Linguistic Interpretation of Speech Errors

Daniela Gîfu
"Alexandru Ioan Cuza" University of Iaşi, Romania
daniela.gifu@info.uaic.ro

Marius Cioca
"Lucian Blaga" University of Sibiu, Romania
marius.cioca@ulbsibiu.ro

***Abstract*—The paper is an attempt to illustrate the linguistic interpretation of speech, known that it remains insufficiently resolved, especially for Romanian. The cause is given by the multitude of criteria that can or should be considered important in speech processing. The aim of this study is to develope a computational tool in order to identify the possible errors related to the morphosintactic structure of speech. Our goal is to assist users who can receive automatically different suggestions that can help them to improve the quality of their text. Thus, we chose an interdisciplinary approach through speech analysis that brings together the key fields of linguistics, computer science and so on. The analysis involves a careful reading of the text, implying the examination of the language used to understand how the speaker's communicative intention is reflected in language. Therefore, the users would be able to visualize different texts and to analyse them at the lexical and grammatical level, in order to identify certain types of errors in the writing related to morphosyntactic appearance. Then they can correct the texts. This stud intends to help direct beneficiaries (students, journalists, etc.), but, also, specialists and researchers in natural language processing field in order to improve the writing skills and comprehension of texts.**

***Index Terms*—speech, classification, spellchecking, grammatical correction, natural language processing.**

I. Introduction

The option for the topic of linguistics interpretation of speech errors is still a challenge in NLP (*Natural Language Processing*) field. This survey comes from the need to develop an application for Romanian language for morphosintactic analysis of the content in order to facilitate correct text spelling, named LAT (*Linguistic Analysis Tool*). This is a computational intelligence method. The speech is characterized by an important level of diversity and one of the most known typologies structures it in three classes, necessary in speech interpretation: *enunciative*[1], *communicational* and *situational* [10].

Moreover, speech recognition demands a certain level of spelling, which represents our goal for this paper.

The concept, "speech" has no plural; it designates a generic field, the relation between language and parameters of the nonlinguistcs reality. The literature provides a few classic definitions. For this study, we consider the functional part of it very important, the speech analysis being "the analysis of language in use" [2]. In other words, the language "is doing some job in some context" [8]. Only a part of what can be said is accessible, forming a system and defining an identity [9]. According to these opinions, we develop a method useful to fix different errors that might appear in a speech.

The paper is structured in five sections. After a brief introduction about the importance of this study, section 2 mentions some important computer applications focused on the examination of speech. Section 3 describes the LAT architecture, and section 4 presents shortly the work methodology. Last section highlights conclusions and mentions the future work.

II. Related works

To develop this application, the folowing tools were explored as: Tropes[1], RO-Balie[2] [4], and other competing software as Gate[3], Oak[4] or Minor Third[5]. All these applications, except RO-Balie, could be trained, using machine learning techniques.

Tropes V1.0, developed in 1994 by Pierre Molette and Agns Landr, is the first semantic analysis software based on propositional analysis principles [7]. Tropes is now available in two versions: French or English. It creates a specific thesaurus for a certain domain (literary, philosophic, politic and scientific), performing a morphosyntactic analysis on the text by identifying the morphologic category of all the words, including: nouns, verbs, adjectives, pronouns, connectors, etc. It is important to note that Tropes can yield information about a text. For instance, to detect and quantify different categories of the emotional lexicon in French (EMOTAIX-Tropes) [11].

The similarities between Tropes and LAT are that both perform a morphosyntactic analysis on the text, identifying the main part of speech [6].

RO-Balie is an extension of BALIE[6], known as a multilingual text processing tool designed to support information extraction. The tool supports five languages: French, German, Spanish, English, and Romanian. It is used for language identification, text segmentation in sentences, tokenization, and part-of-speech tagging. Balie text processing consists of

[1]Tropes Text analysis software, designed for TextMining, Qualitative Analysis, Semantic Categorization and Keywords extraction. In 1997, Acetic launches Tropes V3.0 automates the ACD (remarkable phrases), and chronological analysis of the text, starting from the political texts (http://www.semantic-knowledge.com/tropes.htm).

