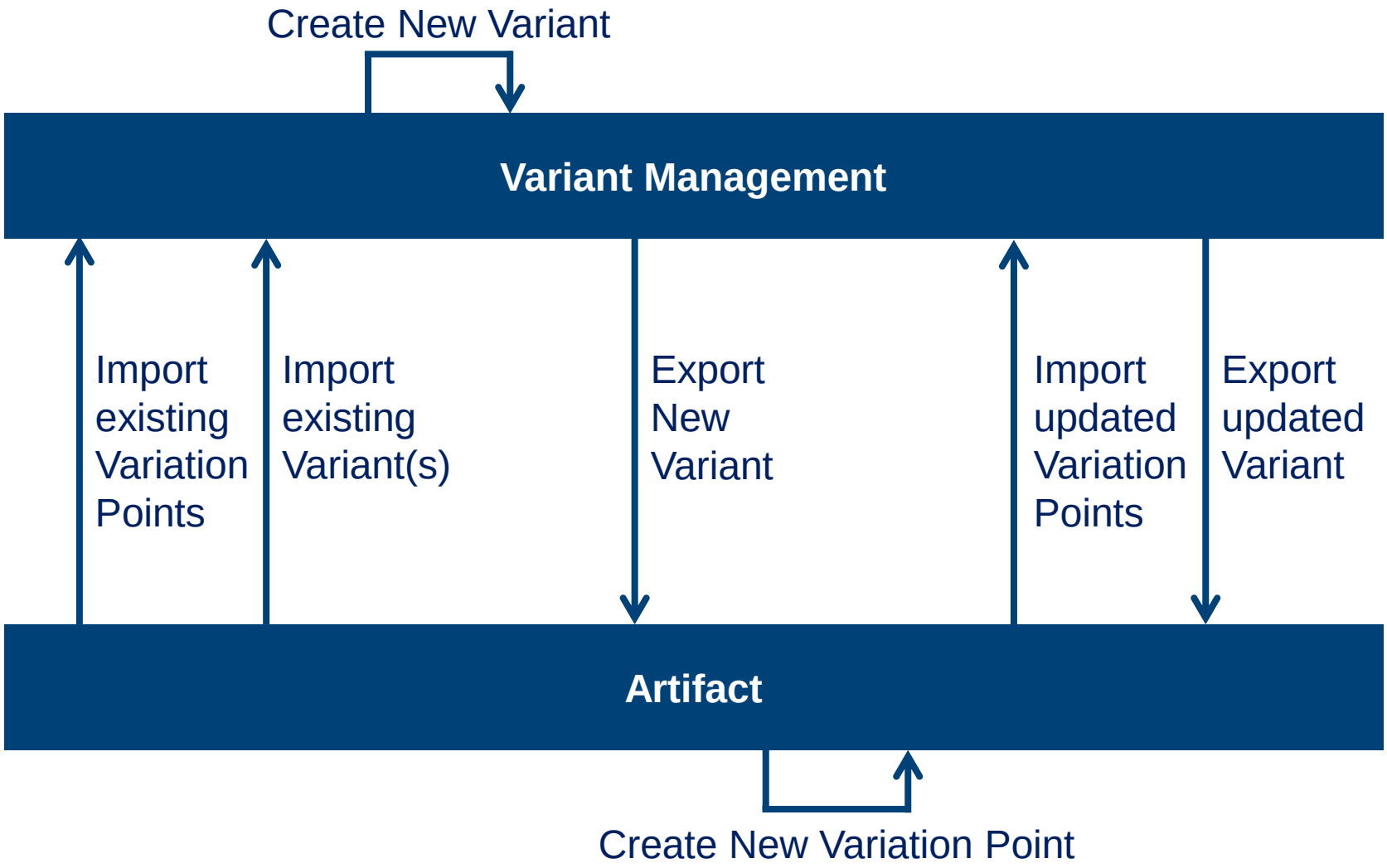
