Simpatico: A Text Simplification System for Senate and House Bills

Miguel Collantes
College of Computer Studies
De La Salle University
2401 Taft Ave., Malate, Manila
miguel_collantes@dlsu.edu.ph

Maureen Hipe
College of Computer Studies
De La Salle University
2401 Taft Ave., Malate, Manila
maureen_hipe@dlsu.edu.ph

Juan Lorenzo Sorilla
College of Computer Studies
De La Salle University
2401 Taft Ave., Malate, Manila
juan_sorilla@dlsu.edu.ph

Laurenz Tolentino
College of Computer Studies
De La Salle University
2401 Taft Ave., Malate, Manila
laurenz_tolentino@dlsu.edu.ph

Briane Samson
College of Computer Studies
De La Salle University
2401 Taft Ave., Malate, Manila
briane.samson@dlsu.edu.ph

ABSTRACT

In the Philippines, public participation in the creation of Senate and the House bills is very minimal. This could be the case because of the lack of accessibility of an ordinary citizen to take part in the lawmaking process and the complexity of the terms and grammar rules being used in legal proceedings called legalese. eParticipation is seen to be a solution to increase public participation but it only solves the process' accessibility and not the complexity of the bills. Simplification solves this problem by transforming technical jargons and complicated phrases into words or phrases that are easier to understand. Also, existing simplification systems do not cover Philippine Senate and House bills as part of their domain. This research focuses on developing a text simplification system using lexical and syntactic simplification method for Philippine Senate and House bills called Simpatico. It aims to simplify legalese to plain English, which a majority of the Philippine population can understand. The system is based on a mix of previous simplification systems and makes use of different tools in order to accomplish certain tasks for simplification. It is composed of 3 major components: the preprocessing module, lexical simplification module, and the syntactic simplification module.

Categories and Subject Descriptors

I.2.7 [Artificial Intelligence] Natural Language Processing – *language parsing and understanding, speech recognition and synthesis, text analysis.*

Keywords

eParticipation; Philippine Senate and House bills; Lexical Simplification; Syntactic Simplification.

1. INTRODUCTION

In the Philippines, there is a lack of public participation in the creation of the Senate and House bills. The complexity of legal documents remain as an obstacle to improve good governance and citizen participation [2]. Documents such as the Philippine Senate and House bills require comprehensive knowledge of legalese and the English language from the reader.

1.1 Legalese

Legalese is a formal and technical language used by legislators to constitute a bill and contains terms and grammar rules which can only be understood by certain individuals belonging to a specific functional literacy level and type. Legalese is full of law-Latin and Norman-French words/terms.

The key features of legalese are :

- **Terms of Art** – consists of terms that are quite technical and unfamiliar to the layman. Mostly derived from French and Latin. Ordinary words are given different meanings (ex. *considerations* means contract in legalese).
- **Lack of punctuation** – punctuations were considered as ambiguous and unimportant. Mostly used in conveyances and deeds. Modern legal drafts does include punctuation to help readers.
- **Doublets and Triplets** – a single legal concept is conveyed using two or three words in order to create the sense of completeness. (ex. *terms and conditions).*
- **Unusual word order** – French grammatical structures influenced English legal writing.
- **Unfamiliar pro-forms** – pro-forms (ex. *the same, the said, etc.*) are used to replaced nouns but the nouns are still kept in legalese.
- **Pronomial adverbs** – hereof, thereof, etc. are used in legalese to avoid repeating names or phrases.
- **-er, -or, and –ee name endings** – words and titles are added with the name endings to represent the reciprocal

and opposite relationship.

- **Phrasal verbs** – usually used in quasi-technical sense such as *enter into, put down, write off,* etc.

1.2 Text Simplification

Text simplification is a process that reduces the syntactic or lexical complexity of a text with the original meaning and information of the content is preserved [17]. It can rewrite text into simpler versions which in turn makes the information available to a much broader audience such as the non-native speaker of its selected domain. A system using text simplification aims to make text easier for its target user to comprehend. The process includes first, the construction of the structural representation of a text then second, the application of a sequence of rules for identifying and extracting the words to be simplified [5].

Text simplification can be used in multiple domains such as English medical literature by SimText and Supreme Court decision documents by Elexsim. Although text simplification can be used in different domains, existing systems can only simplify text that is specific to their own domain. SimText is capable of simplifying English medical literature but was incapable of simplifying text outside its own domain after testing.

