Research on NLP for RE at UPC: a Report

Ricard Borrull
Universitat Politecnica
de Catalunya (UPC)
Barcelona, Spain
rborrull@essi.upc.edu

Dolors Costal
Universitat Politecnica
de Catalunya (UPC)
Barcelona, Spain
dolors@essi.upc.edu

Xavier Franch
Universitat Politecnica
de Catalunya (UPC)
Barcelona, Spain
franch@essi.upc.edu

Carme Quer
Universitat Politecnica
de Catalunya (UPC)
Barcelona, Spain
cquer@essi.upc.edu

Abstract

[**Team Overview**] The Software and Service Engineering Group (GESSI) of UPC has traditionally conducted research in many fields of software engineering. [**Research Plan on NLP**] As a result of our participation in the OpenReq project, natural language processing (NLP) has become one of our highest priority research fields. We are using NLP for interdependency detection and requirements reuse, being the center piece of both tasks the identification of similar requirements.

1 Team Overview

The Software and Service Engineering Group (GESSI, https://gessi.upc.edu/) of the Universitat Politecnica de Catalunya (UPC) conducts research in many fields of software engineering, with particular emphasis on requirements engineering, software quality, software architecture, service-oriented computing, open source software, software modelling and empirical research.

Lately, this research has focused on the topics of three H2020 research projects in which we are participating: SUPERSEDE (on software evolution and self-adaptation), Q-Rapids (on the management of quality requirements in agile projects) and OpenReq (on applying intelligent recommendation and decision technologies for requirements engineering).

In the last year, as a result of our participation in the OpenReq project [Open], we began to study the application of natural language processing (NLP) techniques for the specification of natural language requirements and for the improvement of the quality of requirements. In the following section we outline the directions of research in the NLP field that we plan to explore.

2 Research Plan on NLP for RE

The overall goal of the OpenReq project [Open] is to develop intelligent recommendation and decision technologies that support different phases of RE, specifically elicitation, specification, analysis, management and negotiation.

The project focuses on using artificial intelligence-based techniques that proactively support stakeholders, both as individuals and as groups, within the scope of RE.

Inside OpenReq, UPC is working, among others, in the interdependency detection and requirements reuse part. We envisage using NLP for both tasks. In particular, this is of special importance for two of the trials of the project. One of them deals with bid projects of thousands of requirements, where they want to identify similar and dependent requirements in the same bid and from previous bids, and to reuse requirements knowledge of previous bids in the bit at hand; it is expected that this will reduce the cost of the bid phase and the requirements analysis phase by 10%. The other trial has a database with almost a hundred thousand requests (containing requirements and bugs). They have not that much interest in the reuse functionality, but in the identification of similar and dependent requests. It is estimated that at least 1000 requests (annually) could improve their management or be avoided.

2.1 Interdependency Detection

Interdependency detection will combine the identification of explicit and non-explicit interdependencies in the requirements. By explicit interdependencies, we mean explicit references in a requirement to other requirements, while by non-explicit interdependencies we mean those ones that are not explicitly stated in the requirements but that can be identified by analyzing the requirements both from a syntactic and semantic point of view.

For the *detection of explicit interdependencies*, we aim at following the approaches used in the well-known area of cross-references detection and resolution. In the first approach, we will identify natural language patterns that are used in requirements text to refer to other requirements and use them to propose new interdependencies. This approach will be based on works such as those of [Bre08] and [Pal03]. In the future, though, we will consider a second approach that consists in the possibility to extract these patterns automatically as done in [San17]. Although [San17] extract patterns used for cross-referencing in legal texts, we believe that a similar approach for automatic extraction of such patterns can be extrapolated to requirements specifications.

Regarding the *detection of non-explicit interdependencies*, the first step will be the identification of pairs of similar requirements (both from a syntactic and semantic point of view). *Similarity detection* (i.e., syntactic similarity) and *paraphrase detection* (i.e., semantic similarity) are approaches closely related to detection of interdependencies between requirements.

