Abstract

In 1999, SAP started to combine the Unified Modeling Language (UML) and the Fundamental Modeling Concepts (FMC) language. The result is an SAP internal standard for modeling, called Technical Architecture Modeling (TAM). TAM comprises block diagrams, component diagrams, package diagrams, class diagrams, activity diagrams, sequence diagrams, state diagrams, and use case diagrams. TAM is used for both conceptual modeling as well as design modeling.

While many works on reasoning on conceptual schemas focus on class diagrams and state diagrams, the most often used diagram type at SAP is the block diagram. For example, class models are used rarely, as they are “too close to real code.” In general, developers and architects prefer structural diagrams (e.g., block diagrams), thus we need to ask ourselves the questions, if we can reason over such models and what kind of properties help to improve the software development.
Software Development Life Cycle

When are Models Used?

- Core processes
  - Define first release
  - Define next release
  - Engineer requirements
  - Plan and prepare launch & ramp-up
  - Transfer ramp-up knowledge
  - Evaluate with customers
  - Create prototype
  - Define portfolio
  - Generate & manage ideas
  - Define solution business cases for next release portfolio
  - Manage development program
  - Create and document architecture
  - Write and translate product document
  - Assemble product
  - Validate product
  - Create Service
  - Develop software
  - Plan and execute testing
  - Deliver product
  - Manage Development Landscape
  - Validate solution
  - Package solutions for products in default release

- Management processes
  - Plan and prepare launch & ramp-up
  - Transfer ramp-up knowledge
  - Evaluate with customers
  - Create prototype
  - Define portfolio
  - Generate & manage ideas
  - Define solution business cases for next release portfolio
  - Manage development program
  - Create and document architecture
  - Write and translate product document
  - Assemble product
  - Validate product
  - Create Service
  - Develop software
  - Plan and execute testing
  - Deliver product
  - Manage Development Landscape
  - Validate solution
  - Package solutions for products in default release

- Support processes
  - Provide patches for security vulnerabilities detected by external security researchers
  - Prepare, execute and drive mass adoption via GTM launch & ramp-up
  - Execute launch and drive mass adoption
  - Execute ramp-up operations
  - Plan and prepare launch & ramp-up
  - Transfer ramp-up knowledge
  - Evaluate with customers
  - Create prototype
  - Define portfolio
  - Generate & manage ideas
  - Define solution business cases for next release portfolio
  - Manage development program
  - Create and document architecture
  - Write and translate product document
  - Assemble product
  - Validate product
  - Create Service
  - Develop software
  - Plan and execute testing
  - Deliver product
  - Manage Development Landscape
  - Validate solution
  - Package solutions for products in default release

- Current QMS scope, corresponding to former PIL: G3, G4, P2D, D2P, P2R, P2P
Conclusion and Next Steps

Many open questions:

- How to make class models “more abstract” to use them early in development
- How to integrate reasoning at later steps (e.g., datatype definitions in a PL)
- How to link the different models (diagrams) for reasoning (behavioral models)
- Ultimately: How to reason over different model types
- What kind of reasoning can be done on block diagrams

What we are currently starting

- Motivate the use of “refined” block diagrams (including technical details)
- Development of “light-weight” reasoning techniques supporting threat-models
  - Exclude certain threats/countermeasures
  - Propagate threats/countermeasures
  - Infer requirements for models/implementation in later development steps

Goal: Reduce effort necessary for passing production quality checks/validation