1.2 Related Systems

SimText is an example of a text simplification system that converts collegiate level texts to high school level texts using NLP techniques (Damay). SimText uses lexical and syntactic simplification techniques to simplify medical texts. Simtext uses A Nearly New Information Extraction (ANNIE) system to perform preprocessing on an input document and then applies a simplification module to the preprocessor's output. SimText's ability to simplify different domains of text is based on the knowledge sources stored in the system. Text Simplification for Children (TSC) is another system that uses lexical and syntactic simplification on a text. TSC brings down the text's readability level to that of a child [Belder] and applied those simplification techniques in news and encyclopedia articles.

While SimText and TSC uses both lexical and syntactic simplifcation to reduce the readability level of a text, Elexsim is an unsupervised English lexical system with multiword expressions handling. Elexsim applies multiword expression extraction after it's preprocessing module before applying lexical simplification [11]. Another system, Putting It Simply (PiS), is a two-phase system consisting of rule extraction and then lexical simplification afterwards [Elhadad]. PiS makes use of a context aware method in applying lexical simplifications.

Lexical simplification is present on every text simplification systems. FASTSUBS is a search algorithm that seeks to improve the performance issues caused by exhaustive and heuristic algorithmgs used in lexical simplification systems. It finds the K most likely lexical substitutes based on an N-gram language model instead of computing the probability of every word before deciding the top K [17].

1.3 eParticipation

eParticipation makes it convenient for citizens to participate in the government's decision-making process. It makes information of political areas more accessible and simpler. It is being used in Europe and U.S.A to encourage citizens to participate in legislative matters. eParticipation can range from print and electronic forms, public conferences and hearings, to online forums and voting polls. VoteTocracy, an American website, aims to give the public access to bills and manually simplifies them in a jargon-free manner through their in-house analysts and and software engineers who provide them data sourcing, aggregation, and information extraction [11].

2. METHODOLOGY

Various existing avenues are available for accessing Senate and House bills; however, the proposed system will only make use of text from bills available in the official website of the Senate of the Philippines (http://www.senate.gov.ph) as they provide electronic copies of bills in PDF format which can be used in the system. Furthermore, the said website hosts a great deal of information about the Philippine Senate, committees, secretariat and most importantly, they provide an up to date collection of legislative documents such as bills and laws.

In order for the system to be able to work with the bills more effectively, it first needs to be preprocessed so to make a representation of the bill that the system can easily understand and process. Currently, there are a lot of existing tools and techniques for preprocessing which involve removing unwanted elements in the text input such as punctuations, capital letters, and stopwords among many others. However, the choice of preprocessing tools to be employed in the system will only be limited to the ones freely available on the web namely CoreNLP.

Text simplification aims to transform complex information to its equivalent simpler versions. There are three types involved in simplifying text which are lexical, semantic, and syntactic. For this research, the simplification techniques to be employed in the system are only limited to Lexical and Syntactic simplification. Lexical simplification involves changing a complicated word into a more simple, easy to understand word. For example, humongous or enormous will be huge or big under lexical simplification. On the other hand, syntactic simplification involves separating a complex sentence into smaller simpler sentences [6].

The system will be evaluated and tested through the metrics that will be defined. In addition, people who have little knowledge on political jargon or anyone who is not familiar with the political field to be able to know if the simplified text is indeed easier to read than the original will also read the output to be able to determine whether the simplification made was indeed effective.

3. SYSTEM OVERVIEW

Simpatico is composed of four major modules, the preprocessing module, the multiword expression module, the lexical simplification module, and the syntactic simplification module. Each module will have its own specific parameters and functions that will help in the simplification of the text. The system will first go through the preprocessing module which will prepare the text for simplification. Afterwards, the system will undergo multiword extraction which is used to gather and process multiword expressions in the text. Next, the lexical simplification module will gather and simplify complicated or complex words and phrases into their simpler, easy to understand counterparts. Finally, the system will undergo syntactic simplification which will split complex sentences into simpler sentences.

3.1 Preprocessing Module

The preprocessing module is comprised mostly of the Stanford CoreNLP toolkit. It makes use of a pipeline system wherein raw text is put into an Annotation object and a series of Annotators add information in an analysis pipeline, which can be outputted in either XML or plain text format. Simpatico accepts sentences from legal documents in plain text format as input. The system then gets the results of the toolkit as plain text using the following annotators: tokenize, ssplit, pos, lemma, parse, and dcoref. The first part is the tokenizer (see figure 3-1), which transforms the text into a set of tokens. This step is vital to the system because this is a primary procedure before any other processing can be done on the text. The tokenizer is used for splitting the text into a sequence of tokens that can be processed by the next set of processes.

