
Technische Universität Braunschweig

Institut für Programmierung und Reaktive Systeme

# eDeltaMBT

## Delta-oriented Model-based Software Product Line Testing

**Sascha Lity, Malte Lochau, Ina Schaefer**

Eclipse Testing Day 2012, 05.09.2012, Darmstadt

# Outline

# Software Product Lines (1/2)

1980's

# Software Product Lines (1/2)

1980's 

2012 

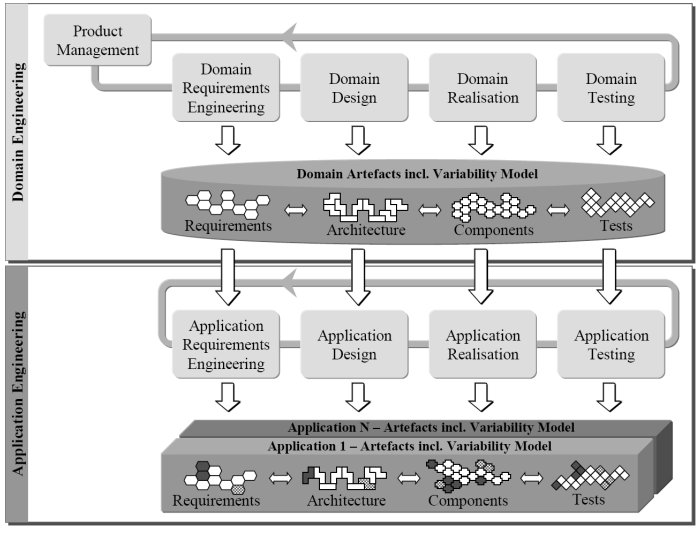
- Mass customization [Dav87, Car11] of complex (software) systems
- **Software Product Lines**

  *[...] explicit specification of commonality and variability between variants in a family of similar [software] products by means of features [PBL05].*

# Software Product Lines (2/2)

## SPL Engineering [PBL05]

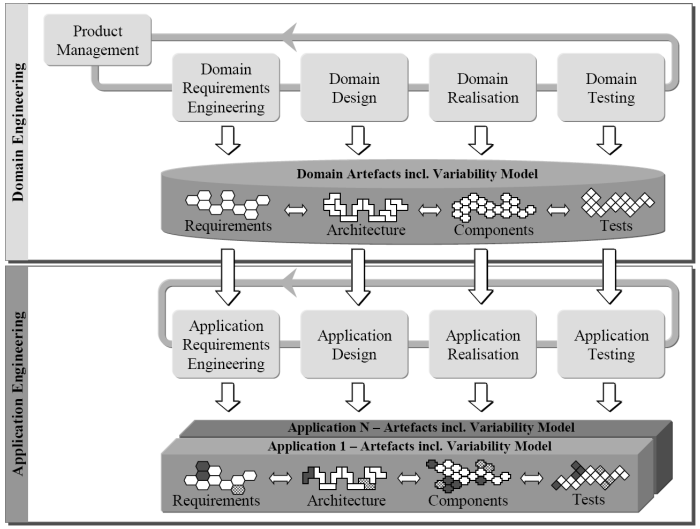
- Domain engineering: design for reuse
- Application engineering: design with reuse



# Software Product Lines (2/2)

## SPL Engineering [PBL05]

- Domain engineering: design for reuse
- Application engineering: design with reuse




## SPL Philosophy

- Features denote explicit product configuration parameters
- Common core platform to derive product variants
- Systematic reuse of engineering artifacts among product variants

# Testing Software Product Lines

Product-by-product testing is infeasible

- High number of potential product variants
- Limited resources
- Redundancies due to commonality/similarity

# Testing Software Product Lines

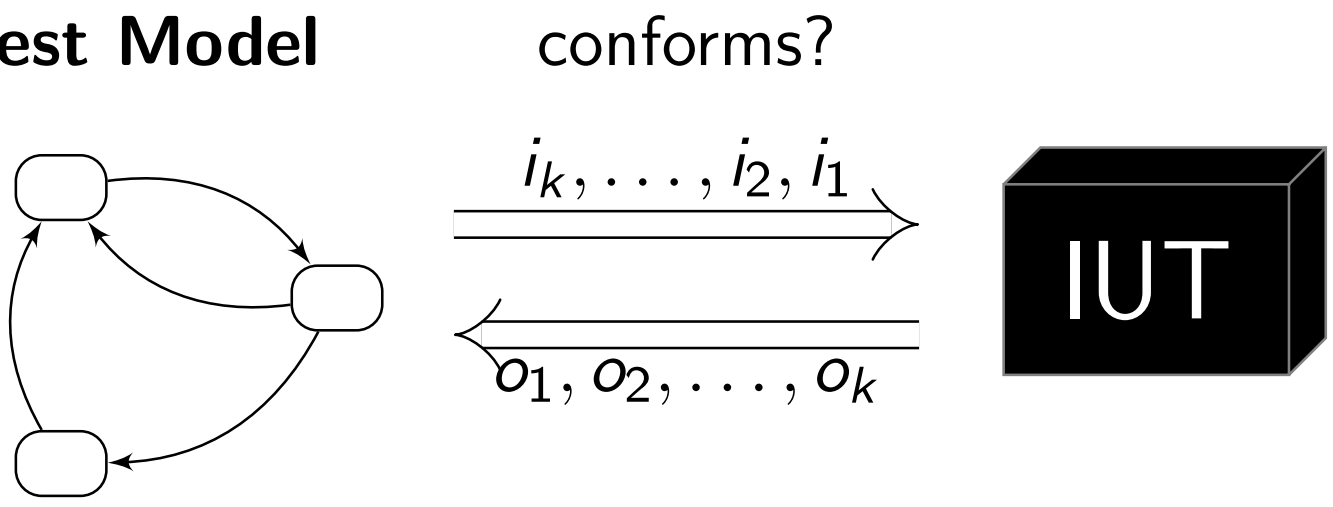
Product-by-product testing is infeasible

- High number of potential product variants
- Limited resources
- Redundancies due to commonality/similarity

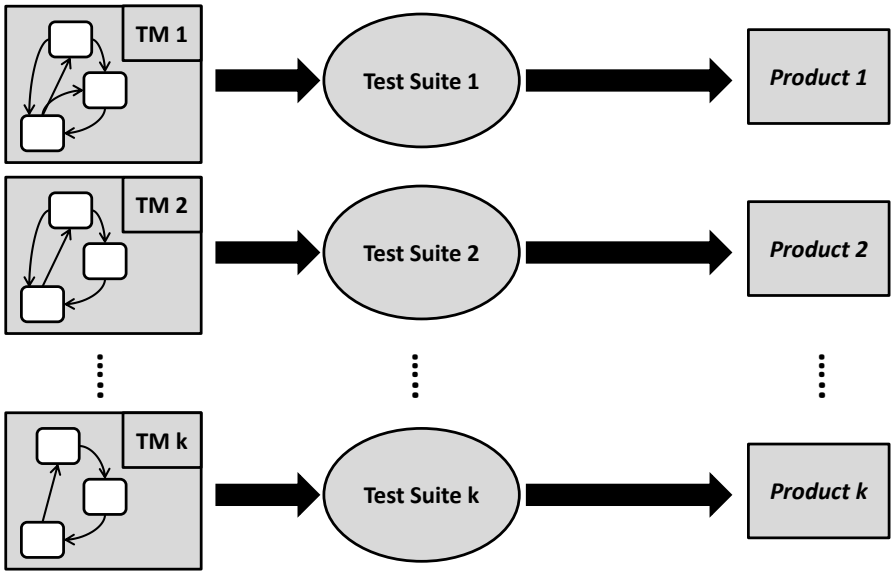
*Model-based testing is well-suited for testing software product lines [cf. Olimpiew, 2008]*

# Model-based Testing

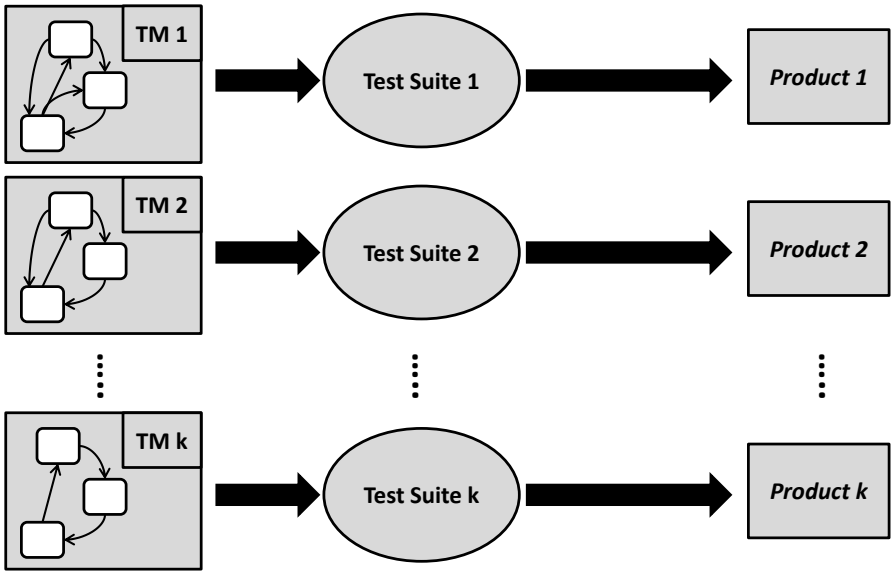
*Model-based testing is the automation of the design of black-box tests [UL06].*



# Model-based Software Product Line Testing



# Model-based Software Product Line Testing




- Regard commonality and variability
- Adapt reuse principles to SPL testing (?)
  - Reusable test model
  - Reusable test cases
  - Reusable test results

# Delta-oriented Model-based SPL Testing

## Delta-oriented Test Modeling

- Adaption of delta modeling [Sch10] to state machines
- Definition of reusable test models

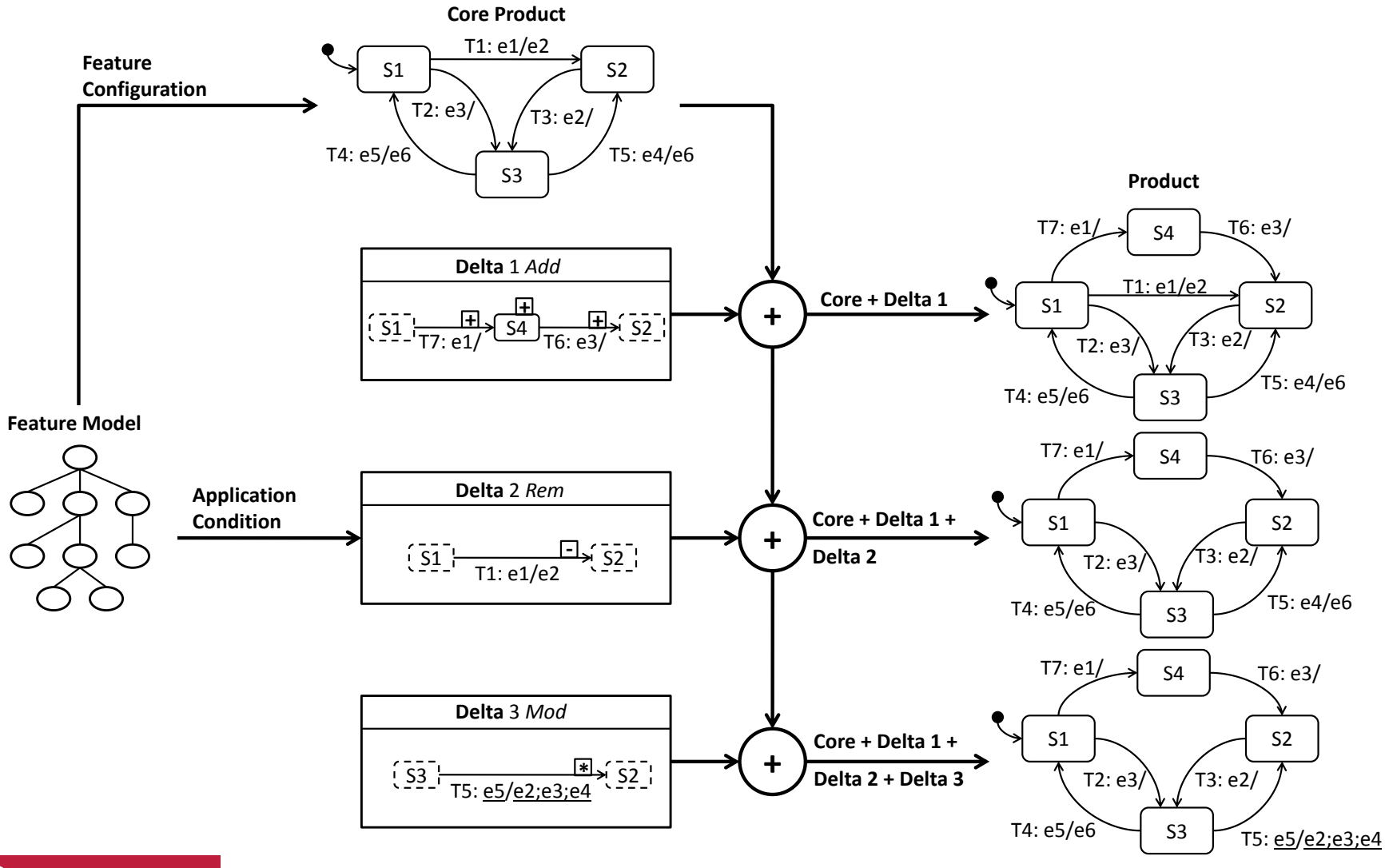
## Delta-oriented Test Artifact Evolution

