Identifying Technical Debt through Code Comment Analysis

Mário André de Freitas Farias
Undergraduate Program in Systems of Information, Federal Institute of Sergipe
Aracaju - Brazil
mario.andre@ifs.edu.br

Rodrigo Oliveira Spínola
Graduate Program in Systems and Computer, Salvador University
Salvador - Brazil
rodrigo.spinola@unifacs.br

Manoel G. de Mendonça Neto
Graduate Program in Computer Science, Federal University of Bahia
Salvador - Brazil
manoel.mendonca@ufba.br

ABSTRACT

The software industry often has to deal with several challenges to deliver and maintain products. Unfortunately, development challenges lead developers to take shortcuts or use workarounds. That is what the Software Engineering (SE) community now calls Technical Debt (TD).

The identification of TD is an important step to effectively manage TD items, keeping the amount of TD under control and making it manageable and explicit. Researchers have developed automated approaches to identify TD items using indicators derived from source code metrics. However, those indicators do not always point to TD that developer teams consider problems and cannot identify many types of relevant TD.

The concept of self-admitted technical debt (SATD) considers debt that is intentionally introduced. Our strategy is to consider code comments as an information source for SATD. This work seeks to expand the knowledge frontier on automated TD identification by developing an approach to automatically identify SATD items, and classifying them into different TD types, through code comment analysis

To do that, we focused on qualitative and quantitative analysis to synthesize evidence, summarizing, integrating, combining, and comparing the findings of a set of experiments. To combine quantitative and qualitative approaches, we used a triangulation methodology to analyze how code comment analysis approach supports the SATD identification.

We proposed an initial model and contextualized vocabulary aiming to select comments possibly related to SATD. The model is a contextualized structure of patterns that focuses on the use of word classes and code tags to provide a SATD vocabulary aiming to support the detection of different types of SATD through comment analysis using text patterns. The model provides a structure that systematically allows combining terms to create a large vocabulary on SATD.

Next, we developed a tool named *eXcomment*. The tool extracts, filters, and selects comments from source code using the vocabulary. The tool was developed based on techniques from text mining, such as data selection, preprocessing, tokenizing the unstructured text, extracting and searching for terms. Lastly, we explored, evaluated and incrementally evolved this approach through a family of four empirical studies (*FindTD*). Two of them had the purpose of characterizing and improving our strategy to identify SATD items (*FindTD I* and *III*), whereas the others had the purpose of evaluating our approach (*FindTD II* and *IV*).

The *FindTD I* was an exploratory study performed to characterize the feasibility of the proposed model to support the detection of TD through code comments analysis. The results showed that the CVM-TD provides a vocabulary that makes it possible to extract comments that can be used to support SATD identification.

Following, the promising initial outcome motivated us to evaluate CVM-TD with other data sources. Thus, we performed the *FindTD II* (a controlled experiment) analyzing the overall accuracy of CVM-TD when classifying candidate comments and factors that influence the identification of SATD. The results indicated that CVM-TD provided promising results considering the accuracy values. English reading skills have an impact on the TD detection process. We identified a list of the 20 most chosen patterns by participants as decisive to indicate TD items.

Next, we performed *FindTD III* based on insights gained in *FindTD II*. We changed the experimental setup and controlled other variables. Our primary goal in this experiment was to analyze the set of comment patterns identified and classified in the previous experiment. We carried out a qualitative analysis to improve the model and the vocabulary, identifying the most significant patterns, and the relationship between comment patterns and TD types. The results provided us a new release of the contextualized vocabulary.

Our last study was *FindTD IV*. The main goal of this study was to evaluate the whole process to identify and classify SATD items through code comment analysis automatically. We applied qualitative and quantitative data, aiming to evaluate and evolve once more our strategy to improve the detection and classification of SATD items through code comment analysis. Some evidence shows that identifying SATD items through code comment analysis can be a hard task. However, our approach can be used by developers to determine TD items automatically in some cases, classifying them in some types of TD. Also, even when that is not possible, it provides support to facilitate the analysis of comments by developers in order to detect SATD items.

Our findings indicate that our approach makes it possible to select a list of suitable comments to support automatic SATD identification. The performed studies provided new evidence on how software engineers can use code comments to detect and classify SATD items automatically. Thus, this research contributes to bridge the gap between the TD identification area and code comment analysis, successfully using code comments to detect several types of SATD.