

Study Protocol

Paula Rachow, Matthias Riebisch

This is the study protocol for the paper “An Architecture Smell Knowledge Base for Managing Architecture Technical Debt”. The study protocol follows the guidelines of Kitchenham and Charters [KC07].

Background

Many software projects suffer from architecture erosion and architecture technical debt [Ver+20]. One challenge is to identify affected parts and prioritize them for refactoring. Architecture smells are indicators of architecture technical debt, but they are ambiguous and their impact is hard to assess. There have been some attempts to build a catalog [LR06; Bro+98; Gar+09; AFT19] but there is not one yet that is as established as the code smell catalogue of Fowler [Fow18].

To improve the understanding of architecture smells and their impact, we want to build a knowledge base that improves understanding of architecture smells and identifies violated software design principles and affected quality attributes. The design principles help our understanding of what causes architecture smells, while the impaired quality attributes represent the consequences. To fill this knowledge base, we want to search the existing literature to identify these relations. This ontology will provide a knowledge base that architects can use to prioritize the smells according to the project’s individual quality goals.

Research Questions

The research questions to be addressed by this study are:

- Which architecture smell violates which software design principle?
- Which architecture smell impairs maintainability or rather one of the maintainability sub-characteristics?

Selection Process

The search process is a manual search of literature. The nominated databases and conferences are shown below.

Sources to be Searched

- ACM
- IEEE
- Web of Science
- Science Direct
- Wiley Online Library
- Springer

Inclusion Criteria

Literature fulfilling the following inclusion criteria:

- includes architecture smells
- principles named in the context of architecture smell
- maintainability attribute connected to architecture smell

Exclusion Criteria

1. Remove Duplicates
2. Remove non-english
3. Exclude based on study selection criteria
4. Remove non-full-text
5. Remove after full-text reading
6. Remove duplicate data source

Primary study selection process

The relevant candidate and selected studies will be selected by a single researcher. We will maintain a list of candidate papers that were rejected with reasons for the rejection.

Quality Assessment

We will verify the choice of databases and search strings by selecting existing studies that included connections of architecture smells and software design principles and maintainability characteristics [Gar+09; AFT19; LR06; Gar+09]. We will also use already found synonymous terms for architecture smells for our search strings other studies already found [TLB19; Fon+19]. We will perform backward snowballing

Data Collection

The data extracted from each paper will be:

- The source
- The author(s)
- Named architecture smells
- named principles and quality attributes connected to AS

The data will be extracted by one researcher.

Data Analysis

The data will be tabulated (ordered alphabetically by the first author name) to show the basic information about each study. In addition to the general information of the article, we will extract the following information:

- The defined AS will be collected
- the violated Principles are entered into the ontology
- the connected QAs are entered into the ontology References

References

- [AFT19] Umberto Azadi, Francesca Arcelli Fontana, and Davide Taibi. “Architectural Smells Detected by Tools: A Catalogue Proposal”. In: *Proceedings of the Second International Conference on Technical Debt*. TechDebt ’19. Montreal, Quebec, Canada: IEEE Press, 2019, pp. 88–97. doi: 10 . 1109 / TechDebt . 2019 . 00027.
- [Bro+98] William H Brown et al. *AntiPatterns: refactoring software, architectures, and projects in crisis*. John Wiley & Sons, Inc., 1998.
- [Fon+19] Francesca Arcelli Fontana et al. “PageRank and Criticality of Architectural Smells”. In: *Proceedings of the 13th European Conference on Software Architecture - Volume 2*. ECSA ’19. Paris, France: Association for Computing Machinery, 2019, pp. 197–204. isbn: 9781450371421. doi: 10 . 1145 / 3344948 . 3344982.
- [Fow18] Martin Fowler. *Refactoring: Improving the Design of existing Code*. Addison-Wesley Professional, 2018.
- [Gar+09] Joshua Garcia et al. “Toward a Catalogue of Architectural Bad Smells”. In: *Architectures for Adaptive Software Systems* (2009), pp. 146–162. doi: 10 . 1007 / 978 - 3 - 642 - 02351 - 4_10.

- [KC07] Barbara Ann Kitchenham and Stuart Charters. *Guidelines for performing Systematic Literature Reviews in Software Engineering*. Tech. rep. EBSE 2007-001. Keele University and Durham University Joint Report, July 2007.
- [LR06] Martin Lippert and Stephen Rook. *Refactoring in large software projects: performing complex restructurings successfully*. UK: John Wiley & Sons, 2006.
- [TLB19] Fangchao Tian, Peng Liang, and Muhammad Ali Babar. “How Developers Discuss Architecture Smells ? An Exploratory Study on Stack Overflow”. In: *Proceedings - 2019 IEEE International Conference on Software Architecture, ICSA 2019* February (2019), pp. 91–100. doi: 10 . 1109 / ICSA . 2019 . 00018.
- [Ver+20] Robert Verdeccia et al. “Architectural Technical Debt : A Grounded Theory”. In: *European Conference on Software Architecture (ECSA 2020)*. Ed. by A. Jansen et al. Springer, Cham., 2020, pp. 202–219. doi: [https : / / doi . org / 10 . 1007 / 978 - 3 - 030 - 58923 - 3 _14](https://doi.org/10.1007/978-3-030-58923-3_14).