- Adaption of principles of regression testing
- Incremental evolution based on changes of test models
- Reuse of test cases and test results

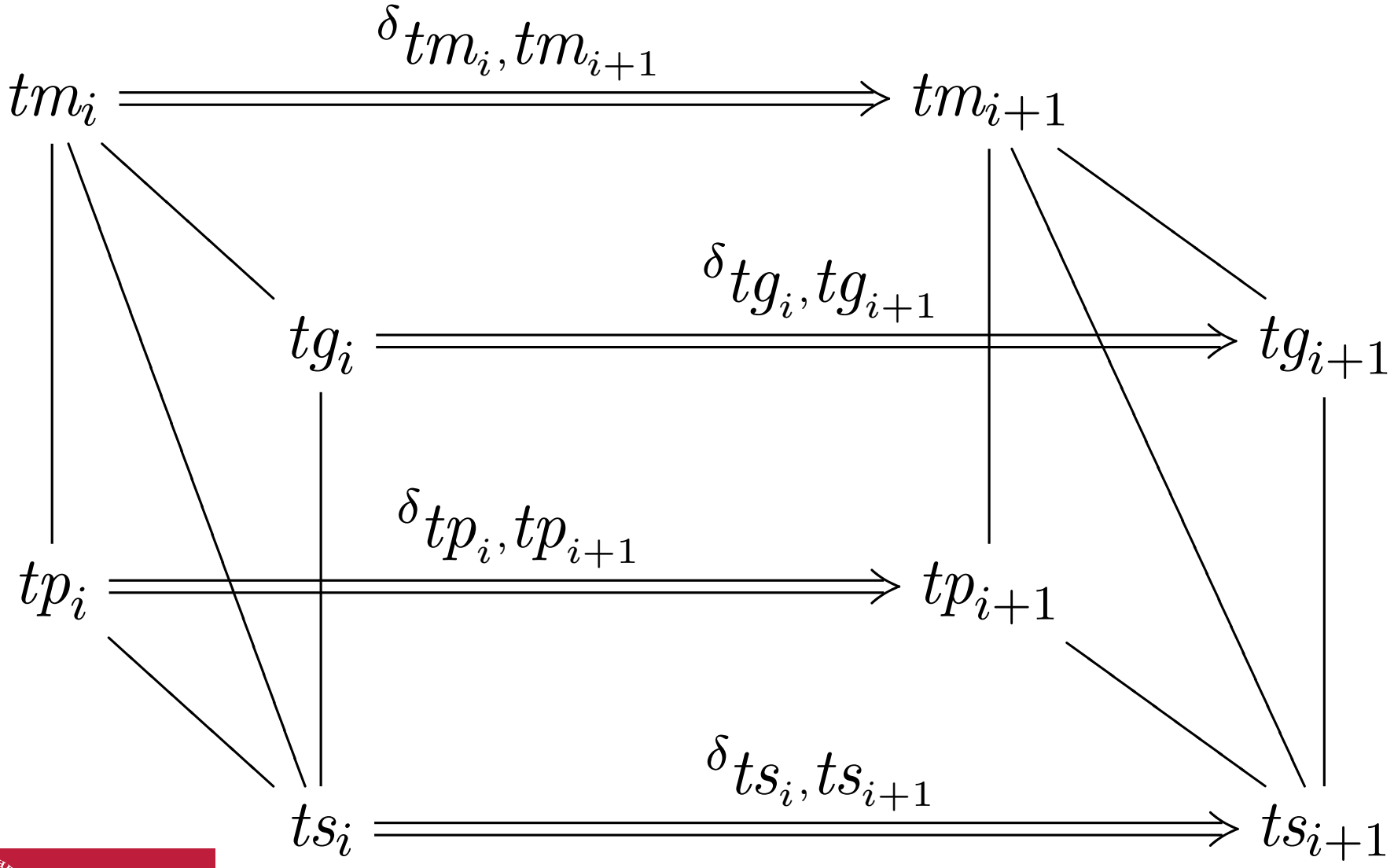
[Lochau et al.: Incremental Model-based Testing of Delta-oriented Software Product Lines, TAP 2012]

[Lity et al.: Delta-oriented Model-Based SPL Regression Testing, PLEASE 2012]

# Delta-oriented Test Modeling



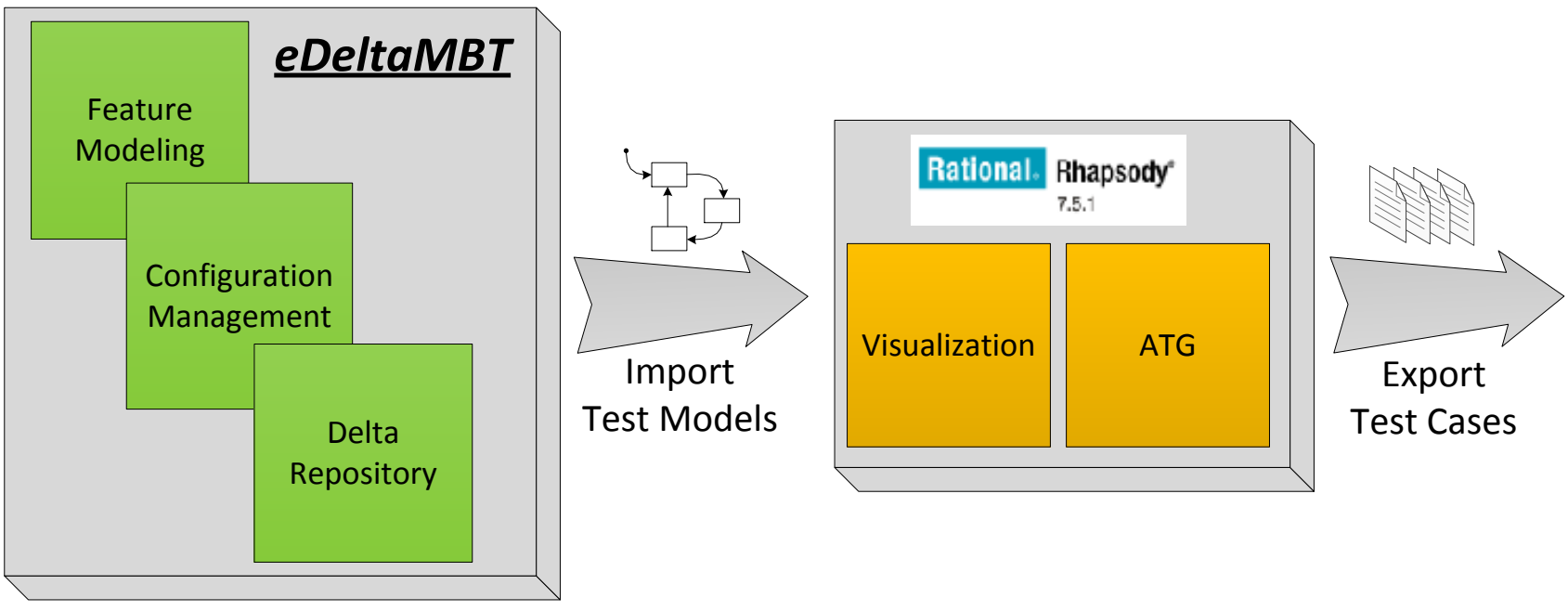
# Delta-oriented Test Artifact Evolution



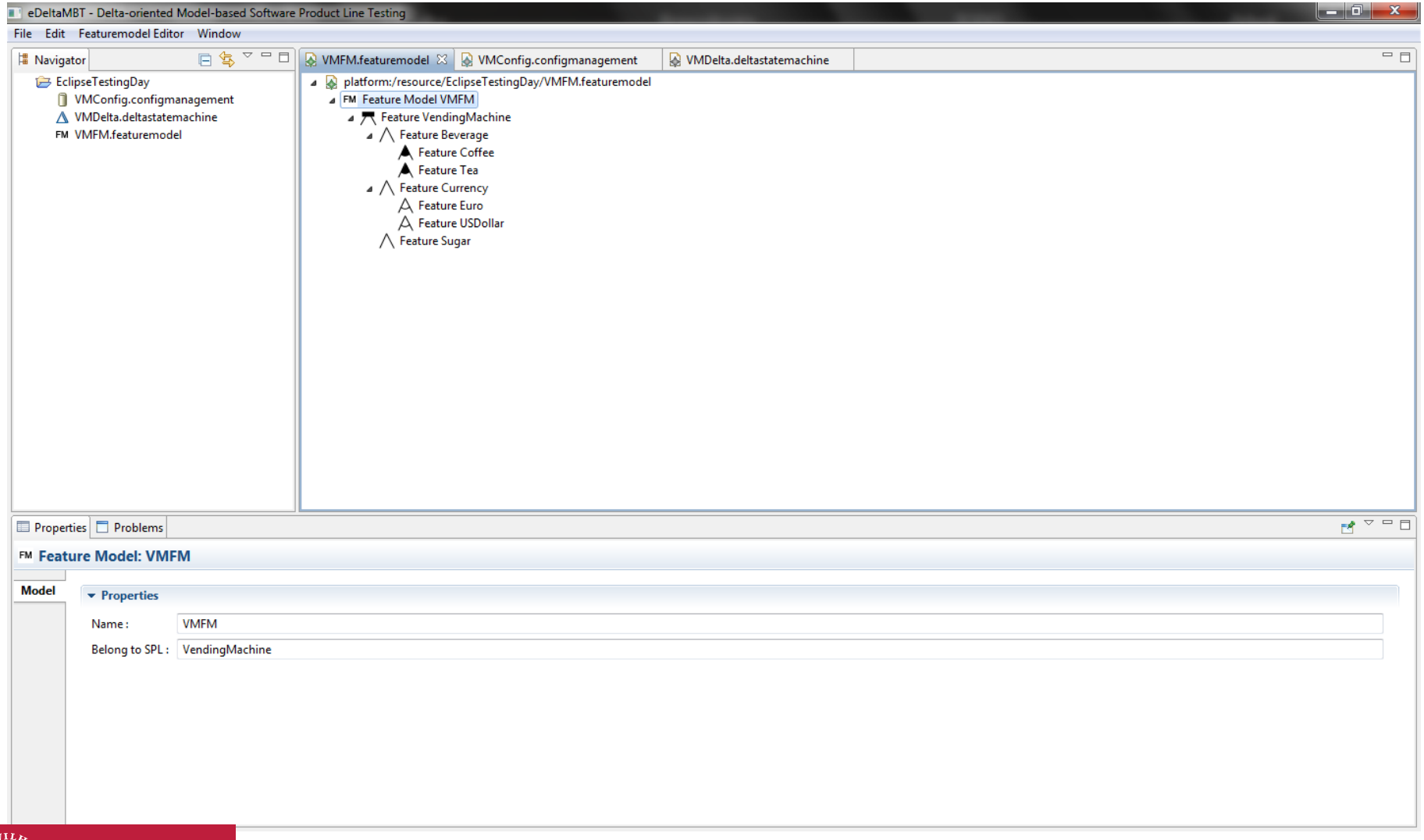
# Tool Support

- Crucial for efficient SPL testing
  - Automated test model generation
  - Automated test case generation
  - Automated test artifact evolution
  - Automated test case execution
  - ...
- Using existing (well-established) frameworks/tools
  - Eclipse Modeling Framework
  - EMF Validation Framework
  - Eclipse/RCP extendable
  - IBM Rational Rhapsody (Eclipse Plug-in)

