Engineer
Your
Competitive
Advantage

BigLever
SOFTWARE, INC.

A Feature Ontology to Support Feature-based Product Line Engineering

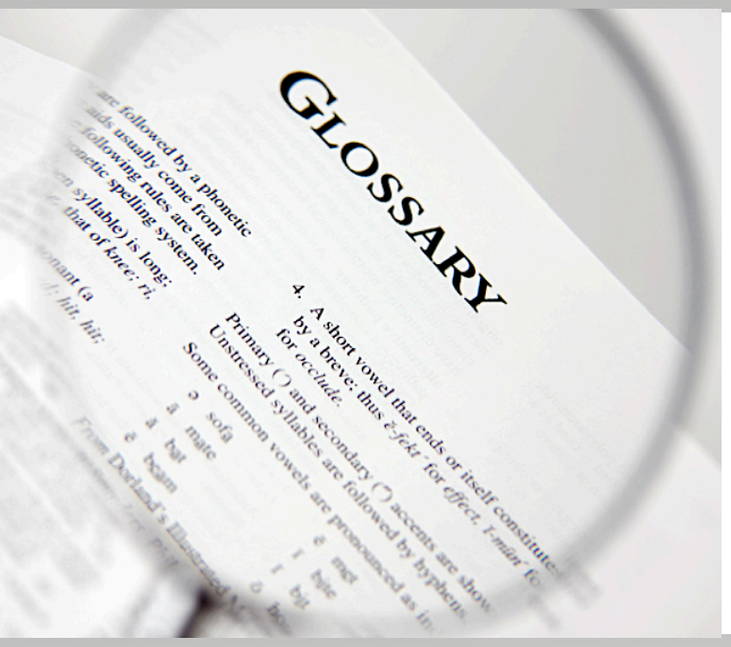
Charles Krueger, PhD, CEO, BigLever
Paul Clements, PhD, VP of Customer Success, BigLever

27th Annual INCOSE International Symposium (IS 2017)
Adelaide, Australia
July 15-20, 2017

Product Line Engineering (PLE) Defined

Product Line:

a family of similar products or systems with variations in features and functions

