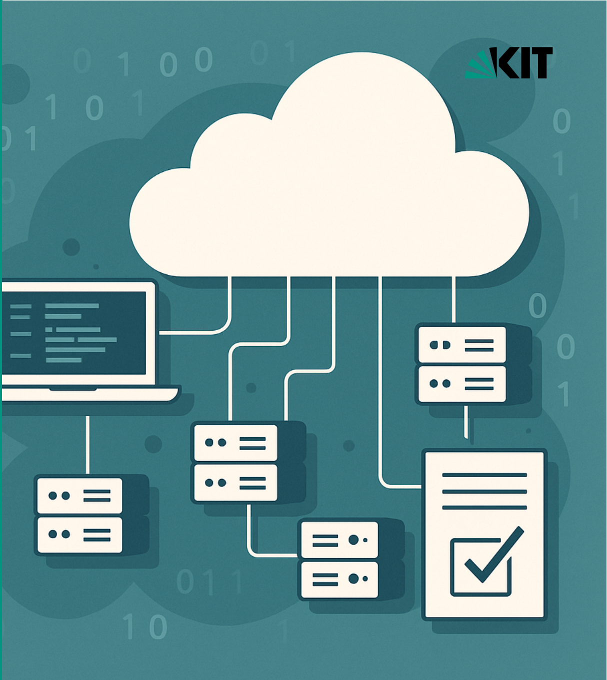
LLM-Assisted Microservice Performance Modeling

Automated Generation of Parametric Performance Models

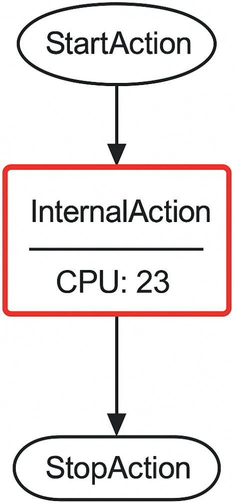
Maximilian Hummel, Nathan Hagel, Minakshi Kaushik, Jan Keim, Erik Burger, Heiko Koziolek
Karlsruhe Institute of Technology (KIT)
ABB Corporate Research Center
SSP 2025 – November 4–5, 2025 – Kiel, Germany



Motivation

Software Performance Engineering Dilemma:

- Architectural performance modeling is helpful to obtain **early** insights into system performance.
- The Palladio Component Model (PCM) [Reussner et al. 2016] is a suitable option for that. But the PCM must be enriched with realistic **resource demands**.
- Resource demands are often derived from systematic measurements. But in the early design phase, there is no system to measure yet.
- **Options to define resource demands during the early design phase:**
  - Expert modeling (manual, requires expertise)
  - Build prototype (late feedback, design changes)
  - Skip analysis (risky)



Introduction

Challenge: Early Design Software Performance Analysis

- Requires measurements or expert knowledge to calibrate PCM
- Time-consuming
- Manual effort scales poorly with system complexity

Research Gap

- Early design phase need fast, automated and reliable estimates
- Current approaches rely on measurements or manual expert estimates
- Missing: Systematic way for performance model generation in early design

Our Approach:

- Leverage Large Language Models (LLMs) to generate parametric performance models
- Input: Textual artefact (Code, documenation etc.) & target resource (CPU,Memory, Network)
- Output: Interpretable, parametric performance models

Example: Parametric Performance Model

Scenario: A microservice providing different API endpoints.

Goal: CPU utilization for one endpoint as a function of workload parameters

Parametrized Performance Model

$$\text{Avg. CPU Util. \%} = \alpha_0 \cdot \text{request\_rate} + \alpha_1 \cdot \text{payload\_size} + \alpha_2$$

- **Parameters:** Request rate, payload size, coefficients (α_i)
- **Output:** CPU utilization prediction

Related Work

Measurement-based Approaches

- Use a running system to create performance models [Spinner et al. 2015; Jindal, Podolskiy and Gerndt 2019]
- High accuracy
- Expensive and time-consuming

Related Work

Measurement-based Approaches

- Use a running system to create performance models [Spinner et al. 2015; Jindal, Podolskiy and Gerndt 2019]
- High accuracy
- Expensive and time-consuming

Model-based Approaches

- Model-driven techniques aim to provide earlier in-sights based on Architecture [Pinciroli, Aleti and Trubiani 2023]
- Early-stage applicable
- Need a running system or expert knowledge at some point for calibration

Related Work

Measurement-based Approaches

- Use a running system to create performance models [Spinner et al. 2015; Jindal, Podolskiy and Gerndt 2019]
- High accuracy
- Expensive and time-consuming

Model-based Approaches

- Model-driven techniques aim to provide earlier in-sights based on Architecture [Pinciroli, Aleti and Trubiani 2023]
- Early-stage applicable
- Need a running system or expert knowledge at some point for calibration

