Software
Systems
Engineering

Stiftung Universität Hildesheim 2003

Introduction to Product Line Engineering

Prof. Dr. Klaus Schmid
schmid@sse.uni-hildesheim.de

Prof. Dr. Klaus Schmid

- In Software Engineering since 1992 (starting with Requirements Engineering
- Working in PLE since 1997
- Current focus on
  - Requirements Engineering
  - Product Line Engineering
  - Adaptive Systems

Formerly: Department Head @ Fraunhofer IESE

Currently: Professor of Software Engineering @ University of Hildesheim

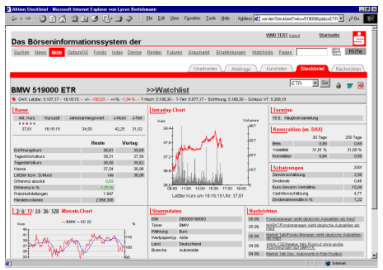
Contents

Observation

Modern software development organizations

Embedded Product Lines

Information Systems



Observation

- Many companies develop over time many similar systems
- Systems may vary
  - According to different countries
  - Different sub-markets (e.g., Diesel vs. gasoline)
  - Different Market Segments
  - or simply due to different customer requirements

➢ How can we optimize this process?

Project → Product Line

Vision of Product Line Engineering

Key Goal:

exploit commonality in externally (visible) properties of the software (system) in terms of commonality of the implementation

Product Line Engineering vs. Traditional Software Engineering

Project focus → Integrated development of a set of products

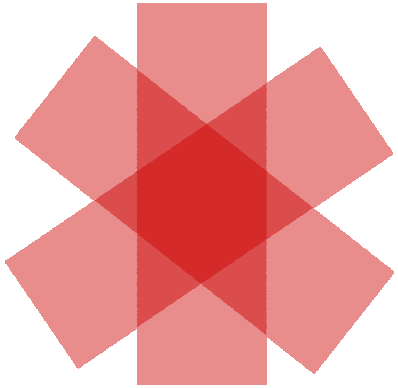
All further issues are a consequence of this focus shift:

- How to relate system properties and system implementation?
- How to deal with differences among systems?
- What products to plan for… ?
- ..

Contents

Core Idea of Product Line Engineering

- Product 1
- Product 2
- Product 3



- **Core idea:** Similarity of Products =
  Commonality + (reg.) Variability + Product-specific parts

  ↓ develop once ↓ make selectable ↓ single development

Product Line Engineering / Product Family Engineering

Complete Shift of Viewpoint

instead of **producing a product** and reusing parts
produce a set of products in an integrated manner

⇒ ***Engineer differences***

Core Principles

- anchor the development in the business portfolio
- treat variations (differences) among products as first class entities
  ⇒ variability management
- establish a common technical basis to support plug-and-play reuse
  ⇒ reference architecture

Principles

Product Line Engineering:

Software product line engineering is a systematic approach for the integrated development of a set of products while simultaneously exploiting the commonality of products

Note:

- ***Product*** refers here to anything an organizational unit delivers!

This allows so-called ***hierarchical product lines***, i.e., the product developed in one product line is part of the reuse platform of the second

Definition

Marketed (Software) Product Line:

A set of products that are marketed together as sharing a common set of concepts or features.

Engineered (Software) Product Line:

A set of products that are engineered together so as to share major parts of their implementation.

Development *with* vs. Development *for*

Development *with* reuse:

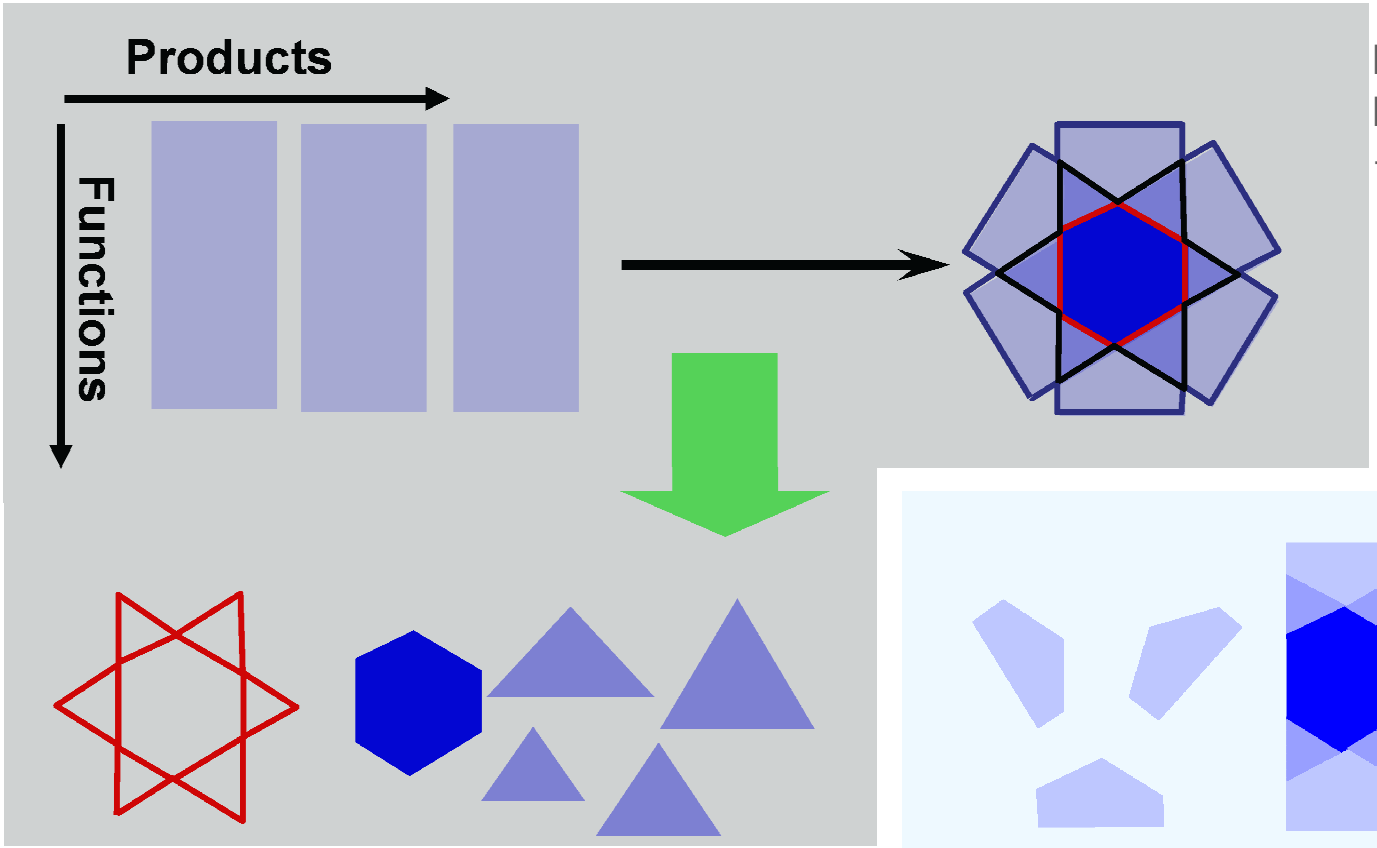
Any software development activities where development artifacts are used that were not specifically developed for these activities.

Development *for* reuse:

Any software development activities that aim at developing software artifacts with the specific purpose of reusing them in a different project context.

