Feature model analysis and configuration: a 10 years journey with configuration stops

David Benavides
benavides@us.es

Configuration workshop, Graz – Sept 2018

@davbencue


Economy Class
Economy Class
Fr:HOCHIMINH CITY
To:KUALA LUMPUR
HỌ TÊN / NAME
Economy Class
883
Y
NGÀY / DATE
17OCT
20
BOARDING TIME
15:40
SEQ. NO.:36
Economy Class
Fr:HOCHIMINH CITY
To:KUALA LUMPUR
HỌ TÊN / NAME
CHUYẾN BAY / FLIGHT
883
Y
NGÀY / DATE
17OCT
GHẾ / SEAT
7A
SỐ TT / SEQ. NO.
GHI CHÚ / REMARKS
dreamstime

Sevilla, España
miércoles, 22:00
Despejado

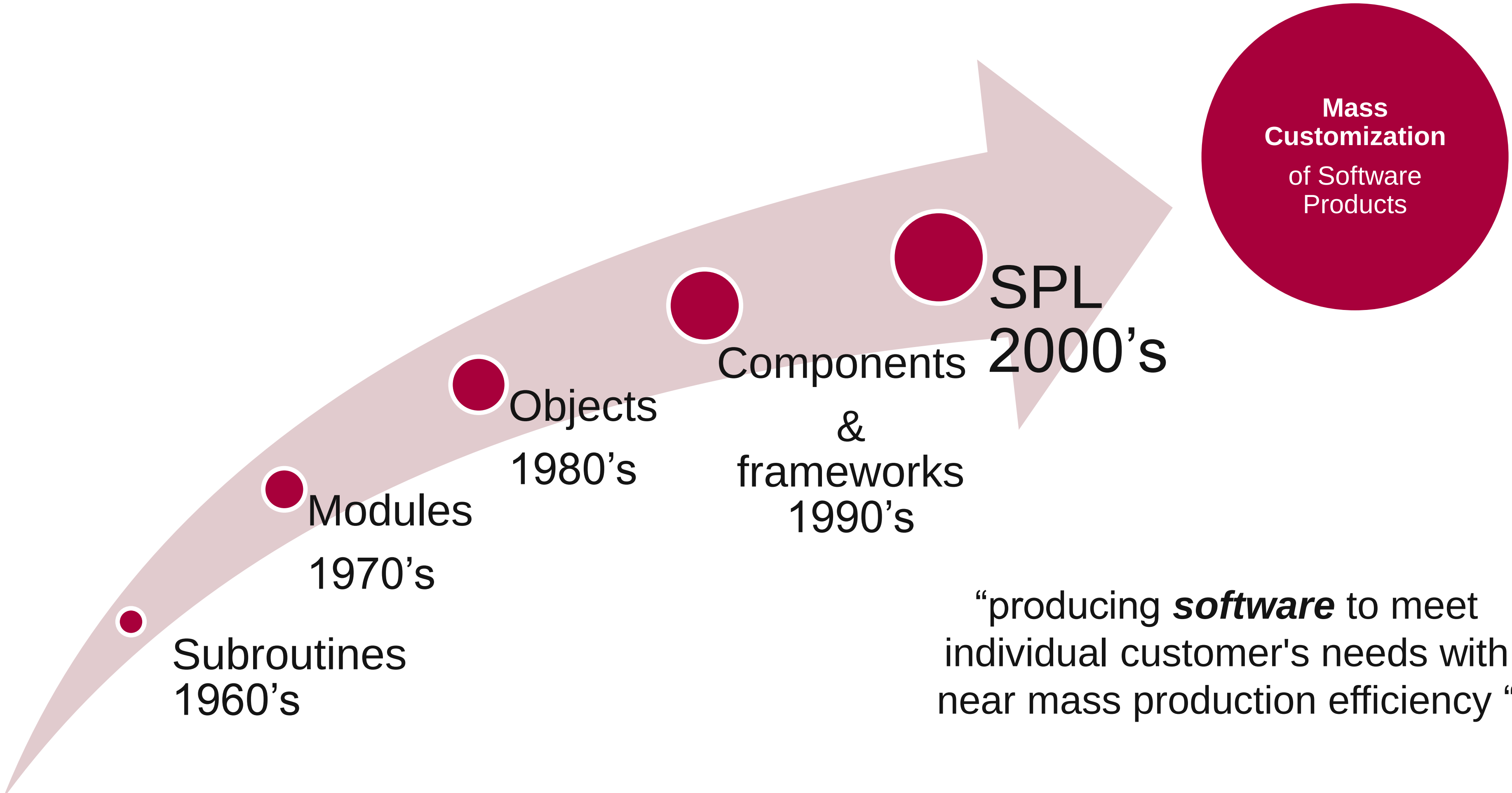
29 °C | °F

Graz

miércoles, 22:00
Despejado

7 °C | °F

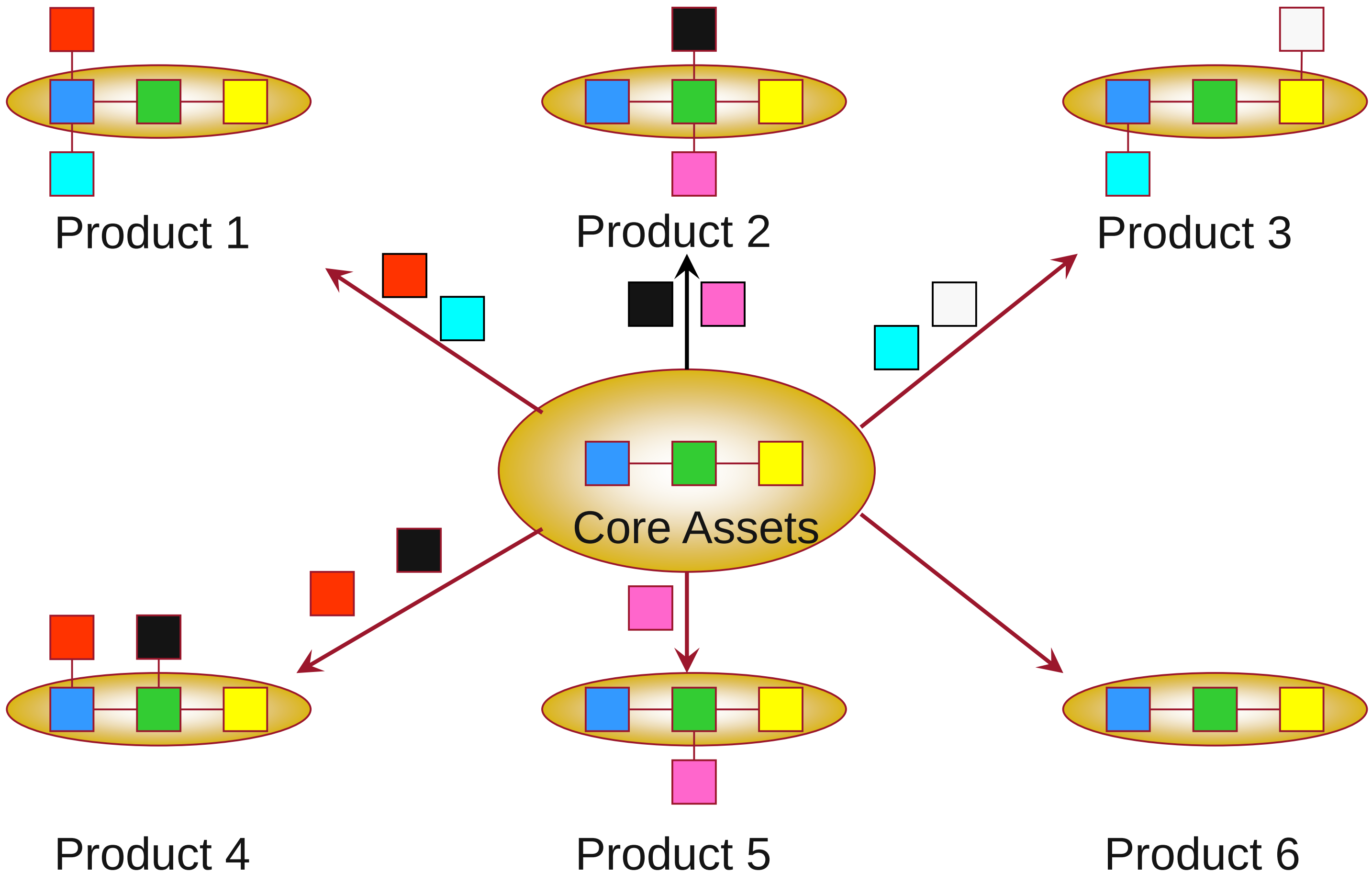
Scope of the journey: **software** product lines



“producing ***software*** to meet individual customer's needs with near mass production efficiency “

Software product lines

Product Lines Approach (*mass customization*)



A more practical view of the SPL framework

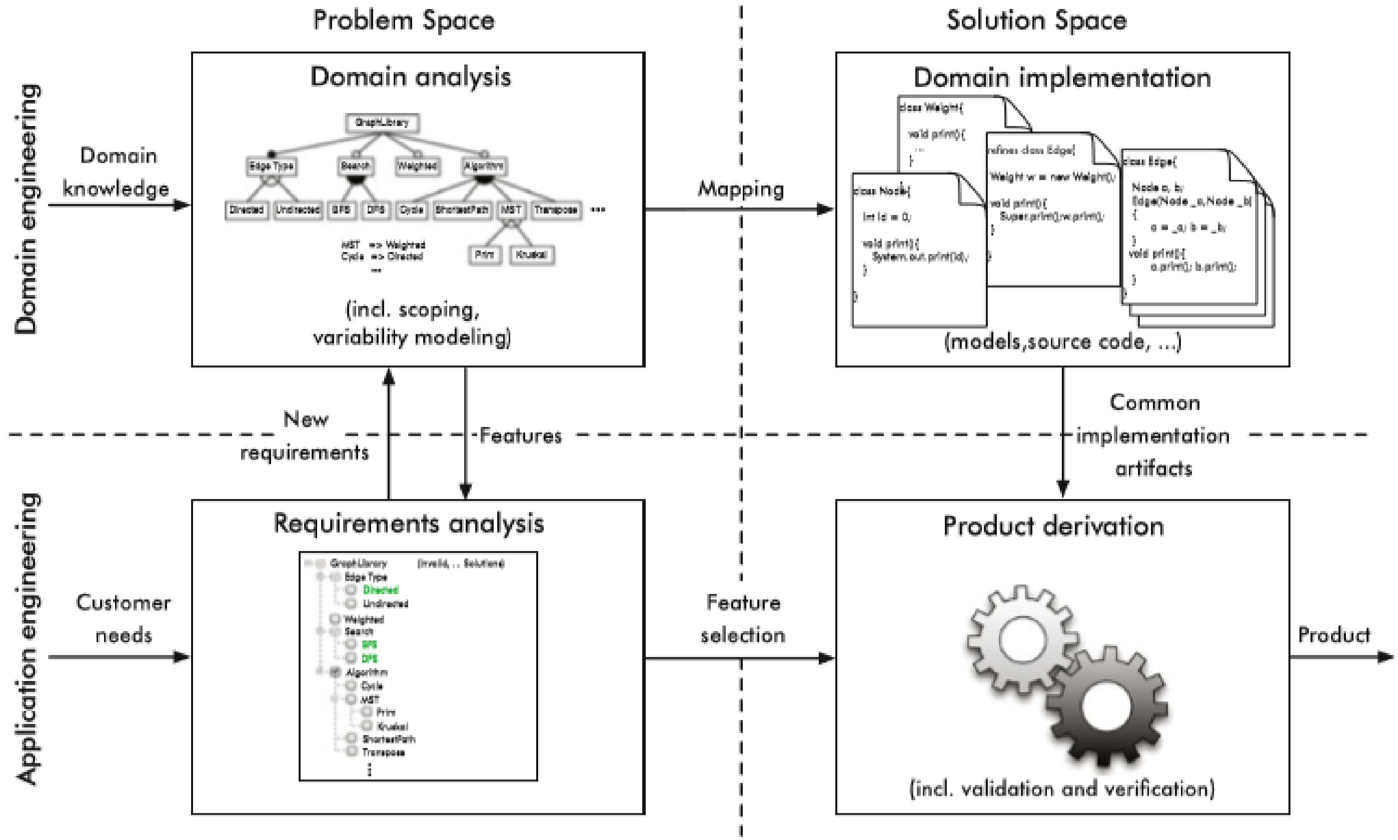
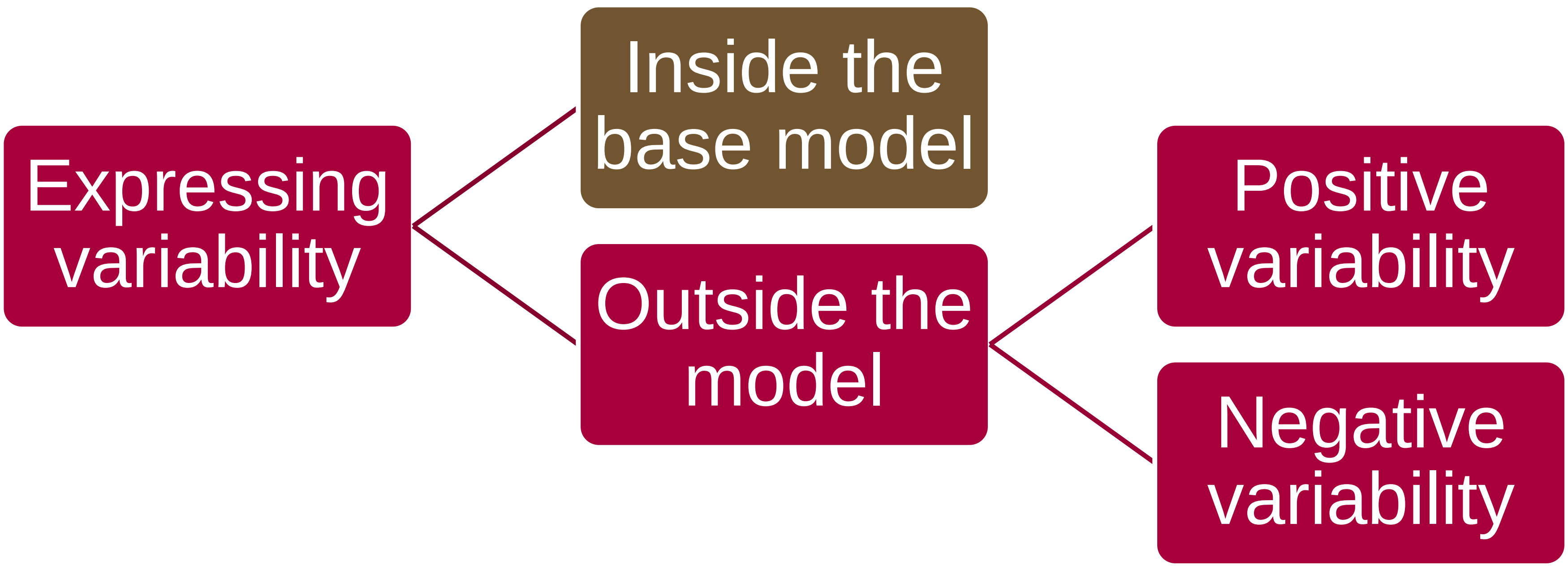


Fig. 1.1 An overview on software product-line engineering

From “Mastering Software Variability with FeatureIDE”

How to model variability?

