Fraunhofer FOKUS

Model-Based Testing of Software Product Lines

Top-Down and Bottom-Up Approach

Dipl.-Inf. Hartmut Lackner, 17. März 2013, Rome, 8th MBT Workshop

Contents

- **Software Product Lines**
  - Product Line Design
  - Variability in Behavior Models
  - Feature Mapping
- **Model-Based Testing**
  - Models for Testing
  - Automation of Test Design
- **Model-Based Testing for Product Lines**
  - Top-Down vs Bottom-Up
  - Comparison
- **Summary**

What is a Product Line?

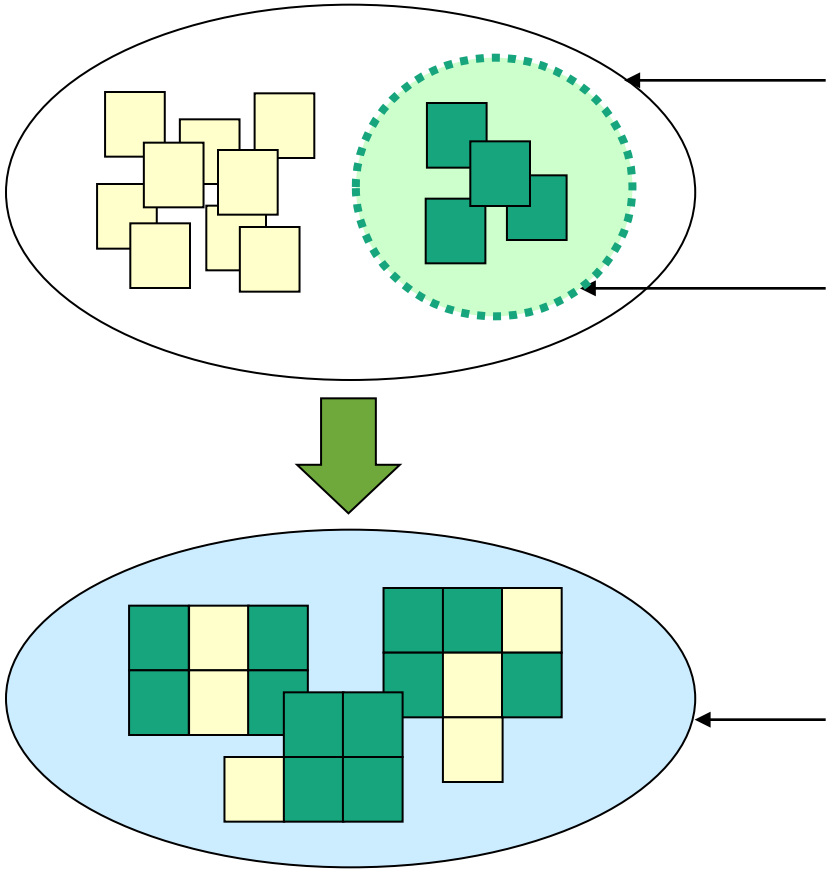
Software Product Lines

- Analogue to product lines from end-user products:
  - similar use,
  - similar features („look-and-feel“),
  - different pricing
- Most important for business domains where there is a necessity to offer similar but different products
- Explicit modeling of commonalities and differences within the products of a line

CMU SEI

„A software product line (SPL) is a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way.“

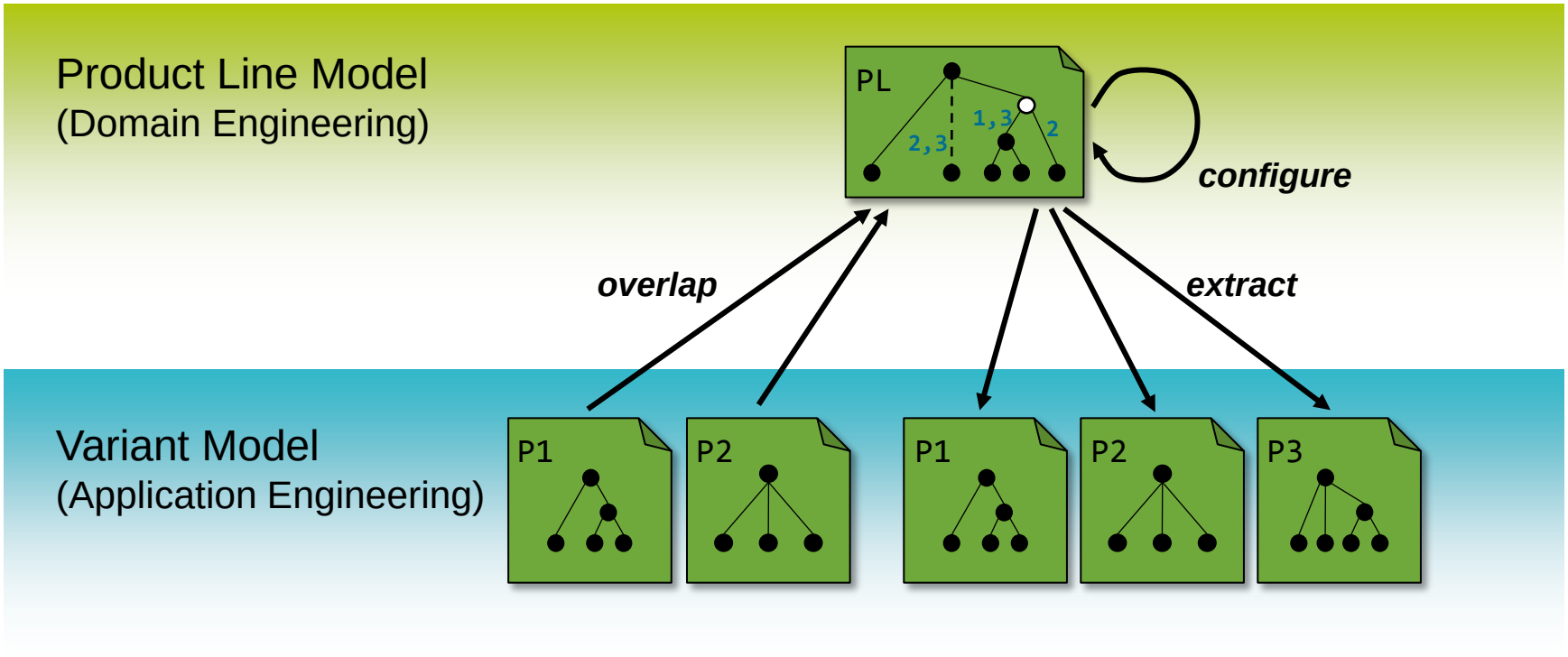
Building Blocks, Platform, Product Family



- **»Product Blocks«**
- **Product Platform:**
  Commonalities of a product line
- **Product Family:**
  Products that are based on the same platform.

