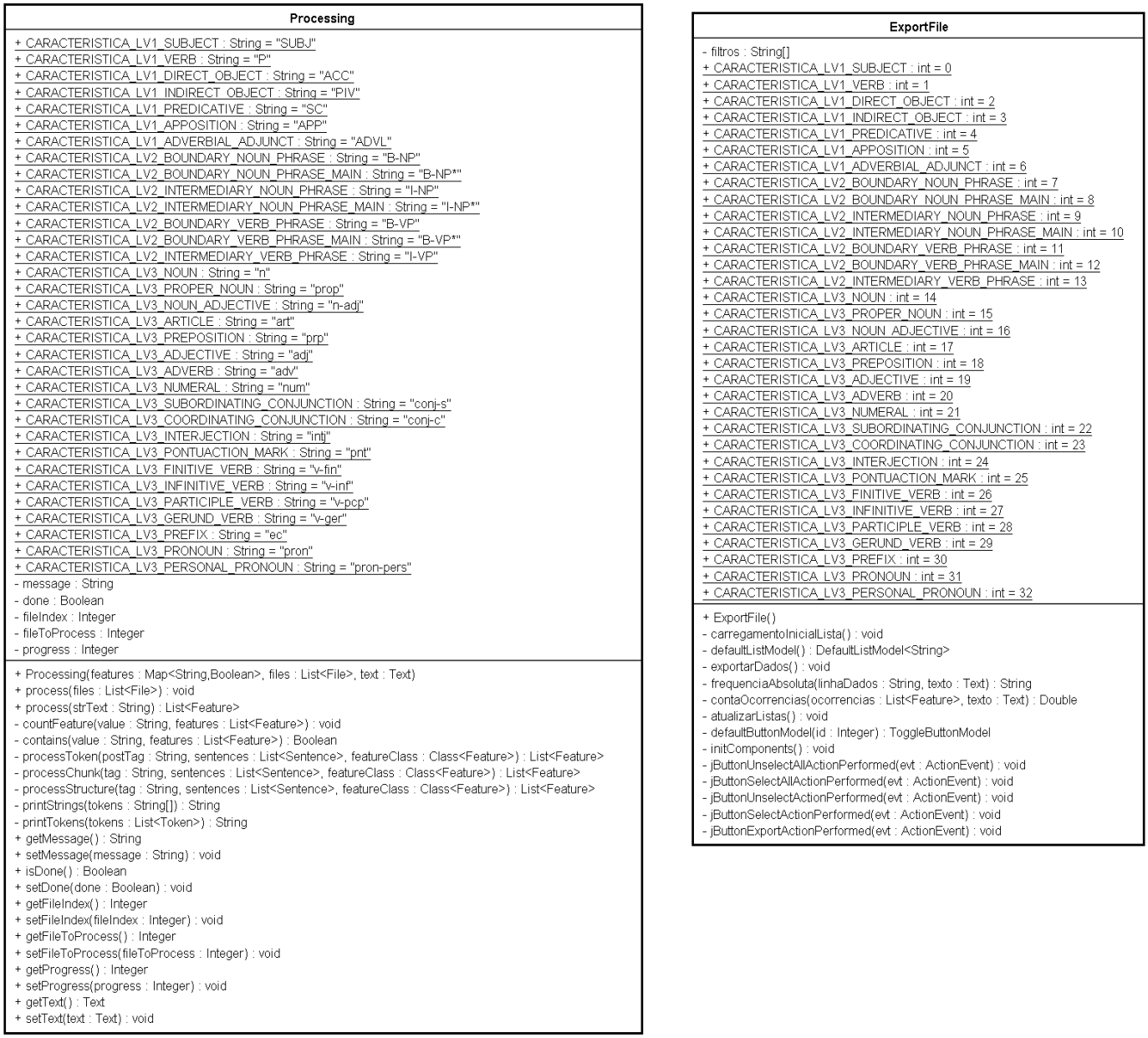
Quadro 7 - Caso de uso de alto-nível: Seleção dos arquivos
Fonte: Elaborado pelo Autor.

No Quadro 8 é exibido o caso de uso de alto-nível referente ao caso de uso “Análise do texto”.

Caso de uso: Análise do texto
O usuário acessa a tela do *software* e clica no respectivo item do menu principal "Cadastros" e seleciona o item "Processamento". Nessa tela, o usuário clica no botão "Escolher Arquivos", onde será apresentada a caixa de diálogo, na qual o usuário deve escolher os arquivos, em seguida, clicar no botão "OK". Os arquivos selecionados são apresentados na área localizada abaixo do botão. Após deverá selecionar as características a serem extraídas a partir da seleção das caixas de seleção, que identificam e após clicar no botão "Processar"

Quadro 8 - Caso de uso de alto-nível: Análise do texto.
Fonte: Elaborado pelo Autor.

Definidas as funcionalidades do *software*, elaborou-se um diagrama de classes representando as entidades que representam as informações principais que são armazenadas na base de dados conforme apresentado na Figura 10.



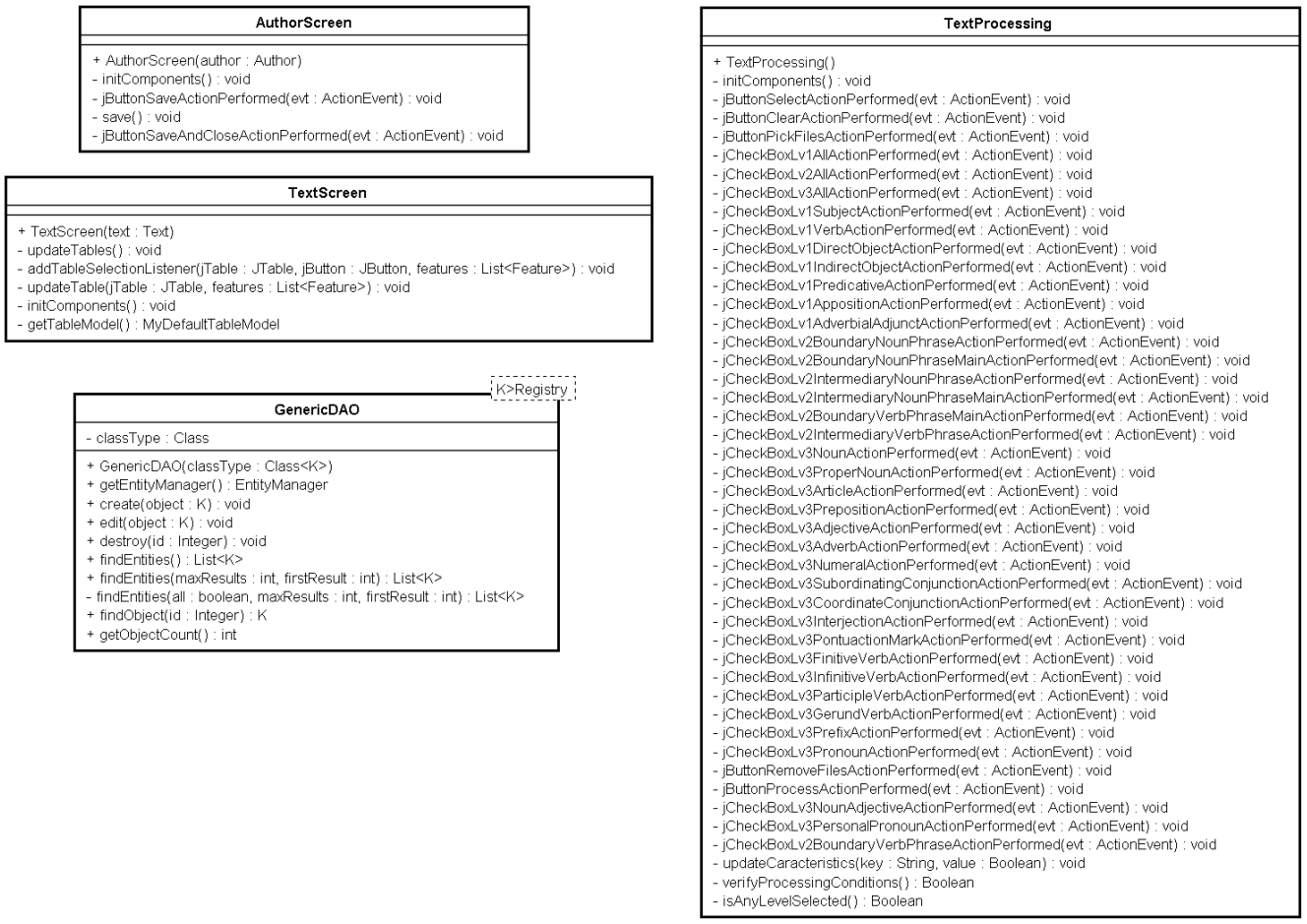


Figura 10 - Diagrama de classes das principais classes da aplicação
Fonte: Elaborado pelo Autor

Foram omitidas no diagrama exibido na Figura 10, as demais classes responsáveis por funções de manipulação e persistência de dados e classes da interface gráfica da aplicação.

Após o desenvolvimento do diagrama de classes foi elaborado o diagrama de entidade e relacionamento, sendo que o completo encontra-se como apêndice 1 e uma parte dele está apresentado na Figura 11 para posterior criação da base de dados.

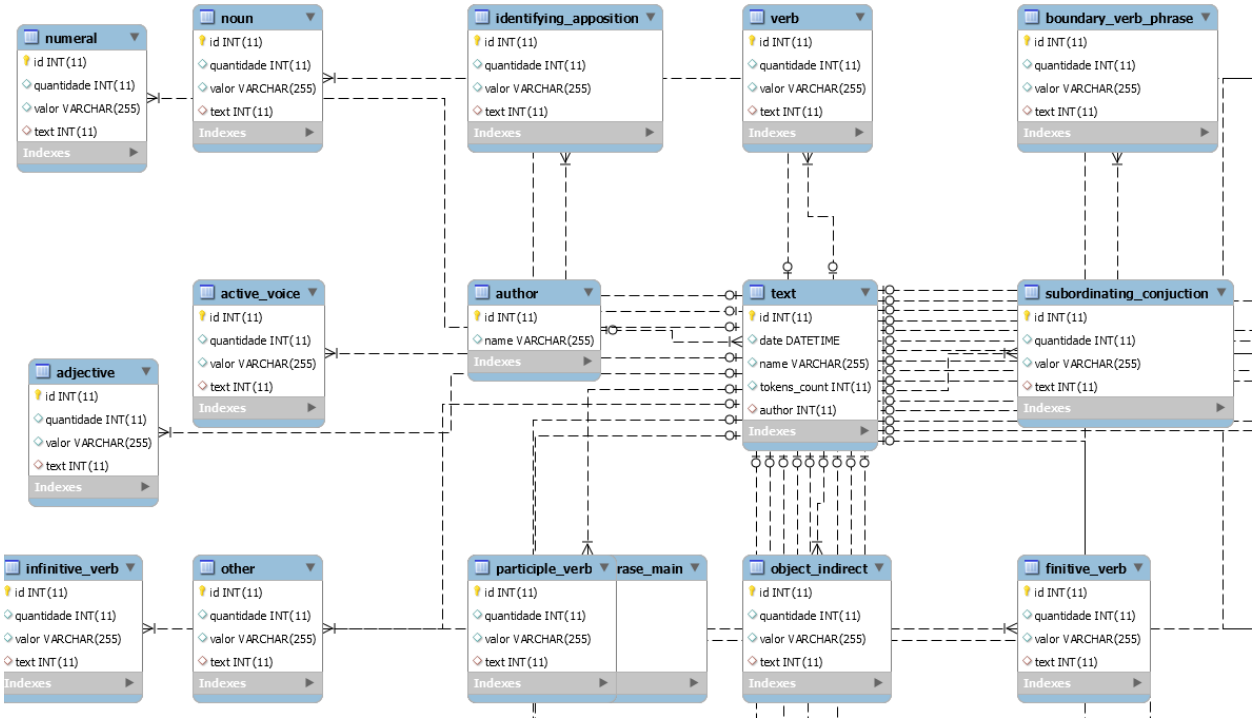


Figura 11 - Diagrama de entidades e relacionamentos: Algumas tabelas
Fonte: Elaborado pelo Autor

3.7 IMPLEMENTAÇÃO DO SISTEMA

Nesta seção são apresentados detalhes de implementação do *software*. Tendo em vista as características de uso justifica-se, a princípio, o desenvolvimento para a plataforma *desktop*, de forma a facilitar o processo de distribuição, instalação e uso do *software* concebido, e também suprimir eventuais despesas com registro de domínio e hospedagem (caso a opção fosse desenvolver uma aplicação para *internet*) ou mesmo evitar o eventual trabalho de configuração de um servidor *web* local, além das preocupações extras com segurança inerentes às aplicações distribuídas.

No desenvolvimento do *software* foi utilizada a linguagem Java em sua versão Standard Edition, seu uso, nessa versão, é voltado para desenvolvimento de aplicações *desktop*. É possível utilizar, em aplicações Java Standard Edition (J2SE), o pacote Java Swing, que possui por sua vez uma série de componentes visuais para aplicação no processo de desenvolvimento de interface gráfica.

A IDE Netbeans, escolhida como ferramenta de desenvolvimento, apresenta uma série de funcionalidades comuns às suas concorrentes, porém, destaca-se destas por dispor de recurso

de desenvolvimento visual de interface gráfica em aplicações J2SE, o que implica em um ganho significativo de tempo de desenvolvimento do *software* em questão.

Na Listagem 1 temos a codificação da classe responsável pela inicialização do sistema. É possível verificar que é designada a essa classe tornar visível a tela principal com os menus do *software*.

```
01 public class Main {
02
03    public static ComponentFactory COMPONENT_FACTORY = ComponentFactory.create(new Locale("pt", "BR"));
04    public static Analyzer ANALYSER = Main.COMPONENT_FACTORY.createPipe();
05
06    public static void main(String[] args) {
07       /* Cria e exibe a tela inicial da aplicação */
08       java.awt.EventQueue.invokeLater(new Runnable() {
09          public void run() {
10             new MainScreen().setVisible(true);
11          }
12       });
13    }
14 }
```

Listagem 1 - Codificação da classe principal
Fonte: Elaborado pelo Autor.

No código exibido na Listagem 1 verifica-se a utilização da classe *ComponentFactory* na criação de dois objetos estáticos, sendo um objeto do tipo *ComponentFactory* e outro do tipo *Analyser*.

Para a criação de uma instância da classe *ComponentFactory* deve-se utilizar o método estático implementado nesta classe chamado “create”, o qual espera a passagem de um objeto por parâmetro, do tipo *Locale*, com informações sobre o tipo de linguagem que será posteriormente utilizada nas análises sintáticas de textos, conforme pode ser visto na linha 03 da Listagem 1. Isso deve-se ao fato de que a classe *ComponentFactory* pode ser configurada para trabalhar com outras linguagens além do português do Brasil.

Ainda é possível verificar na linha 04 da Listagem 1 a criação de uma instância de objeto do tipo *Analyser* através da invocação do método “createPipe” da instância da classe *ComponentFactory* anteriormente criada. Com isso obtém-se um objeto do tipo *Analyser* configurado para trabalhar com a linguagem português do Brasil, como consequência da configuração, relatada anteriormente, do objeto da classe *ComponentFactory*.

No primeiro momento em que o *software* é executado, o banco de dados encontra-se vazio, sendo assim, faz-se necessário o cadastramento de autores, os quais posteriormente podem ser utilizados para o processamento de textos de sua autoria. Para isso foi implementada a rotina demonstrada na Listagem 2. Fica claro o uso da biblioteca Swing para desenvolvimento da interface, sendo que no código listado constam classes deste pacote como *JInternalFrame*.

