# An Approach to Extract a Conceptual Model from Natural Language Specifications

Leandro Antonelli[1], Juliana Delle Ville[1], Matias Alfredo Adorno[1], Letizia Paola Ballestero[2], Santiago Andrés Cecconato[1], Alejandro Fernandez[1 3], Gregorio Maclen[2], Giuliana Maltempo[1], Juliana Edda Mattei[2], Luciana Tanevitch[1], Diego Torres[1 3 4],

[1]Lifia, Fac. de Informática, UNLP, La Plata, Bs As, Argentina
[2]Fac. de Informática, UNLP, La Plata, Bs As, Argentina
[3] Comisión de Investigaciones Cientificas (CICPBA)
[4] Departamento de Ciencia y Tecnología, UNQ
{leandro.antonelli, juliana.delleville, matias.adorno, santiago.cecconato, alejandro.fernandez, giuliana.maltempo, luciana.tanevitch, diego.torres}@lifia.info.unlp.edu.ar
{letiziapaolaballestero, gregorio.maclen, juliana.mattei.29792}@gmail.com

**Abstract**— Requirements engineering is a critical phase in software development. Errors committed in the requirements become costly problems later on. Artifacts described in natural language are the most suitable for the participants of the process: experts and the software development team. But, natural language can have some issues because of the diversity of participants' writing and reading specifications. Thus, it is necessary to guide and assist the production of these artifacts. This paper proposes an approach to acquire specifications collaboratively checking the writing style and synthesizing a conceptual model to summarize the specification to help participants to have an overview of the knowledge gathered in order to provide a consistent and coherent model. This paper also describes a prototype that uses natural language processing and artificial intelligence tools to support the process.

**Keywords**- requirements; conceptual model; natural language.

## 1. Introduction

Requirements engineering is a critical stage of software development. Errors made at this stage can cost up to 200 times to repair [5]. In this stage, two different groups of people participate: clients that state their needs and knowledge to be included in the software application, and the development team that has to understand the specification to provide a software application that satisfies it. Experts and development team belong to different worlds and use different languages [28]. The experts use the language of the domain while the development team uses a computer science language. To cope with this communication gap it is important to use artifacts

Proceedings of the 26th Workshop on Requirements Engineering (WER23), August 15-17, 2023, Porto Alegre, RS, Brazil.
**DOI**: 10.29327/1298356.26-12

in natural language [23]. Nevertheless, natural language specifications can have many defects [17] [25]. For example, they can have an ambiguity that can appear in different ways. One type of ambiguity is vagueness; that is, expressions can have situations that are not defined (for example, borderline cases) [4].

Kernel sentences [18] [10] also known as basic sentences, are declarative constructions, in active voice, always affirmative with only one verb. Requirements specifications generally are described using long sentences, and this characteristic increases the possibility of the presence of ambiguity in them [14]. Thus, with the objective of simplifying the sentences and reducing their complexity, it is very useful to use kernel sentences to specify requirements [7]. Kernel sentences can help to reduce some defects in the requirements specification, but there are some other issues like duplication or inconsistency, that need a model synthesis technique to summarize and organize the requirements [35]. Moreover, the synthesis of a large set of requirements is a good technique to cope with the complexity [20] [13], and to improve the quality of the requirements regarding cohesion and consistency [11].

This paper proposes an approach that receives a natural language specification as input and provides some feedback by analyzing the text according to the kernel sentences writing style, and derives from the text a conceptual model. It is important to mention that the input of the approach (the specification) should not have any specific format. It should only be a specification in natural language. Then, the conceptual model obtained as output includes concepts, attributes and relationships. The rest of the paper is organized in the following way. Section 2 describes some background about kernel sentences. Section 3 details our contribution, the proposed approach. Section 4 describes the tool to support the process. Section 5 reviews some related work. Finally, Section 6 discusses some conclusions.