[2]RO-BALIE is available for download at http://www.site.uottawa.ca/˜ofruna/ RO-Balie/RO-Balie.html

[3]http://gate.ac.uk

[4]http://nlp.cs.nyu.edu/oak

[5]http://minorthird.sourceforge.net

[6]BALIE is a Java open-source software issued under the GNU General Public License. It is hosted by SourceForge and it is available at http://balie.sourceforge.net

a structured list of tokens, by applying machine learning techniques [10].

The similarities between RO-Balie and LAT are that both allow extracting the tokens from the text and identifying the main part-of speech, and are using the Nave Bayes classifier (here, to identify the speech type).

In addition, LAT performs grammar and spellchecker on the text.

III. LAT ARCHITECTURE

To implement the LAT application as a development environment we used *Spring Tool Suite (STS)*, version 3.6.4 *RELEASE*, and as a programming language, the application was written in Java, within Java EE platform (*Enterprise Edition*).

The server part was performed using Spring MVC (*model-view-controller*) framework[7] and Apache Tiles 3 framework, and on the client side, for design, we used Bootstrap[8][9], and for Ajax calls to server, the javascript was chosen (jquery 1.11.1).

Following the analyses carried out, the application offers the user the possibility to print or copy the data in the table, or to save the results in PDF, XLX or CSV format [3]. This functionality has been achieved by using the plugin "*TableTools*"[10] offered by the "*DataTables*"[11] software, a plug-in for the jQuery JavaScript library.

A. STS

Spring Tool Suite (STS)[12] is a development environment based on the Eclipse platform and created for the developmen of Spring applications, offering support both for the implementation, debugging, running and launching of applications and for refactoring, AOP (Aspect Oriented Programing)[13], validations, Spring beans, Spring Persistence (Hibernate + JPA), Spring MVC, support for Java 8, integration with Maven, navigation and advanced searching etc. The development process of this environment is much easier and faster, both because of its functionalities and because of the possibility to combine all these with the desired Java version (Java 7), with the Web part and Java EE from Eclipse. LAT was created with Spring MVC, that will be presented in the following, followed by the description of the MVC used on the server side.

B. Spring, Spring Security and Spring MVC (11)

Spring[14] is the most popular development environment, *open-source*, created in order to tackle the complexity of developing Java enterprise applications, the framework consisting of six well-defined modules (Figure 1)[15].

Among the benefits of using this framework, there can be mentioned: the developing of enterprise applications, using Java objects called POJO (*Plain Old Java Objects*)[16], the modular organization of packages, the loose coupling by *dependency injection* and by using the concepts of the IoC (*Inversion of Control*)[17] process, the declarative programming based on aspects and many others.

Dependency injection allows the decoupling to be concrete, the classes easier to test by injecting mocks[18], and the code, easier to understand and to maintain, a principle called *Clean Code*.

Spring Security 4.0.0 Release

The authentification mechanism within the application was created using the authentification and access control framework *Spring Security*[19], that offers several security services for *enterprise*-type applications and that can be integrated with Spring MVC. Basically, this avoids the unauthorized usage of the application, increasing its safety.

Spring MVC[20]

Spring MVC is an *open-source* framework, based on the *design pattern model-view-controller*, realized in order to simplify the implementation and testing of web Java applications, being completely integrated with Spring. The MVC pattern provides a *loose coupling* by separating the application's logic in three components: input logic, business logic and UI logic. The model encapsulates the application data by using *POJO* objects, the view being in charge of displaying the model data. This consists in HTML that can be interpreted by the client's browser, the controller being in charge of processing the user requests, building the model that will be sent on to the view.

Based on the design pattern Front Controller[21], Spring web MVC is built around a central servlet, called Dispatcher Servlet[22], which treats all *HTTP requests and responses*.

Basically, after receiving the HTTP request, the servlet consults the *Handler Mapping* [23] in order to decide which Spring MVC Controller has to treat the request. Afterwards, it sends a request to the selected controller and consults a *View Resolver*[24] in order to map the request at the implementation of a specific view, a view that can be - or not - a JSP (*Java Server Page*).