No código da listagem em questão é possível verificar a utilização de uma classe de manipulação de dados que implementa o padrão de projeto (*Design Pattern*) DAO (*Data Access Object*).

```
01 public class AuthorScreen extends javax.swing.JInternalFrame {
02
03    private Author author;
04
05    public AuthorScreen(Author author) {
06       initComponents();
07       this.author = author;
08       this.jTextFieldName.setText(this.author.getName());
09    }
10
11    /**
12     * Este método é responsável pela criação de instâncias de todos os componentes
13     * visuais integrantes da interface gráfica, tais como paineis, botões, caixas de seleção, etc.
14     */
15    private void initComponents() {
16    // CODIFICAÇÃO SUPRIMIDA
17    }
18
19    private void jButtonSaveActionPerformed(java.awt.event.ActionEvent evt) {
20       this.save();
21    }
22
23    private void save() {
24       this.author.setName(this.jTextFieldName.getText().trim().toUpperCase());
25
26       if ((this.author.getId() instanceof Integer) && (this.author.getId() > 0)) {
27          try {
28             new AuthorDAO().edit(this.author);
29          } catch (Exception ex) {
30             Logger.getLogger(AuthorScreen.class.getName()).log(Level.SEVERE, null, ex);
31          }
32       } else {
33          new AuthorDAO().create(this.author);
34       }
35    }
36
37    private void jButtonSaveAndCloseActionPerformed(java.awt.event.ActionEvent evt) {
38       this.save();
39       this.dispose();
40    }
41 }
```

Listagem 2 - Codificação do cadastro de autor
Fonte: Elaborado pelo Autor.

Na sequência, o usuário pode acessar à tela de processamento que é utilizada para seleção de arquivo de texto e autores e escolha das características sintáticas a serem extraídas do texto. As características sintáticas foram divididas em três níveis com suas respectivas classes sintáticas conforme dispõem a codificação da biblioteca CoGrOO, cujas funcionalidades são utilizadas no *software* desenvolvido para colocar os "rótulos" conforme a função desempenhada por uma palavra ou fragmento do texto.

Para seleção do arquivo de texto a ser processado foi utilizada o componente *JFileChooser* da biblioteca Swing que possibilita a seleção de arquivos contidos em discos de armazenamento conectados ao computador e isso é demonstrado na Listagem 3.

```
01 private void jButtonPickFilesActionPerformed(java.awt.event.ActionEvent evt) {
02       JFileChooser jFileChooser = new JFileChooser();
03
04       FileFilter fileFilter = new FileFilter() {
05
06          @Override
07          public boolean accept(File f) {
08             if (f.isDirectory()) {
09                return true;
10             }
11             return f.getName().endsWith("txt");
12          }
13
14          @Override
15          public String getDescription() {
16             return "Arquivos de texto (*.txt)";
17          }
18       };
19
20       jFileChooser.setFileFilter(fileFilter);
21       jFileChooser.setMultiSelectionEnabled(true);
22       jFileChooser.showOpenDialog(MainScreen.jDesktopPaneMain);
23
24       this.files = (Arrays.asList(jFileChooser.getSelectedFiles()));
25
26       if (this.files != null) {
27          ((DefaultListModel) this.jListFiles.getModel()).removeAllElements();
28          for (File file : files) {
29             ((DefaultListModel) this.jListFiles.getModel()).addElement(file.getName());
30          }
31       }
32       this.jButtonProcess.setEnabled(this.verifyProcessingConditions());
33 }
```

Listagem 3 - Codificação da árvore de arquivos
Fonte: Elaborado pelo Autor.

Para que a codificação ficasse dentro dos padrões de orientação a objetos e visando a redução do número de linhas de código os atributos que contém os "rótulos" das características fornecidas pela biblioteca CoGrOO foram inicializadas como atributos globais, estáticos e finais da classe Processing, caracterizando constantes, pois assim durante a rotulação do texto e extração da característica selecionada somente é feita a comparação com a constante tornando a execução mais eficaz. A Listagem 4 exibe a criação dos atributos constantes sendo valorizados com os rótulos utilizados pela biblioteca CoGrOO.

```
01 public class Processing {
02
03    public static final String CARACTERISTICA_LV1_SUBJECT = "SUBJ";
04    public static final String CARACTERISTICA_LV1_VERB = "P";
05    public static final String CARACTERISTICA_LV1_DIRECT_OBJECT = "ACC";
06    public static final String CARACTERISTICA_LV1_INDIRECT_OBJECT = "PIV";
07    public static final String CARACTERISTICA_LV1_PREDICATIVE = "SC";
```

```
08    public static final String CARACTERISTICA_LV1_APPOSITION = "APP";
09    public static final String CARACTERISTICA_LV1_ADVERBIAL_ADJUNCT = "ADVL";
10
11    public static final String CARACTERISTICA_LV2_BOUNDARY_NOUN_PHRASE = "B-NP";
12    public static final String CARACTERISTICA_LV2_BOUNDARY_NOUN_PHRASE_MAIN = "B-NP*";
13    public static final String CARACTERISTICA_LV2_INTERMEDIARY_NOUN_PHRASE = "I-NP";
14    public static final String CARACTERISTICA_LV2_INTERMEDIARY_NOUN_PHRASE_MAIN = "I-NP*";
15    public static final String CARACTERISTICA_LV2_BOUNDARY_VERB_PHRASE = "B-VP";
16    public static final String CARACTERISTICA_LV2_BOUNDARY_VERB_PHRASE_MAIN = "B-VP*";
17    public static final String CARACTERISTICA_LV2_INTERMEDIARY_VERB_PHRASE = "I-VP";
18
19    public static final String CARACTERISTICA_LV3_NOUN = "n";
20    public static final String CARACTERISTICA_LV3_PROPER_NOUN = "prop";
21    public static final String CARACTERISTICA_LV3_NOUN_ADJECTIVE = "n-adj";
22    public static final String CARACTERISTICA_LV3_ARTICLE = "art";
23    public static final String CARACTERISTICA_LV3_PREPOSITION = "prp";
24    public static final String CARACTERISTICA_LV3_ADJECTIVE = "adj";
25    public static final String CARACTERISTICA_LV3_ADVERB = "adv";
26    public static final String CARACTERISTICA_LV3_NUMERAL = "num";
27    public static final String CARACTERISTICA_LV3_SUBORDINATING_CONJUNCTION = "conj-s";
28    public static final String CARACTERISTICA_LV3_COORDINATING_CONJUNCTION = "conj-c";
29    public static final String CARACTERISTICA_LV3_INTERJECTION = "intj";
30    public static final String CARACTERISTICA_LV3_PONTUACTION_MARK = "pnt";
31    public static final String CARACTERISTICA_LV3_FINITIVE_VERB = "v-fin";
32    public static final String CARACTERISTICA_LV3_INFINITIVE_VERB = "v-inf";
33    public static final String CARACTERISTICA_LV3_PARTICIPLE_VERB = "v-pcp";
34    public static final String CARACTERISTICA_LV3_GERUND_VERB = "v-ger";
35    public static final String CARACTERISTICA_LV3_PREFIX = "ec";
36    public static final String CARACTERISTICA_LV3_PRONOUN = "pron";
37    public static final String CARACTERISTICA_LV3_PERSONAL_PRONOUN = "pron-pers";
38
39  // demais variáveis e métodos
40 }
```

Listagem 4 - Codificação das constantes
Fonte: Elaborado pelo Autor.

Após ter selecionado os arquivos de texto que devem ser processando, indicar as características a serem extraídas e autor é possível iniciar o processamento, onde estes textos serão analisados e as informações extraídas de acordo com a parametrização indicada. Como resposta visual ao processamento do texto o sistema exibe mensagens atualizadas sobre o andamento do processamento em uma área específica da interface gráfica do *software*. A Listagem 5, exibe a codificação da funcionalidade desenvolvida para esse fim.

```
01 private void jButtonProcessActionPerformed(java.awt.event.ActionEvent evt) {
02  new Thread() {
03
04     @Override
05     public void run() {
06        jButtonProcess.setEnabled(false);
07        jButtonClear.setEnabled(false);
08        jButtonRemoveFiles.setEnabled(false);
09        jButtonPickFiles.setEnabled(false);
10        jButtonSelect.setEnabled(false);
11
12        Processing processing = new Processing(caracteristics, files, text);
13        processing.process(files);
14        jLabelStatus.setText("Processando...");
15
```

```
16          while (!processing.isDone()) {
17              try {
18                  jLabelStatus.setText(processing.getMessage());
19                  sleep(1);
20              } catch (InterruptedException ex) {
21                  Logger.getLogger(TextProcessing.class.getName()).log(Level.SEVERE, null, ex);
22              }
23
24          }
25
26          jButtonProcess.setEnabled(true);
27          jButtonClear.setEnabled(true);
28          jButtonRemoveFiles.setEnabled(true);
29          jButtonPickFiles.setEnabled(true);
30          jButtonSelect.setEnabled(true);
31
32          jLabelStatus.setText("Processamento concluído.");
33      }
34  }.start();
35  }
```

Listagem 5 - Codificação do botão "Processamento"
Fonte: Elaborado pelo Autor.

É possível notar, na Listagem 5, a utilização de uma *thread* para criação de um processo exclusivo para processamento e exibição de mensagens de atualização sobre o andamento do processamento no componente destinado para esse fim. A necessidade de trabalhar com processos separados simultaneamente aparece frequentemente na computação. Para vários programas distintos sendo executados simultaneamente, normalmente o próprio sistema operacional gerencia isso através de vários processos em paralelo. Em Java, usamos a classe *Thread* do pacote java.lang para criar linhas de execução paralelas. No exemplo contido na Listagem 5 todo o código contido no método “run” será delegado ao novo processo criado e executado de forma paralela ao processo existente da aplicação.

As formas com que o sistema permite trabalhar com as informações extraídas dos textos processados são: a consulta direta pela interface do *software*, onde é possível verificar os elementos extraídos e a contagem de ocorrências, ou a exportação das informações para um arquivo ARFF.

O componente *JTable* foi utilizado nas telas de consulta direta de informações. Este componente faz parte da biblioteca Swing e pode ser utilizado para exibição de dados de forma tabular.

Para criação de uma instância da classe *JTable* é possível a passagem por parâmetro para o construtor desta classe de um objeto de classe que implemente a interface *TableModel*. Há no próprio pacote de classes do Java classes que já implementam esta interface, como por exemplo as classes *AbstractTableModel* e *DefaultTableModel*. Um fator importante que deve ser destacado em relação a isso é que o objeto do tipo *TableModel* em questão servirá como

provedor de dados para que o componente *JTable* seja populado, posteriormente este pode ser utilizado para, por exemplo, obter um objeto original relativo a uma célula ou linha selecionada, para que se faça possível uma atualização, exclusão ou simples visualização de detalhes não exibidos do objeto.

A interface *TableModel* obriga a implementação de um método chamado "*getValueAt*", sendo que este método recebe por parâmetro as "coordenadas" de linha e coluna, e retorna o valor contido na célula em questão. Por necessidade de correção de uma eventual análise inválida o sistema permite a exclusão de informações extraídas através do processamento de textos, sendo assim, tornou-se ideal a implementação de uma classe própria que atenda ao modelo proposto pela interface *TableModel* de forma com que o método "*getValueAt*" retornasse o objeto representado por uma linha exibida no componente *JTable*, sendo assim, à classe criada deu-se o nome de *MyDefaultTableModel*.

A classe *MyDefaultTableModel*, conforme exibido na Listagem 6 herda todos os métodos da classe *DefaultTableModel* de forma que seus objetos podem ser então utilizados como provedores de dados para componentes *JTable*. A opção pela herança da classe DefaultTableModel deu-se pelo fato de que tornaria possível a sobrescrita de determinados métodos, criação de novos métodos, sem a necessidade de implementação de todos os métodos previstos na interface *TableModel* já implementados na classe *DefaultTableModel*. Outro detalhe no desenvolvimento da classe *MyDefaultTableModel* foi a utilização do recurso chamado *generics*, onde é possível definir uma abstração do tipo de dado que será retornado pelo método "*getValueAt*" dependendo do objeto repassado na criação da instância de seu objeto.

```
01 public abstract class MyDefaultTableModel<K> extends DefaultTableModel {
02
03    public abstract void addRow(K objeto);
04
05    public void addRow(K objeto, Boolean selecionada) {
06    }
07
08    @Override
09    public void addRow(Object... valores) {
10       super.addRow(valores);
11    }
12
13    public void setColumnIdentifiers(String... identificadores) {
14       super.setColumnIdentifiers(identificadores);
15    }
16 }
```

Listagem 6 - Codificação da classe MyDefaultTableModel
Fonte: Elaborado pelo Autor.

Para geração do arquivo além das demais seleções - autor, características - é necessário selecionar o tipo de frequência a ser adicionado ao arquivo, seja ela relativa, onde o resultado é obtido da divisão entre a frequência - o valor que é observado na população - e a quantidade de elementos da população, ou absoluta onde é representado o valor absoluto observado na população. A Listagem 7 mostra o método responsável pelo cálculo da frequência de acordo com o tipo selecionado.

```
01 private Double contaOcorrencias(List<? extends Feature> ocorrencias, Text texto) {
02  Double conta = 0.0;
03  for (Feature ocorrencia : ocorrencias) {
04     conta += ocorrencia.getQuantidade();
05  }
06
07  if (this.jRadioButtonRelativeType.isSelected()) {
08     conta = conta / texto.getTokensCount();
09  }
10
11  return conta;
12 }
```

Listagem 7 - Método de cálculo de frequência
Fonte: Elaborado pelo Autor.

Na Listagem 8 é demonstrado o método responsável pela geração de arquivo no formato ARFF, contendo as informações extraídas de textos já processados de acordo com um ou mais autores selecionados.

As informações para geração do arquivo são primeiramente armazenadas em uma variável do tipo *List*, nesta variável, além das linhas de cabeçalho onde são incluídas informações sobre o tipo de frequência, características sintáticas selecionadas e identificação dos autores selecionados, também são armazenados os dados dos textos processados. Depois que as informações necessárias para geração do arquivo são preparadas e armazenadas na variável o *software* faz uso do componente *JFileChooser*, para que seja exibida ao usuário uma janela onde é possível selecionar o local de destino para geração do arquivo e informar o nome do arquivo a ser gerado.

```
01 private void exportarDados() {
02  List<String> linhas = new ArrayList<>();
03
04  // INICIO - GERAÇÃO DO CABEÇALHO
05  if (this.jRadioButtonRelativeType.isSelected()) {
06     linhas.add("@Relation Relativa");
07  } else {
08     linhas.add("@Relation Absoluta");
09  }
10
11  for (Integer identificador : filtrosSelecionados) {
12     linhas.add("@Attribute " + filtros[identificador] + " Integer");
13  }
14
15  linhas.add("");
```

```
16  String autores = "@Attribute Class {";
17
18  for (Author autor : autoresSelecionados) {
19      autores += "A" + (this.autoresSelecionados.indexOf(autor) + 1) + (((autoresSelecionados.indexOf(autor) + 1) ==
autoresSelecionados.size()) ? "" : ",");
20  }
21  autores += "}";
22
23  linhas.add(autores);
24  linhas.add("");
25  linhas.add("@data");
26  // FINAL - GERAÇÃO DE CABECALHO
27  //INICIO - GERAÇÃO DE DADOS
28  String linhaDados = "";
29  for (Author autor : autoresSelecionados) {
30      for (Text texto : autor.getTexts()) {
31          linhaDados = "";
32          linhaDados = frequenciaAbsoluta(linhaDados, texto);
33          linhas.add(linhaDados + ("A" + (autoresSelecionados.indexOf(autor) + 1)));
34      }
35  }
36
37  // FINAL - GERAÇÃO DE DADOS
38
39  // INICIO - GERAÇÃO ARQUIVO
40  try {
41      JFileChooser jFileChooser = new JFileChooser();
42      jFileChooser.setFileFilter(new FileNameExtensionFilter("Arquivos ARFF", "arff"));
43      jFileChooser.showSaveDialog(this);
44      File arquivo = new File(jFileChooser.getSelectedFile() + ".arff");
45      if (arquivo.createNewFile()) {
46          try (BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(arquivo))) {
47              for (String linha : linhas) {
48                  bufferedWriter.write(linha);
49                  bufferedWriter.newLine();
50              }
51          }
52      }
53  } catch (IOException ex) {
54      Logger.getLogger(ExportFile.class.getName()).log(Level.SEVERE, null, ex);
55  }
56  // FINAL - GERAÇÃO ARQUIVO
57 }
```

Listagem 8 - Método de geração de arquivo ARFF
Fonte: Elaborado pelo Autor.