## 2. Kernel Sentence

A kernel sentence is a simple construction with only one verb. It is also active, positive and declarative. For example, let's consider the kernel sentence "The client deposits on the account." It states that the subject "client" performs an action ("deposits") on a certain object ("account"). Another example of a kernel sentence is "The client of the bank opens an account". This sentence has the same structure while it describes a different action ("opens"), and the subject is described in more detail ("client of the bank"). This is a very simple example, and we know that "client" and "client of the bank" refers to the same subject, but in a complex new domain, this synonym could not be so easy to identify. Now, consider a couple of sentences that are not kernel sentences. The first one, "The client deposits and adds interest into the account", has two verbs. And it is no completely correct. It is right that the client deposits money into the account, but the interest is added by the bank and not by the client. That is why simple (kernel) sentences are better for specifying software. The second one, "The account is opened" is written in passive voice, and it also misses the real subject who performs the action. It could be the client, but it also could be the company that the client works for and decided to open an account to pay his salary. These are two simple examples of how can kernel sentences writing style help to improve specifications.

## 3. The Proposed Approach

The proposed process has the objective of providing a conceptual model that summarizes a set of artifacts specified through natural language. The proposed approach analyzes every sentence to obtain concepts, relations and behavior. As an intermediate result, the proposed approach suggests checking some characteristics of the sentences in order to improve their description according to the kernel sentences philosophy. The process proposed considers two different activities: (i) kernel sentence checking, and (ii) conceptual model derivation. The specification is the input of the approach, which is provided to the kernel sentence checking activity. As a result of this activity, some feedback is provided to the experts. Then, the conceptual model activity produces the output. The proposed approach provides some rules to identify concepts, relationships and behavior. Moreover, two sets of words (or expressions) are considered to help the application of the rules: the set of candidate concepts and the set of excluded concepts (both sets can also be empty).

Let's consider the following example with one specification that does not satisfy kernel sentence principles: "The client of the bank opens and closes an account. The client deposits and withdraws in the account." The proposed approach will provide the feedback that both sentences have more than one verb. Then, considering that the set of the candidate and the excluded concepts are empty, the approach will identify the following concepts: "client" and "account". The concept "client" will have no behavior and the concept "account" will have the behavior: "open", "close", "deposit", and "withdraw". Finally, the concept "client" will have an association with the concept "account". Figure 1 summarizes the proposed approach and the example.

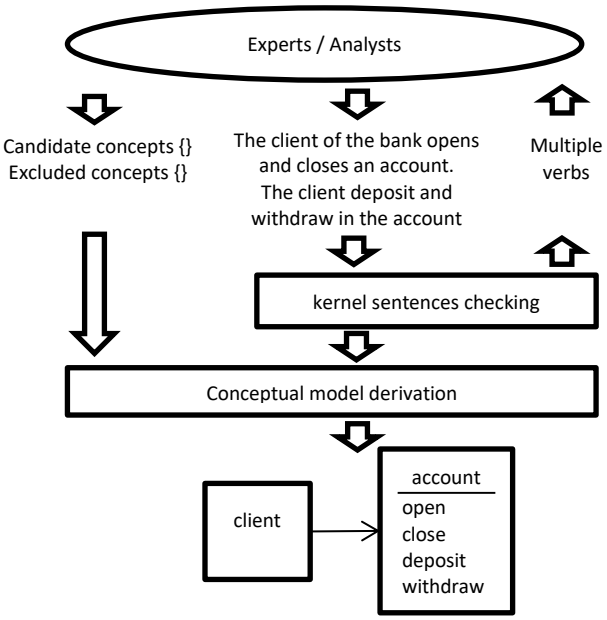


**Figure 1.** Simple derivation of the conceptual model

The first activity, kernel sentence checking, relies on analyzing the structure of the sentences used in the specification to assess whether they satisfy the philosophy of the kernel sentences or not. Four checking are performed: (i) tacit subject, (ii) multiple verbs, (iii) passive voice, and (iv) the presence of adjectives and adverbs. The second activity consists of deriving the conceptual model, which is performed through the identification of the following elements: (i) concepts, (ii) behavior, and (iii) relationships (two types of relationships: composition and association).

