Software Product Line Engineering

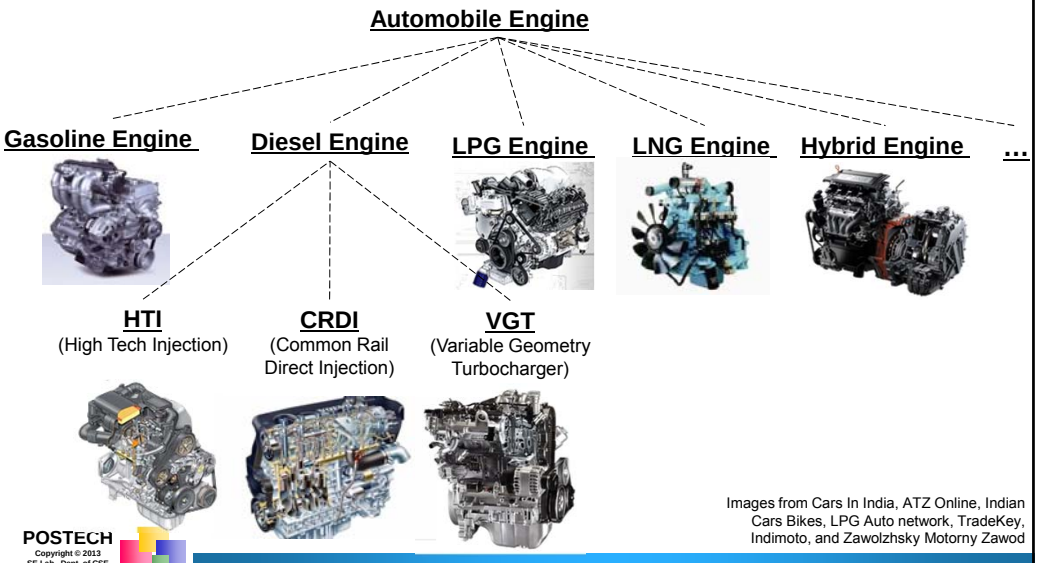
Software Product Line Engineering: A New SE Paradigm

강 교 철
kck@postech.ac.kr

Pohang University of Science and Technology (POSTECH)

Prologue

Variety of automobile engines



Images from Cars In India, ATZ Online, Indian Cars Bikes, LPG Auto network, TradeKey, Indimoto, and Zawolzhsky Motorny Zawod

1

Software Product Line Engineering

Prologue

- Soft software lives long; Hard software has no life
- We are developing software as if it is hardware
- Product line software engineering is about making software soft
- Product line engineering shows what software engineering ought to be

2

Agenda

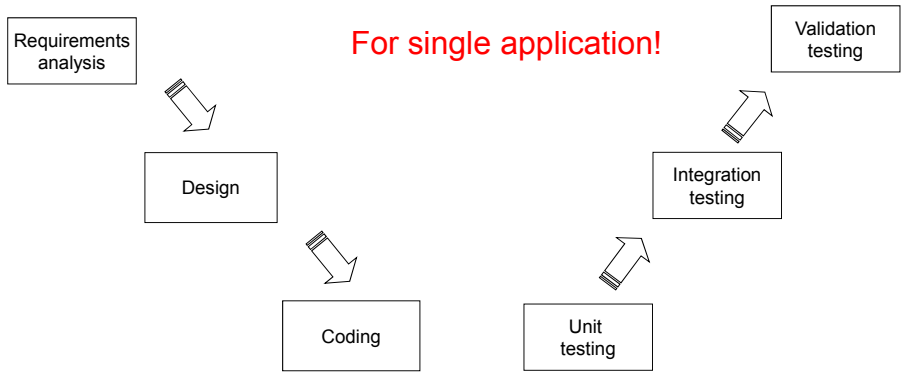
- Traditional Software Engineering
- Product Line Engineering and Variability Management
- Feature-Oriented Product Line Engineering
- Product Line Adoption Issues

3

Software Product Line Engineering

Traditional Software Engineering

- Typical software engineering process



POSTECH

4

Traditional Software Engineering

- Development of single application system
- Requirements analyst
  - Elicit and specify requirements for a single target system
- Designer
  - Design for the specified requirements

POSTECH

5

Software Product Line Engineering

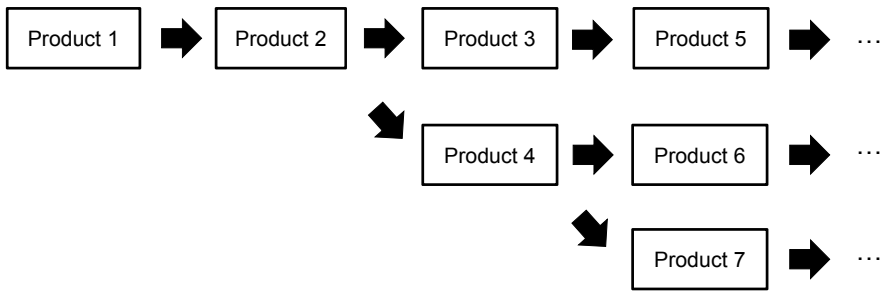
But what happens

- Software maintenance requires
  - Adding new functions
  - Removing existing functions
  - Changing functions
  - Correcting errors and improving performance
- But
  - Source code modification without refactoring
  - Copy-and-modify reuse

6

Traditional Software Engineering

- Copy-and-modify reuse



7

Software Product Line Engineering

As the results

- Proliferation of versions
- Design and code decay
  - Bad structure
  - Spaghetti code
  - Unused code
  - Brittle code
- Software maintenance can cost as much as 80% of the entire lifecycle cost

The problem

- Software is developed like hardware
- There is little effort to build "Softness" into software
- The traditional approach can no longer support development of products with
  - Diverse market needs
  - Time-to-market pressure
  - Fierce feature competition

Software Product Line Engineering

Softness of Software

- Why important
  - Functionality of products implemented as software (e.g., more than 10M lines of code for TV products)
  - Diverse market needs
  - Time-to-market pressure (e.g., new TV models every 3-6 months)
  - Software as valuable asset of an organization
    - Accumulated systems development knowledge is packaged as reusable asset