How to model variability



Inside the model

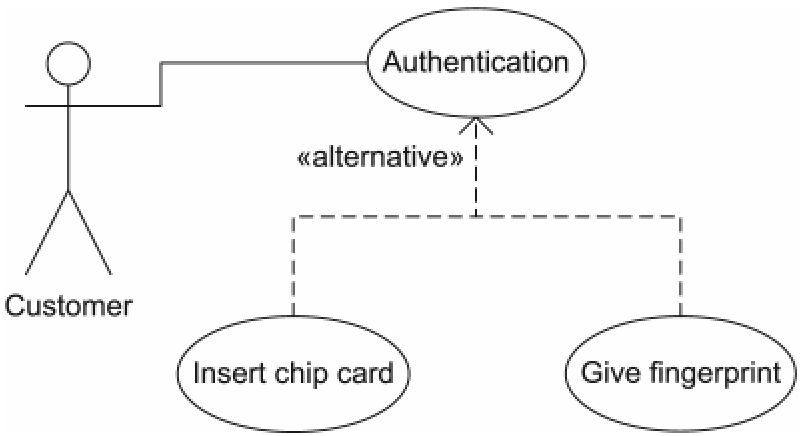


Figure 5: Example of an alternative relationship

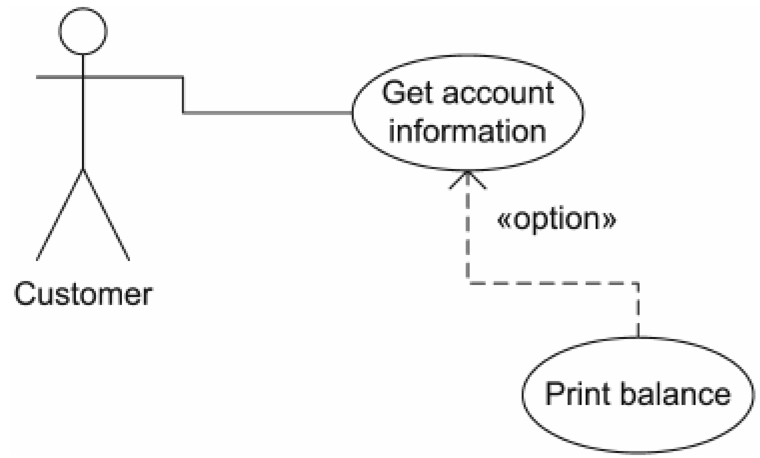
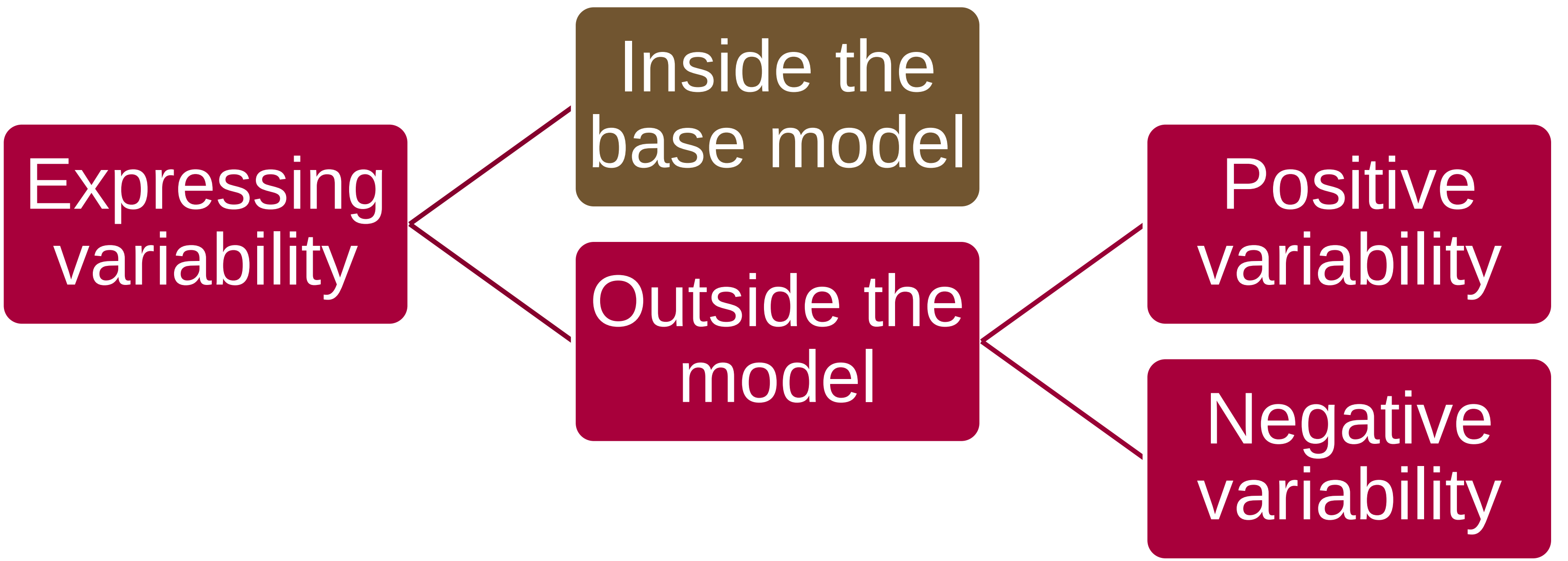


Figure 6: Example of an optional relationship

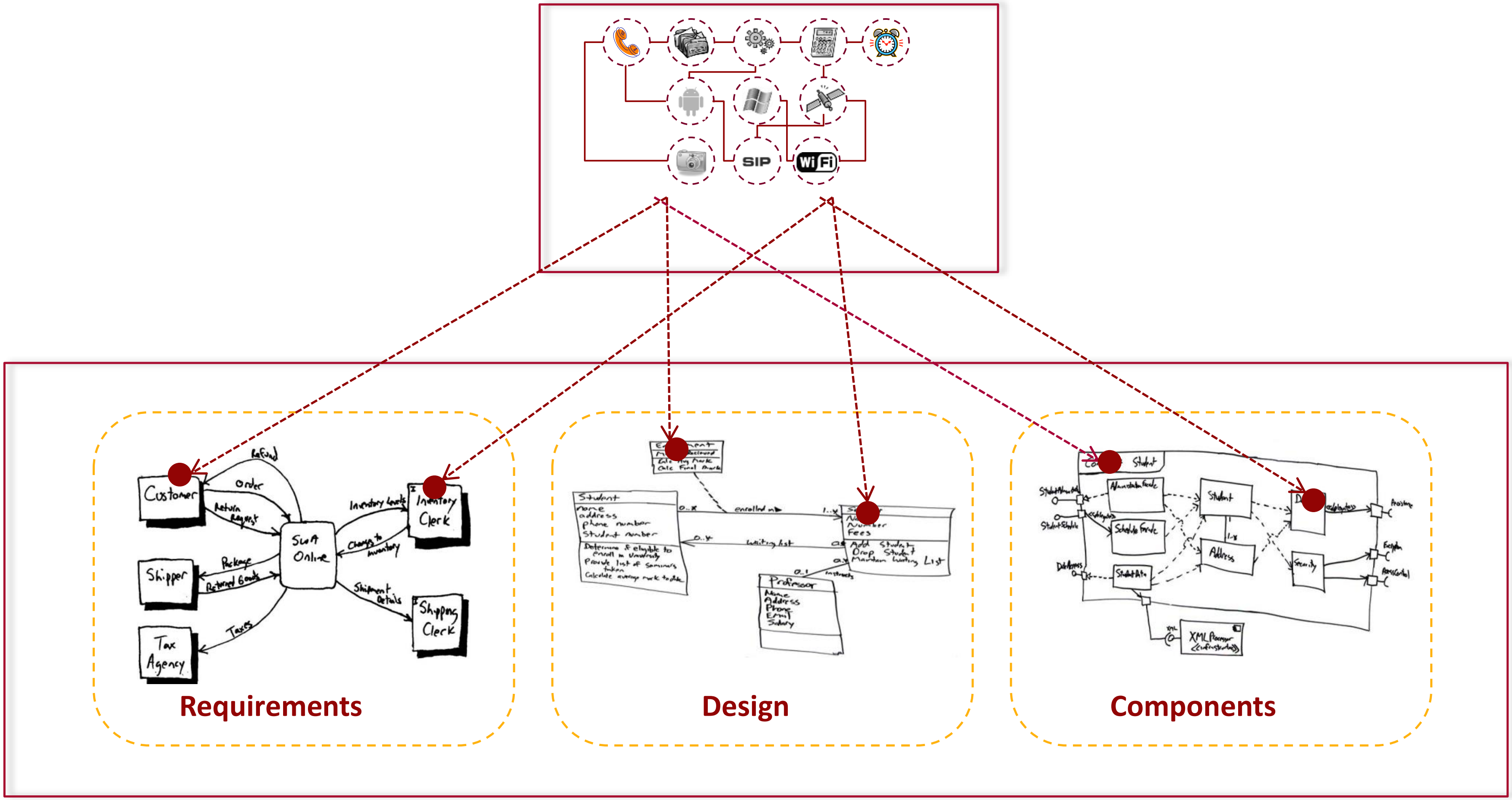
Taken from [Modeling variability by UML use case diagrams] by Von der Massen et al.

