Low-Code vs. Model-Driven Architecture

Markus Reiter

October 20, 2020

supervised by Prof. Dr. Ruth Breu

Outline

- Motivation
- Low-Code Architecture
- Model-Driven Architecture
- Criticisms
- Planned Project Procedure

Motivation

- What can and can’t low-code and model-driven tools do?
- Are they a viable alternative to traditional development?
- When to choose one approach over the other?

Low-Code Architecture

- provides pre-built application components
- graphical user interface for creating both the application logic as well as the user interface
- typically aimed at end-users rather than developers

Model-Driven Architecture

- provides a set of guidelines for the structuring of specifications
- code (fully or partially) generated from models, e.g. from UML diagrams
- aimed at developers to speed up development

Criticisms

- Low-Code Architecture
  - Unsuitable for implementing scalable and mission-critical applications.
  - Increase in unsupported applications built by shadow IT, i.e. applications which are not controlled by a company's IT department.
- Model-Driven Architecture
  - UML diagrams lack details included in the code itself.
  - "the Code is the design" - Should models be derived from code instead of code from models?
- Do these approaches actually make development easier and cheaper?

Planned Project Procedure

- Evaluate a set of Low-Code tools.
- Evaluate a set of Model-Driven tools.
- For each tool, determine what it can and can't do.
- Compare the two sets with each other and draw some conclusions.