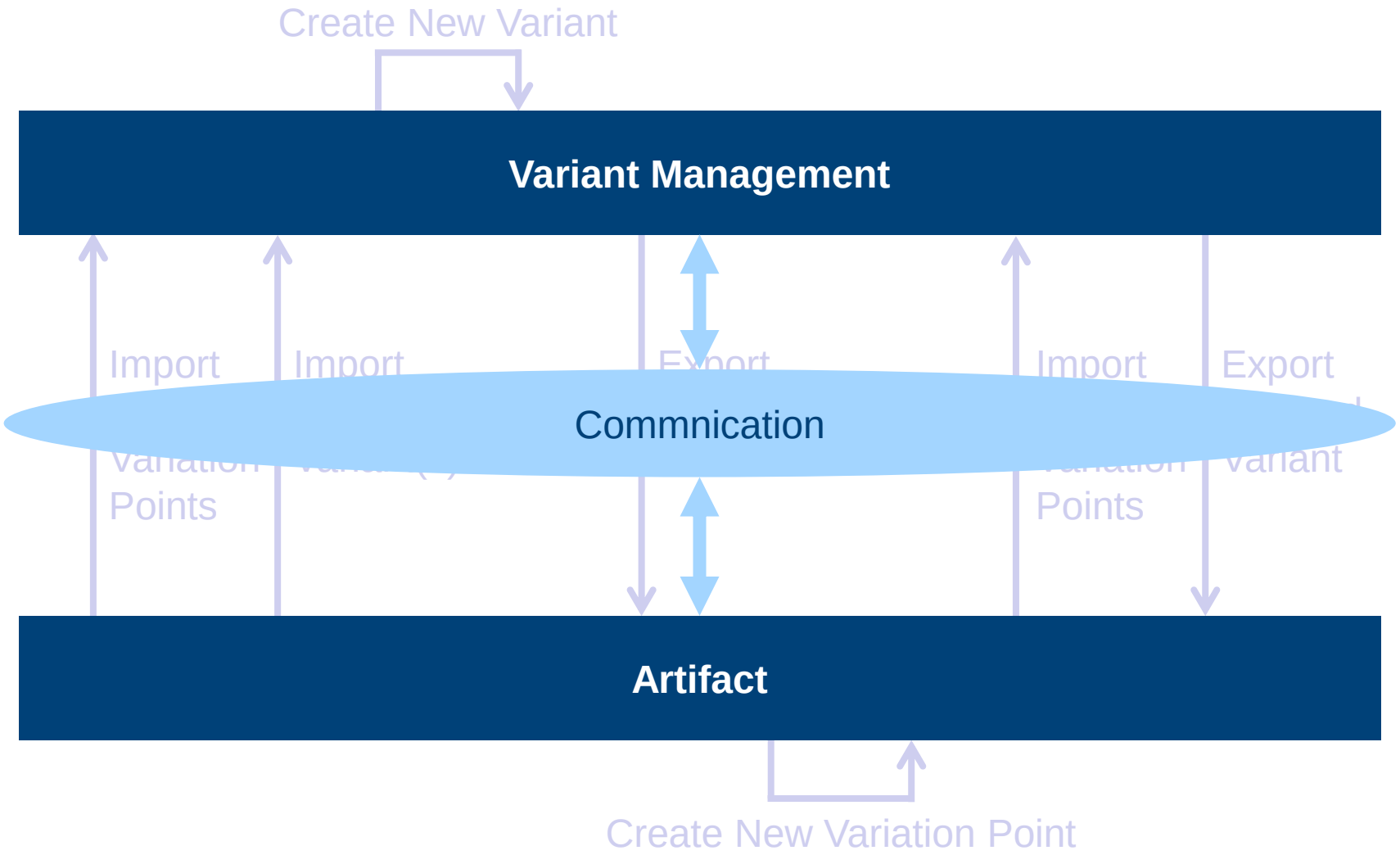
The Variability Exchange Language

