Ohjelmisto-arkkitehtuurit S2015

Variability Management

Juha Tiihonen

Outline

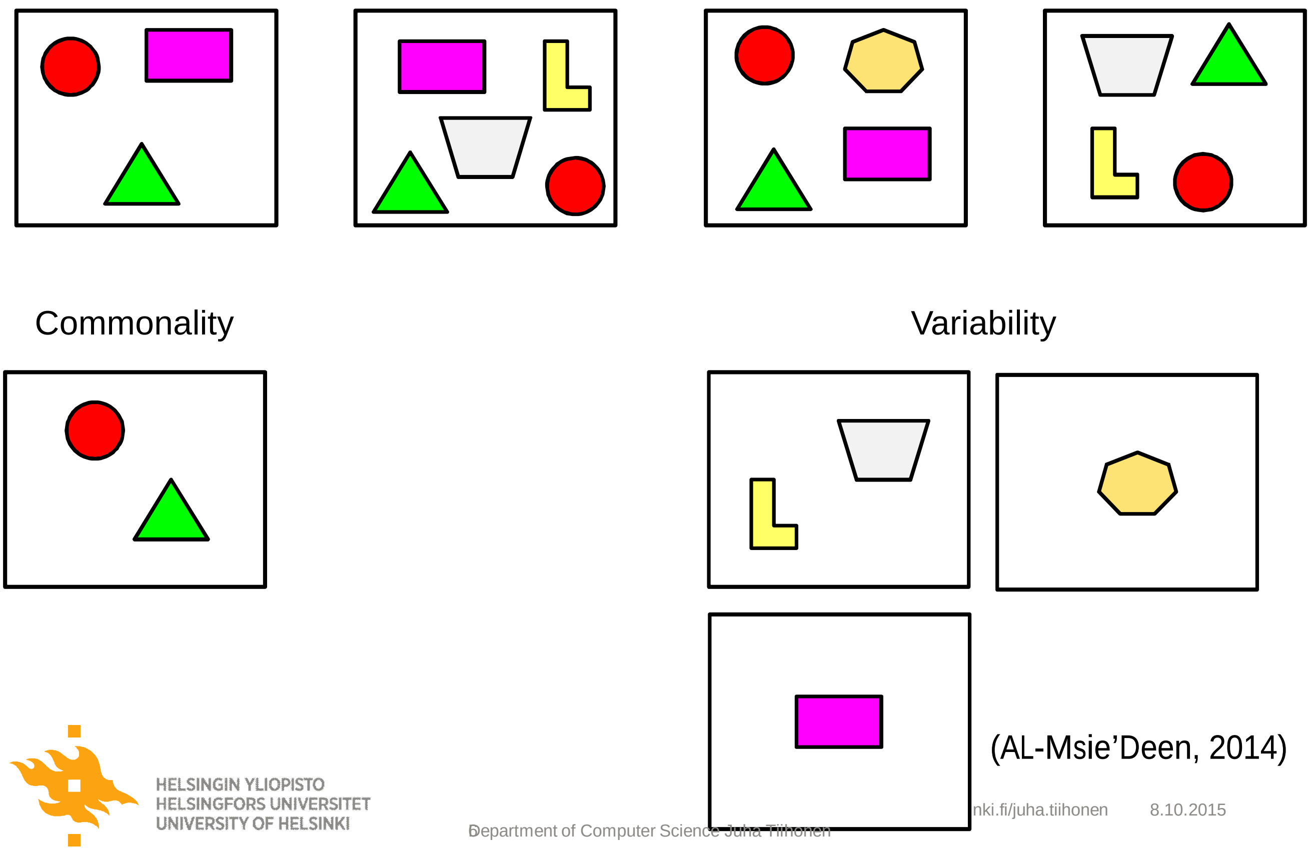
- Variability & Variability management defined
- Variability modeling, variability realization
- Tools
- Variability analysis
- Current (own) research
- Want to know more? Pointers & references

Outline

- Variability & Variability management defined
- Variability modeling, variability realization
- Tools
- Variability analysis
- Current (own) research
- Want to know more? Pointers & references

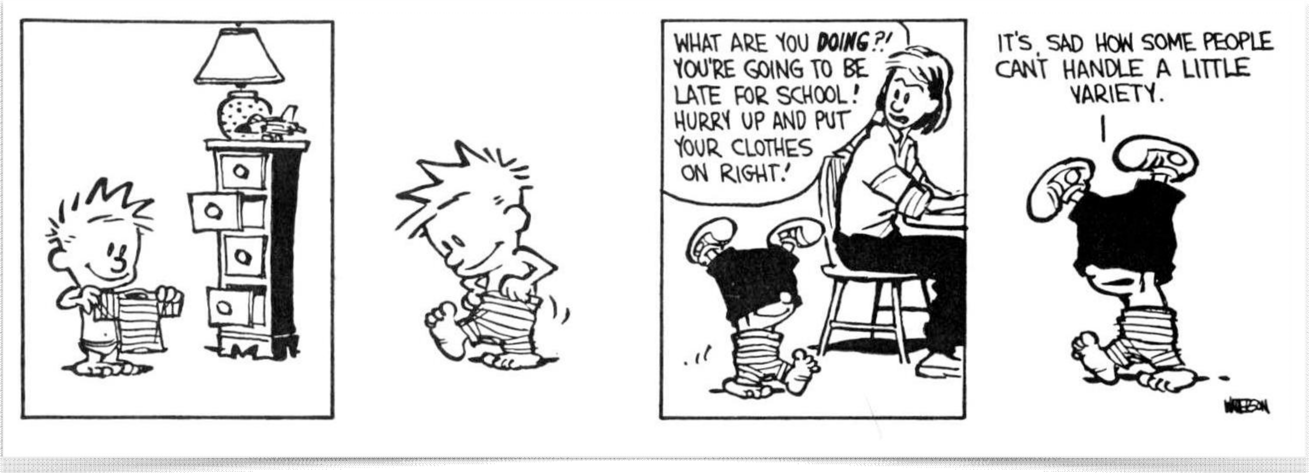
+Tallennustoiminnot			
Harjoitustiedostot (yhteenvedoilla) – 30			•
Harjoitustiedostot (yhteenvedoilla) – 1	•		
Kertymät		•	•
Yhdistettävät harjoitustiedostot			
Muistia jäljellä -lukema			
Muutettavissa oleva tallennusväli – 1s, 2s, 5s, 15s, 60s			
R-R-tallennus			
Harjoitustiedostot (yhteenvedoilla) – 99			
Viikoittainen historia			
Viikoittainen historia – saatavilla polarpersonaltrainer.comissa			
+Harjoitustoiminnot			
Intervalliopastus – syke / vauhti / matka		•	
Kierrosten määrä – 99		•	•
Muistuttajat – kalorikulutus-, matka- tai aikaperusteinen			•
HeartTouch – rannelaitteen käyttö ilman näppäimiä	•		
Graafinen tavoitealueen osoitin		•	•
Automaattinen näytön vieritys			•
Automaattinen kierrostallennus		•	•
Näytön suurennus/pienennys		•	
Intervalliajastimet – aika- tai matkaperusteinen		•	
ZonePointer		•	•

Commonality and Variability



(AL-Msie'Deen, 2014)

Variability



- 'Software variability is the ability of a software system or artefact to be efficiently extended, changed, customized or configured for use in a particular context' (Svahnberg et al., 2005, p. 706)
- Product line variability: how the applications of a product line can differ
  - product line variability & commonalities define the scope of a product line
- Part of product line variability is expressed via software variability
- Variability → flexibility for product differentiation and diversification

Variability Management (VM) (1)

- You need management, when there is enough variability
- "Variability Management (VM) encompasses the activities of
  - explicitly representing variability in software artefacts throughout the lifecycle,
  - managing dependencies among different variabilities, and
  - supporting the instantiations of those variabilities" (Schmid and John, 2004).

⇒ At core of SPLs is the identification and management of commonalities and variations in the systems' artefacts

- Research and practice: variability management is a central concern in SPLs

Variability management (2)

- Involves complex and challenging tasks, needs to be supported by
  - appropriate approaches,
  - techniques, and
  - tools (Bosch et al., 2001; Schmid and John, 2004)
- Ability to represent variability
  - With large number of variants, representation of them becomes important
    - Adequate concepts for practitioners
    - Proper simplicity, clarity and rigour of concepts
- Management processes
- Tools
- “Systematically identifying and appropriately managing variabilities among different systems of a family are the key characteristics that distinguish SPLE from other reuse-based software development approaches” (Chen et al., 2009).

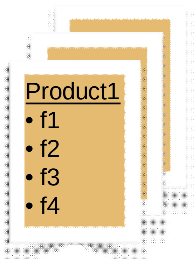
Outline

- Variability & Variability management defined
- Variability modeling
- Tools
- Variability analysis
- Current (own) research
- Want to know more? Pointers & references

Feature (piirre)

- “a distinguishable characteristic of a concept (system, component, etc.) that is relevant to some stakeholder of the concept” (Czarnecki et al., 2000)
- “a logical unit of behavior specified by a set of functional and non-functional requirements” (Bosch, 2000)
- Many other similar definions exist (10+) (Berger et al., 2015)
- The usage of term feature and good/bad features, etc. have been characterized in ”What is a feature?: a qualitative study of features in industrial software product lines” (Berger et al., 2015)
  - SPLC 2015 best paper award

