Improving Requirements with NLP Techniques

Author: Alejandro Rago

Affiliation: ISISTAN Research Institute (CONICET) - UNICEN University

Elaborating "good" requirements specifications is an important factor for the success of a software project. Requirements are normally expressed using textual descriptions in natural language, but not without problems. Some requirements documentation techniques, such as use cases specifications, often focus on functionality and leave many concerns understated in the text and scattered through several documents. These concerns, commonly known as crosscutting or architecturally-relevant concerns, often come from business goals or quality attributes that must be clearly identified by analysts and developers, as these concerns can have a far-reaching effect in the development process. Not treating these concerns at early development stages can lead to poor design solutions that become difficult (and costly) to fix afterwards. Unfortunately, searching for concerns in textual requirements is a difficult and time-consuming task for analysts, because requirements are often poorly modularized and there is text duplicated across documents.

Automated techniques able to interpret textual documents and extract information about concerns can be of great help for the analyst's work, allowing her/him to get a quick view of duplicated functionalities and potential concerns at the outset of development and thus make informed decisions. Several approaches to analyze textual requirements have relied on Natural Language Processing (NLP) techniques in order to inspect the text and extract "hidden" information. Some techniques have been proposed for analyzing the modularity of requirements, although only a few researchers have addressed duplicate functionality in textual specifications [1]. Likewise, there are several techniques for discovering architecturally-relevant concerns scattered in textual requirements [5]. Yet, such techniques do not work well in real-world specifications because of their poor semantic support (in the context of software development).

The main objective of our work is to assist analysts for improving the quality of requirements documentation and fostering the inspection of architecturally-significant concerns. The hypothesis is that by leveraging on NLP techniques and domain knowledge (about use cases) is possible to develop techniques that reveal duplicate functionality and concern-related information. The proposed approach makes use of NLP modules to process textual requirements, labeling meta-information such as time tenses, syntactical functions, and morphological lemmas. To improve the modularity of use cases, we feed the information outputted by NLP-based modules into a sequence alignment algorithm for exposing duplicate requirements and suggesting candidate refactorings for the use cases. Sequence alignment originated in bioinformatics for finding duplicated chains in DNA strings. To reveal information about latent architectural concerns, we complement the analysis of textual use cases with semantic NLP modules (e.g,

dependency parsing, semantic role labeling) and specific concepts derived from use cases, called domain actions. Then, analysts can query this information via concern-specific rules for identifying a given concern.

We have started to build tools to test the techniques, emphasizing on the analysts' user experience. Initial prototypes of the techniques are currently available for download[1]. Technologically speaking, the tools are built on top of the Unstructured Information Management Architecture (UIMA) and Rule-based Text Annotation (RUTA), two frameworks that enable the configuration of state-of-the-art NLP modules in different arrangements. We currently have publications reporting on one particular technique for discovering architectural concerns [2,3] and have submitted another one about the technique for improving the modularity of use cases [4].

We have made considerable progress so far in the development of the approach. We built a core engine for executing UIMA modules in an Eclipse environment and investigated a number of NLP packages publicly available (e.g., OpenNLP and MateTools). These packages were wrapped as UIMA annotators and arranged into a customizable NLP pipeline. In addition, we developed UIMA modules able to recognize requirements-specific concepts using machine learning algorithms. We also adapted a sequence alignment algorithm to work on textual use cases, making the necessary adjustments to perform well in this domain, and implemented a rule-based module on top of RUTA that allows analysts to run queries to reveal architectural concerns.

We have performed preliminary experiments of the techniques in several case-studies, obtaining promising results. Nonetheless, we plan to test the techniques with more case-studies to further corroborate our findings. Particularly, we are interested in analyzing the influence of discovering latent architectural concerns on poorly-modularized requirements versus discovering them on well-modularized requirements. Some of the next steps in our research are: (i) investigate querying specialized concerns, such as quality-attribute concerns or business goals, directly from stakeholders workshops or business/mission statements; (ii) if the concern identification is performed on multiple text documents, explore the traceability links between the concerns extracted from each document; and (iii) analyze the performance of the techniques when used in combination with the criteria of human analysts.

References

1. Ciemniewska, A., Jurkiewicz, J.: Automatic Detection of Defects in Use Cases. Master's thesis, Poznan University of Technology (2007)
2. Rago, A., Marcos, C., Diaz-Pace, A.: Uncovering quality-attribute concerns in use case specifications via early aspect mining. Req. Eng. 18(1), 67–84 (March 2013)
3. Rago, A., Marcos, C., Diaz-Pace, A.: Assisting requirements analysts to find latent concerns with reassistant. Aut. Soft. Eng. (June 2014)
4. Rago, A., Marcos, C., Diaz-Pace, A.: Identifying duplicate functionality in textual use cases by aligning semantic actions (under review). Soft. & Sys. Mod. (2014)
5. Sampaio, A., Rashid, A.: EA-Miner: towards automation in AORE. In: Transactions on AOSD III, LNCS, vol. 4620, pp. 4–39 (2007)

[1] http://code.google.com/p/reassistant/ & http://code.google.com/p/ucrefactoring/