Michael Himsolt, Daimler AG
Martin Große-Rhode, FOKUS
Michael Schulze, pure-systems
Sandro Schulze, TU Braunschweig

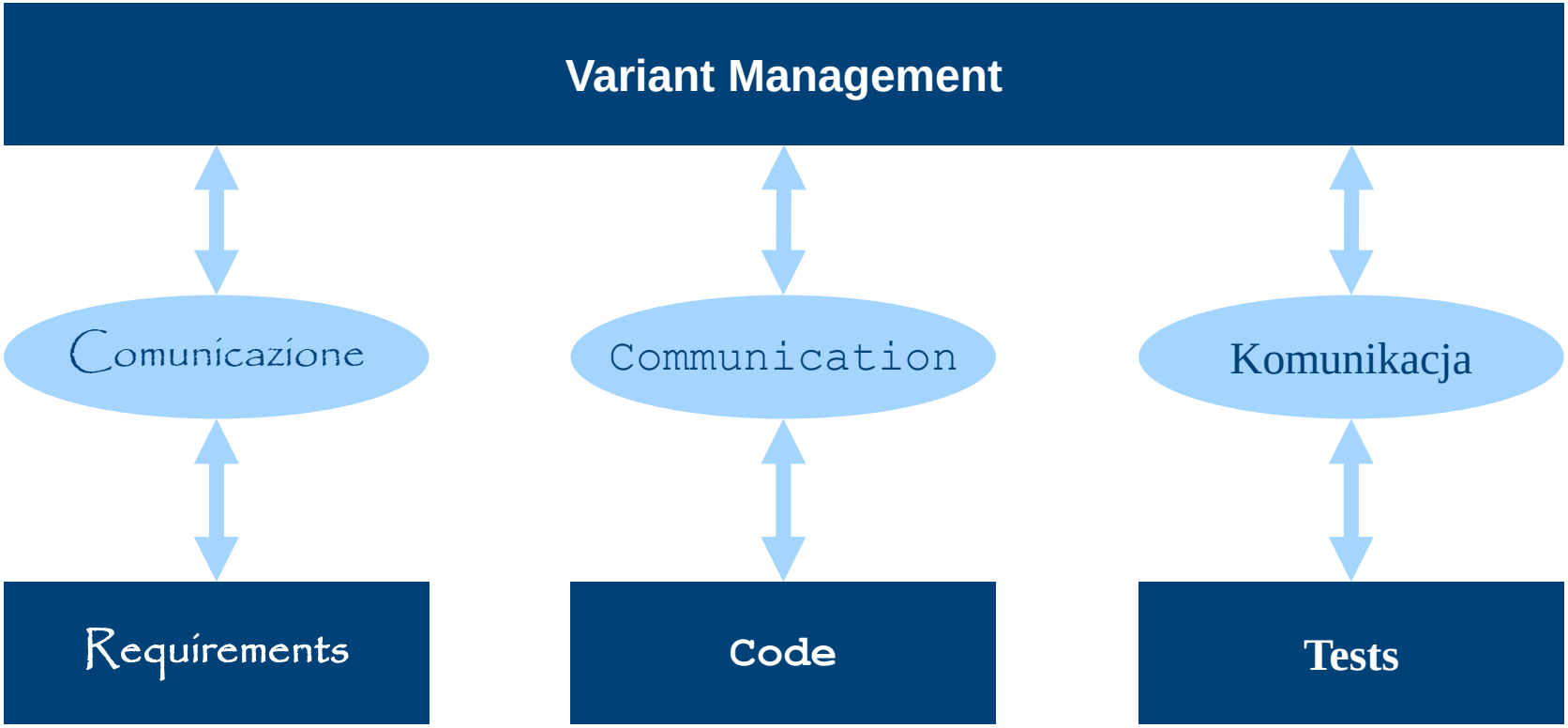
A Typical Workflow



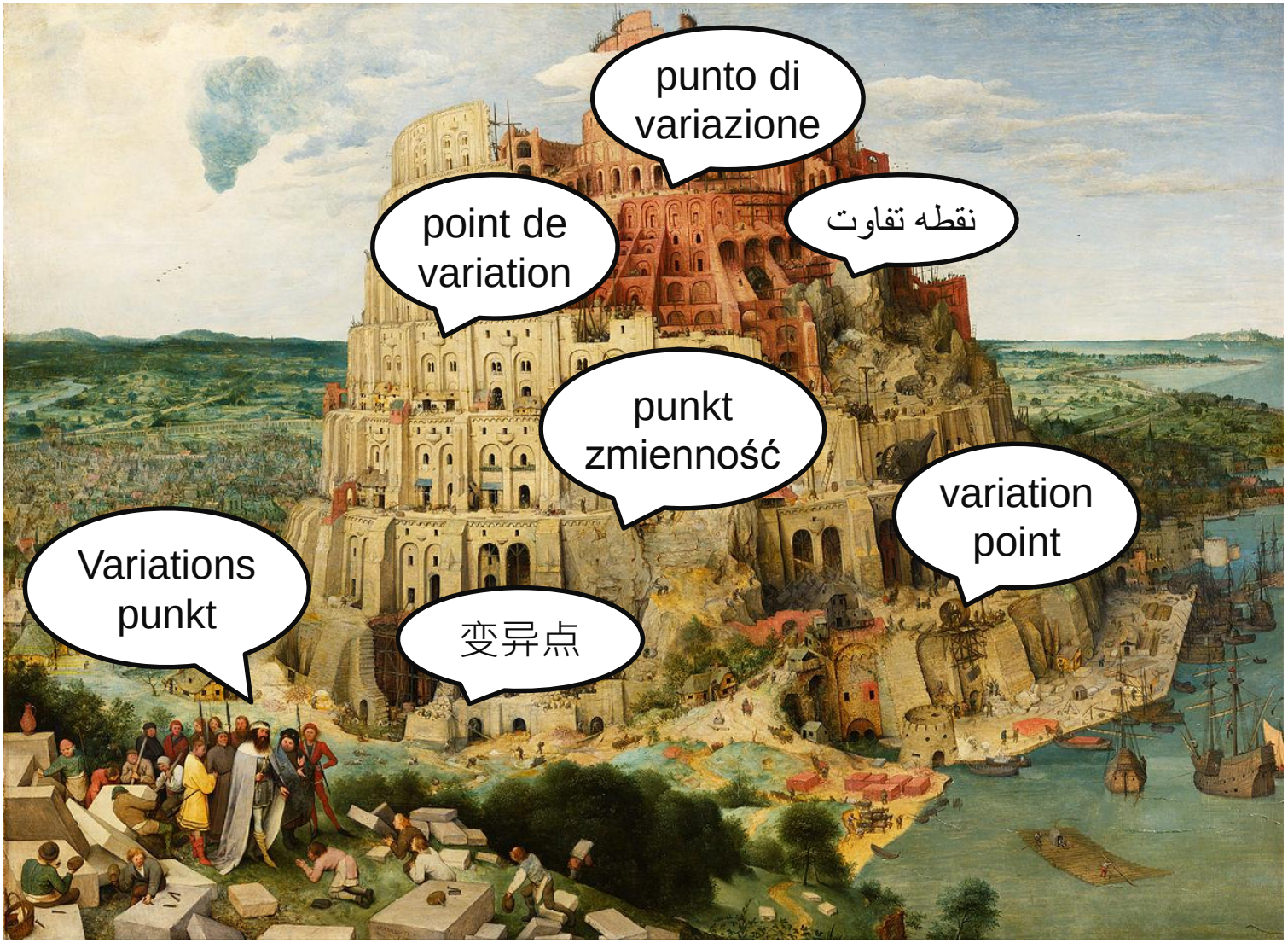
A Typical Workflow



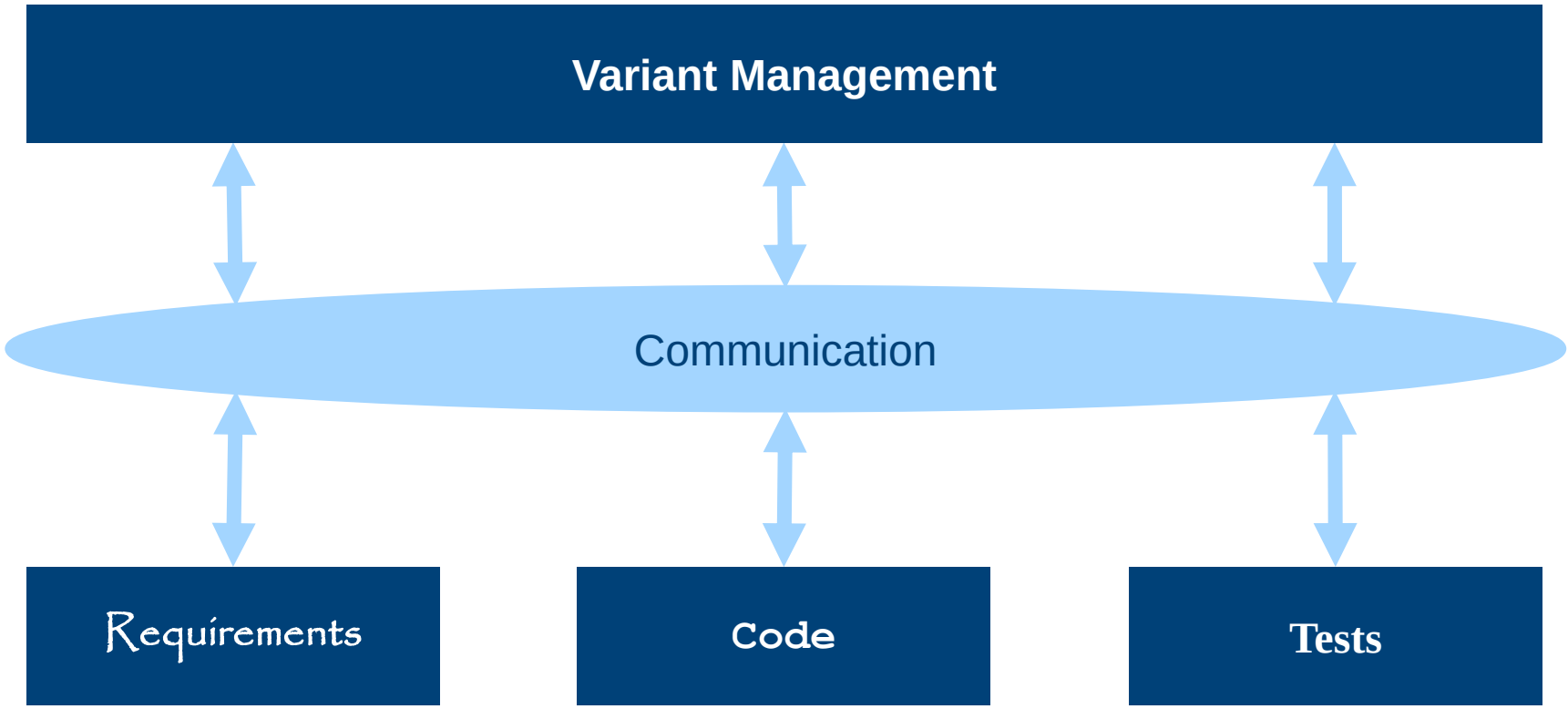
