Detecting Inconsistencies in Software Architecture Documentation Using Traceability Link Recovery

Jan Keim[1], Sophie Corallo[1], Dominik Fuchß[1], Anne Koziolek[1]

Abstract: In our talk, we present our publication about detecting inconsistencies using traceability link recovery that was originally published at the 2023 IEEE 20th International Conference on Software Architecture (ICSA).

Keywords: Inconsistency Detection; Traceability Link Recovery; Consistency; Documentation; Software architecture; Software engineering

Motivation Our paper that was published at the 2023 IEEE 20th International Conference on Software Architecture (ICSA) is about Software Architecture Documentations (SADs). SADs are important to improve the development, maintenance, and evolution of a software system [MT10] and prevent deterioration [Pa94]. One issue with SADs is inconsistency, especially between different kinds of SADs like informal natural language software architecture documentations (NLSADs) and formal architecture description languages like UML or PCM. Not all inconsistencies are problematic [NER01] and developers need to decide on fixing or tolerating inconsistencies. However, developers cannot decide on undetected inconsistencies, making them problematic.

In this paper, we present an approach to detect two kinds of inconsistencies between NLSAD and software architecture models (SAMs): *unmentioned model elements (UMEs)* and *missing model elements (MMEs)*. UMEs are model elements such as components that exist in the model but are not mentioned within the NLSAD. MMEs are inconsistencies that occur when an NLSAD contains an architectural element that is not part of the model. To detect these inconsistencies, our approach Architecture Documentation Consistency (ArDoCo) makes use of the existing approach SoftWare Architecture Text Trace link Recovery (SWATTR) for traceability link recovery (TLR) to identify commonalities in different artifacts (cf. [Ke23]) to then be able to identify differences. Overall, we look at the following research questions: **RQ1** To what extent do changes to SWATTR improve the performance for TLR? **RQ2** How well does the approach detect UMEs? **RQ3** How well does the approach detect MMEs?

Approach As Figure 1 shows, ArDoCo enhances the SWATTR approach (cf. [Ke21]) that identifies probable model elements (Recommended Instances, RIs) before creating trace links. These RIs can be used to identify MMEs. The precision of the results is further improved with a filtering step. Similarly, the approach compares the existing model elements with the found trace links to identify undocumented ones (UMEs).

[1] Karlsruhe Institute of Technology, Germany, {jan.keim,sophie.corallo,dominik.fuchss,koziolek}@kit.edu

doi:10.18420/sw2024_37

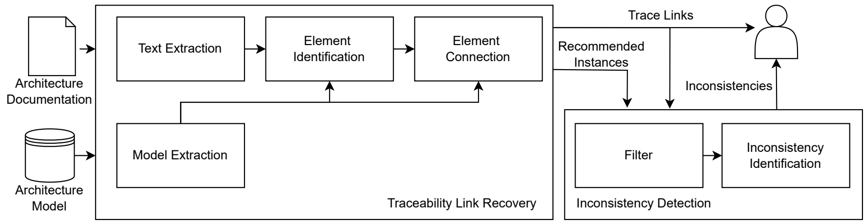

Fig. 1: Overview of the approach for inconsistency detection [Ke23]

Evaluation We evaluated the approach with a benchmark (cf. [Fu22]) containing five projects from different domains containing differently sized SADs and SAMs in different versions. We measured the performance of our approach using the metrics precision, recall, F_1-score, accuracy, specificity, and Φ (a.k.a. Matthews correlation coefficient (MCC)), comparing the results with baseline approaches. The approach achieves an excellent weighted average F_1-score of 0.81 for TLR and 0.86 for UME, and a promising F_1-score of 0.34 (accuracy 0.77), significantly outperforming the baselines (with significance level α of 0.05).

Conclusion Our approach ArDoCo is effective for TLR and promising in detecting inconsistencies in SAD using TLR. Despite these results, there is room for improvement, especially in the detection of model elements in the NLSAD. Moreover, the approach currently does not make use of or consider relations. As such, we want to explore them in future work.

Data Availability: We provide a replication package that includes the code, baselines, our evaluation data, and results at `https://doi.org/10.5281/zenodo.7555195`.

Bibliography

[Fu22] Fuchß, Dominik; Corallo, Sophie; Keim, Jan; Speit, Janek; Koziolek, Anne: Establishing a Benchmark Dataset for Traceability Link Recovery between Software Architecture Documentation and Models. In: 2nd MSR4SA - Co-located with 16th ECSA. 2022.

[Ke21] Keim, Jan; Schulz, Sophie; Fuchß, Dominik; Kocher, Claudius; Speit, Janek; Koziolek, Anne: Tracelink Recovery for Software Architecture Documentation. In: Software Architecture. Springer International Publishing, pp. 101–116, 2021.

[Ke23] Keim, Jan; Corallo, Sophie; Fuchß, Dominik; Koziolek, Anne: Detecting Inconsistencies in Software Architecture Documentation Using Traceability Link Recovery. In: 2023 IEEE 20th International Conference on Software Architecture. pp. 141–152, 2023.

[MT10] Medvidovic, Nenad; Taylor, Richard N.: Software architecture: foundations, theory, and practice. In: 32nd ICSE. volume 2, pp. 471–472, 2010.

[NER01] Nuseibeh, Bashar; Easterbrook, Steve; Russo, Alessandra: Making inconsistency respectable in software development. Journal of Systems and Software, 58(2):171–180, 2001.

[Pa94] Parnas, D.L.: Software aging. In: 16th ICSE. pp. 279–287, 1994.