Redefining abstractions in software engineering using AI

To deal with ever-increasing complexity and scale of software, Software Engineering has relied heavily on abstractions. These abstractions range from abstracting binary nature of computing hardware to compilers, interpreters, high-level programming languages, and abstractions in the form of software architecture. While these abstractions help to produce software that can perform complex functionalities, they also introduce certain overhead in terms of complexity and performance. Hence, besides solving real business and societal problems, software engineering is also concerned with solving challenges that originated due to the abstractions we introduced.

Artificial intelligence, and especially generative AI in form of LLMs keeps proving that it can assist software engineers with a wide spectrum of tasks. In this work, we aim to select one of the many abstractions introduced in software engineering and use AI to try and bridge it, i.e., remove the abstraction and use AI as an interpreter between engineers and computers.

Expected benefits are more efficient code optimized for the underlying hardware and more efficient creation of software solutions. In return, we expect to see higher performance of software, spending less computing resources improving sustainability, and decreasing the price of digitalisation.

To execute this work, we will first select one abstraction in software engineering and quantify its benefits and overhead. Then, we will summarize existing AI tools and select the most adequate ones for the task at hand. Then, we will create a conceptual solution, reason how it fits into standard development methodologies, and implement it in a form of a toolchain. Finally, we aim to evaluate the solution in an industrial environment.