O *framework* de persistência Hibernate, é utilizado em todas as rotinas de persistência e recuperação de informações da base de dados atualmente. Por não possuir uma base de dados extensa é impossível avaliar como definitiva a utilização de tal *framework* sendo assim necessários testes mais aprofundados utilizando uma base de dados com um número de registros muito maior do que a atual para verificação da latência das requisições.

Na Listagem 9 é mostrada a codificação da classe Author, onde é possível identificar as anotações de persistência, sendo estas necessárias para manipulação por parte do *framework* Hibernate.

```
01 @Entity
02 @Table(name = "author")
03 public class Author extends Registry {
04     @Column
05     private String name;
06     @OneToMany(mappedBy = "author")
07     private List<Text> texts;
08 
09  // Implementação de getters e setters...
10 }
```

Listagem 9 - Codificação da entidade "Author"
Fonte: Elaborado pelo Autor.

Todas as transações com a base de dados são executadas com a utilização de classes que implementam o padrão de projeto DAO. Na aplicação em questão foi desenvolvida uma classe abstrata genérica de manipulação onde são executadas as funções comuns a todas as classes DAO. Esta classe abstrata é estendida pelas classes DAO relacionadas a cada entidade de persistência, e nessas classes DAO especializadas são desenvolvidas somente as rotinas específicas de cada caso. Na Listagem 10 é demonstrada a codificação da classe genérica GenericDAO, que é uma classe padrão das demais, respeitando assim, a orientação a objetos.

```
01 public class GenericDAO<K extends Registry> implements Serializable {
02 
03     private EntityManagerFactory entityManagerFactory;
04     private EntityTransaction entityTransaction;
05     private EntityManager entityManager;
06     private Class classType;
07 
08     public GenericDAO(Class<K> classType) {
09         this.entityManagerFactory = JPAFactory.getEntityManagerFactory();
10         this.entityManager = this.entityManagerFactory.createEntityManager();
11         this.classType = classType;
12     }
13 
14     public EntityManager getEntityManager() {
15         return entityManagerFactory.createEntityManager();
16     }
17 
18     public void create(K object) {
19         this.entityManager.getTransaction().begin();
20         this.entityManager.persist(object);
21         this.entityManager.getTransaction().commit();
22     }
23 
24     public void edit(K object) throws NonexistentEntityException, Exception {
25         try {
26             this.entityManager.getTransaction().begin();
27             object = this.entityManager.merge(object);
28             this.entityManager.getTransaction().commit();
29         } catch (Exception ex) {
30             String msg = ex.getLocalizedMessage();
31             if (msg == null || msg.length() == 0) {
32                 Integer id = object.getId();
33                 if (findObject(id) == null) {
34                     throw new NonexistentEntityException("The object with id " + id + " no longer exists.");
35                 }
36             }
37             throw ex;
```

```
38      }
39  }
40
41  public void destroy(Integer id) throws NonexistentEntityException {
42      this.entityManager.getTransaction().begin();
43      K object;
44      try {
45          object = (K) this.entityManager.getReference(classType, id);
46          object.getId();
47      } catch (EntityNotFoundException enfe) {
48          throw new NonexistentEntityException("The object with id " + id + " no longer exists.", enfe);
49      }
50      this.entityManager.remove(object);
51      this.entityManager.getTransaction().commit();
52  }
53
54  public List<K> findEntities() {
55      return this.findEntities(true, -1, -1);
56  }
57
58  public List<K> findEntities(int maxResults, int firstResult) {
59      return this.findEntities(false, maxResults, firstResult);
60  }
61
62  private List<K> findEntities(boolean all, int maxResults, int firstResult) {
63      Query query = this.entityManager.createQuery("FROM " + classType.getCanonicalName() + "");
64      if (!all) {
65          query.setMaxResults(maxResults);
66          query.setFirstResult(firstResult);
67      }
68      return query.getResultList();
69  }
70
71  public K findObject(Integer id) {
72      return (K) this.entityManager.find(classType, id);
73  }
74
75  public int getObjectCount() {
76      CriteriaQuery cq = this.entityManager.getCriteriaBuilder().createQuery();
77      Root<K> rt = cq.from(classType);
78      cq.select(this.entityManager.getCriteriaBuilder().count(rt));
79      Query q = this.entityManager.createQuery(cq);
80      return ((Long) q.getSingleResult()).intValue();
81  }
82 }
```

Listagem 10 - Classe de manipulação de dados GenericDAO
Fonte: Elaborado pelo Autor.

3.8 O *CORPUS* DE TEXTOS LITERÁRIO

O *corpus* de textos literários foi formado a partir da coleta de amostras de textos de autores da literatura brasileira de diferentes períodos literários que encontravam-se disponíveis para leitura e reprodução, ou seja, sem ferir direitos autorais, em dois sítios da Internet: Domínio Público[1] e Projeto Gutemberg[2]. A construção desse corpus se deu como forma de auxiliar a manipulação do software, o planejamento das aulas e também auxiliar na integração dos

[1] http://www.dominiopublico.gov.br/pesquisa/PesquisaObraForm.jsp
[2] https://www.gutenberg.org/wiki/PT_Principal

conteúdos de gramática e literatura. Para tanto esse corpus será disponibilizado juntamente com o software em um domínio da Internet para acesso de toda a comunidade.

3.9 CONSIDERAÇÕES FINAIS

A observação de metodologias de desenvolvimento de software atualmente aplicados e a escolha de tecnologias e ferramentas influenciam diretamente no planejamento, delimitação e características funcionais do software desenvolvido.

O levantamento e documentação de requisitos a e representação destes por meio de diagramas UML além do desenvolvimento de diagramas complementares podem ser utilizados para auxiliar o entendimento do projeto do sistema por parte dos *stakeholders*, além de manter clara a delimitação e funcionalidades do sistema para uma equipe de desenvolvimento.

A opção por determinadas ferramentas e tecnologias de desenvolvimento devem levar em consideração fatores, como o ambiente de execução da aplicação, a compatibilidade de integração entre estas, sendo que escolhas equivocadas podem ocasionar o desenvolvimento de um software com desempenho ou usabilidade insatisfatórios ou até mesmo inviabilizar o desenvolvimento.

O software desenvolvido descrito neste trabalho possui como principal funcionalidade a extração de características sintáticas de textos de determinados autores e a disponibilização dos dados obtidos, tanto para visualização e análise por parte do usuário na própria interface gráfica do software, que se constitui em sua principal aplicação.

Tendo em vista seu uso como apoio ao aprendizado da gramática na disciplina de Língua Portuguesa e Literatura Brasileira leva a visualização das características e consequentemente fomentar a discussão em sala de aula, além disso, agregou-se a função de exportação destes dados em arquivos ARFF (*Attribute-Relation File Format*) devido as lacunas encontradas nas pesquisas cientificas em relação a análise de documentos. A funcionalidade de análise e identificação de características sintáticas foi viabilizada pela utilização da biblioteca de correção ortográfica CoGrOO, a qual possui algoritmos de rotulagem destas características.

Após o processo de extração das informações, estas são armazenadas em uma base de dados de forma a permitir posteriormente uma leitura por parte do sistema de forma agilizada, sendo que, o processo de análise do texto por parte da biblioteca CoGrOO tem um gasto computacional considerável, sendo assim possível a utilização dos dados extraídos para ensino, análise ou geração de arquivo ARFF.

4 RESULTADOS E DISCUSSÃO

Este capítulo apresenta o que foi obtido como resultado do trabalho, que é um *software* desenvolvido para extração de características sintáticas de textos em língua portuguesa brasileira. São apresentados as principais interfaces do *software* de maneira que demonstrem sua utilização como software de apoio ao aprendizado.

4.1 APRESENTAÇÃO DO SOFTWARE

O *software* descrito neste trabalho objetiva disponibilizar, por meio de uma interface gráfica, um mecanismo de automação para de extração de características sintáticas de textos e posterior disponibilização das informações em arquivos ARFF, sendo que a forma padrão, até então, para desempenho de tais tarefas é manual e subjetiva.

O *software* possibilita, que seja executada a seleção e processamento de múltiplos arquivos de texto de forma automatizada, cabendo ao usuário, apenas, o cadastramento de autores, a seleção dos arquivos de texto e as características a serem extraídas, permitindo que as particularidades da língua portuguesa brasileira sejam analisadas, fornecendo maior precisão e rapidez no processo da extração.

O *software* será disponibilizado à toda comunidade acadêmica e científica, objetivando auxiliar no processo ensino aprendizagem da sintática na língua portuguesa. Em correlato, propiciar uma maior quantidade de estudos na área de estilometria forense. Enfim, nas próximas seções são detalhadas informações essenciais que foram utilizadas no desenvolvimento do *software*.

4.2 DESCRIÇÃO DO SOFTWARE

A seguir são apresentadas as telas das principais funcionalidades do *software* exibindo a disposição de componentes, também são comentadas as formas de interação inicialmente propostas para as funcionalidades chave.

Ao acessar o *software* pela primeira vez é apresentado ao usuário a tela principal contendo as funcionalidades organizadas em menus conforme exibido na Figura 12:

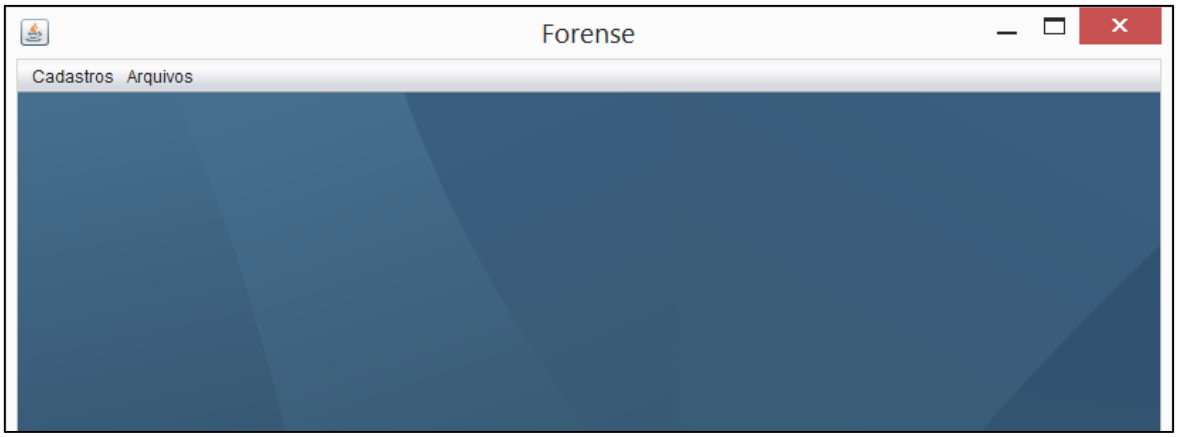


Figura 12 - Tela principal do software
Fonte: Elaborado pelo Autor

Inicialmente o sistema possui as seguintes funcionalidades:

- Tela principal permitindo o acesso às funcionalidades do *software*;
  - Tela de cadastramento de autores;
  - Tela para verificação das características extraídas após o processamento;
  - Tela de processamento de textos;
  - Tela de geração do arquivo ARFF;

Na tela de cadastro do autor, conforme ilustrado na Figura 13, é possível realizar o cadastramento, alterações ou mesmo exclusão de autores, nesse caso observa-se que foram cadastrados seis autores de diferentes períodos literários.

Figura 13 - Tela de cadastramento, alteração e exclusão de autores
Fonte: Elaborado pelo Autor

Na Figura 14 é exibida a tela onde pode ser executada a seleção e processamento de arquivos de texto, onde também é indicado o autor ao qual os textos que devem ser processados

pertencem, bem como, as características sintáticas, que devem ser extraídas da análise e persistidas na base de dados do software.

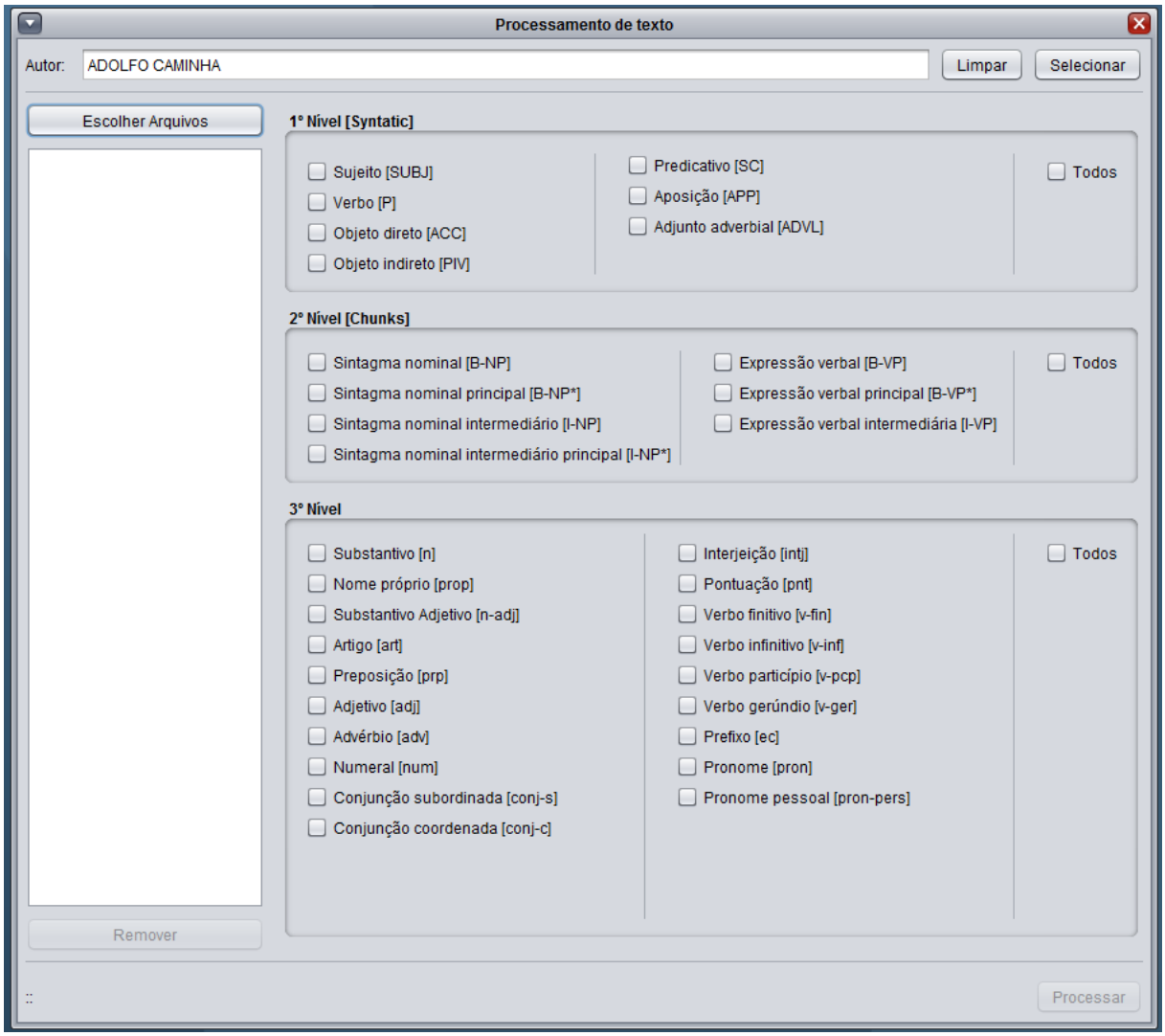


Figura 14 - Tela de processamento de textos
Fonte: Elaborado pelo Autor

Na sequência a partir da Figura 15 é possível verificar a seleção dos textos dos autores de diferentes períodos literários, que compõem o corpus de textos para que seja possível que o professor caminhe pela complementaridade dos conteúdos de estilos literários e gramática no ensino de Língua Portuguesa.