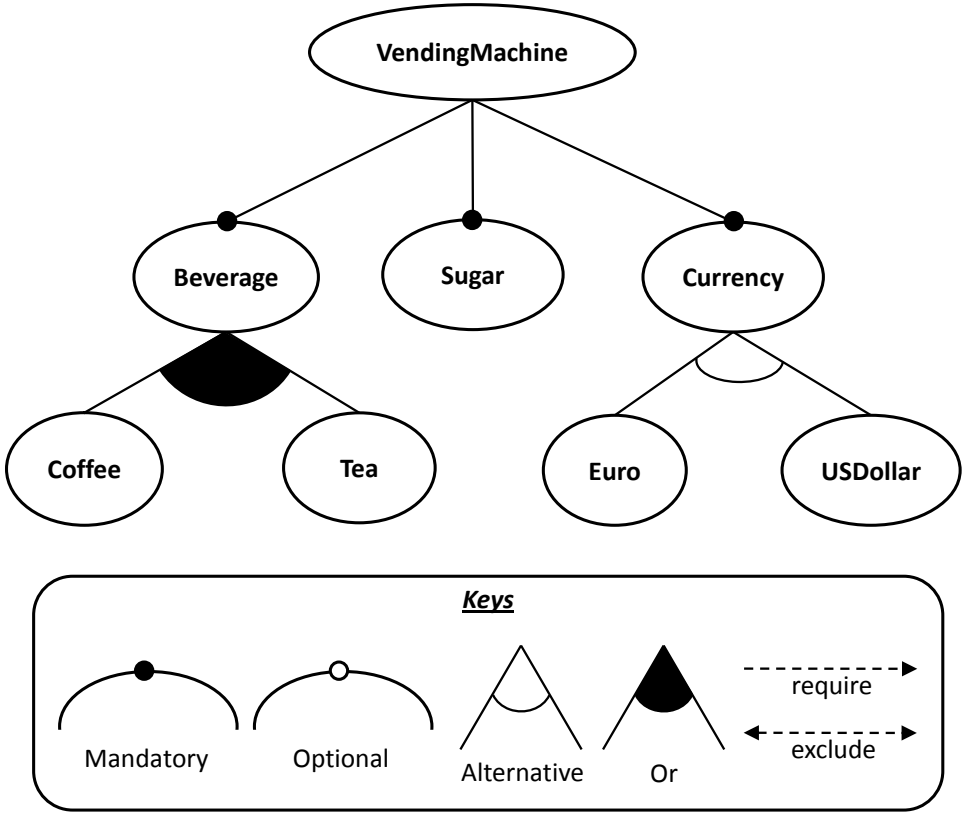
# Tool Chain


eDeltaMBT
Feature Modeling
Configuration Management
Delta Repository
Import Test Models
Rational. Rhapsody 7.5.1
Visualization
ATG
Export Test Cases

# eDeltaMBT – GUI

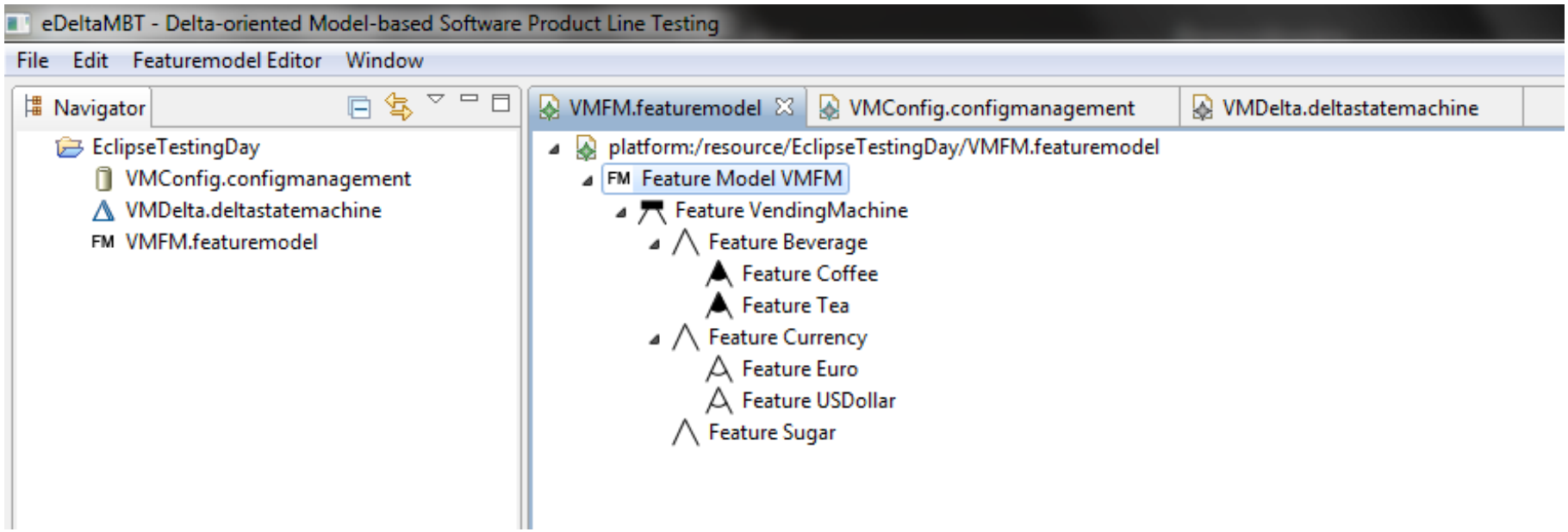


# Example – Feature Model

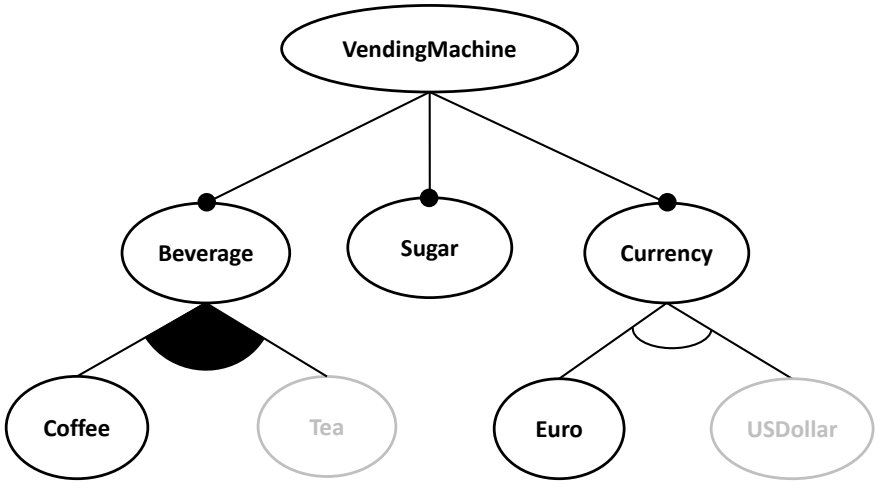



[A. Fantechi and S. Gnesi. Formal modeling for product families engineering. In SPLC, pages 193-202, Sept. 2008]