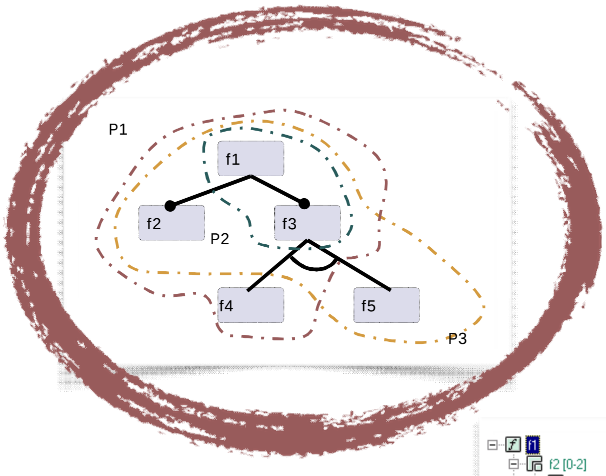
Feature variability



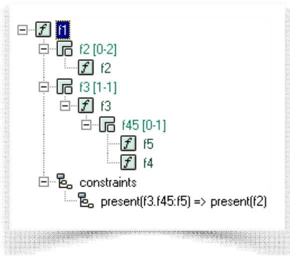
Product feature list

	P1	P2	P3
f1	+	+	+
f2	+		+
f3	+	+	+
f4	+		
f5			+

Product – feature - matrix

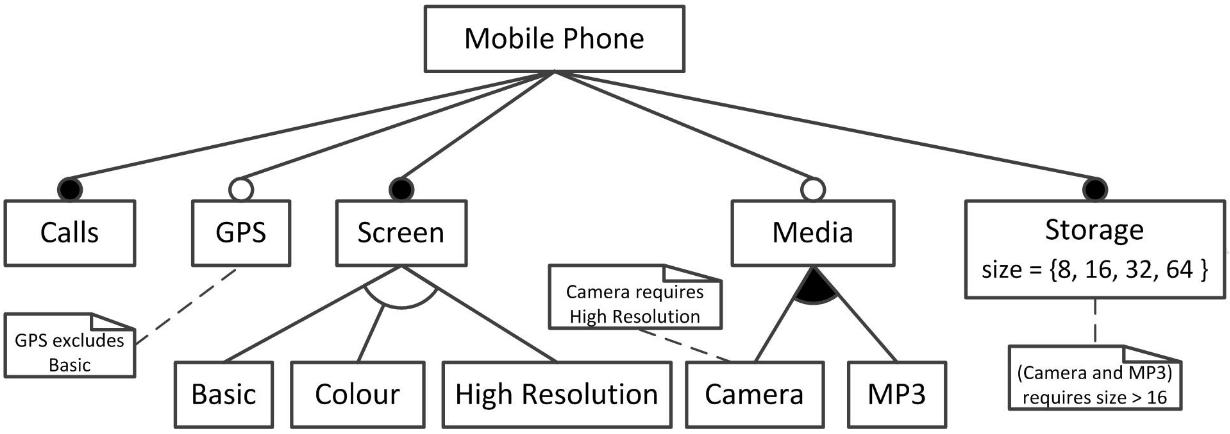


Feature model

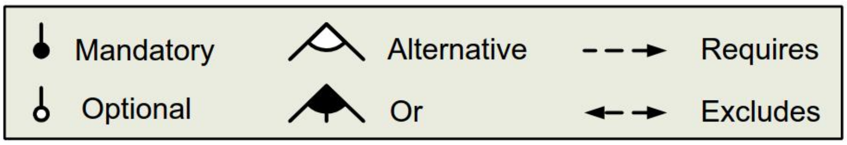


Feature configuration model

Feature Models (FM)



- The most popular means of variability representation (Chen et al., 2009)



(Myllärniemi et al., 2014)
Adapted from (Benavides et al., 2010)

'Extended feature models': add attributes

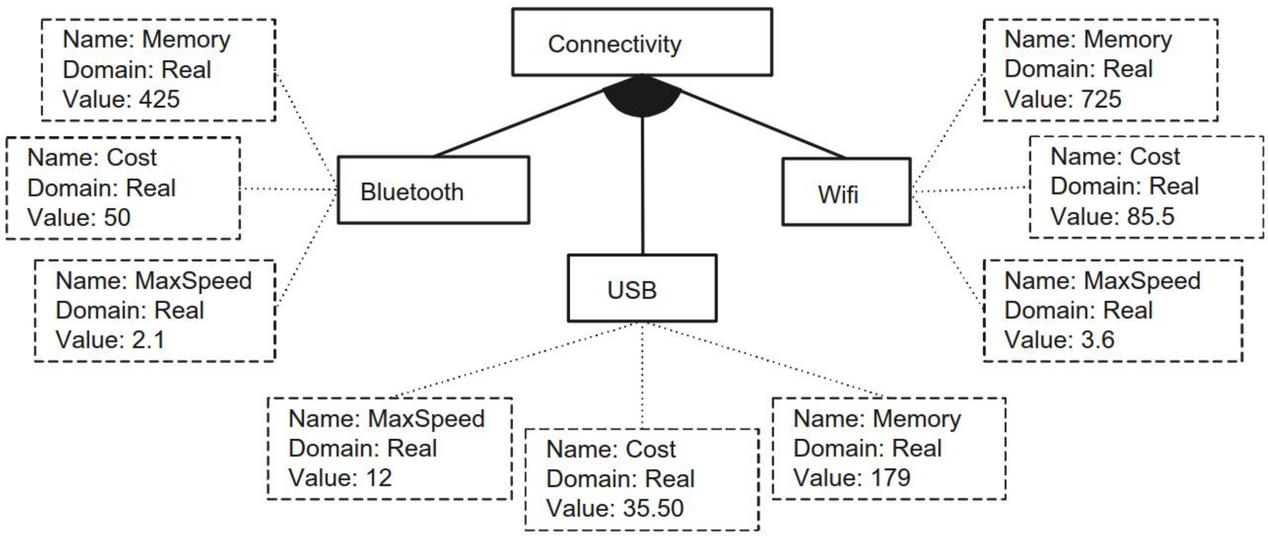
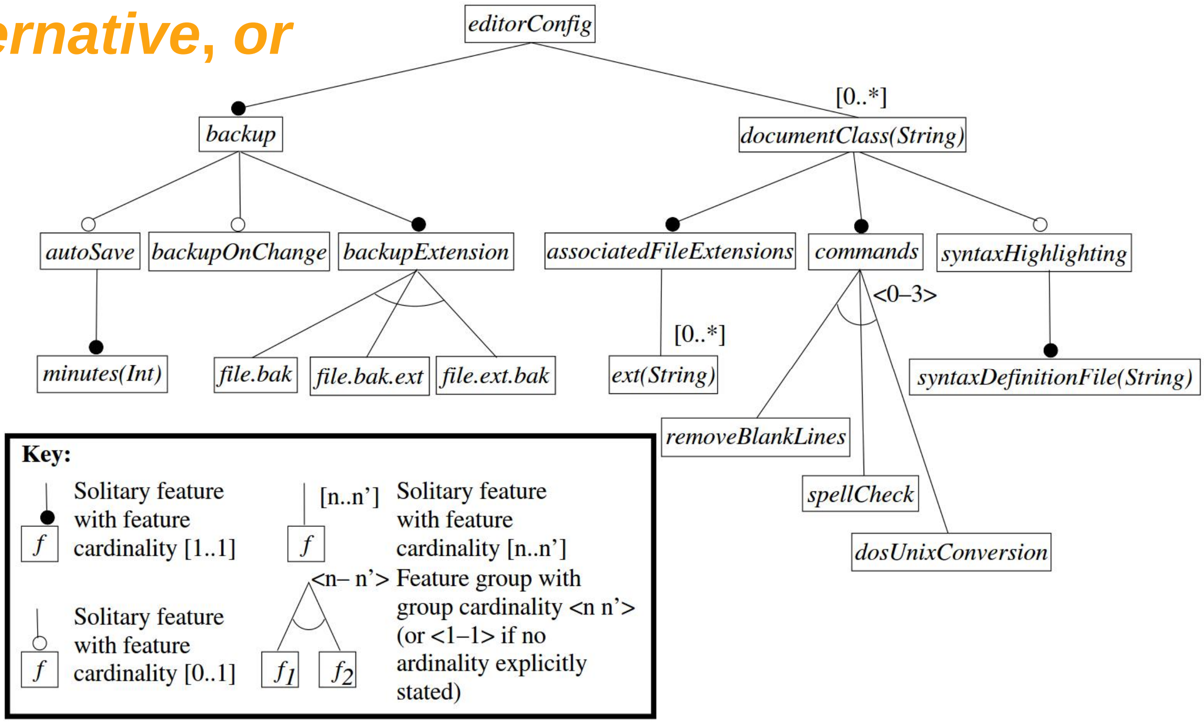


Fig. 2. A sample extended feature model.

- Usually: an attribute has at least name, domain, value
- Constaints on attributes / their values

(Benavides et al., 2010)

Cardinality based feature model: cardinality instead of *mandatory*, *optional*, *alternative*, *or*



(Czarnecki et al., 2005)

Orthogonal Variability Modeling (OVM) terminology: Variation point, variability

- A *variation point:* documents a variable item defining "what can vary"
  - without saying how it can vary (**colour** of a car)
  - "A variation point is a representation of a variability subject within domain artefacts enriched by contextual information. "(Pohl et al., 2005)
  - "place in a design or implementation that identifies a location at which variation occurs"
  - facilitate the systematic documentation and traceability of variability, development for reuse and with reuse, assessment, and evolution
- A *variant:* documents a concrete variation of a variation point defining "how something can vary".
  - **blue** in colour of a car
  - "A variant is a representation of a variability object within domain artefacts." (Pohl et al., 2005)
  - "A variant identifies a single option of a variation point and can be associated with other artefacts to indicate that those artefacts correspond to a particular option." (Pohl et al., 2005)
- *Variability constraints* restrict the variability
  - permissible combinations of variants, e.g. selection of one variant requires or excludes the selection of another variant

(Metzger and Pohl, 2014; Pohl et al., 2005)

Other ways to model software variability

- Orthogonal variability modeling: variation points, variants, constraints, no modeling of commonality
  - OVM (Pohl et al., 2005), Covamof (Sinnema et al., 2004), CVL (OMG, 2015)
- Decision modeling: Questions with constraints and a workflow ("wizard")
  - an overview in (Schmid et al., 2011)
- Clafer (Bąk et al., 2014): Odd but possibly effective mixing of classes and features
- Koala component model: like IC-circuit diagram (van Ommering et al., 2000)
- Kumbang ontology (Asikainen et al., 2007) & Koalish (Asikainen et al., 2004) – extension of Koala: Feature model + component model + types
- Also possible: borrow methods from knowledge-based configuration

Variability realization

- Variability is realized using the capabilities of programming languages, compilers, and linkers
- Approaches
  - use of inheritance
    - e.g., implement alternative sub-classes for an abstract super-class
  - conditional compilation
    - e.g., using preprocessor directives such as #ifdef
  - binary replacement
    - e.g., providing the linker with alternative implementations of libraries
  - aspect-oriented programming
    - e.g., the 'weaving' of alternative code
- Conditional compilation has received significant attention, e.g.
  - type-safe feature modularity
  - treatment of feature dependencies.
- Svahnberg, M., van Gurp, J. and Bosch, J. (2005), "A taxononomy of variability realization techniques", *Software---Practice and Experience*, Vol. 35 No. 8, pp. 705–754.