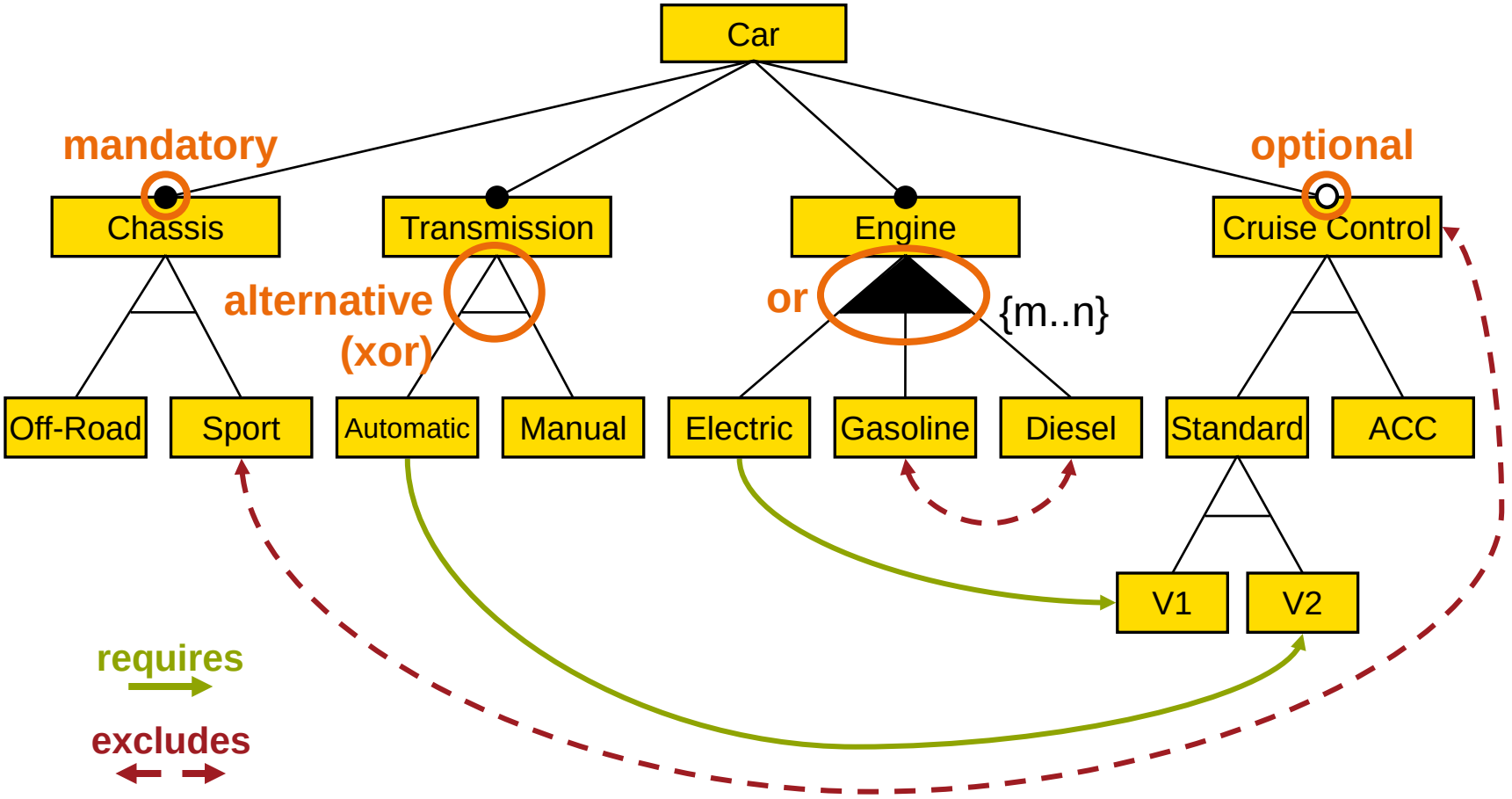
Quelle: Blackenfelt 2001

Product Line Model vs. Variant Model



Example of a »simple« Product Line Model

Variants of a Vehicle

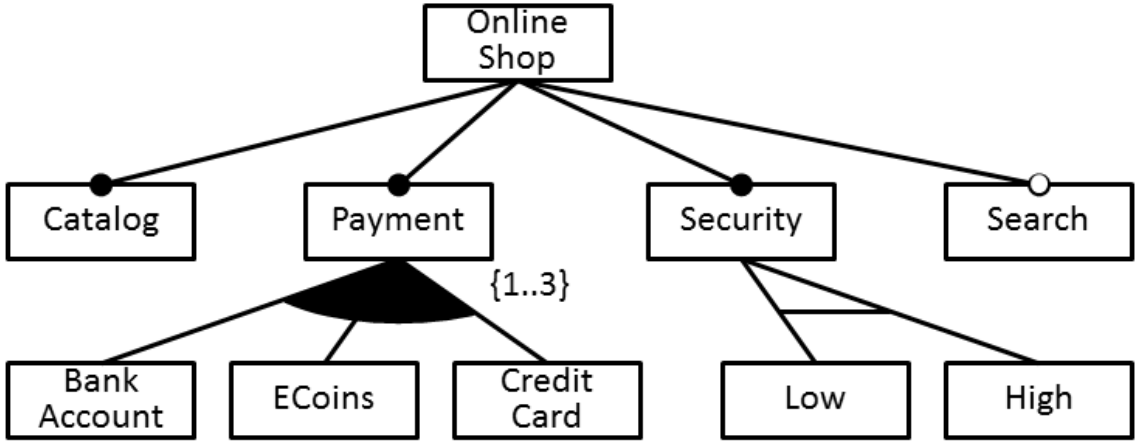


Fraunhofer FOKUS

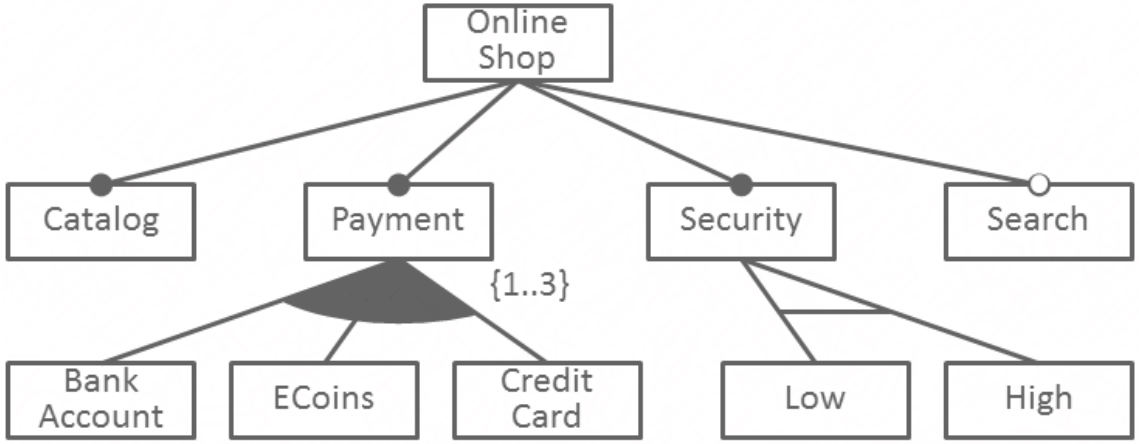
Case Study Example

A Toy Example

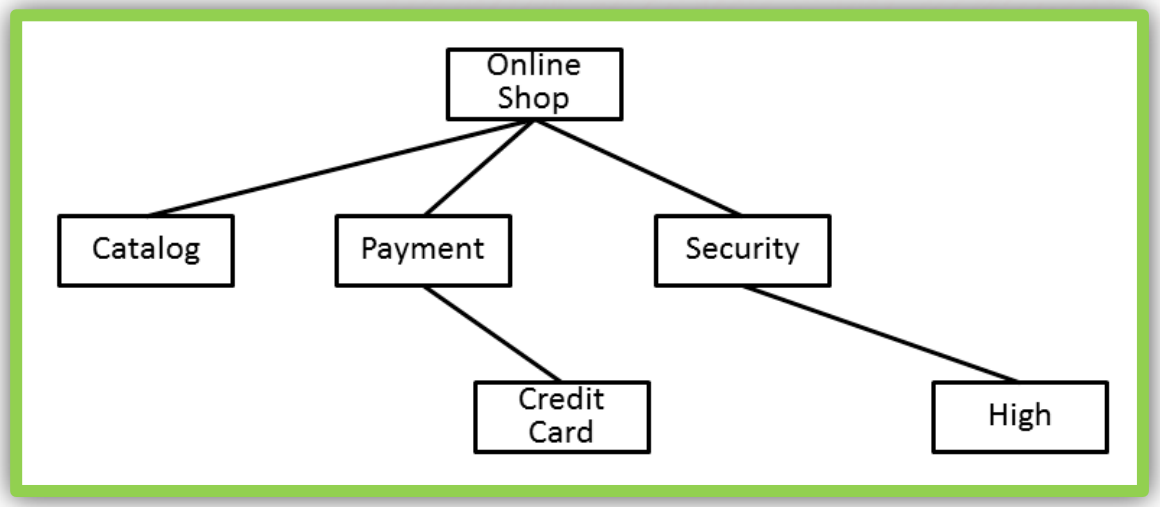
An even simpler Example: Online Shop Family