# eDeltaMBT – Feature Modeling

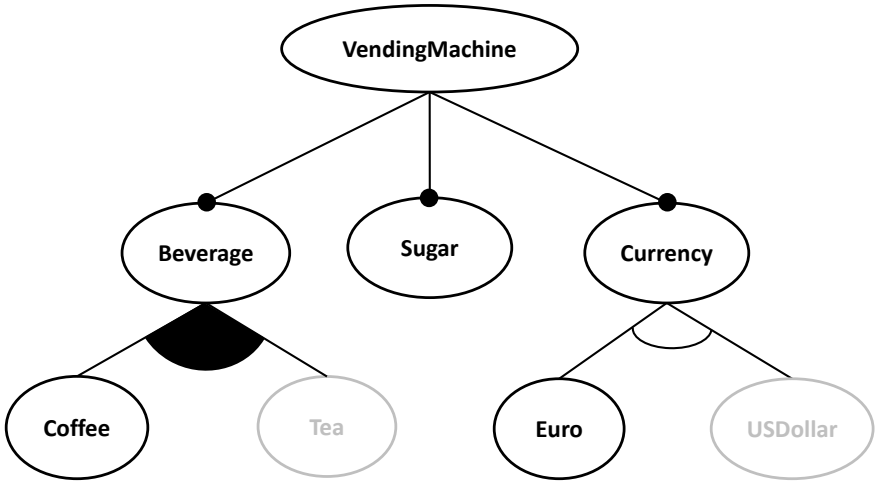


# Example – Configurations

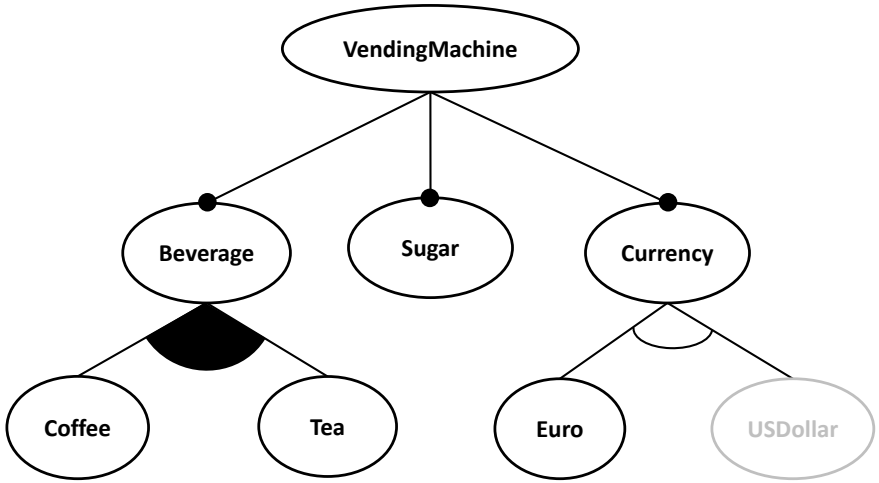



Feature Configuration Core

# Example – Configurations

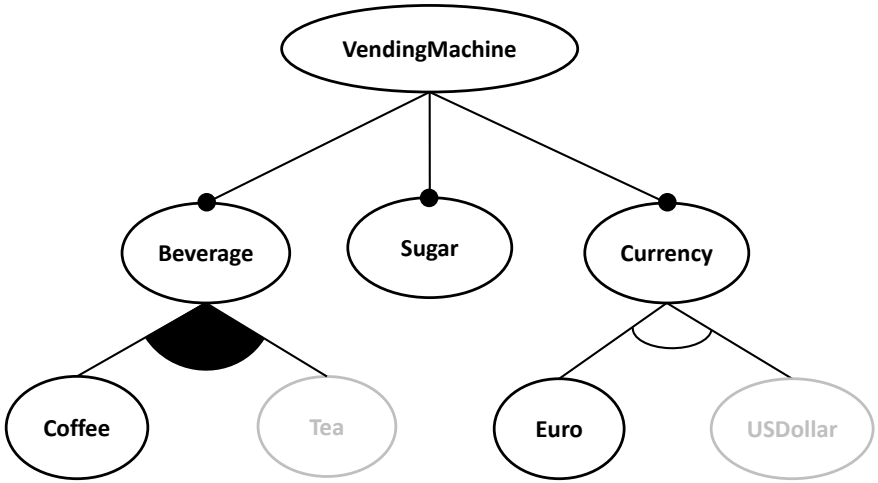



Feature Configuration Core

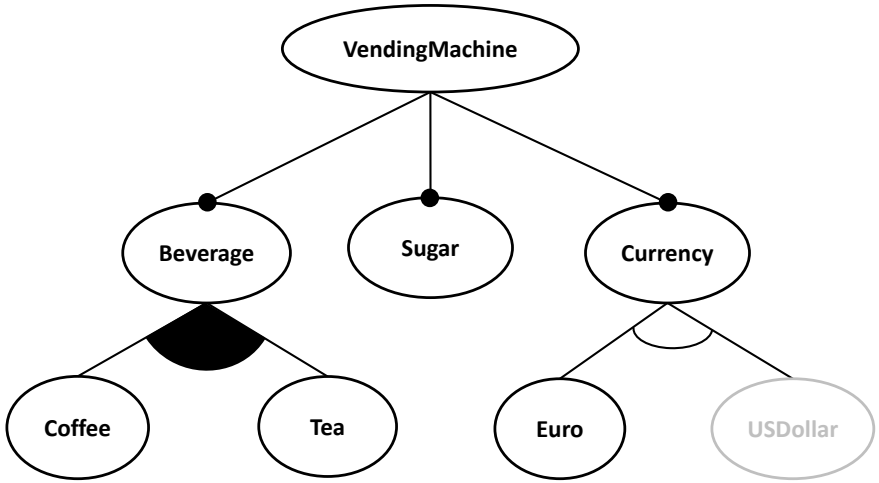



Feature Configuration P1

# Example – Configurations




Feature Configuration Core

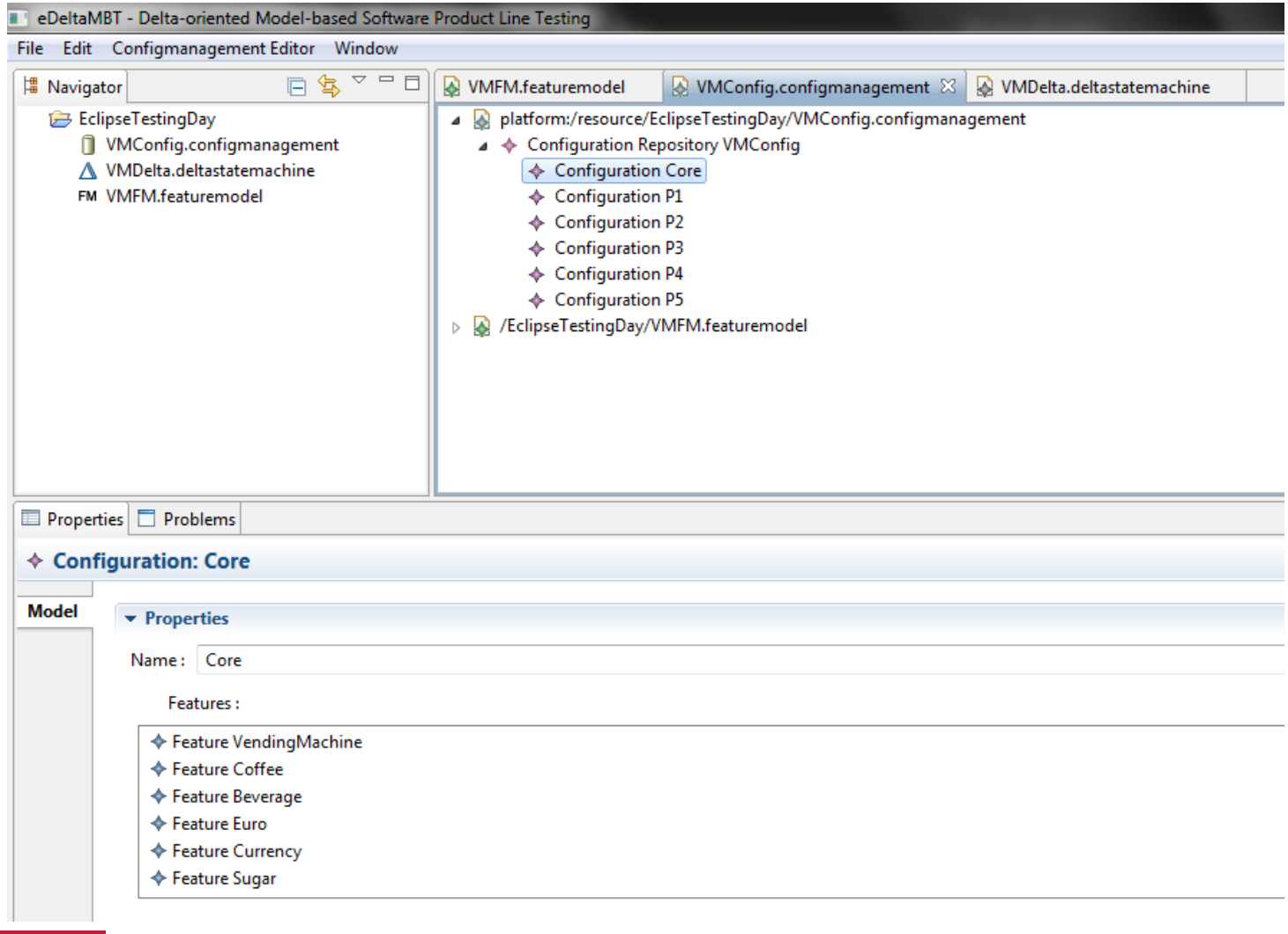