(Metzger and Pohl 2014)

New approaches

- New types of programming languages consider features and variability as first-class concepts
  - explicitly handle feature modularity and feature dependencies/interactions at the language
  - Feature-oriented programming (FOP)
    - supports the flexible and modular composition of systems from individual features
    - "a feature module encapsulates changes that are made to a program in order to add a new capability or functionality"
  - Delta-oriented programming
    - a compositional programming language
    - a product line is realized by a core module and a set of delta modules
    - The core module implements a valid application developed with single system development techniques
    - Delta modules specify changes to be applied to the core module to implement additional applications
    - Changes to the core model include the adding of additional code (as in FOP), but also removing and even the modification of code
    - Maintainability?
- Variability often cross-cuts the decomposition structure "Cross-cutting variability"
  - Introduce additional composition operations on top of sequential composition
  - treat features as aspects

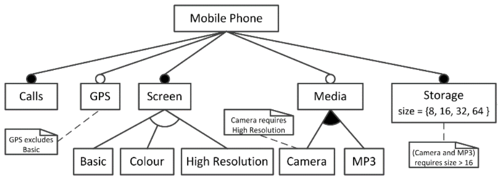
(Metzger and Pohl 2014)

Outline

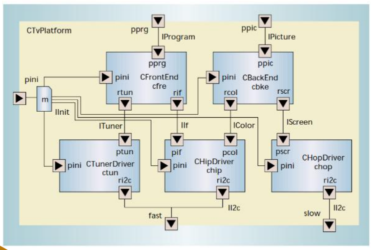
- Variability & Variability management defined
- Variability modeling, variability realization
- Tools
- Variability analysis
- Current (own) research
- Want to know more? Pointers & references

Kumbang tools ≈ Features + Koala Structure

Feature modeling



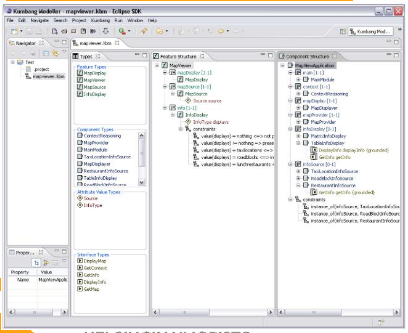
Structural modeling (Koala)



Conceptual foundation

Kumbang ontology

Domain Engineering: Kumbang modeler

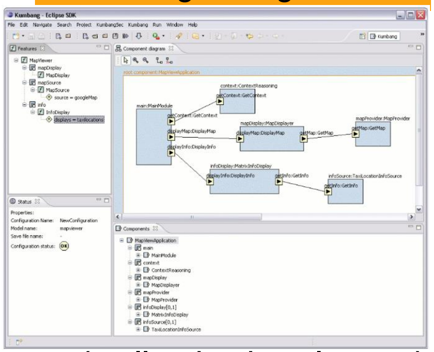


Kumbang ontology has a simple mapping to Weight Constrain rules (a form of logic programming) providing clear formal semantics

.

Answer set programming solvers (Smodels, (or clasp)) provide inference

Application Engineering: Kumbang configurator



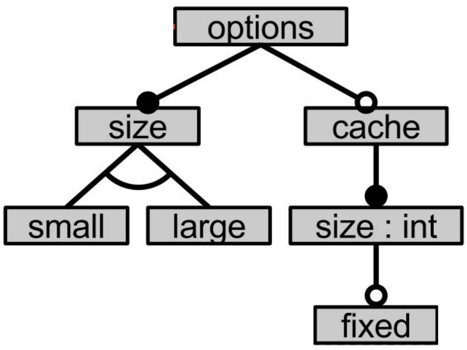
(Myllärniemi et al., 2007)

Clafer

- Clafer unifies class, association, and property (attribute, reference, role) into a single construct called *clafer* (CLAss FEature Relationship)
- A clafer declaration includes multiplicities and may optionally contain a superclafer or a reference to a clafer or both.

```
abstract options
  xor size
    small ?
    large ?
  cache ?
    size → integer
      fixed ?
  [ small && cache ⟹ fixed ]
```

Fig. 7 Feature model of component options in Clafer



(Bąk et al., 2014)

Clafer meaning

```
abstract Person
   Name -> string
   xor Gender
      Male
      Female
   Married ?
   Address
      Street -> string
         UnitNo -> integer ?
JohnDoe : Person
   [ Name = "John" ]
   [ Male ]
   [ Married ]
   [ Street = "12 Main St." ]
   [ UnitNo = 3 ]
```

An abstract clafer called `Person`.
A concrete child reference clafer of `Person` with type `string`.

`UnitNo` is a child of `Street`, which it turn is a child of `Address`.

A concrete clafer called `JohnDoe`, which is followed by a list of constraints restricting the set of persons to persons with the given `Name`, `Street`, etc.

Some constraints do not set values but simply assert clafers, such as, `Male` and `Married`, that is, JohnDoe can be characterized as a `Married` `Male`.

(Antkiewicz, 2015)

Clafer tools

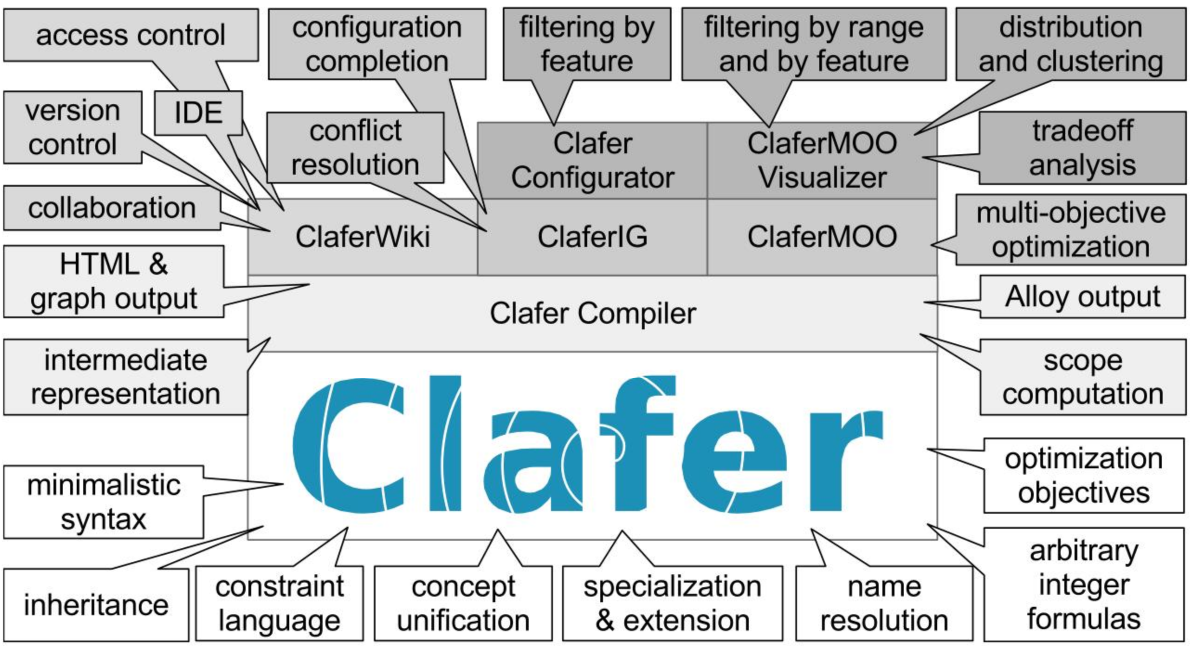
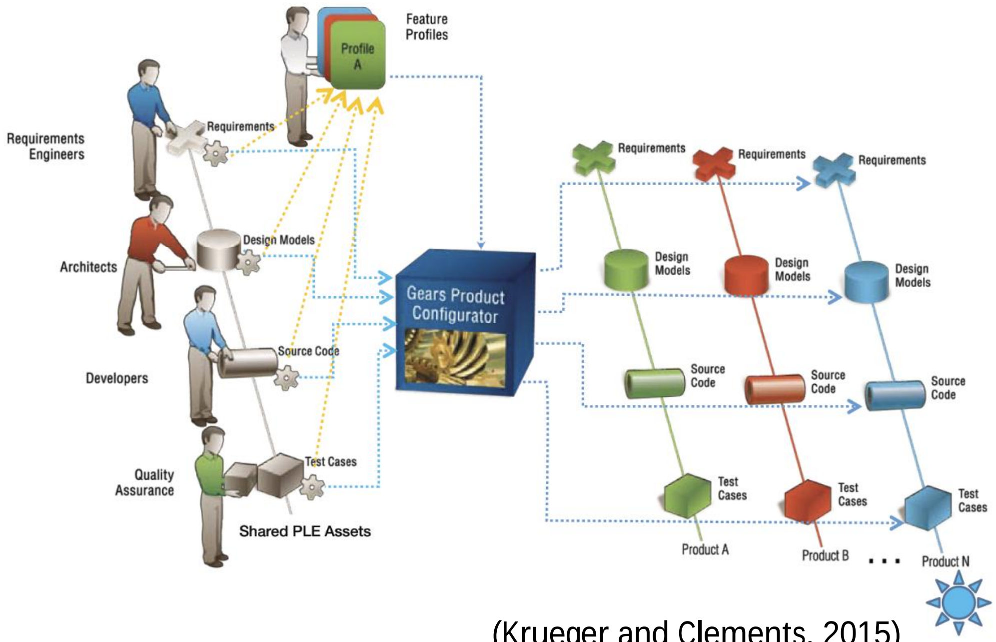