How to model variability



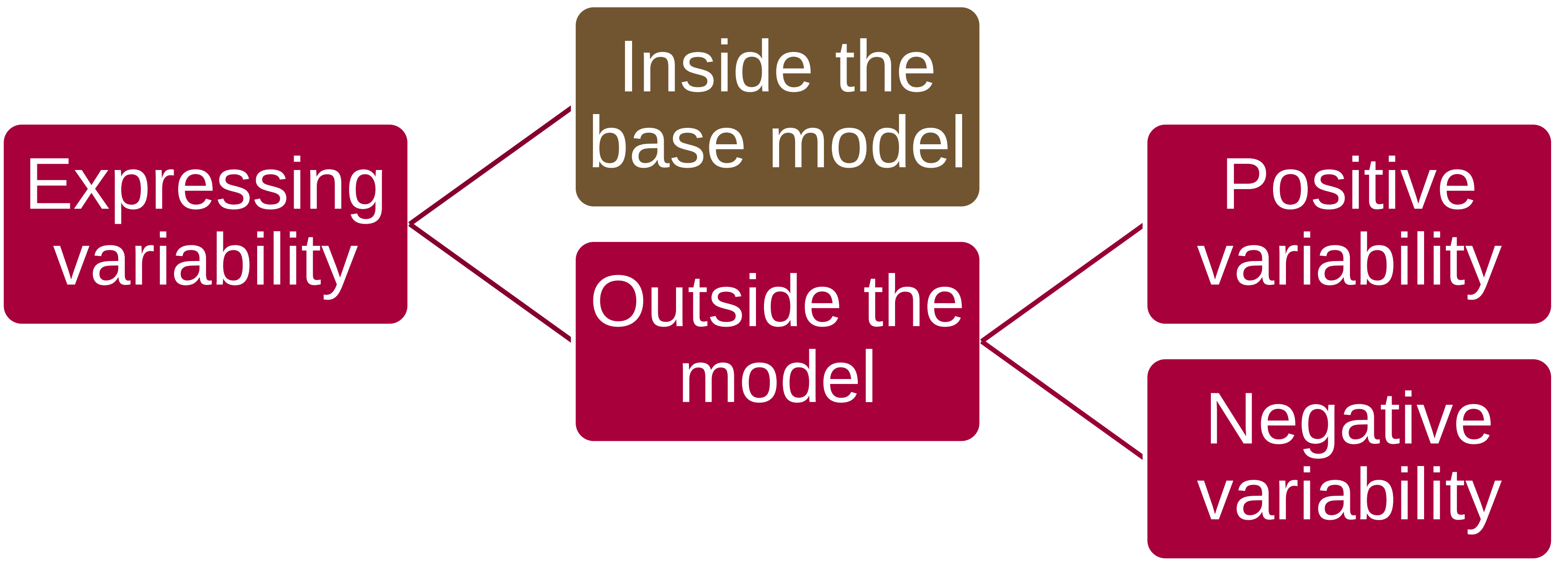
Outside the model

Variability Model

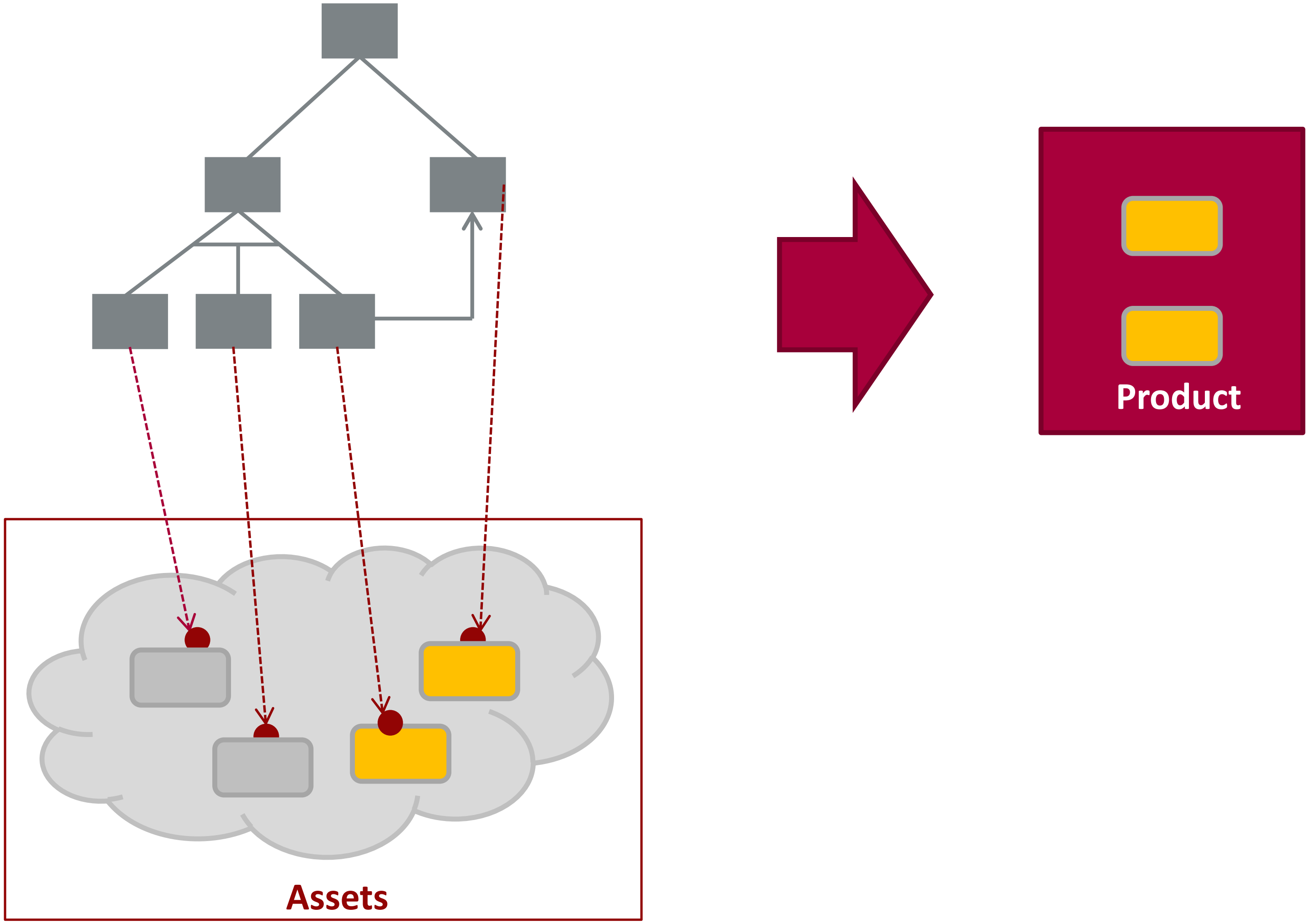


Base models

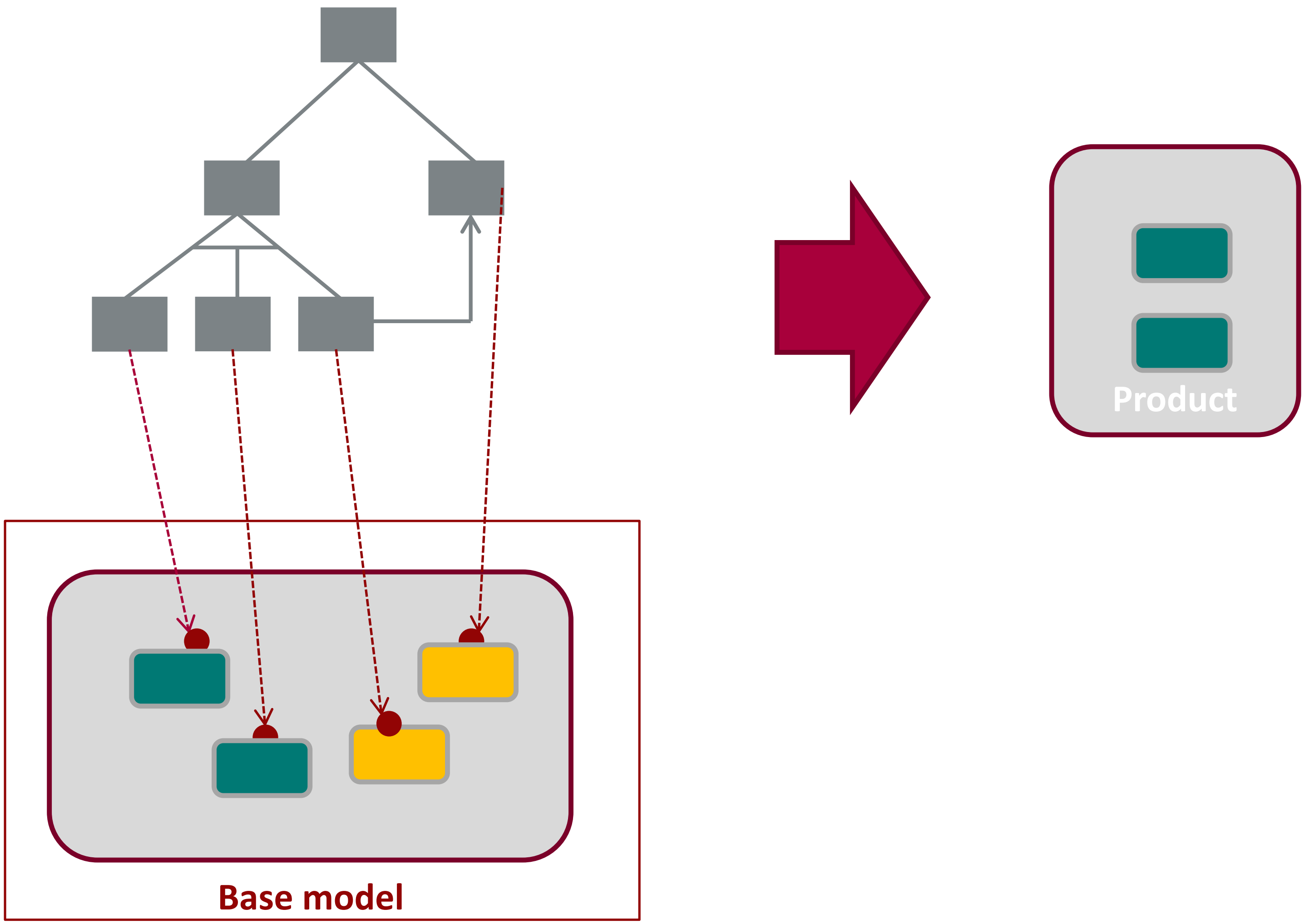
How to model variability



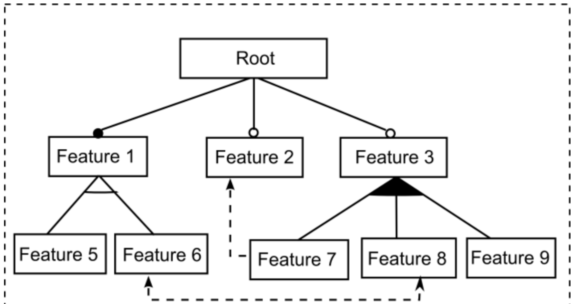
Positive variability



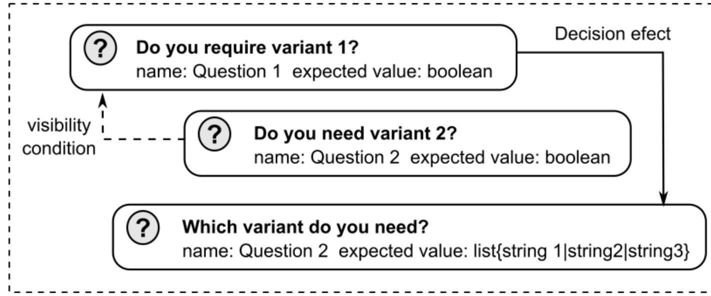
Negative variability



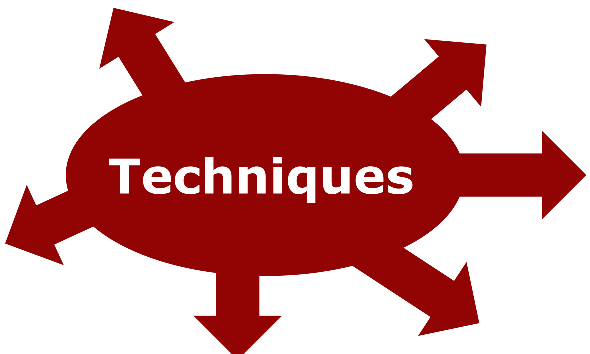
How to model variability



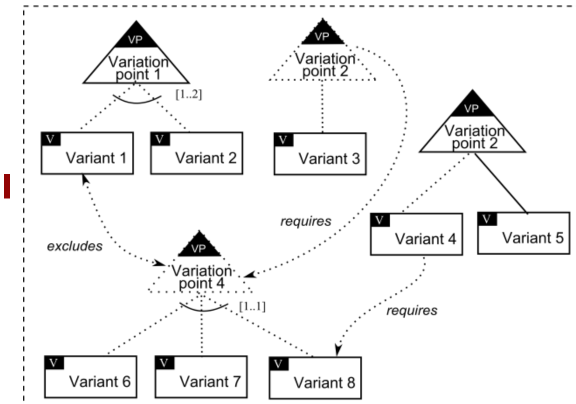
Feature modelling



Decision modelling

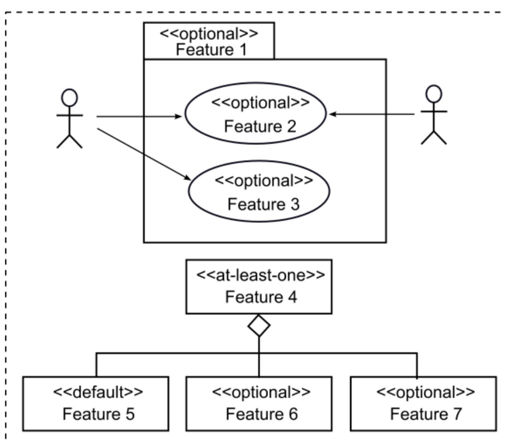


Orthogonal variability modelling

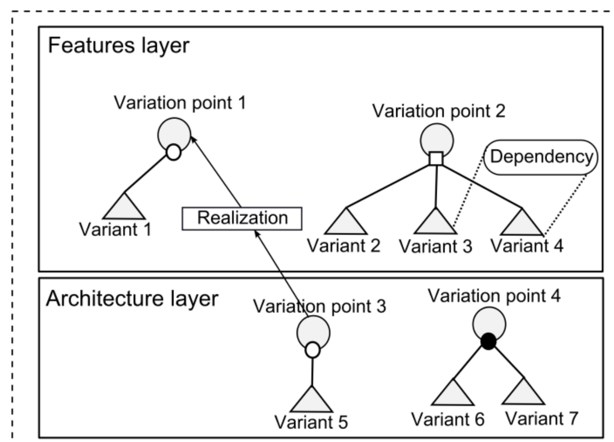


Ad-hoc solutions: tables, textual docs, …

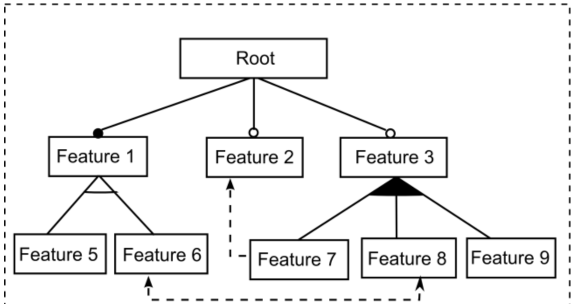
UML-based



COVAMOF



How to model variability

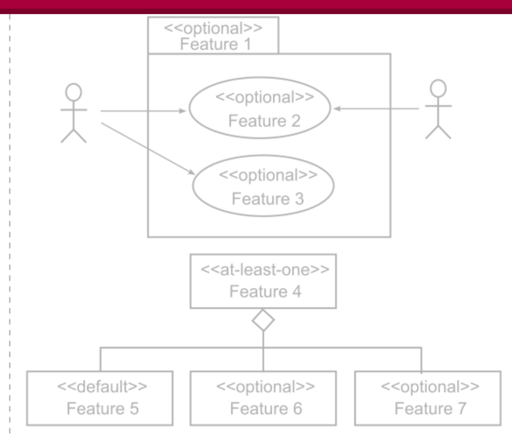


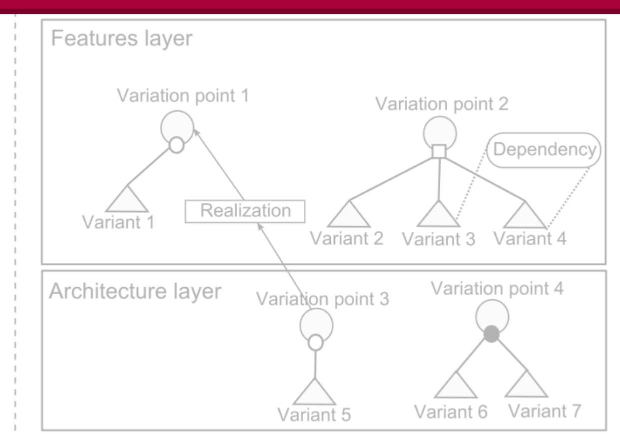
Feature modelling

Decision modelling

Ad-hoc
tables,
...

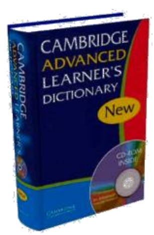
Feature models were first introduced by Kang et al. in 1990





How to specify a particular product?

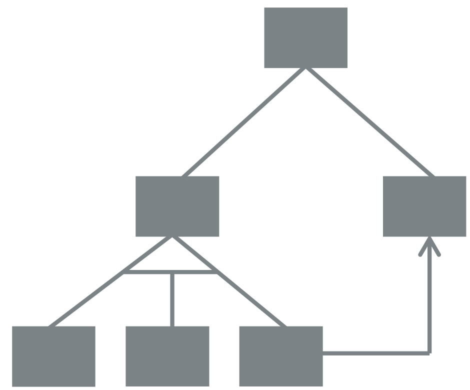
FEATURE



"An important part of something"

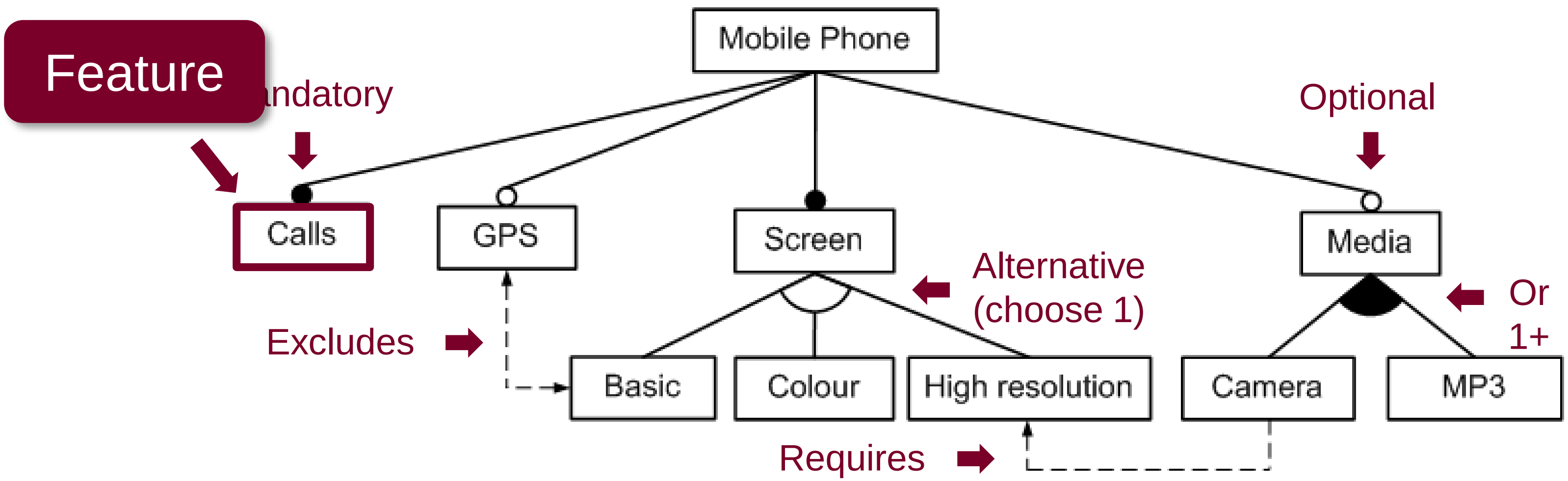
"A prominent or distinctive characteristic of a software system"

How to specify an SPL?



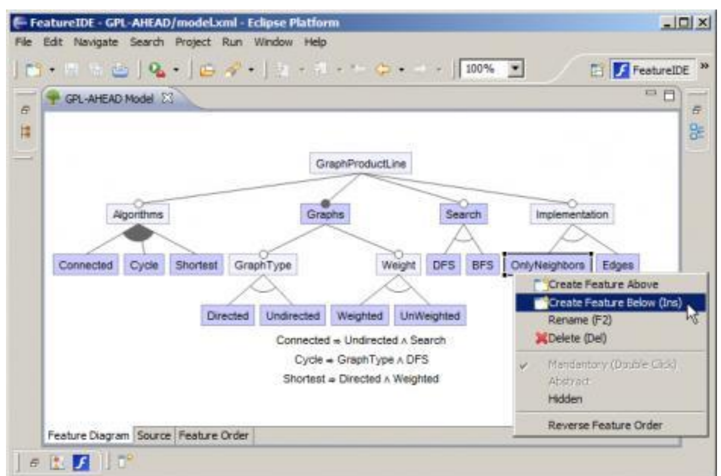
"Feature Model: A hierarchically arranged set of features to represent all possible products of an SPL"

Feature models

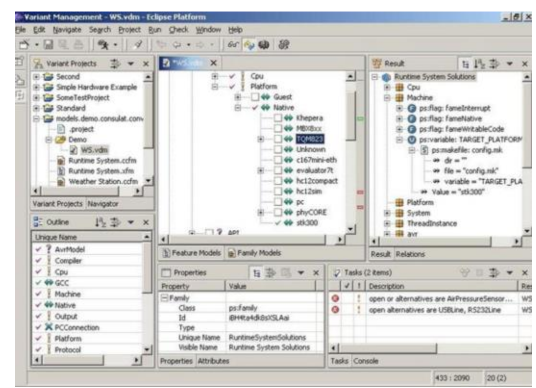


pure·systems

Feature IDE



BigLever Software, Inc.



Formal methods

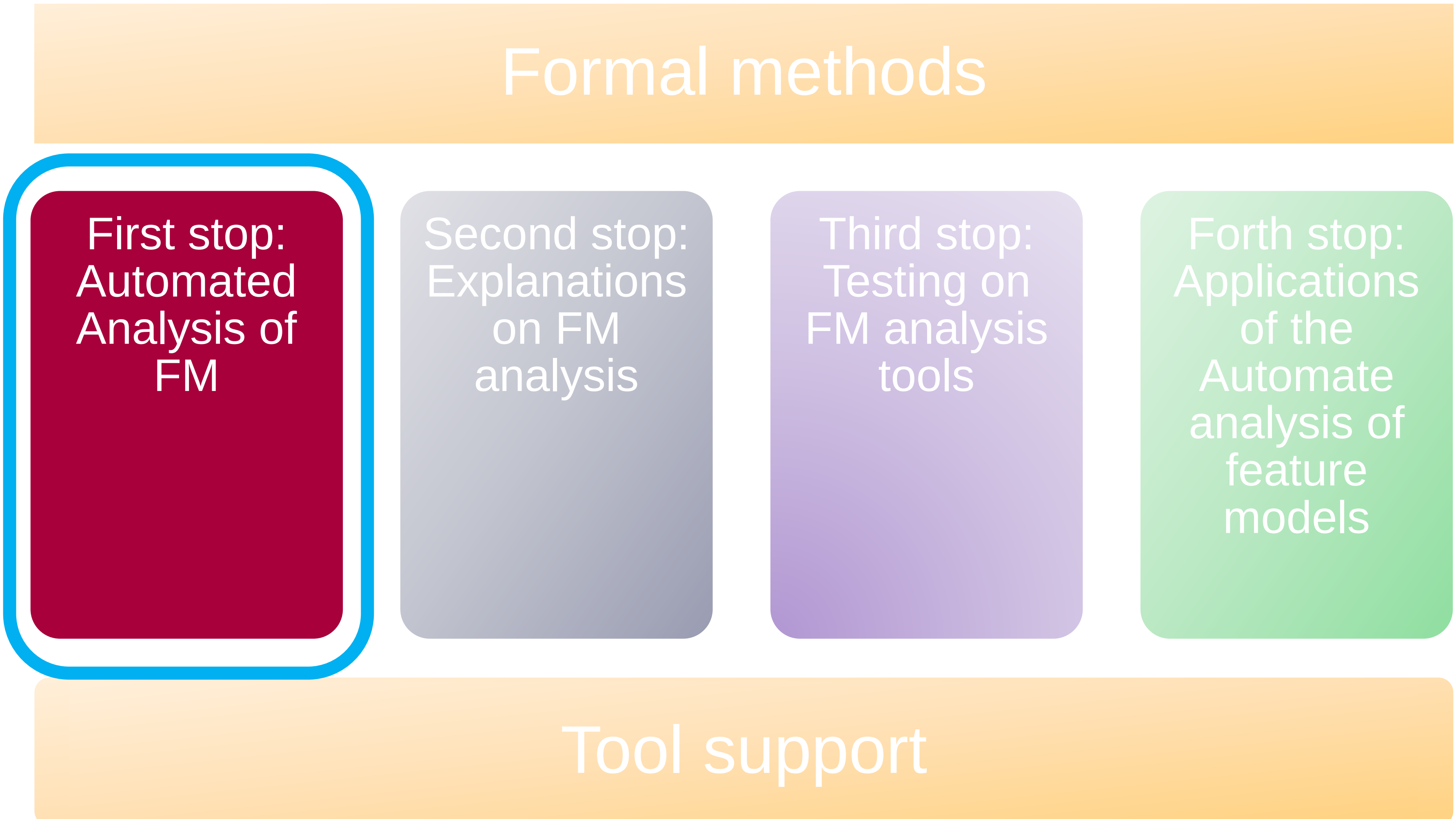
First stop: Automated Analysis of FM

Second stop: Explanations on FM analysis

Third stop: Testing on FM analysis tools

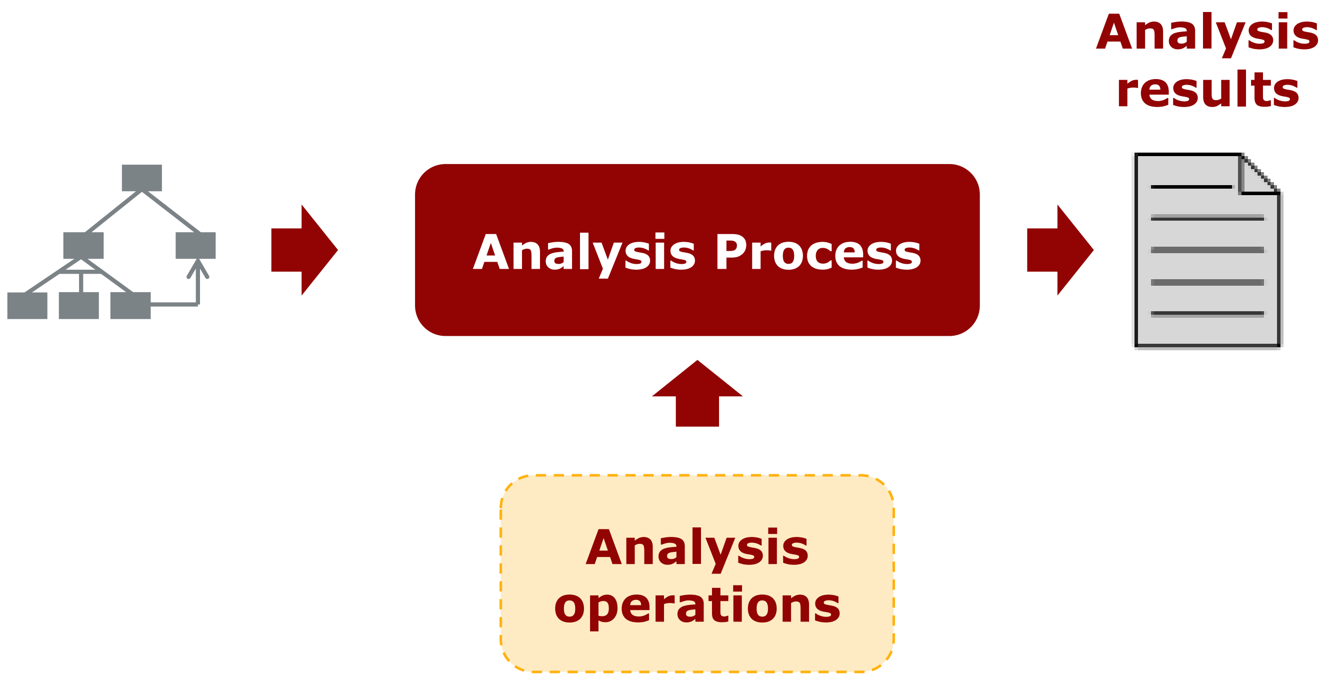
Forth stop: Applications of the Automate analysis of feature models