Product Line Engineering / Product Family Engineering

Basic Approach



Domain Engineering
for reuse

Analysis

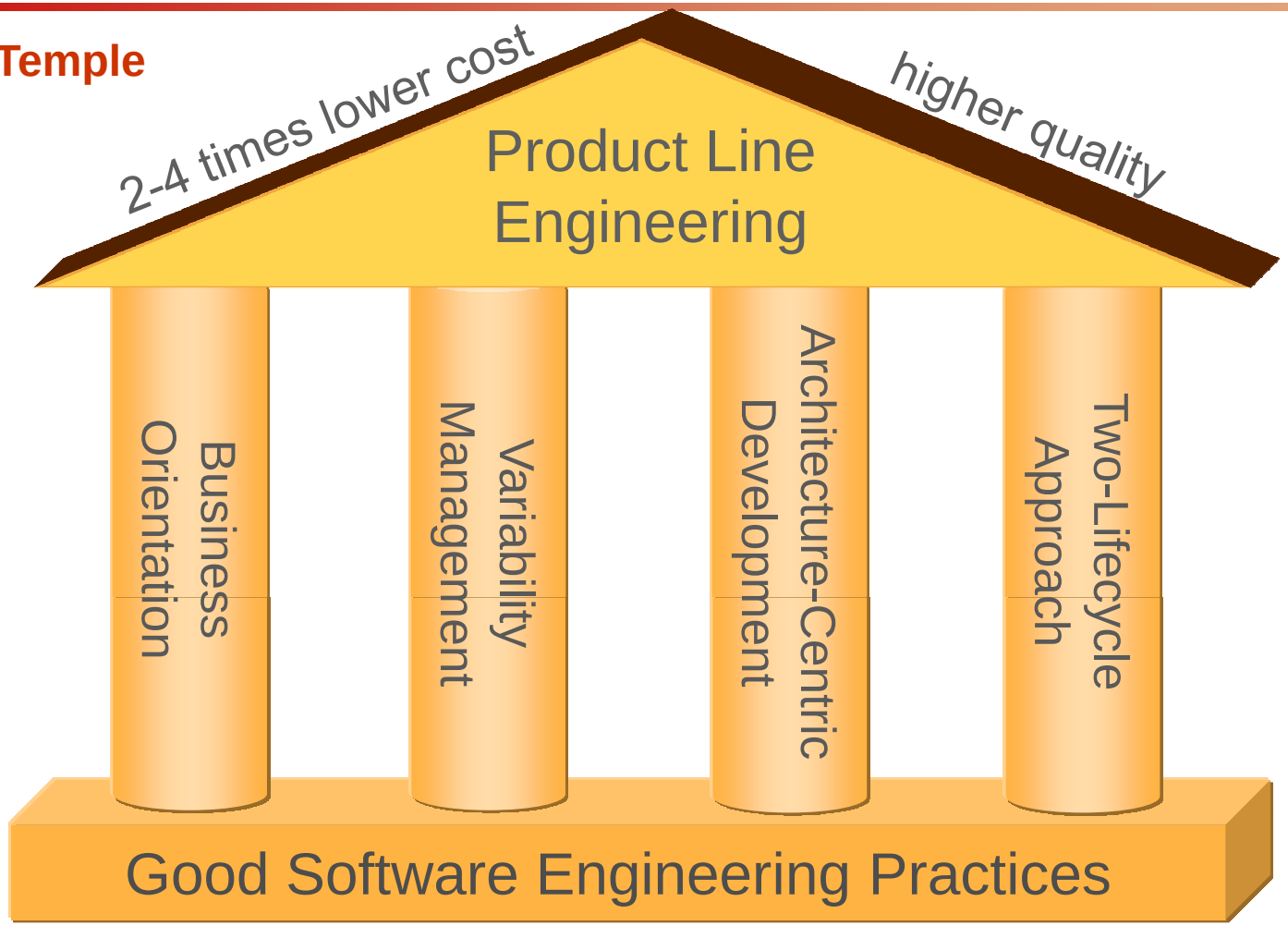
Application Engineering
with reuse

Construction

Contents

The Four Pillars

PLE Temple



Business Orientation

What does it mean:

- explicit focus on to be delivered functionality (make reusable what pays)
- the investment in reuse drives the overall strategy

What does it *not* mean:

- key criterion for a product is whether it fits in the product line

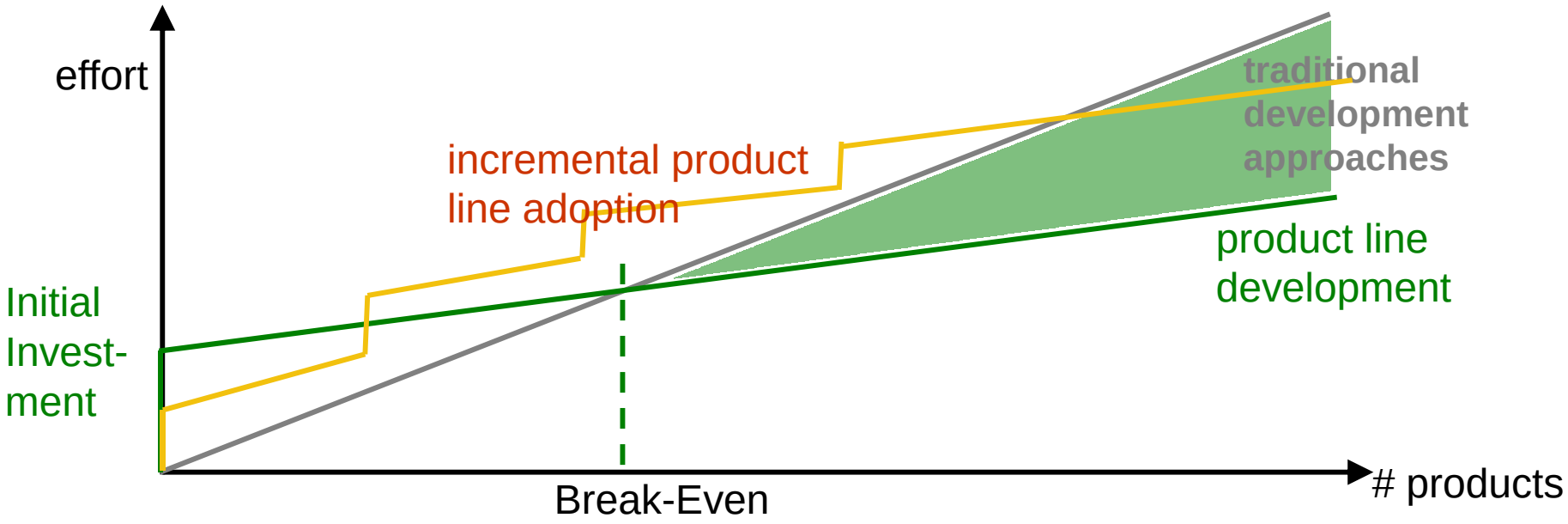
Why is it needed:

- bi-directional dependency of product line infrastructure and strategy
- classical domain engineering approaches lacked this and were over-engineered

Example:

- we set up a new product line: first we analyze the markets and decide on key functionalities that we want to bring to the market

Economics


effort
incremental product line adoption
traditional development approaches
product line development
Initial Invest-ment
Break-Even
products

Contents

Variability management (1)

What does it mean:

- easy addition of variabilities to products
- management of variability over product line life-time

Why is it needed:

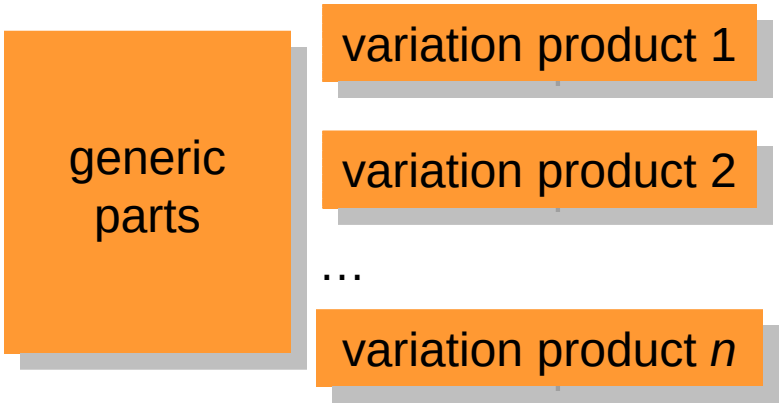
- focus and manage differences among products
- minimal effort for retrieval and adaptation is required to differ from opportunistic reuse
- artifacts over life-cycle are closely connected