JSP[25] is a technology used for the server side, common for the implementation of the view component from the MVC model. LAT consists of a series of JSP, populated bymeans of the models and mapped with the help of the controllers. For

[7]http://www.tutorialspoint.com/spring/spring_web_mvc_framework.htm

[8]http://getbootstrap.com/

[9]http://bootswatch.com/cerulean/

[10]https://www.datatables.net/extensions/tabletools/

[11]http://datatables.net/extensions/tabletools/api#fnResizeButtons

[12]https://spring.io/tools

[13]http://docs.spring.io/spring/docs/current/spring-framework-reference/html/aop.html

[14]http://www.tutorialspoint.com/spring/spring_overview.htm

[15]http://docs.spring.io/spring-framework/docs/3.0.x/reference/overview.html

[16]http://www.shaunabram.com/beans-vs-pojos/

[17]http://docs.spring.io/spring/docs/current/spring-framework-reference/html/beans.html

[18]http://docs.spring.io/spring/docs/current/spring-framework-reference/html/testing.html

[19]http://projects.spring.io/spring-security/

[20]http://www.tutorialspoint.com/spring/spring_web_mvc_framework.htm

[21]http://www.tutorialspoint.com/design_pattern/front_controller_pattern.htm

[22]http://www.tutorialspoint.com/spring/spring_web_mvc_framework.htm

[23]http://www.codejava.net/frameworks/spring/understanding-spring-mvc?start=4

[24]http://www.studytrails.com/frameworks/spring/spring-mvc-view-resolver.jsp

[25]http://www.oracle.com/technetwork/java/javaee/jsp/index.html

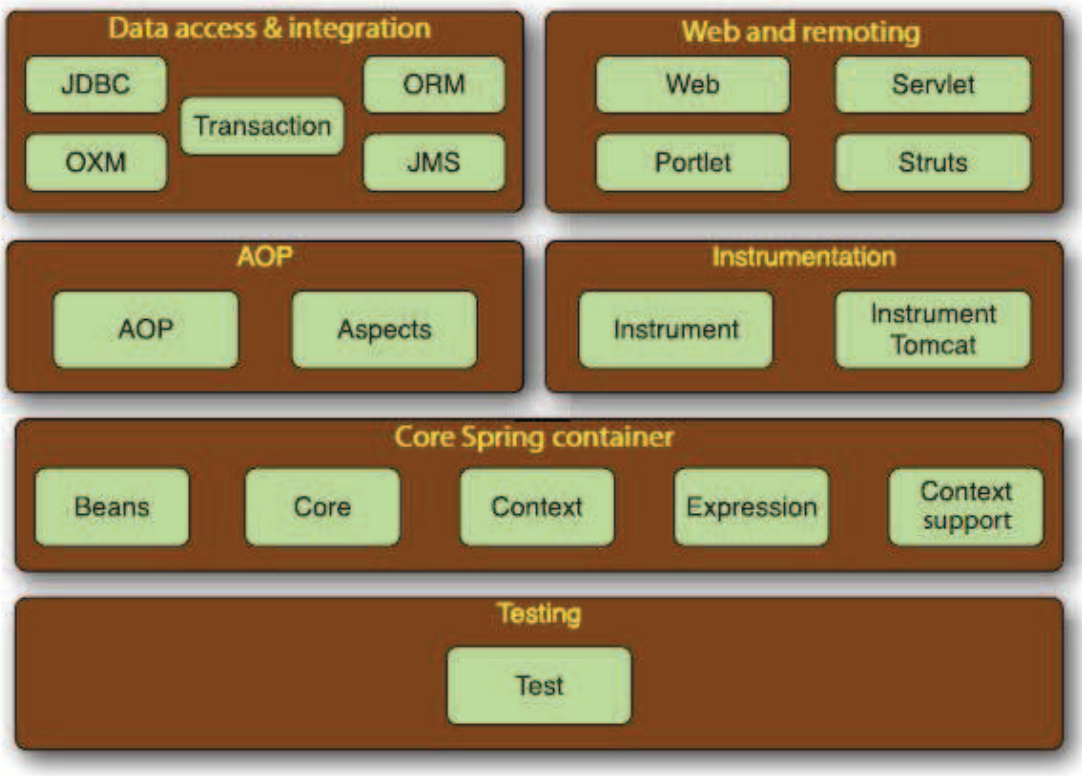

Fig. 1. Spring Framework Modules.

a better structuring of the pages, there was used *Apache Tiles 3.0*[26].