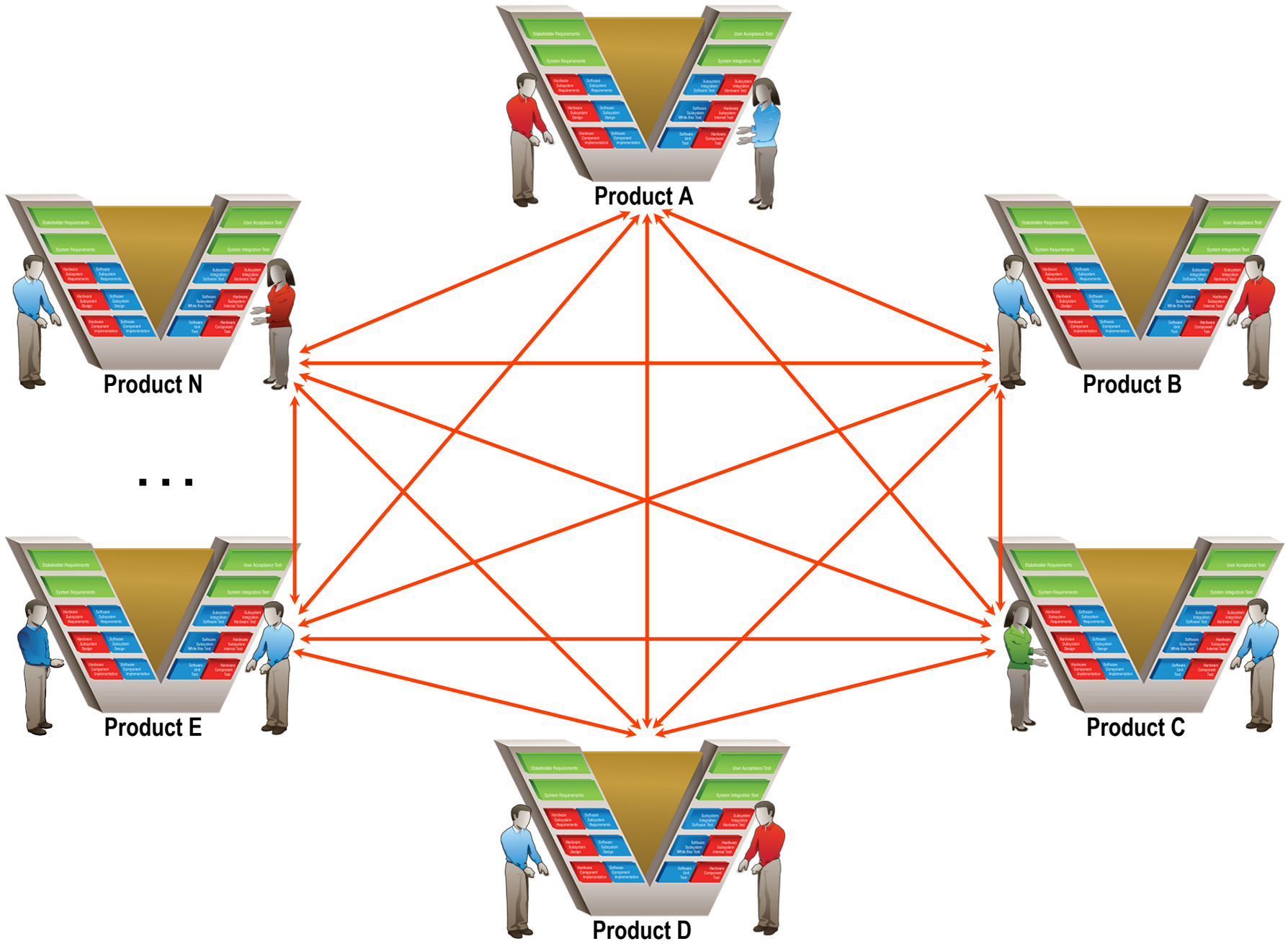


Product Line Engineering:

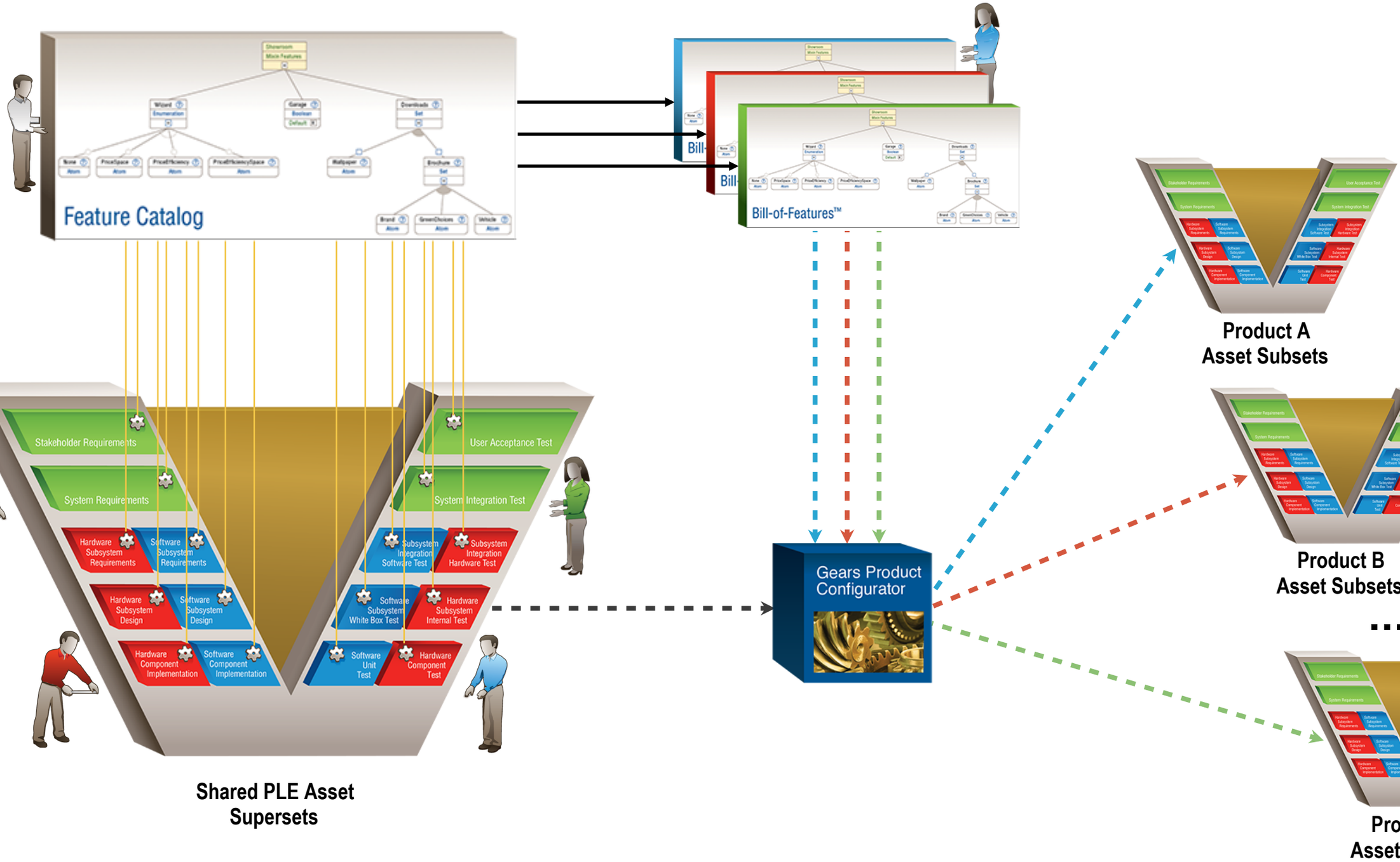
the engineering of a product line using a *shared set of engineering assets*, a *managed set of features*, and an *automated means of production*…