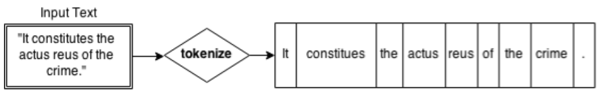

Figure 3-1. Diagram showing the conversion of a string input into tokens.

Following the tokenizer, the sentence splitter (SSPLIT) will split the whole text into individual sentences. The sentence splitter is key to improving the performance because it will process the text per sentence instead of as a whole.

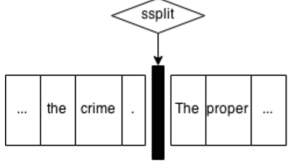

Figure 3-2. Diagram showing SSPLIT splitting tokens after a punctuation mark.

Afterwards, the Part of Speech (POS) Tagger will mark the tokens based on their parts of speech (see figure 3-3). Stanford CoreNLP makes use of the Penn tree tag set for marking each token with the appropriate part of speech name.

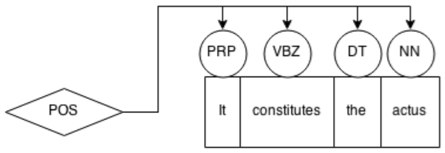

Figure 3-3. Diagram showing POS tagging each token with a speech tag.

Following the POS tagger is the Lemmatization, which gets the token's base word or root word (see figure 3-4). Lemmatization is also vital for improving the system's performance because it could help the system find a replacement words faster during simplification.

constitues
LEMMA
constitue

Figure 3-4. Diagram showing the word "constitues" and its lemma "constitute".

Stanford CoreNLP also generates a parse tree. It will be used for syntactic analysis in order to generate constituent and dependency parse trees. These parse trees will be used by the syntactic simplification module in order to simplify sentences syntactically. An example of a parse tree is shown in figure 3-5.

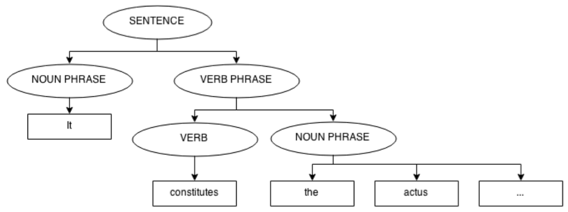

Figure 3-5. Diagram presenting a constituency parse tree.

Coreference (DCOREF) graphs are also generated in the preprocessing module which will also be used by the syntactic module. DCOREF is capable of pointing a mention to a target which is seen in figure 3-6.

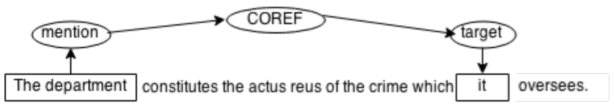

Figure 3-6. Diagram showing "the department" is co-referenced to the word "it".

The result of tokenization, lemmatization, and part of speech tagging will then be stored into a package called *language* which contains the classes *Word* and *PreSentence* wherein *Word* represents a word inside the text while the *PreSentence* class represents a sentence inside the text. The Word class contains attributes that would describe a word for the other modules to use such as a word's substitute, it's lemma, and it's original word as seen in figure 3-7.

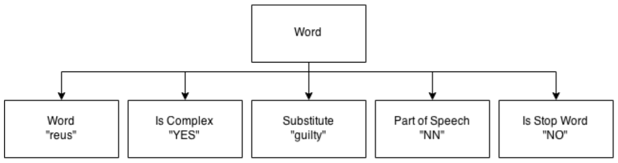

Figure 3-7. Diagram showing the structure of the Word object. (Not all attributes of the object Word are shown).

The PreSentence class contains an array list of the Word class (see figure 3-9). The lexical and syntactic modules process a text by sentences and this makes PreSentence an important object of the system.

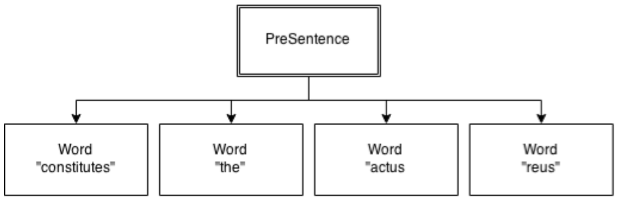

Figure 3-8. Diagram showing the structure of the PreSentence object which consists of an ArrayList of the object Word.