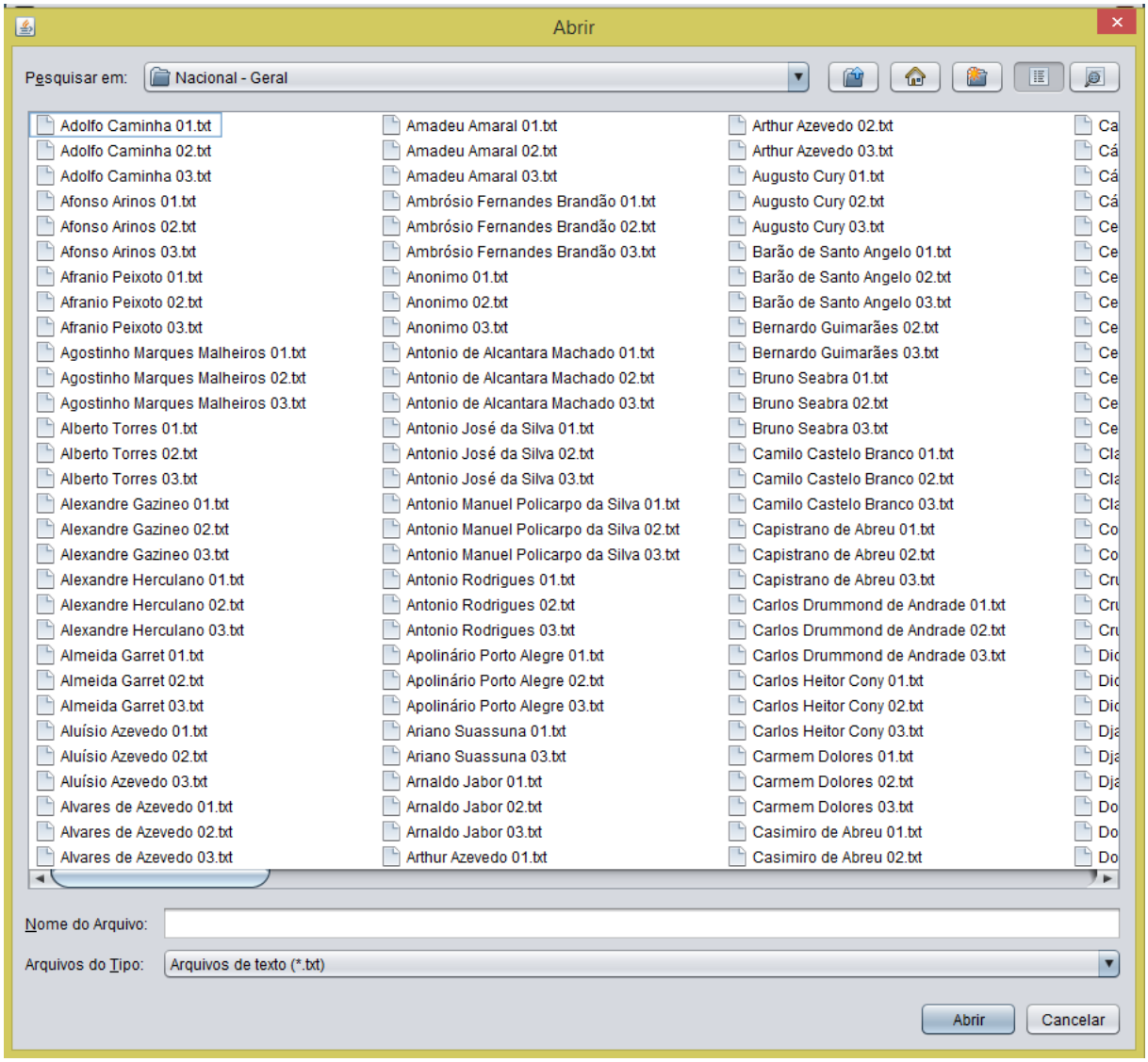


Figura 15 - Exemplo dos textos que compõem o corpus literário
Fonte: Elaborado pelo Autor

Após a seleção dos textos, que no caso desse corpus literário temos três amostras para cada autor, o professor poderá orientar em qual nível se baseará o estudo, sendo que na Figura 16 visualiza-se a seleção de três amostras de textos do autor Adolfo Caminha, onde serão trabalhadas as características presentes no Nível 1.

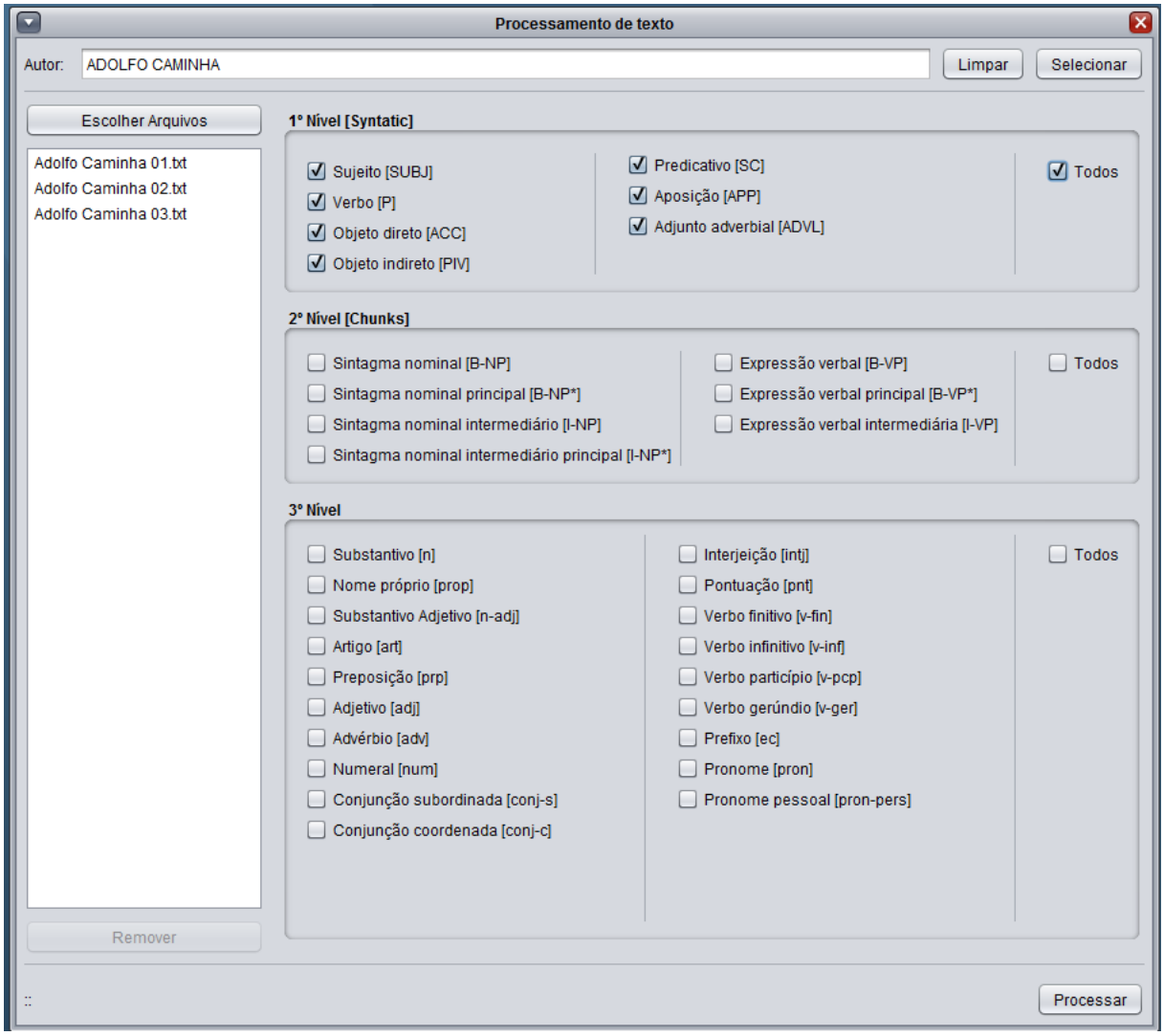


Figura 16 - Exemplo de seleção de autor, textos e características do nível 1
Fonte: Elaborado pelo Autor

Além disso, é possível, também, analisar as características nos três níveis sintáticos (Figura 17).

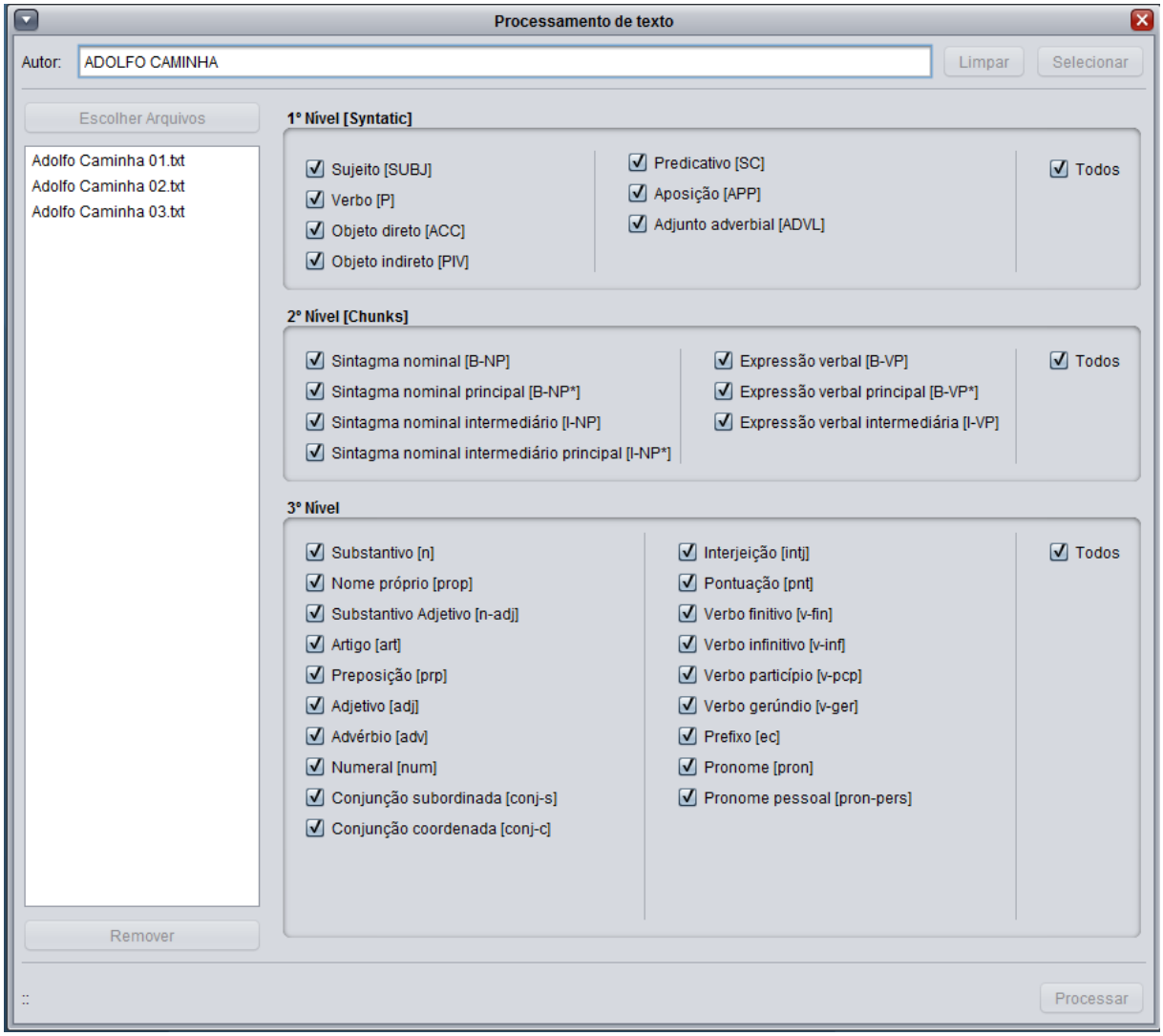


Figura 17 - Exemplo de processamento com os três níveis de análise
Fonte: Elaborado pelo Autor

Após os processamentos dos respectivos textos o aluno é orientado a visualizar as características extraídas, sendo que, primeiramente ele deverá selecionar o texto processado a partir do seu autor (Figura 18).

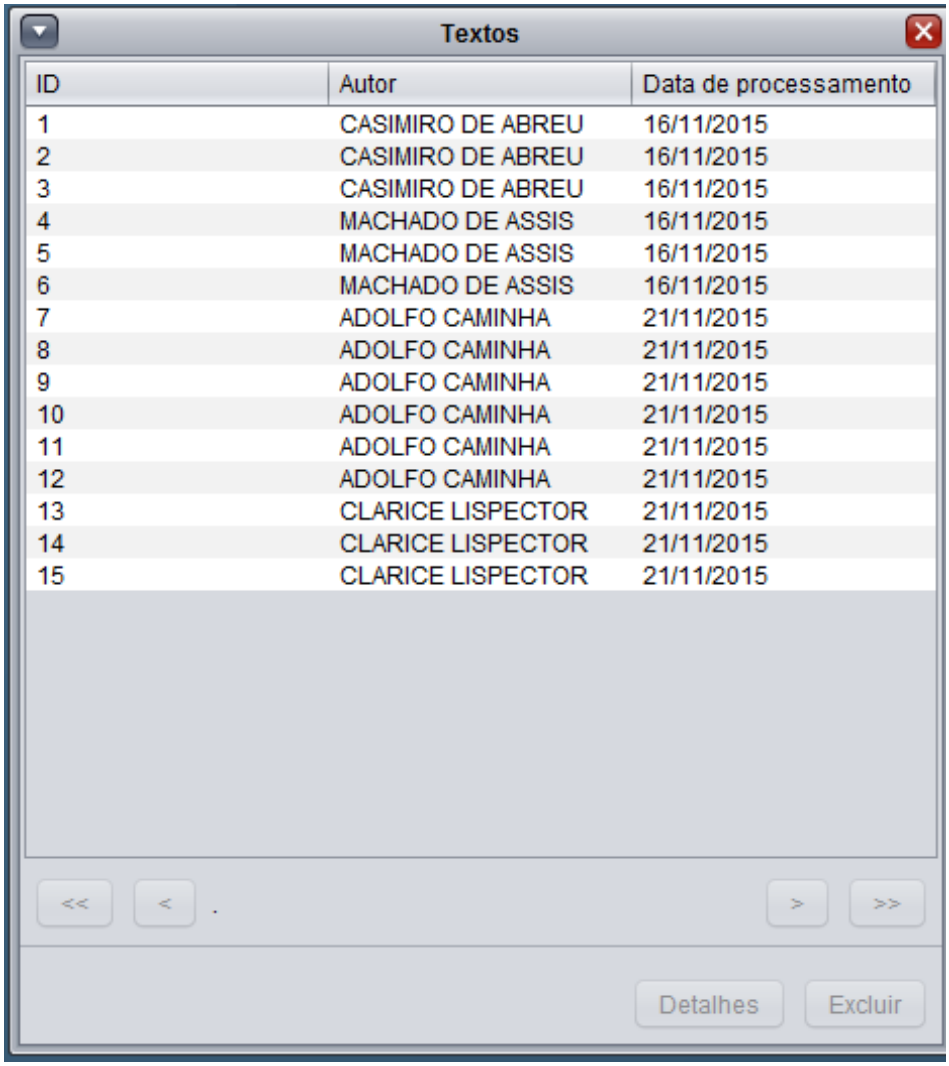


Figura 18 - Exemplo de textos processados para visualização das características
Fonte: Elaborado pelo Autor

Na sequência, o aluno verifica as características extraídas após o processamento acessando a tela de consulta conforme exibido na Figura 19, que nesse caso, demonstra características do tipo sujeito presentes no nível de extração.