Figure 1: Architecture and Capabilities of Clafer and Tools

(Antkiewicz et al., 2013)

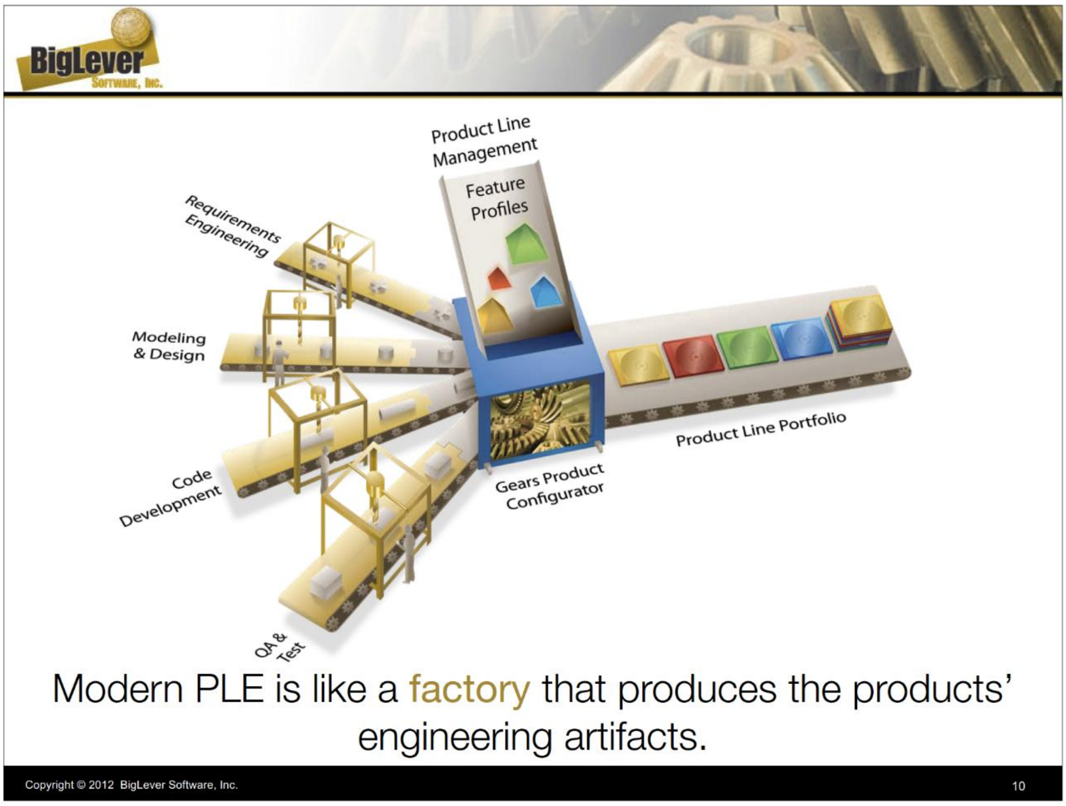
BigLever Software: Gears

Gears exercises the variation points according to the feature profile of the product you want to build.



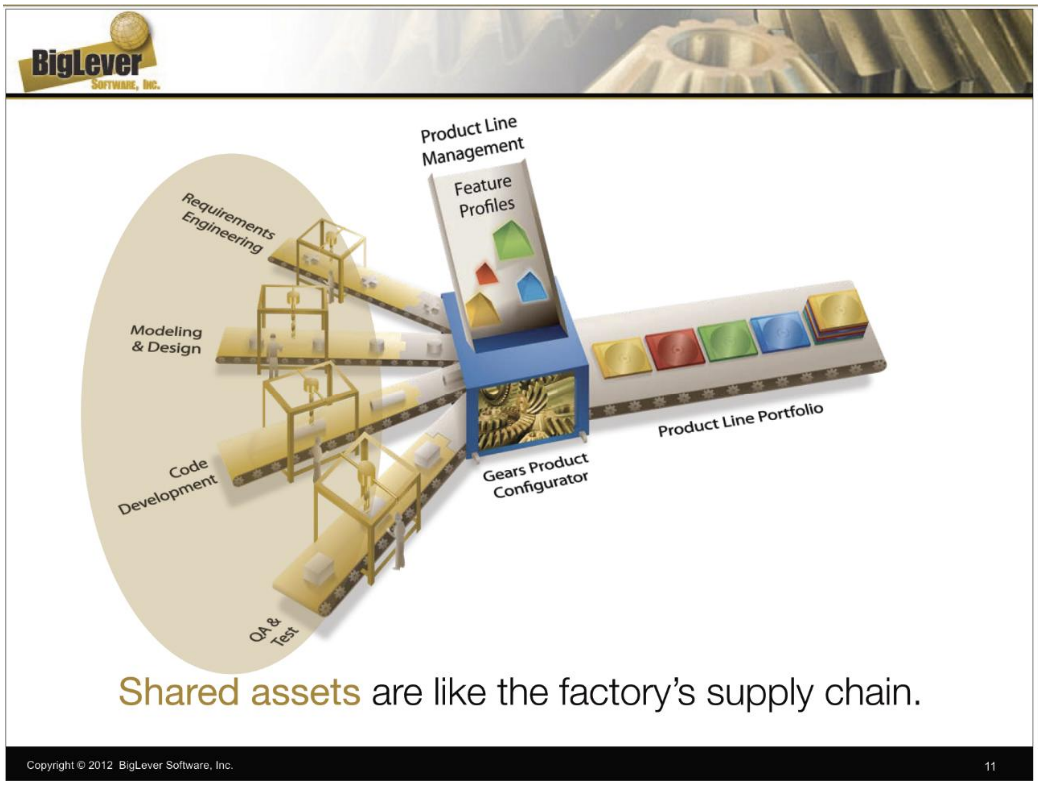
(Krueger and Clements, 2015)

Gears & configurable SPLs (1)



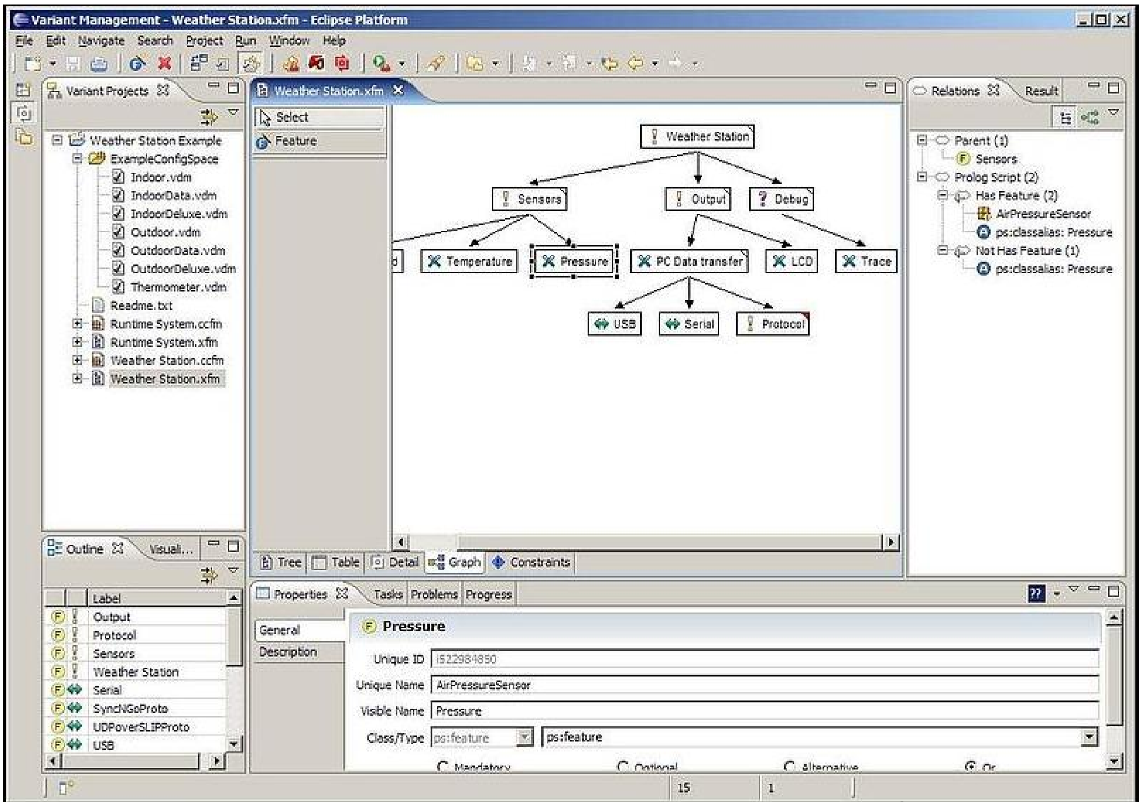
(Krueger and Clements, 2015)

Gears & configurable SPLs (2)



