ExArch: Enabling Architecture Traceability by LLM-based Architecture Component Name Extraction

Dominik Fuchß [1], Haoyu Liu [1], Tobias Hey [1], Jan Keim [1], and Anne Koziolek [1]

Abstract: Our paper [Fu25], published at the 22nd IEEE International Conference on Software Architecture (ICSA), investigates traceability link recovery (TLR) between software architecture documentation (SAD) and source code, leveraging large language models (LLMs) to generate intermediate artifacts.

Keywords: Traceability Link Recovery, Large Language Models, Software Architecture, Model Extraction

Introduction Software development involves various artifacts, each representing different abstraction levels and system aspects. The lack of explicit relationships between these artifacts often hinders their effective utilization. To address this, TLR techniques are employed to establish, maintain, and manage explicit trace links between artifacts. Some TLR approaches use intermediate artifacts to bridge the semantic gap, facilitating the linking of related artifacts. The Transitive links for Architecture and Code (TransArC) [Ke24] approach, for example, recovers trace links between SADs and source code using manually created software architecture models (SAMs) as intermediates. However, the reliance on such intermediate artifacts limits applicability, as they are frequently unavailable in practice.

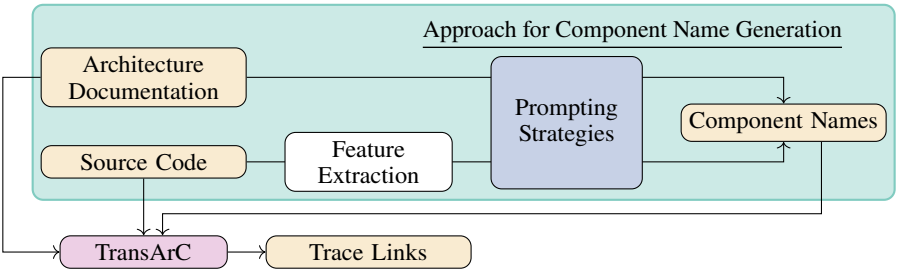

Fig. 1: Overview of the ExArch Approach for TLR [Fu25]. Artifacts in orange, prompting in blue, feature extraction in white, and TransArC [Ke24] in purple.

[1] Karlsruhe Institute of Technology, Germany,
dominik.fuchss@kit.edu, https://orcid.org/0000-0001-6410-6769;
haoyu.liu@kit.edu, https://orcid.org/0009-0002-7676-5010;
hey@kit.edu, https://orcid.org/0000-0003-0381-1020;
jan.keim@kit.edu, https://orcid.org/0000-0002-8899-7081;
koziolek@kit.edu, https://orcid.org/0000-0002-1593-3394

doi:10.18420/se2026-20

Methods We introduce ExArch, a novel approach to facilitate TLR between SAD and source code without requiring manually created SAMs. Our method utilizes the capabilities of LLMs to interpret both natural language and code, enabling the automated extraction of component names — serving as simple SAMs — from SAD and/or source code.

Research Questions

RQ1: Is the performance of architecture TLR with LLM-extracted component names as intermediate artifacts comparable to using manually created SAMs?

RQ2: Does our approach perform better than state-of-the-art TLR between SAD and code without SAMs as intermediates?

RQ3: Is the performance of current open-source LLMs comparable to the performance of closed-source ones?

RQ4: How does the performance of the approach differ when using different artifacts to generate the SAMs?

Results We compare against state-of-the-art TLR methods. ExArch achieves comparable performance to TransArC (weighted average F_1 with GPT-4o: 0.86 vs. 0.87), yet it does not require an SAM (RQ1). Also, ExArch significantly surpasses the baseline ArDoCode that does not utilize SAMs (RQ2; w. avg. F_1 0.62). On average, we show that the evaluated closed-source models perform better than the evaluated open-source models on this task (RQ3). Finally, using only SAD for generation works better than our other options (RQ4).

Conclusion In conclusion, our work indicates that large language models (LLMs) can be used to bridge the gap between SAD and source code by extracting component names. To promote replicability, transparency, and extensibility, we release the ExArch source code, datasets, and results as part of our replication package[2], enabling further research and development in this area. Our approach demonstrates that it is important how LLMs are utilized in a process: Generation of component names using LLMs performed better than state-of-the-art methods that solely rely on LLMs for TLR.

References

[Fu25] Fuchß, D. et al.: Enabling Architecture Traceability by LLM-based Architecture Component Name Extraction. In: 2025 IEEE 22nd International Conference on Software Architecture, Odense, Denmark. 2025.

[Ke24] Keim, J. et al.: Recovering Trace Links Between Software Documentation And Code. In: Proceedings of 46th IEEE International Conference on Software Engineering. 2024.

[2] Replication Package: https://doi.org/10.5281/zenodo.14506935