Softness of Software

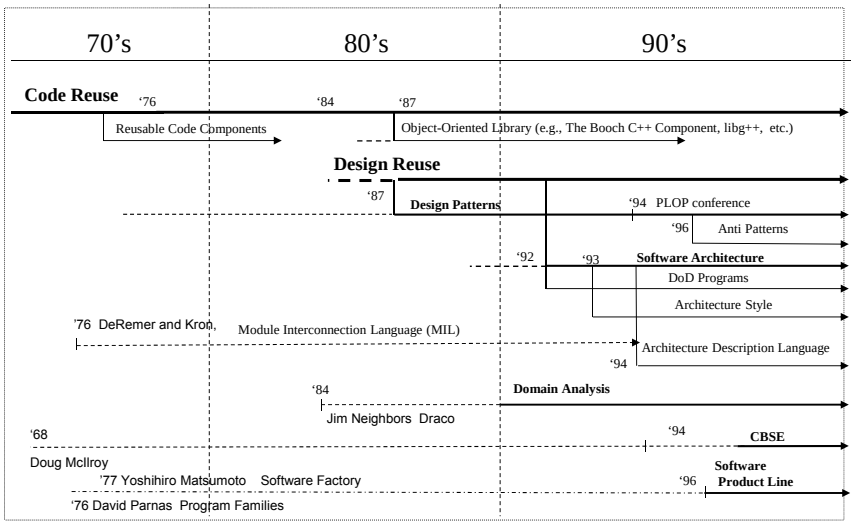
- Softness
  - Maintainability
  - Adaptability
  - Portability
  - Interoperability
  - Reusability

Software Product Line Engineering

Related engineering principles and techniques

- Modularity
- Abstraction
- "Layering"
- David Parnas'
  - Information hiding
  - Program families
- Meta programming and application generators
- There are many mechanisms to use; How do we know what to "hide"

Evolution of Reuse Concepts



Software Product Line Engineering

Agenda

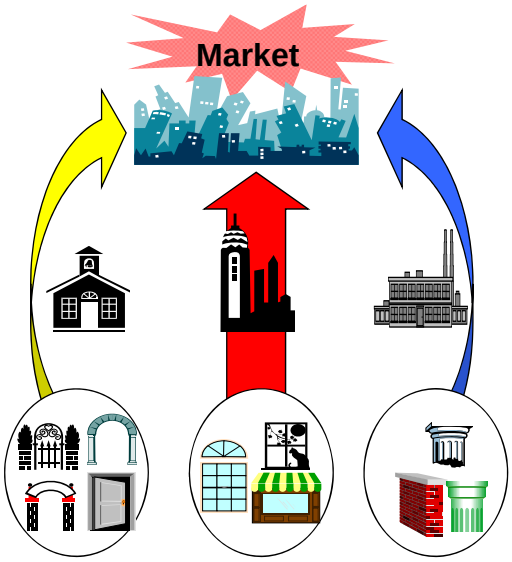
- Maintainability and Reusability
- **Product Line Engineering and Variability Management**
- Feature-Oriented Product Line Engineering
- Product Line Adoption Issues

Product Line Engineering

- Systematic Reuse in the Context of a Product Line: "Building softness into software"
  - Product line: "a family of systems sharing a common set of features"

Software Product Line Engineering

- Software Product Line Engineering (SPLE) is an emerging software engineering paradigm, which guides organizations toward the development of products from core assets rather than the development of products one by one from scratch.

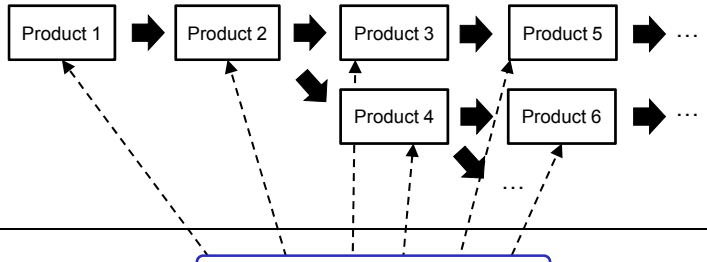


POSTECH

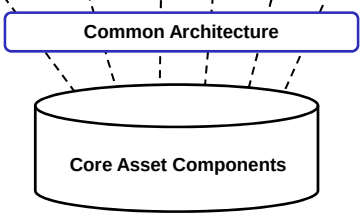
16

Product Line Software Engineering

Traditional Software Engineering



- Development and maintenance of many similar software products
- High software maintenance cost



- Development and maintenance of common architecture and core asset components
- Software products are instantiated from the core asset components based on the common architecture
- High reusability and low software maintenance cost

Software Product Line Engineering

POSTECH

17

Software Product Line Engineering

SEI: Software Product Line Practice

http://www.sei.cmu.edu/plp/essentials/

Celsius Tech

http://www.sei.cmu.edu/plp/essentials/

Software Product Line Engineering

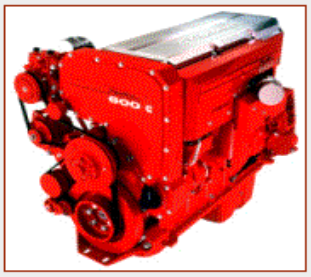
Cummins Inc.

Carnegie Mellon
Software Engineering Institute

Cummins Inc.: Diesel Engine Control Systems

Over 20 product groups with over 1000 separate engine applications

product cycle time was slashed from 250 person-months to a few person-months
Build and integration time was reduced from one year to one week
quality goals are exceeded
customer satisfaction is high
product schedules are met



http://www.sei.cmu.edu/plp/essentials/

- Important technical elements of product line engineering
  - Commonality and Variability
  - Architecture (structure, “bone”)
    - Stability based on common properties
  - Variation points and Variants
    - Flexibility based on expected variations
  - Encapsulation of design decisions that may change
    - Information hiding, abstraction, etc.