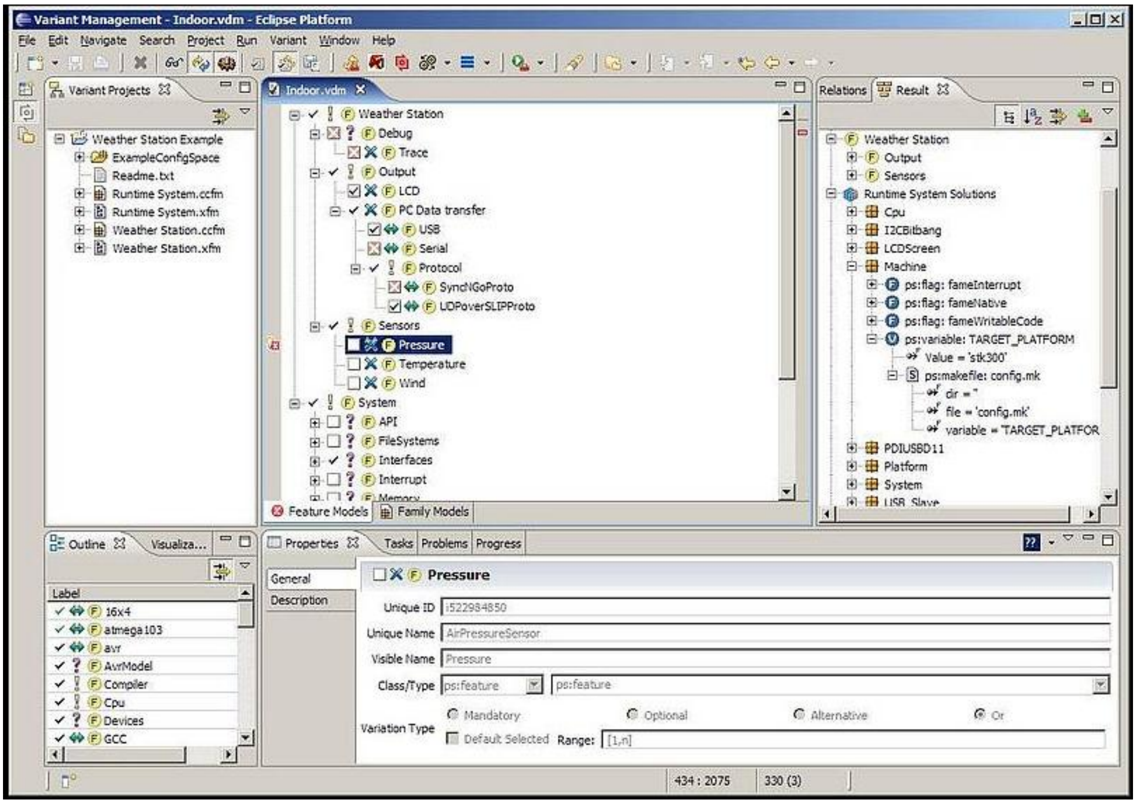
(Krueger and Clements, 2015)

Pure::variants Graphical feature modeling



(Pure.Systems, 2015)

Pure::variants Configuration Editor



(Pure.Systems, 2015)

Outline

- Variability & Variability management defined
- Variability modeling, variability realization
- Tools
- Variability analysis
- Current (own) research
- Want to know more? Pointers & references

(Automated) Analysis of Feature models: big picture

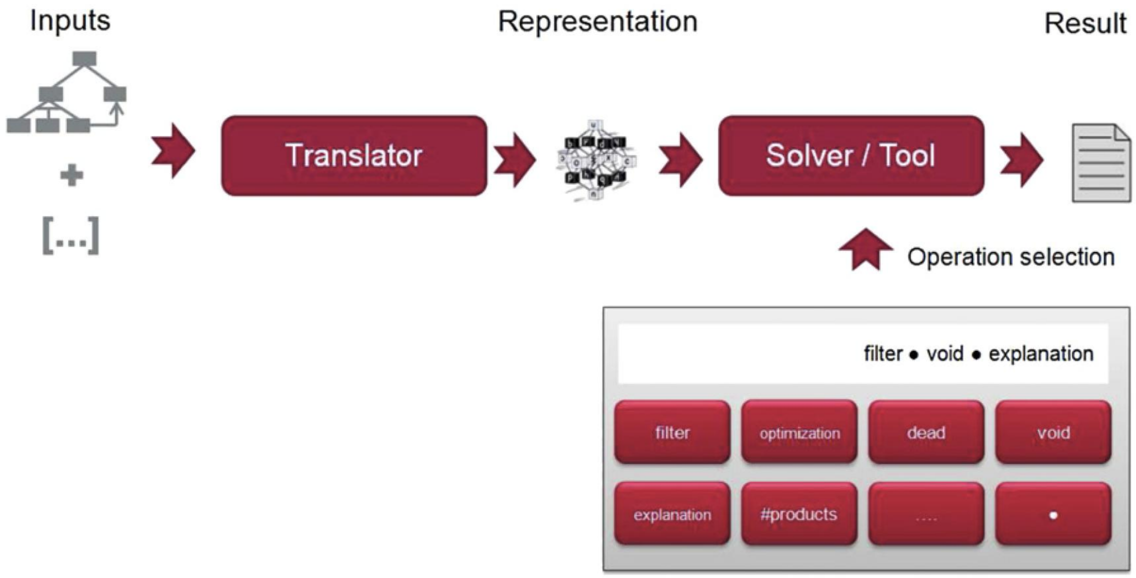
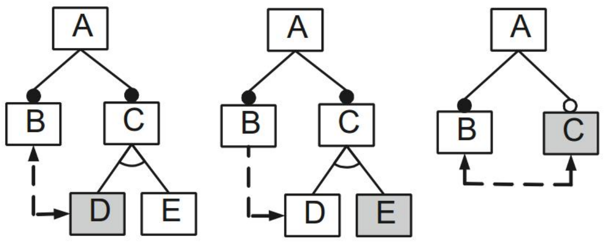


Fig. 4. Process for the automated analysis of feature models.

(Benavides et al., 2010)

Examples of FM analyses

- Benavides et al. (2010) see product derivation / configuration as one form of analysis (requirements are additional inputs)
- Examples
  - Valid product:
    - Input: feature model + a product (i.e. set of features)
    - Output: a value that indicates if the products is valid according to the feature model
  - Dead features i.e. features that cannot appear in any of the products of the software product line



(Benavides et al., 2010)

Fig. 6. Common cases of dead features. Grey features are dead.

Analyses of feature models

(Benavides et al., 2010)

Table 3

Summary of operations and support. 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], Benavides et al. [10–12], Benavides et al. [15], Djebii etal. [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], 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], Batory et al. [7], Schobbens et al. [42,68,69], Trinidad et al. [80], Von der Massen et al. [89].

PL

	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]
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			+		+	+	+	+		+	+	

CP / DL / Multi

	Benavides *et al.* [10, 11, 12] (CP)	Benavides *et al.* [15] (CP)	Djebii *et al.* [34] (CP)	Trinidad *et al.* [78, 76] (CP)	White *et al.* [99] (CP)	White *et al.* [97] (CP)	Abo Zaid *et al.* [1] (DL)	Fan *et al.* [35] (DL)	Wang *et al.* [92, 93] (DL)	Benavides *et al.* [14] (Multi)	Benavides *et al.* [16] (Multi)	Segura [70] (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	+			+		+		+				

Others / No support

	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]	Batory *et al.* [7] (No support)	Schobbens *et al.* [42, 68, 69] (No support)	Trinidad *et al.* [80] (No support)	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	B	C	B	B	B	C	C	C	B	B	B	B	B	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

Outline

- Variability & Variability management defined
- Variability modeling, variability realization
- Tools
- Variability analysis
- Current (own) research
- Want to know more? Pointers & references

Evidence-based software engineering (EBSE)

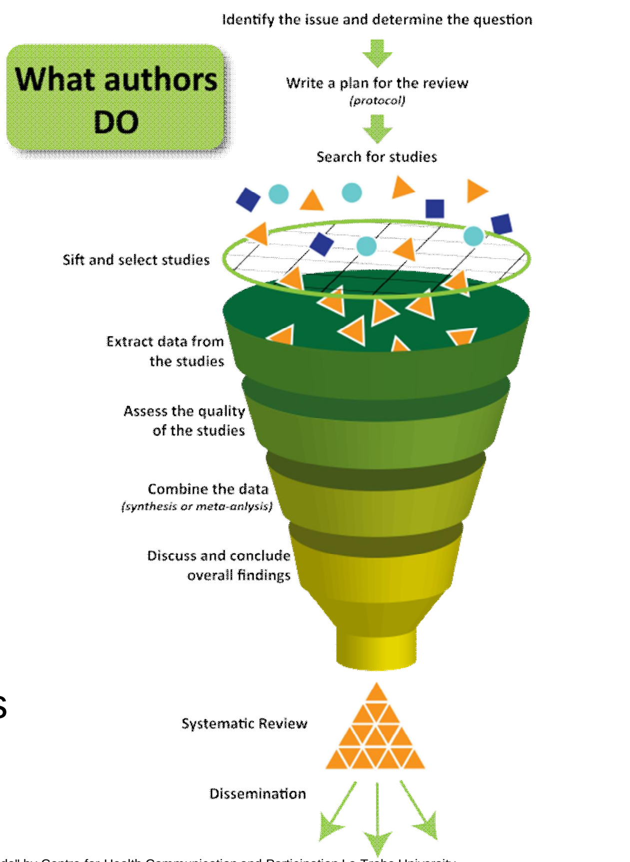
- "EBSE aims to improve decision making related to software development and maintenance by integrating current best evidence from research with practical experience and human values."
  - Idea borrowed from evidence-based medicine

1. Relevant problem or information need → an **answerable question**
2. **Search the literature** for the best available evidence
3. **Critically appraise the evidence** for its validity, impact, and applicability
4. **Integrate** the appraised evidence & practical experience and the current (customer's) context to make decisions
5. **Evaluate performance** and seek ways to improve it

(Dybå et al., 2005)

Systematic reviews (SRs)

- Summarize studies and synthesize evidence about a specific topic following a predefined, systematic and reliable research method
- Systematic literature reviews (SLR)
- Systematic mapping studies (Map)
- Tertiary studies summarize SRs
- SPLs and variability increasingly summarized in SRs



"Systematic reviews - what authors do" by Centre for Health Communication and Participation La Trobe University, Australasian Cochrane Centre - http://navigatingeffectivetreatments.org.au/exploring_systematic_reviews_what_authors_do.html. Licensed under CC BY-SA 3.0 via Commons – https://commons.wikimedia.org/wiki/File:Systematic_reviews_-_what_authors_do.png#/media/File:Systematic_reviews_-_what_authors_do.png

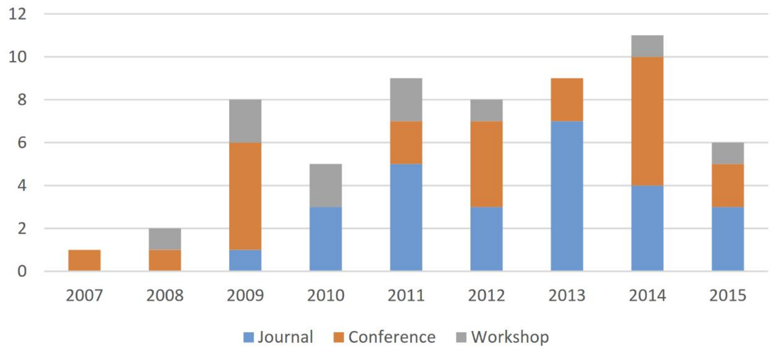
Evidence based SPL and Variability management?

