CHALMERS | UNIVERSITY OF GOTHENBURG
Department of Computer Science and Engineering, Helena Ólafsdóttir
Software Engineering Master Thesis Proposal, 30hec

Automatically extracting information from software architecture documents

Student: Helena Ólafsdóttir

Supervisor: Michel Chaudron

Completed courses relevant for thesis work:
DAT231 Empirical software engineering
TDA231 Algorithms for machine learning and inference
DAT345 Techniques for large-scale data
DAT340 Applied machine learning
TIN175 Introduction to artificial intelligence
DAT306 Industrial project in software engineering

1 Introduction

The world is moving faster than ever with new technology and new knowledge surfacing every day. Current generations have gotten used to the speed of new technology and have started to expect technology to solve or at least aid with most of their tasks and activities. Still to this day however, software developers spend vigorous amounts of time manually comprehending software architectures, e.g. by reading software architecture documentations (SADs). The idea of using technology to reduce this manual and tedious activity is not new and many advances have been made in attempt to make software comprehension easier and less time consuming. Most engineers and architects however still spend too much time comprehending systems, and even when they only need answers to a few specific questions, they still need to go through vast amounts of source codes, diagrams and documentations.

It is apparent that many steps need to be taken in order to replace this old-fashioned way of software comprehension. In this research, we would like to contribute to that large goal, by taking one of those necessary steps.

2 Statement of the problem

The On-Demand Developer Documentation (OD3) (Robillard et al 2017) points out the high cost and low immediate return of SADs and introduce a vision for better satisfying information needs of software developers. The OD3 proposes a system that would generate appropriate documentation based on user queries, i.e. a user should be able to ask questions and retrieve relevant information from source code, documentations, etc. There are many challenges that need to be overcome to create the system described. One of those is retrieving content and structural information from SADs, documents written in natural language, which we intend on focusing on in this research.

3 Purpose of the study

This research will take one of the steps necessary to create a system similar to the one visioned in the On-Demand Developer Documentation. In order to answer questions about the content of a SAD, it's important to possess knowledge about different parts of it, such as what a specific diagram is showing you, and what a specific section or sentence is discussing. The purpose of this study is therefore to classify information contained in software architecture documentation into predefined categories of architectural knowledge.

4 Review of the literature

On-Demand Developer Documentation (Robillard et al 2017). As mentioned before, this paper demonstrates the need for research in this area, with the ending goal being a system where users can ask software architecture related questions and get answers based on the software's source code and documentations.

Towards automatically generating explanations of software systems (Tao et al 2018). In this paper by Tao and Roodbari, an ontology is defined and manually filled with information from a SAD. The information in the ontology is then used to answer user queries. This paper takes a very important step towards developing a system similar to the OD3. By classifying the information from the SAD first, we are taking a large step towards automatically structuring the knowledge contained in the SADs, and possibly automatically filling an ontology.

Mohamed Soliman, a doctorate student from the University of Hamburg, has been looking into various aspects of the problem of acquiring relevant knowledge to make technology design decisions (Soliman 2018; Soliman et al 2017; Soliman et al 2015). Through his research, he has created an ontology of architectural concepts and classified stack overflow posts based on this ontology. He then used the ontology to create a web-based search engine that proved more effective than the conventional keyword-based search. Soliman's work can be used as basis for the classification approach and his categories of architectural knowledge will be kept in mind.

A systematic Mapping Study on Text Analysis Techniques in Software Architecture (Tang et al 2018). This literature review thoroughly covers research using different text analysis techniques in the field of software architecture, i.e. classification, clustering, search and information retrieval, etc. From this mapping study, it's apparent that many researchers are focusing on different ways to use text analysis techniques to extract information that can be helpful to software architects and engineers. However, none of the papers reviewed use machine learning techniques to extract general architecture knowledge from SADs. There are three papers mentioned in this literature review that have work related to our research; *Bridging the gap between software architecture rationale formalisms and actual architecture documents: An ontology-driven approach* (López et al 2009), recognizes the need of information extraction from SADs and has a similar idea of a system that helps

users grasp specific knowledge, using information extraction and ontologies. However, their approach to extract information deviates from ours, as they use rule-based learning for this purpose. *Semi-automated Design Guidance Enhancer (SADGE): A Framework for Architectural Guidance Development* (Anvaari et al 2014), develops a framework for extracting architectural knowledge. Again however, rule-based learning is user for information extraction. *Personalized architectural documentation based on stakeholders' information needs* (Nicoletti et al 2011) attempts to identify parts of SADs that might be of interest of specific users. Their approach is to identify software architecture related concepts in SADs and use them to classify their sentences into predefined categories. Their approach is however semi-automated as they require manual semantic annotations with the SAD document, in order to produce results.

There has also been a lot of research related to categorization in software engineering, for example research related to classifying documents, online posts or parts there of as software related or not, or as useful or not (Maalej et al, 2015; Robillard et al, 2015; Bacchelli et al, 2012; Zhou et al 2014)

The approach of this research will be based on Soliman's previous work, however the novelty of our research will be the focus on software architecture documentations, as well as our contribution to a new and exciting development within the field of software architecture, the OD3 vision.