LLM-based Approaches

- Use LLMs for performance modeling [Nguyen-Nhat et al. 2024; Zhang, Hassan and Drechsler 2025; Hu et al. 2024]
- Process textual artefacts to derive raw numerical predictions for e.g execution time
- No parameterization, not applicable for architecture performance modeling (e.g with PCM)

Methodology Overview

Objective:

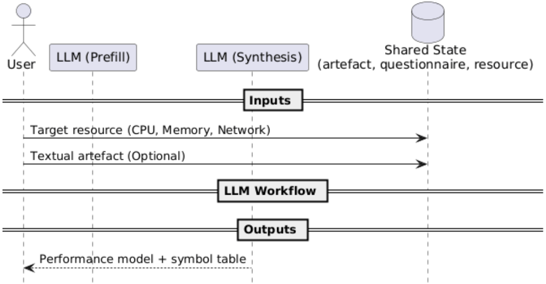
Generate parametrized performance models using an LLM workflow and a performance questionnaire

Inputs

- Target Resource: CPU, Memory, Network
- Textual Artefact (Optional): Code, Documentation etc.

Outputs

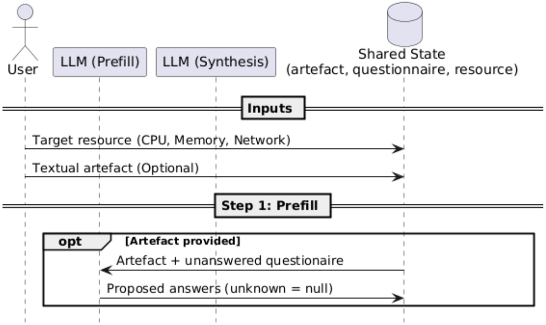
- LLM generated parametric performance model
- Symbol table with definitions
- Documented assumptions



Two-Step LLM Workflow - Step 1

Step 1: Prefill

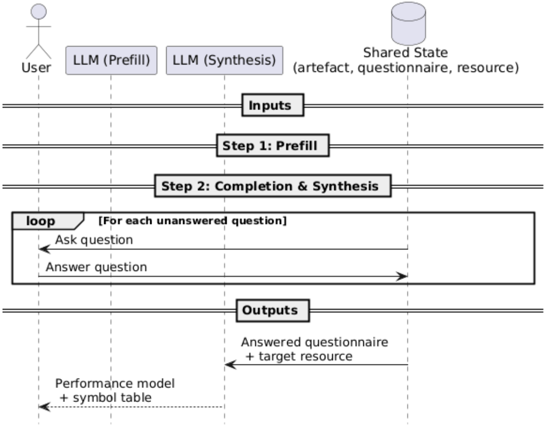
- LLM analyzes textual artifact
- Extracts known performance information
- Pre-populates **performance questionnaire** containing questions regarding:
  - Microservice business logic
  - Deployment
  - Workload characteristic



Two-Step LLM Workflow - Step 2

Step 2: Completion & Synthesis

- User answers the unanswered questions from step 1
- LLM generates performance model based on filled performance questionnaire and target resource
- LLM acts as "performance engineer"



Prompting Strategy

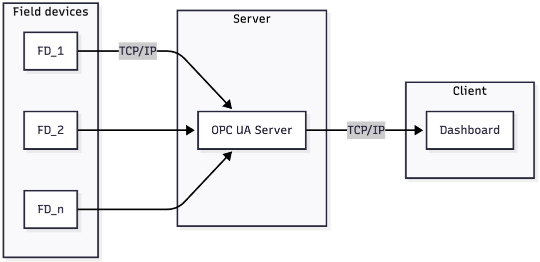
A role-based few-shot prompting approach

- Dedicated prompt templates for **Prefill** and **Synthesis**
  - **Prefill**: Supplied with an artefact and unanswered questions to elicit candidate answers for those questions based on the artefact
  - **Synthesis**: Assigns the LLM the role of an experienced performance engineer and uses the completed questionnaire to generate a performance model along with a symbol table, assumptions, and rationale
- Structured output format (JSON/YAML) with **few-shot** examples
- Clear **role** definition: "You are an expert performance engineer..."

Evaluation Setup

Use Case: OPC UA Server

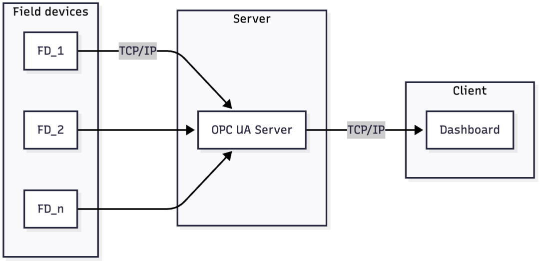
- Industry-standard protocol for data exchange between field devices (FDs) and clients in automation environments.
- Client-Server architecture
- Performance-critical application domain [Burger et al. 2019]



Evaluation Setup

Use Case: OPC UA Server

- Industry-standard protocol for data exchange between field devices (FDs) and clients in automation environments.
- Client-Server architecture
- Performance-critical application domain [Burger et al. 2019]