A valid Variant

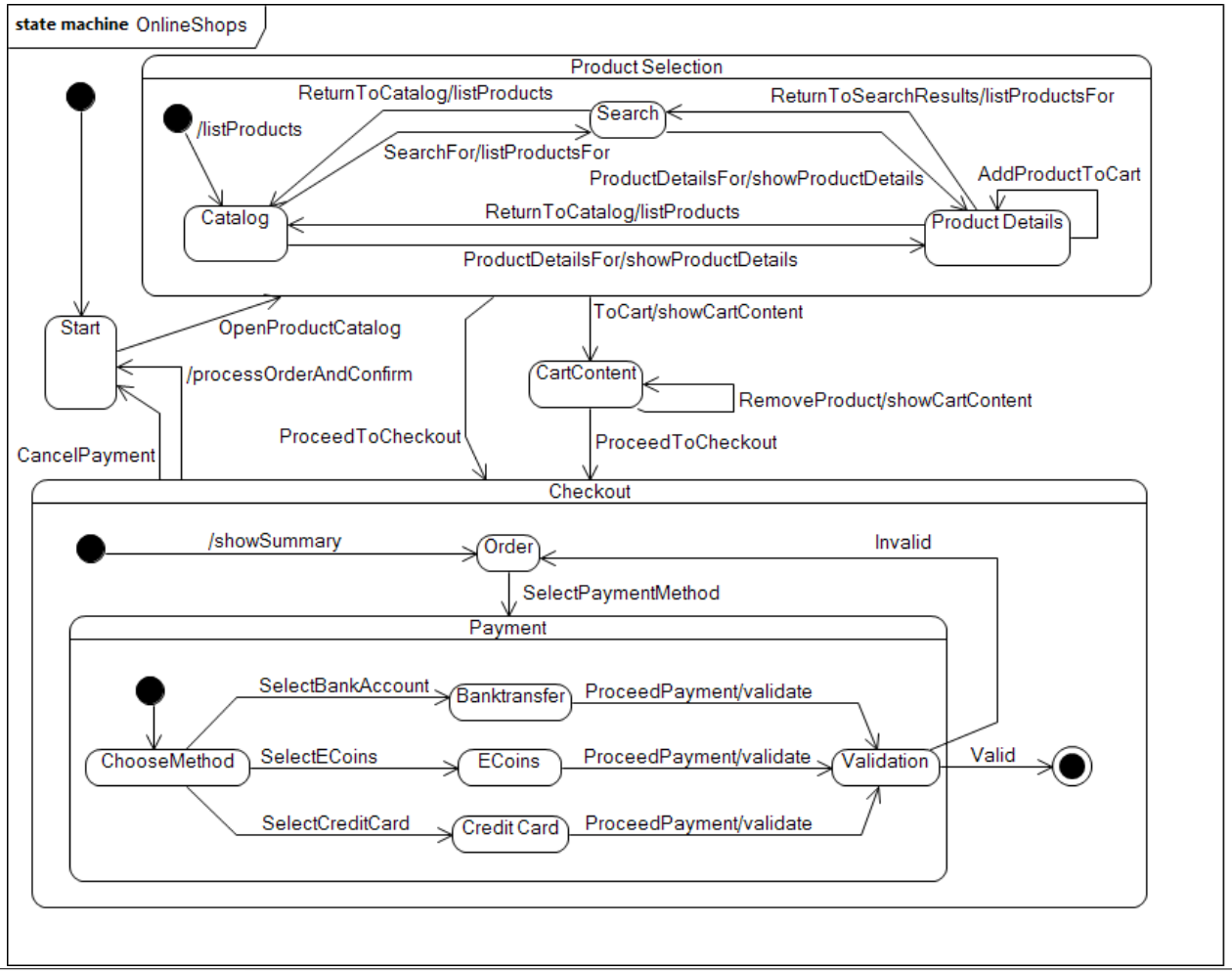


Credit Card implies High



Behavior Model of the Product Family

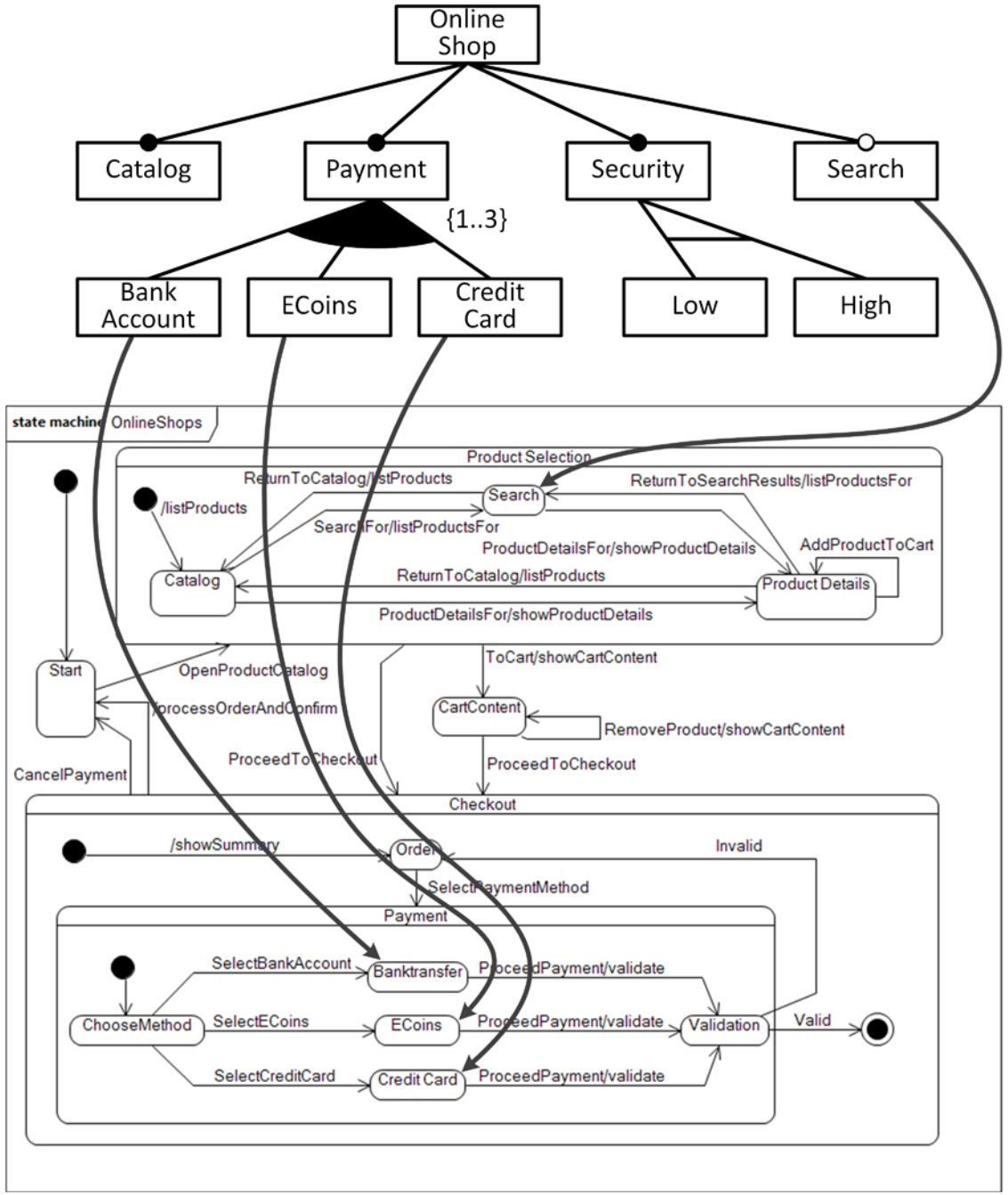
150% Model