Feature Configuration P1

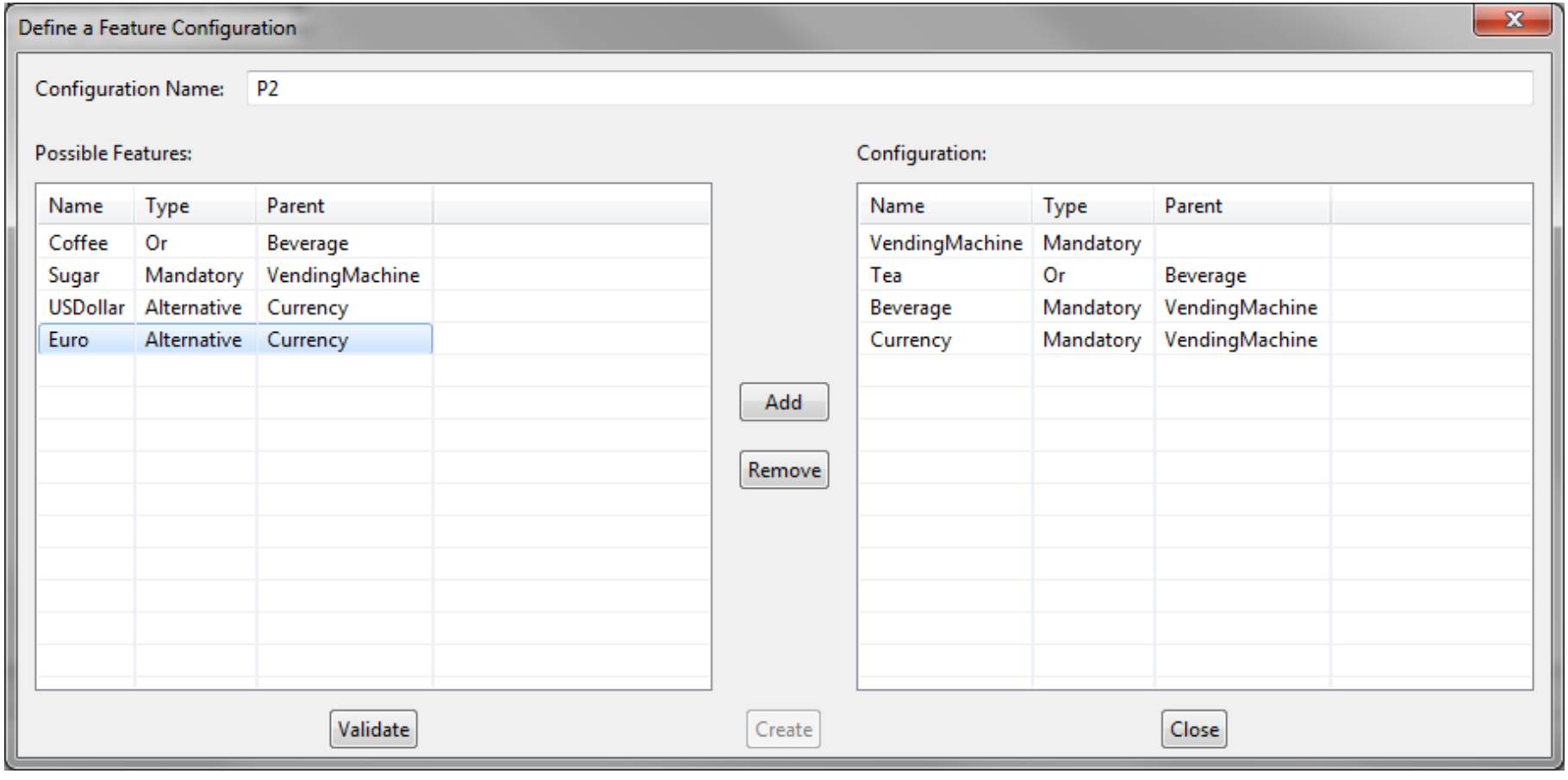
## Configurations

- 8 features combinable to 6 possible product variants

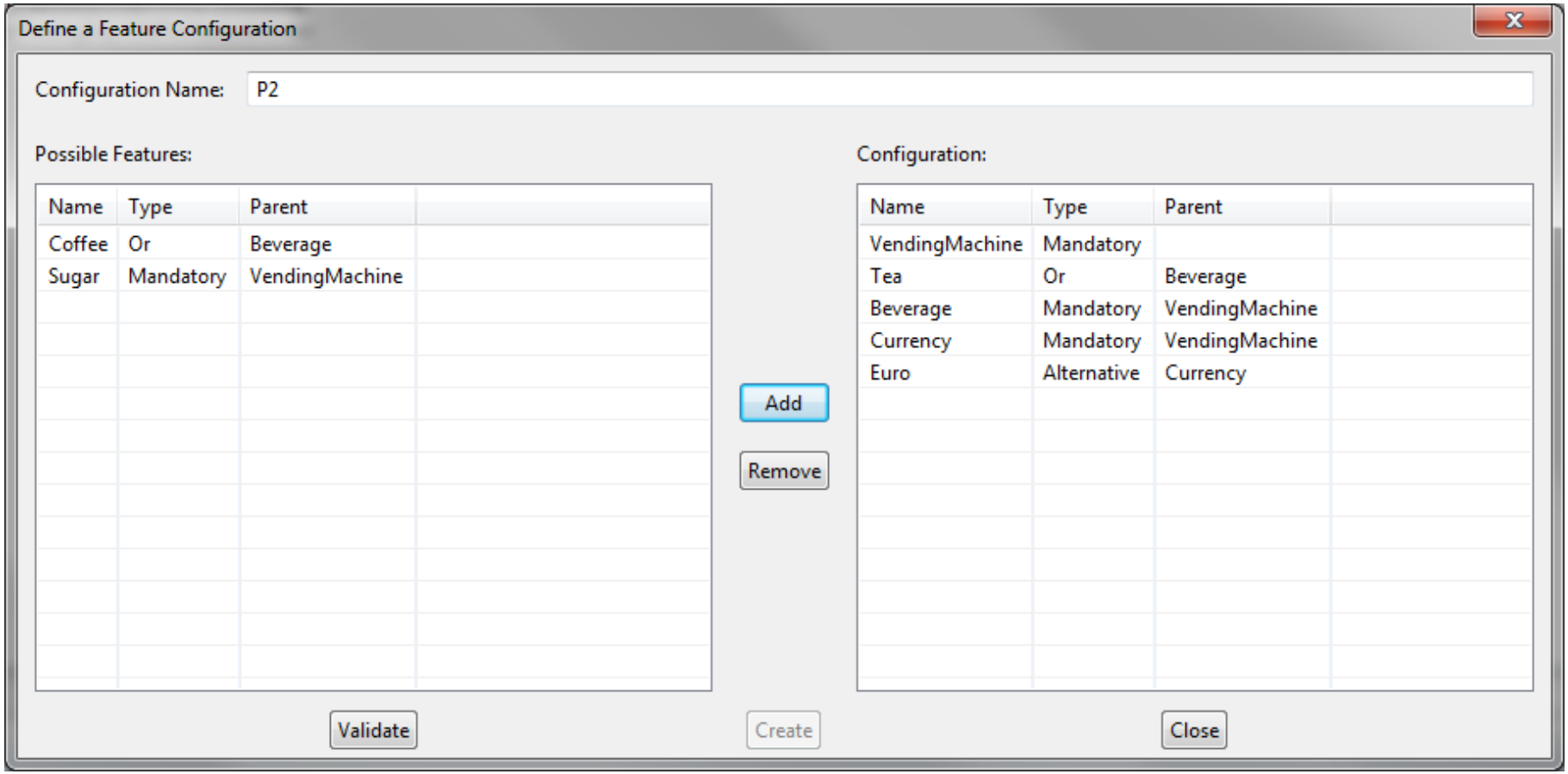
# eDeltaMBT – Configuration Management



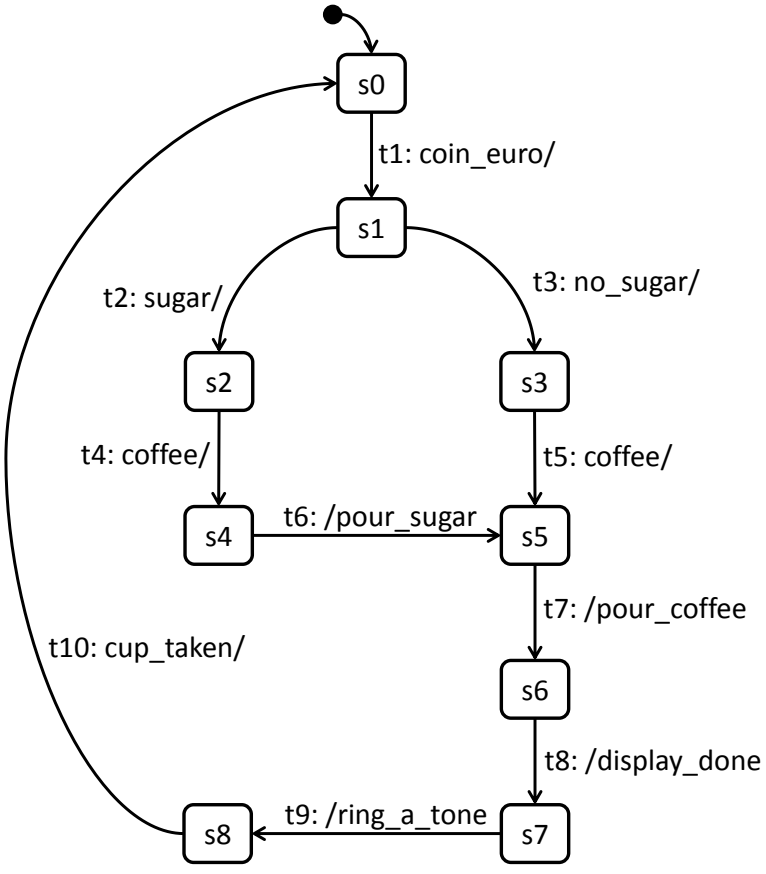
# eDeltaMBT – Configuration Management



# eDeltaMBT – Configuration Management

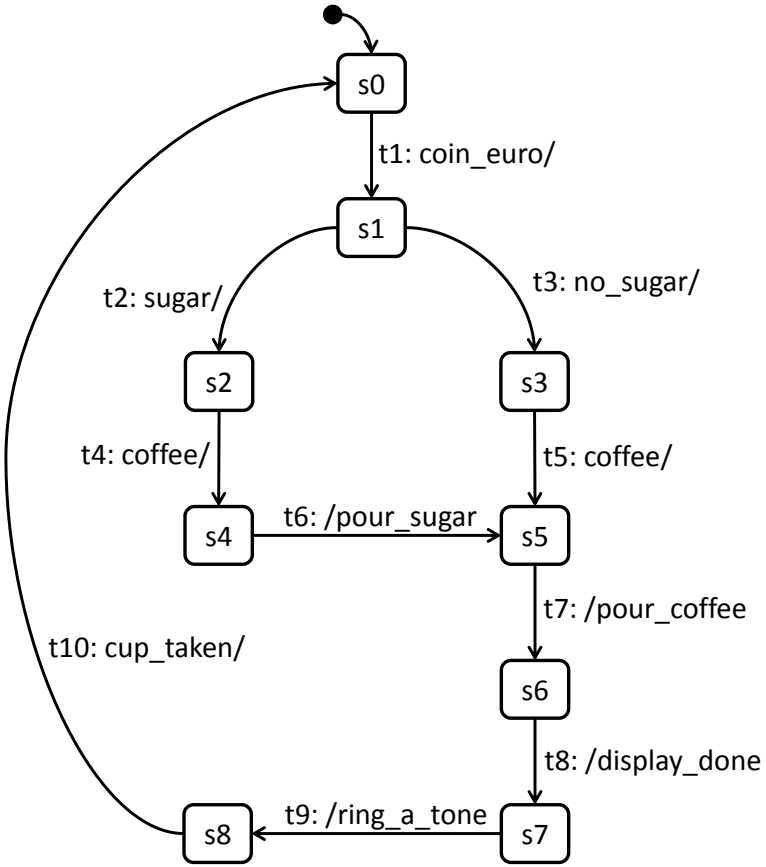


# Example – Delta Repository

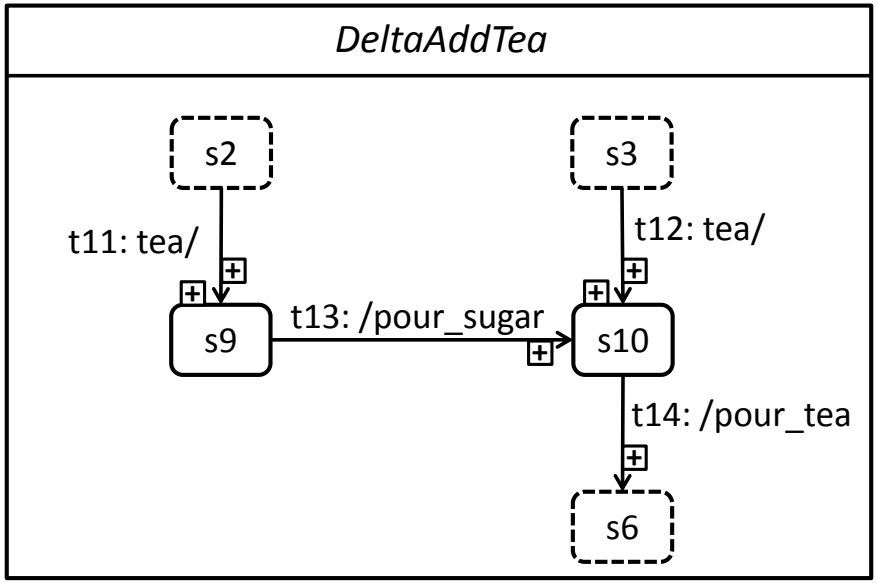



