ConIPF Variability Modelling Framework

Jan Bosch
Professor of Software Engineering
University of Groningen, Netherlands
Jan@JanBosch.com
http://www.janbosch.com

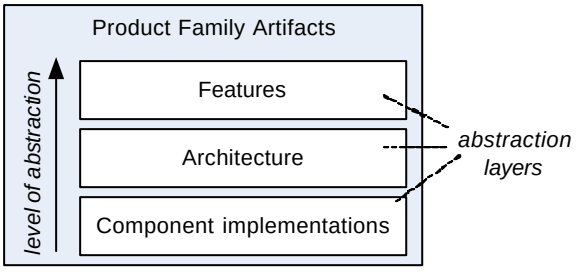
Outline

- requirements
- variation points
- dependencies
- COVAMOF
- conclusion

Requirements

- Uniform and first-class representation of variation points in all abstraction levels
- Hierarchical organization of variability representation
- Dependencies, including complex dependencies, should be treated as first class citizens in the modeling approach.
- The interactions between dependencies should be represented explicitly

Variation Points

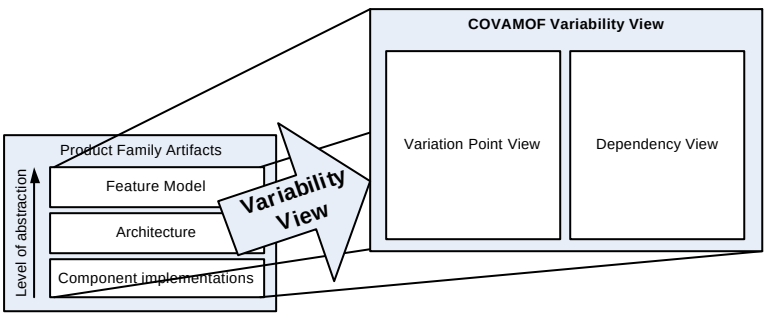


- Type: Optional, alternative, optional variant, variant, value
- Realization technique
- Binding time
- State: Open, Closed

Dependencies

- Simple vs Complex
- Statically analyzable vs Dynamically analyzable
- Dependency Interaction

COVAMOF Variability View



COVAMOF - Notation

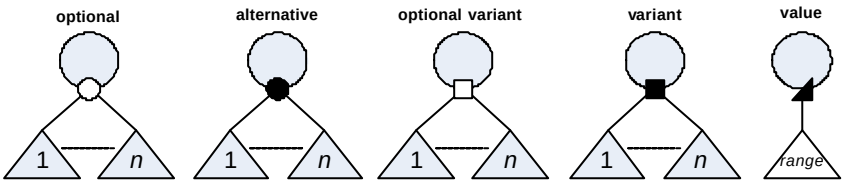
- Notation

Variation Point | Variant | Dependency | Relation

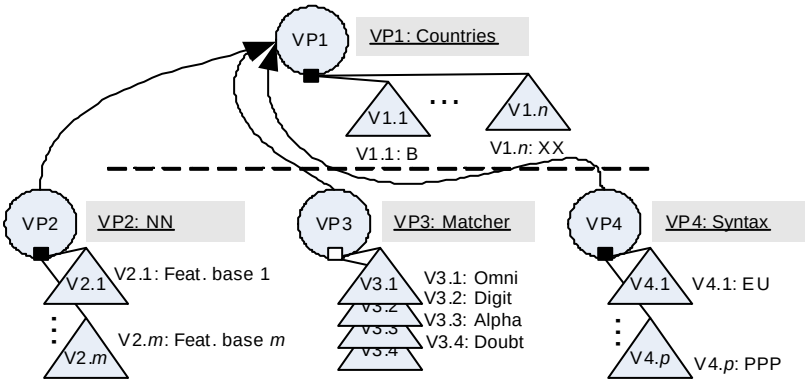
- Relations:
  - Realization relation
  - Artifact relation
  - Dependency interactions

Variation Points

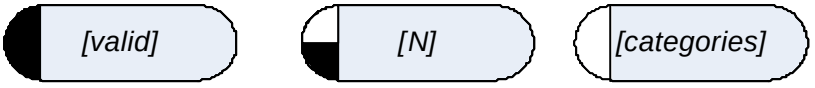
- Description
- State
- Rationale
- Realization mechanism
- Binding time