Feature Mapping

Maps features to model elements

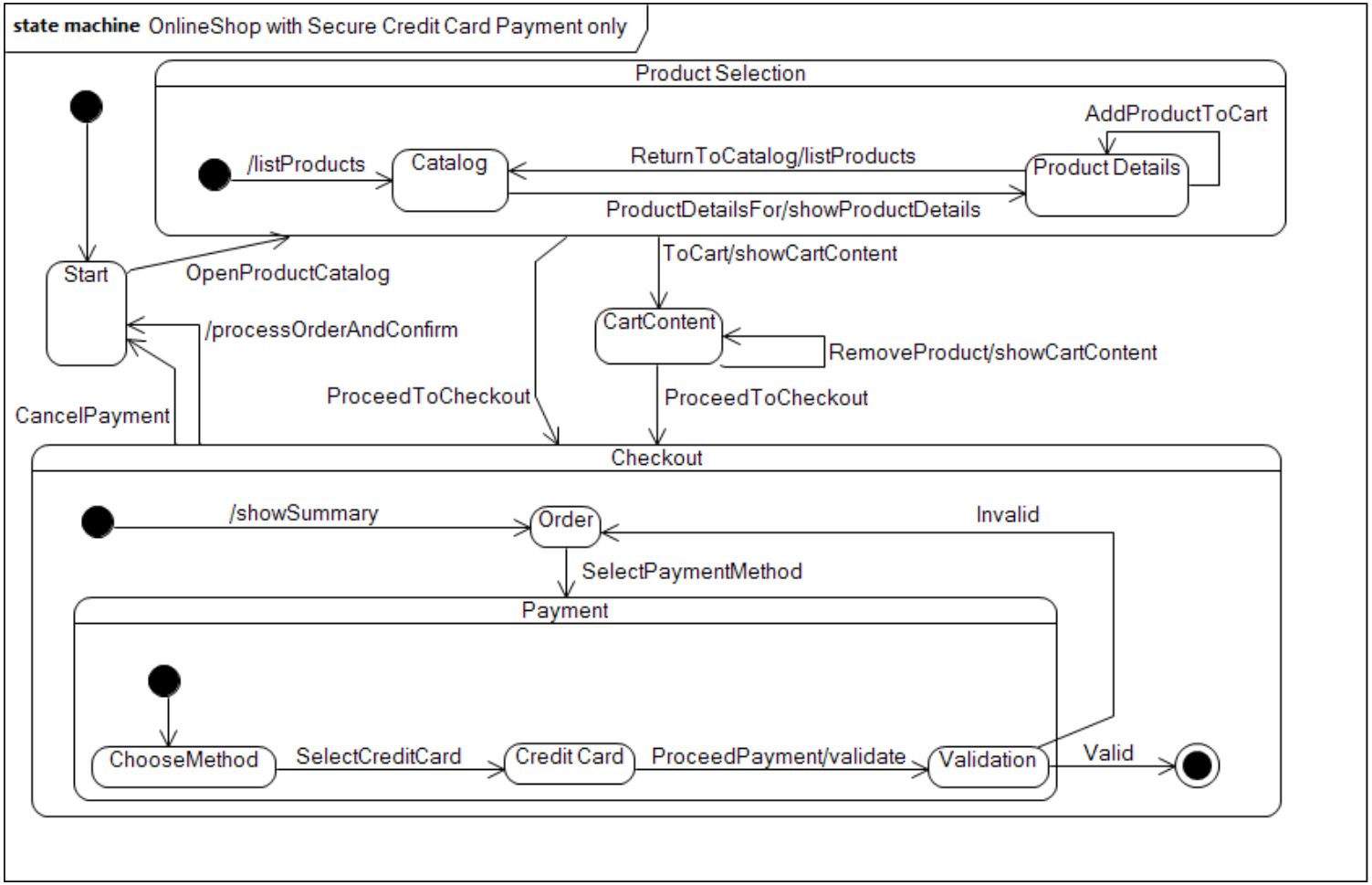
If a feature is selected for a variant the corresponding elements in the state machine are present