3.2 Multiword Extraction Module

Because the preprocessing module is only limited to recognizing and marking single word language constructs and expressions, the multiword extraction (MWE) module will identify and extract multiword expressions in the text (see figure 3-9). Before multiword extraction can begin, the text must first be properly processed before anything could be done. Hence, the need for preprocessing using the Stanford CoreNLP. The Multiword Extraction module will be handled by the jMWE toolkit [8]. The toolkit will make use of the default corpus provided by jMWE. And because the default corpus lacks expressions in the legal domain, a list containing legalese MWEs will be manually appended to the default corpus. This list is in a text format that could be read by the system and it includes latin terms.

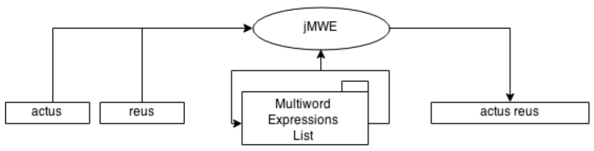

Figure 3-9. Merging of a multiword expression into a single Word object

3.3 Lexical Simplification Module

The lexical simplification module first identifies if there are a set of tokens eligible for direct substitution. Their eligibility is based on a list of Latin terms such like “contra”, “persona non grata”, etc. and surplus words which are are generated by compound prepositions such as “by means of” instead of the simpler “by” or “in favor of” instead of “for”. The list is provided by WordNet and a custom made law dictionary, which contains latin and legal terms and includes their definition. Using this law dictionary, if a token (or a set of tokens) is found existing inside the dictionary, then a corresponding synonym, which can also be found on the said lists, will be substituted to the token(s). Afterwards, tokens that do not require complexity analysis will be identified and ignored by the system in order to optimize performance. Such tokens include tokens within parentheses, tokens within quotation marks, numeric characters, and tokens tagged as proper nouns.

Complexity analysis is then carried out by the system through the use of SUBTLEX-US corpus, particularly their *Zipf Scale* values. A Zipf scale has four properties: 1) a logarithmic scale, 2) should have a relatively few points (Likert rating scale from 1 to 7), 3) middle value of the scale should separate the low-frequency words from the high-frequenct words, and 4) should have a straightforward unit. Zipf has a logarithmic scale which goes from 1 (very low frequency words) to 6 (very high frequency content words) or 7 (a few function words, pronouns, and verb forms like “have”). Each word is searched in the corpus and words with a zipf value of less than 4 or that are not found on the corpus is marked as complex by the system.

Table 3. Table showing the Zipf Scale’s value, its corresponding frequency per million words, and example words that belongs to their scale (Heuven et al., 2014).

Zipf Value	fpmw	Example
1	.01	Antifungal, bioengineering, farsighted, harelip, proofread
2	.1	Airstream, doorkeeper, neckwear, outsized, sunshade
3	1	Beanstalk, cornerstone, dumpling, insatiable, perpetrator
4	10	Dirt, fantasy, muffin, offensive, transition, widespread
5	100	Basically, bedroom, drive, issues, period, spot, worse
6	1,000	Day, great, other, should, something, work, years
7	10,000	And, for, have, I, on, the, this, that, you

Words with zipf values of 4-7 are considered high frequency words and words with zipf values of 1-3 are low frequency words (see table 3). The implementation follows this concept as words with zipf values of 1-3 are more likely to be complex as stated by [9]. Consequently, each token tagged as complex is checked for their appropriate synsets using available on WordNet through RiWordNet. Choosing which synset is appropriate is done through word sense disambiguation using Babelfy. Afterwards, the word with the highest frequency on the returned synset will be selected as the substitute of the complex word. The tense of the complex word will then be applied to the substitute word through

SimpleNLG.

3.3 Syntactic Simplification Module

The final module, the syntactic simplification module, will simplify complex sentences into simpler sentences. The syntactic simplification module is composed into two parts, syntactic analysis and transformation.

Syntactic analysis is first executed on the final text output of the lexical simplification module in order to generate the typed dependencies and the constituency parse trees of each sentence in the input text. The same Stanford CoreNLP pipeline that is used in the preprocessing of the input text is utilized once more to perform the analysis in order to conserve resources. The resulting dependencies and parse tree will then be passed to the modules responsible for splitting the sentences in the following order: compound sentence splitting, relative clause splitting, appositive splitting, and finally, passive to active voice conversion. Each module returns their respective results in String format and before being passed to the next module, it is first analyzed again by the Stanford CoreNLP in order to get the new constituency parse tree and the new typed dependencies since it is likely that the parse trees have been altered by the previous module and may not represent the actual syntactic make up of the sentence anymore.