C. Apache Tiles

Based on the design pattern —textitComposite[27], *Apache Tiles*[28] is an open-source framework, built in order to simplify the development process of the user interface. This framework represents the simplest and most elegant solution for developing web applications based on any MVC technology.

The general architecture of LAT is based on *Apache Tiles*. The main template was sketched by defining the footer and the header, components that will be contained by each JSP of the application, to which the page's content (body) part was added, that will be replaced, according to the context, with particular information, specific for each action.

Below there is presented the main template of the application, built using *Tiles*, a template that was extended in order to add new content to the JSP defined in the main template.

```
 <tiles-definitions>
  <definition name="defaultTemplate"
template="/WEB-INF/views/template/default
/template.jsp">
   <put-attribute name="title" expression=
"No title"/>
   <put-attribute name="header" value="/WEB
-INF/views/template/default/header.jsp" />
   <put-attribute name="footer" value="/WEB
-INF/views/template/default/footer.jsp" />
  </definition>
</tiles-definitions>
```

Here is an example for extending the main template:

```
 <definition extends="defaultTemplate"
name="upload/viewUploadedContent">
<put-attribute name="body" value="/WEB-INF
/views/upload/viewUploadedContent.jsp"/>
<put-attribute name="title" expression=
"View uploaded content"/> </definition>
```

D. Apache Commons FileUpload

The loading of the files to be analyzed is done using Apache Commons FileUpload, API that allows the processing of a POST-type HTTP request whose content of the type "multipart / form-data" is, in accordance with the standard *RFC 1867*. It is an application that simplifies the process of uploading files, the returned response being easy to use within the web application.

E. Hibernate

For saving data there was used *PostgresSQL*, version 9.3, while for mapping Java objects with the tables from the

[26]https://tiles.apache.org/framework/whats-new.html
[27]http://www.tutorialspoint.com/design_pattern/composite_pattern.htm
[28]https://tiles.apache.org/

database and for mapping the data from Java with SQL-type data, there was used *Hibernate*.

Hibernate is an *open-source* framework that offers the possibility to interogate data by generating SQL calls to retrieve or search information stored in the database. Thus, by means of the facilities offered by *Hibernate*, the mapping of objects is done automatically, without the need to execute a manual conversion.

All these technologies and frameworks have simplified the development process, the structuring and realizing of the application being faster.

IV. Work methodology

LAT becomes a very useful application for the Romanian linguistic analysis, being able to determine the defining features of the type of speech that it represents [5].

The application (Figure 2) was focused on identifying the context of the analyzed text. It allows analysis of TXT, DOC, DOCX, ODT files. On entry the texts can have or not diacritics, with LAT we can make the correct diacritical marks, especially when the word can be confused with another Romanian word (e.g. *laturi* vs. *laţuri* vs. *lături*). These situations are not too rare.

LAT is structured in four modules:

(1) *spellchecking* - to identify errors in spelling or grammar, repeated words, and also identify if there is no punctuation or it used incorrectly, by integrating the open-source program LanguageTool[29] (LT) (Figure 3). Also, it can detect the text language.

(2) *POS-tagging* - to identify parts of speech categories and morpho-syntactic information of tokens, by using the POS-tagger tool [12].

For example: *oameni sfinţi*

```
 <W Case="direct" Definiteness="no"
Gender="masculine" LEMMA="om" MSD="Ncmprn"
Number="plural" POS="NOUN" Type="common"
id="55.39" offset="205">oameni</W> <W
Case="direct" Definiteness="no" Gender=
"masculine" LEMMA="sfânt" MSD="Afpmprn"
Number="plural" POS="ADJECTIVE" id=
"55.44" offset="229">sfinţi</W>
```

(3) identifying the speech type, using Nave Bayes classifier. To increase the speed of processing a classification request and, also, to improve performance, the @PostConstruct[30] method annotation was used. (4) creating diagrams after dividing text into paragraphs, a list of ChartDTOs objects (diagrams) is created. There are 3 types of diagrams supported by the app: *PieChart 3D*, *ColumnChart* by using *Google Charts* API[31] and *PolarAreaChart* by using *Chart.js* API[32].