Software Product Line Engineering

Technical Advances

- Paradigm change
  - From single systems to product line/family
  - "Good software engineering" focusing on maintainability and reusability
- Commonality and variability analysis
  - Feature analysis
- Domain-oriented Architectures and Components (from objects and collaborations)
  - Variation points and variants
  - "high option potentials"
- Domain specific languages and generators

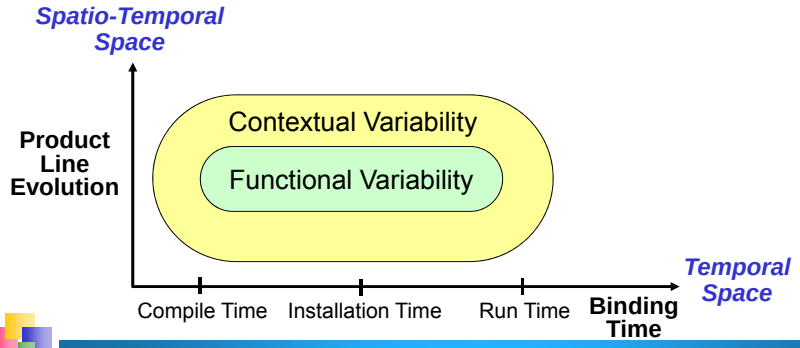
Basic Principles

- **How do we develop product line software?**
  - **Variability analysis: Think about what may change!**
    - **Looking across a family of applications in the product line**
      - **In different markets**
    - **Looking ahead for anticipated changes**
      - **Expected requirements from a market analysis**
      - **Emerging markets**
      - **Study of emerging technologies**
    - **Exploring product usage contexts**
      - **Both static and dynamic dimensions**
  - **Building variability into software**
    - **Architecture: Variation points and variants**
    - **Component: Hide (encapsulate) design decisions that may change!**
      - **Abstract and expose unchanging functional properties as interface!**

Software Product Line Engineering

Commonality and Variability

- Functional Variability
- Contextual Variability
- Binding Time Variability
- Evolution of Functional and Contextual Variability

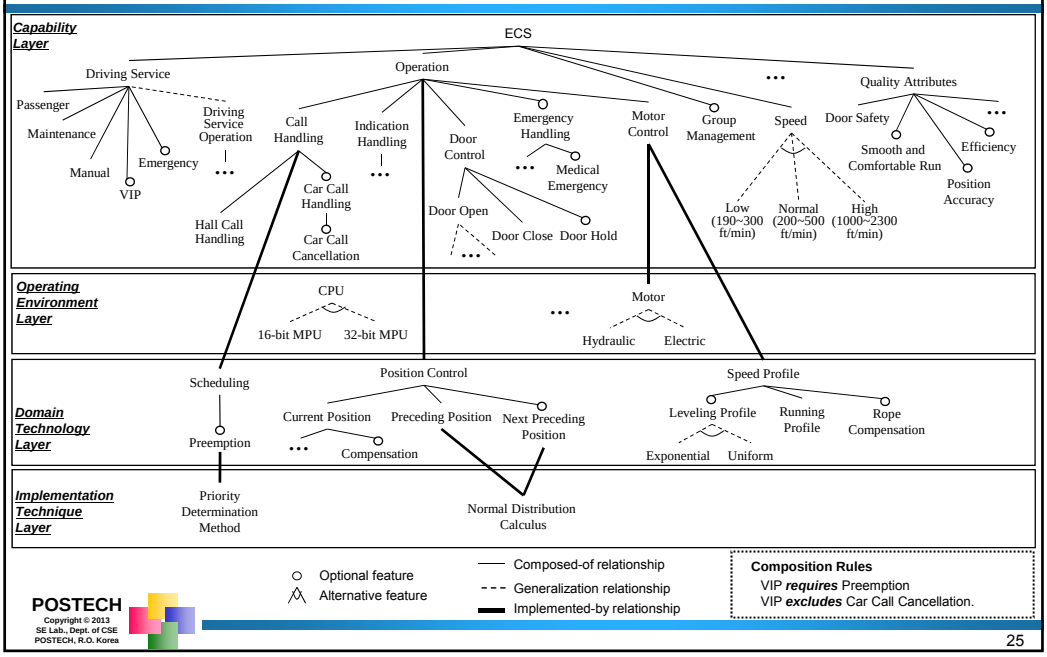


POSTECH
Copyright © 2013 SE Lab., Dept. of CSE POSTECH, R.O. Korea

24

Feature Model Example: ECS Product Line

Functional Variability



POSTECH
Copyright © 2013 SE Lab., Dept. of CSE POSTECH, R.O. Korea

25

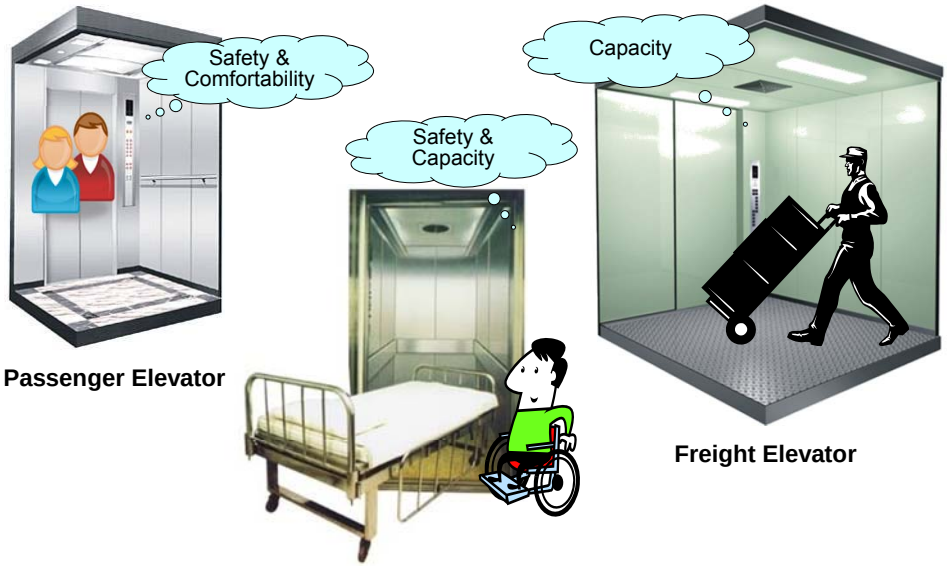
Software Product Line Engineering

Contextual Variability

- Product contexts
  - Different/evolving operating environments (e.g., technologies)
  - Different legal and cultural constraints
  - Different marketing strategies
  - Different/ evolving market needs
  - Dynamically changing usage contexts

Usage Context Example: Elevators

Contextual Variability



Software Product Line Engineering

Three Perspectives of Feature Binding

Binding Time Variability

What features will be included in the products?