The first task, the identification of concepts, relies on identifying the nouns [3] that can be found as the subject and the direct object of the sentence. For example, "The client opens an account" sentence has two nouns "client" and "account", and both should be identified as concepts. This task is enriched with two sets provided by the experts: a set of candidate concepts and a set of excluded concepts. Since requirements use domain-specific expression it is valuable for these sets to avoid the biases of the automatic identification [13]. These candidate and excluded concept sets are simple lists of words and expressions that are used in combination with the noun identification to identify complex expressions as noun or to compare with the nouns identified to discard it. The prototype described in the following section uses these lists to train a machine-learning component. Let's consider the sentence, "The client of the bank opens an account". There are two nouns in the subject: "client" and "bank"; thus, the rule will identify both as different concepts. Nevertheless, the concept "client of the bank" is a concept including two nouns. Thus, candidate concepts set should include "client of the bank" to identify it. Let's consider the sentence "The client losses his card," where the concept "card" is beyond the scope of the system. To avoid being identified, "card" should be included in the excluded concepts set.

The second task, the identification of behavior, relies on identifying verbs in between two concepts. Particularly, the verb is a behavior of the second concept [3]. Let's consider the sentence "The client opens an account" the verb "opens" should be defined as a behavior associated to "account".

The third task, the identification of relationships, can be split into two subtasks: the identification of composition and association. The composition can be identified by looking for concepts (nouns) that appear on the same side of the verb [19]. Let's consider the sentence, "The client of the bank opens an account". If "client of the bank" is not added as a concept candidate set, two concepts are identified "client" and "bank". Since both appear in front of the verb, "client" should be part of "bank" (the first noun is part of the second one). Association can be identified by looking for concepts (nouns) that appear on different sides of the verb [3]. Let's consider the sentence "The client opens an account"; the concept "client" is associated with the concept "bank".

## 4. Tool Support

A software prototype that can be used to support the application of the proposed approach was implemented. The prototype is a web application implemented following a service-oriented architecture. The core of the application and its services are implemented in Python [29], while the web components are implemented with

Django [12], and the APIs are implemented with Flask [15]. Python [29] is also used to communicate to the Spacy [33] and the NTLK [26] libraries used to deal with natural language processing, as well as TensorFlow [34] and Sklearn [32], the machine learning components. These two technologies are particularly used to improve the identification of concepts as a complement of the rules. They are trained with the candidate and excluded concepts set. Finally, the framework Plant UML [27] is used to display the model. It is important to mention that the prototype includes many libraries because is a framework that can be extended to provide different functionality easily. The prototype implements two roles of users, (i) administrators of the projects and (ii) experts/analysts. The administrator of the projects can create projects by adding users and defining the type of artifacts they are going to use. The experts and analysts can add contributions, that is, add new artifacts or edit artifacts created by another one. While the user types the text, the application checks whether the sentence satisfies the characteristics of a kernel sentence. If something wrong is detected, the application informs the issue.

## 5. Related works

Shuttleworth et al. [31] propose a semi-automatic approach to generate a conceptual model from descriptions of a phenomenon. They use pattern-based grammatical rules and an NLP dependency parser as we do. Nevertheless, they do not use machine learning techniques. Robeer et al. [30] propose to automatically derive conceptual models from User Stories. Their conceptual model is not as rich as ours. Lucassen et al. [24] are also concerned about providing a conceptual model. This approach is automated with no human participation. We believe that human participation is vital to building the knowledge iteratively. Fliedl et al. [16] present a strategy to perform the analysis of complex sentences such as 'if/then constructions; it is interesting the analysis of conditional sentences. Letsholo et al. [22] propose a tool for automatically constructing analysis models that relies on a set of conceptual patterns. It is an interesting approach, although the results are conditioned by the patterns used. Kashimira et al. [21] also deal with natural language specification to derive a conceptual model, but they specifically derive an entity-relationship model. Al-Safadi et al. [1] proposed a semi-automated approach for the design of databases in enhanced-ERD notation using semantic analysis of the content. Bogatyrev et al. [6] present a framework for conceptual modeling that combines the usage of conceptual graphs and Formal Concept Analysis.