Core Test Model

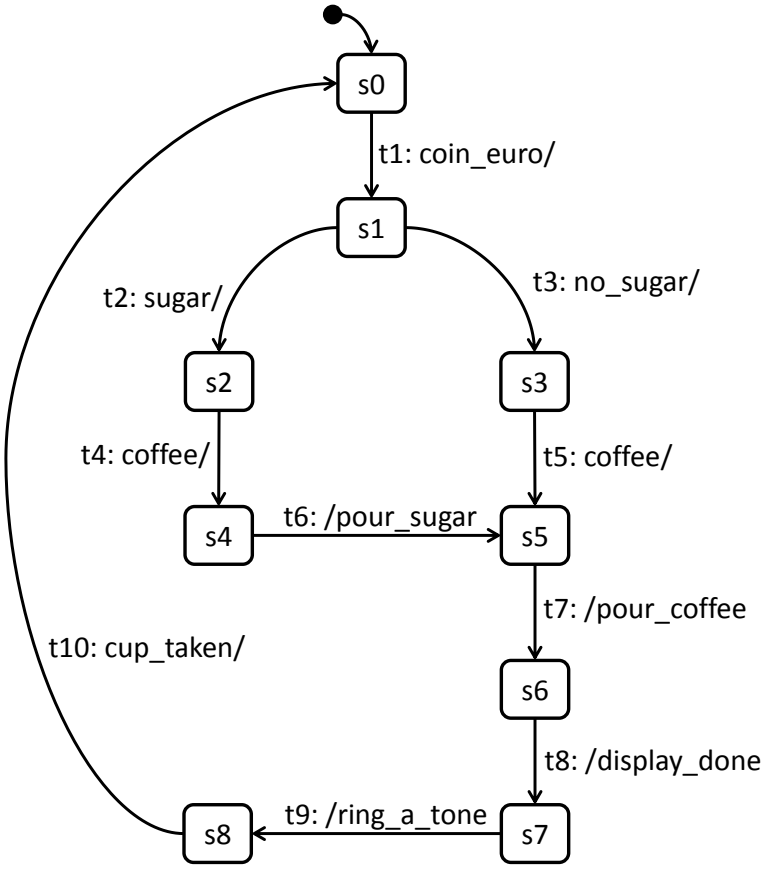
# Example – Delta Repository



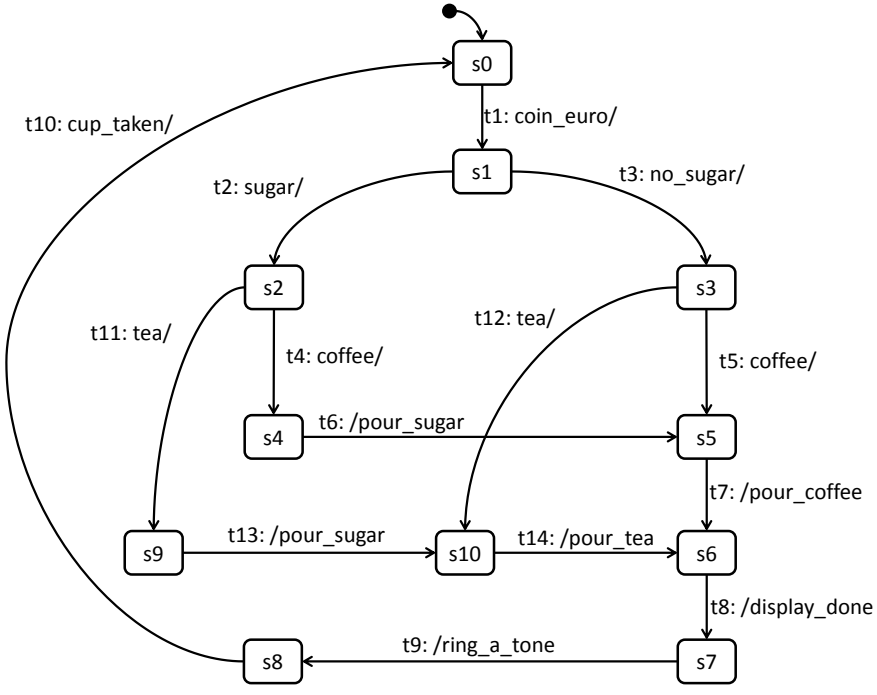

Core Test Model



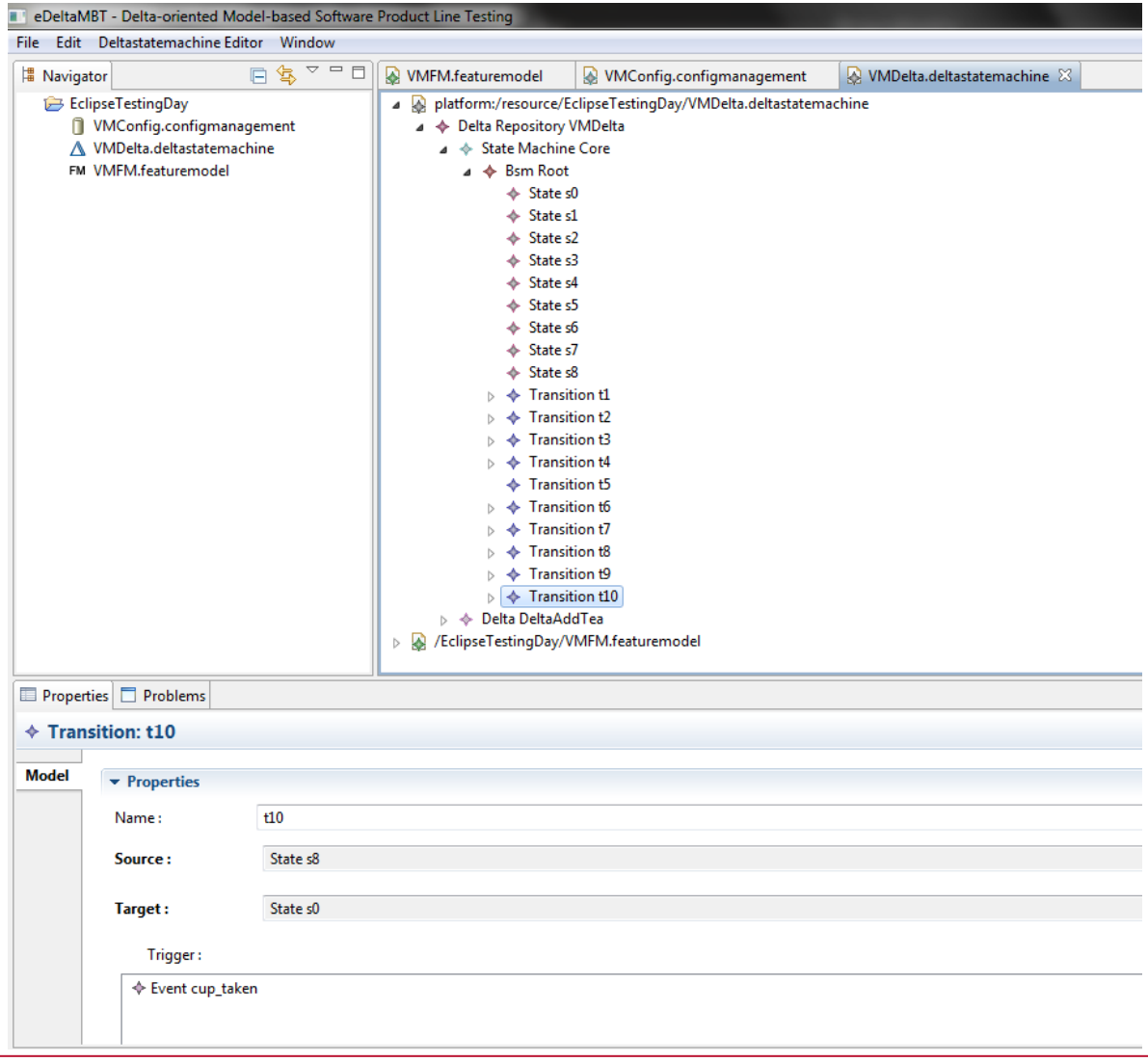
# Example – Delta Repository



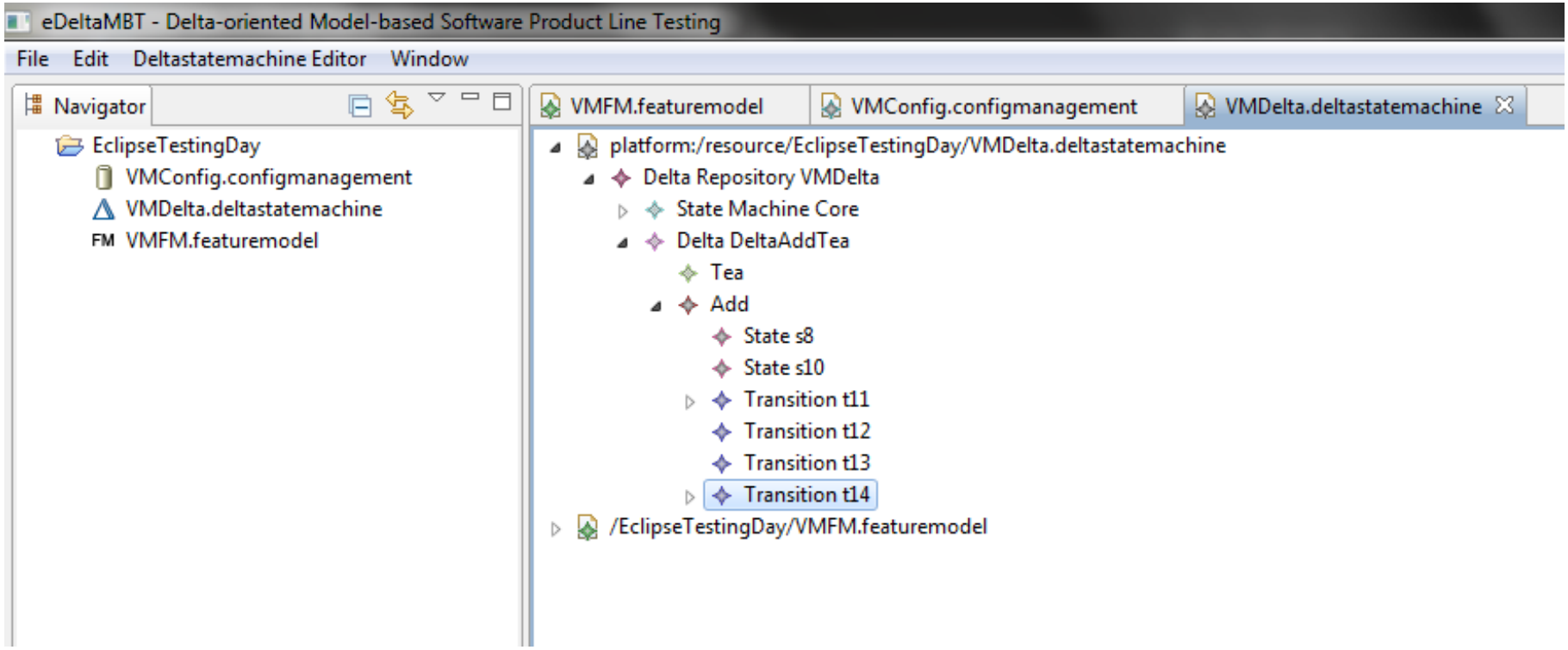

Core Test Model



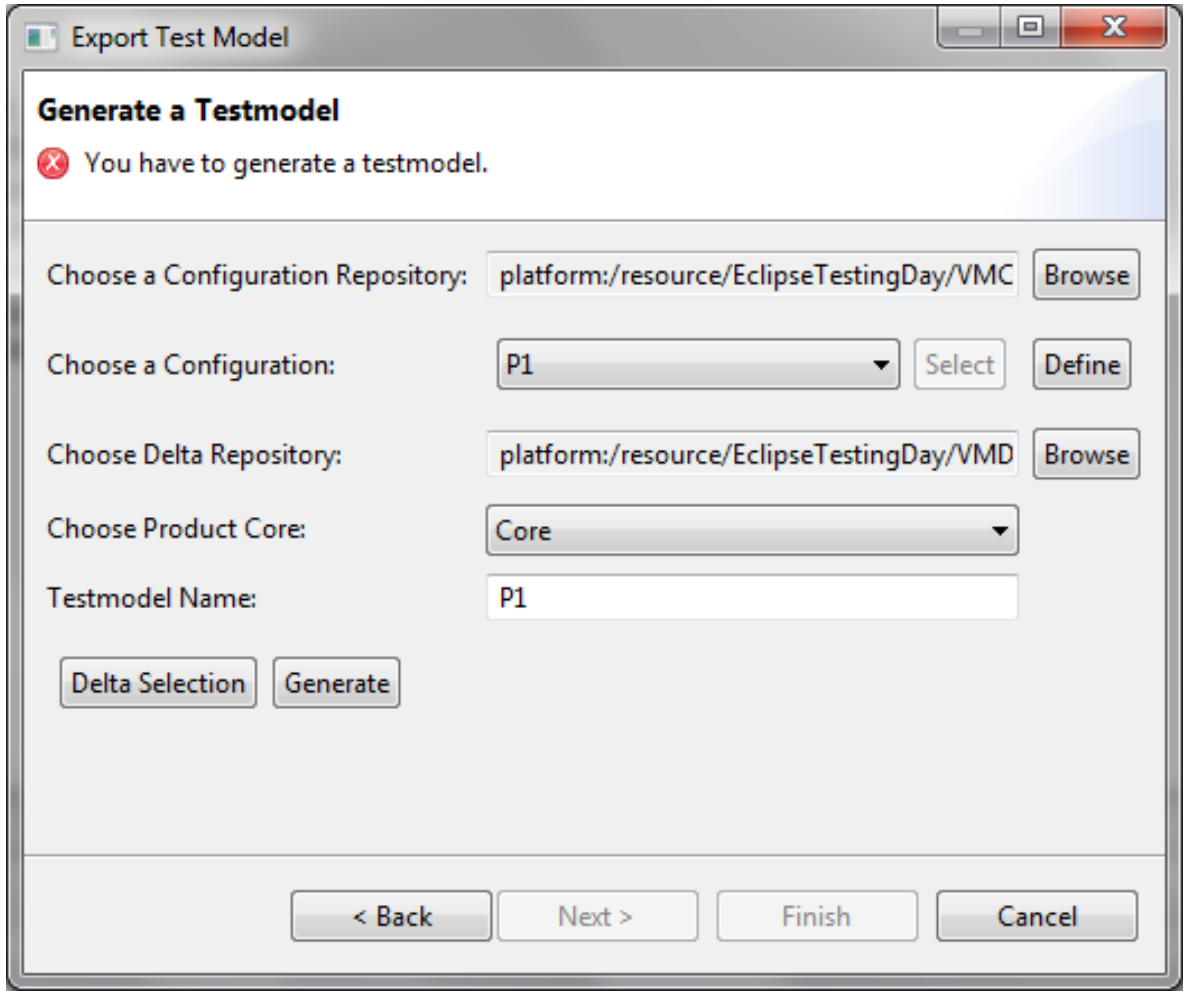
# eDeltaMBT – Delta Repository



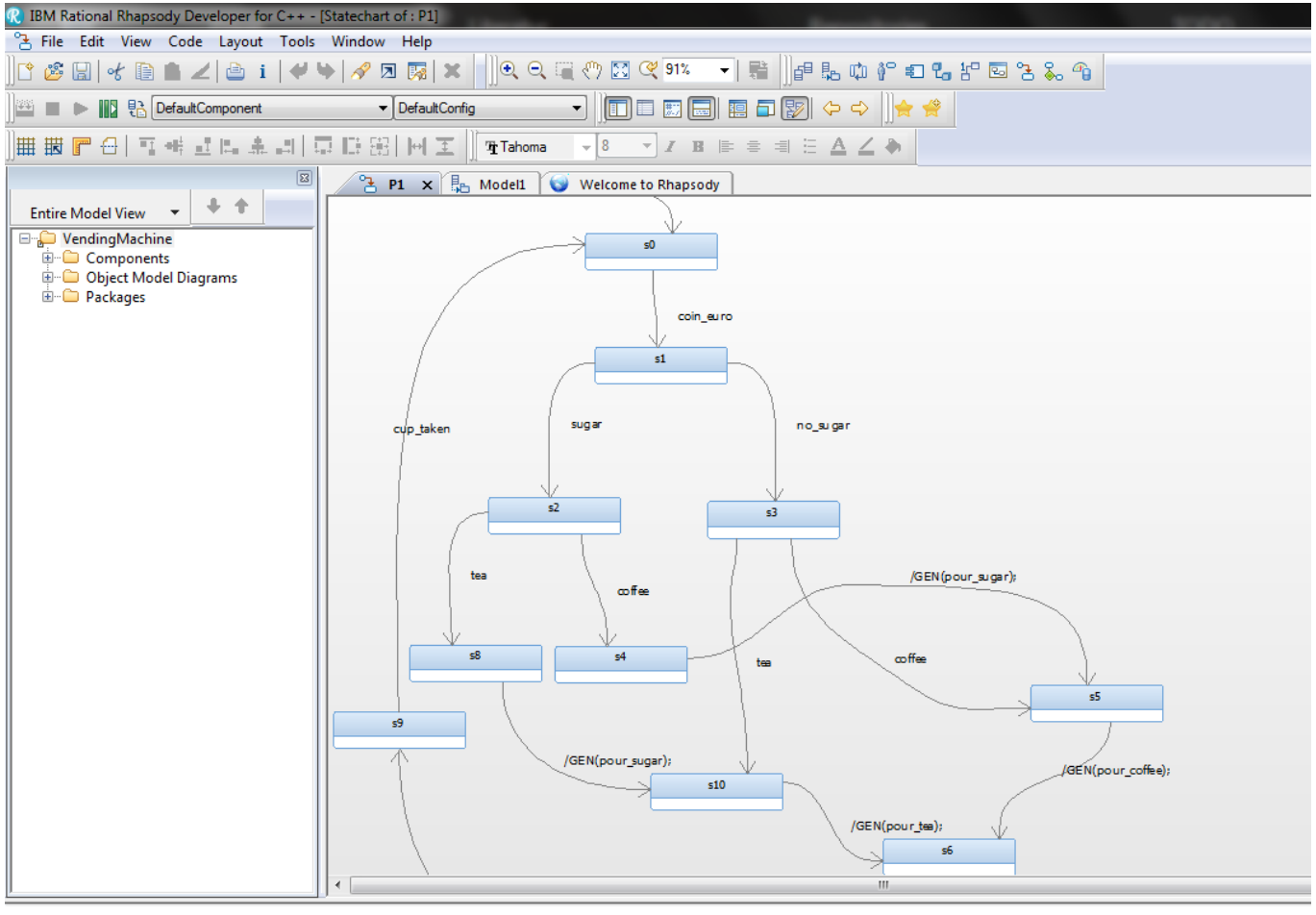
# eDeltaMBT – Delta Repository



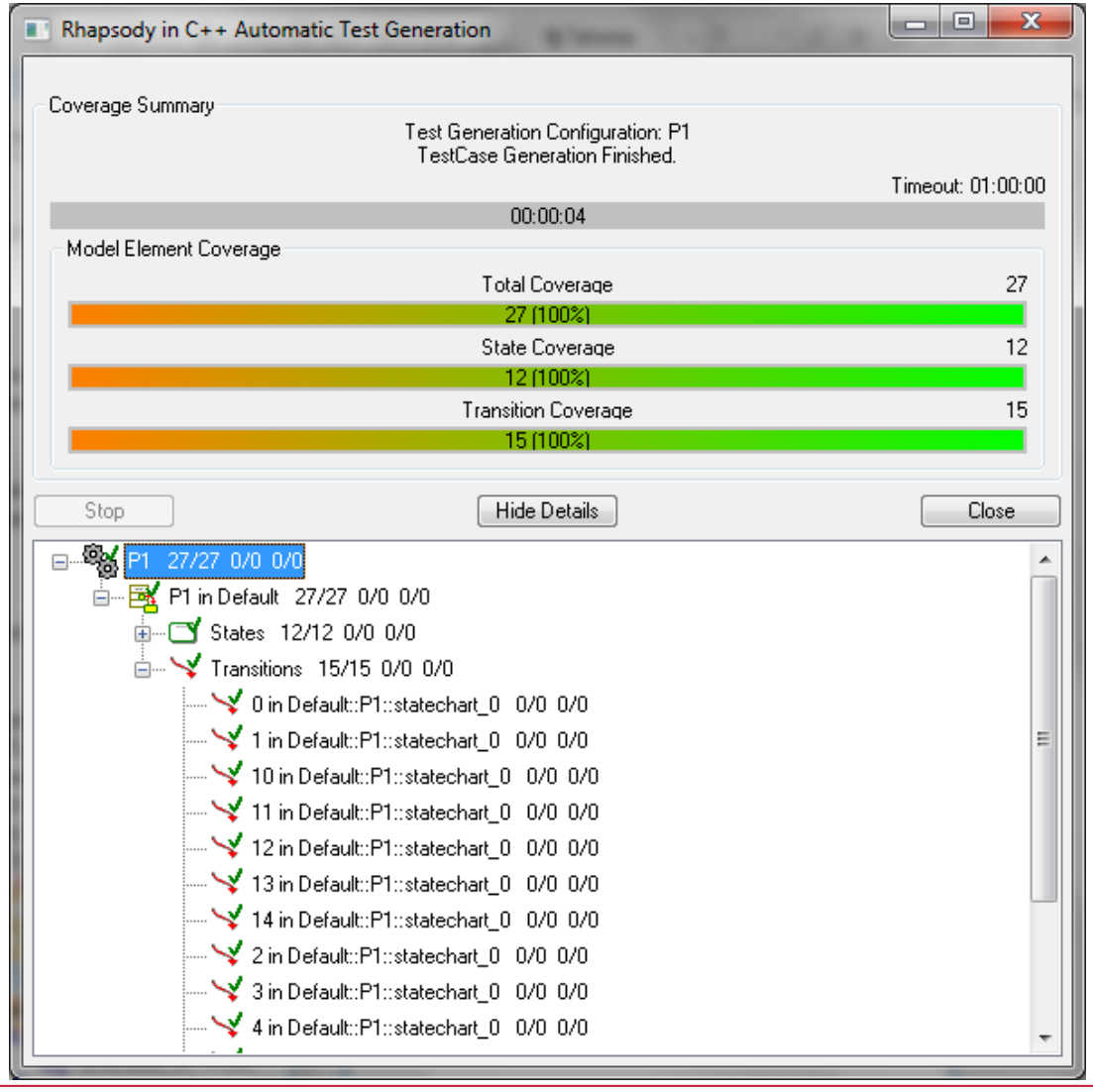