Example



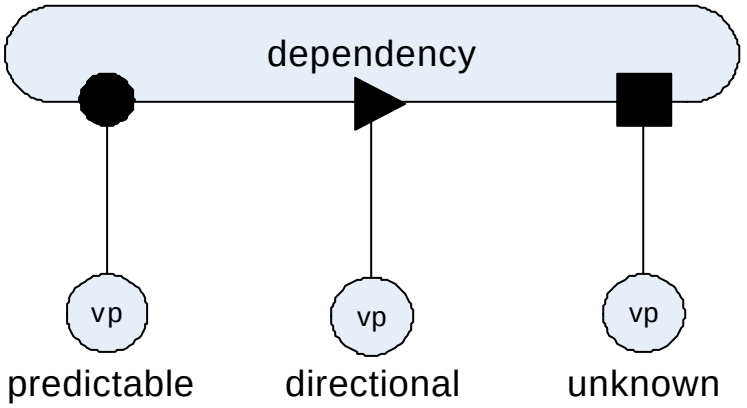
Dependencies



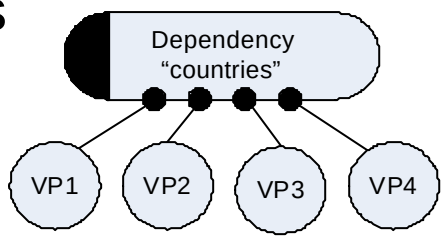
logical numerical nominal

- Description
- Validation time
- Associated variation points
- Dependency interaction

Association types



Examples

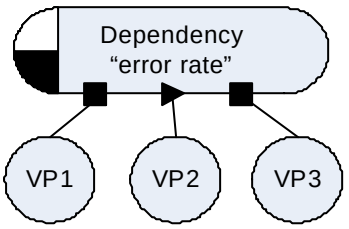


VP1: Countries

VP2: NN

VP3: Matcher

VP4: Syntax

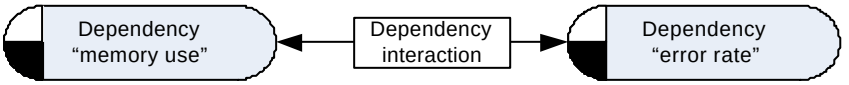


VP1: NN

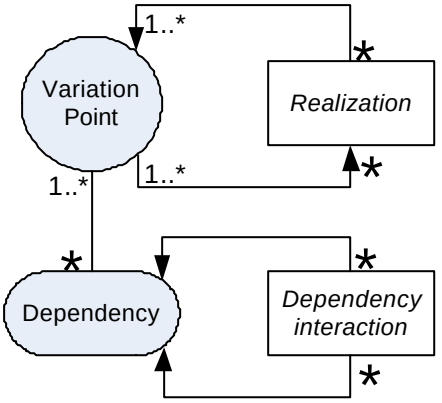
VP2: Matcher

VP3: Isolate

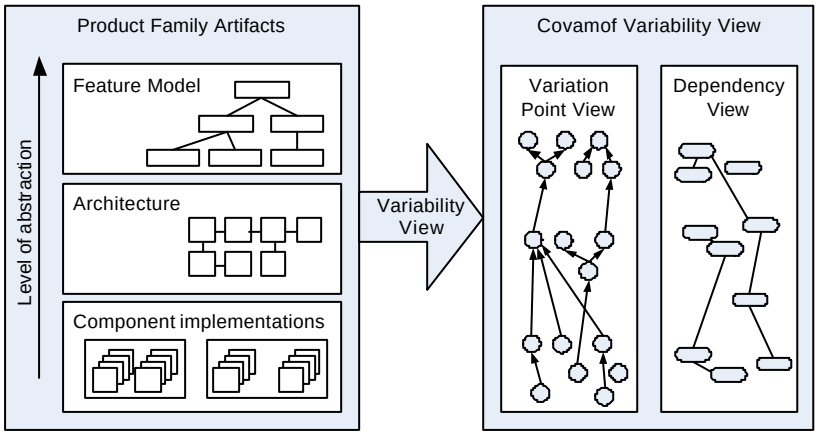
Example dependencies



CVV Meta-model (excerpt)