When they will be included?

- Asset or product development time

- Pre-operation time

- Operation time

When they will be available?

- Asset or product development time

- Pre-operation time

- Operation time

28

Feature-Oriented Domain Analysis (FODA)

Technical Report
CMU/SEI-90-TR-21
ESD-90-TR-222
November 1990

Feature-Oriented Domain Analysis
(FODA)
Feasibility Study

Kyo C. Kang
Sholom G. Cohen
James A. Hess
William E. Novak
A. Spencer Peterson

Domain Analysis Project

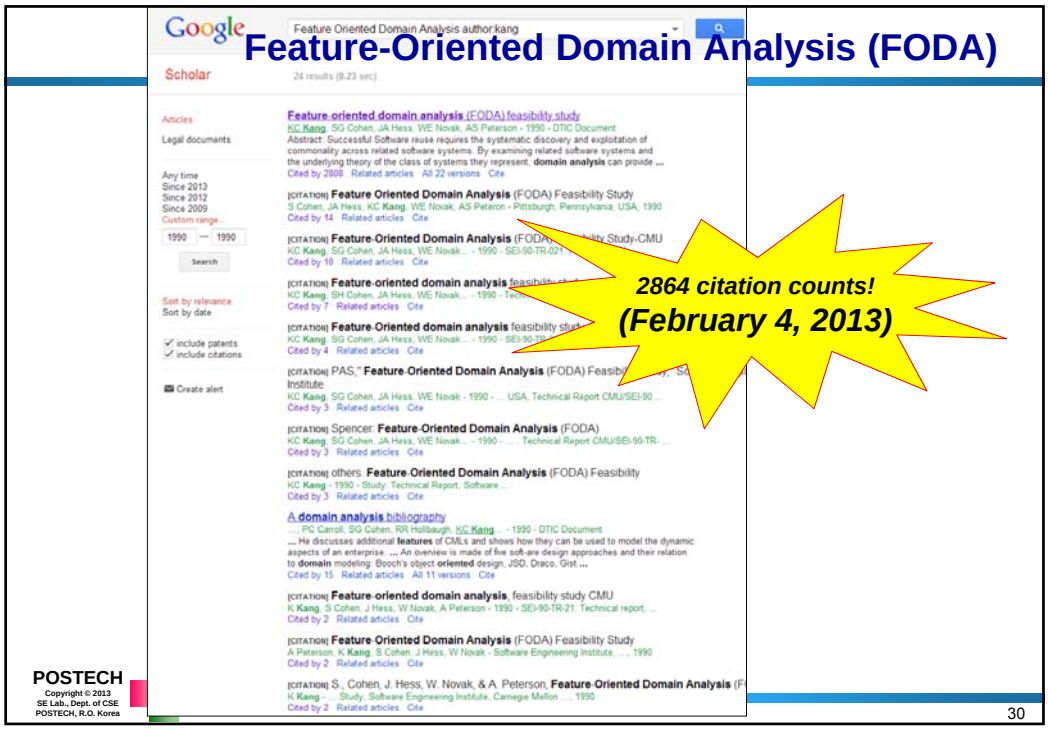
Approved for public release.
Distribution unlimited.

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

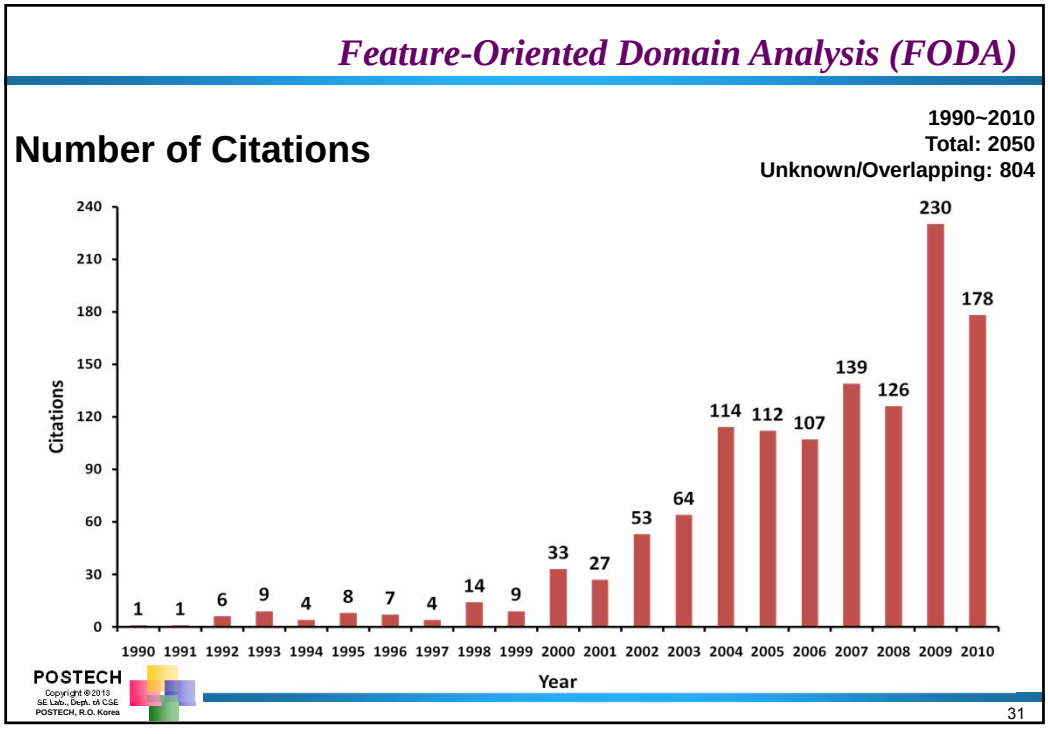
29

Software Product Line Engineering

Feature-Oriented Domain Analysis (FODA)