Raatikainen M., Tiihonen J., Männistö T. Systematic Reviews on Software Product Lines and Variability: A Tertiary Study. Work in Progress

- A tertiary study: 59 systematic reviews on SPLs and variability
  - SRs included ~2500 primary studies (duplicates included)
- Domain engineering addressed more often than application engineering
  - scarcely explicit separation
- Few practitioner guidelines
  - except listings and taxonomies of existing research
- Focus mostly on researchers' interests
  - Identify gaps in the research
  - Justify future research
- Lack of empirical primary studies
  - ⇒ missing basis for building an evidence based foundation for SPLs & Variability management

☹

Topic category	#	Studies
Requirements engineering	7	S1 S3 S25 S26 S32 S40 S50
Design	8	S6 S19 S27 S47 S56 S59
Testing	10	S7 S8 S15 S24 S29 S30 S31 S35 S43 S46
Variability management	13	S4 S10 S11 S12 S13 S18 S20 S22 S23 S34 S46 S49 S50
Quality attributes	7	S36 S38 S39 S41 S45 S55
Process model	2	S14 S53
Maintenance	4	S3 S27 S28 S51
Management	6	S2 S16 S25 S33 S48 S57
Specific SPLs	11	S5 S9 S17 S21 S36 S37 S42 S44 S52 S54 S58
Empirics	6	S11 S13 S22 S24 S25 S26



Systematic reviews

Domain / application

Topic category	#	Studies
Requirements engineering	7	S1 S3 S25 S26 S32 S40 S50
Design	8	S6 S19 S27 S47 S56 S59
Testing	10	S7 S8 S15 S24 S29 S30 S31 S35 S43 S46
Variability management	13	S4 S10 S11 S12 S13 S18 S20 S22 S23 S34 S46 S49 S50
Quality attributes	7	S36 S38 S39 S41 S45 S55
Process model	2	S14 S53
Maintenance	4	S3 S27 S28 S51
Management	6	S2 S16 S25 S33 S48 S57
Specific SPLs	11	S5 S9 S17 S21 S36 S37 S42 S44 S52 S54 S58
Empirics	6	S11 S13 S22 S24 S25 S26

Realisation: none
~2500 articles included in SR:s (inclusive duplicates)

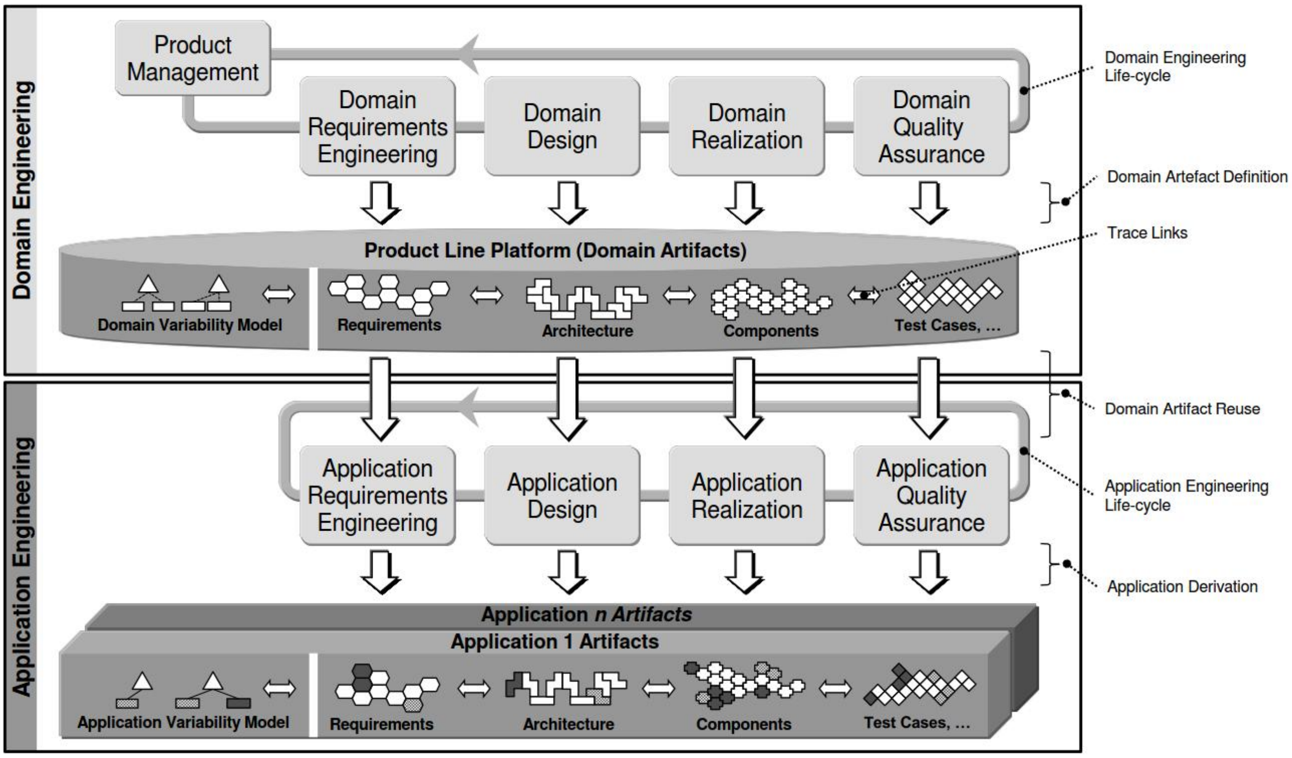


Figure 1: SPLE Framework (adapted from [1])

Note: any life-cycle or process model (e.g., V-model, spiral model, agile models) can be used

(Metzger and Pohl, 2014; [1]=Pohl et al., 2005)

Potential to learn from knowledge-based configuration?

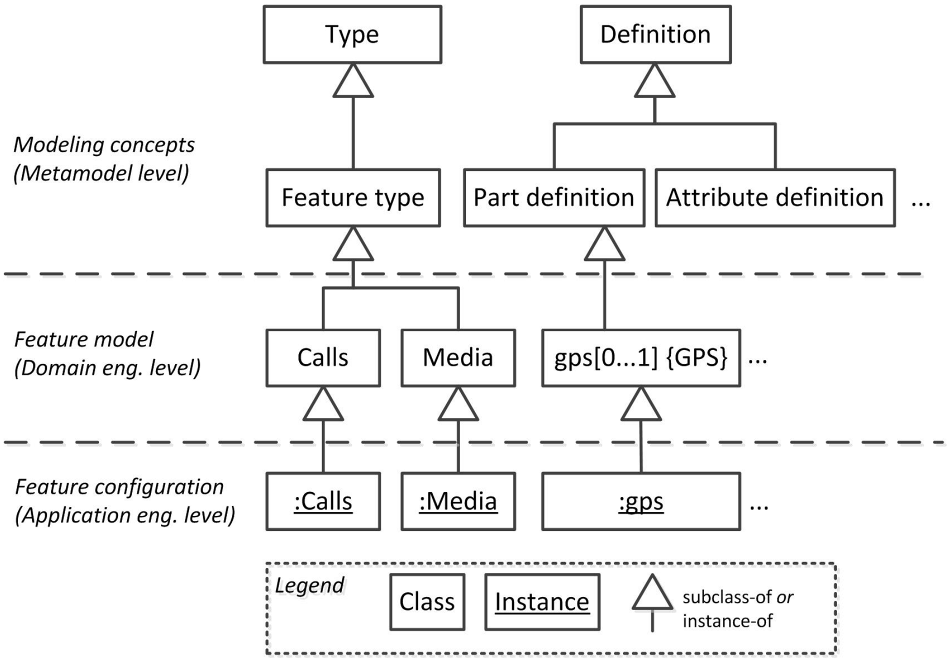
- Product configuration has long history
- Variability management in product configuration shares with software product lines, including similarities in conceptual foundation
  - Potential for knowledge sharing
- SPL modeling has been researched a lot.
- There is potential to transfer principles from product configuration to SPL community

Tiihonen J., Raatikainen M., Myllärniemi V, Männistö T.
Applying Principles from Knowledge-based Configuration to Configurable Software Product Lines
Work in progress

Some principles and potential effects to aim for

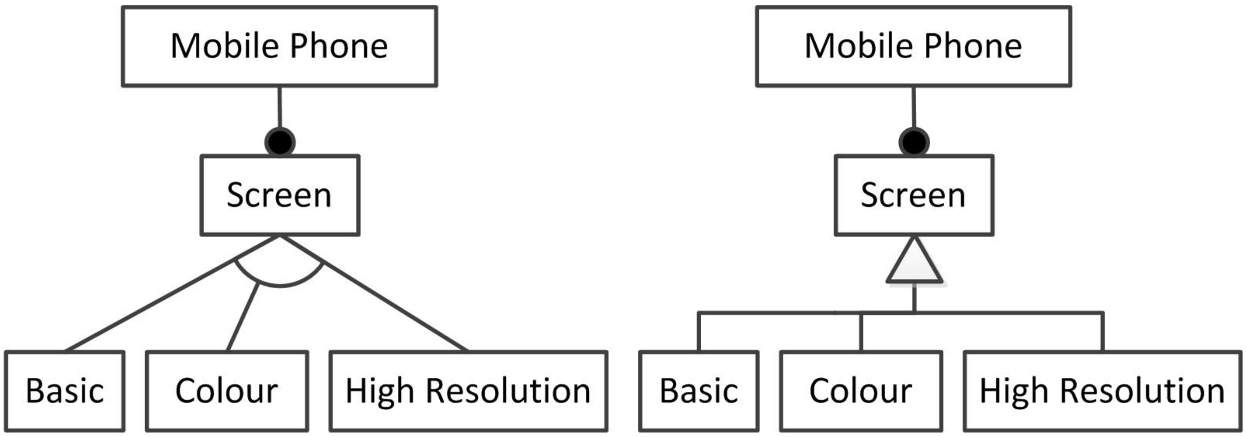
- Separation between types and instances
  - Conceptual separation of domain and application models
  - Types modularize models
  - Reuse via instantiation of types
- Conceptual clarity
  - Distinct relationships such as has-part and is-a
  - Cardinality as a basis for compositional relationships
  - Balance between representational gap and simplicity
- Concepts before representation
  - Domain phenomena as concepts with semantics
  - Multiple representations of concepts such as textual and graphical
  - Equivalence and synchronization of different representations.
  - Representations need a conceptual basis.
  - Support different viewpoints with corresponding concepts.