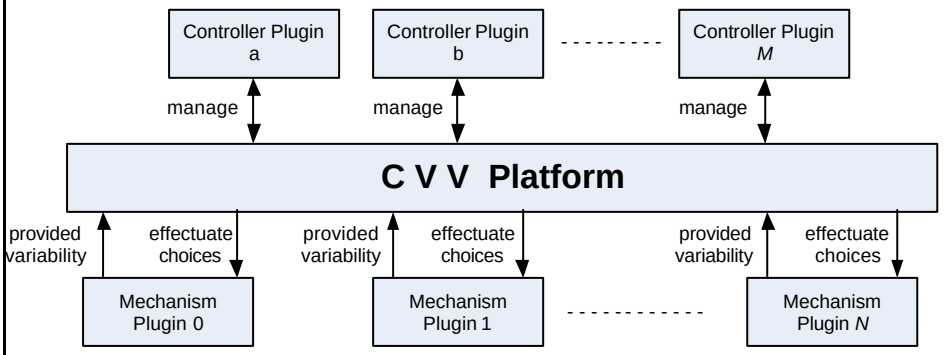
COVAMOF Variability View



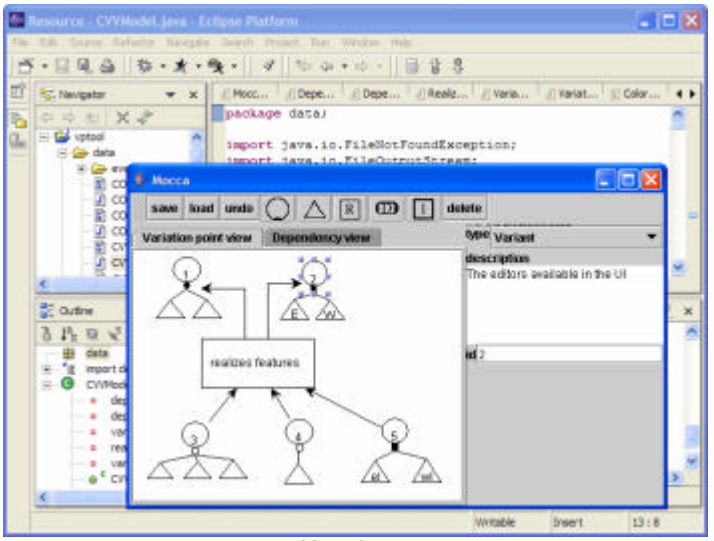
Tool Support: Mocca (1)

- Uniform and first-class representation of variation points and dependencies in all abstraction levels.
- Intrinsic and extrinsic models
- Support multiple views
- Cover complete lifecycle

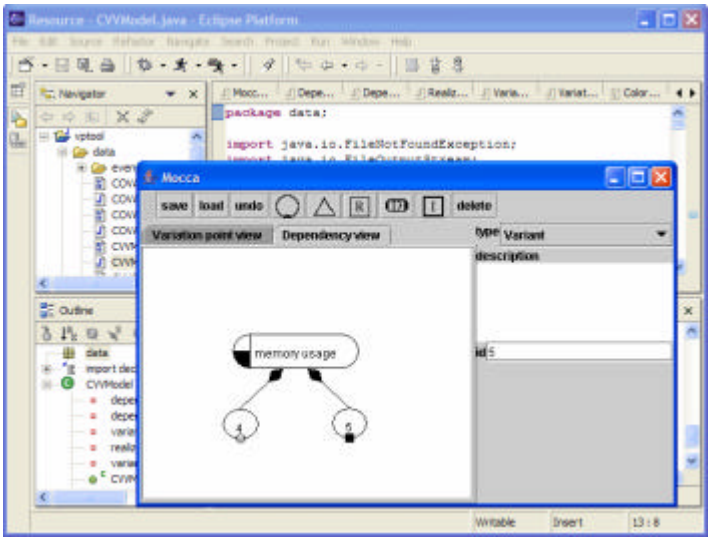
Tool Support: Mocca (2)



Mocca: Variation Point View



Mocca: Dependency View



Conclusion

- Uniform and first-class representation of variation points in all abstraction levels
- Hierarchical organization of variability representation
- Dependencies, including complex dependencies are treated as first class citizens
- The interactions between dependencies are represented explicitly

Related work

- A Meta-model for Representing Variability in Product Family Development (Bachman et al., 2003)
- Mapping Variabilities onto Product Family Assets (Becker, 2003)
- Generic Modelling using UML Extensions for Variability (Clauss, 2001)
- Multiple-View Meta-Modeling of Software Product Lines (Gomaa & Shin, 2002)
- A Composable Software Architecture for Consumer Electronics Products (van Ommering, 2000)
- Systematic Integration of Variability into Product Line Architecture Design (Thiel & Hein, 2002)