Tool support

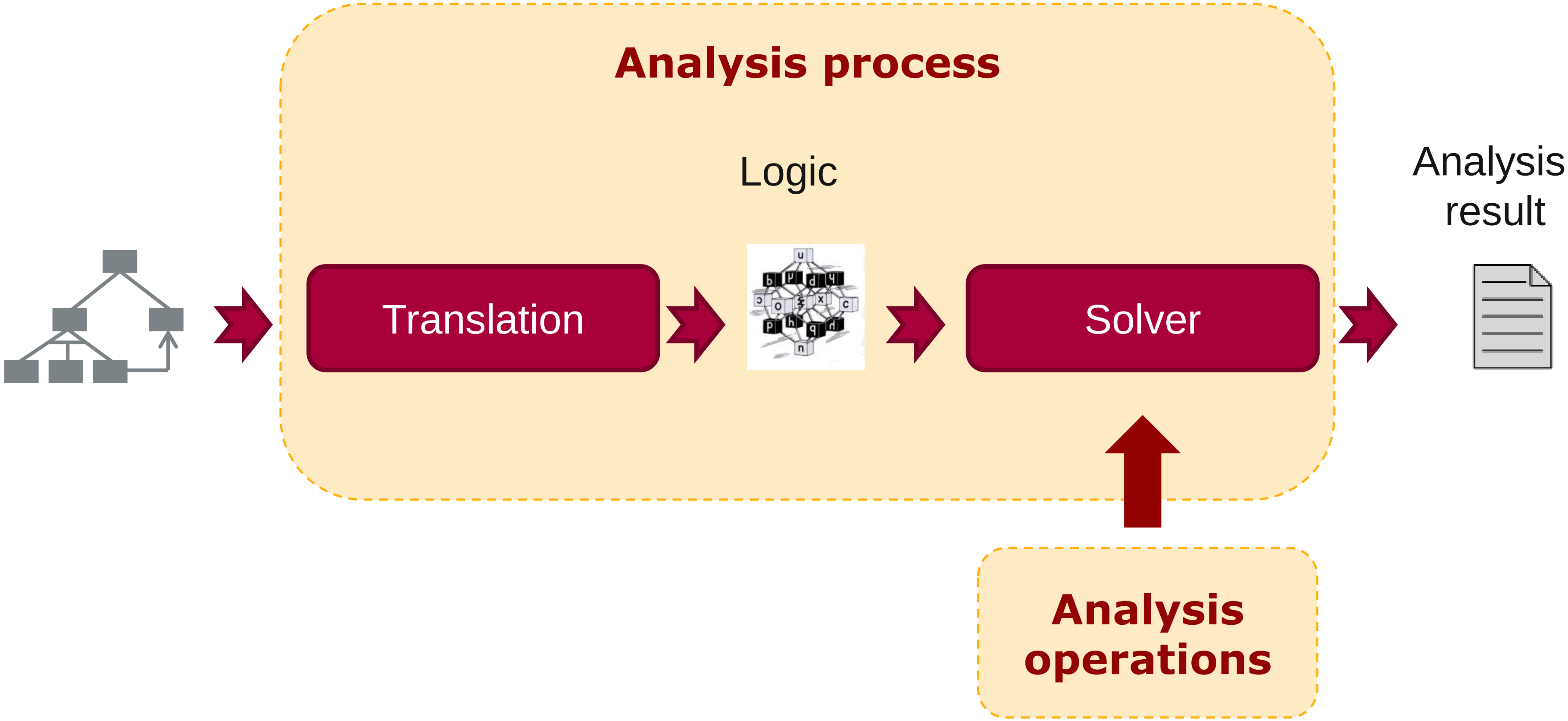

Formal methods
First stop: Automated Analysis of FM
Second stop: Explanations on FM analysis
Third stop: Testing on FM analysis tools
Forth stop: Applications of the Automate analysis of feature models
Tool support

Challenge 1: Automated analysis of Feature Models

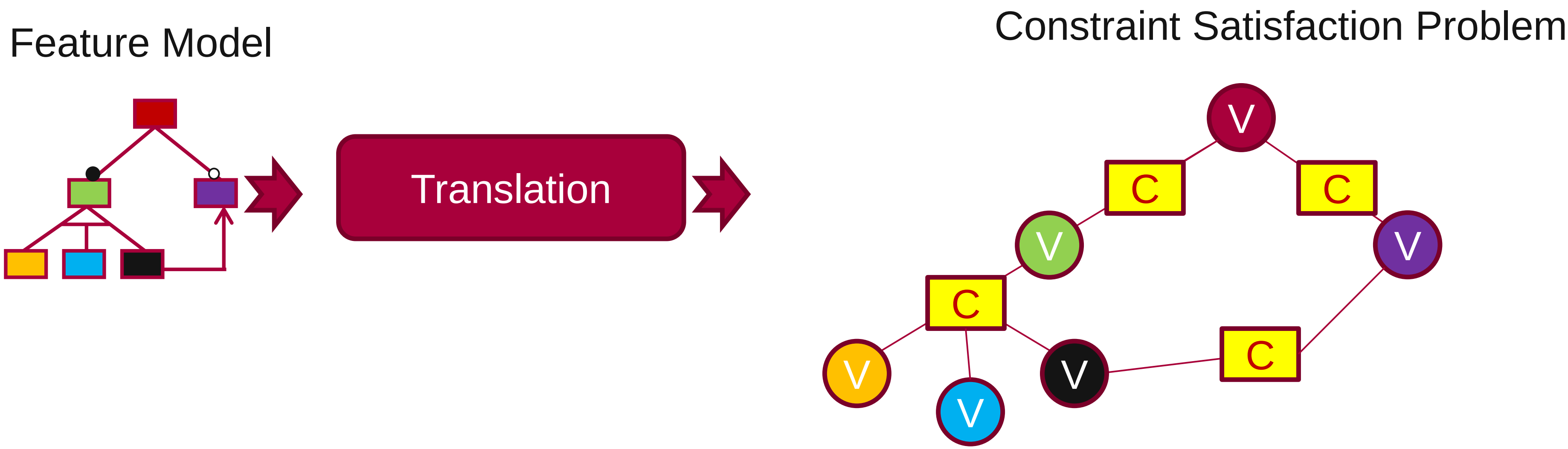
Computer-aided, extraction of useful information from feature models



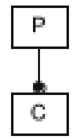
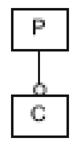
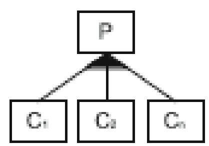
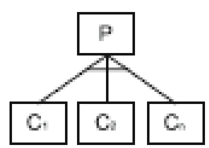
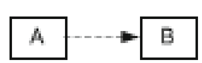
Analysis process



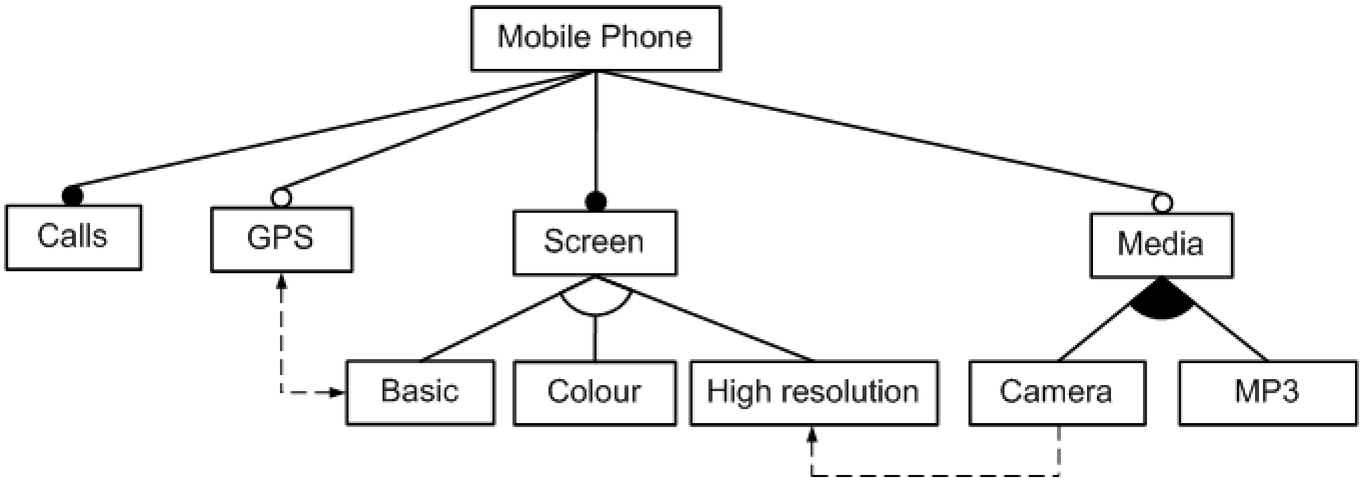
Feature models as CSPs



Feature models as Propositional formulas

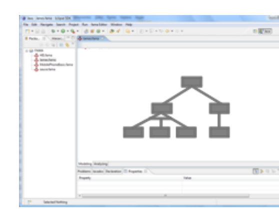
Relationship		PL Mapping
MANDATORY		$P \leftrightarrow C$
OPTIONAL		$C \rightarrow P$
OR		$P \leftrightarrow (C_1 \vee C_2 \vee \ldots \vee C_n)$
ALTERNATIVE		$(C_1 \leftrightarrow (\neg C_2 \wedge \ldots \wedge \neg C_n \wedge P)) \wedge$ $(C_2 \leftrightarrow (\neg C_1 \wedge \ldots \wedge \neg C_n \wedge P)) \wedge$ $(C_n \leftrightarrow (\neg C_1 \wedge \neg C_2 \wedge \ldots \wedge \neg C_{n-1} \wedge P))$
IMPLIES		$A \rightarrow B$
EXCLUDES		$\neg(A \wedge B)$

Automated analysis of feature models:
Computer-aided extraction of information from FMs

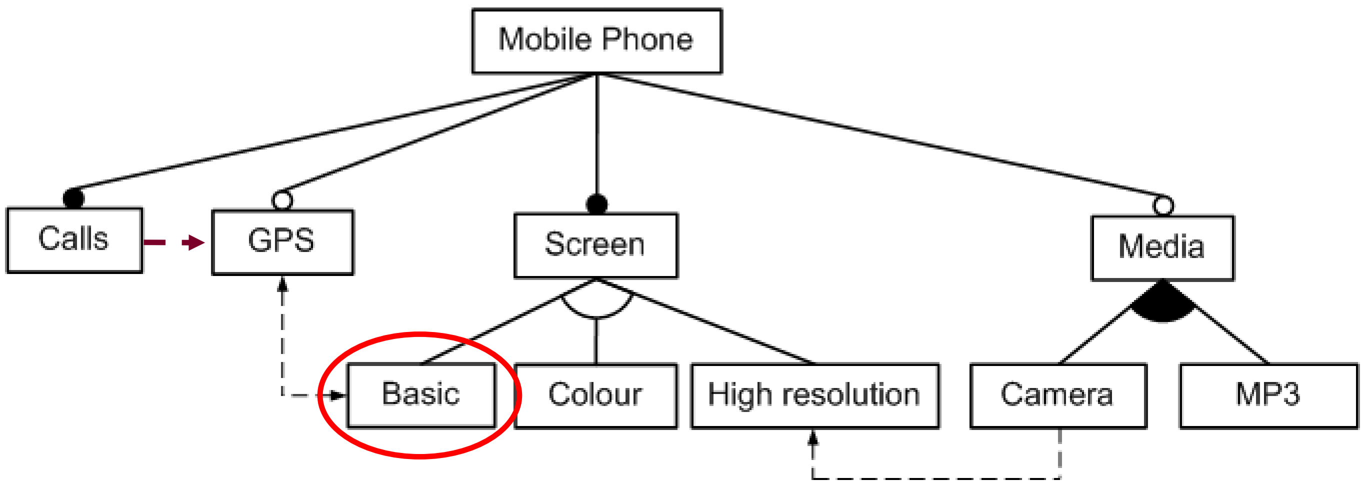


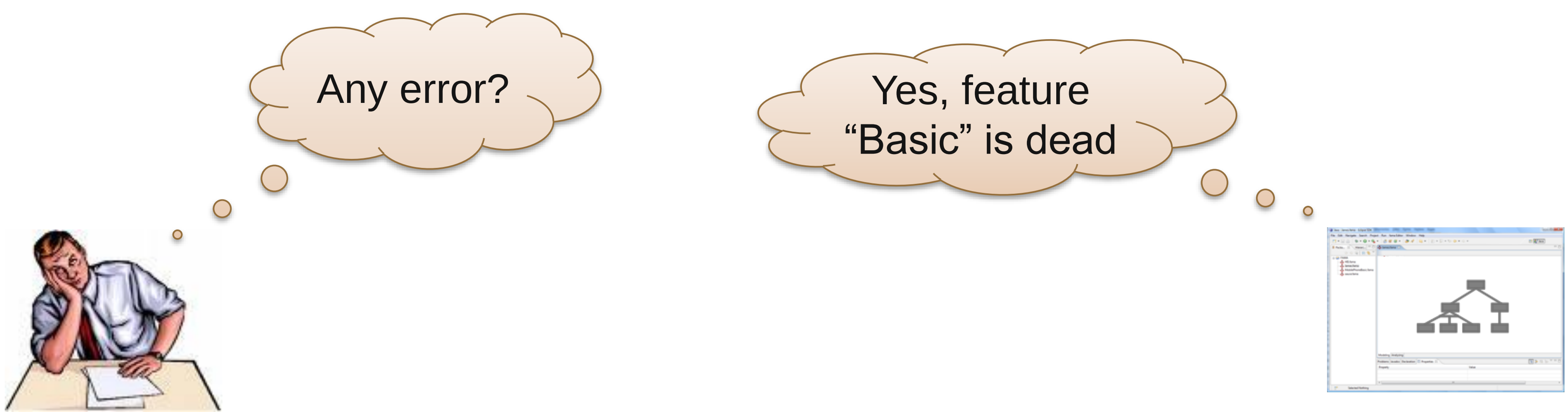
How many products?