The relationship between similarity and paraphrase detection and interdependency detection is evident in the case where two requirements have almost exactly the same formulation. Imagine the requirements "*The user interface should use the letter type Arial letter type.*" and "*The user interface should use the letter type Calibri letter type.*". It is clear that these two requirements cannot be used in the same system (since it is not possible to use two letter types for the whole user interface), so these requirements are related by an *OR* interdependency (using the terminology proposed by Carlshamre [Car01]). However, even in other cases there are commonalities. As an example, if requirement *R1* states that *"It shall be possible to filter personal data by name and address"* and requirements *R2* states that *"The system shall allow to filter personal data by age"*, it would probably be convenient to treat these two requirements at the same time to save development resources. This example can be considered as an *ICOST* interdependency, according again to the terminology described in [Car01]. Other examples of interdependencies that can trigger a similarity analysis at a lexical level can be, e.g., the conflicting requirements "*The button shall be blue.*" and "*The button shall be red.*".

Therefore, identifying syntactically and semantically similar requirements could be used as a basis to identify related requirements. As there are several well-known components already developed to detect similar texts in English, our aim is to select one of these components to be integrated and expanded in OpenReq. At the time being, we envision adding some pre-processing, such as clustering or classification techniques, to the use of the component to reduce the set of requirements for which the component will run: if we reduce the set to which a requirement is compared to just the group of requirements it pertains according to a clustering algorithm, the possibilities to achieve better matches will improve. The components we are currently evaluating are:

- **Cortical** (http://cortical.io/). Among its functionalities, it provides a method to measure the similarity using the Cosine metric between two given texts. In this case, the method is not parameterizable.

- **DKPro** (https://dkpro.github.io/). DKPro provides, among others, a comprehensive repository of text similarity measures ranging from ones based on simple n-grams and common subsequences to more complex ones based on high-dimensional vector comparisons and structural, stylistic, and phonetic measures.

- **Gensim** (https://radimrehurek.com/gensim/tutorial.html). It includes implementations for popular algorithms such as LSA, LDA and RP. Gensim allows loading a corpus of texts to which a sentence can be compared. The calls to the algorithms are parameterizable.
- **Semilar** (http://www.semanticsimilarity.org/). The methods offered by Semilar, which are completely parameterizable, range from simple lexical overlap methods to methods that rely on word-to-word similarity metrics to more sophisticated fully unsupervised methods that derive the meaning of words and sentences such as LSA and LDA to kernel-based methods for assessing similarity. In addition, one can select the tokenizer, tagger, stemmer and parser to be used as pre-processing (having as options, for instance, the libraries OpenNLP, Stanford parser and WordNet).
- **Scikit-learn** (http://scikit-learn.org/stable/documentation.html). It allows the transformation of texts into vectors, using TF-IDF among other algorithms, and the measurement of the similarity over them using the Cosine metric.

However, we do not discard to add new components to this list if the results of our evaluations are not good enough or if other researchers know of other components that give better results.

The second step of non-explicit interdependencies section is to improve and expand the requirements similarity detection with further features, such as:

- Creating a specific list of synonyms that are domain dependent, so that the similarity algorithms can know when two words that in principle are not synonyms are actually synonyms in a specific domain.
- Constructing models that can help to detect interdependencies by relating concepts on this model. This is similar to the ontology used in [Zhu05], but in our model, relationships will broaden the scope of the previous work, which is focused on inconsistencies. For instance, if we know that technologies A and B are incompatible, A and B will be related in this model as conflicting. Therefore, when these two technologies are used at the same time in a single project, we can extrapolate that the requirements stating technologies A and B are actually conflicting and a new interdependency of this type will be created among them.

2.2 Requirements Reuse

To achieve requirements reuse, we plan to adapt the *PABRE* framework [Fra13] [Ren09], which stands for PAtterns-Based Requirement Elicitation, to OpenReq. The core asset in the PABRE framework are *Software Requirement Patterns* (SRPs). In a nutshell, an SRP is a set of *Templates* that pursue the same goal in a system to be developed. Each template corresponds to the text to be used as a requirement and, if necessary, some optional *Parameters* (with a set of possible values: range of numbers to be used, possible strings, etc.) that need to be instantiated when applying the pattern. As a goal can be achieved in different ways, an SRP consists of several *Forms*, each one representing a different solution for achieving the goal. In additon, Forms are organized into *Parts*, each of them being a template: a *Fixed Part*, which is always applied if the form is chosen, and some *Extended Parts*, which may be applied or not. Finally, SRPs are classified using *Classification Schemas*, which are hierarchies of classifiers that facilitate the organization of SRPs.