# eDeltaMBT – Test Model Import



# eDeltaMBT – Test Model Import



# Rhapsody – Test Case Generation

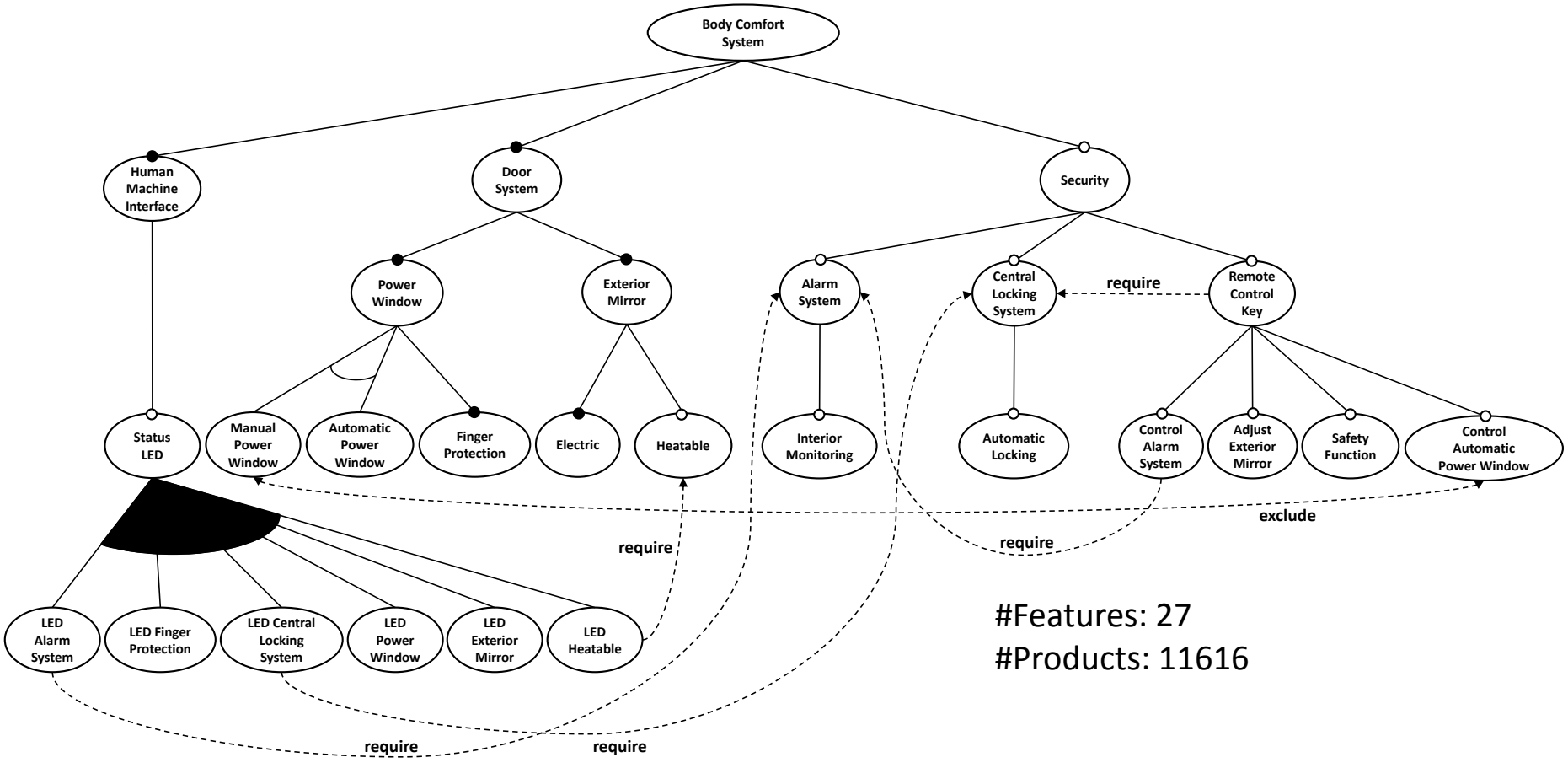


# Evaluation

## Objectives:

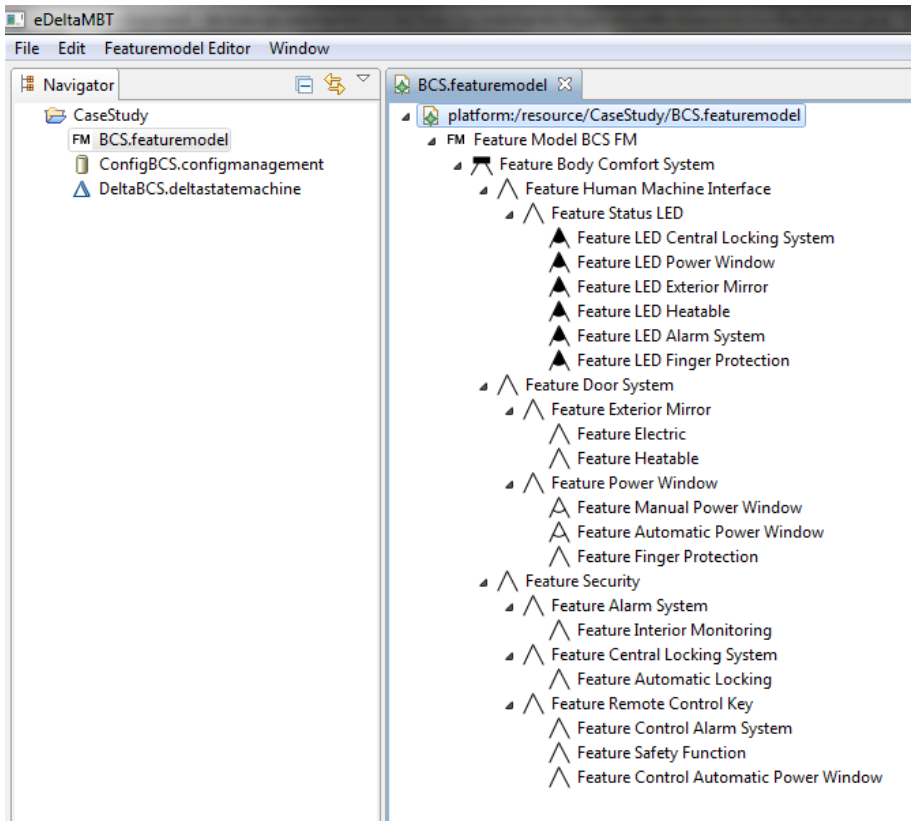
- Application of the delta-oriented SPL testing approach
  - Application of delta test modeling
  - Validation of tool support
- Comparison of results with an existing SPL testing approach (MoSo-PoLiTe [OZLG11])