14

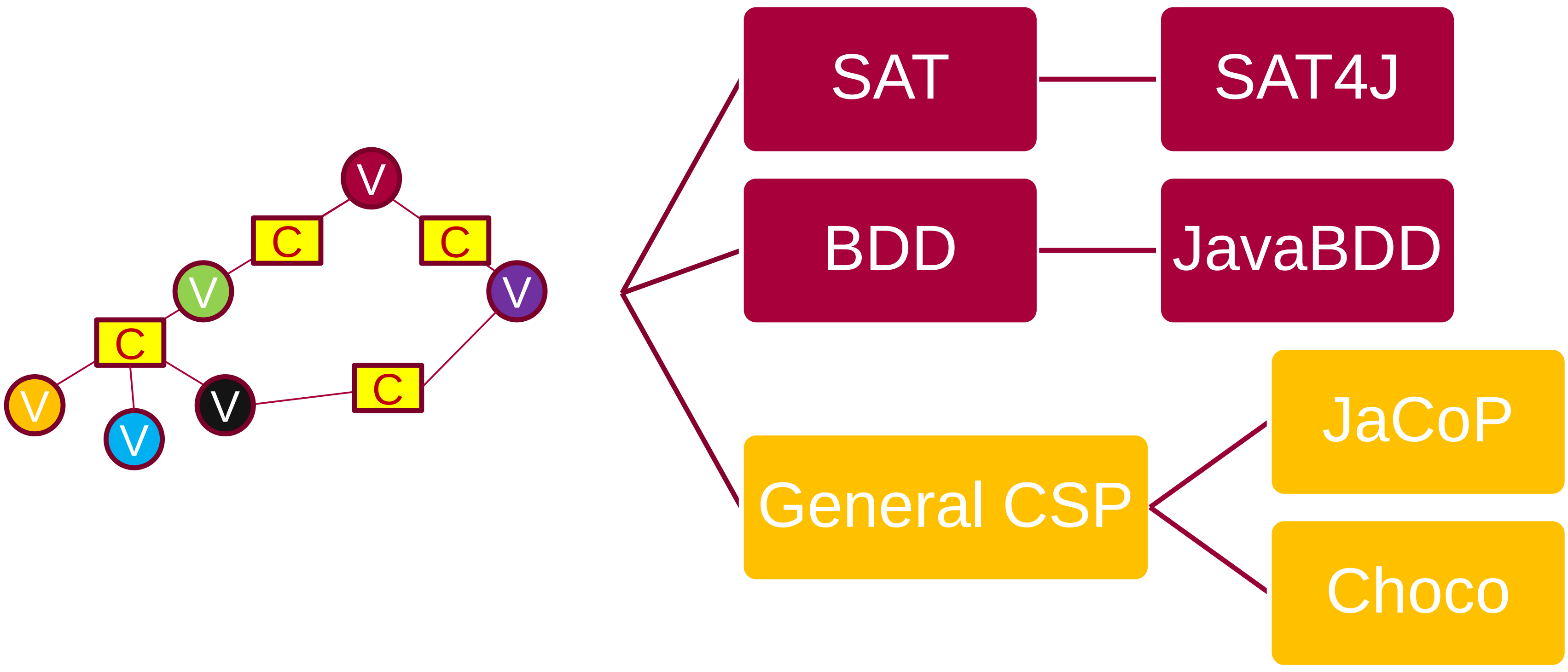


Automated analysis of feature models:
Computer-aided extraction of information from FMs

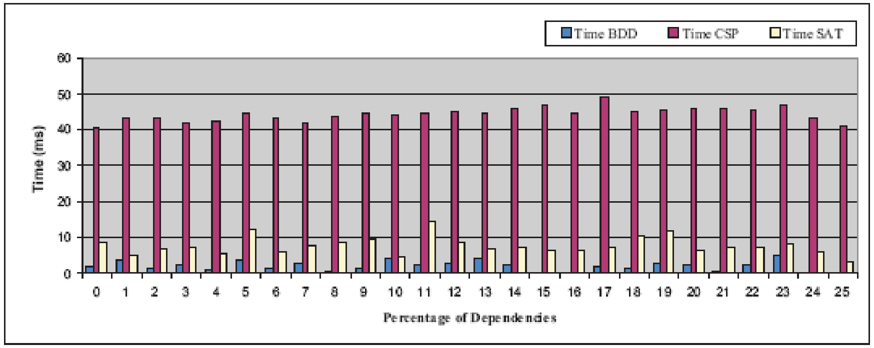


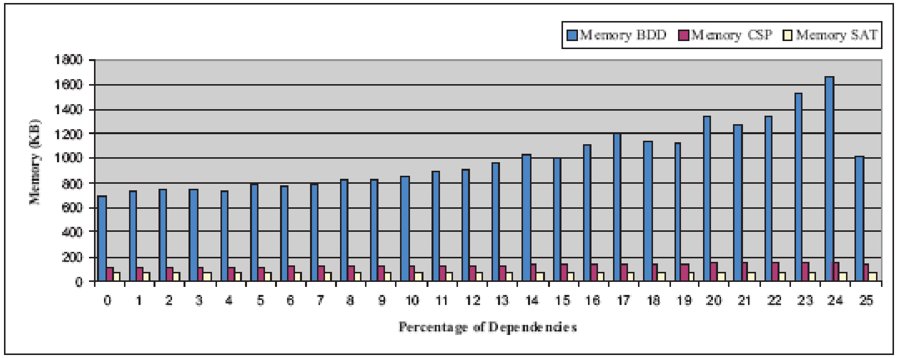


Analysis implementations



Different solvers, different performance





	BDD	[illegible]	General CSP
Memory complexity	☹	☺	☺
Time complexity	[illegible]	☺	😐
Counting solutions	[illegible]	☹	☹
Type of variables	☹	☹	☺
Advanced FM[illegible]on	☹	☹	☺

Multisolver

Automated analysis of SPL

Why it's an important problem?

Doing this by hand is an error prone task in large-scale feature models

Detecting properties at early stage of development and along all the life cycle

It's the base for other more complicated tasks, i.e. product *configuration*

Challenge 1: Automated analysis of SPL:

Computer-aided, extraction of useful information from SPL models

	Batory [5]	Czarnecki et al. [30]	Gheyi et al. [37]	Mannion et al. [51, 52]	Mendonca et al. [57]	Mendonca et al. [56]	Sun et al. [74]	Thüm et al. [75]	van der Storm [86, 87]	Zhang et al. [102, 101]	Zhang et al. [103]	Yan et al. [100]
	PL											
Void feature model	+	+		+		+	+		+	+	+	+
#Products		+		⊕								
Dead features		~				+				+	+	+
Valid product	+	+	+	+			+					
All products	+		+	⊕			+					
Explanations	+	~					+					
Refactoring			+				⊕	+				
Optimization												
Commonality												
Filter		+										
Valid partial configuration	+	+							+			
Atomic sets					+					⊕		
False optional features										⊕	+	
Corrective explanations												
Dependency analysis												
ECR					⊕	+						
Generalization			⊕					+				
Core features						+						
Variability factor												
Arbitrary edit								+				
Conditional dead features												
Homogeneity												
LCA					+							
Muti–step configuration												
Roots features					+							
Specialization								+				
Degree of orthogonality		~										
Redundancies												
Variant features												
Wrong cardinalities												
Feature model notation	B	C	B	B	B	B	B	B	B	B	C	B
Extended feature model												
Formalization			+		+	+	+	+		+	+	

	Benavides et al. [10, 11, 12]	Benavides et al. [15]	Djebbi et al. [34]	Trinidad et al. [78, 76]	White et al. [99]	White et al. [97]	Abo Zaid et al. [1]	Fan et al. [35]	Wang et al. [92, 93]	Benavides et al. [14]	Benavides et al. [16]	Segura [70]
	CP						DL			Multi		
Void feature model	+	+	+	+				+	+	+	+	+
#Products	+	+	+							+	+	+
Dead features				+			+					
Valid product					+				+			
All products	+										+	
Explanations				+			+		+			
Refactoring									+			
Optimization	⊕		+									
Commonality	⊕			+							+	
Filter	⊕		+									
Valid partial configuration												
Atomic sets												+
False optional features				+								
Corrective explanations					+							
Dependency analysis												
ECR												
Generalization												
Core features												
Variability factor	⊕											
Arbitrary edit												
Conditional dead features												
Homogeneity												
LCA												
Muti–step configuration						+						
Roots features												
Specialization												
Degree of orthogonality												
Redundancies												
Variant features												
Wrong cardinalities												
Feature model notation	B	C	C	B	B	B	B	B	B	B	C	B
Extended feature model	+		+		+	+	+					
Formalization	+			+		+		+				

	Bachmeyer et al. [4]	Cao et al. [20]	Fernandez et al. [36]	Hemakumar [41]	Gheyi et al. [38]	Kang et al. [43]	Mendonca et al. [55]	Osman et al. [59, 60]	Salinesi et al. [66]	Van den Broek et al. [84]	Van Deursen et al. [88]	Von der Massen et al. [90]	Von der Massen et al. [91]	White et al. [98, 96]
	Others													
Void feature model		+			+	⊕		+	+	+	+	+		
#Products			+							+	⊕		+	
Dead features						⊕		+	+	+				
Valid product	+					⊕						+		
All products		+								+	⊕			
Explanations						⊕		+		+				
Refactoring					+									
Optimization								+						+
Commonality			+											
Filter										+				
Valid partial configuration						⊕								
Atomic sets							+							
False optional features														
Corrective explanations								+						
Dependency analysis						⊕	+							
ECR														
Generalization														
Core features														
Variability factor														
Arbitrary edit														
Conditional dead features				+										
Homogeneity			+											
LCA														
Muti–step configuration														
Roots features														
Specialization														
Degree of orthogonality														
Redundancies														
Variant features														
Wrong cardinalities														
Feature model notation	B	B	C	B	B	B	C	C	C	B	B	B	B	B
Extended feature model						+		+						+
Formalization	+		+		+			+					+	

	Batory et al. [7]	Schobbens et al. [42, 68, 69]	Trinidad et al. [80]	Von der Massen et al. [89]
	No support			
Void feature model		~	~	
#Products			~	
Dead features	~		~	~
Valid product	~	~	~	
All products			~	
Explanations	~		~	
Refactoring		~	~	
Optimization	~		~	
Commonality			~	
Filter			~	
Valid partial configuration			~	
Atomic sets				
False optional features			~	⊖
Corrective explanations				⊖
Dependency analysis				
ECR				
Generalization				
Core features			⊖	
Variability factor			~	
Arbitrary edit				
Conditional dead features				
Homogeneity				
LCA				
Muti–step configuration				
Roots features				
Specialization				
Degree of orthogonality				
Redundancies				~
Variant features			~	
Wrong cardinalities			~	
Feature model notation	B	C	C	B
Extended feature model	+		+	
Formalization		+		

\+ Supported ~ No support ⊕ Supported(first reference) ⊖ No support (first reference) B Basic feature model C Cardinality–based feature models

Table 3: Summary of operations and support

David Benavides, Sergio Segura, Antonio Ruiz Cortés: Automated analysis of feature models 20 years later: A literature review. Inf. Syst. 35(6): 615-636 (2010)

SPL

$model : Model$ [SPL characteristic model]

$features : \mathbb{F}_1\, Feature$ [SPL feature set]

$\Phi\ model = features$

[other invariants can be added in the CML]

$\_ \prec \_ : Product \leftrightarrow SPL$

$$\forall p : Product;\ spl : SPL \bullet$$
$$p \prec spl \ \Leftrightarrow$$
$$(\, p \subseteq spl.features \ \wedge\ p \lll spl.model \,)$$

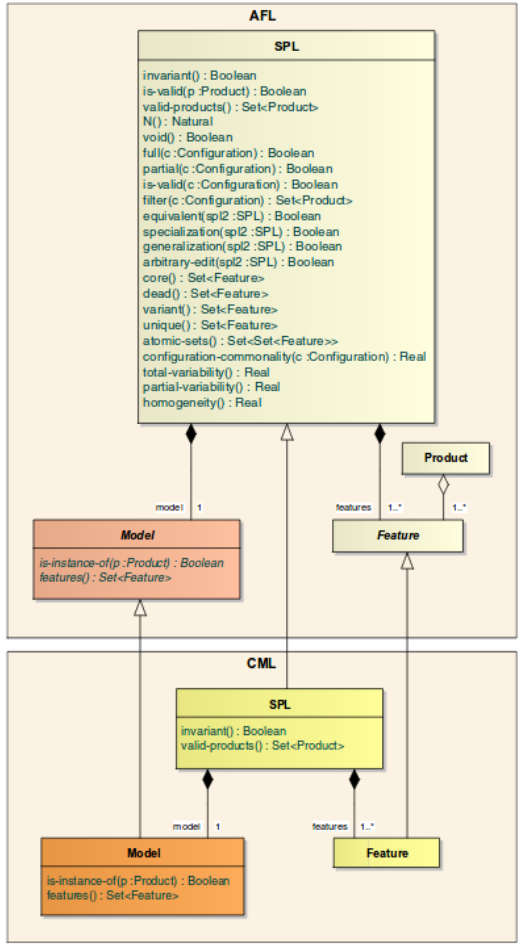
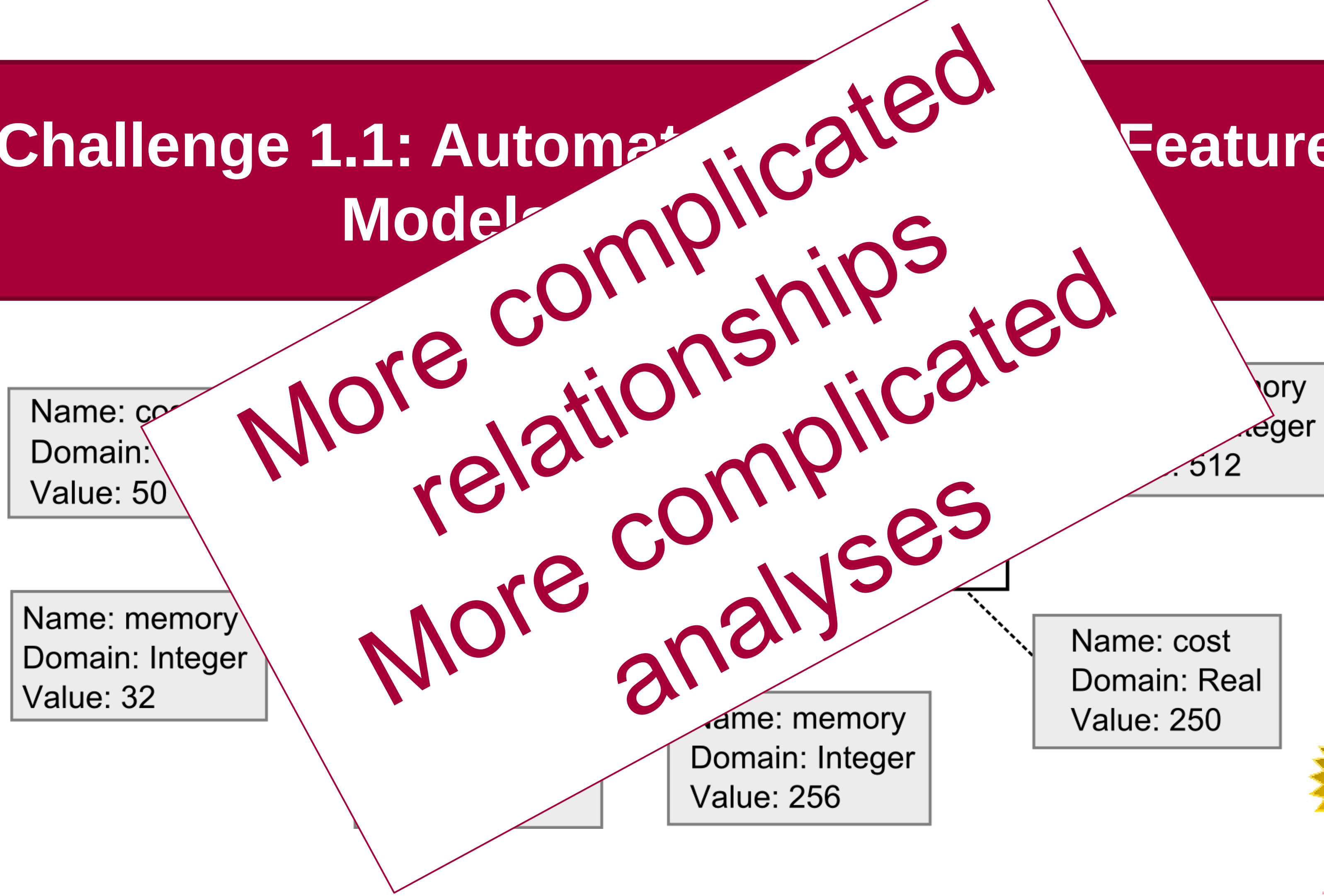


Fig. 3 UML class diagram of the FLAME architecture

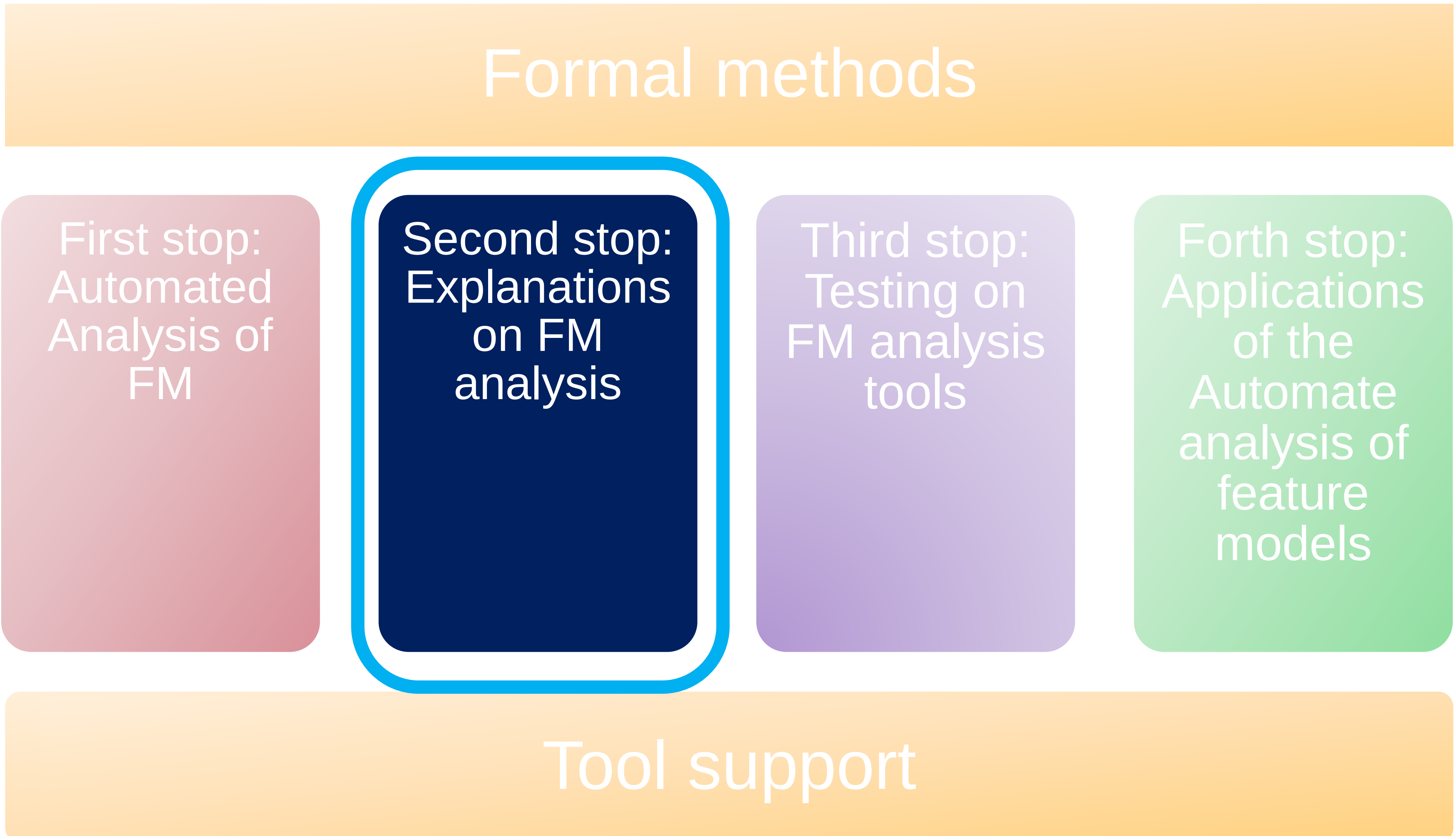
Amador Durán, David Benavides, Sergio Segura, Pablo Trinidad, Antonio Ruiz Cortés: FLAME: a formal framework for the automated analysis of software product lines validated by automated specification testing. Software and System Modeling 16(4): 1049-1082 (2017)

Are boolean feature models enough?