Feature-Oriented Domain Analysis (FODA)

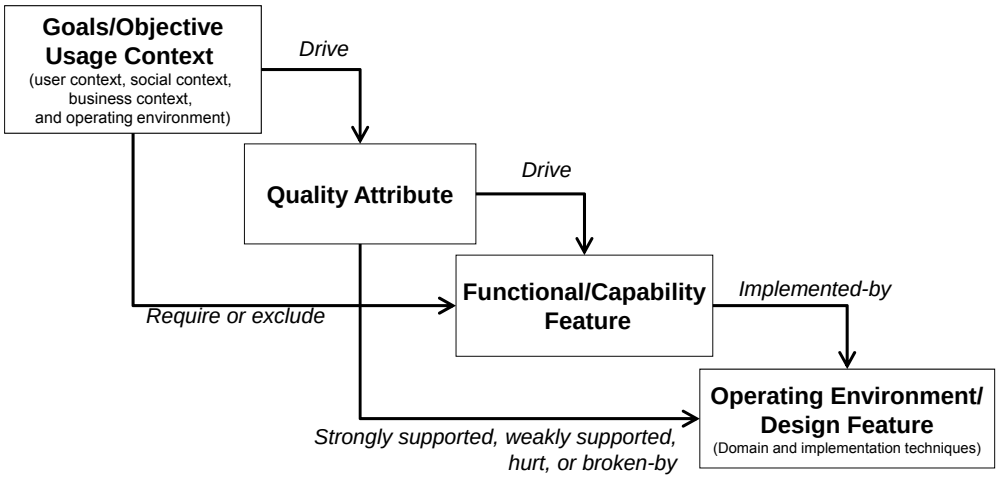


Software Product Line Engineering

Feature-Oriented Domain Analysis (FODA)

32

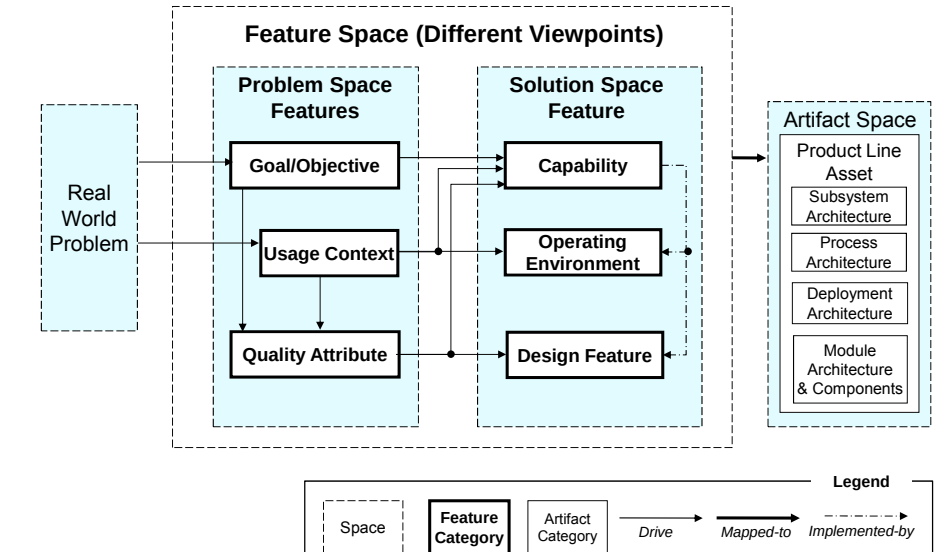
Product Line Feature Modeling



33

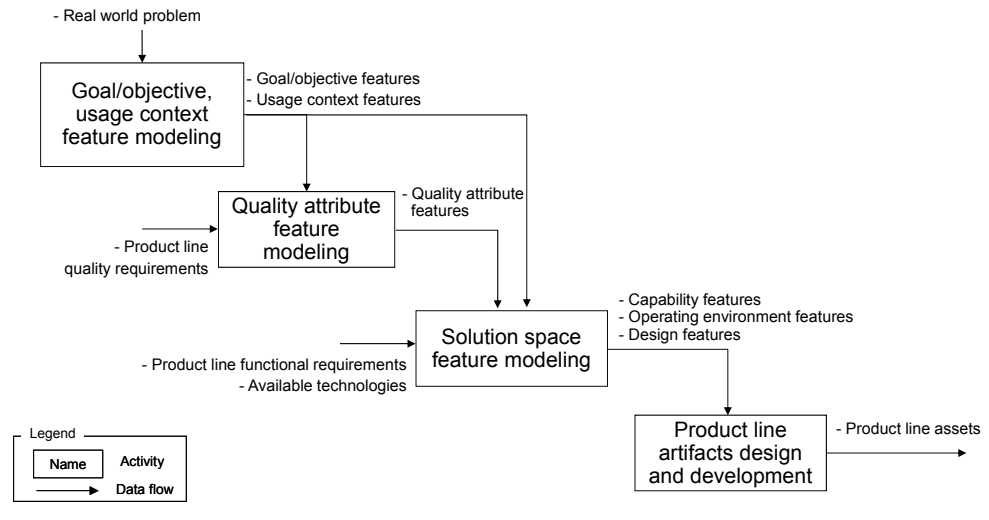
Software Product Line Engineering

Feature Space Overview



34

Modeling and Development Process



35

Software Product Line Engineering

Example: Elevator Control System

Real World Problem

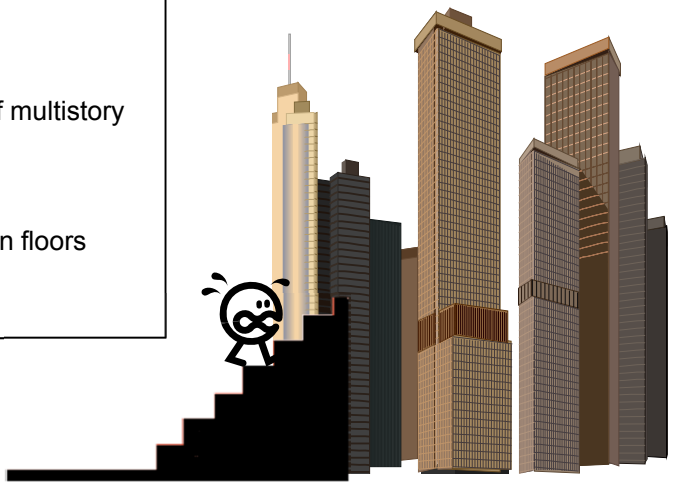
Real World Problem:

Context:

The number of floors of multistory buildings is increasing.