## 6. Conclusions and future work

This paper proposes an approach to derive a conceptual model from specifications written in natural language. Moreover, the proposed approach also checks the written style and suggests if necessary to improve the style according to the kernel sentences philosophy. We have performed a preliminary evaluation of the applicability of the process using the Systems Usability Scale (SUS) [8] [9]. Although SUS is mainly used to assess the usability of software systems, it was proven to be

effective in assessing products and processes [2]. The proposed approach can be used in different scenarios. It can be used to assist a single analyst eliciting from a large number of sources or it can also be used in a collaborative elicitation process where many contributors provide their requirements. In both cases, the objective of the proposed approach is to provide an overview of the whole consolidated knowledge in order to help to write a consistent and coherent specification. This approach is part of a bigger approach. We are working in a big process that considers different inputs: large documents (produced for a legacy system), short pieces of information (produced by instant messaging or similar), and specification produced by experts. Although we have developed and tested some pieces of this general approach, we have to continue working on the development of the tool, mainly on usability and performance issues.

## Acknowledgment

This paper is partially supported by funding provided by the STIC AmSud program, Project 22STIC-01.

## References

1. Al-Safadi, L. A. E.: “Natural Language Processing for Conceptual Modeling.” J. Digit. Content Technol. its Appl. 3, pp 47-59, 2009.
2. Bangor, A., Kortum, P. T., Miller, J. T.: "An empirical evaluation of the system usability scale." Intl. Journal of Human–Computer Interaction 24.6, pp. 574-594, 2008.
3. Bashir, N., Bilal, M., Liaqat, M., Marjani, M., Malik, N., Ali, M.: "Modeling Class Diagram using NLP in Object-Oriented Designing," 2021 National Computing Colleges Conference (NCCC), Taif, Saudi Arabia, pp. 1-6, 2021.
4. Berry, D., Kamsties, E. and M. Krieger. From Contract Drafting to Software Specification: Linguistic Sources of Ambiguity. Handbook, University of Waterloo, 2003.
5. Boehm, B.W.: Software Engineering, Computer society Press, IEEE, 1997.
6. Bogatyrev, M., Samodurov, K.: Framework for Conceptual Modeling on Natural Language Texts, CDUD@CLA, pp 13-24, 2016.
7. Boyd, N. S.: “Using Natural Language in Software Development.” In: Journal of Object-Oriented Programming - Report on Object Analysis and Design, 11-9, 1999
8. Brooke, J.: “SUS-A quick and dirty usability scale” Usability evaluation in industry, 189(194), pp. 4-7, 1996.
9. Brooke, J: "SUS: a retrospective", Journal of usability studies 8.2, pp.29-40, 2013.
10. Chomsky, N.: The Logical Structure of Linguistic Theory. Plenum Press, New York, 1975.
11. Dick, J., Hull, E., Jackson, K.: Requirements Engineering, Springer, fourth edition, 2011.
12. Django, https://www.djangoproject.com/, accessed: 2023-03-05
13. Duarte, R. B., Junior, J., Araújo, R., Wanderley, F., Lencastre, M.: “ Uma Abordagem Colaborativa de Modelagem Conceitual de Informação utilizando Mind Maps.” Workshop on Requirments Engineering, CIBSE, pp 575 588, 2014.
14. Ferrari, A., Spagnolo, G. O., Gnesi, S.: "PURE: A Dataset of Public Requirements Documents," 2017 IEEE 25th International Requirements Engineering Conference (RE), Lisbon, Portugal, 2017, pp. 502-505, doi: 10.1109/RE.2017.29.