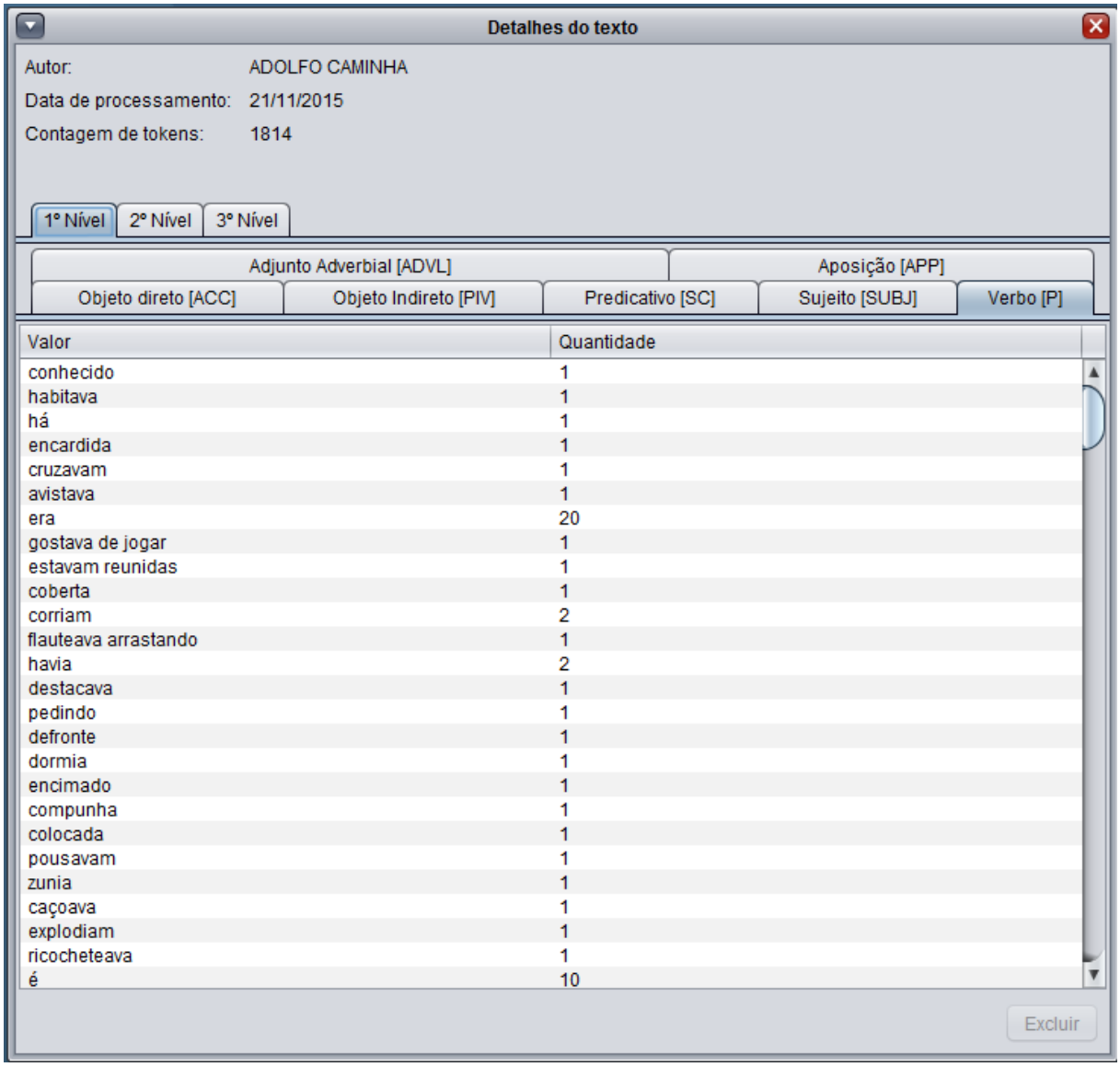


Figura 19 - Exemplo de características extraídas no nível 1
Fonte: Elaborado pelo Autor

Já no nível 2 onde estão localizados os sintagmas, observa-se os sintagmas nominais, que compõem o texto (Figura 20)

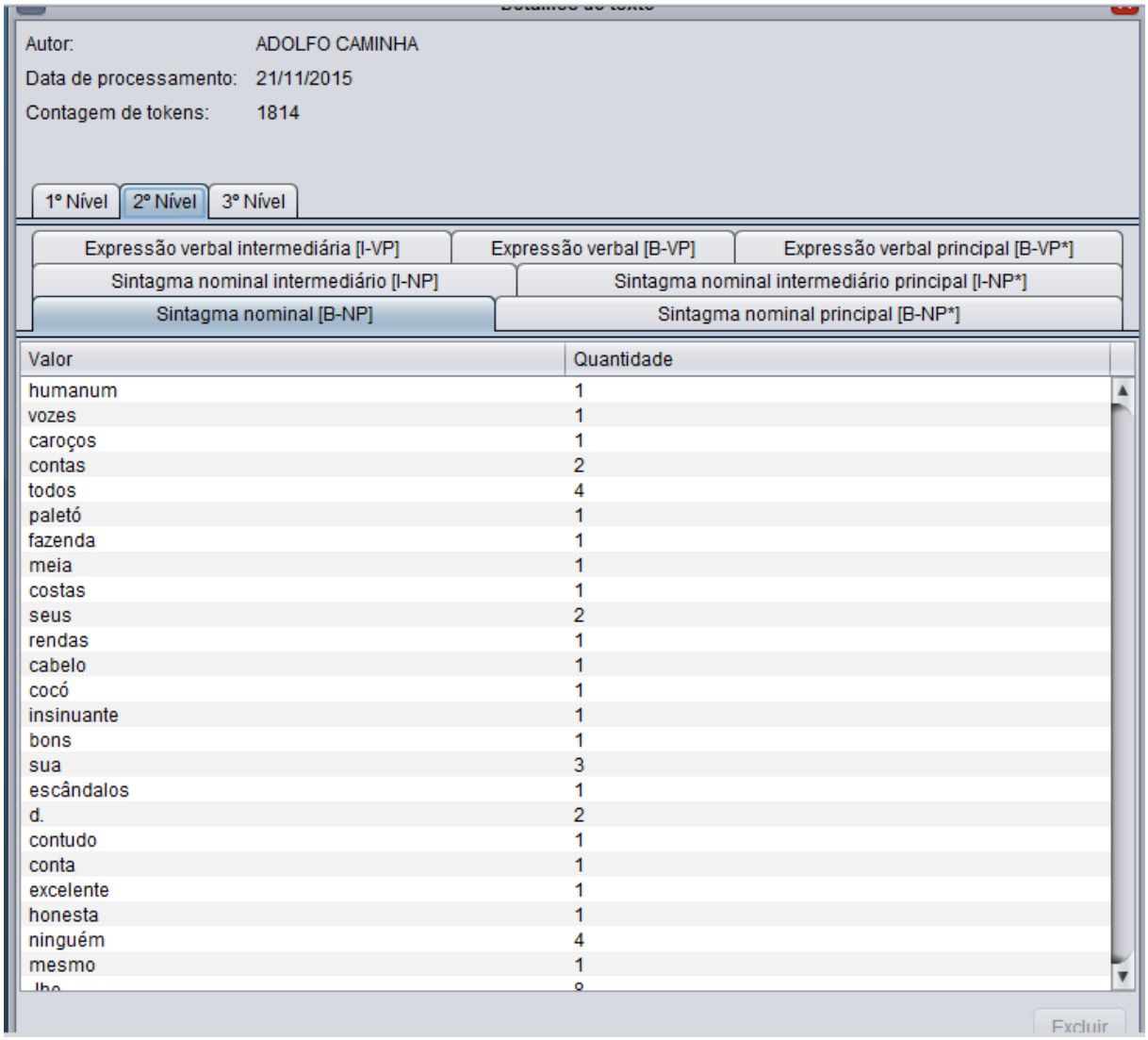


Figura 20 - Exemplo de características extraídas no nível 2
Fonte: Elaborado pelo Autor

E na Figura 21 visualiza-se a característica adjetivo presente no nível 3.

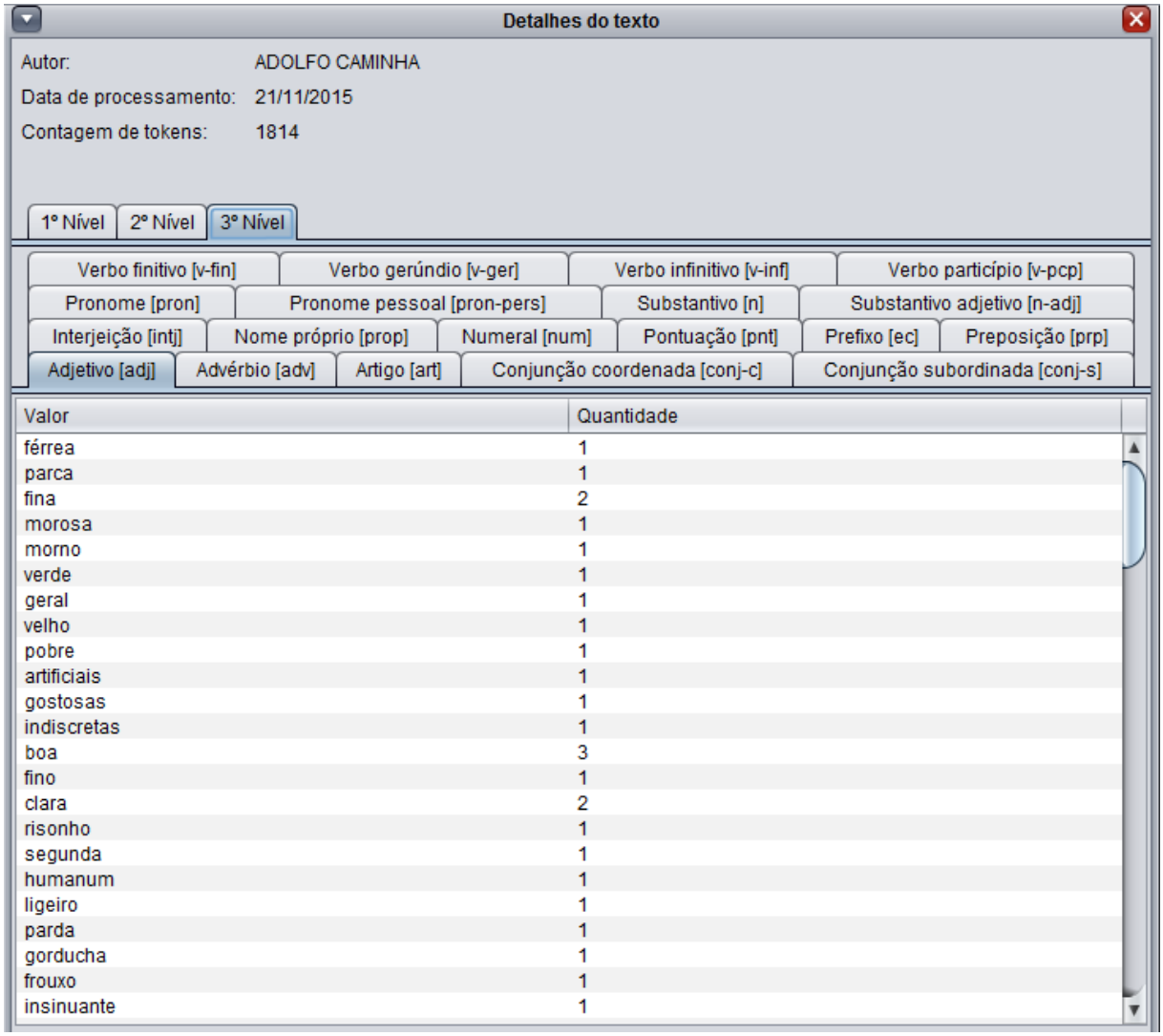


Figura 21 - Exemplo de características extraídas no nível 3
Fonte: Elaborado pelo Autor

Com esses resultados e o processamento de diversos autores, o software permite que o aluno observe dentro de textos de diferentes períodos literários e consequentemente seus autores, as particularidades de escrita de cada autor e período literário que o caracteriza (Tabela 6), além de observar de forma gráfica as funções das palavras dentro de uma frase, ou seja, de forma dinâmica ele observa a análise gramatical.

Tabela 6 - Análise do texto quanto as características de escrita do estilo literário do autor

	Quinhentismo	**Naturalismo**
	Pero Vaz de Caminha	**Adolfo Caminha**
Verbo	56	116
Objeto Direto	122	283
Objeto Indireto	62	140

Predicado	20	39
Aposto	5	45
Sujeito	10	30
Pronome	182	505
Preposição	110	300
Adjetivo	150	341
Conjunção Subordinada	30	17
Conjunção Coordenada	8	17
Verbo no Infinitivo	95	0
Verbo no Passado	30	212

Fonte: Elaborado pelo Autor

Falando mais especificamente, dos estilos literários com a aplicação dos quantitativos, no software Excel é possível gerar gráficos que demonstrem as diferenças de escritas de cada estilo literário com relação a utilização dos recursos da língua portuguesa brasileira (Gráfico 1).

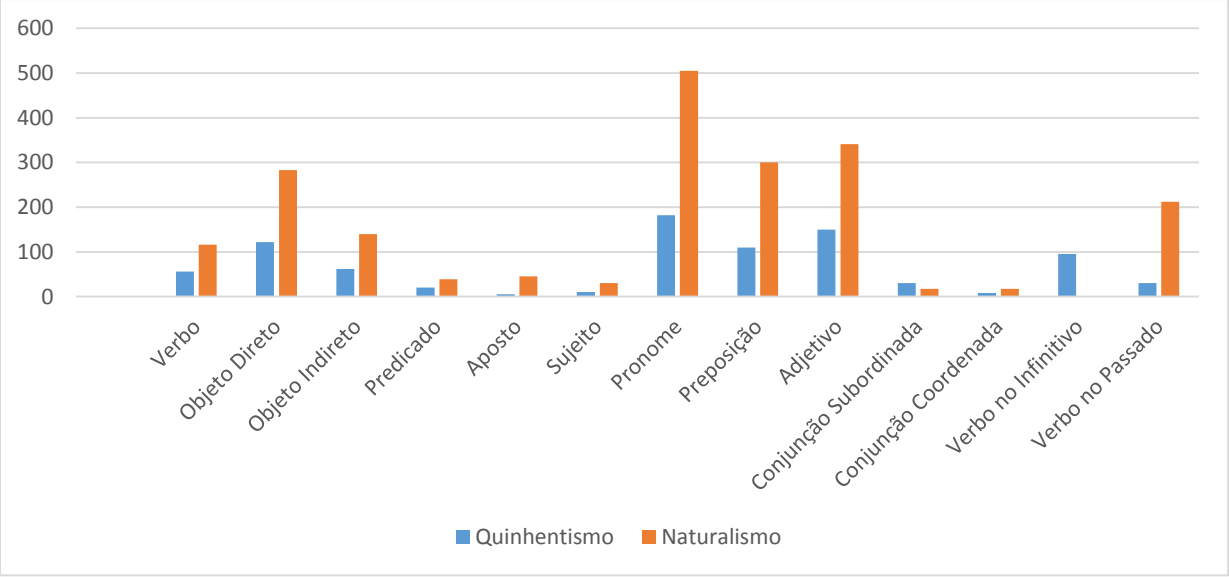


Gráfico 1 - Comparação entre o uso das características entre dois estilos literários
Fonte: Elaborado pelo Autor

Observando o Gráfico 1 infere-se que o quinhentismo utiliza-se de grande quantidade de adjetivos em sua escrita em comparação as demais características, o que se justifica por ser um período de escrita basicamente informativa, ou seja, os navegantes informado a Coroa de Portugal como era a sua nova Colônia, já no naturalismo a característica, que mais se sobressai é o pronome o que vai de encontro há duas características do período, que são o objetivismo e a impessoalidade, ou seja, faz uso de pronomes de forma a não deixar explícito o sujeito da frase.

Assim como pode-se realizar análises em textos, mesmo que curtos, é possível planejar a aula demonstrando as características em frases, como demonstra-se na Figura 22, onde optou-se pelo autor Carlos Drummond de Andrade, poeta, que fez parte do modernismo para dar continuidade a navegação dentro dos diferentes conteúdos do currículo da disciplina de Língua Portuguesa.