Problem:

Moving objects between floors becomes difficult.



POSTECH

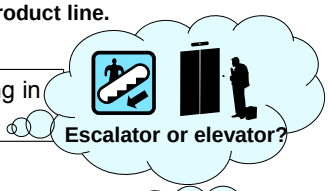
36

Example: Elevator Control System

Problem Space Features

- **Goal/objective features**
  - **Represent what a system should achieve in order to solve real world problems.**
  - **Important to determine the scope of a product line.**
    - **Product line analysts, market analysts, and developers can establish an explicit boundary of a product line and can share a common understanding about the ultimate goals of the product line.**

"Moving objects between different floors of a building in an efficient way."



Refined to

"Moving objects between different floors of a building ***vertically*** in an efficient way ***using a cage with doors***."

POSTECH

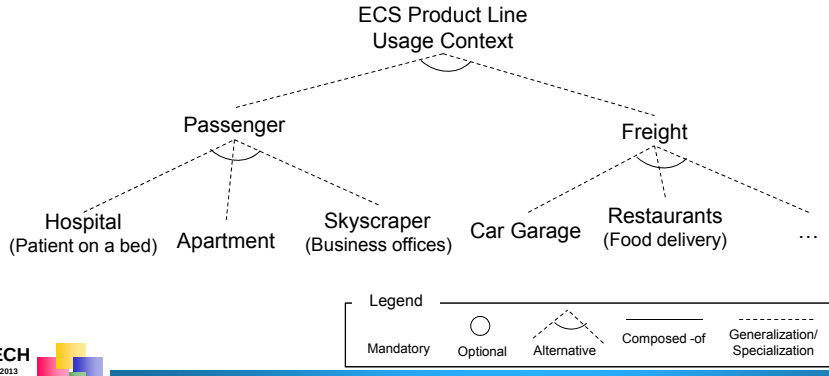
37

Software Product Line Engineering

Example: Elevator Control System

Problem Space Features

- **Usage context features**
  - **Represent any contextual setting in which a product is deployed and used.**
  - **Include physical environments, user profiles, social or legal issues, business concerns, etc.**



38

Example: Elevator Control System

Problem Space Features

- **Goal/objective features and usage context features drive quality attribute features.**

Usage Context Feature		Refined Goal/Objective Feature	Important Quality Attribute Feature
Passenger		Moving people between different floors of a building vertically in an efficient way using a cage with doors.	Door Safety, Usability
	Hospital	Moving people (*e.g.* patients, doctors, and nurses), wheelchairs or hospital beds between different floors of a hospital vertically in an efficient way using a cag e with doors.	Smooth and Comfortable Run, Position Accuracy
	Skyscraper	Move people (*e.g.* workers or travelers) between diff erent floors of a skyscraper vertically in an efficient w ay using a cage with doors.	High Speed
Freight		Move freight between different floors of a building ve rtically in an efficient way using a cage with doors.	Freight Damage Prevention, Low Speed, Position Accuracy

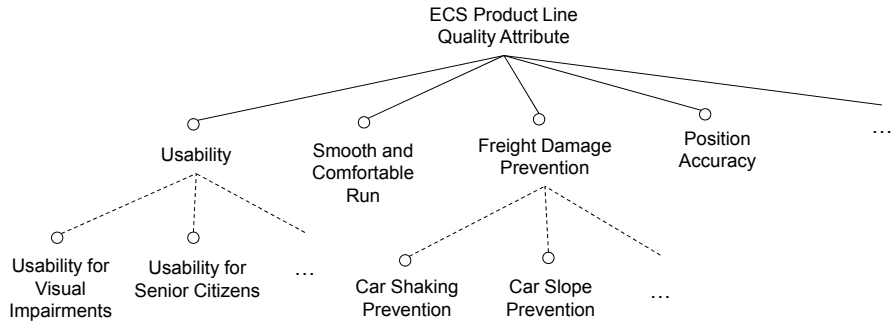
39

Software Product Line Engineering

Example: Elevator Control System

Problem Space Features

- **Quality attribute features**
  - **Represent non-functional requirements that a system should meet along with its functional requirements.**



Example: Elevator Control System

Problem Space Features

- **Goal/objective features and usage context features drive quality attribute features.**
  - Mappings between usage context (UC) features and quality attribute (QA) features:

QA Features / UC Features		Door Safety	Usability	Smooth and Comfortable Run	Position Accuracy	High Speed	Freight Damage Prevention	Low Speed
Passenger		V	V					
	Hospital	V	V	V	V			
	Skyscraper	V	V			V		
Freight					V		V	V

Software Product Line Engineering

Example: Elevator Control System

Solution Space Features

- **Goal/objective features and usage context features drive solution space features.**