[29]http://mvnrepository.com/artifact/org.languagetool/languagetool-core
[30]https://docs.oracle.com/javaee/5/api/javax/annotation/PostConstruct.html
[31]https://developers.google.com/chart/interactive/docs/dev/dsl_get_started
[32]http://www.chartjs.org/docs/

V. Conclusions and future work

LAT tools can be used successfully for speech analysis, including at lexical level by grammar checking, and spellchecking. Now, we are working to compare the results obtained by the program himself and the results obtained using a rule-based method. This method is based on pattern matching. We need a gold corpus of hundreds sentences, extracted from documents within different genres (a work in progress). We are also interested to identify the semantic level. Thus, the application should be extended by adding a new component responsible for to relate meanings to syntactic structure.

Acknowledgment

The LAT software has been developed by Iuliana Mariana Bejan from the Faculty of Computer Science "Alexandru Ioan Cuza" of Iaşi.

References

[1] Benveniste, E., *Problémes de linguistique générale*. Paris, Gallimard, pp. 118-131, 1966, (republished in 1974).
[2] Brown, G., Yule, G., *Discourse analysis*, Cambridge University, Press, Cambridge, 1983.
[3] Buraga, S.C., Cioca, M. Using XML Technologies for Information Integration within an e-Enterprise, 7th International Conference on Development and Application Systems DAS, undere the care of IEEE Romanian Section, pp. 343-348, 2004.
[4] Frunză O., Inkpen, D., and Nadeau, D., *A text Processing Tool for Romanian Language*. In Proceedings of the EuroLAN 2005 Workshop on Cross-Language Knowledge Induction, 2005.
[5] Gîfu, D., Cioca, M. Online Civic Identity. Extraction of Features, Procedia - Social and Behavioral Sciences, Volume 76, pp. 366371, 2013.
[6] Gîfu, D., Cioca, M. Detecting Emotions in Comments on Forums, in International Journal of Computers Communications & Control, 9(6), 694-702, 2014.
[7] Grivel, L., and Bousquet, O., *A discourse analysis methodology based on semantic principles - an application to brands*, Journalists and consumers discourses. In Journal of Intelligence Studies in Business 1, pp. 76-86, 2011.
[8] Halliday, M. A. K.An introduction to functional grammar, pp. 1-32, 1994.
[9] Maingueneau, D., *Genéses du discours*, Mardaga, Liége, 5, 1984.
[10] Petitjean, A., *Les Typologies textuelles*. In Pratiques, no. 62, pp. 86-125, 1989.
[11] Piolat, A. and Bannour, R., *An example of text analysis software (EMOTAIX-Tropes) use: the influence of anxiety on expressive writing*, Current Psychology Letters [Online], 2009.
[12] Simionescu, R., *UAIC Romanian Part of Speech Tagger*, resource on nlptools.info.uaic.ro, "Alexandru Ioan Cuza" University of Iaşi, 2011.