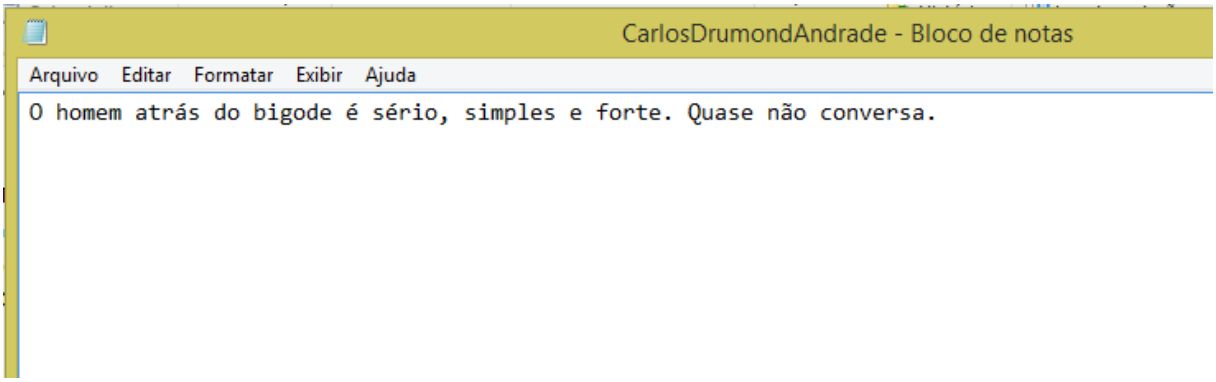


Figura 22 - Frase do autor Carlos Drummond de Andrade
Fonte: Elaborada pelo Autor

Para evidenciar de forma gráfica as funções de cada palavra dentro do contexto da frase, optou-se por montar a árvore sintática da frase acima na plataforma web Lx - Center[3] conforme a Figura 23.

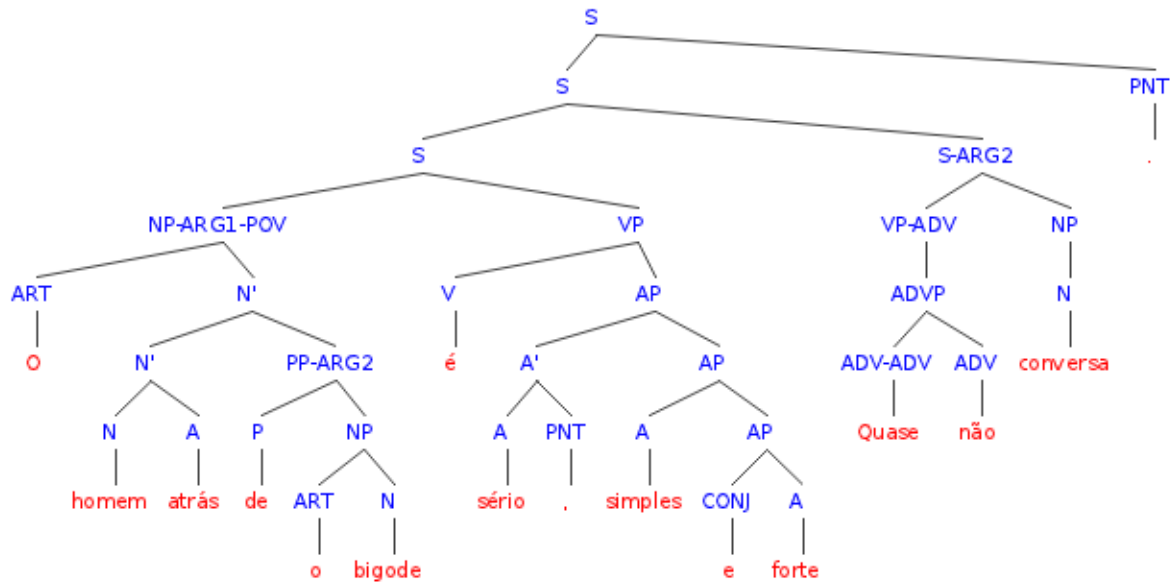


Figura 23 - Árvore sintática
Fonte: Elaborada pelo Autor

[3] http://lxcenter.di.fc.ul.pt/services/pt/LXSRLPT.html

Correlaciona-se então, com o software de forma que o aluno possa visualizar na árvore a construção da frase pelo autor e os níveis de análise possível fazendo uma comparação entre a árvore e a análise disponibilizada pelo software, sendo que no 1º nível, visualiza-se que os elementos sintáticos, que compõem essa frase são o sujeito caracterizado com a *tag N'* na árvore e visualiza-se no software na Figura 24 como "o homem atrás do bigode".

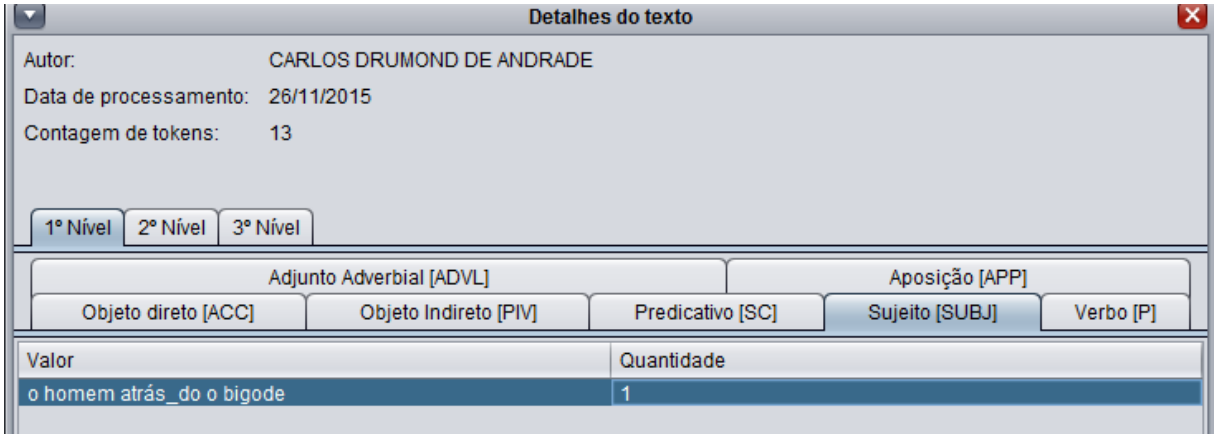


Figura 24 - Característica sujeito do Nível 1
Fonte: Elaborada pelo Autor

Já o predicado que constitui-se pelo restante da frase (Figura 25)

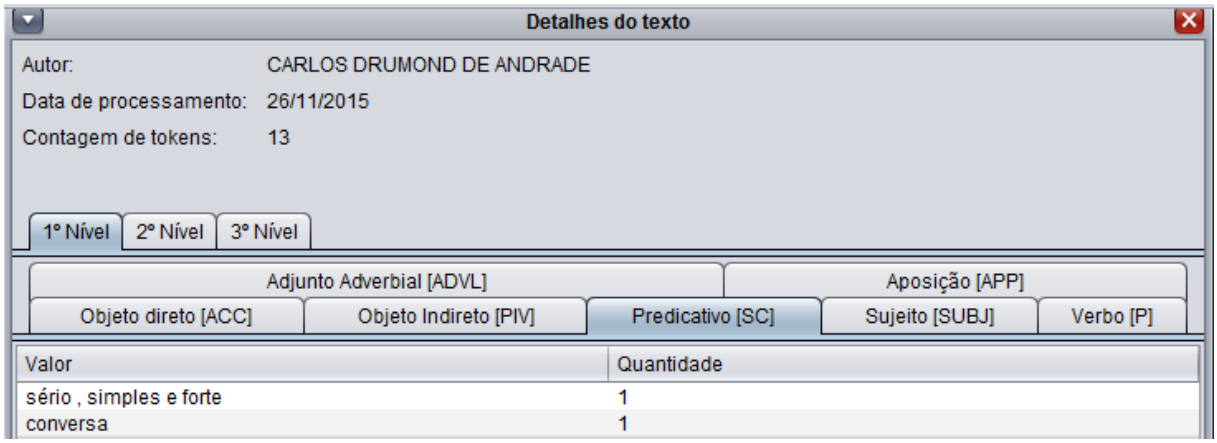


Figura 25 - Característica predicado
Fonte: Elaborada pelo Autor

E o verbo como "é" na árvore identificado pela *tag* V e o adjunto adverbial como "quase não" identificado na árvore pela *tag* ADVP e que são demonstrados no software (Figura 26).

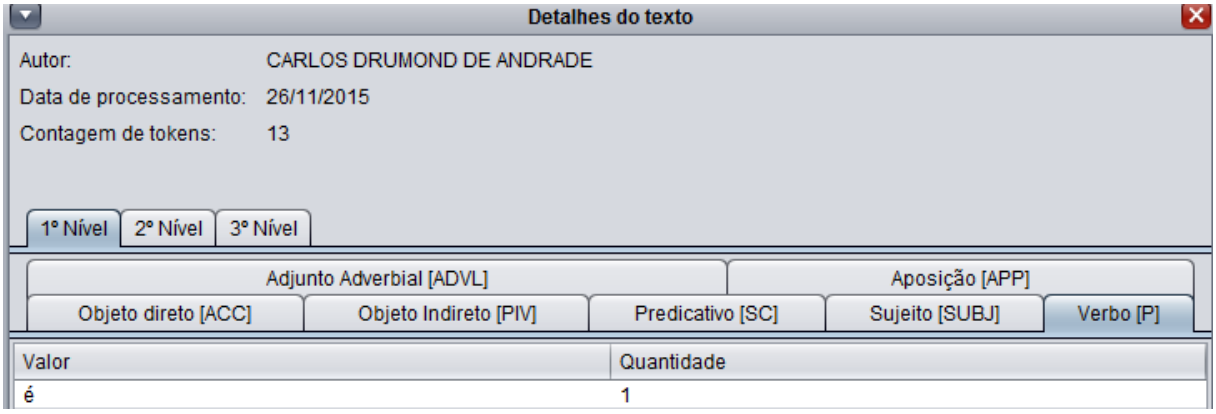


Figura 26 - Característica Verbo
Fonte: Elaborada pelo Autor

Quanto ao nível 2 na árvore da Figura 23 são identificadas pelas *tags* NP para sintagma nominal e VP para sintagma verbal, que também podem ser identificados pelo software com expressão verbal (Figura 27),

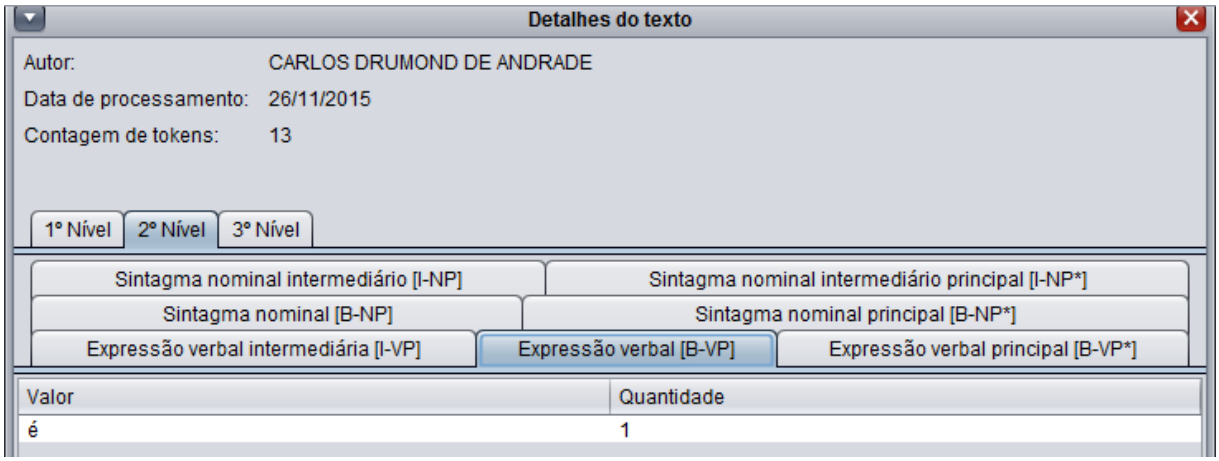


Figura 27 - Característica Sintagma Verbal do Nível 2
Fonte: Elaborada pelo Autor

E como sintagma nominal (Figura 28),

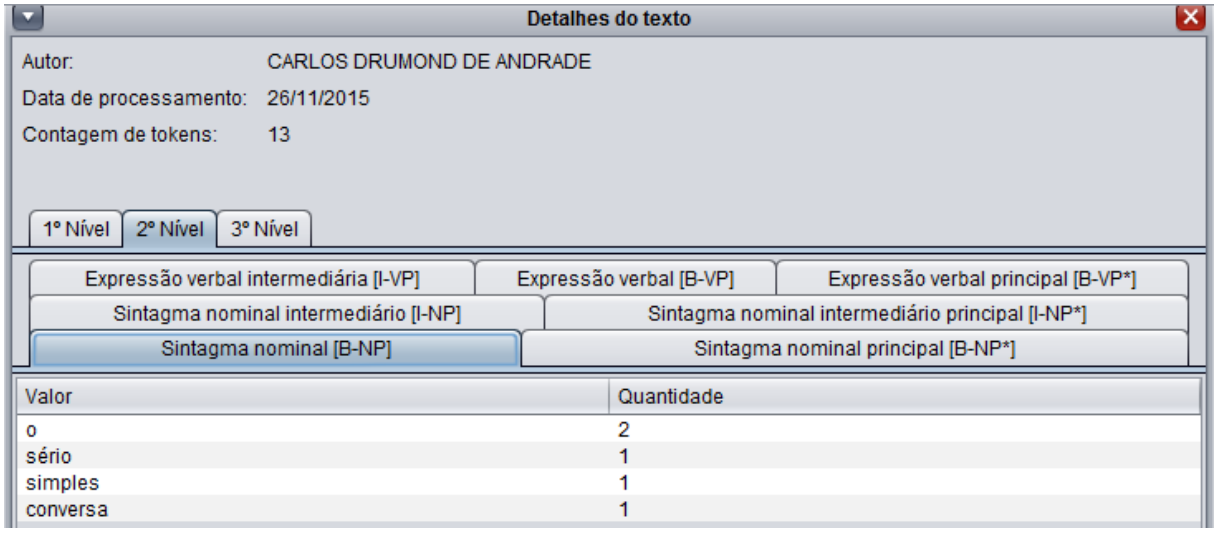


Figura 28 - Característica Sintagma Nominal do Nível 2
Fonte: Elaborada pelo Autor

Ressalta-se aqui que os sintagmas são unidades de análise diferentes da análise sintática comum, pois na análise sintagmal a frase é formada por fatores, que tem uma relação de dependência entre si onde um elemento e o determinante e outro o determinado sendo possível identificar os sintagmas, a partir dos elementos que compõem a oração como já foi exemplificado na secção 3.4.1.