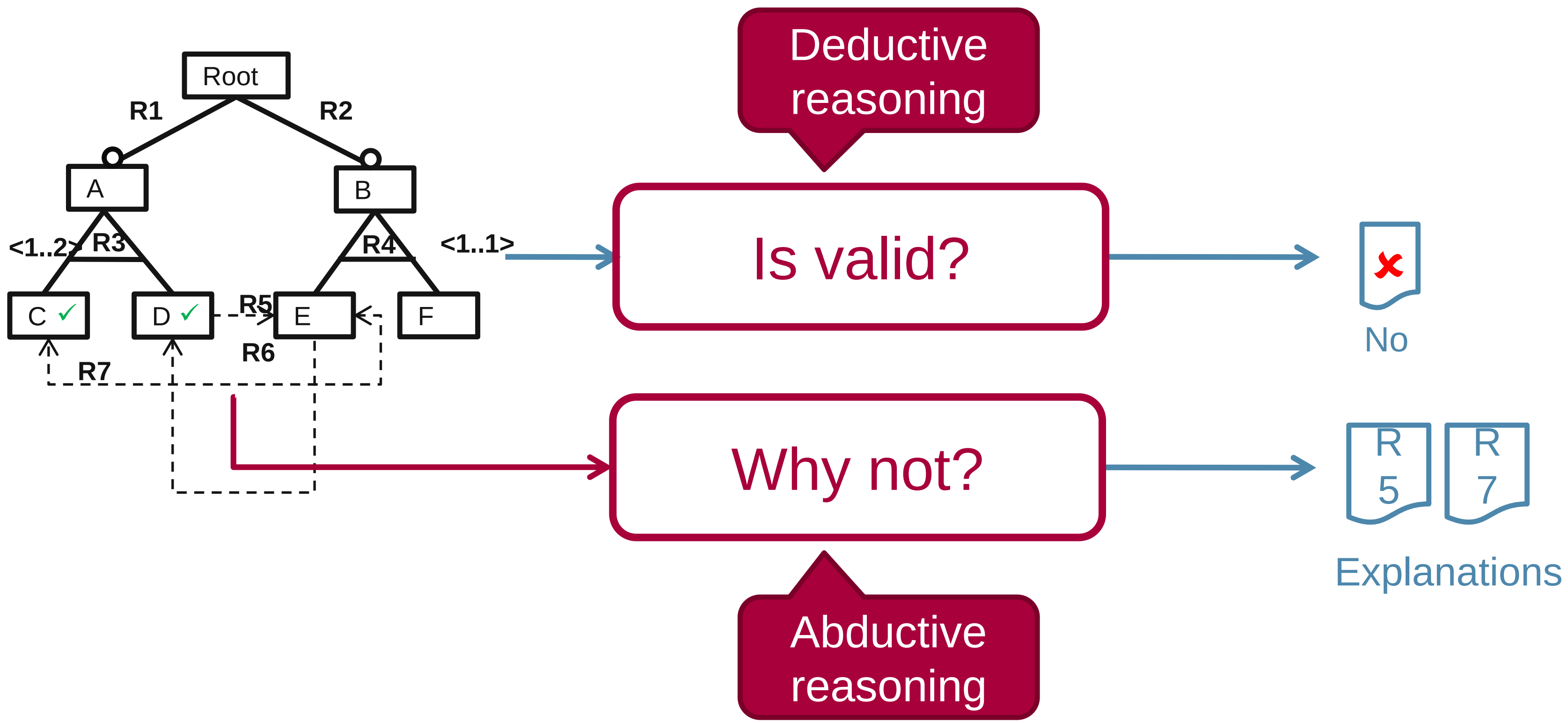
Challenge 1.1: Automat… Feature Models



- David Benavides, Pablo Trinidad Martín-Arroyo, Antonio Ruiz Cortés: Automated Reasoning on Feature Models. CAiSE 2005: 491-503
- F Roos-Frantz, D Benavides, A Ruiz-Cortés, A Heuer, K Lauenroth Quality-aware analysis in product line engineering with the orthogonal variability model. Software Quality Journal

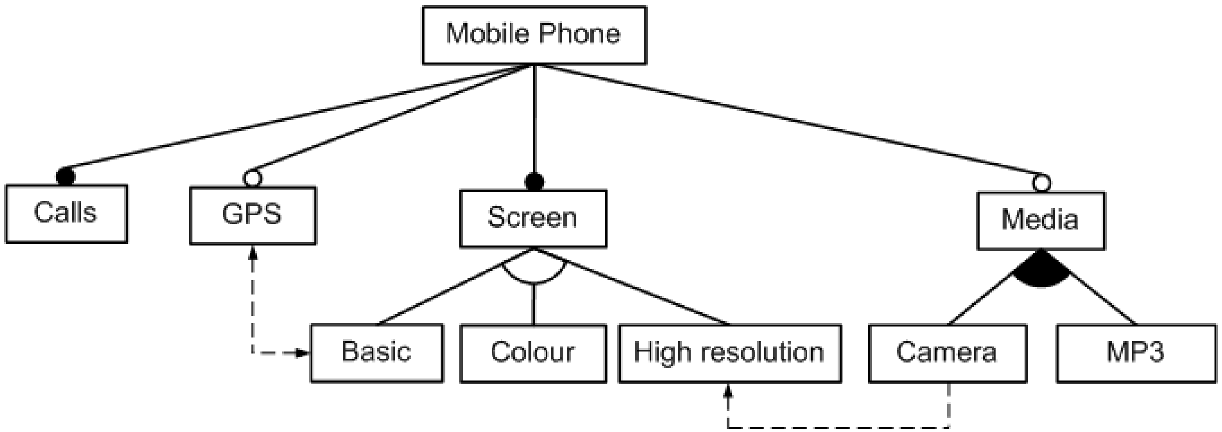

Formal methods
First stop: Automated Analysis of FM
Second stop: Explanations on FM analysis
Third stop: Testing on FM analysis tools
Forth stop: Applications of the Automate analysis of feature models
Tool support

Challenge 2: Explanations on the Automated analysis of SPL



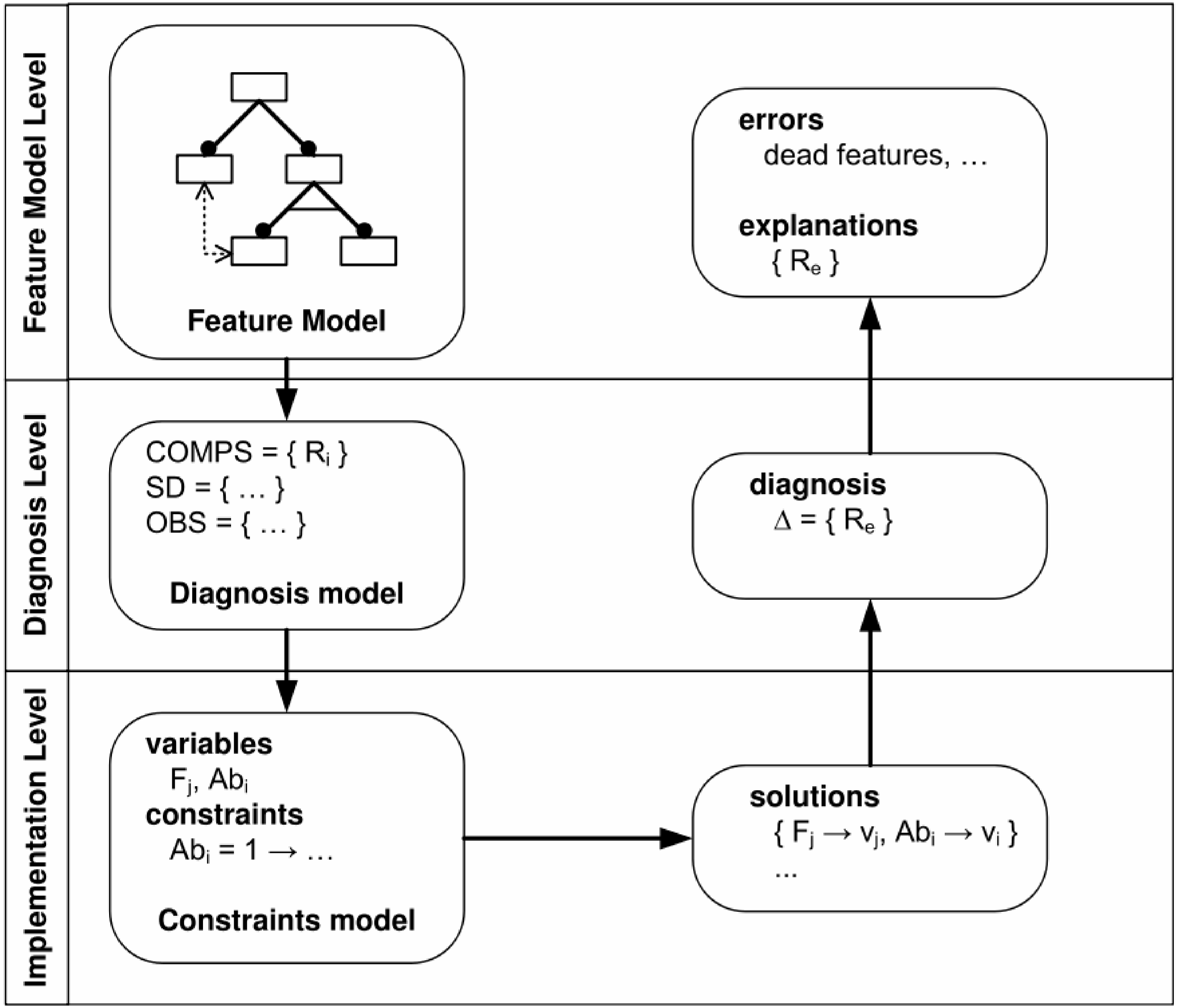
Challenge 2: Explanations on the Automated analysis of SPL

Ch 2.1 with feature models



Pablo Trinidad, David Benavides, Amador Durán, Antonio Ruiz Cortés, Miguel Toro: Automated error analysis for the agilization of feature modeling. Journal of Systems and Software 81(6): 883-896 (2008)

Ch 3.1 with feature models



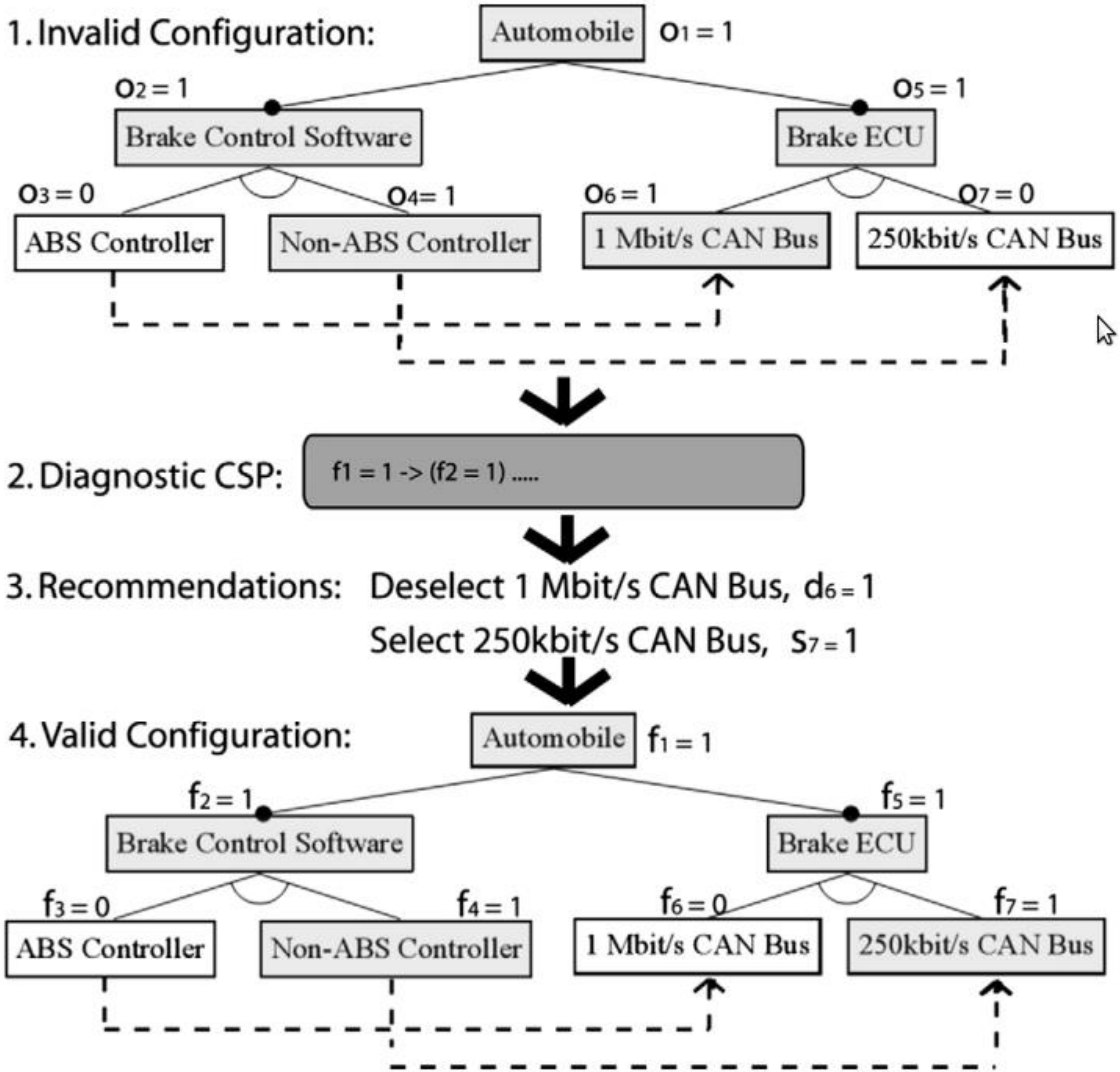
Challenge 2: Explanations on the Automated analysis of SPL

Ch 2.2 with configurations

Jules White, David Benavides, Douglas C. Schmidt, Pablo Trinidad, Brian Dougherty, Antonio Ruiz Cortés: Automated diagnosis of feature model configurations. Journal of Systems and Software 83(7): 1094-1107 (2010)

Alexander Felfernig, Rouven Walter, José A. Galindo, David Benavides, Seda Polat Erdeniz, Müslüm Atas, Stefan Reiterer: Anytime Diagnosis for Reconfiguration. J. Intell. Inf. Syst. 51(1): 161-182 (2018)

Ch 3.2 with configurations (Reiter's based)