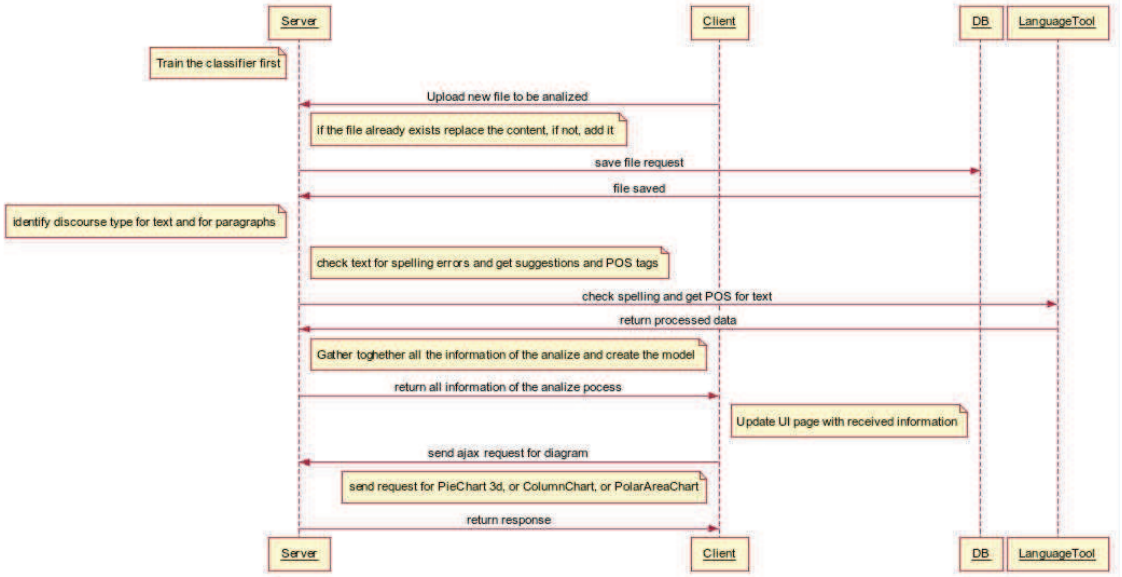

Fig. 2. Work session with LAT.

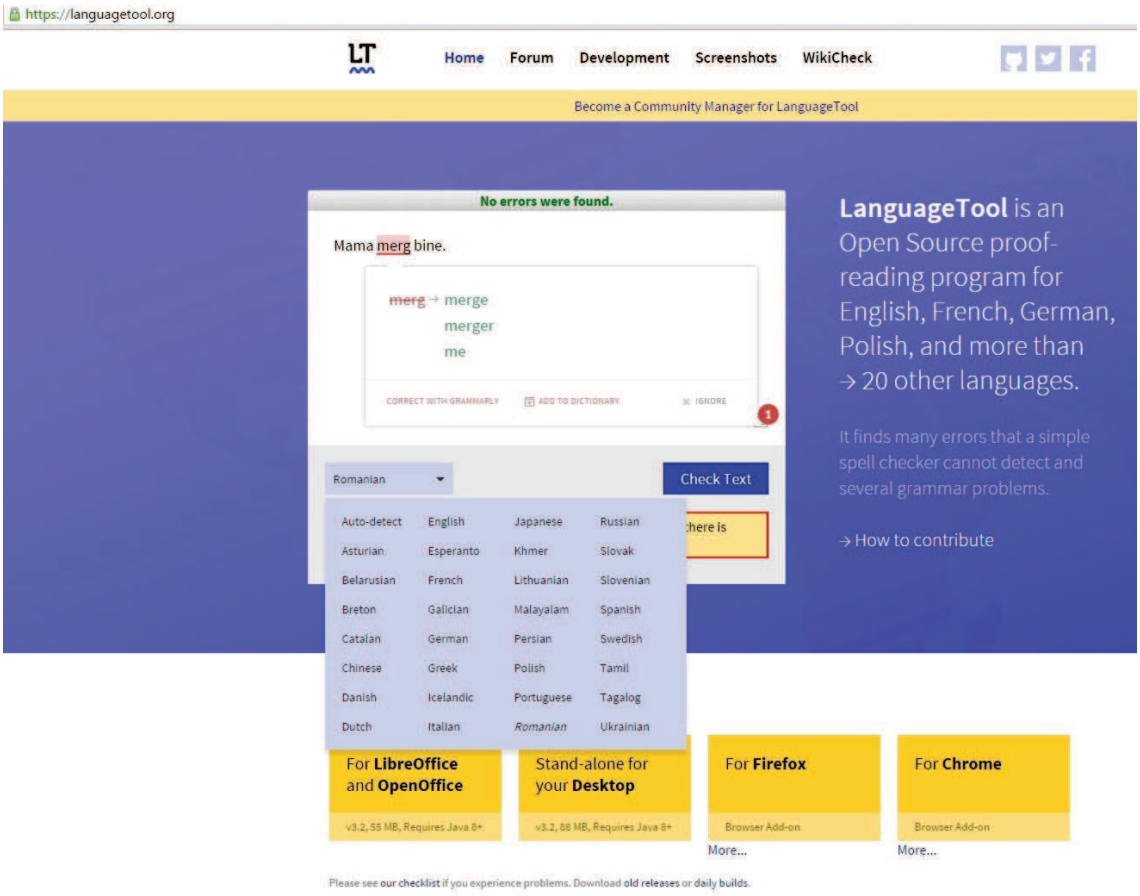

Fig. 3. Work session with LT.