Example:

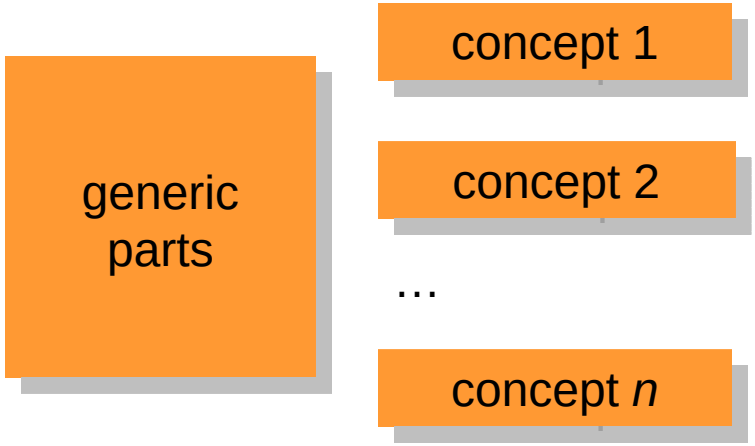
- we want to add a calendar feature in a product;
  this has been around before, thus we would like to select it (no searching) and it should be included in the final product

Variability management (2)

product-oriented variation



concept-based variation

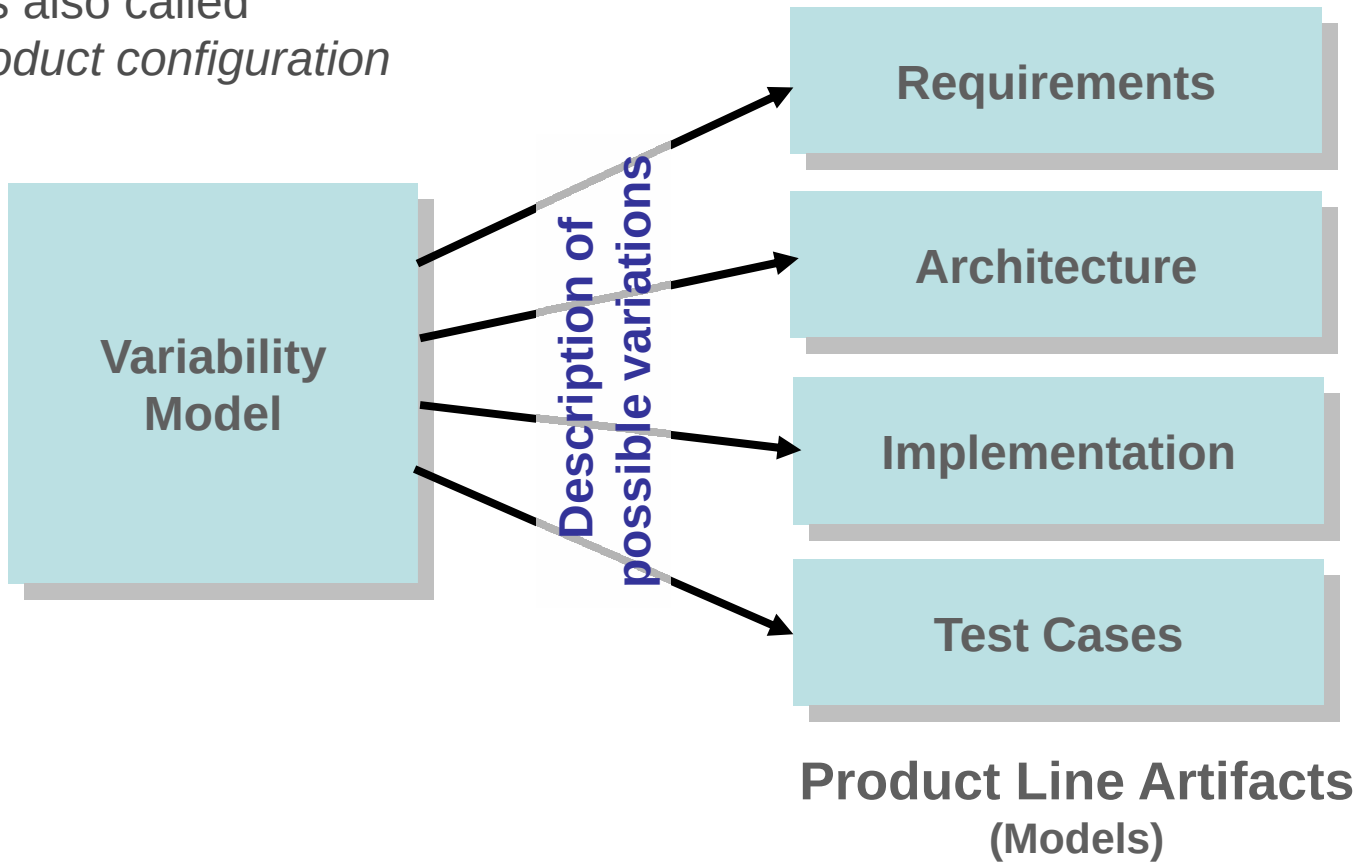


Product Definition
product 1: concept 1,2, ,4,5, ,…
product 2: concept 1, ,3,4, ,6,…