- taking advantage of the **commonality** shared across the family
- efficiently and systematically managing the **variation** among the systems

PLE is a move away from product-centric duplication, branch & merge, clone-and-own, N^2 coordination



Feature-based PLE Factory Workflow



What's a Feature?

Features

- “A distinguishing characteristic that sets products in a product line apart from each other” [3]
  - This can range from large scale customer facing capabilities — like autonomous driving on an automobile — to fine grained implementation details like algorithm tradeoffs in range detection
- Engineering realization of distinguishing characteristics
  - Products are differentiated by differences in all of the places where the digital engineering representations of any two products differ from each other.
    - Requirements objects, model elements from design specs, test cases, lines of software, mechanical parts in a Bill of Materials, paragraphs or sections in a user’s manual, slides in training courseware, and much more

[3] Kang, K.; Cohen, S.; Hess, J.; Novak, W.; & Peterson, A. “Feature-Oriented Domain Analysis (FODA) Feasibility Study” (CMU/SEI-90-TR-021, ADA235785). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon, 1990.

Features

- In a product line comprising thousands to tens of thousands of instances there may easily be millions of these differences.
- While these are undeniably “distinguishing characteristics,” albeit tiny ones, and undeniably important to manage correctly, they cannot be features: No organization could create and manage a bank of millions of features.
- More precisely, there cannot be a one-to-one correspondence between a feature and a variation in an engineering artifact.

Features are Abstractions

- There must be a one-to-*many* correspondence between features and engineering artifact variation.
- Another word for a one-to-many mapping is *abstraction [4].*
- Features are *abstractions* of variations; one feature can configure multiple artifact-level variations.

[4] Saitta, Lorenza, and Zucker, Jean-Daniel, *Abstraction in Artificial Intelligence and Complex Systems,* Springer Science & Business Media, 2013.

A Feature Ontology for Feature-based PLE

Ontology

- An ontology is an “explicit formal specification of the terms in a domain and relations among them” [1], in order to “share common understanding of the structure of information among people or software agents” [2].

[1] Gruber, T.R. (1993). A Translation Approach to Portable Ontology Specification. *Knowledge Acquisition* **5**: 199-220.

[2] Noy, Natalya F., and McGuinness, Deborah L., “Ontology Development 101: A Guide to Creating Your First Ontology,” Stanford University, Stanford University, Stanford, CA, 94305, http://protege.stanford.edu/publications/ontology_development/ontology101-noy-mcguinness.html, downloaded 09 November 2016.