Formal methods

First stop: Automated Analysis of FM

Second stop: Explanations on FM analysis

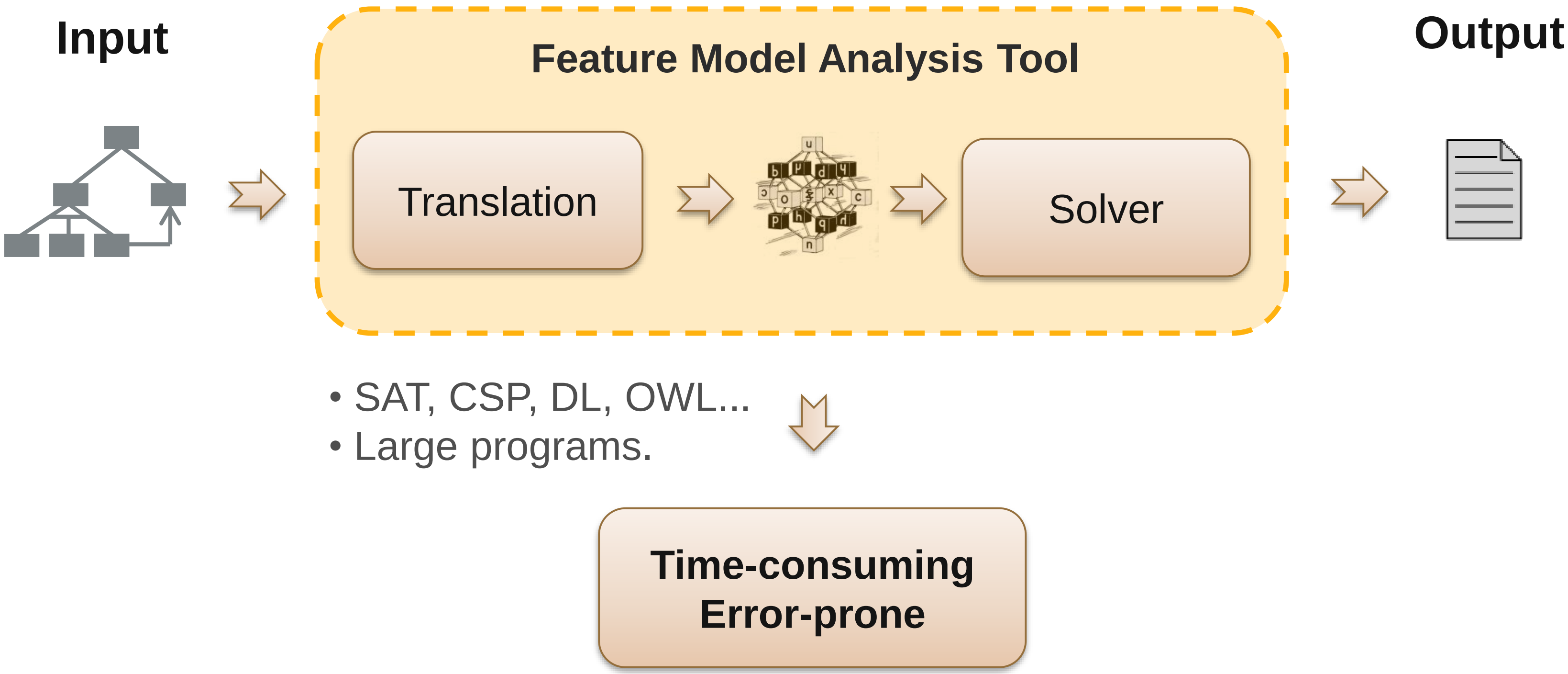
Third stop: Testing on FM analysis tools

Forth stop: Applications of the Automate analysis of feature models

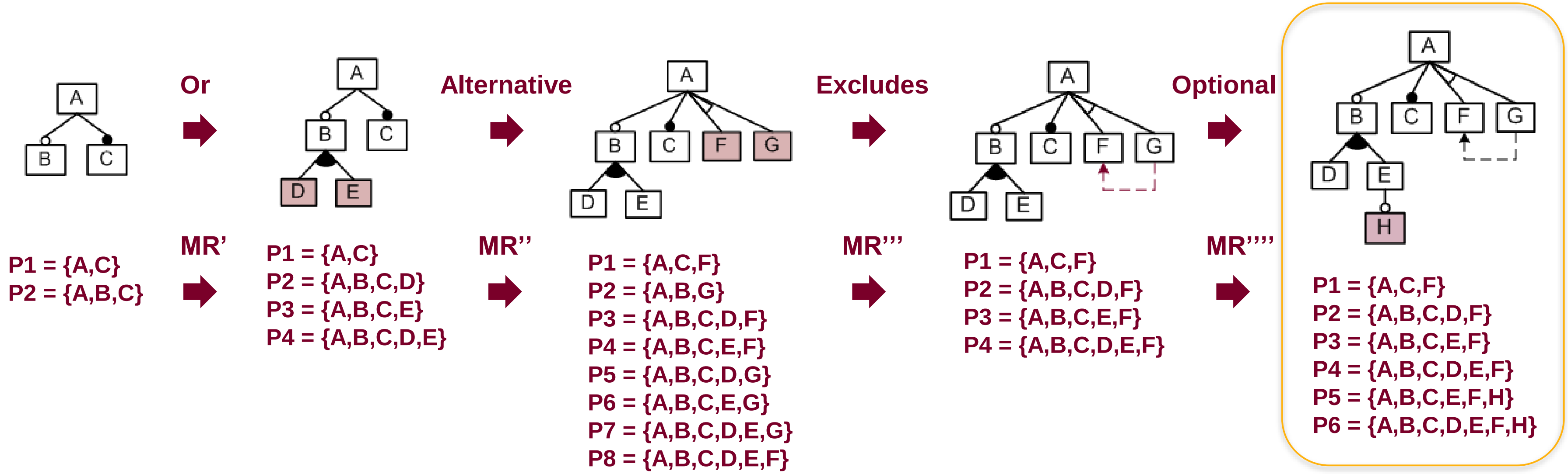
Tool support

Functional Testing

How to detect faults in feature model analysis tools?



Functional Testing



Operation

Is the model consistent?

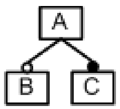
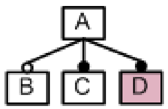
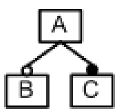
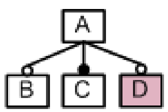
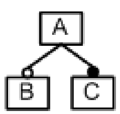
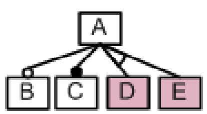
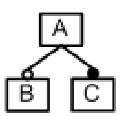
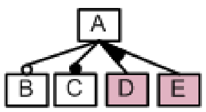
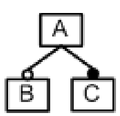
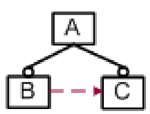
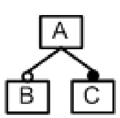
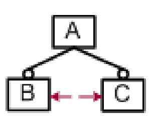
Expected output

Yes

6 products

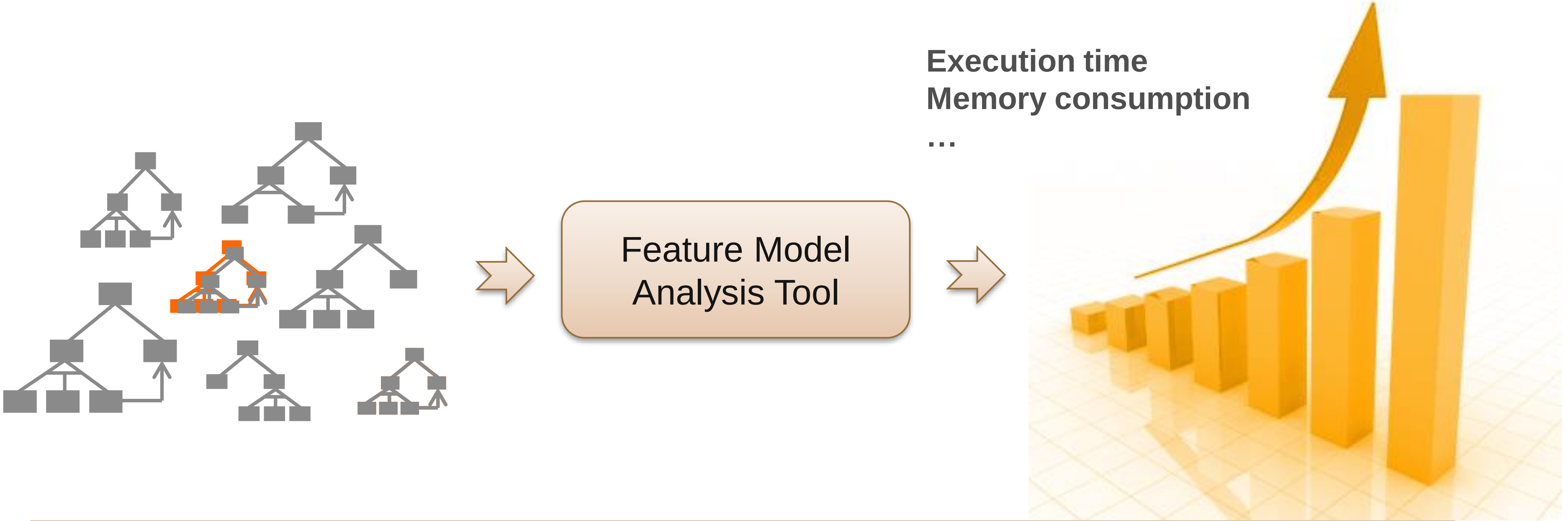
Sergio Segura, Robert M. Hierons, David Benavides, Antonio Ruiz Cortés: [Automated metamorphic testing on the analyses of feature models](). Information & Software Technology 53(3): 245-258 (2011)

Challenge 3.1: Functional Testing

	FM	FM'	Metamorphic relation
MANDATORY			$\#products(FM') = \#products(FM) \wedge \forall P'(P' \in products(FM') \Leftrightarrow \exists P \in products(FM) \cdot (pf \in features(P) \wedge P' = P \cup \{f\}) \vee (pf \notin features(P) \wedge P' = P))$
OPTIONAL			$\#products(FM') = \#products(FM) + filter(FM, \{pf\}, \phi) \wedge \forall P'(P' \in products(FM') \Leftrightarrow \exists P \in products(FM) \cdot P' = P \vee (pf \in features(P) \wedge P' = P \cup \{f\}))$
ALTERNATIVE			$\#products(FM') = \#products(FM) + (\#C - 1)\#filter(FM, \{pf\}, \phi) \wedge \forall P'(P' \in products(FM') \Leftrightarrow \exists P \in products(FM) \cdot (pf \in features(P) \wedge \exists c \in C \cdot P' = P \cup \{c\}) \vee (pf \notin features(P) \wedge P' = P))$
OR			$\#products(FM') = \#products(FM) + (2^{\#C} - 1)\#filter(FM, \{pf\}, \phi) \wedge \forall P'(P' \in products(FM') \Leftrightarrow \exists P \in products(FM) \cdot (pf \in features(P) \wedge \exists S \in \wp(C) \cdot P' = P \cup S) \vee (pf \notin features(P) \wedge P' = P)))$
REQUIRES			$products(FM') = products(FM) \setminus filter(FM, \{f\}, \{g\})$
EXCLUDES			$products(FM') = products(FM) \setminus filter(FM, \{f, g\}, \phi)$

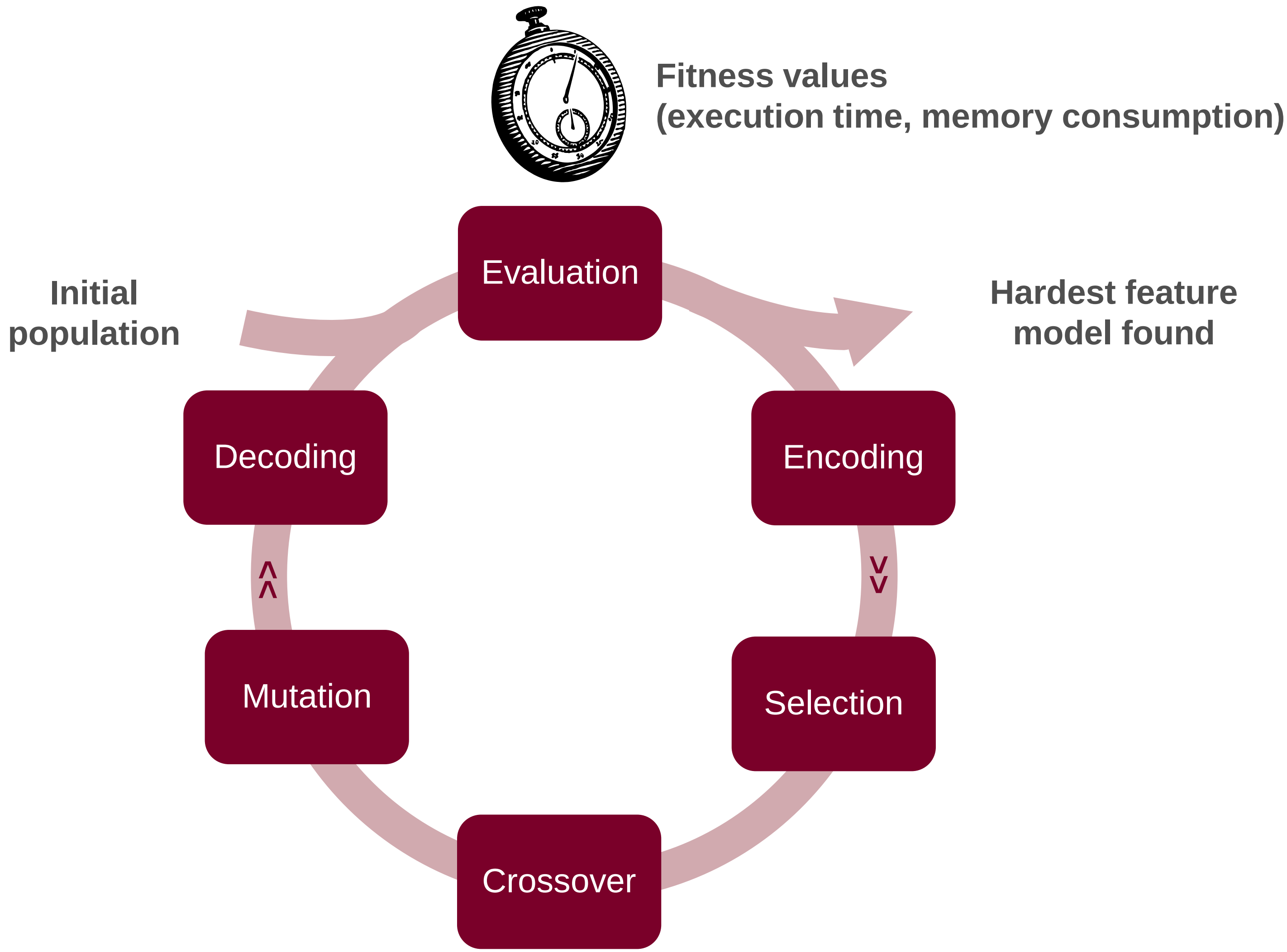
Performance Testing

How to know the performance of FM analysis tools in pessimistic cases?



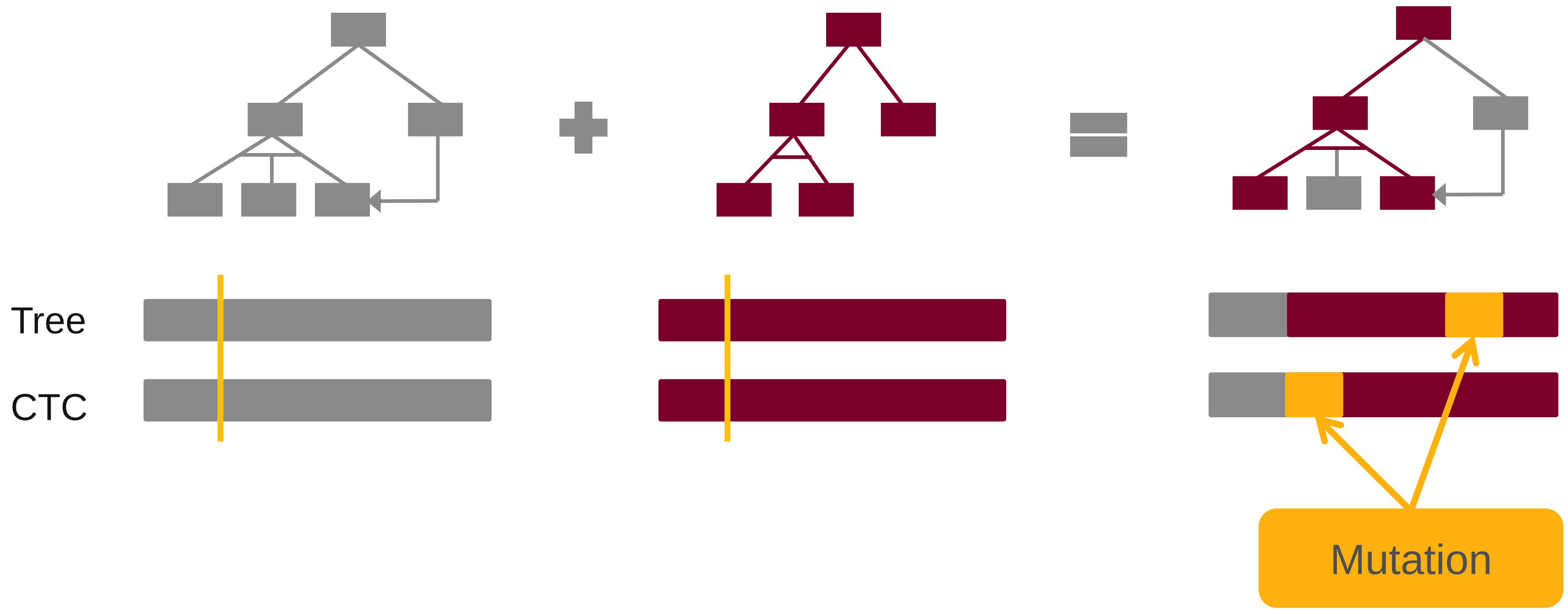
Sergio Segura, José Antonio Parejo, Robert M. Hierons, David Benavides, Antonio Ruiz Cortés: Automated generation of computationally hard feature models using evolutionary algorithms. Expert Syst. Appl. 41(8): 3975-3992 (2014)

