MAANA: An Automated Tool for DoMAin-specific HANdling of Ambiguity

Saad Ezzini*§, Sallam Abualhaija*§, Chetan Arora‡*, Mehrdad Sabetzadeh†*, Lionel C. Briand*†
*SnT Centre for Security, Reliability and Trust, University of Luxembourg, Luxembourg
‡Deakin University, Geelong, Australia
†School of Electrical Engineering and Computer Science, University of Ottawa, Canada
Email: {saad.ezzini, sallam.abualhaija}@uni.lu, chetan.arora@deakin.edu.au, {m.sabetzadeh, lbriand}@uottawa.ca

***Abstract*—MAANA (in Arabic: "meaning") is a tool for performing domain-specific handling of ambiguity in requirements. Given a requirements document as input, MAANA detects the requirements that are potentially ambiguous. The focus of MAANA is on coordination ambiguity and prepositional-phrase attachment ambiguity; these are two common ambiguity types that have been studied in the requirements engineering literature. To detect ambiguity, MAANA utilizes structural patterns and a set of heuristics derived from a domain-specific corpus. The generated analysis file after running the tool can be reviewed by requirements analysts. Through combining different knowledge sources, MAANA highlights also the requirements that might contain unacknowledged ambiguity. That is when the analysts understand different interpretations for the same requirement, without explicitly discussing it with the other analysts due to time constraints. This artifact paper presents the details of MAANA. MAANA is associated with the ICSE 2021 technical paper titled "Using Domain-specific Corpora for Improved Handling of Ambiguity in Requirements". The tool is publicly available on GitHub and Zenodo.**

***Index Terms*—Requirements Engineering, Natural-language Requirements, Ambiguity, Natural Language Processing, Corpus Generation, Wikipedia.**

I. Introduction

Natural language (NL) is the most common medium for specifying systems and software requirements in industry. While using NL is advantageous in many ways, doing so comes with an inherent risk: ambiguity [1], [2]; this means that a given requirements statement may be interpretable in multiple ways [3]. If ambiguity is not identified and mitigated early, it can lead to costly misunderstandings in later stages of development, with the potential to negatively impact the overall success of a project [4].

Handling ambiguity in requirements is complicated by the fact that requirements specifications often use a domain-specific vocabulary [5]. Consequently, generic NLP disambiguation tools are usually not very effective over requirements. Moreover, requirements ambiguity can be *unacknowledged*, meaning that different analysts may have different interpretations of the same requirement without even being aware of their disagreement. Such ambiguities are likely to be overlooked during inspections and requirements validation activities, since the analysts would discuss only what is ambiguous from their perspective.

§Corresponding Author

MAANA is a tool for detecting coordination ambiguity (CA) and prepositional-phrase attachment ambiguity (PAA) in requirements [1], [6], [7]. More details on the problem definition are available in our technical paper [8]. To establish an overall understanding of our tool, we briefly discuss the ambiguity types that it can handle.

Coordination is a structure that links together two sentence elements (called conjuncts) using a coordinating conjunction (e.g., "and" or "or") [9]. CA can potentially occur when the two conjuncts are preceded or followed by a modifier [7]. The sentence can then be interpretable in two ways, depending on whether only the conjunct next to the modifier is being modified or both conjuncts are being modified [1]. To illustrate, consider requirement **R1. Service availability shall measure the outage of LEO satellites and terminals**. The *first read* (i.e., first interpretation) occurs when the modifier "LEO" (low-earth orbit) modifies both conjuncts "satellites" and "terminals" (i.e., LEO [*satellites and terminals*]). The *second read* (i.e., second interpretation) occurs when the modifier "LEO" modifies "satellites" (i.e., [*LEO satellites*] and terminals).

A prepositional-phrase (PP) attachment is a PP preceded by a verb and a noun phrase [6]. Virtually all PP attachments have the potential for PAA, because they can be interpretable in two ways, depending on whether the PP modifies the preceding verb or the preceding noun phrase. To illustrate, consider requirement **R2. The outage management platform shall provide administrators with the ability to categorize outages with discrete tags**. The *verb attachment* (i.e., first interpretation) occurs when the PP "with discrete tags" is attached to the verb "categorize" (i.e., [*categorize*] outages [*with discrete tags*]). The *noun attachment* (i.e., second interpretation) occurs when the PP is attached to the noun "outages" (i.e., categorize [*outages with discrete tags*]).

MAANA detects the ambiguity in R1 and R2 using structural patterns and a set of heuristics computed over a domain-specific corpus that is automatically extracted from Wikipedia.

II. Tool Architecture

Fig. 1 depicts the architecture of MAANA. The tool is implemented as an Apache Maven project [10] and builds on the Apache UIMA framework [11]. Below, we discuss the main modules of MAANA, marked A – E in Fig. 1.