Usage Context Feature	Refined Goal/Objective Feature	Solution Space Feature
Passenger	Moving people between different floors of a building vertically in an efficient way using a cage with doors.	Car Call Handling, Elevator Air Conditioner Control, Elevator Light Control
Passenger – Hospital	Moving people (*e.g.* patients, doctors, and nurses), wheelchairs or hospital beds between different floors of a hospital vertically in an efficient way using a cage with doors.	Medical Emergency, Door Hold, Low Speed (About 190 ~ 300 Ft/Min), Large Capacity for Hospital Bed (About 1000 ~ 2500kg)
Passenger – Skyscraper	Move people (*e.g.* workers or travelers) between different floors of a skyscraper vertically in an efficient way using a cage with doors.	Double-Deck, High Speed (About 1000 ~ 2300 Ft/Min)
Freight	Move freight between different floors of a building vertically in an efficient way using a cage with doors.	Door Hold, Low Speed (About 60 ~ 190 Ft/Min)
Freight – Car	Move car between different floors of a building (*e.g.* parking garage or manufacturer's storage) vertically in an efficient way using a cage with doors.	Car Call Handling, Elevator Light Control, Large Capacity for Car (About 3000 ~ 5000 Kg)

42

Example: Elevator Control System

Problem Space Features

- **Goal/objective features and usage context features drive solution space features.**
  - Mappings between usage context (UC) features and solution space (SS) features:

SS Features / UC Features	Car Call Handling	Elevator Air Conditioner Control	Medical Emergency	Double-Deck	Door Hold	Low Speed (About 60 ~ 190 Ft/Min)	Large Capacity for Car (About 3000 ~ 5000 Kg)
Passenger	V	V					
Hospital	V	V	V		V		
Skyscraper	V	V		V			
Freight					V	V	
Car	V				V	V	V

43

Software Product Line Engineering

Example: Elevator Control System

Solution Space Features

- **Quality attribute features drive solution space features.**

Solution Space Features \ Quality Attribute Features		Position Accuracy	Smooth and Comfort Run	Freight Damage Prevention	Low Cost
Calculation	Absolute Position (Optional)	++			-
	Relative Position (Optional)	+			+
Compensation of Current Position		+			-
Leveling Profile			+		
	Exponential Profile (Alternative)		++		
	Uniform Profile (Alternative)		+		
Rope Compensation			+		
Weight Balancing Control				++	

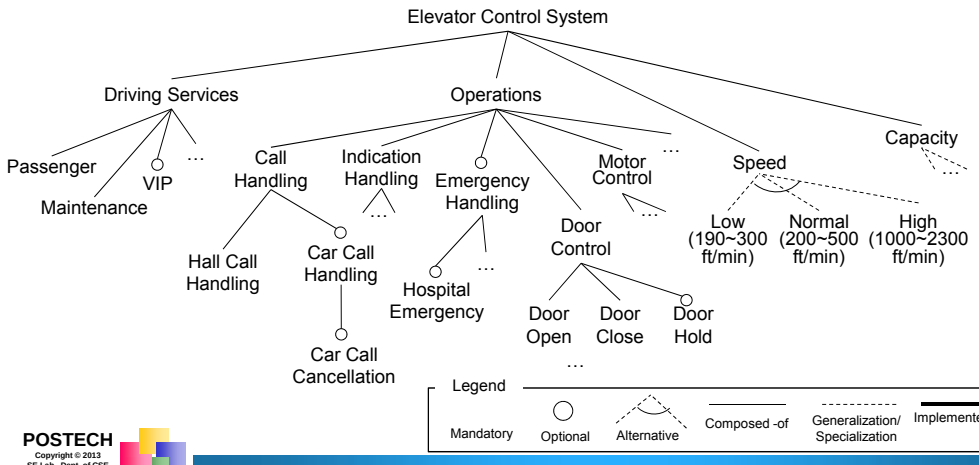
++ (strongly support), + (weakly support), - (hurt), -- (break)

44

Example: Elevator Control System

Solution Space Features

- **Capability features**
  - **Represent end-user visible characteristics of system such as service, operation and function.**



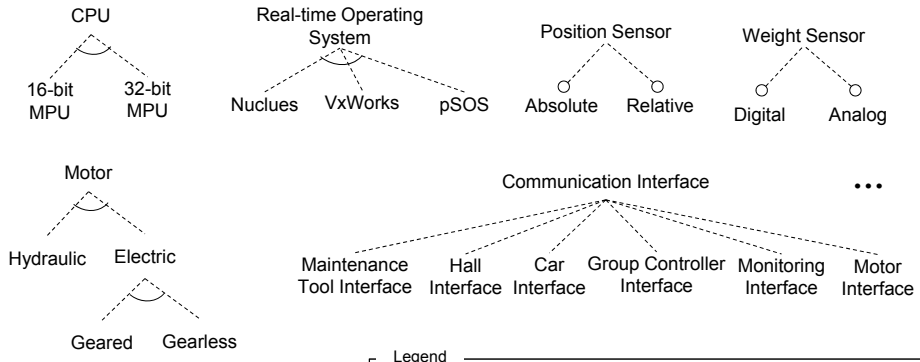
45

Software Product Line Engineering

Example: Elevator Control System

Solution Space Features

- **Operating environment features**
  - **Represent target environments where a product is deployed and operated in.**

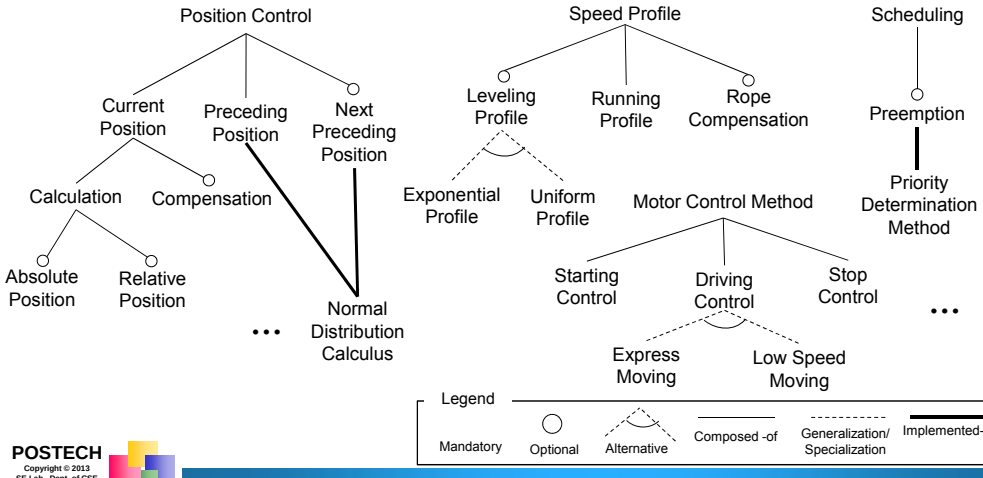


46

Example: Elevator Control System

Solution Space Features

- **Design features**
  - **Represent domain technologies and implementation techniques.**



