Towards the Automatic Quality Evaluation of RESTful APIs Using Design Rule Violations

MSc Research Project

# Organization

|  |  |
| --- | --- |
| Examiner: | Prof. Dr. Stefan Wagner |
| Supervisors: | Dr. Justus Bogner, Sebastian Kotstein |
| Students: | Daniel Abajirov <st155165@stud.uni-stuttgart.de>, Timothy Ernst <st157344@stud.uni-stuttgart.de>, Manuel Merkel <st155131@stud.uni-stuttgart.de> |
| Timeframe: | 2022-05-30 – 2022-11-30 |

# Context & Motivation

REST(ful) APIs and web services [1] have become very popular in the industry and represent one of the commonly used ways to expose functionality via a well-defined and technology-agnostic interface. Despite their popularity, there is no standardized set of design rules for REST APIs. Different publications [2,3,4] propose rules and best practices to build cleaner and more maintainable REST APIs.

Although, existing work studied the detection of (anti-)patterns in REST APIs as well as the adherence to best practices (e.g., [6],[7]), a recent study analyzed the perception of industry experts regarding the importance and software quality impact of 82 REST API design rules [5]. Using these results, it may therefore be possible to create next-generation tool support for REST APIs that takes important or effective rules into account, and links them to quality attributes.

# Objectives and Tasks

The goal of the research project is therefore to analyze and select existing RESTful API design rules (primarily from [5]) and define metrics or heuristics for detecting their violations. Based on this analysis, a tool-supported approach with a CLI should be implemented, which automatically detects and reports these design rule violations in an actionable way, and links them to software quality attributes. Additionally, a GUI or a plugin for an IDE can be created. The final tool support should then be evaluated using a benchmark with respect to its recall and precision. More detailed research questions will be defined during the project.

# Methods

The project will be conducted primarily in an exploratory and iterative fashion, switching between analysis, implementation, and testing. To gather the necessary knowledge, a set of papers will be used to extract and analyze design rules [5,10]. Some linters to check for rule violations already exist [8], such as “Speccy” [9], most of them relying on the OpenAPI Specification [4]. These will be analyzed and potentially reused. For the development of the tool, some form of prototyping will be used [11], possibly even with a specific approach for user interfaces [12]. Benchmark tests with different RESTful API rules could be used for evaluation [13]. The detailed design should be developed by the students.

# References

[1] C. Pautasso, “RESTful web services: Principles, patterns, emerging technologies,” in Web Services Foundations, vol. 9781461475, A. Bouguettaya, Q. Z. Sheng, and F. Daniel, Eds. New York, NY: Springer New York, 2014, pp. 31–51.

[2] MASSE, Mark. REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces. " O'Reilly Media, Inc.", 2011.

[3] PALMA, Francis, et al. Semantic analysis of restful apis for the detection of linguistic patterns and antipatterns. International Journal of Cooperative Information Systems, 2017, 26. Jg., Nr. 02, S. 1742001

[4] OpenAPI Specification v3.1.0. https://spec.openapis.org/oas/v3.1.0

[5] Kotstein, S., Bogner, J. (2021). Which RESTful API Design Rules Are Important and How Do They Improve Software Quality? A Delphi Study with Industry Experts. In: Barzen, J. (eds) Service-Oriented Computing. SummerSOC 2021. Communications in Computer and Information Science, vol 1429. Springer, Cham. https://doi.org/10.1007/978-3-030-87568-8\_10

[6] Palma, F., Dubois, J., Moha, N., Guéhéneuc, YG. (2014). Detection of REST Patterns and Antipatterns: A Heuristics-Based Approach. In: Franch, X., Ghose, A.K., Lewis, G.A., Bhiri, S. (eds) Service-Oriented Computing. ICSOC 2014. Lecture Notes in Computer Science, vol 8831. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-45391-9\_16

[7] Palma, F., Gonzalez-Huerta, J., Moha, N., Guéhéneuc, Y., & Tremblay, G. (2015). Are RESTful APIs Well-Designed? Detection of their Linguistic (Anti)Patterns. ICSOC.

[8] https://nordicapis.com/8-openapi-linters/

[9] Speccy, https://github.com/wework/speccy

[10] Bogner, J., Wagner, S., Zimmermann, A. (2020). Collecting Service-Based Maintainability Metrics from RESTful API Descriptions: Static Analysis and Threshold Derivation. In: , et al. Software Architecture. ECSA 2020. Communications in Computer and Information Science, vol 1269. Springer, Cham. https://doi.org/10.1007/978-3-030-59155-7\_16

[11] N. M. Devadiga, “Tailoring architecture centric design method with rapid prototyping,” in 2017 2nd International Conference on Communication and Electronics Systems (ICCES), 2017, pp. 924–930.

[12] D. Baumer, W. Bischofberger, H. Lichter, and H. Zullighoven, “User interface prototyping-concepts, tools, and experience,” in Proceedings of IEEE 18th International Conference on Software Engineering, 1996, pp. 532–541.

[13] S. E. Sim, S. Easterbrook and R. C. Holt, "Using benchmarking to advance research: a challenge to software engineering," 25th International Conference on Software Engineering, 2003. Proceedings., 2003, pp. 74-83, doi: 10.1109/ICSE.2003.1201189.