15. Flask, https://flask.palletsprojects.com/, accessed: 2023-03-05
16. Fliedl, G., Mayerthaler, W., Winkler, C., Kop, C., Mayr, H. C.: "Enhancing requirements engineering by natural language based conceptual predesign," IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.99CH37028), Tokyo, Japan, pp. 778-783, 1999.
17. Gramajo, M. G., Ballejos, L. C., Ale, M., "Hacia la Evaluación Automática de la Calidad de los Requerimientos de Software usando Redes Neuronales Long Short Term Memory", Workshop on Requirements Engineering WER, 2020.
18. Harris, Z. S.: Co-occurrence and transformation in linguistic structure. (Linguistic Society of America) pp. 390- 457, 1957.
19. Ibrahim, M. Ahmad, R.: "Class Diagram Extraction from Textual Requirements Using Natural Language Processing (NLP) Techniques," 2010 Second International Conference on Computer Research and Development, Kuala Lumpur, Malaysia, pp. 200-204, 2010.
20. Insfrán, E. Tejadillos, E. Martí, S. Burbano, M. :"Transformación de Especificación de Requisitos en Esquemas Conceptuales usando Diagramas de Interacción". Workshop on Requirements Engineering (WER) pp: 91-105, 2002
21. Kashmira, P. G. T. H., Sumathipala, S., "Generating Entity Relationship Diagram from Requirement Specification based on NLP," 2018 3rd International Conference on Information Technology Research (ICITR), Moratuwa, Sri Lanka, pp. 1-4, 2018.
22. Letsholo, K. J., Zhao, L., Chioasca, E. V.: "TRAM: A tool for transforming textual requirements into analysis models," 2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE), Silicon Valley, CA, USA, pp. 738-741, 2013.
23. Lim, S. L., Finkelstein, A.: "StakeRare: Using Social Networks and Collaborative Filtering for Large-Scale Requirements Elicitation", IEEE transactions on software engineering, Volume 38, Issue 3, May-Jun 2012, DOI 10.1109/TSE.2011.36, pp 707-735, 2012
24. Lucassen, G., Robeer, M., Dalpiaz: Extracting conceptual models from user stories with Visual Narrator. Requirements Eng 22, pp 339–358, 2017.
25. Nascimento, R., Guimaraes, E., Lucena, M. "Requirements Smells como Indicador de Qualidade para Histórias de Usuários: Estudo Exploratório", Workshop on Requirements Engineering WER, 2021.
26. NLTK, https://www.nltk.org/, accessed: 2023-03-05
27. PlantUML, available at: https://plantuml.com/, accessed on 27th February 2023.
28. Potts, C.: "Using schematic scenarios to understand user needs," in Proceedings of the 1st conference on Designing interactive systems: processes, practices, methods, & techniques, 1995
29. Python, https://www.python.org/, accessed: 2023-03-05
30. Robeer, M., Lucassen, G., van der Werf, J. M. E. M., Dalpiaz, F., Brinkkemper, S.: "Automated Extraction of Conceptual Models from User Stories via NLP," 2016 IEEE 24th International Requirements Engineering Conference (RE), Beijing, China, pp. 196-205, 2016.
31. Shuttleworth, D. Padilla, J.: "From Narratives to Conceptual Models via Natural Language Processing," 2022 Winter Simulation Conference (WSC), Singapore, pp. 2222-2233, 2022.
32. Sklearn, https://scikit-learn.org/, accessed 2023-03-05
33. Spacy https://spacy.io/, accessed 2023-03-05
34. TensorFlow, https://www.tensorflow.org/, accessed: 2023-03-05
35. Zhao, L., Alhoshan, W., Ferrari, A., Letsholo, K. J., Ajagbe, M. A., Chioasca, E., Batista-Navarro, R. T.: Natural Language Processing for Requirements Engineering: A Systematic Mapping Study. ACM Comput. Surv. 54, 3, Article 55 (April 2022), 41 pages, 2021.