Three instantiation levels of FM according to the KBC approach



Adapted from (Soininen et al., 1998)

Ambiguous FM concepts: alternative was originally meant to imply specialization: is-a instead of consists-of! (Kang et al., 1990)



Summary

- SPLs can be effective -- significant potential gains
- Business based on SPLs is not easy but it is doable
  - E.g. important management (human) aspects not discussed today
- There are numerous methods and some (quite mature) tools
  - Many research proposals have not ben validated
- Active research
  - But limited evidence-based advice for practitioners
- Many challenges and research opportunities exist
  - Linda Northrop: Major challenges include Accelerating SPL development, Software assurance, Scaling

Outline

- Variability & Variability management defined
- Variability modeling, variability realization
- Tools
- Variability analysis
- Current (own) research
- Want to know more? Pointers & references

Systematic reviews

S1 V. Alves, N. Niu, C. Alves, and G. Valença, "Requirements engineering for software product lines: A systematic literature review," *Information and Software Technology*, vol. 52, no. 8, pp. 806–820, 2010.
S2 W. K. G. Assunção and S. R. Vergilio, "Feature location for software product line migration: A mapping study," in *International Software Product Line Conference (SPLC) - Volume 2*, 2014, pp. 52–59.
S3 N. Bakar, Z. Kasirun, and N. Salleh, "Feature extraction approaches from natural language requirements for reuse in software product lines: A systematic literature review," *Journal of Systems and Software*, vol. 106, pp. 132–149, 2015.
S4 C. Bezerra, R. Andrade, and J. Monteiro, "Measures for quality evaluation of feature models," in *International Conference on Software Reuse (ICSR)*, 2014, pp. 282–297.
S5 V. A. Burégio, S. R. de Lemos Meira, and E. S. de Almeida, "Characterizing dynamic software product lines-a preliminary mapping study," in *International Software Product Line Conference (SPLC) - Volume 2*, 2010, pp. 53–60.
S6 D. Cabrero, J. Garzas, and M. Piattini, "Understanding product lines through design patterns," in *International Conference on Software and Database Technologies (ICSOFT)*, 2007.
S7 I. do Carmo Machado, J. D. McGregor, and E. Santana de Almeida, "Strategies for testing products in software product lines," *SIGSOFT Software Engineering Notes*, vol. 37, no. 6, pp. 1–8, Nov. 2012.
S8 I. do Carmo Machado, J. D. McGregor, Y. C. Cavalcanti, and E. S. de Almeida, "On strategies for testing software product lines: A systematic literature review," *Information and Software Technology*, vol. 56, no. 10, pp. 1183–1199, 2014.
S9 D. Castelluccia and N. Boffoli, "Service-oriented product lines: A systematic mapping study," *SIGSOFT Software Engineering Notes*, vol. 39, no. 2, pp. 1–6, Mar. 2014.
S10 L. Chen, M. Ali Babar, and A. Nour, "Variability management in software product lines: a systematic review," in *International Software Product Line Conference (SPLC)*, 2009, pp. 81–90.
S11 L. Chen, M. Ali Babar, and C. Cawley, "A status report on the evaluation of variability management approaches," in *International Conference on Evaluation and Assessment in Software Engineering (EASE)*, 2009, pp. 118–127.
S12 L. Chen and M. Ali Babar, "A survey of scalability aspects of variability modeling approaches," in *International Software Product Line Conference (SPLC) - Volume 2*, 2009.
S13 L. Chen and M. Ali Babar, "A systematic review of evaluation of variability management approaches in software product lines," *Information and Software Technology*, vol. 53, no. 4, pp. 344–362, 2011.
S14 J. Díaz, J. Pérez, P. P. Alarcón, and J. Garbajosa, "Agile product line engineering - a systematic literature review," *Software - Practice and Experience*, vol. 41, no. 8, pp. 921–941, 2011.
S15 E. Engström and P. Runeson, "Software product line testing - a systematic mapping study," *Information and Software Technology*, vol. 53, no. 1, pp. 2–13, 2011.
S16 J. Ferreira Bastos, P. A. da Mota Silveira Neto, E. S. de Almeida, and S. R. de Lemos Meira, "Adopting software product lines: A systematic mapping study," in *International Conference on Evaluation and Assessment in Software Engineering (EASE)*, 2011, pp. 11–20.
S17 P. Gadelha Queiroz and R. Vaccare Braga, "Development of critical embedded systems using model-driven and product lines techniques: A systematic review," in *Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS)*, 2014, pp. 74–83.
S18 M. Galster, D. Weyns, D. Tofan, B. Michalik, and P. Avgeriou, "Variability in software systems — a systematic literature review," *IEEE Transactions on*

Systematic reviews

S19 I. Groher and R. Weinreich, "Variability support in architecture knowledge management approaches: A systematic literature review," in *Hawaii International Conference on System Sciences (HICSS)*, 2015, pp. 5393–5402.
S20 R. Heradio, D. Fernandez-Amoros, J. Cerrada, and I. Abad, "A literature review on feature diagram product counting and its usage in software product line economic models," *International Journal of Software Engineering and Knowledge Engineering*, vol. 23, no. 8, pp. 1177–1204, 2013.
S21 G. Holl, P. Grünbacher, and R. Rabiser, "A systematic review and an expert survey on capabilities supporting multi product lines," *Information and Software Technology*, vol. 54, no. 8, pp. 828–852, 2012.
S22 A. Hubaux, A. Classen, M. Mendonça, and P. Heymans, "A preliminary review on the application of feature diagrams in practice," in *International Workshop on Variability Modelling of Software-Intensive Systems (VaMos)*, 2010, pp. 53–59.

S23 A. Hubaux, T. Tun, and P. Heymans, "Separation of concerns in feature diagram languages: A systematic survey," *ACM Computing Surveys*, vol. 45, no. 4, pp. 51:1–51:23, 2013.
S24 M. Johansen, Ø. Haugen, and F. Fleurey, "A survey of empirics of strategies for software product line testing," in *International Conference on Software Testing, Verification, and Validation Workshops (ICSTW)*, 2011, pp. 266–269.
S25 M. Khurum, T. Gorschek, and K. Pettersson, "Systematic review of solutions proposed for product line economics," in *International Software Product Line Conference (SPLC) - Volume 2*, S. Thiel and K. Pohl, Eds., 2008, pp. 277–284.
S26 M. Khurum and T. Gorschek, "A systematic review of domain analysis solutions for product lines," *Journal of Systems and Software*, vol. 82, no. 12, pp. 1982–2003, 2009.
S27 J. Kim, S. Kang, and J. Lee, "A comparison of software product line traceability approaches from end-to-end traceability perspectives," *International Journal of Software Engineering and Knowledge Engineering*, vol. 24, no. 4, pp. 677–714, 2014.
S28 M. A. Laguna and Y. Crespo, "A systematic mapping study on software product line evolution: From legacy system reengineering to product line refactoring," *Science of Computer Programming*, vol. 78, no. 8, pp. 1010–1034, 2013.
S29 B. P. Lamancha, M. P. Usaola, and M. P. Velthius, "Software product line testing: A systematic review," in *International Conference on Software and Database Technologies (ICSOFT)*, 2009, pp. 23–30.
S30 B. Lamancha, M. Polo, and M. Piattini, "Systematic review on software product line testing," in *Communications in Computer and Information Science*, vol. 170, 2013, pp. 58–71.
S31 J. Lee, S. Kang, and D. Lee, "A survey on software product line testing," in *International Software Product Line Conference (SPLC)*, 2012, pp. 31–40.
S32 L. B. Lisboa, V. C. Garcia, D. Lucrédio, E. S. de Almeida, S. R. de Lemos Meira, and R. P. de Mattos Fortes, "A systematic review of domain analysis tools," *Information and Software Technology*, vol. 52, no. 1, pp. 1–13, 2010.
S33 L. Lobato, T. Bittar, P. c. Neto, I. MacHado, E. e. De Almeida, and S. Meira, "Risk management in software product line engineering: A mapping study," *International Journal of Software Engineering and Knowledge Engineering*, vol. 23, no. 4, pp. 523–558, 2013.
S34 R. E. Lopez-Herrejon, L. Linsbauer, and A. Egyed, "A systematic mapping study of search-based software engineering for software product lines," *Information and Software Technology*, vol. 61, no. 0, pp. 33–51, 2015.

