



## Universal Variability Language: Current State

MODEVAR'25 | Contributions by many great people, Talk by Sebastian Krieter | 03.02.2024

# Universal Variability Language

- Community Effort within MODEVAR initiative
- Textual format for variability models
- Simplify exchange

```
features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Cheese {Calories 5}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      "Greetings on box"

constraints
  Pineapple => Ham
```

# Universal Variability Language

- Community Effort within MODEVAR initiative
- Textual format for variability models
- Simplify exchange
- Simple core language
  - Boolean constraints & features
- Extensions for complex expressions

```
features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Cheese {Calories 5}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      "Greetings on box"

constraints
  Pineapple => Ham
```

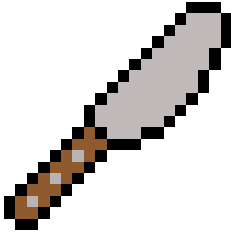
# Universal Variability Language

- Community Effort within MODEVAR initiative
- Textual format for variability models
- Simplify exchange
- Simple core language
  - Boolean constraints & features
- Extensions for complex expressions
- **Let's take a look!**

```
features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Cheese {Calories 5}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      "Greetings on box"

constraints
  Pineapple => Ham
```

# Language Levels Tradeoff



VS



- + Simple
- + Easy to understand
- Limited applicability

- + Covers more use cases
- Complex
- Harder to understand

# Language Levels Overview

## Boolean

- Boolean constraints & features
- Feature attributes for information

```
features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Cheese {Calories 5}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      "Greetings on box"
constraints
  Pineapple => Ham
```

## Arithmetic

- Numeric constraints over feature attributes
- Expressions such as ==

```
include
  Arithmetic.*
features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Cheese {Calories 5}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      "Greetings on box"
constraints
  Pineapple => Ham
  sum(Calories) < 28
```

## Type

- Feature types
- Constraints over typed features

```
include
  Arithmetic.*
  Type.*
features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Integer Cheese {Calories 5, Unit 'g'}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      String "Greetings on box"
constraints
  Pineapple => Ham
  sum(Calories) < 28
  Cheese < 300
  len("Greetings on box") < 100
```

# Language Levels The Pain

```
include
  Arithmetic.*

features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Cheese {Calories 5}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      "Greetings on box"

constraints
  Pineapple => Ham
  sum(Calories) < 28
```

```
features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Cheese {Calories 5}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      "Greetings on box"

constraints
  Pineapple => Ham
```

```
include
  Arithmetic.*
  Type.*

features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Integer Cheese {Calories 5, Unit 'g'}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      String "Greetings on box"

constraints
  Pineapple => Ham
  sum(Calories) < 28
  Cheese < 300
  len("Greetings on box") < 100
```



# Language Levels The Pain

```
include
  Arithmetic.*

features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Cheese {Calories 5}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      "Greetings on box"

constraints
  Pineapple => Ham
  sum(Calories) < 28
```

```
features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Cheese {Calories 5}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      "Greetings on box"

constraints
  Pineapple => Ham
```

```
include
  Arithmetic.*
  Type.*

features
  Pizza
    mandatory
      Dough {Calories 10, Type 'Wheat'}
      Integer Cheese {Calories 5, Unit 'g'}
      Sauce
        alternative
          Tomato {Calories 2}
          Pesto {Calories 4}
    optional
      Mushrooms {Calories 1}
      Ham {Calories 7}
      Pineapple {Calories 2}
      String "Greetings on box"

constraints
  Pineapple => Ham
  sum(Calories) < 28
  Cheese < 300
  len("Greetings on box") < 100
```