Hardware Configuration

- Intel Core Ultra 7 165H
- 32 GB RAM
- Windows 11
- Isolated measurement environment

Workload Characteristics

- OPC UA signals: 100–10,000
- CPU utilization measured
- Multiple measurement runs
- Statistical validation

Comparison & Evaluation Criteria

Baseline

Measured CPU utilization from actual system running the OPC UA Server

Artefact Input Variants (3 Artefact Types)

- **None:** Manual answering questionaire without providing an artefact
- **Documentation:** Use documentation as artefact
- **Code:** Use source code as artefact

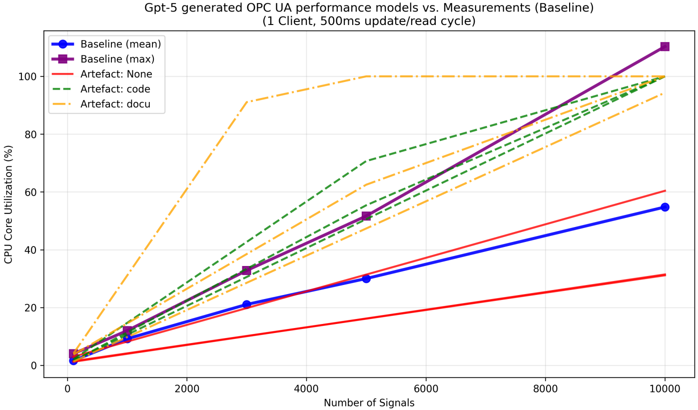
Evaluation Criteria

- **Trend consistency:** Does LLM-generated performance models follow actual behavior?
- **Accuracy:** How close to measurements?
- **Artefact influence:** Impact of different input types

Results Overview

Key Finding:

All generated performance models reproduce increasing CPU load trend, but accuracy varies



- Nine models generated (three per artefact type)
- **Note:** Only two red lines visible (two models identical)

Manual (None)

Underestimates max values

Code-based

Best approximation, correct slope

Docu-based

Similar quality, slightly less precise

Conclusion & Future Work

Current Limitations

- Single microservice case study
- One hardware configuration
- CPU-focused metrics only
- Limited workload scenarios

Future Research

- Multiple services & platforms
- Memory, network, I/O metrics
- Hybrid LLM + measurement (MCP)
- Integration with Palladio

Questions & Discussion

Thank You!

Contact

Maximilian Hummel

Maximilian.Hummel@kit.edu

Code & Evaluation Results

Available at: https://doi.org/10.5281/zenodo.17310391

SSP 2025 – Kiel, Germany

Burger, Andreas et al. (2019). 'Bottleneck Identification and Performance Modeling of OPC UA Communication Models'. In: *Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering*. ICPE '19. Mumbai, India: Association for Computing Machinery, pp. 231–242. ISBN: 9781450362399. DOI: 10.1145/3297663.3309670. URL: https://doi.org/10.1145/3297663.3309670.

Hu, Jiekang et al. (2024). 'LLM4MDG: Leveraging Large Language Model to Construct Microservices Dependency Graph'. In: *2024 IEEE 23rd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom)*, pp. 859–869. DOI: 10.1109/TrustCom63139.2024.00128.

Jindal, Anshul, Vladimir Podolskiy and Michael Gerndt (2019). 'Performance Modeling for Cloud Microservice Applications'. In: *Proceedings of the 2019 ACM/SPEC International Conference on Performance Engineering*. ICPE '19. Mumbai, India: Association for Computing Machinery, pp. 25–32. ISBN: 9781450362399. DOI: 10.1145/3297663.3310309. URL: https://doi.org/10.1145/3297663.3310309.

Nguyen-Nhat, Minh-Khoi et al. (2024). 'LLMPerf: GPU Performance Modeling meets Large Language Models'. In: *2024 32nd International Conference on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS)*, pp. 1–8. DOI: 10.1109/MASCOTS64422.2024.10786558.

Pinciroli, Riccardo, Aldeida Aleti and Catia Trubiani (2023). 'Performance Modeling and Analysis of Design Patterns for Microservice Systems'. In: *2023 IEEE 20th International Conference on Software Architecture (ICSA)*, pp. 35–46. DOI: 10.1109/ICSA56044.2023.00012.

Reussner, Ralf H et al. (2016). *Modeling and simulating software architectures: The Palladio approach*. MIT Press.

Spinner, Simon et al. (2015). 'Evaluating approaches to resource demand estimation'. In: *Performance Evaluation* 92, pp. 51–71.

Zhang, Weiyan, Muhammad Hassan and Rolf Drechsler (2025). 'LLM-assisted Performance Estimation of Embedded Software on RISC-V Processors'. In: *2025 IEEE 28th International Symposium on Design and Diagnostics of Electronic Circuits and Systems (DDECS)*, pp. 7–12. DOI: 10.1109/DDECS63720.2025.11006767.