In OpenReq, we will adapt the structure of SRP defined in the PABRE framework to the needs of the project. Additionally, we will need to define a catalogue of SRPs for OpenReq (possibly one per trial). Therefore, it is important that the population of this catalogue will combine automatic extraction with expert assessment. The automatic extraction will need some type of NLP processing, and probably machine learning. For instance, we believe topic modelling could be used to extract the common topics found in requirements specifications, clustering techniques to group requirements, and syntactic and semantic similarity identification to find requirements that are repeated along the different specifications. These techniques might help to populate the SRP catalogue.

Finally, we envisage different ways in which this catalogue of patterns could be used in OpenReq. Here, we present the ones that will need the support of NLP:

1. *Propose SRP that are dependent.* Given a requirement, using a similarity algorithm it will be possible to recover similar SRP (by analyzing the templates in the SRPs with the given requirement). Then, once we know the similar SRPs to a given requirement, if they are dependent to other SRPs, we can propose these dependent SRP so they can be reused. For instance, *R1* is similar to a template in the requirement pattern *SRP2*, which is dependent on the requirement pattern *SRP3*. In that case, *SRP3* could be proposed to be reused for *R1*.

2. *Propose SRP the are related.* Again, given a requirement, using a similarity algorithm it will be possible to recover similar SRP (by analyzing the templates in the SRPs with the given requirement) (e.g., *R1* is similar to a template of *SRP1*). Then, if *SRP1* is in classifier *C1* other SRPs in the same classifier could be proposed for reuse (e.g., *SRP2*). So, for *R1*, *SRP2* will be proposed since they both under the same classifier. A similar approach could be used for the keywords. If *SRP1* has keyword *K1*, we can look in the catalogue for other SRPs that have keyword *K1* (e.g., *SRP3*), and these SRPs could be proposed for reuse. So, in that case, for *R1*, *SRP3* will be proposed for reuse since they both are both about keyword *K1*.

2.3 Further Issues

It is important to highlight here the fact that OpenReq will deal not only with English language, but also with Italian and German, since one of the trials deals with requirements written in Italian, and another trial deals with requirements (partially) written in German. This diversity of languages used to write the text analyzed supposes a challenge for the project. The majority of the existing NLP approaches target the English language, as they are trained and validated using English text corpora. Although NLP approaches and software libraries exist for both languages [Bas15] [Reh12], we have to check that their performances (e.g., precision) are not inferior compared to the well-established, English-based ones (specially when used for parsing and similarity detection).

Acknowledgements

The work presented in this paper has been conducted within the scope of the Horizon 2020 project OpenReq, which is supported by the European Union under the Grant Nr. 732463.

References

[Bas15] Basili R., Bosco C., Delmonte R., Moschitti A.; Simi M. *Harmonization and development of resources and tools for Italian natural language processing within the PARLI Project* Springer, 2015.

[Bre08] Breaux, T.; Anton, A. *Analyzing regulatory rules for privacy and security requirements* IEEE Transaction on Software Engineering, 34(1), 520, 2008.

[Car01] Carlshamre, P.; Sandahl, K.; Lindvall, M.; Regnell, B.; och Dag, J. N. *An industrial survey of requirements interdependencies in software product release planning* 5th IEEE International Symposium on Requirements Engineering, 84-91 2001.

[Fra13] Franch, X.; Quer, C.; Renault, S.; Guerlain, C.; Palomares, C. *Constructing and using software requirement patterns Managing requirements knowledge* Springer, 2013.

[Pal03] Palmirani, M.; Brighi, R.; Massini M. *Automated extraction of normative references in legal texts* 9th international conference on artificial intelligence and law (ICAIL03), 105106, 2003.

[Open] *OpenReq: Intelligent Recommendation and Decision Technologies for Community-Driven Requirements Engineering* (Horizon 2020 Project, https://www.openreq.org).

[Reh12] Rehbein I., Ruppenhofer J., Sporleder C.; Pinkal M. *Adding nominal spice to SALSA - frame-semantic annotation of German nouns and verbs* Conference on Natural Language Processing (KONVENS), 2012.

[Ren09] Renault, S.; Mndez, O.; Franch, X.; Quer, C. *A Pattern-based Method for building Requirements Documents in Call-for-tender Processes* Int J Comput Sci Appl, 6(5), 175202, 2009.

[San17] Sannier, N.; Adedjouma, M.; Sabetzadeh, M.; Briand, L.C. *An automated framework for detection and resolution of cross references in legal texts* Requirements Engineering, 22(2), 215-237, 2017.

[Zhu05] Zhu, X.; Jin, Z. *Inconsistency measurement of software requirements specifications: an ontology-based approach* 10th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS), 402-410, 2005.