Adding More Artifacts



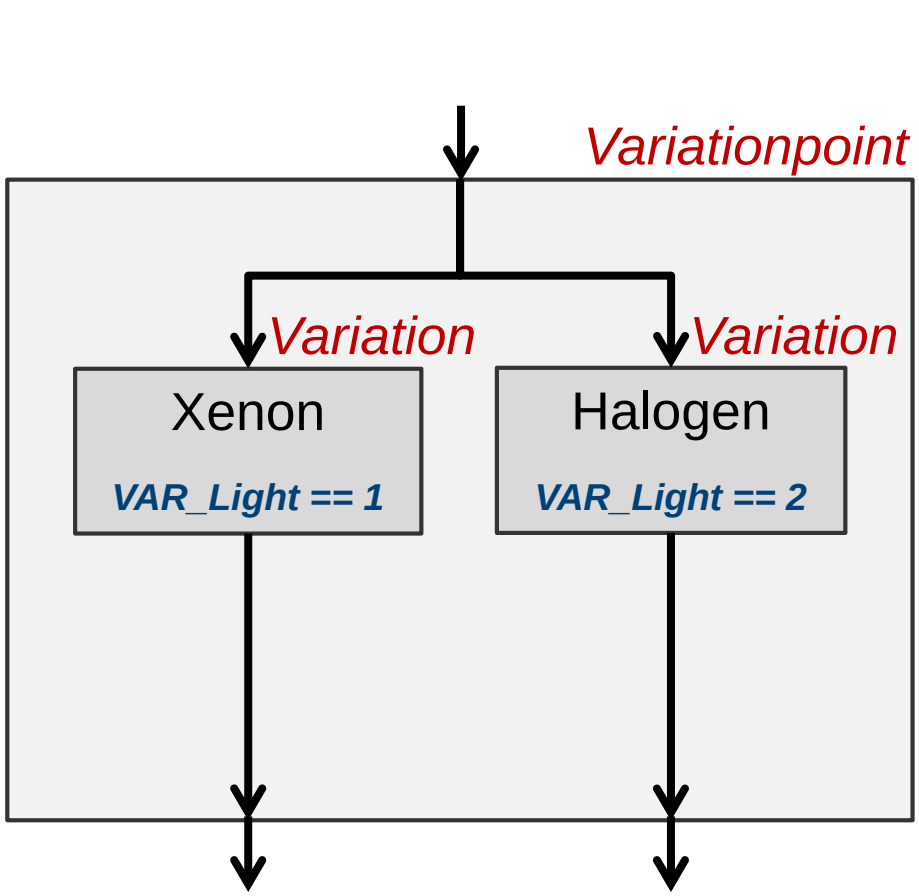
From a project 3000 years ago…



How Things Should Be



Example: 150% Model



Configurationmodel

Featuremodel

Configuration

Variability Exchange Language

Standardizes data structures and their representations

- Artifacts
  - Variation points (names)
  - Variations (names, conditions)
- Binding times
- Expressions
- Link to artifacts and elements in artifacts
- Variants

Defines an API

- Import variation points
- Import variants
- Export variants
- Export variation points