Com relação ao 3º nível estuda-se as classes mais comuns das palavras que na árvore da Figura 23 é evidenciada pelas *tags:* A, ADV, N, P e CONJ, que significam adjetivo e no software pode ser visualizado na Figura 29,

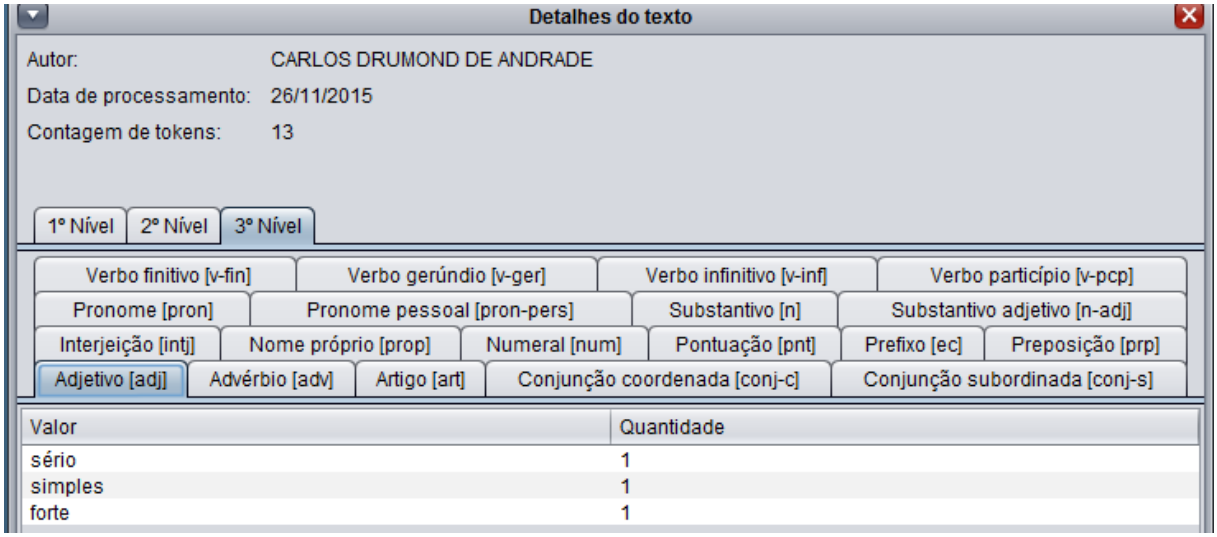


Figura 29 - Característica Adjetivo do Nível 3
Fonte: Elaborada pelo Autor

Advérbio que no caso do software fica evidenciado em (Figura 30)

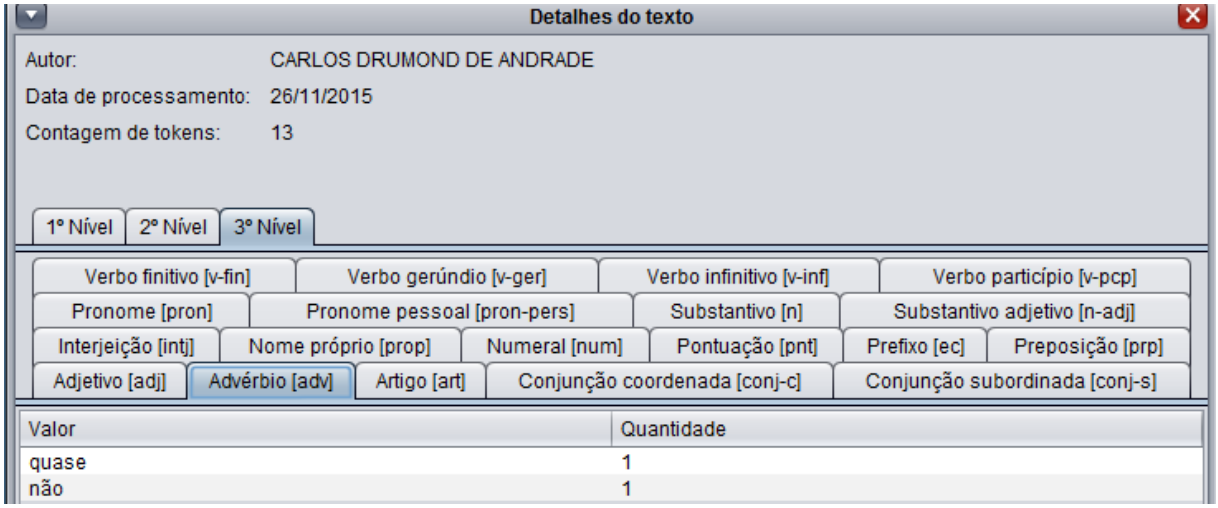


Figura 30 - Característica Advérbio do Nível 3
Fonte: Elaborada pelo Autor

O substantivo que demonstra-se no software (Figura 31),

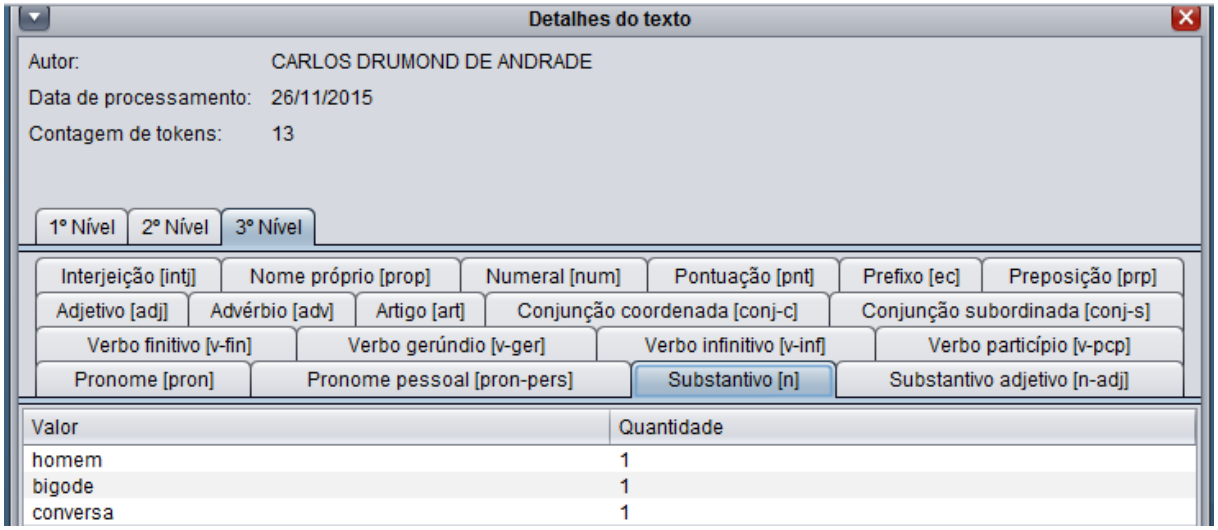


Figura 31 - Característica Substantivo do Nível 3
Fonte: Elaborada pelo Autor

A preposição que fica evidenciada em (Figura 32),

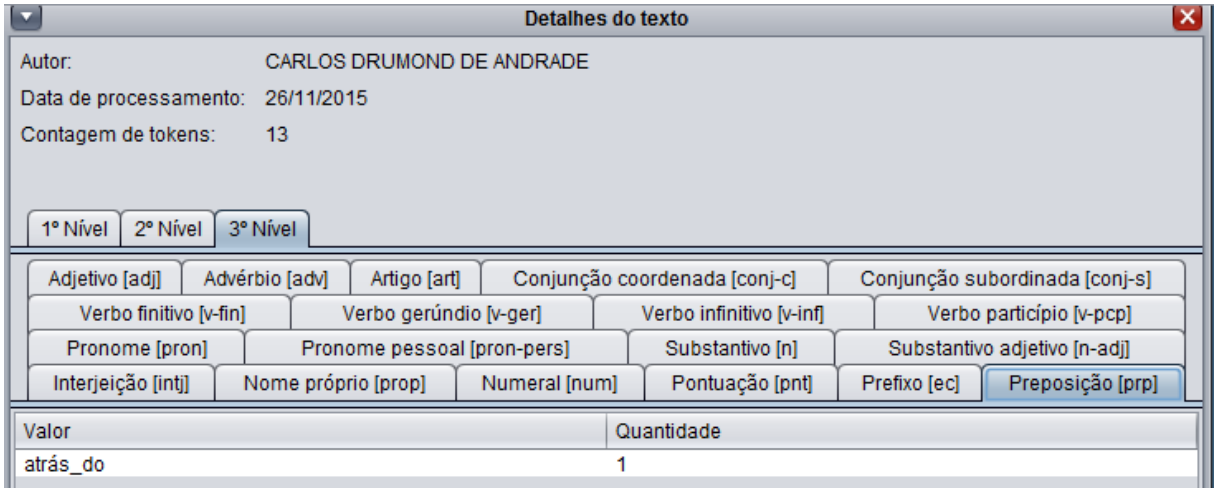


Figura 32 - Característica Preposição do Nível 3
Fonte: Elaborada pelo Autor

E a conjunção, respectivamente, que também visualiza-se no software (Figura 33).

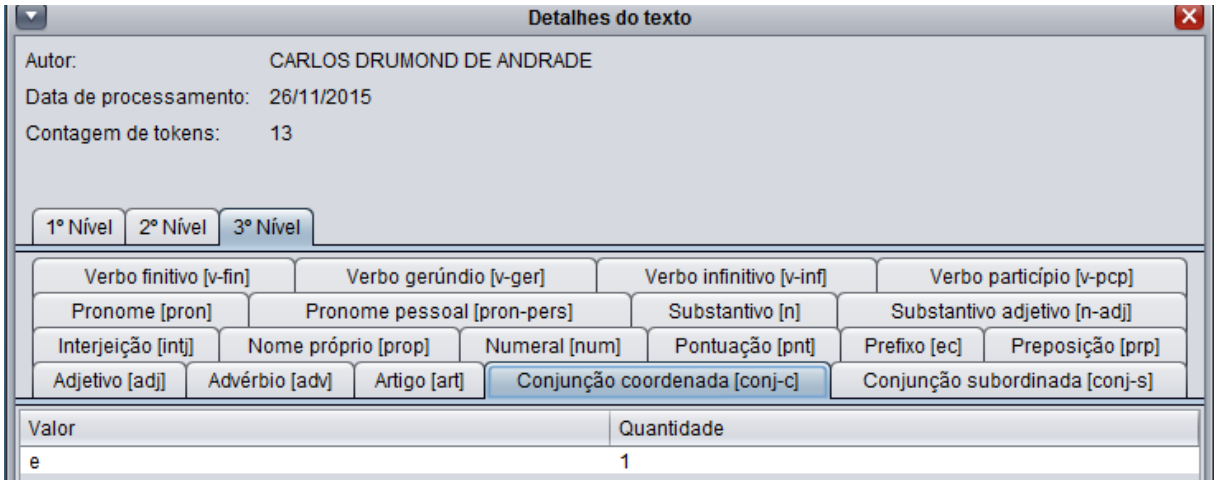


Figura 33 - Característica Conjunção do Nível 3
Fonte: Elaborada pelo Autor

No 3º nível podem-se extrair cada uma das classes de palavras em separado, combinadas ou no conjunto completo, sendo esta análise denominada morfológica, pois estuda as palavras de acordo com a classe gramatical a que elas pertencem como já foi explanado na seção 3.4.1. Ainda, o software permite observar outras classificações para as palavras que formam uma oração e constroem um texto.

Para tanto, fica evidente que a partir da utilização do software pode-se transitar e interligar dois conteúdo do currículo da disciplina de Língua Portuguesa: a gramática e a literatura, possibilitando estimular o aluno a leitura, a pesquisa e a discussão.

Além disso, implementou-se a tela exibida na Figura 34, onde é possível realizar a geração de arquivos ARFF. Este tipo de arquivo é compatível com o software Weka, que possui funcionalidade de identificação ou atribuição de autoria.

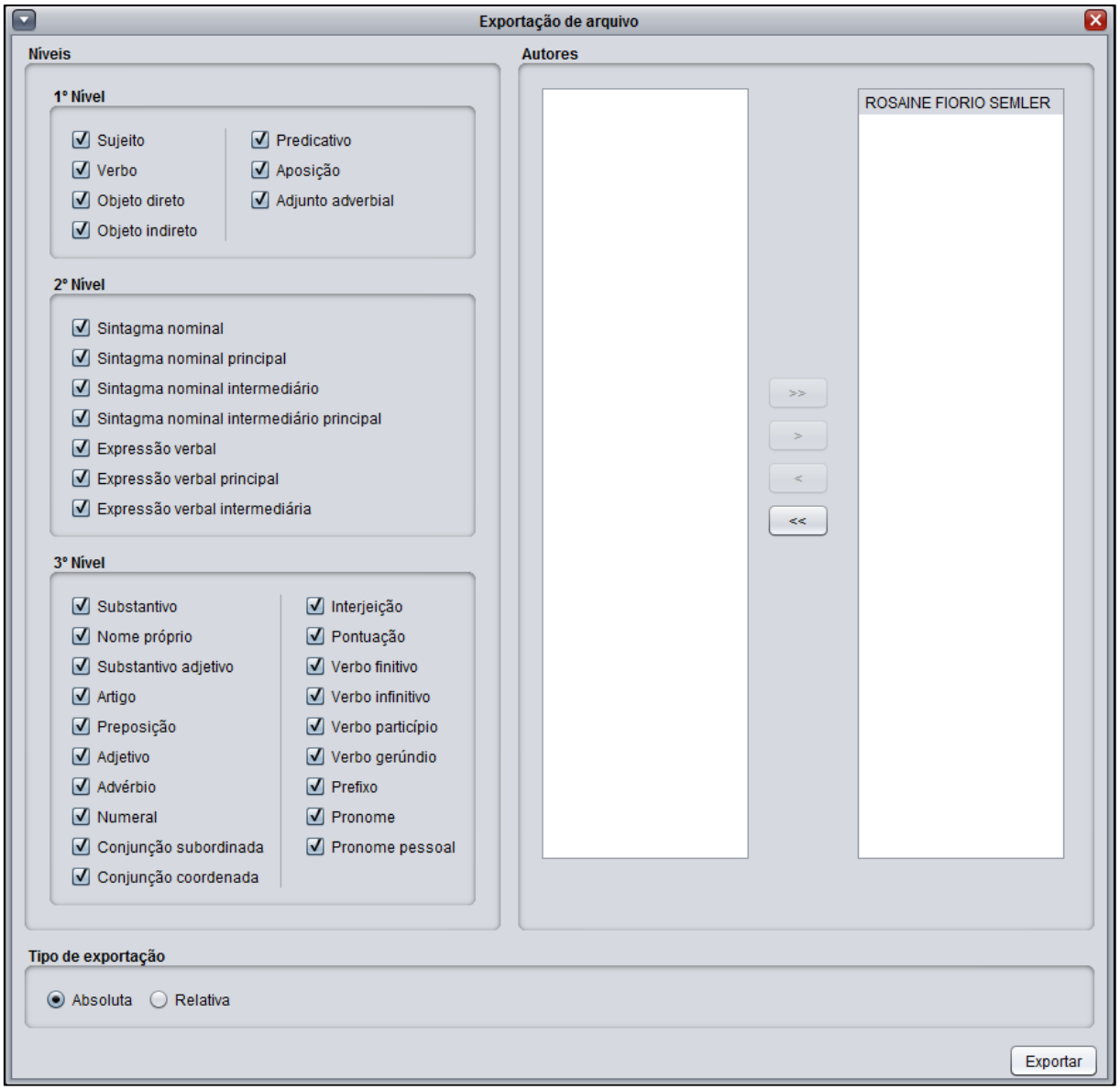


Figura 34 - Tela de geração de arquivo ARFF
Fonte: Elaborado pelo Autor

Enfim, com a disponibilização desse software tem-se um meio de auxílio no processo ensino aprendizagem da disciplina de Língua Portuguesa de forma a enriquecer a prática do professor e propiciar que o aluno compreenda e apreenda o conteúdo com o objetivo de auxilia-lo na construção de seu conhecimento.

5 CONCLUSÃO

Iniciando pelas tabuletas de argila, papiros, pergaminhos, prensas, quadro de giz e chegando aos computadores e softwares todos de alguma forma fizeram ou fazem parte da vida cotidiana da sociedade e, consequentemente, transformam a sociedade e permeiam a história da educação de forma a transformá-la, também. Assim, mantendo a ideia de que somos herdeiros de nossas invenções, é importante que o professor de Língua Portuguesa articule o trabalho com os conhecimentos construídos através da história com as novas tecnologias da informação e comunicação.

No entanto a escola é muito mais tradicional do que inovadora e isso se reflete nas metodologias de ensino, pois ainda faz uso do falar/ditar do professor e algum uso da impressão, sendo assim, integrar as formas de ensino com as tecnologias da informação e comunicação de maneira que o professor consiga mediar práticas de leitura, escrita, oralidade e análises da língua a partir de diferentes mídias e softwares de apoio ao aprendizado é importante para formação do aluno como crítico de sua sociedade e em especial de sua língua. Além disso, compreender que o professor encontra-se em um processo gradativo de assimilação das inovações tecnológicas deve ser levada em conta para que as mudanças que se almejam para a educação sejam alcançadas, porém para se chegar a esse patamar é preciso que o professor deixe de lado seus receios tecnológicos e arrisque-se em novas ideias e ações criativas. Então, em conformidade com o objetivo geral fica evidente que apesar da falta de incentivos e da quase inexistência de softwares que auxiliem o professor de Língua Portuguesa no ensino dos conteúdos de gramática e estilos literários é importante o uso de algum tipo de apoio que possa demonstrar ao alunado as diferentes características gramaticais da Língua Portuguesa Brasileira e, ainda ,incentivar a leitura, a análise das diferentes formas de utilização dos recursos da língua pelos autores da literatura brasileira e fomentar a busca por conhecimento e as discussões entre aluno-professor e aluno-aluno.

Com isso, o *software* desenvolvido permite a extração e análise de características sintáticas da língua portuguesa. Durante o desenvolvimento da aplicação não foram identificados problemas críticos que inviabilizassem a continuidade do projeto. As escolhas das tecnologias para o desenvolvimento mostraram-se boas no que diz respeito à produtividade e o resultado obtido foi satisfatório.