Challenge 3.2: Performance Testing

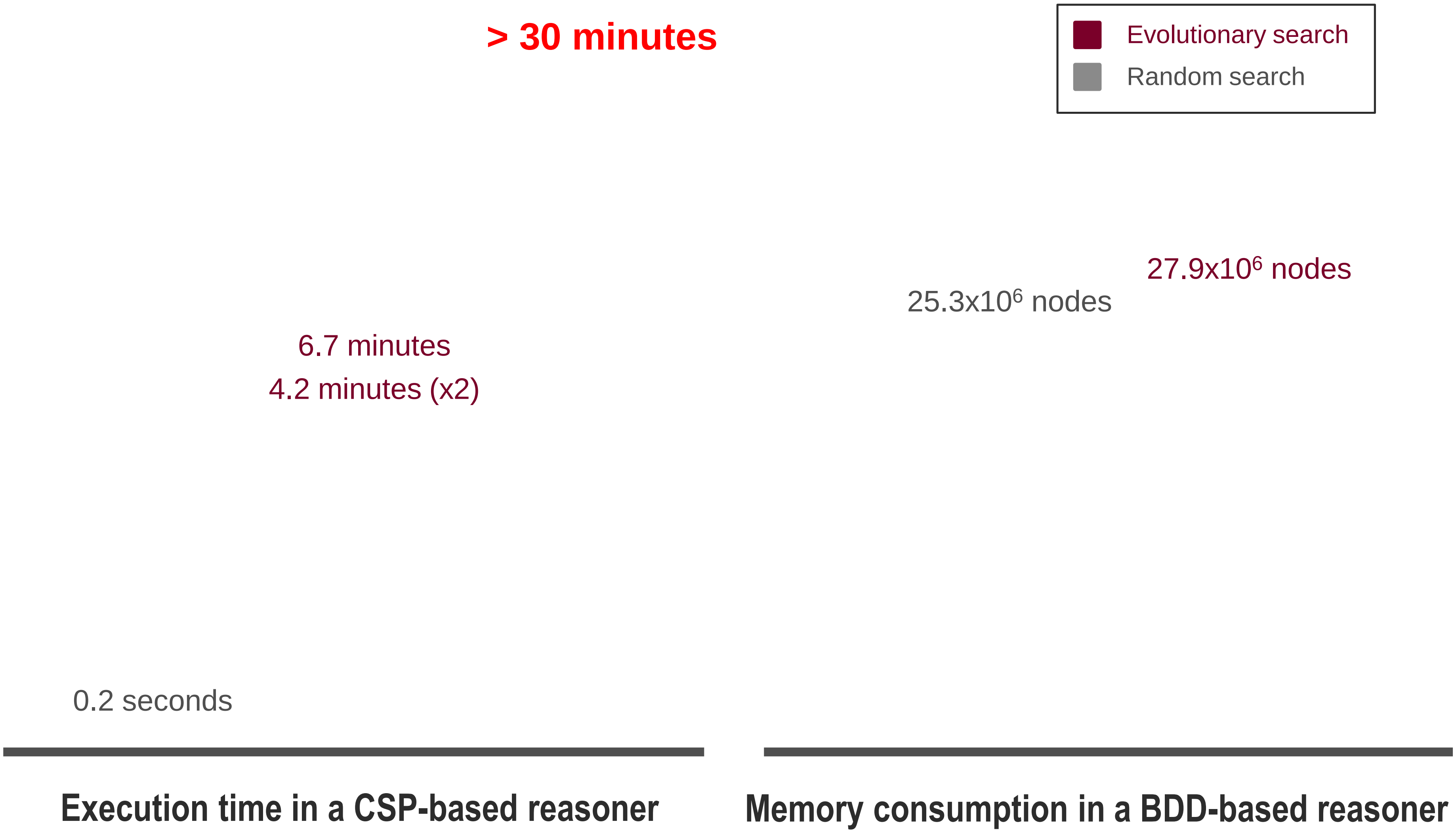


Challenge 3.2: Performance Testing

Encoding - Crossover - Mutation



Challenge 3.2: Performance Testing



But what's the application of all these???

Formal methods

First stop: Automated Analysis of FM

Second stop: Explanations on FM analysis

Third stop: Testing on FM analysis tools

Forth stop: Applications of the Automate analysis of feature models

Tool support

Computing
https://doi.org/10.1007/s00607-018-0646-1

Automated analysis of feature models: Quo vadis?

José A. Galindo[1] · David Benavides[1] · Pablo Trinidad[1] ·
Antonio-Manuel Gutiérrez-Fernández[1] · Antonio Ruiz-Cortés[1]

Received: 23 March 2017 / Accepted: 18 July 2018

Abstract

Feature models have been used since the 90s to describe software product lines as a way of reusing common parts in a family of software systems. In 2010, a systematic literature review was published summarizing the advances and settling the basis of the area of automated analysis of feature models (AAFM). From then on, different studies have applied the AAFM in different domains. In this paper, we provide an overview of the evolution of this field since 2010 by performing a systematic mapping study considering 423 primary sources. We found six different variability facets where the AAFM is being applied that define the tendencies: product configuration and derivation; testing and evolution; reverse engineering; multi-model variability-analysis; variability modelling and variability-intensive systems. We also confirmed that there is a lack of industrial evidence in most of the cases. Finally, we present where and when the papers have been published and who are the authors and institutions that are contributing to the field. We observed that the maturity is proven by the increment in the number of journals published along the years as well as the diversity of conferences and workshops where papers are published. We also suggest some synergies with other areas such as cloud or mobile computing among others that can motivate further research in the future.

Some results from the literature

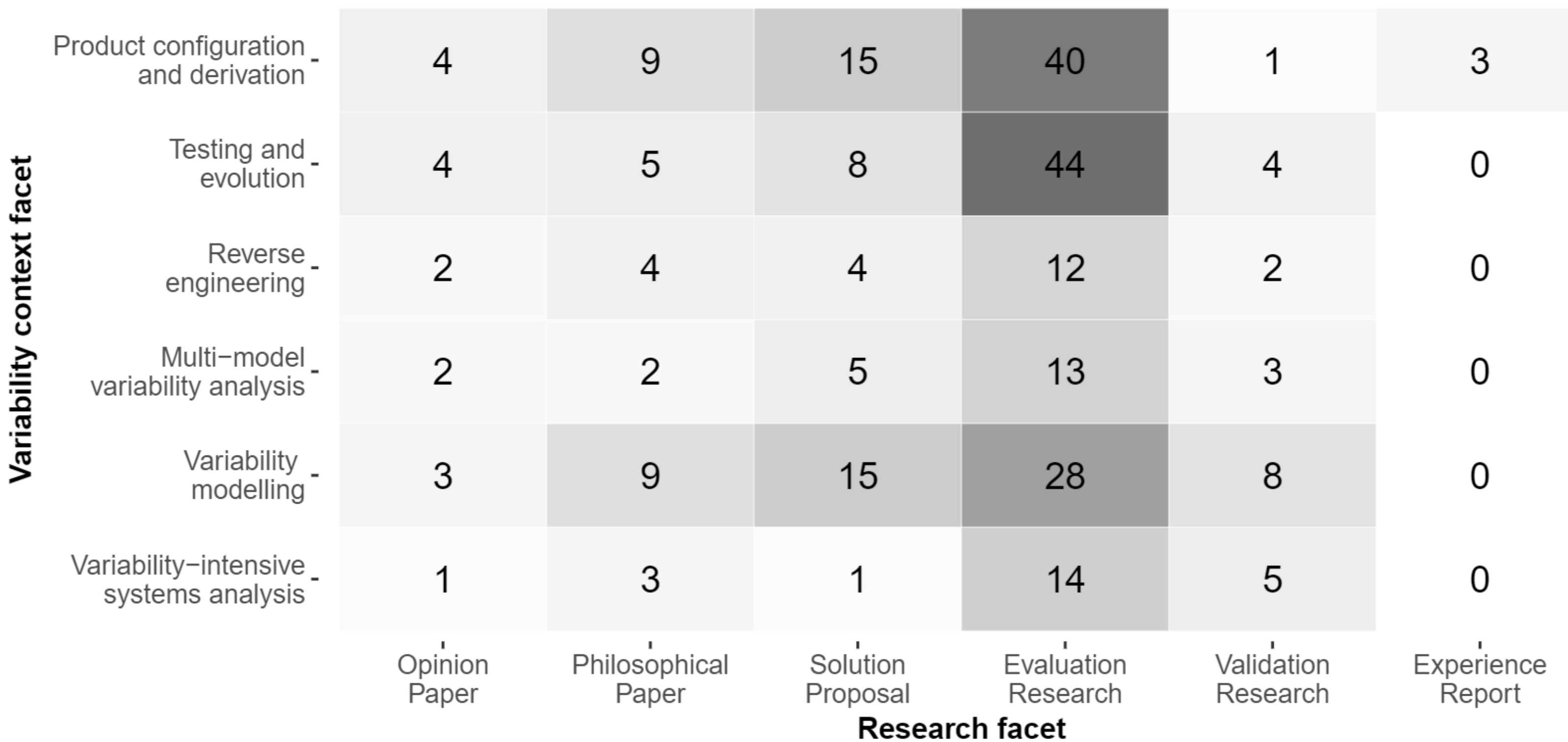
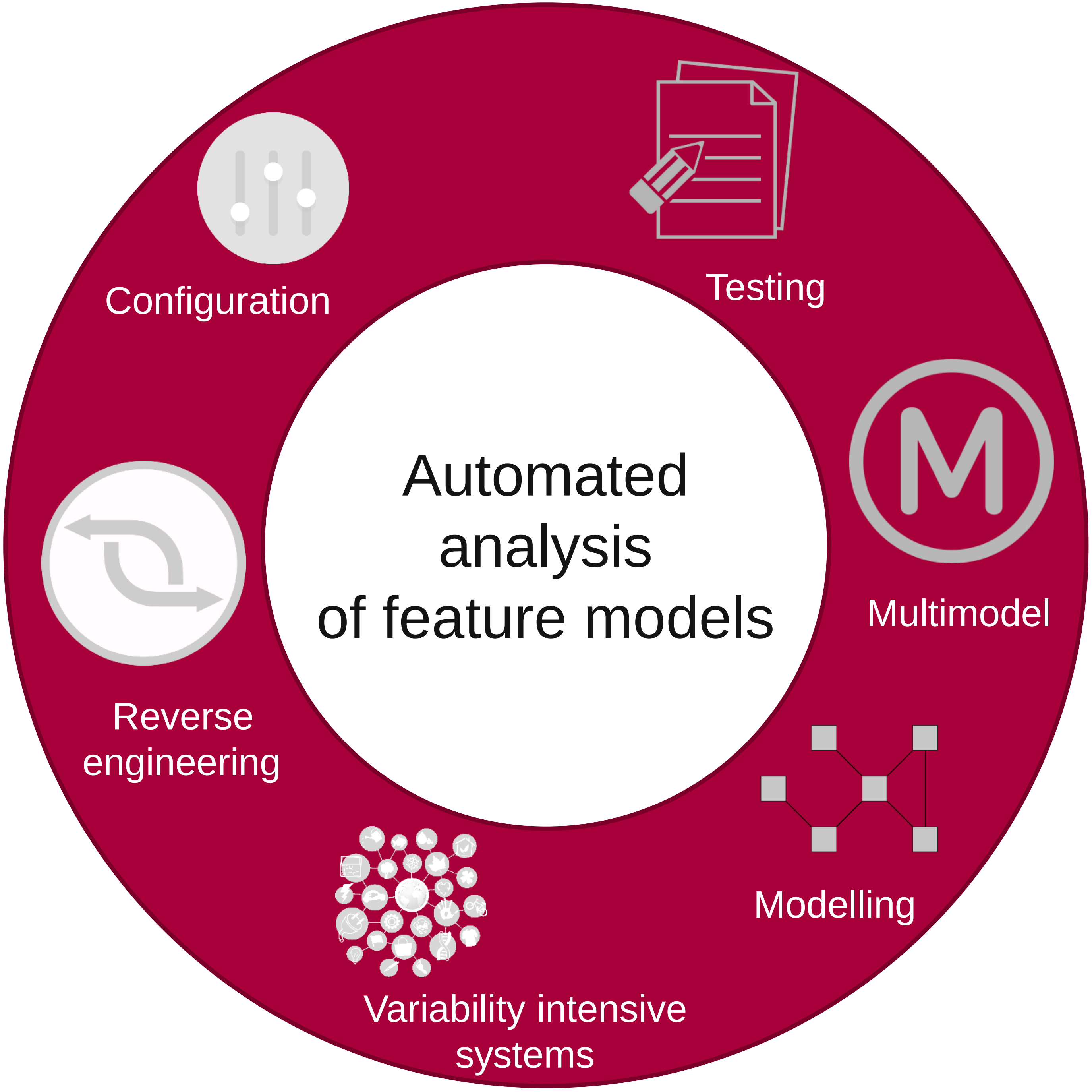


Fig. 11: Visualization of the systematic map


Automated analysis of feature models
Configuration
Testing
Multimodel
Modelling
Variability intensive systems
Reverse engineering

Formal methods

Challenge 1: Automated Analysis of FM

Challenge 2: Explanations on FM analysis

Challenge 3: Testing on FM analysis tools

Challenge 4: Applications of the Automate analysis of feature models

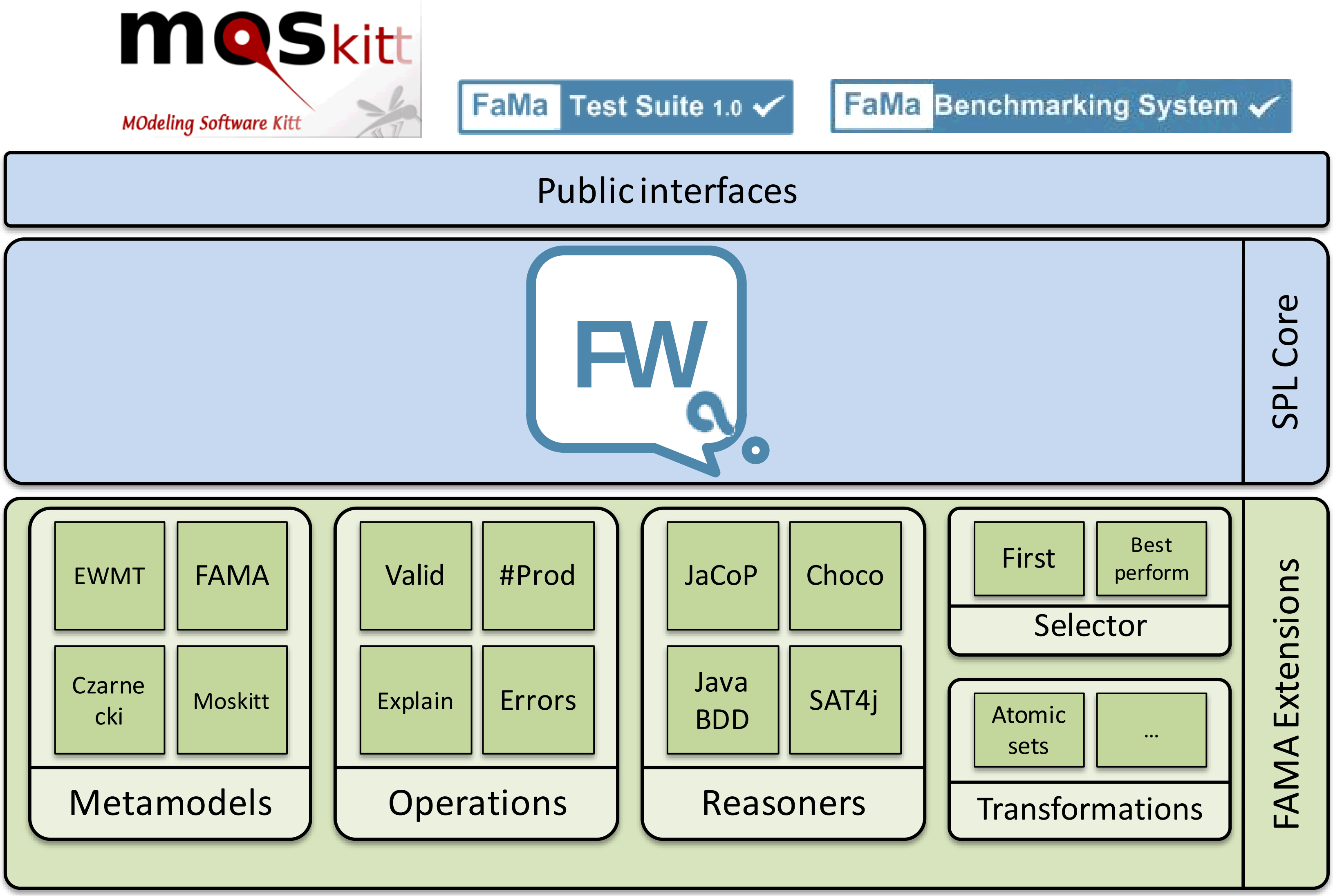
Tool support

Tooling the Automated analysis of SPL

www.isa.us.es/fama

https://github.com/isa-group/FaMA

FAMA Architecture



Tooling the Testing of FM analysis tools

Benchmarking and **T**es**T**ing on the analys**y**is of feature models

BeTTy

https://betty.services.governify.io/
https://github.com/isa-group/BeTTy

Metamorphic test data generation	Evolutionary FM generation	Random FM generation	Benchmarking support

Formal methods

First stop: Automated Analysis of FM

Second stop: Explanations on FM analysis

Third stop: Testing on FM analysis tools

Forth stop: Applications of the Automate analysis of feature models

Tool support

Intermediate stop in the journey (ICSR 2013)

Automated Analysis in Feature Modelling and Product Configuration

David Benavides[1], Alexander Felfernig[2], José A. Galindo[1], and Florian Reinfrank[2]

[1] University of Seville
Av. de la Reina Mercedes S/N, 41012 Seville, Spain
{benavides,jagalindo}@us.es
[2] Institute for Software Technology
Graz University of Technology
Inffeldgasse 16b/II
Graz, Austria
{afelfern,freinfra}@tugraz.at

SPL Team and collaborators

Feature model analysis and configuration: a 10 years journey with configuration stops

David Benavides
benavides@us.es

Configuration workshop, Graz – Sept 2018

@davbencue