Feature-based PLE Feature Ontology

- Based on abstraction **levels**, where each level comprises:
  - Data structure for the **feature model** at that level
  - Multiple **feature profile** instances of the feature model
  - Abstraction provided to the next higher level in the ontology

- Must reduce complexity from ranging from 1,000,000 options and variants down to 10 options and variants

- Oriented around the diverse feature management roles and responsibilities across an enterprise
  - Implementors, designers, architects, chief engineers, product management, product marketing, portfolio management, manufacturing, service, IoT operations, sales, corporate strategy

Level 0
Asset Variation Points

- Purpose and roles
  - Asset engineers create feature-based variation points to implement differentiating characteristics
- Data structure for the feature model
  - Each variation point is either optional or comes in multiple variants
- Feature profiles
  - Variation point behaviors result in different possible instantiations
- Abstraction provided to the next higher level
  - None. Feature abstractions much be discovered.
  - Why, why, why…

Level 1
Primitive Standalone Features

- Purpose and roles
  - Feature engineers create feature abstractions for root-cause functional and non-functional variation
  - Can be derived bottom-up or top-down
- Data structure for the feature model
  - Each feature has a name and a type, such as boolean, enumeration, set, record, integer, …
- Feature profiles
  - Profiles are defined according to the possible instantiations of the feature type
- Abstraction provided to the next higher level
  - The feature names, types, and profile instantiations are directly provided

Level 2
Component Feature Model

- Purpose and roles
  - Feature and component architects modularize primitive features into component feature models
- Data structure for the feature model
  - Tree structured aggregation of primitive features
- Feature profiles
  - A named list of desired profiles are defined according to possible instantiations of the feature model
- Abstraction provided to the next higher level
  - Component Feature — component feature model is encapsulated with named feature profiles and the bundle is given an abstract feature name

Level 3
Subsystem Feature Model

- Purpose and roles
  - Feature and subsystem architects modularize component feature models into subsystem feature models
- Data structure for the feature model
  - List structured aggregation of component features
- Feature profiles
  - A named list of desired profiles are defined according to possible instantiations of the feature model
- Abstraction provided to the next higher level
  - Subsystem Feature — subsystem feature model is encapsulated with named feature profiles and the bundle is given an abstract feature name

Level 4
System Feature Model

- Purpose and roles
  - Feature catalog and system architects modularize subsystem feature models into system feature models
- Data structure for the feature model
  - List structured aggregation of component and subsystem features
- Feature profiles
  - A named list of desired profiles are defined according to possible instantiations of the feature model
- Abstraction provided to the next higher level
  - System Feature — system feature model is encapsulated with named feature profiles and the bundle is given an abstract feature name

Level 5
System-of-systems Feature Model

- Purpose and roles
  - Feature catalog and portfolio owners modularize system feature models into system-of-system feature models
- Data structure for the feature model
  - List structured aggregation of component, subsystem, and system features
- Feature profiles
  - A named list of desired profiles are defined according to possible instantiations of the feature model
- Abstraction provided to the next higher level
  - System-of-Systems Feature — system-of-systems feature model is encapsulated with named feature profiles and the bundle is given an abstract feature name

Level 5++
Feature Bundle Overlay

- Purpose and roles
  - Product marketing portfolio designers overlay marketing feature bundles onto system-of-system feature models
  - System-of-system feature models with feature bundle overlays are used by upstream and downstream operations, such as portfolio planning, manufacturing, sales, IoT, service, and more
- Data structure for the feature model
  - Overlay on a subset of the system-of-system feature model that constrains desired marketing feature combinations
- Feature profiles
  - A named list of desired bundle profiles are defined according to possible instantiations of the overlay model
- Abstraction provided to the next higher level
  - Bundle combines multiple system, subsystem, and component members into one

Feature Ontology Abstractions Levels

Level	Abstraction	Purpose and Roles	Complexity Potential
5++	Feature Bundle Overlay	Product marketing portfolio designers overlay marketing feature bundles onto system-of-system feature models	2^{10}
5	System-of-Systems Feature Model	Feature catalog and portfolio owners modularize system feature models into system-of-system feature models	2^{50}
4	System Feature Model	Feature catalog and system architects modularize subsystem feature models into system feature models	2^{100}
3	Subsystem Feature Model	Feature and subsystem architects modularize component feature models into subsystem feature models	2^{250}
2	Component Feature Model	Feature and component architects modularize primitive features into component feature models	$2^{1,000}$
1	Primitive Standalone Feature	Feature engineers create feature abstractions for root-cause functional and non-functional variation	$2^{10,000}$
0	Asset Variation Points	Asset engineers create feature-based variation points to implement differentiating characteristics	$2^{1,000,000}$

www.biglever.com