Systematic reviews

S35 R. Lopez-Herrejon, S. Fischer, R. Ramler, and A. Egyed, "A first systematic mapping study on combinatorial interaction testing for software product lines," in *International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, 2015, pp. 1–10.
S36 S. Mahdavi-Hezavehi, M. Galster, and P. Avgeriou, "Variability in quality attributes of service-based software systems: A systematic literature review," *Information and Software Technology*, vol. 55, no. 2, pp. 320–343, 2013.
S37 B. Mohabbati, M. Asadi, D. b. Gašević, M. Hatala, and H. Müller, "Combining service-orientation and software product line engineering: A systematic mapping study," *Information and Software Technology*, vol. 55, no. 11, pp. 1845–1859, 2013.
S38 S. Montagud and S. Abrahão, "Gathering current knowledge about quality evaluation in software product lines," in *International Software Product Line Conference (SPLC)*, 2009, pp. 91–100.
S39 S. Montagud, S. Abrahão, and E. Insfran, "A systematic review of quality attributes and measures for software product lines," *Software Quality Journal*, vol. 20, no. 3-4, pp. 425–486, 2012.
S40 M. B. S. de Moraes, E. S. de Almeida, and S. Romero, "A systematic review on software product lines scoping," in *Experimental Software Engineering Latin American Workshop (ESELAW)*, 2009, p. 63.
S41 C. Moraga, M. Moraga, M. Genero, and M. Piattini, "A systematic literature review on software product line quality," in *International Conference on Software and Database Technologies (ICSOFT)*, vol. 2, 2011, pp. 269–272.
S42 Y. Morais and G. E. Thaís Burity, "A systematic review of software product lines applied to mobile middleware," in *International Conference on Information Technology: New Generations (ITNG)*, 2009, pp. 1024–1029.
S43 P. A. da Mota Silveira Neto, I. do Carmo Machado, J. D. McGregor, E. S. de Almeida, and S. R. de Lemos Meira, "A systematic mapping study of software product lines testing," *Information and Software Technology*, vol. 53, no. 5, pp. 407–423, 2011.
S44 E. Murugesupillai, B. Mohabbati, and D. Gašević, "A preliminary mapping study of approaches bridging software product lines and service-oriented architectures," in *International Software Product Line Conference (SPLC) - Volume 2*, 2011, pp. 11:1–11:8.
S45 V. Myllärniemi, M. Raatikainen, and T. Männistö, "A systematically conducted literature review: Quality attribute variability in software product lines," in *International Software Product Line Conference (SPLC)*, 2012, pp. 41–45.
S46 C. R. L. Neto, P. A. M. S. Neto, E. S. de Almeida, and S. R. de Lemos Meira, "Mapping study on software product lines testing tools," in *International Conference on Software Engineering & Knowledge Engineering (SEKE)*, 2012, pp. 628–634.
S47 E. A. Oliveira Junior, I. M. S. Gimenes, and J. C. Maldonado, "Software product line evaluation: Categorization and evolution over the years," in *International Conference on Distributed Multimedia Systems (DMS)*, 2012, pp. 83–88.
S48 J. Pereira, K. Constantino, and E. Figueiredo, "A systematic literature review of software product line management tools," in *International Conference on Software Reuse (ICSR)*, 2014, pp. 73–89.
S49 R. Rabiser, P. Grünbacher, and D. Dhungana, "Requirements for product derivation support: Results from a systematic literature review and an expert survey," *Information and Software Technology*, vol. 52, no. 3, pp. 324–346, 2010.
S50 I. S. Santos, R. M. C. C. Andrade, and P. A. Santos Neto, "How to describe spl variabilities in textual use cases: A systematic mapping study," in *Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS)*, 2014, pp. 64–73.
S51 A. R. Santos, R. P. de Oliveira, and E. S. de Almeida, "Strategies for consistency checking on software product lines: A mapping study," in *International Conference on Evaluation and Assessment in Software Engineering (EASE)*, 2015, pp. 5:1–5:14.
S52 R. Santos Rocha and M. Fantinato, "The use of software product lines for business process management: A systematic literature review," *Information and Software Technology*, vol. 55, no. 8, pp. 1355–1373, 2013.
S53 I. F. da Silva, P. A. da Mota Silveira Neto, P. O'Leary, E. S. de Almeida, and S. R. de Lemos Meira, "Agile software product lines: A systematic mapping study," *Software - Practice and Experience*, vol. 41, no. 8, pp. 899–920, 2011.

Systematic reviews

S54 J. da Silva, F. Pereira da Silva, L. do Nascimento, D. Martins, and V. Garcia, "The dynamic aspects of product derivation in DSPl: A systematic literature review," in *International Conference on Information Reuse and Integration (IRI)*, 2013, pp. 466–473.
S55 L. R. Soares, P. Potena, I. d. C. Machado, I. Crnkovic, and E. S. d. Almeida, "Analysis of non-functional properties in software product lines: A systematic review," in *EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA)*, 2014, pp. 328–335.
S56 E. de Souza Filho, R. de Oliveira Cavalcanti, D. Neiva, T. Oliveira, L. Lisboa, E. de Almeida, and S. de Lemos Meira, "Evaluating domain design approaches using systematic review," in *European Conference on Software Architecture (ECSA)*, 2008, pp. 50–65.
S57 E. Tüzün, B. Tekinerdogan, M. Kalender, and S. Bilgen, "Empirical evaluation of a decision support model for adopting software product line engineering," *Information and Software Technology*, vol. 60, pp. 77–101, 2015.
S58 T. Vale, G. B. Figueiredo, E. S. de Almeida, and S. R. de Lemos Meira, "A study on service identification methods for software product lines," in *International Software Product Line Conference (SPLC) - Volume 2*, 2012, pp. 156–163.
S59 G. Vale, E. Figueiredo, R. Abilio, and H. Costa, "Bad smells in software product lines: A systematic review," in *Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS)*, 2014, pp. 84–94.

Other major sources of information

- A Framework for Software Product Line Practice, Version 5.0
  - http://www.sei.cmu.edu/productlines/frame_report/index.html
- Klaus Pohl, Günter Böckle, Frank van der Linden: Software Product Line Engineering – Foundations, Principles, and Techniques
- A bibliography of 600 articles categorized according to 'the big picture' of Pohl & Linden
  - http://www.sse.uni-due.de/en/fose14/
- Software Product Line Conference (SPLC), 2000-
- SPLC Hall of Fame: http://splc.net/fame.html
  - Accepted through the SPLC conference series
  - Concrete cases described, many embedded SW

References

- AL-Msie'Deen, R. (2014), "Reverse Engineering Feature Models From Software Variants to Build Software Product Lines."
- Asikainen, T., Männistö, T. and Soininen, T. (2007), "Kumbang: A domain ontology for modelling variability in software product families", *Advanced Engineering Informatics*, Vol. 21 No. 1, pp. 23–40.
- Asikainen, T., Soininen, T. and Männistö, T. (2004), "A Koala-Based Approach for Modelling and Deploying Configurable Software Product Families", *Software Product-Family Engineering*, Lecture Notes in Computer Science, Springer, Vol. 3014, pp. 225–249.
- Benavides, D., Segura, S. and Ruiz-Cortés, A. (2010), "Automated analysis of feature models 20 years later: A literature review", *Information Systems*, Elsevier, Vol. 35 No. 6, pp. 615–636.
- Berger, T., Lettner, D., Rubin, J., Grünbacher, P., Silva, A., Becker, M., Chechik, M., et al. (2015), "What is a feature?", *Proceedings of the 19th International Conference on Software Product Line - SPLC '15*, ACM Press, New York, New York, USA, pp. 16–25.

References

- Bosch, J. (2000), *Design & Use of Software Architectures: Adopting and Evolving a Product Line Approach*, Addison-Wesley.
- Bosch, J., Florijn, G., Greefhorst, D., Kuusela, J., Obbink, H. and Pohl, K. (2001), "Variability Issues in Software Product Lines", *Proc. of the Fourth International Workshop on Product Family Engineering(PFE-4)*, Bilbao, Spain.
- Chen, L., Ali Babar, M. and Ali, N. (2009), "Variability management in software product lines: a systematic review", *International Software Product Line Conference*, SPLC '09, Carnegie Mellon University, Pittsburgh, PA, USA, pp. 81–90.
- Czarnecki, K., Eisenecker, U.W. and and Eisenecker, U.W. (2000), *Generative Programming Methods, Tools, and Applications*, ACM Press/Addison-Wesley.
- Dybå, T., Kitchenham, B.A. and Jorgensen, M. (2005), "Evidence-based software engineering for practitioners", *IEEE Software*, Vol. 22 No. 1, pp. 58–65.

References

- Metzger, A. and Pohl, K. (2014), “Software Product Line Engineering and Variability Management: Achievements and Challenges”, *Proceedings of the on Future of Software Engineering*, FOSE 2014, ACM, New York, NY, USA, pp. 70–84.
- Myllärniemi, V., Tiihonen, J., Raatikainen, M. and Felfernig, A. (2014), “Using Answer Set Programming for Feature Model Representation and Configuration”, in Felfernig, A., Forza, C. and Haag, A. (Eds.),*Proceedings of the 16th International Configuration Workshop (confWS 2014)*, Graz University of Technology, Graz, Austria, Vol. 1220, pp. 1–8.
- OMG. (2015), “Common Variability Language Wiki”, *Common Variability Language Wiki*.
- Pohl, K., Böckle, G. and van der Linden, F.J. (2005), *Software Product Line Engineering: Foundations, Principles and Techniques*, Springer Berlin Heidelberg, Secaucus, NJ, USA, doi:10.1007/3-540-28901-1.
- Schmid, K. and John, I. (2004), “A customizable approach to full lifecycle variability management”, *Science of Computer Programming*, Vol. 53 No. 3, pp. 259–284.

References

- Schmid, K., Rabiser, R. and Grünbacher, P. (2011), “A comparison of decision modeling approaches in product lines”, in Heymans, P., Czarnecki, K. and Eisenecker, U.W. (Eds.),*5th International Workshop on Variability Modelling of Software-Intensive Systems, Namur, Belgium*, ACM International Conference Proceedings Series, ACM, pp. 119–126.
- Sinnema, M., Deelstra, S., Nijhuis, J. and Bosch, J. (2004), “COVAMOF: A Framework for Modeling Variability in Software Product Families”, in Nord, R.L. (Ed.),Lecture Notes in Computer Science, Springer Berlin Heidelberg, Berlin, Heidelberg, Vol. 3154, pp. 197–213.
- Soininen, T., Tiihonen, J., Männistö, T. and Sulonen, R. (1998), “Towards a general ontology of configuration”, *AI EDAM*, Vol. 12 No. 4, pp. 357–372.
- Svahnberg, M., van Gurp, J. and Bosch, J. (2005), “A taxononomy of variability realization techniques”, *Software---Practice and Experience*, Vol. 35 No. 8, pp. 705–754.
- Tiihonen, J., Lehtonen, T., Soininen, T., Pulkkinen, A., Sulonen, R. and Riitahuhta, A. (1998), “Modelling Configurable Product Families”, *4th WDK Workshop on Product Structuring*, Delft, The Netherlands, pp. 29–50.

Thank you for your attention!

Questions?