Relation of Variability Model and Product Line Artifacts

This is also called

product configuration



Contents

Architecture-centric development (1)

What does it mean:

- a common (reference) architecture provides a blue-print for all products

Why is it needed:

- components realizing variabilities can be easily added, deleted, exchanged, etc.
- we keep context and interfaces fixed !

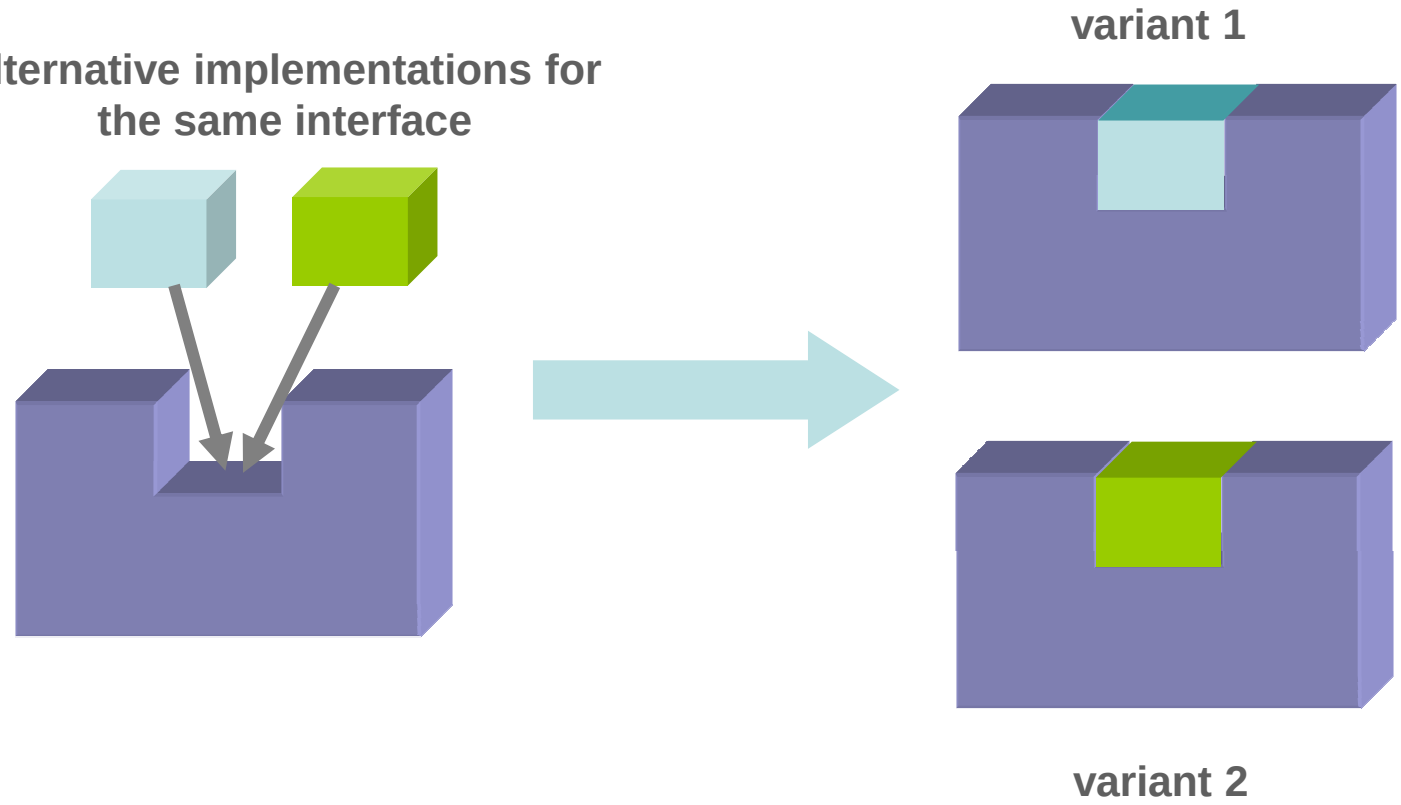
Example:

we are going to develop a mobile phone, different variants will provide different forms of calendar functionality:

we set up a common architecture with a common calendar interface and different components accepting this interface.

Architecture-centric development (2)

alternative implementations for the same interface



Contents

Two-Lifecycle Approach (1)

What does it mean:

- development *with* and *for* reuse are conceptually present
- both are independent life-cycles (e.g., methods, timing, etc.)

Why is it needed:

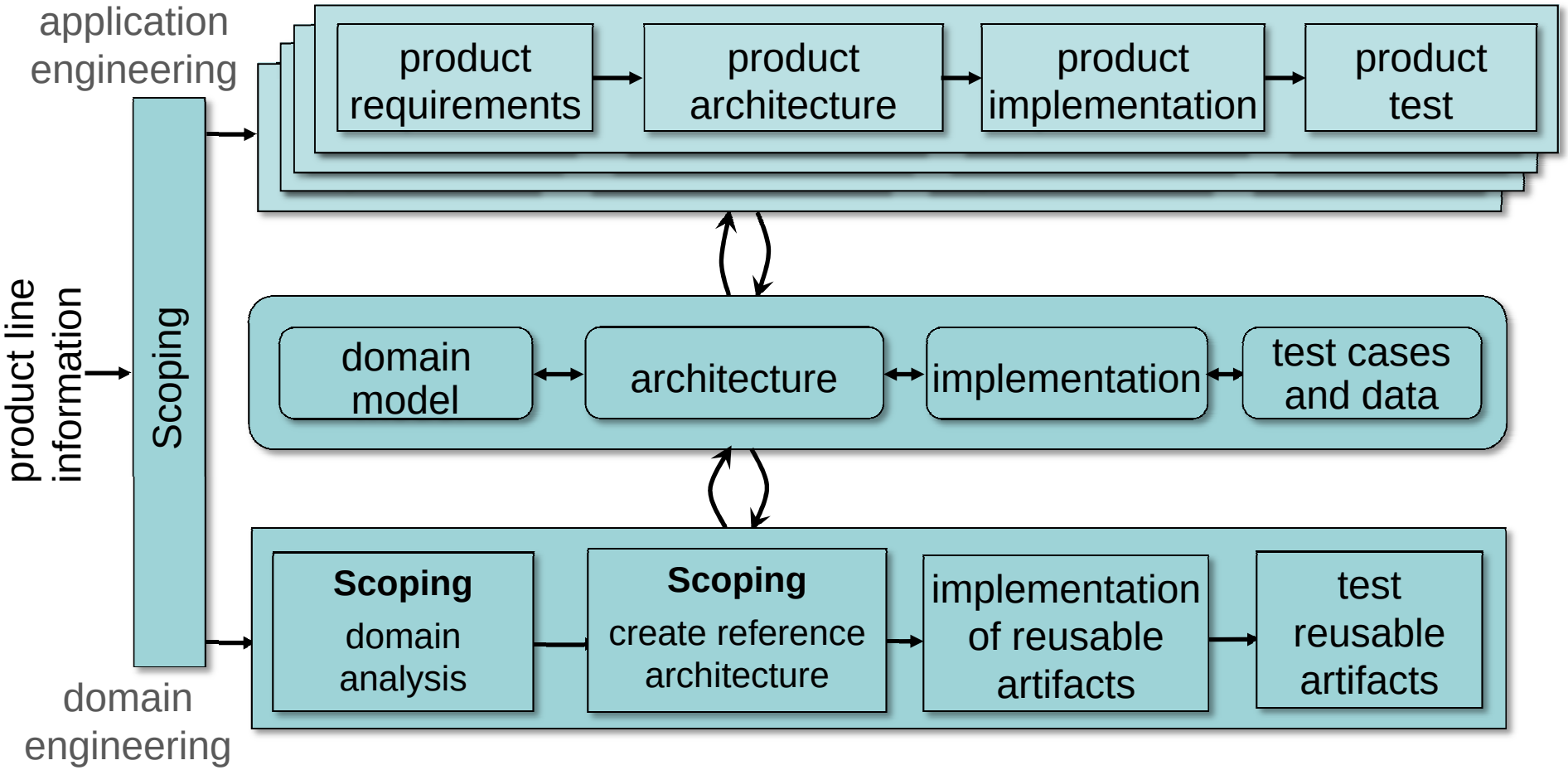
- product line infrastructure engineering should not be overwhelmed by daily product concerns
- needs for long-lived product line and fast customer reaction are different