... More Requirements

Keep It Simple

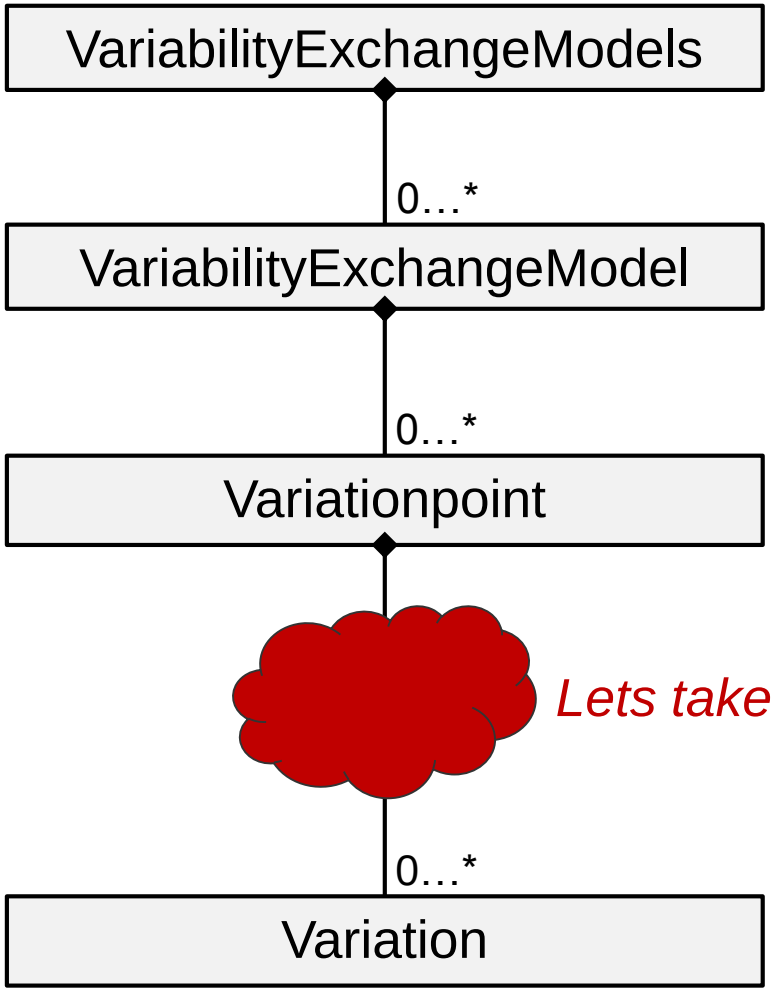
Fits our use cases

Make it easy to implement

Add more complexity in a later step

No assumptions on how variability is implemented

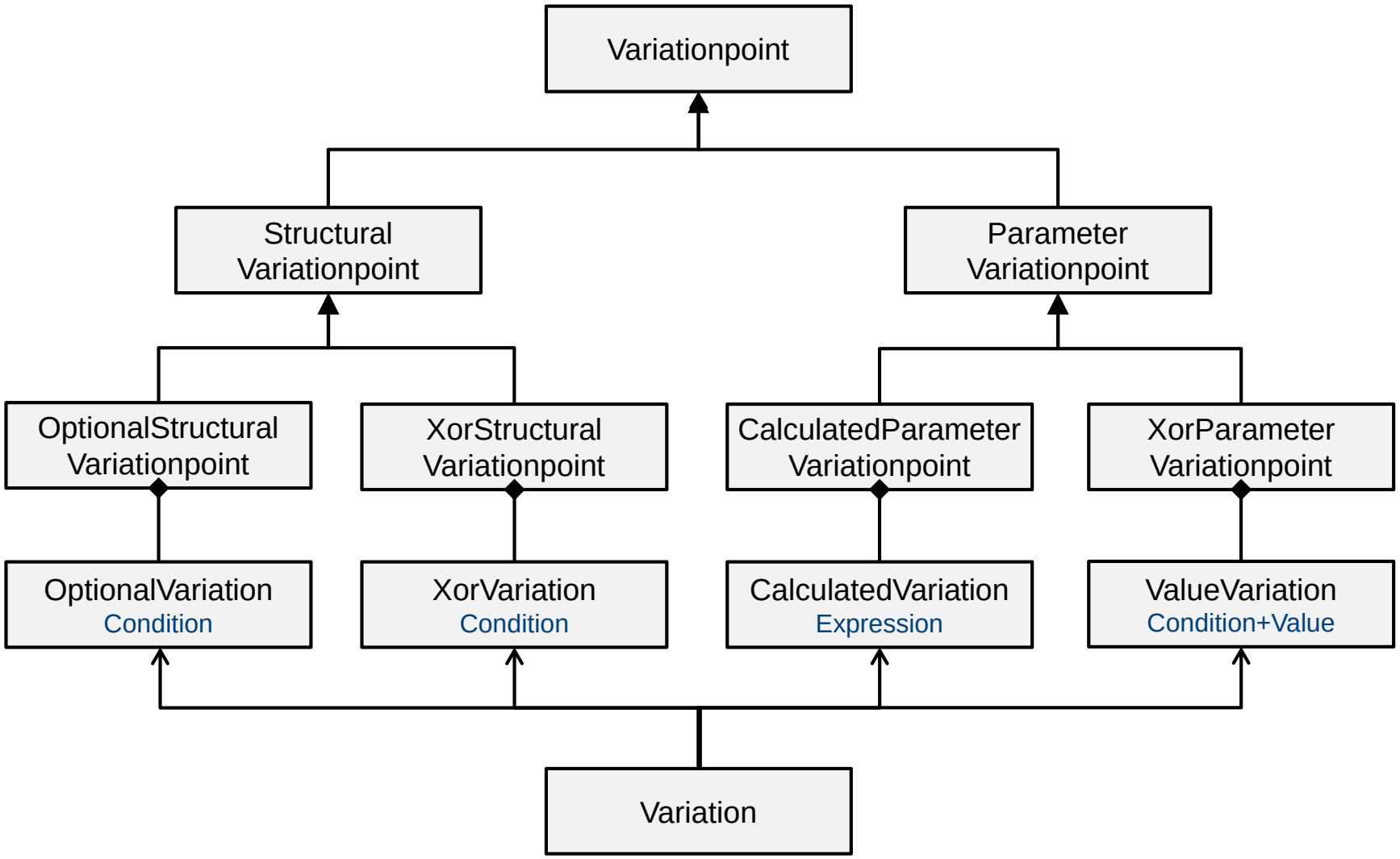
Overall Structure



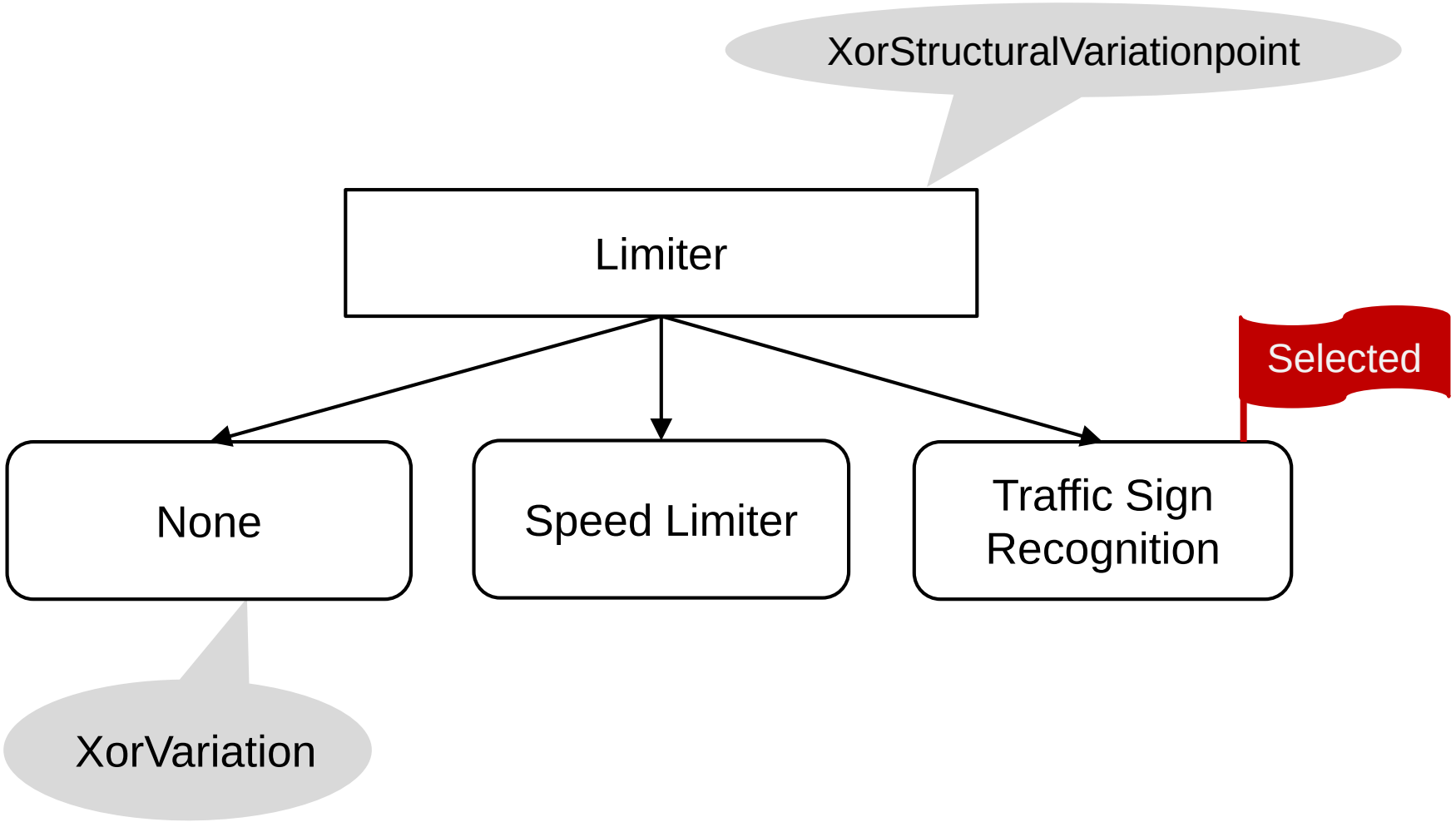
DOORS, Simulink, C-Code, Tests, ...

Lets take a closer look at this

The Variationpoint Family



Example for Variationpoints



Variationpoint Configuration In XML

```
<variability-exchange-model type="variationpoint-description" id="vem1">

  <xor-structural-variationpoint id="variationpoint1" >

    <variation id="variation1">
      <single-feature-condition>
        <feature><name>NoLimiter</name></feature>
      </single-feature-condition>
    </variation>

    <variation id="variation2">
      <single-feature-condition>
        <feature><name>SpeedLimiter</name></feature>
      </single-feature-condition>
    </variation>

    <variation id="variation3">
      <single-feature-condition>
        <feature><name>TrafficSignRecognition</name></feature>
      </single-feature-condition>
    </variation>

  </xor-structural-variationpoint>

</variability-exchange-model>
```

Variationpoint Description In XML

```
<variability-exchange-model type="variationpoint-configuration" id="vcm1">

  <xor-structural-variationpoint id="variationpoint1" >

    <variation id="variation1" selected="false">
      <single-feature-condition>
        <feature><name>NoLimiter</name></feature>
      </single-feature-condition>
    </variation>

    <variation id="variation2" selected="false">
      <single-feature-condition>
        <feature><name>SpeedLimiter</name></feature>
      </single-feature-condition>
    </variation>

    <variation id="variation3" selected="true">
      <single-feature-condition>
        <feature><name>TrafficSignRecognition</name></feature>
      </single-feature-condition>
    </variation>

  </xor-structural-variationpoint>

</variability-exchange-model>
```

More about Variationpoints

Variations may have dependencies on other variations

- “Require”
- “Conflicts”

