A Command Line Tool for Managing Recurring Architectural Decisions

Student

Raphael Schellander

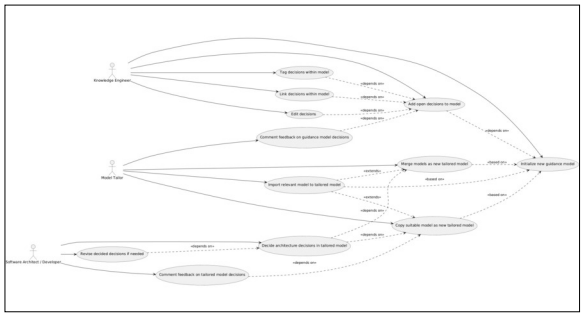
Introduction: As decisions regarding software architecture become increasingly important for the long-term maintainability of complex systems, there is a growing need for tools that can document and guide these decisions. Building on an initial conceptual exploration of decision management tooling and the creation of a Clean Architecture decision handbook, this follow-up project aimed at developing a fully functioning command line prototype ready for practical use. The prototype intends to help software architects and other stakeholders manage recurring architectural decisions in a structured, consistent and efficient manner.

Approach: The primary objective of this project was to design and implement a Command Line Interface (CLI) tool that supports the entire lifecycle of Architectural Decision Records (ADRs). The tool enables users to create, tag, link, justify, and validate decisions efficiently. Thanks to its metadata indexing capability and its support for distributed, editable Markdown files, the tool is able to handle large decision models flexibly. A notable feature is the tool's capability to support guidance modeling, enabling the import, merging, and instantiation of decisions across projects to facilitate the reuse of curated architectural knowledge. Particular attention was paid to nesting and linking decisions to ensure that the decision models can reflect complex architectural reasoning across multiple logical layers and project contexts. The CLI tool, called ADG, was implemented in Go and developed iteratively using agile practices, clean architecture principles, modular design, version control and unit testing. ADG offers a total of 18 commands, including init, add, edit, decide, tag, link, comment, validate, view, import, copy/clone and merge.

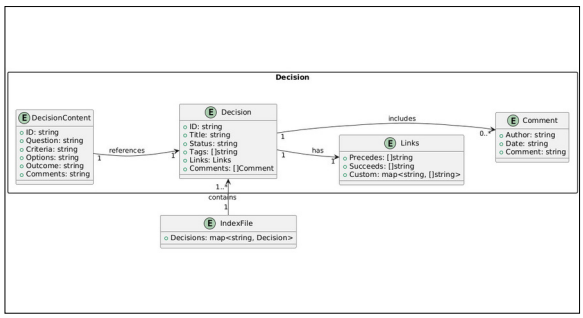
A validation experiment involving two participants, one experienced architect and one graduate student in Computer Science, was conducted to evaluate the tool's usability and suitability for real-world scenarios. The experiment consisted of a hands-on walkthrough in which participants were guided through the process of extending and tailoring an existing decision model step by step.

Result: The thesis result is an extensible CLI prototype ready for experimentation and adoption. It allows three different user roles to work together to manage architectural decisions throughout the entire lifecycle: Knowledge Engineers can define reusable decision templates and guidance models to address recurring architectural challenges. Model Tailors can then adapt and instantiate these models for specific projects, selecting only relevant decisions and customizing them as required. Meanwhile, Software Architects/Developers can document, link and revise their decisions in a structured format, benefiting from model-driven guidance and traceability. This division of roles promotes consistency across teams. Ultimately, ADG provides a practical, extensible solution for improving the creation, reuse and maintenance of architectural decisions in modern software projects.

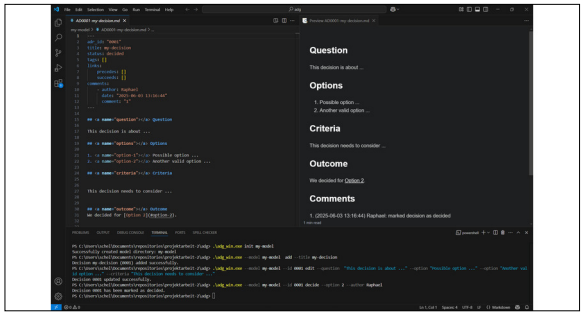
Use Case Diagram
Own presentment



Domain Model
Own presentment



Workflow of initializing a model, then adding, editing, and deciding a decision (in Visual Studio Code)
Own presentment



Advisor
Prof. Dr. Olaf Zimmermann

Subject Area
Computer Science