The next step is to transform the text by splitting compound sentences, simplifying relative clauses, simplifying appositives, and passive voice to active voice conversion. Compound sentences are detected and then split. These sentences mainly consist of two independent clauses conjoined together by a conjunction. A compound sentence that was split will require relative clause simplification. A sentence that contains a relative clause that is greater than or equal to the simplification treshold of 25 words, will be split into more sentences as recommended by [18]. Appositives are then scanned in each sentence. Appositives that were found are modified starting from the parent of the noun phrase it contains and then goes all the way down in the tree. A rule-based approach is then performed in order to perform passive to active voice conversion. Sentences that are in passive voice are converted into active voice through the manipulation of the tree, specifically Stanford's CoreNLP's constituency parse tree.

4. Results and Discussions

To examine the effect on readability of Simpatico's syntactic and lexical simplification module, 50 sentences were randomly sampled from various Senate and House bills in different stages of the legislative process. The readability scores of syntactically simplified sentences were taken and compared to the readability scores of non-simplified sentences, lexically simplified sentences, as well as both lexically and syntactically simplified sentences. The readability formulas used were the Gunning-FOG Index, SMOG Index, and the Flesch-Kincaid Reading ease formula. However, these readability indexes only use either word length, word count, syllable count, or sentence count as variables and thus cannot tell anything about the grammaticality, cohesiveness, or meaning of the simplified sentence. Table 4-1 shows the readability scores of the original as well as the simplified sentences. The readability levels and text statistics are taken from online readability tools readability-score.com and read- able.com.

Table 4-1. Readability grade level before and after simplification. Flesch-Kincaid Grade Level (F- K), Gunning-Fog Score (G-F), SMOG Index (SMOG), Average of all scores (AVG), Original or unsimplified text (Original), Syntactic and Lexical Simplification (Syn-Lex), Syntactic simplification only (Sny-only), Lexical Simplfication only (Lex-only).

Readability measure	Original	Lex-only	Syn-only	Syn-Lex
G-F	28.9	26.6	25.2	23.2
SMOG	20.1	18.5	18.2	16.8
F-K	26	24	22.4	20.7
AVG	**25**	23	**22**	**20.2**

According to table 4-1, the simplified documents that underwent syntactic and lexical simplification yielded better readability compared to the original text as the average grade level decreased by 5. This appears to be also true with Syn-only and Lex-only as their average readability grades decreased by 3 and 2 from the original text respectively. The results of Lex-only suggests that even lexical constructs can contribute to the overall readability of the text. As opposed to Lex-only, Syn-only underwent a slight increase in the character count. This likely because of the additional words that results in the duplication of a noun phrase as well as the addition of other words such as determiners in the simplification of relative clauses and appositives. Out of the 50 sentences simplified by the system, only a total of 14 sentences were syntactically simplified. 9 of which are relative clauses and none of which are appositives.

The text statistics of Lex-Syn is somewhere between the Lex-only and the Syn-only. A notable difference is the much lower average words per sentence which is generally the result of splitting sentences into shorter ones and the simplficiation of complex terms to more simple terms (since complex terms tend to be more longer). However, despite the reduction in grade level, the Syn- Lex results are still far from the goal of high school level English with a grade level of 20.2. The ideal score would be between 7-12 as it pertains to both Junior and Senior high school level, our target audience, in our education system here in the Philippines today (K-12). Thus, the results on readability are still undesirable and still needs improvement.

5. CONCLUSIONS

This research aims create a system that could simplify Senate and House bills to promote eParticipation in in the Philppines. Furthermore, the research aims to make Senate and House bills understandable to those without any legal background by reducing the complexity of sentence structures and terms. In addition, majority of the Filipino people are not aware that they could actually take part in the law making process by exercising their opinions on bills. For example, the Guingona project or the Crowdsourcing Act of 2013 (theguingonaproject.com) aims to provide a means for the Filipino public to participate in the law making process through an online portal where they can voice their opinions and suggestions.