Variationpoints may be hierarchical

- Inspiration: subsystems in Simulink, structured requirements

Variationpoints hay have binding times

- Multiple binding times per variation point, actually
- Need to select one before the binding starts

Variationpoints and Variations know their artifact elements

- Implemented in XML by using `<xs:any>`

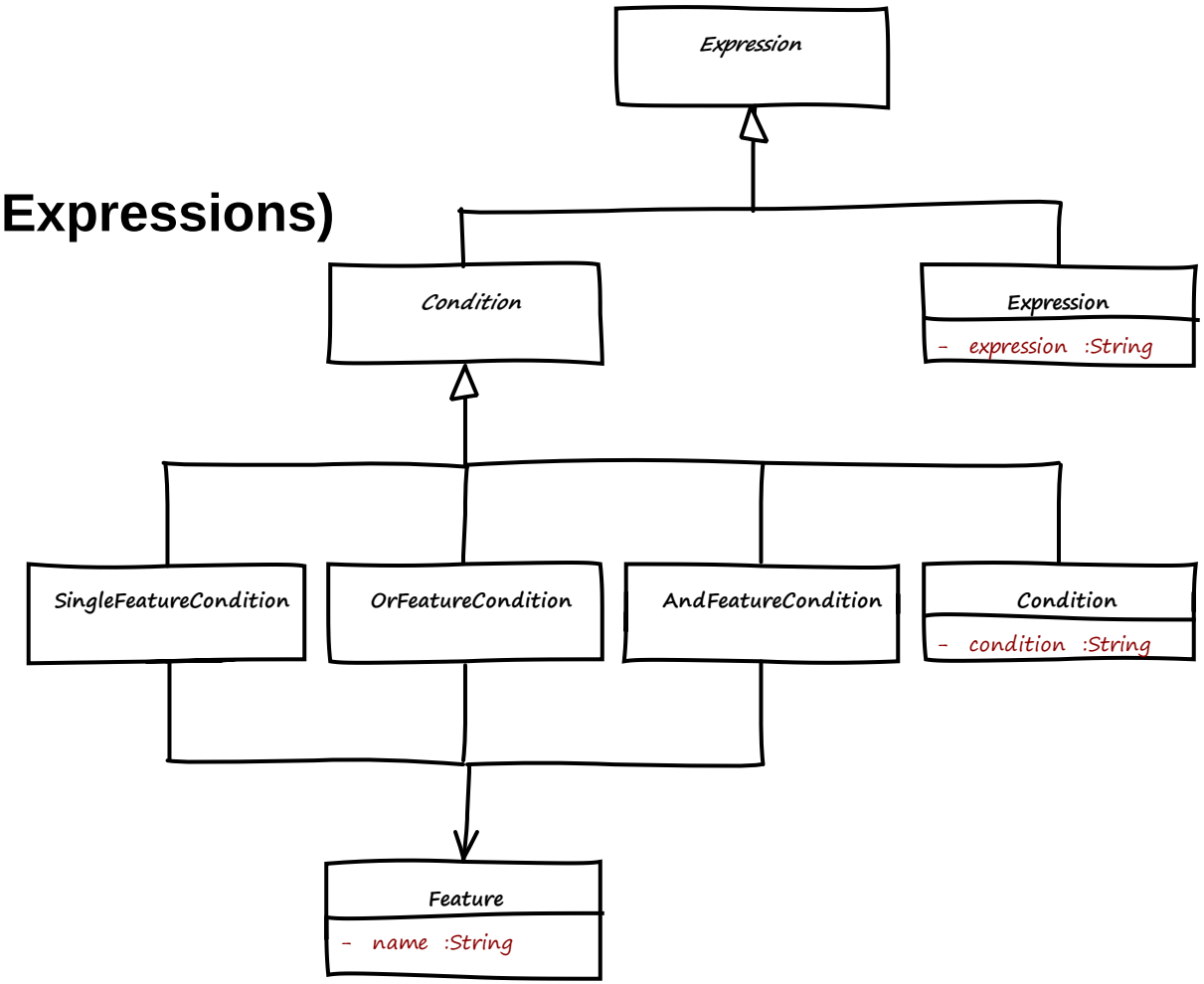
Expressions and Conditions

Expressions

- Complex

Conditions (Boolean Expressions)

- Feature
- Feature $\wedge$ Feature
- Feature $\vee$ Feature
- Complex



The API

Four basic API methods

- Import Variationpoints
- Import Variants
- Export Variants
  - may add or delete variants
- Export Variationpoints
  - may add or delete variationpoints/variations

Not every implementation needs to support the full API. There is an attribute *capability* that indicates which API calls are supported.

Schedule

Specification of the Variability Exchange Language

- UML
- XML Schema
- Textual specification

Prototype Implementation

- Will be done as part of the SPES project

Will be publicly available as SPES Deliverable in Mid-2015

Conclusion


<variationpoint>
<variationpoint>
<variationpoint>
<variationpoint>
<variationpoint>
<variationpoint>
<variationpoint>