Example:

- a product line infrastructure may live for several years (decade and more) and may be developed by hundreds of developers
- a specific product development may respond within a week to a customer request and may include less than a dozen developers

PLE Introduction

The Four Pillars / Two-Lifecycle Approach



Process Dimension

The following issues must be addressed:

- *Domain engineering*
  Those processes that perform the Domain Engineering work
- *Application engineering*
  Those processes that perform the Application Engineering work
- *Collaboration*
  Those processes that perform the collaboration activities between Domain and Application Engineering

Note:

- In hierarchical product lines these issues can be iterated as well
- In large organizations mapping these issues on organizational structures becomes very important

Contents

Two Product Line Examples

Context	**market maker**	**Philips Medical**
Size of Organisation	~25	> 1000
Migration mode to PLE	new product line	migration of existing
Localization	one site	world-wide
Domain	financial information	medical systems
Type	web-based	Embedded + desktop
Results		
Cost reduction	break even: ~5	2-4 times
TTM Reduction	2-4	>2
Maintenance cost (reduction)	~60%	
Quality	reduced quality cost	~50% defect density
Issues	incr. of issue res. time	

Contents

Summary

- Product line engineering
  - Focus on set of systems
  - Engineer differences among systems

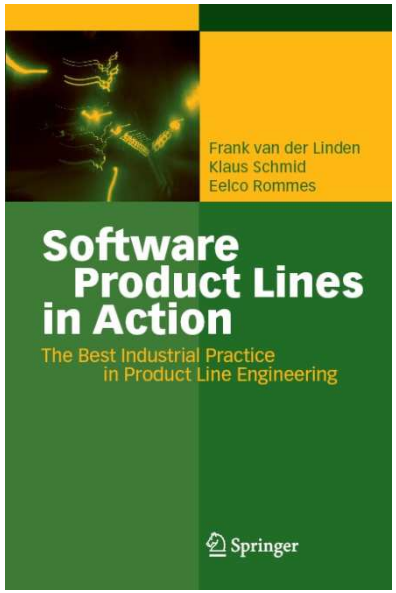
- Principles of software product lines
  - business-oriented
  - variability management
  - architecture-centric
  - two-lifecycle approach

Further Material

- Product line engineering from a practitioner perspective
- Families Evaluation Framework
- Many industrial case studies!

http://www.spl-book.net

Linden, Schmid, Rommes
Product Lines in Action
Springer, 2007



Summary

Questions & Open Issues?