5 Research question and/or Hypotheses

The aim of this thesis is to classify different parts of SADs (granularity to be defined) with regards to their relation to different categories of architectural knowledge. The main research question to be answered is the following:

RQ1*: How to automatically extract relevant information from software architecture documents?*

There are many aspects that need to be investigated in order to sufficiently answer this research question. The main sub-questions of this research are:

RQS1: *What categories of architecture knowledge are generally incorporated in SADs?*

RQS2: *What granularity of classification will yield the most relevant results?*

RQS3: *What type of questions will the user generally ask and for what types of questions will this approach work best?*

RQS4: *What is the most efficient presentation of the results?*

6 The Design – Methods and Procedures

The approach used in this study will be a combination of design science and experimentation with machine learning. The aim of this thesis is to build a system where users can query a SAD and be presented with relevant information extracted from the SAD. Machine learning techniques will be used to classify sentences/sections of SADs into AK categories. The categories will then be used to select the sentences/sections that should be a part of the response to the user query. In order to know what kind of queries to expect from the user, some scoping work will have to be performed. Following are some examples of possible user queries:

- *List all user stories of this system*
- *List all non-functional requirements of this system*
- *Which programming languages were used to write this system?*
- *What architecture pattern was followed during the implementation of this system?*
- *Give me all the information you have about the system's payment module*

At the same time, we will look into different ways to refine the responses to the user, possibly by using machine learning techniques to extract the main nouns from sentences (e.g. to extract the name of a programming language), or by asking the user to enter additional information about what he seeks. Simultaneously, user studies will be performed to find out the most efficient way to query and present results.

The data used for this thesis is a collection of SADs from GitHub collected by Michel Chaudron et al. containing approximately 100 SADs collected from open-source projects.

The following list is a high-level plan for the execution of this research:

1. Extract the content and structure of the SADs, a tool will be used for this purpose.
2. Determine the categories of architectural knowledge that will be used for classification.
    - These categories will be determined using qualitative content analysis, both by looking through current literature on architectural knowledge and by iterating through the data provided for this research.
3. Create ground truth by manually annotating the SADs, using the defined categories.
4. Extract features from the SADs.
    - Various techniques will be investigated, such as bag-of-words, tf-idf, n-grams and ontologies.
5. Train and evaluate a classifier using the extracted features.
    - Different supervised classification algorithms will be investigated such as k-nearest neighbor, decision trees, and possibly neural networks.
    - Unsupervised clustering algorithms might then also be interesting for comparison.

6. Build a system for handling user queries
    - We will investigate different types of user queries as well as the best way to present the user with the result to his query. Test users will be used to evaluate the system.

7 Limitations and Delimitations

To begin with, the study will focus on software architecture documentations, eliminating all other natural text, source code, etc. However, we are aware of the possibility of investigating how including StackOverFlow data from Soliman's research (Soliman 2018) might affect the performance of the classifier.

Level of annotation will also be limited, i.e. we might decide to only classify sentences, paragraphs, sections, etc. The annotation level will be decided alongside the qualitative content analysis of architectural knowledge categories.

To begin with, the study will only investigate supervised learning algorithms, with the possibility of extending to unsupervised clustering algorithms.

8 Significance of the study

The results of this study should be of interest to both practitioners and researchers. First of all, this study will contribute to a vision that various software engineering researchers are interested in and working on, the OD3 vision. Second of all, this study should appeal to machine learning researchers focusing on natural language processing and extracting knowledge from natural language. Finally, this study will be of interest to practitioners looking for more efficient ways to comprehend architectural knowledge.

9 References

Robillard, M.P., Marcus, A., Treude, C., Bavota, G., Chaparro O., Ernst, N., Gerosa, M.A., Godfrey, M., Lanza, M., Linares-Vásques, M., Murphy, G.C., Moreno, L., Shepherd, D., Wong, E. (2017) On-Demand Developer Documentation. *IEEE International Conference on Software Maintenance and Evolution (ICSME)*, 479-483.

Tao, A., Roodbari, M. (2018) Towards automatically generating explanations of software systems (Master's thesis). Chalmers University of Technology & University of Gothenburg. Gothenburg, Sweden.

Soliman, M.A.M., (2018) Acquiring Architecture Knowledge for Technology Design Decisions (Doctorate thesis). University of Hamburg. Hamburg, Germany.

Soliman, M.A.M., Galster, M., Riebisch, M. (2017) Developing an Ontology for Architecture Knowledge from Developer Communities. IEEE International Conference on Software Architecture (ICSA), 89-92.

Soliman, M.A.M., Riebisch, M., Zdun, U. (2015) Enriching Architecture Knowledge with Technology Design Decisions. *12th Working IEEE/IFIP Conference on Software Architecture*, 135-144.

Tang, A., Bi, T., Liang, P., Yang, C. (2018) A Systematic Mapping Study on Text Analysis Techniques in Software Architecture. *Journal of Systems and Software*, 533-558.

López C., Codocedo V., Astudillo H., Cysneiros L. M. (2012) Bridging the gap between software architecture rationale formalisms and actual architecture documents: an ontology-driven approach. *Science of Computer Programming*, 66-80.

Anvaari M., Zimmermann O. (2014) Semi-automated design guidance enhancer (SADGE): a framework for architectural guidance development. *Proceedings of the 8th European Conference on Software Architecture (ECSA)*, 41-49

Nicoletti, M. Díaz-Pace J.A., Schiaffino S. (2011) Towards software architecture documents matching stakeholders' interests. *Proceedings of the 2nd International Conference on Advances in New Technologies, Interactive Interfaces, and Communicability (ADNTIIC)*, 176-185.

Maalej W., Robillard M. P. (2013) Patterns of knowledge in API reference documentation. *IEEE Transactions on Software Engineering*, 1264-1282.

Robillard M. P., Cheetri Y. B. (2015) Recommending reference API documentation. *Empirical Software Engineering*, 1558-1586.

Bacchelli, A., Sasso, T. D., D'Ambros, M., Lanza, M. (2012) Content classification of development emails. *Proc. 34th IEEE/ACM International Conference on Software Engineering*, 375-385.

Zhou, B., Xia, X., Lo, D., Tian, C., Wang, X. (2014) Towards more accurate content categorization of API discussions. *22nd International Conference on Program Comprehension*, 95-105.