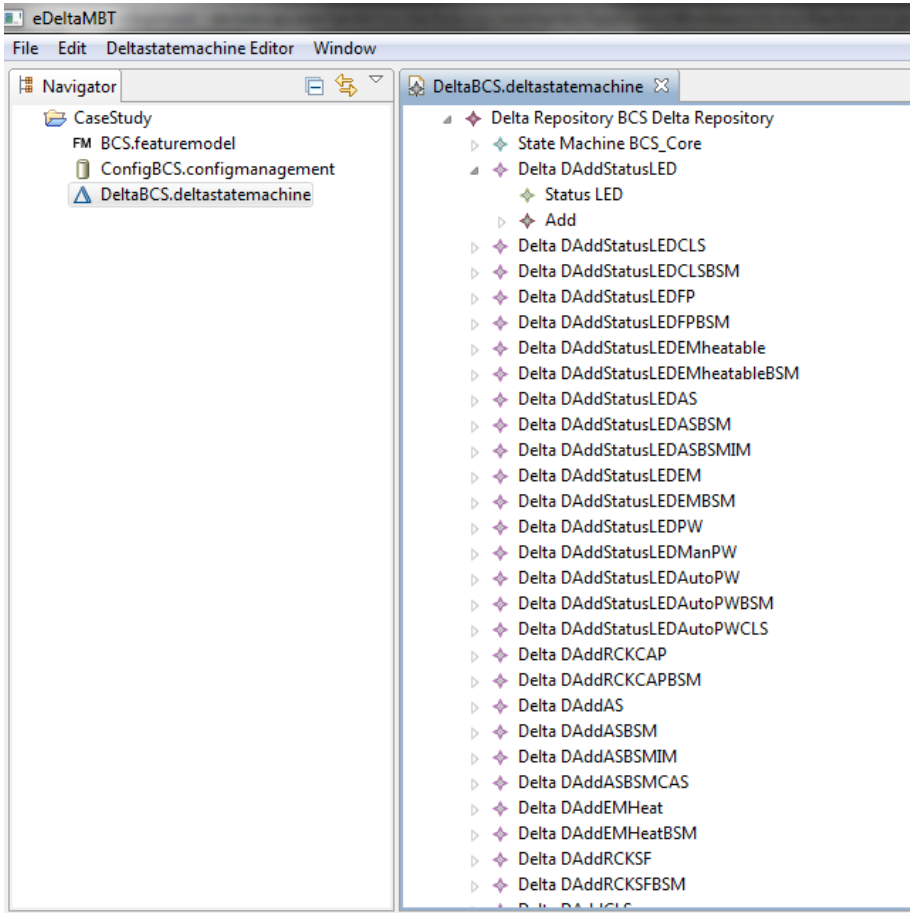
# Body Comfort System



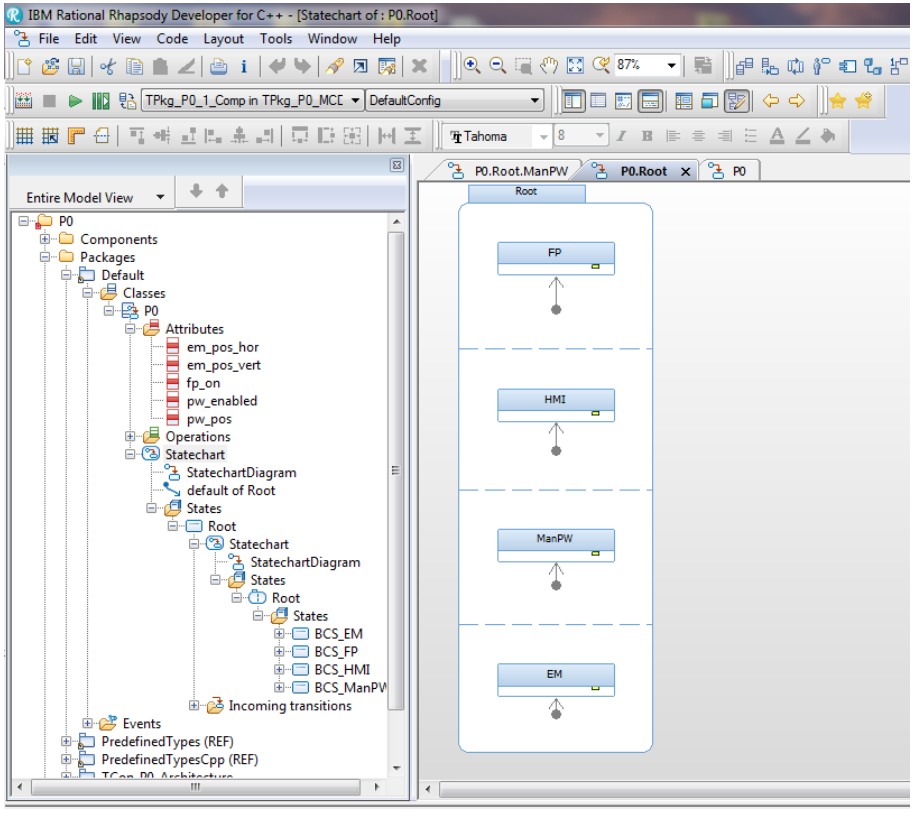

#Features: 27
#Products: 11616

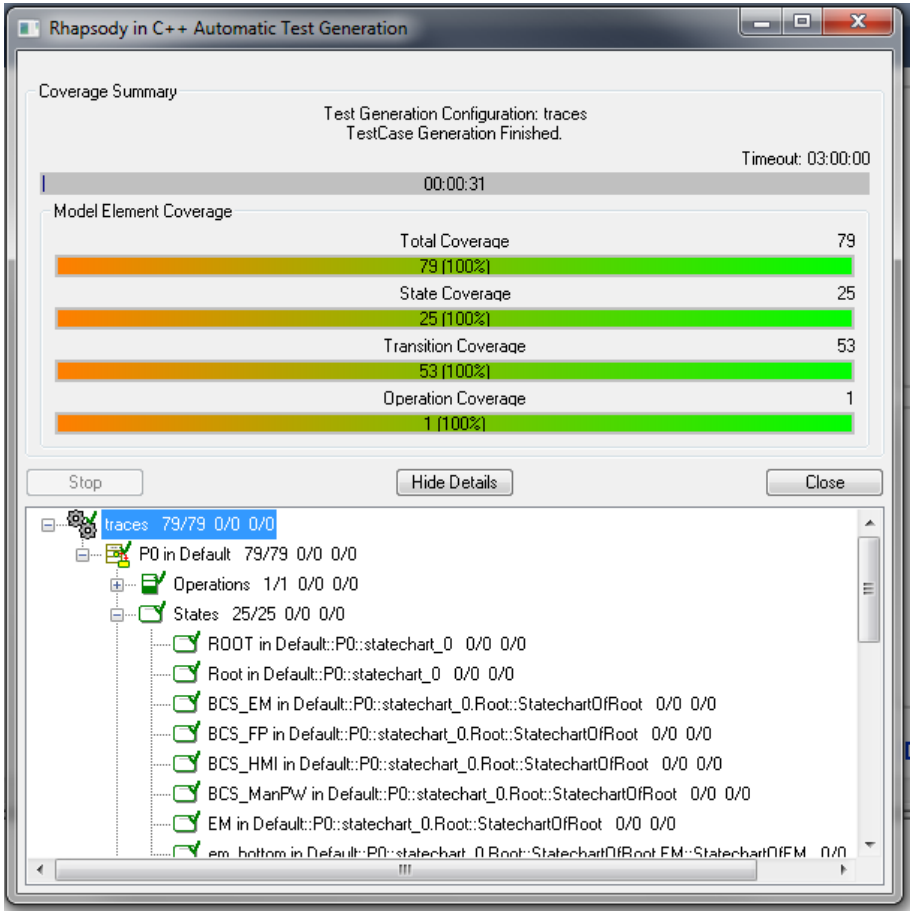
# Results (1/3)


eDeltaMBT
File Edit Featuremodel Editor Window
Navigator
CaseStudy
FM BCS.featuremodel
ConfigBCS.configmanagement
DeltaBCS.deltastatemachine
BCS.featuremodel
platform:/resource/CaseStudy/BCS.featuremodel
FM Feature Model BCS FM
Feature Body Comfort System
Feature Human Machine Interface
Feature Status LED
Feature LED Central Locking System
Feature LED Power Window
Feature LED Exterior Mirror
Feature LED Heatable
Feature LED Alarm System
Feature LED Finger Protection
Feature Door System
Feature Exterior Mirror
Feature Electric
Feature Heatable
Feature Power Window
Feature Manual Power Window
Feature Automatic Power Window
Feature Finger Protection
Feature Security
Feature Alarm System
Feature Interior Monitoring
Feature Central Locking System
Feature Automatic Locking
Feature Remote Control Key
Feature Control Alarm System
Feature Safety Function
Feature Control Automatic Power Window

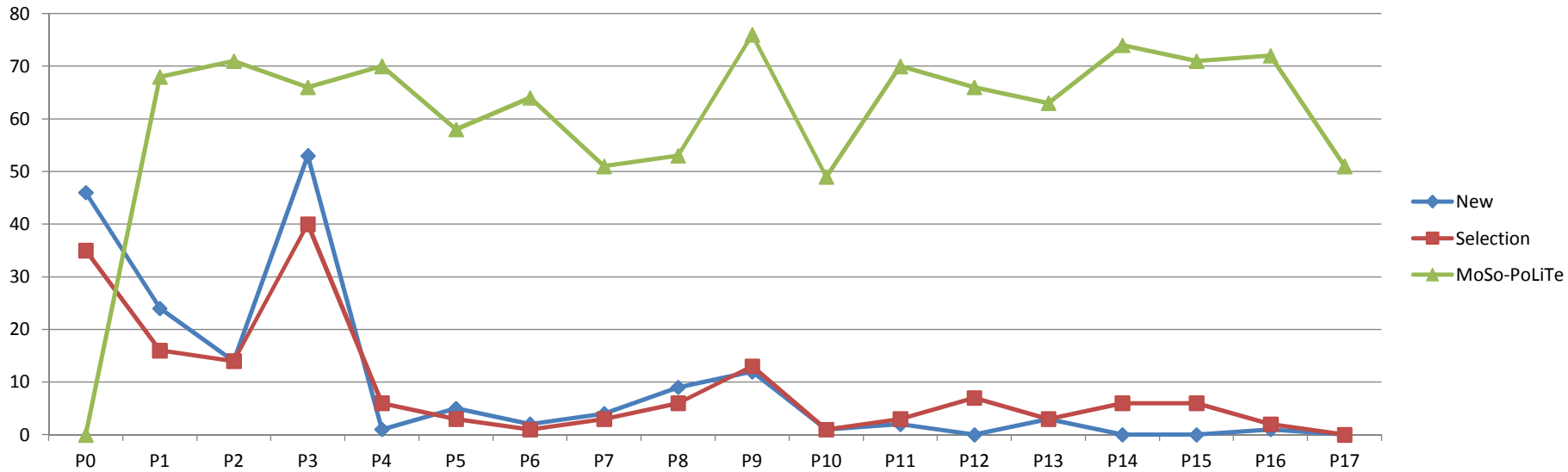

eDeltaMBT
File Edit Deltastatemachine Editor Window
Navigator
CaseStudy
FM BCS.featuremodel
ConfigBCS.configmanagement
DeltaBCS.deltastatemachine
DeltaBCS.deltastatemachine
Delta Repository BCS Delta Repository
State Machine BCS_Core
Delta DAddStatusLED
Status LED
Add
Delta DAddStatusLEDCLS
Delta DAddStatusLEDCLSBSM
Delta DAddStatusLEDFP
Delta DAddStatusLEDFPBSM
Delta DAddStatusLEDEMheatable
Delta DAddStatusLEDEMheatableBSM
Delta DAddStatusLEDAS
Delta DAddStatusLEDASBSM
Delta DAddStatusLEDASBSMIM
Delta DAddStatusLEDEM
Delta DAddStatusLEDEMBSM
Delta DAddStatusLEDPW
Delta DAddStatusLEDManPW
Delta DAddStatusLEDAutoPW
Delta DAddStatusLEDAutoPWBSM
Delta DAddStatusLEDAutoPWCLS
Delta DAddRCKCAP
Delta DAddRCKCAPBSM
Delta DAddAS
Delta DAddASBSM
Delta DAddASBSMIM
Delta DAddASBSMCAS
Delta DAddEMHeat
Delta DAddEMHeatBSM
Delta DAddRCKSF
Delta DAddRCKSFBSM

# Results (2/3)


IBM Rational Rhapsody Developer for C++ - [Statechart of : P0.Root]
File Edit View Code Layout Tools Window Help
Entire Model View
P0
Components
Packages
Default
Classes
P0
Attributes
em_pos_hor
em_pos_vert
fp_on
pw_enabled
pw_pos
Operations
Statechart
StatechartDiagram
default of Root
States
Root
Statechart
StatechartDiagram
States
Root
States
BCS_EM
BCS_FP
BCS_HMI
BCS_ManPW
Incoming transitions
Events
PredefinedTypes (REF)
PredefinedTypesCpp (REF)
P0.Root.ManPW
P0.Root
P0
Root
FP
HMI
ManPW
EM


Rhapsody in C++ Automatic Test Generation
Coverage Summary
Test Generation Configuration: traces
TestCase Generation Finished.
Timeout: 03:00:00
00:00:31
Model Element Coverage
Total Coverage 79
79 [100%]
State Coverage 25
25 [100%]
Transition Coverage 53
53 [100%]
Operation Coverage 1
1 [100%]
Stop
Hide Details
Close
traces 79/79 0/0 0/0
P0 in Default 79/79 0/0 0/0
Operations 1/1 0/0 0/0
States 25/25 0/0 0/0
ROOT in Default::P0::statechart_0 0/0 0/0
Root in Default::P0::statechart_0 0/0 0/0
BCS_EM in Default::P0::statechart_0.Root::StatechartOfRoot 0/0 0/0
BCS_FP in Default::P0::statechart_0.Root::StatechartOfRoot 0/0 0/0
BCS_HMI in Default::P0::statechart_0.Root::StatechartOfRoot 0/0 0/0
BCS_ManPW in Default::P0::statechart_0.Root::StatechartOfRoot 0/0 0/0
EM in Default::P0::statechart_0.Root::StatechartOfRoot 0/0 0/0

# Results (3/3)


80
70
60
50
40
30
20
10
0
P0
P1
P2
P3
P4
P5
P6
P7
P8
P9
P10
P11
P12
P13
P14
P15
P16
P17
New
Selection
MoSo-PoLiTe

# Conclusion

- SPL testing is challenging
  - Efficient test approaches are needed
  - Open field in research
- Combination of model-based and regression testing
  - Reuse of test artifacts
  - Incremental evolution of test artifacts
- Tool support based on well-established frameworks/tools
- Application of testing approach and tool support had positive results

# Future Work

- Modeling with graphical editors
- Automated test artifact evolution
- (Semi-)Automated reasoning about test case and test result reuse
- RCP extension with further research results
- Tool chain extension
- ...

Thank You for Your Attention! Questions?

TU Braunschweig, Institute for Programming and Reactive Systems
Sascha Lity
*Email*: lity@ips.cs.tu-bs.de

# References I


Carnegie Mellon University (CMU) – Software Engineering Institute.
Software Product Lines – Overview.
website, visited last August 2011.

Stanley M. Davis.
*Future Perfect*.
Addison-Wesley, Reading, Mass. :, 1987.

A. Fantechi and S. Gnesi.
Formal modeling for product families engineering.
In *12th International Software Product Line Conference, 2008. SPLC '08.*, pages 193 –202, sept. 2008.

S. Lity, M. Lochau, I. Schaefer, and U. Goltz.
Delta-oriented model-based spl regression testing.
In *3rd International Workshop on Product Line Approaches in Software Engineering (PLEASE), 2012*, pages 53 –56, june 2012.

# References II


Malte Lochau, Ina Schaefer, Jochen Kamischke, and Sascha Lity.
Incremental model-based testing of delta-oriented software product lines.
In Achim Brucker and Jacques Julliand, editors, *Tests and Proofs*, volume 7305 of *Lecture Notes in Computer Science*, pages 67–82. Springer Berlin / Heidelberg, 2012.


Erika Mir Olimpiew.
*Model-based testing for software product lines*.
PhD thesis, Fairfax, VA, USA, 2008.
AAI3310145.


Sebastian Oster, Marius Zink, Malte Lochau, and Mark Grechanik.
Pairwise feature-interaction testing for spls: potentials and limitations.
In *Proceedings of the 15th International Software Product Line Conference*, SPLC '11, pages 6:1–6:8, New York, NY, USA, 2011. ACM.


Klaus Pohl, Günter Böckle, and Frank J. van der Linden.
*Software Product Line Engineering: Foundations, Principles and Techniques*.
Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

# References III

Ina Schaefer.
Variability modeling for model-driven development of software product lines.
*Fourth International Workshop on Variability Modelling of Software-intensive Systems (VaMoS 2010)*, January 2010.

Mark Utting and Bruno Legeard.
*Practical Model-Based Testing: A Tools Approach*.
Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2006.