# Language Levels    How to Resolve?

features

Root

optional

Child1 {Weight 3}

Child2 {Weight 2}

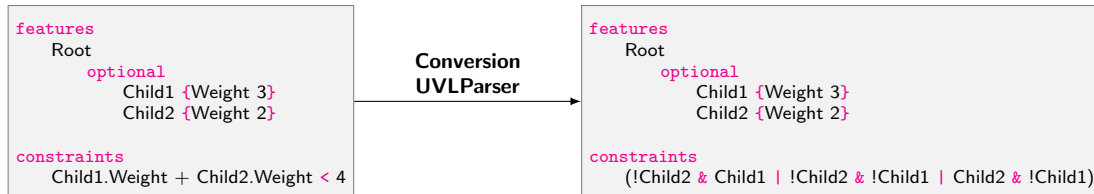
constraints

Child1.Weight + Child2.Weight < 4

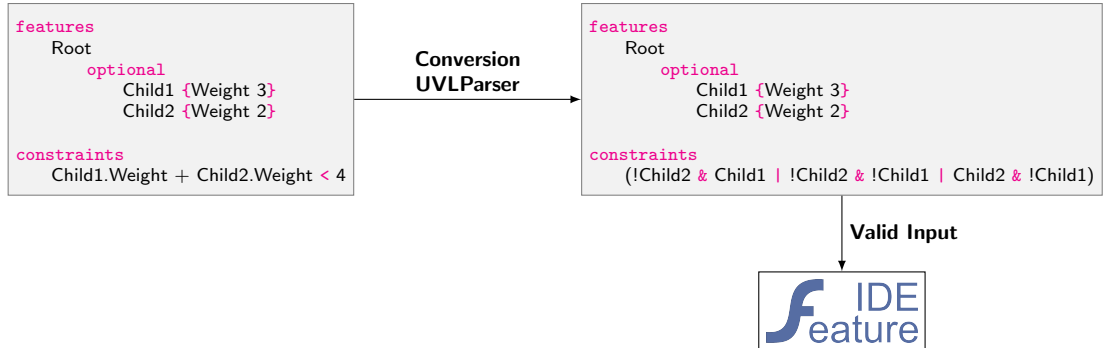
Invalid Input



# Language Levels    How to Resolve?



# Language Levels    How to Resolve?



# -----> Conversion Strategy

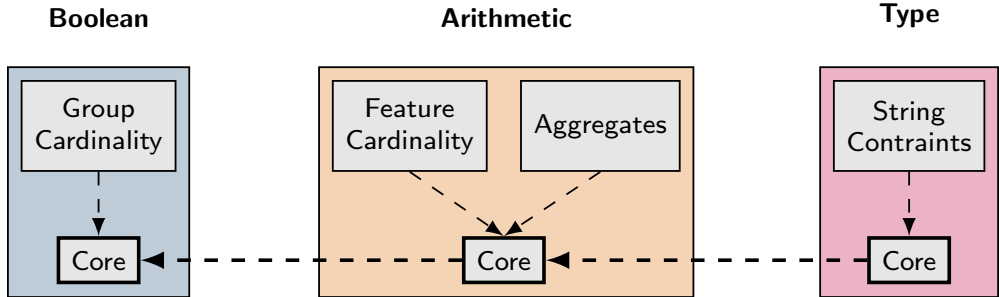


Figure: Language Levels in UVL

# What Can We Do? Parsing

## UVL-Parser


- ANTLR-based
  - Parser generator for many languages
- Currently supported: Java, Python,  JavaScript



Figure: UVL-Parser GitHub Repo

# What Can We Do? Editing

## UVLS

- Textual editing
- Language Server for UVL
- Extension VSCode
- Support for full language



Figure: UVLS GitHub Repo

## UVL-Playground

- Web-based
- Internally uses UVLS
- Includes small UVL tutorial



Figure: UVL Playground

## FeatureIDE

- Graphical Editing
- Limited to Boolean UVL



Figure: FeatureIDE Git Repo

# What Can We Do? Conversion

## TraVarT

- Convert between variability languages
  - Feature models
  - Decision models
  - OVM



Figure: TraVarT GitHub Repo

## UVL Parser Conversions

- Convert between language levels
- Included in Java Parser (Meta Model)



Figure: Meta Model GitHub Repo

# What Can We Do? Analysis

## FLAMA

- Python-based
- Common analyses: SAT, counting
- Reasoning engines: SAT, SMT, BDDs



FLAMA GitHub Repo

## FeatJar

- Java-based
- ...
- ...



FeatJar GitHub Repo



# What Can We Do? Collections

## UVL-models

- Small collection
- Translated from various variability languages
- Limited to boolean



Figure: UVL-Models GitHub Repo

## Feature-Model Benchmark

- GitHub Repository
- > 2,500 models with at least 100 features
- UVL, DIMACS



Figure: Benchmark GitHub Repo

## UVLHub

- Share UVL models
- Automated analysis for each upload
- > 1,500 models available



Figure: UVLHub GitHub Repo

# Universal Variability Language: Current State

- Language as community effort
- Extensible language design
- Language Features
  - Language levels
  - Conversion strategies
  - Import mechanism
- Available tool support:
  - Parsing
  - Editing
  - Conversion
  - Analysis
  - Collections
- **What do we need?**



Figure: <https://universal-variability-language.github.io>

# Universal Variability Language: Current State

**1. Universal Variability Language**

**2. Language Levels**

**3. What Can We Do?**