Fraunhofer FOKUS

A Product‘s Behavior

100% Model

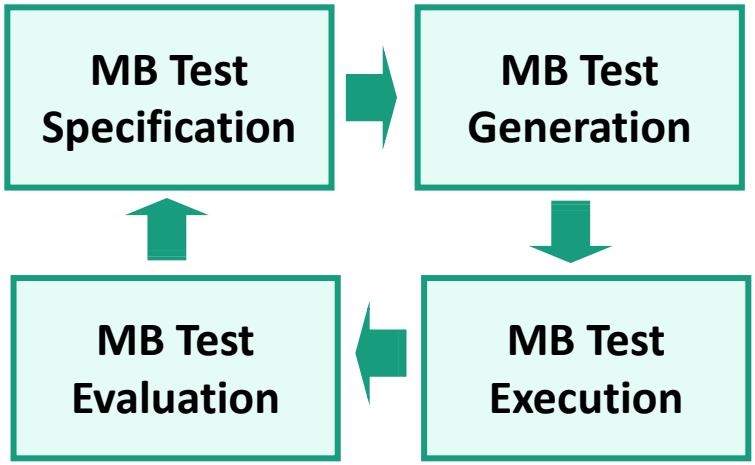


Fraunhofer FOKUS

Model-Based Testing

Model-Based Testing

- Models can be used in several ways
  - Models for **Test Specification**, i.e., Formalization of Requirements
  - Models as the source of **Test Generation**
  - Models as **Test Objects** in model-based development
  - Models as **Test Oracles**, i.e. for evaluation of tests



Tools for Model-Based Testing

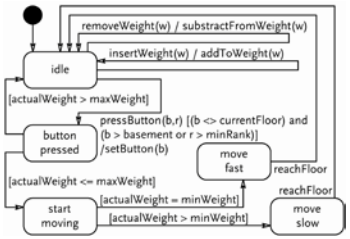
- **Test specification**
  Model creation (Editor and Syntax checker)
  Linking to Requirements-Management-Tools

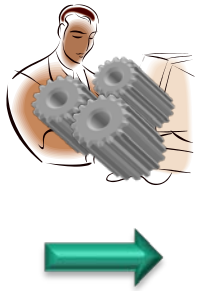
- **Test generation**
  Automatic derivation of tests from models
  Selection of test data

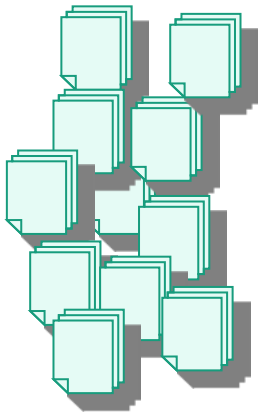
- **Test execution**
  Connection to the target
  Test monitoring and management

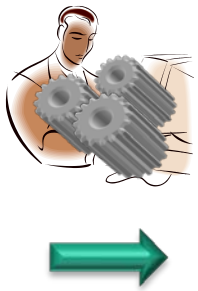
- **Test evaluation**
  coverage checker, report generators,
  verification, model checker

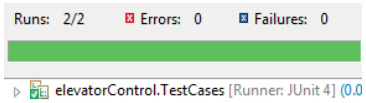
Model-Based Testing: Automation of Test Design