O software possui a capacidade de extrair os três principais níveis de informações sintáticas de uma frase, que são: morfológicas, sintagmas e sintáticas de função. Na classe

morfológica, ou também conhecida como classe de palavras, são extraídas e identificadas até 19 características, tais como: substantivo, artigos e preposições. Já em um segundo nível, as estruturas de sintagmas nominal e verbal, são distribuídas em 7 grupos de características. E, por fim, as características sintáticas de função na frase que são compostos, por exemplo, por sujeito e predicado. Neste caso, são 7 as características que podem ser extraídas.

Em correlato ao desenvolvimento do software também evidenciamos a importância de estudos relacionados a linguística computacional. É uma das áreas de pesquisa e desenvolvimento que ainda existe diversas lacunas, onde a informatização se faz essencial. Então, proporcionar um software que auxilie em qualquer um dos processos da linguística computacional é de suma importância. Neste caso, trabalhamos com o software que pode ser aplicado, principalmente, como apoio ao ensino do conteúdo de gramática na disciplina de Língua Portuguesa sendo que o professor pode se utilizar de textos literários caminhando assim pela complementaridade dos conteúdos e, ainda, pode ser aplicado em casos que envolvam a estilometria e a identificação de autoria.

Quanto ao processo de extração de características dos textos a biblioteca CoGrOO, que é integrada ao *software*, permite a análise de textos e a extração das características sintáticas de forma satisfatória. Entretanto, ainda é possível encontrar eventuais erros em resultados de análises, porém, continua sendo uma ferramenta muito útil por agilizar o processo de extração de características e propiciar que o professor enriqueça sua didática e na sua aplicação em sala de aula possibilite que o aluno observe as características sintáticas e possa discuti–las e ser um sujeito critico de sua língua.

No que concerne a utilização do *framework* de persistência de dados Hibernate, que foi utilizado para integração com o banco de dados, este se mostrou-se uma solução viável para desenvolvimento desta ferramenta. A linguagem de programação Java, utilizada no desenvolvimento da aplicação, mostrou-se eficiente pela facilidade de localização de exemplos e bibliotecas de código livre que auxiliaram e reduziram drasticamente o tempo de desenvolvimento. O banco de dados MySql se mostrou robusto e eficiente, para o armazenamento e manipulação dos dados.

O software desenvolvido foi testado e está em uso, a princípio, para desenvolvimento de uma pesquisa voltada a estilometria. Com relação ao seu uso como apoio no ensino da Língua Portuguesa, algumas funções está sendo implementadas para otimizar seu uso e para que o ensino da gramática em especial da análise sintática possa ser visualizada de forma dinâmica pelo alunado. Além disso o software foi disponibilizado para toda comunidade e continuará

sendo desenvolvido tanto nos aspectos da educação para melhorar sua atuação como software de apoio ao ensino como na área de identificação de autoria, área correlata, que faz uso da linguística em seus processos e que durante o levantamento de requisitos e referencial teórico da pesquisa surgiu como outra aplicação para o software.

Ainda, como trabalhos futuros, pretende-se trabalhar com outras línguas, tal como a língua inglesa e a espanhola. Também estaremos dispondo de um corpus de textos de autores de diferentes estilos da Língua Portuguesa e estilos literários para utilização pelo professorado e aluno no software. Para a área da educação pretende-se desenvolver uma funcionalidade que demonstre a árvore das características sintáticas de forma gráfica.

REFERÊNCIAS

ABRAHÃO, P. R. C. **Modelagem e Implementação de um Léxico Semântico para o Português**. 1997. 118 f. Dissertação (Mestrado em Informática) Instituto de Informática da Pontifícia Universidade Católica do Rio Grande do Sul. Porto Alegre: PUCRS, 1997

ALLEN, J. **Natural Language Understanding**. Redwood City, CA: The Benjamin/Cummings Pub. Co., 1995. 654 p

ALMEIDA, J. J. SIMÕES, A. **Jspellando nas morfolimpíadas: Sobre a participação do Jspell nas morfolimpíadas**. In Diana Santos, editor, Avaliação conjunta: um novo paradigma no processamento computacional da língua portuguesa. IST Press, 2007.

BAGNO, M. Preconceito lingüístico: o que é, como se faz. 8. ed. S. Paulo: Loyola, 1999.

BAKHTIN, M. Estética da criação verbal. São Paulo: Martins Fontes, 1992.

BAUER, C. KING, G. **Java Pesistence com Hibernate**. Rio de Janeiro: Editora Ciência Moderna, 2007

BENVENISTE, É. Problemas de linguística geral I. Campinas, SP: Pontes.1988.

_____. Problemas de linguística geral II. Campinas, SP: Pontes. 1989

BOUILLON, P.; *et al*. Apprentissage de Ressources Lexicales pour l'Extension de Requêtes. In: Jacquemin, Christian (editor). **Traitement Automatique des Langues pour les Recherche d'Information**. Hermès Science Publications, Paris, 2000. p.367-393

BRASIL. Lei de Diretrizes e Bases da Educação Nacional. Lei número 9394, 20 de dezembro de 1996. Brasília, 1996.

BROWN, H. **Teaching by principles: an interactive approach to language pedagogy**. New Jersey: Prentice Hall. 2007

CAGLIARI, L. C. Alfabetização e lingüística. São Paulo: Scipione, 1990.

CÂMARA JR., J. M. Estrutura da língua portuguesa. 16. ed. R. de Janeiro: Vozes, 1986.

CEREJA, W. R. **Ensino de literatura: uma proposta metodológica para o trabalho com literatura**. São Paulo : Atual, 2005

CORREA, R. **Processamento de Linguagem Natural**. Disponível em <https://sites.google.com/site/renatocorrea/temas-de-interesse/processamento-de-linguagem-natural> Acesso em 21//11/2015

CONSELHO DA EUROPA. **Quadro europeu comum de referência para as línguas: aprendizagem, ensino, avaliação**. Porto: Asa, 2001.

COOK, V.. Competence and multi-competence. In G. Brown, K. Malmkjaer & J. Williams (eds.), Performance and Competence in Second Language Acquisition. Cambridge: Cambridge University Press, 1996, pp. 57-69.

COSTA, G.. Português terá 350 milhões de falantes ate o final do século, prevê especialista. Agencia Brasileira, Empresa Brasil de Comunicação. 2013. Disponível em: <http://agenciabrasil.ebc.com.br/noticia/2013-02-28/portugues-tera-350-milhoes-de-falantes-ate-final-do-seculo-preve-especialista> Acesso em 12/12/2013

CRYSTAL, D. **Dicionário de Linguística e Fonética**. 8ª Ed. Rio de Janeiro: Ed. Jorge Zahar, 2000.

DATE, C. J. **Introdução a sistemas de banco de dados.** 8. ed. Rio de Janeiro: Campus, 2004

DEITEL, P.; DEITEL, H. **Java como programar**. 5. ed. São Paulo: Pearson Education do Brasil, 2010

DUARTE, V. M. N. Sintagma Nominal e Sintagma Verbal. Disponivel em < http://www.portugues.com.br/gramática/sintagma-nominal-sintagma-verbal.html> Acesso em 15/10/2014

ELMASRI, R.; NAVATHE, S. B. **Sistemas de banco de dados.** 4. ed. São Paulo: Pearson Addison Wesley, 2005.

FARACO, C.A. Área de linguagem: algumas contribuições para a sua organização. In: KUENZER, A. (Org.) Ensino Médio: construindo uma proposta para os que vivem do trabalho. São Paulo: Cortez, 2002.

FEIGENBAUM, E.; BARR, A. **The handbook of artificial intelligence**. California: Heuristech Press, 1981.

FERREIRA, A. B. H.. **Novo Dicionário Eletrônico Aurélio versão 5.0**. Editora Positivo, 3 edition, 2004.

FUZER, C. **A linguística no tempo: pontos de vista sobre a linguagem**. 2004. Dissertação (Programa de Pós-Graduação em Letras)Universidade Federal de Santa Maria, Santa Maria, 2004

GERALDI, J.W. Concepções de linguagem e ensino de português. In: **O texto na sala de aula**. 4ª. ed. São Paulo, Ática, 2006

GERMAIN, C. & SÉGUIN, H. **Le point sur la grammaire en didactique des langues**. Anjou: Centre Educatif et Culturel, 1999.

GONZALEZ, M.; LIMA, V. L. S. Recuperação de Informação e Processamento da Linguagem Natural. XXIII Congresso da Sociedade Brasileira de Computação, Campinas, 2003. **Anais do III Jornada de Mini-Cursos de Inteligência Artificial**, Volume III, p.347-395.

GUEDES, G. T. A. ***UML - Uma abordagem prática***. 1. ed. São Paulo: Novatec, 2004. 319.

GUTHRIE, L.; *et al.* **The Role of Lexicons in Natural Language Processing**. Communications of the ACM, V.39, N.1, janeiro 1996. p.63-72.

HEUSER, C. A. **Projeto de banco de dados**. 5. ed. Porto Alegre: Sagra Luzzato, 2004.

HIBERNATE, **Core for Java**. Disponível em: <http://docs.jboss.org/hibernate/stable/core/reference/en/pdf/hibernate_reference.pdf>. Acesso em: 04/03/2014.

HOUAISS, Antônio. O português no Brasil. Rio de Janeiro: EBRADF, 1985

LAUDON, K. C.; LAUDON, J. P. **Sistemas de informação**. 4. ed. Rio de Janeiro: LTC,1999

LEME FILHO, T. **Metodologia de desenvolvimento de sistemas**. 1. ed. Rio de Janeiro: Axcel Books, 2003. 154

LÉVY, P. As tecnologias da inteligência coletiva. O futuro do pensamento na era da informática. São Paulo: Editora 34, 1993.

LIBÂNEO, J.C. Adeus professor, adeus professora? Novas exigências educacionais e profissão docente. São Paulo: Cortez, 2011.

LYONS, J. **Introduction to Theorical Linguistics**. Cambridge University, Cambridge, 1968.

MATSUKI, C. T. Modelagem de dados. **SQL Magazine**, v. 5, n. 51, p. 48–49, 2008.

MEDEIROS, E. **Desenvolvendo software com UML 2.0 definitivo.** São Paulo: Pearson Makron Books, 2006.

MENUZZI, S. M.; OTHERO, G. Á. SINTAXE X-BARRA: uma aplicação computacional. **Work. pap. linguíst., n.esp**.: p. 15-29, Florianópolis, 2008

MYSQL AB, Site Oficial do Brasil. Por que MySQL | Mysql Brasil. Disponível em: http://www.mysqlbrasil.com.br/?q=node/2. Acesso em: 04/03/ 2014

OLIVEIRA, F. A. D. de. Processamento de linguagem natural: princípios básicos e a implementação de um analisador sintático de sentenças da língua portuguesa. **Revista de Ciência da Informação**. Rio de Janeiro. n. 5. Maio 2002. Disponível em: <http://www.inf.ufrgs.br/procpar/disc/cmp135/trabs/992/Parser/parser.html>,Acessado em: 15 de junho 2014.

OTHERO, G. Á. **Grammar Play: um parser sintático em Prolog para a língua portuguesa**.2004. Dissertação (Mestrado em Linguística) - Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, 2004.

_____. Teoria X-barra: descrição do português e aplicação computacional. São Paulo: Contexto, 2006.

_____. Teoria X-barra: descrição do português e aplicação computacional. São Paulo: Contexto, 2006.

_____. Lingüista “puro” vs. lingüista “computacional”: revisitando a distinção entre “lingüista de poltrona” e “lingüista aplicado”. Domínios de Lingu@gem – Revista Eletrônica de Lingüística. Ano 2, nº 1, 2008.

PAVELEC, D. F. **Identificação da Autoria de Documentos: Análise Estilométrica da Língua Portuguesa usando SVM**. 2007. 98f. Dissertação (Mestrado em Informática) - Pontifícia Universidade Católica do Paraná, Curitiba, 2007.

PESSANHA, E. C.; DANIEL, M. E. B.; MENEGAZZO, M. A. A história da disciplina Língua Portuguesa no Brasil através dos manuais didáticos (1870-1950). **Educação em Foco**, Juiz de Fora, v. 8, n. 1/2, p. 31-45, 2003/2004.

PEREIRA, J. C. F. **Teoria da Literatura: Anatomia de um Conceito Através da leitura dos Grandes Manuais**. Niteroi: UFF, 2006. Diponivel em:< http://www.bdtd.ndc.uff.br/tde_arquivos/23/TDE-2007-03-21T120009Z-683/Publico/UFF-Tese-juliopereira.pdf> Acesso em 04/10/2013

REGENSTEINER, R. J. **Elementos básicos para o planejamento da automação do varejo**. 1. ed. São Paulo: SENAC, 1999. 106

RIBEIRO, A.E. Textos e hipertextos na sala de aula. In: COSCARELLI, C.V. (Org.) Novas tecnologias, novos textos, novas formas de pensar. Belo Horizonte: Autêntica, 2006

SANTAELLA, L. Culturas e artes do pós-humano: da cultura das mídias à cibercultura. São Paulo: Paulus, 2003.

SCAPINI, I.K.; Relações entre Itens Lexicais. In: Poersch, J. M.; Wertheimer, A.M.C.; Ouro, M.E.P.; Ludwig, E.M.; Scapini, I.K.; Becker, B.F. **Fundamentos de um Dicionário Remissivo**. 1º Encontro do CELSUL, Florianópolis, novembro 1995. Anais, V. 1, p.393-429

SILBERSCHATZ, A.; KORTH, H.; SUDARSHAN, S. S**istema de banco de dados.** 3. ed. São Paulo: Pearson Makron Books, 2006.

SILVA, R. R., LIMA, S. M. B. Consultas em Bancos de Dados Utilizando Linguagem Natural. Disponível em < http://re.granbery.edu.br/artigos/MjQ0.pdf> Acesso em 30/04/2013

SMARSARO, A. das D. **Descrição e formalização de palavras compostas do português do Brasil para elaboração de um dicionário eletrônico**. 2004. 130 f. Tese (Doutorado em Letras) - Programa de Pós-graduação em Letras do Departamento de Letras da Pontifícia Universidade Católica, Rio de Janeiro, 2004.

SOARES, M. Português na escola: história de uma disciplina curricular. In: BAGNO, M. (Org.). Linguística da norma. São Paulo: Loyola, 2002b, p. 155-177.

_____. Que professores de português queremos formar? Movimento, Niterói, n. 3, p. 149-155, 2001.

STALLINGS, W. **Arquitetura e organização de computadores:** projeto para o desempenho. 5. ed. São Paulo: Prentice Hall, 2002.

STAIR, R. M. **Princípios de sistemas de informação**. 2. ed. Rio de Janeiro: LTC, 1998.

VARELA, P. J. **O uso de atributos estilométricos na identificação da autoria de textos**. Dissertação (Mestrado em Informática) -Pontifícia Universidade Católica do Paraná, Curitiba, 2010

VIEIRA, R.; LIMA, V. L. S. de. **Lingüística Computacional: Princípios e Aplicações**. JAIA, SBC, Fortaleza, Brasil, 2001.

VILARIM, G. **Algoritmos: programação para iniciante**. Rio de Janeiro: Ciência Moderna, 2004.

WELLING, L.; THOMSON, L. PHP e MySQL: **Desenvolvimento Web**. 3. ed. Rio de Janeiro: Campus, 2005

WERTHEIMER, A. M. C. O Dicionário Remissivo Comparado aos Outros Dicionários Existentes. In: Poersch, J. M.; Wertheimer, A. M. C.; Ouro, M. E. P.; Ludwig, E. M.; Scapini, I. K.; Becker, B. F. **Fundamentos de um Dicionário Remissivo**. 1º Encontro do CELSUL, Florianópolis, novembro 1995. Anais, V. 1, p.393-429

WILKS, SLATOR, GUTHRIE, Y. A.; SLATOR, B. M.; GUTHRIE, L. M. **Electric Words: Dictionaries, Computers and Meanings**. Cambridge: The MIT Press, 1996. 289 p.

ZINANI, C.J.A. Adolescência: leitura e realidade cultural. 1991. Dissertação (Mestrado em Letras: Teoria da Literatura). Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, 1991.

APÊNDICES

APÊNDICE A – Diagrama Entidade Relacionamento