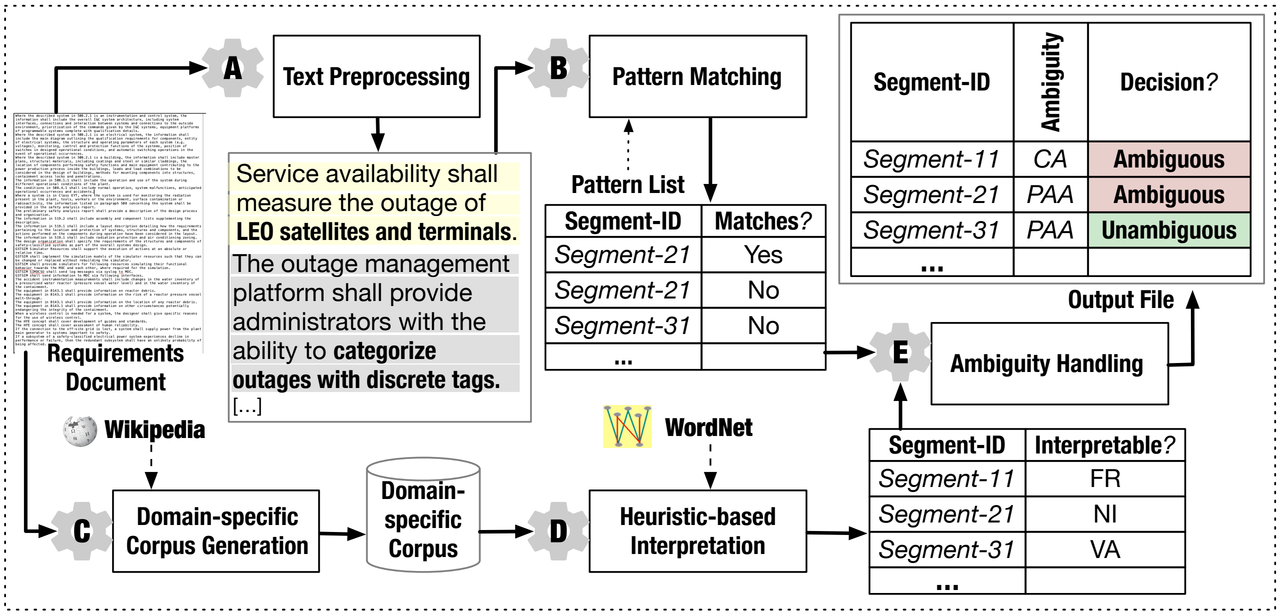
Text Preprocessing
Pattern Matching
Pattern List
Segment-ID | Matches?
Segment-21 | Yes
Segment-21 | No
Segment-31 | No
Segment-ID | Ambiguity | Decision?
Segment-11 | CA | Ambiguous
Segment-21 | PAA | Ambiguous
Segment-31 | PAA | Unambiguous
Output File
Ambiguity Handling
Requirements Document
Service availability shall measure the outage of LEO satellites and terminals.
The outage management platform shall provide administrators with the ability to categorize outages with discrete tags.
[...]
Wikipedia
WordNet
Domain-specific Corpus Generation
Domain-specific Corpus
Heuristic-based Interpretation
Segment-ID | Interpretable?
Segment-11 | FR
Segment-21 | NI
Segment-31 | VA
A
B
C
D
E

Fig. 1. Tool Architecture.

A. Text Preprocessing

This module parses the text of the input requirements document, splitting it into sentences and identifying the requirements that contain coordination and/or PP-attachment segments. The NLP pipeline that MAANA applies consists of a tokenizer, sentence splitter, part-of-speech (POS) tagger, lemmatizer, and constituency parser.

B. Pattern Matching

This module matches the coordination and PP-attachment segments in the input document against a predefined set of patterns. These patterns are meant at identifying those segments that have a higher chance of being ambiguous. The output for each segment is whether or not it matches any of the patterns in the list.

C. Domain-specific Corpus Generation

This module generates a domain-specific corpus from Wikipedia based on (automatically identified) keywords in the input requirements document. For further information about how to set up Wikipedia in order to generate a corpus for a given input document, consult the "Additional Instructions" provided alongside our actual artifact.

D. Heuristic-based Interpretation

This module subjects the identified coordination and PP-attachment segments to a set of interpretation heuristics, as described in our technical paper. The output is a combined decision based on these heuristics. NI suggests that a segment is not interpretable. For coordination, FR and SR suggest that a segment should be interpreted as first read or second read, respectively. And for PP attachment, VA and NA suggest that a segment should be interpreted as a verb attachment or a noun attachment, respectively.

E. Ambiguity Handling

The final module combines the results of modules (B) and (D) to obtain a final verdict for each identified segment. This final verdict is written out to the output file.

Acknowledgement. This work has received funding from Luxembourg's National Research Fund (FNR) under the grant BRIDGES18/IS/12632261 and from NSERC of Canada under the Discovery, Discovery Accelerator and CRC programs. We are grateful to the research and development team at QRA Corp. (Canada) for very valuable insights and assistance.

References

[1] D. Berry, E. Kamsties, and M. Krieger, "From contract drafting to software specification: Linguistic sources of ambiguity, a handbook," 2003. [Online]. Available: http://se.uwaterloo.ca/~dberry/handbook/ambiguityHandbook.pdf
[2] K. Pohl, *Requirements Engineering*, 1st ed. Springer, 2010.
[3] S. Piantadosi, H. Tily, and E. Gibson, "The communicative function of ambiguity in language," *Cognition*, vol. 122, no. 3, 2012.
[4] K. Pohl and C. Rupp, *Requirements Engineering Fundamentals*, 1st ed. Rocky Nook, 2011.
[5] A. Ferrari and A. Esuli, "An NLP approach for cross-domain ambiguity detection in requirements engineering," *Automated Software Engineering*, vol. 26, no. 3, 2019.
[6] C. Schütze, "PP attachment and argumenthood," *MIT working papers in linguistics*, vol. 26, no. 95, 1995.
[7] P. Engelhardt and F. Ferreira, "Processing coordination ambiguity," *Language and Speech*, vol. 53, no. 4, 2010.
[8] S. Ezzini, S. Abualhaija, C. Arora, M. Sabetzadeh, and L. C. Briand, "Using domain-specific corpora for improved handling of ambiguity in requirements," in *ICSE'21*, 2021.
[9] B. Strang, *Modern English Structure*, 2nd ed. Edward Arnold, 1968.
[10] "Apache maven." [Online]. Available: https://maven.apache.org
[11] "Apache uima." [Online]. Available: http://uima.apache.org