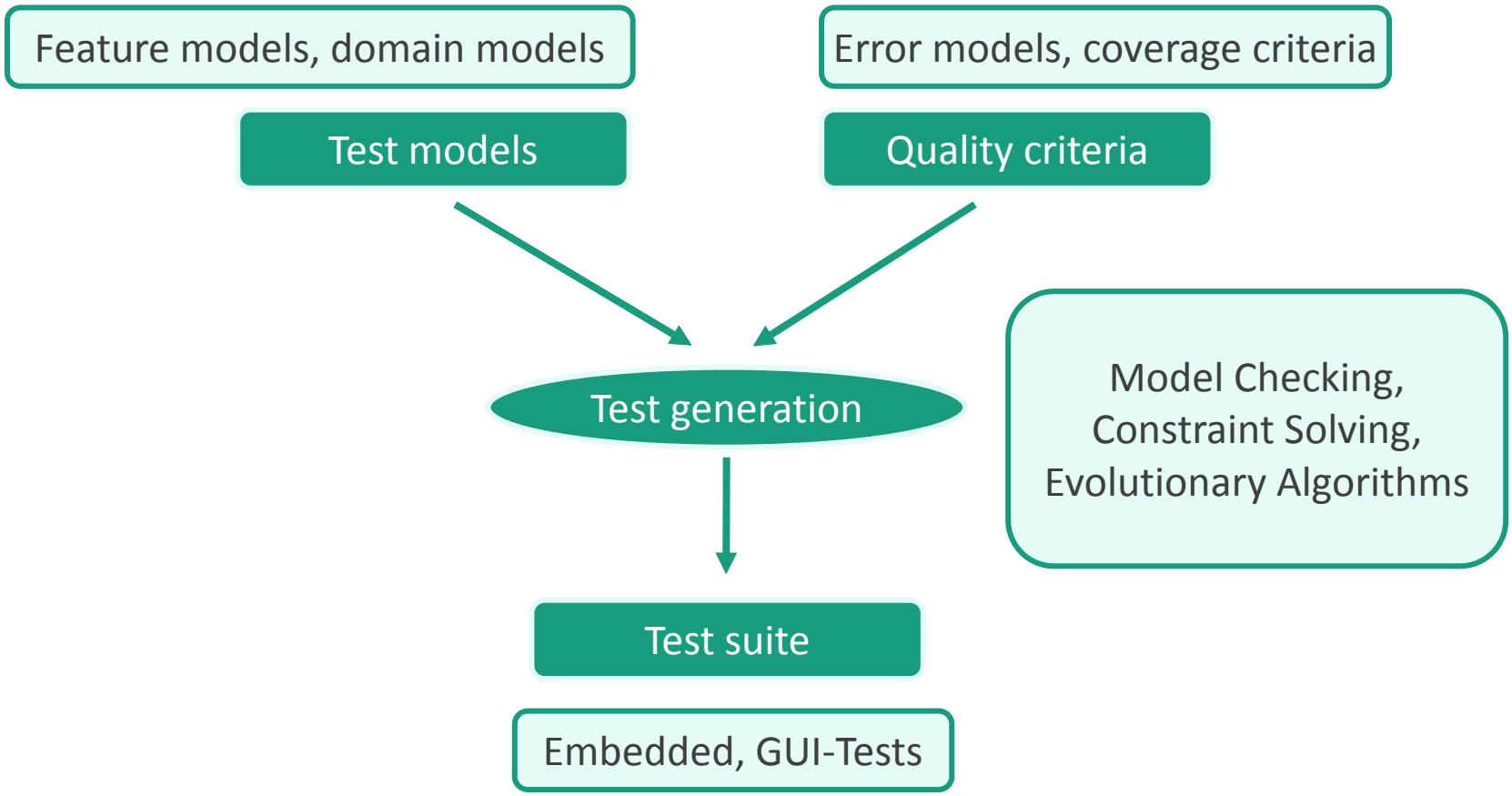




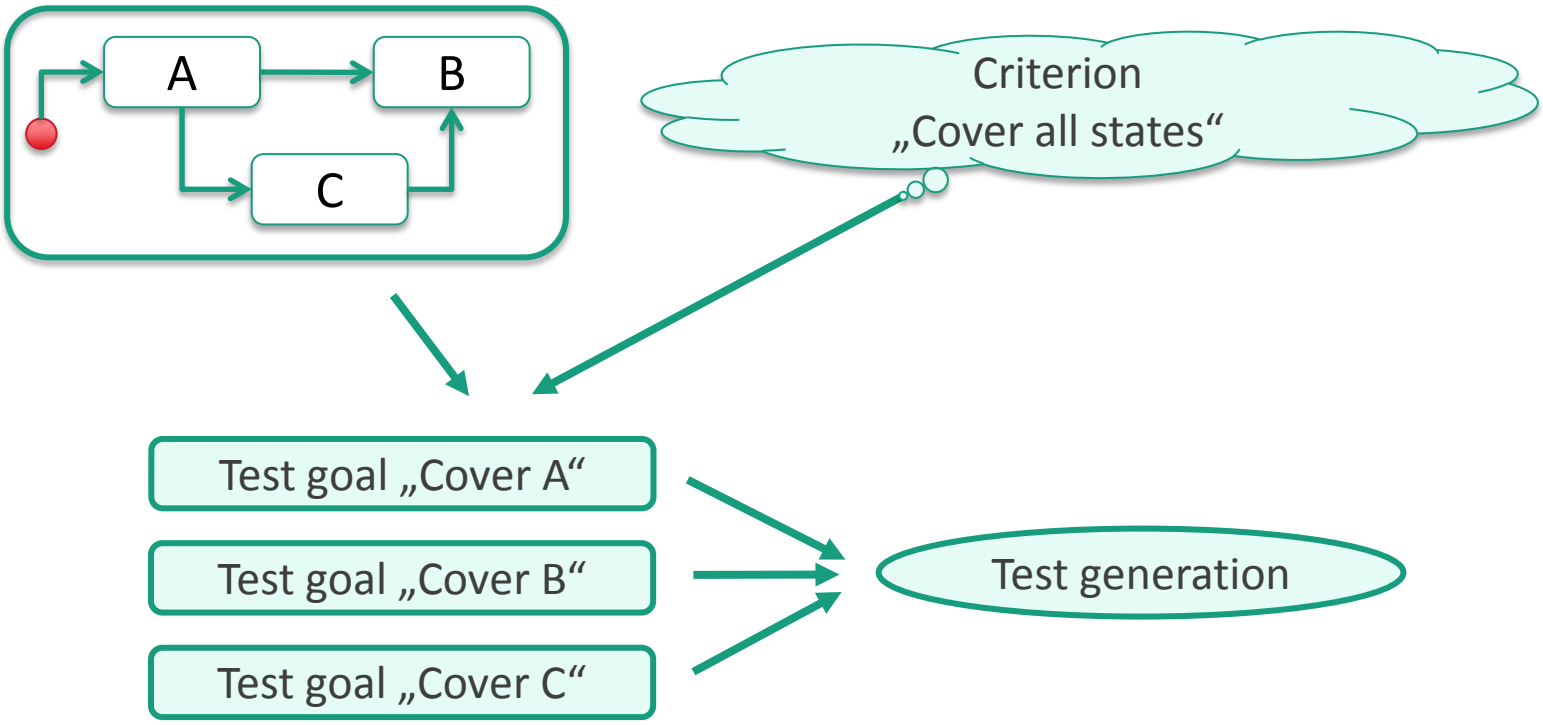


Model **Tests** **Test execution**

Model-Based Testing: Automation of Test Design



Automatic Test Generation with Test Goals



Fraunhofer FOKUS

Model-Based Testing of Product Lines

General Approaches to MBT for SPL

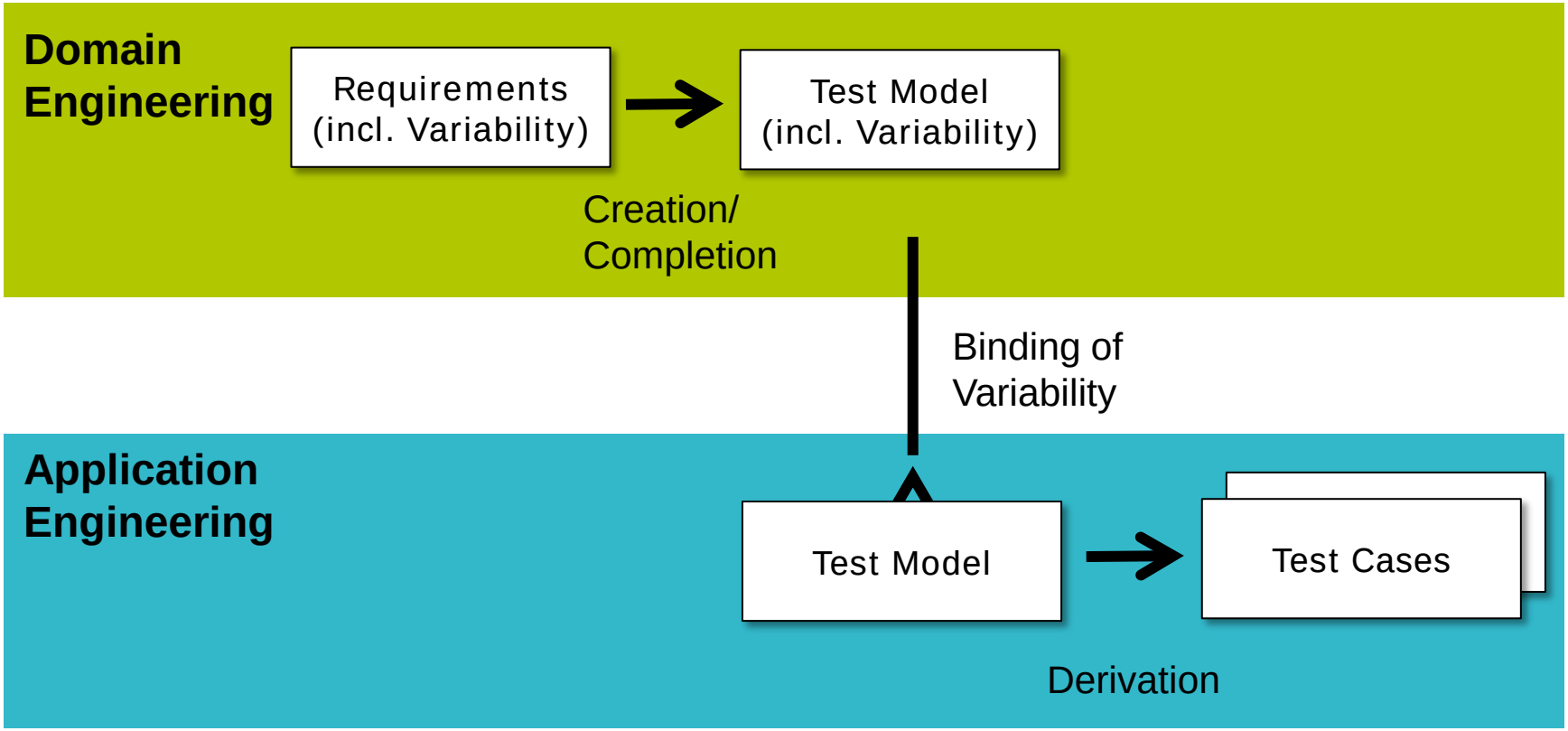
When applying MBT to SPLs there is one fundamental choice:

When to bind variability?

1. Before test case design
2. After test case design

Binding Variability BEFORE Test Case Design

Product-Centric



Top-Down

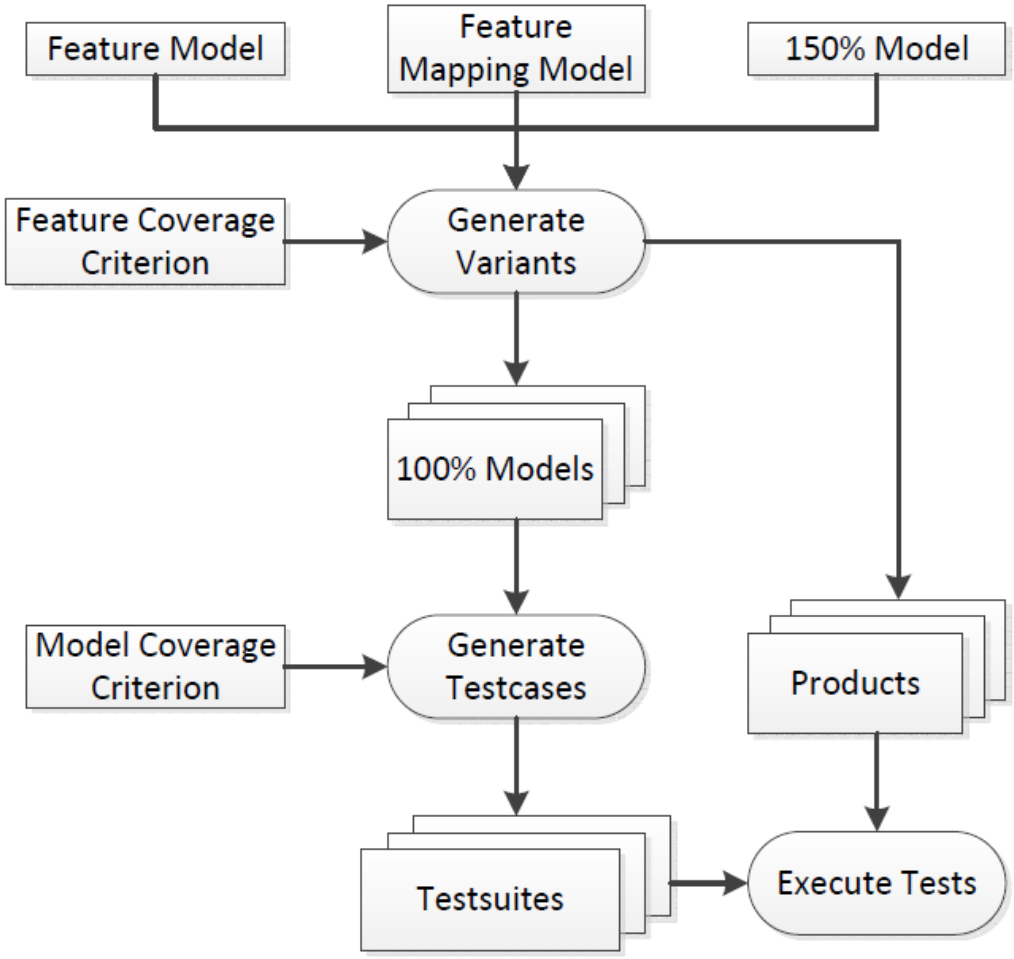
Product-Centric

Variants (100% Models) and Products are generated according to „Feature Coverage Criteria“:

- All-Features-Selected/Unselected
- Pair-/N-wise
- …

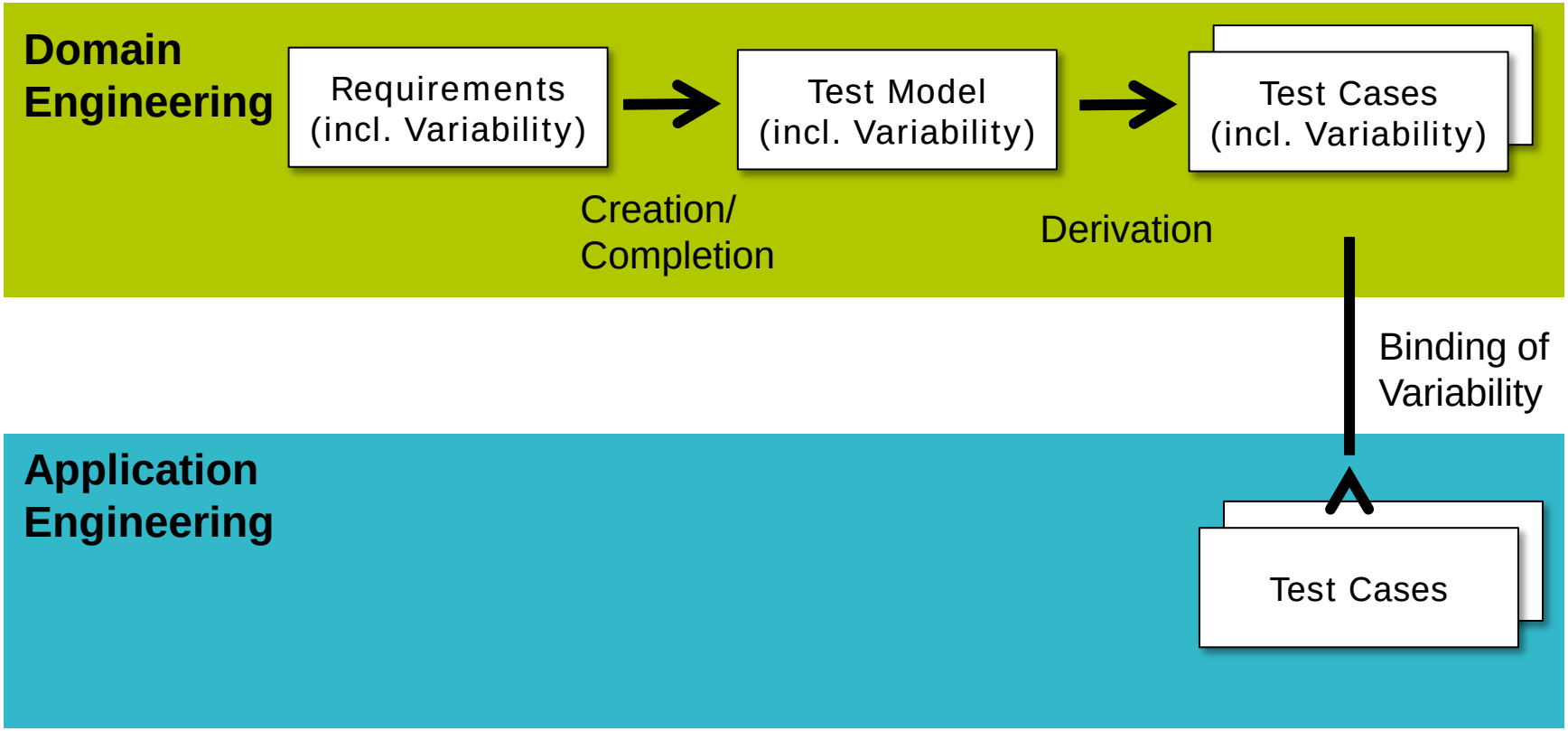
Test cases are generated for every variant/product according to test goals:

- Transition Coverage
- MCDC
- …



Binding Variability AFTER Test Case Generation

Domain-Centric



Bottom-Up

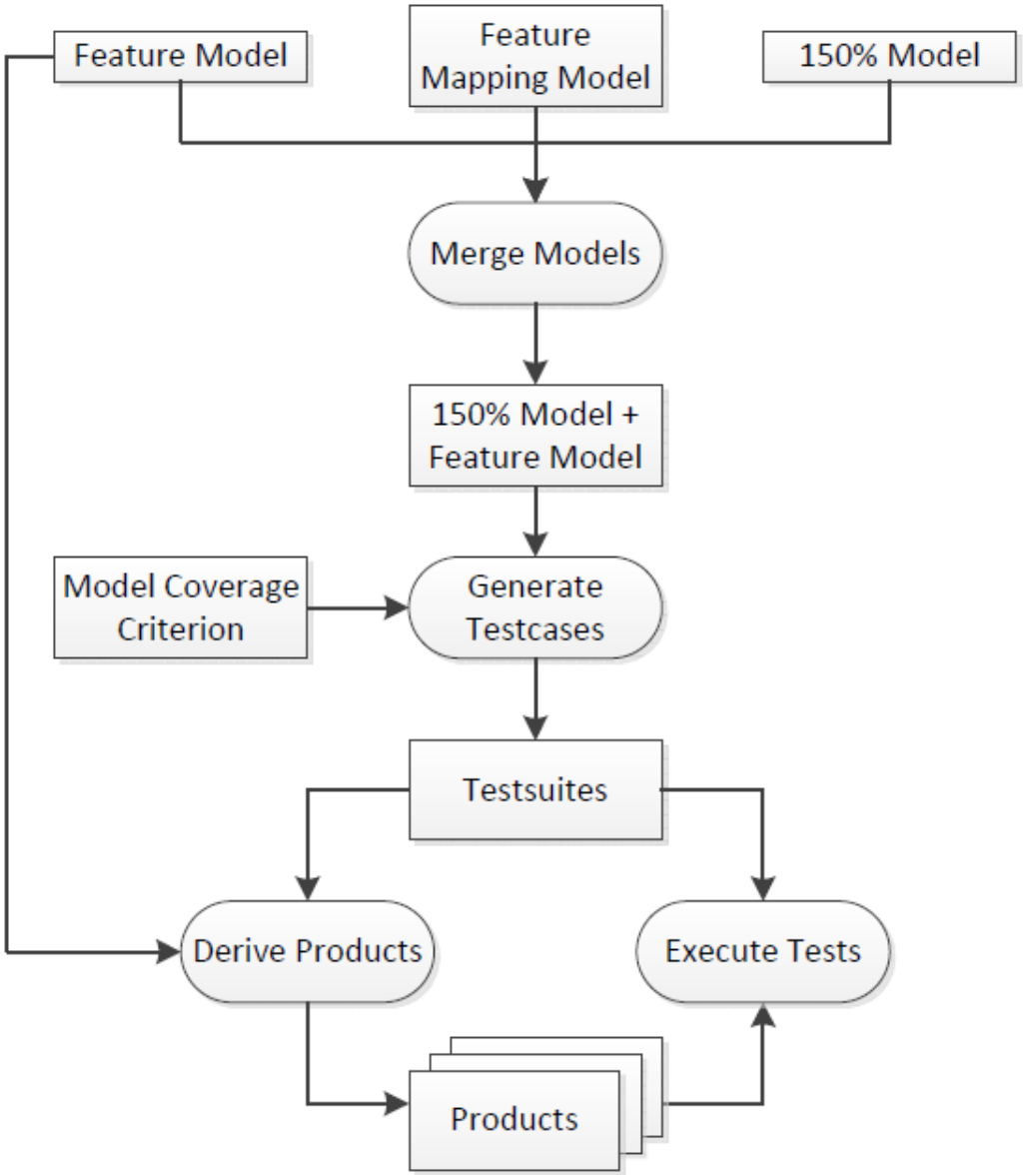
Domain-Centric

The 150% model is merged with the feature model.

Test cases are generated from the merged model. A single test case may be applicable to more than one product.

Products for testing have to be selected from the test suite according to a criterion:

- Minimal number of variants
- All variants
- …



Comparison

Selected Coverage Criteria

- Top-Down
  - Feature Model: All-Features-Selected and –Unselected;
  - State Machine: Transition Coverage
- Bottom-Up
  - State Machine: Transition Coverage
  - Variant Coverage: minimum number of Variants

	Top-Down		Bottom-Up	
	Manual	Automatic	Manual	Automatic
Variants	2	2	1	2
Test Cases	2	18	1	12
Test Steps	43	70	27	59

Automatic test design by Conformiq Designer

Evaluation

- Redundancy-wise, the bottom-up approach seems to be more efficient

→ **Variant selection and test generation heavily depend on the applied coverage criteria**

- Weaker coverage criteria for variant selection can lead to more efficient results for Top-Down

→ **Importance of a single variant for the behavior is not easy to determine.**

Fraunhofer FOKUS

Summary

Summary

- Theoretical considerations for efficiently testing software product lines
  - Top-Down approach (Product-Centric)
  - Bottom-Up approach (Domain-Centric)

Future Work

- Larger Examples
- Complete the tool chain
  - Retrieve a minimal number of variants from the 150% model based test cases
  - …
- More experiments on the pros and cons of each approach
  - Is it advisable to apply strong coverage criteria on feature models or on 100% models?
  - …