The readability of the text simplified by the system was evaluated using popular readability metrics and the correctness of the simplified texts underwent human evaluation through the means of a survey. Results on readability indicate that the most of the simplifications done by the system resulted into little or no improvements in readability. In terms of correctness, the results of the survey indicate that 50% of the lexical simplifications made by the system are agreed upon by the respondents while there is only a minor change in the meaning of the simplified text. Based on the readability metrics, the system did not meet the needed readability score in order to be easily be read by people who are at most highschool graduates. However in the survey, it shows that the respondents generally prefer the output of the system. It is concluded that the system did meet the requirements based on the survey results and the readability indexes to are not enough to completely evaluate the system since it does not directly correlate with the complexity of the text.

6. REFERENCES

[1] Ahmed, I., Bilal, H., Khan, T. & Nawaz, N. (2013). Language of law: stylistic analysis of a legal document. International Journal of Research and Management, Issue 3, Vol. 2. Dept. of English, University of Sargodha, Pakistan.

[2] Balbin, J.P., Landrito, M., Maglalang, C., & Reyes, K. (2010). Senate bill summarizer in a social network environment. Unpublished Undergraduate Thesis. De La Salle University, Manila, Philippines.

[3] Biran, O., Brody, S., & Elhadad, N.(2011). Putting it simply: a context-aware approach to lexical Simplification. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics:shortpapers, pages 496–501, Portland, Oregon, June 19-24, 2011.

[4] Chandrasekar, R., Doran, C., Srinivas, B. (1996). Motivations for text simplification. University of Pennsylvania, Philadelphia, U.S.A.

[5] Coleman, M. & Liau, T. L. (1975). A computer readability formula designed for machine scoring, Journal of Applied Psychology, Vol. 60, pp. 283–284.

[6] Damay, J., Lojico, G., Lu, k., Ong, E., & Tarantan, D.(2007). Simplifying text in medical literature. Journal of Research in Science, Computing and Engineering, 4(1):37-47, April 2007. De La Salle University. Kulkarni, N. & Finlayson, M. (2011).

[7] DuBay, W. H. (2004). "Judges Scold Lawyers for Bad Writing". Plain Language At Work Newsletter (Impact Information) (8).

[8] Finlayson, M.A., Kulkarni, N. (2011) Detecting Multi-Word Expressions Improves Word Sense Disambiguation, Proceedings of the 8th Workshop on Multiword Expressions, Portland, OR. Pp. 20-24.

[9] Heuven, W., Mandera, P., Keuleers, E., Brysbaert, M. (2014). SUBLTEX-UK Subtlex-UK: A new and improved word frequency database for British English. Quarterly Journal of Experimental Psychology, 67, 1176-1190.

[10] Kincaid, J.P., Fishburne, R.P., Rogers, R.L., & Chissom, B.S. (1975). Derivation of new readability formulas for navy enlisted personnel. Research Branch Report 8-75. Chief of Naval Technical Training: Naval Air Station Memphis.

[11] Kraljic, D. (2011). VoteTocracy. Retrieved from www.votetocracy.com.

[12] Lim, J., Sagum, R., & Villaflor, B. (n.d.). ELEXIM: An unsupervised English simplification system with multiword expressions handling. Unpublished Undergraduate Thesis. Polytechnic University of the Philippines, Manila, Philippines.

[13] Manning, C., Surdeanu M., Bauer, J., Finkel, J., Bethard, S., McClosky, D. (2002). The Stanford CoreNLP Natural Language Processing Toolkit. In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp.55-60.

[14] McLaughlin, G. (1969). SMOG grading – A new readability formula. Journal of Reading, 639-646.

[15] Miller G. A. (1995). WordNet: A Lexical Database for English. Communications of the ACM Vol. 38, No. 11: 39-41.

[16] Moro, A., Raganato, A. & Navigli, R. (2014) Entity linking meets word sense disambiguation: a unified approach. Transactions of the Association for Computational Linguistics (TACL), 2, pp. 231-244, 2014.

[17] Siddharthan, A. (2002). An Architecture for a text simplification system. In Proceedings of Language Engineering Conference, Washington DC, USA, 2002, Natural Language and Information Processing Group Computer Laboratory University of Cambridge.

[18] Wydick, R. (1978). Plain English for Lawyers. 66 Cal. L. Rev. 727.

[19] Yuret, D. (2012). FASTSUBS: an efficient and exact procedure for finding the most likely lexical substitutes based on an n-gram language model. IEEE Signal Processing Letters, VOL. 19, No. 11, Nov. 2012