47

Software Product Line Engineering

Example: Elevator Control System — *Solution Space Features*

- **Capability (CA), operating environment (OE), and design (DE) features are mapped each other.**
  - Mappings between CA features and OE features:

CA Features \ OE Features	CPU	Real-time OS	Position Sensor	Weight Sensor	Motor	Hall Interface	Car Interface
Elevator Control System	V	V					
Run Control			V	V	V		
Hall Call Handling						V	
Car Call Handling							V

  - Mappings between CA features and DE features:

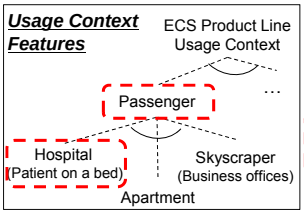
CA Features \ DE Features	Position Control	Weight Balancing Control	Speed Profile	Scheduling	Motor Control Methods
Run Control	V	V	V		
Call Handling				V	
Motor Handling					V

POSTECH

48

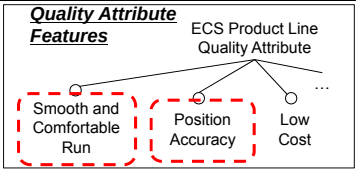
Example: Elevator Control System — *Feature Selection*

Usage Context Features



UC Features \ QA Features	Door Safety	Usability	Smooth and Comfortable Run	Position Accuracy
Passenger	V	V		
Hospital	V	V	V	V
Skyscraper	V	V		

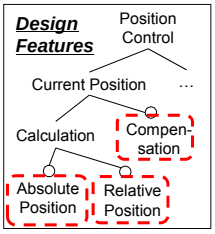
Quality Attribute Features



Solution Space Features \ QA Features		Position Accuracy	Low Cost
Calculation	Absolute Position (Optional)	++	-
	Relative Position (Optional)	+	+
Compensation of Current Position		+	-

++ (strongly support), + (weakly support), - (hurt), -- (break)

Design Features



POSTECH

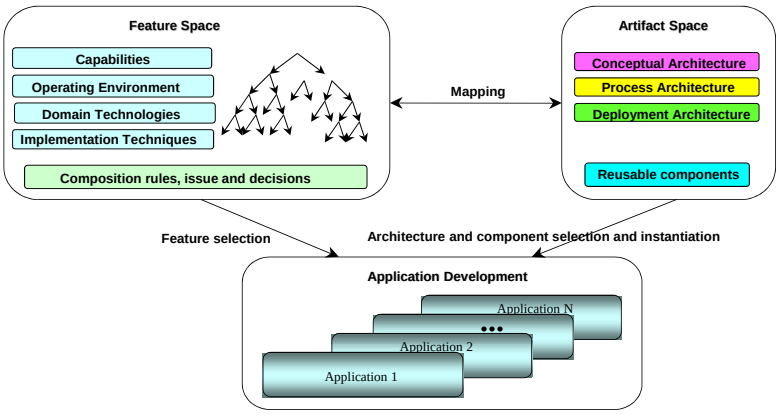
49

Software Product Line Engineering

Agenda

- Maintainability and Reusability
- Product Line Engineering and Variability Management
- **Feature-Oriented Product Line Engineering**
- Product Line Adoption Issues

Method Concept

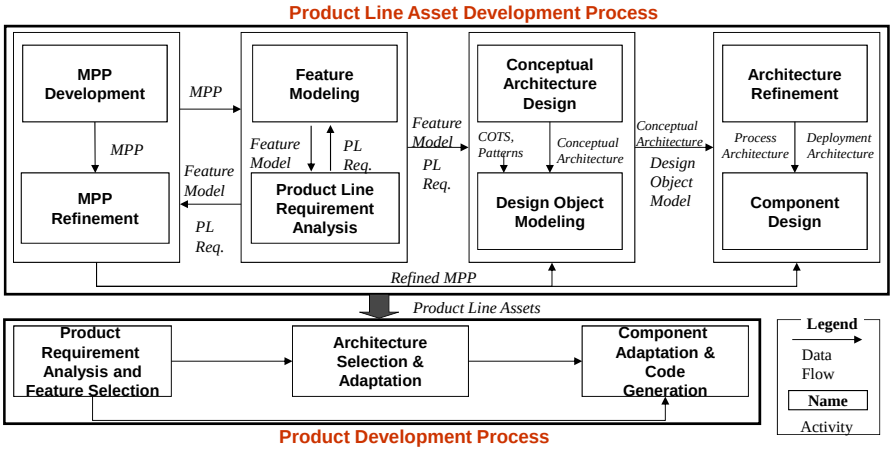


The core of FORM lies in the analysis of domain features and the use of these features to develop reusable and adaptable domain artifacts.

Software Product Line Engineering

Feature-Oriented Product Line Engineering

- FORM (Feature-Oriented Reuse Method)



* MPP: Marketing and Product Plan * PL: Product Line * Req.: Requirements

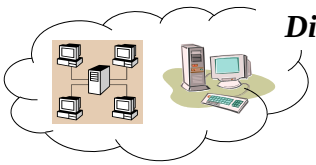
Kyo C. Kang, Jaejoon Lee, and Patrick Donohoe, Feature-Oriented Product Line Engineering, IEEE Software, Vol. 9, No. 4, Jul./Aug. 2002, pp. 58-65.

52

Marketing and Product Plan (MPP)

Different cultural traits and legal constraints

Different computing knowledge backgrounds



Different computing environments

53

Software Product Line Engineering

Marketing Plan

A *marketing plan* includes a market analysis with an assessment of the market, and a marketing strategy with a plan for realizing the business opportunities with products that meets the needs.

The market analysis includes:

- need assessment
- customer profiles
  - end-user skill levels
  - cultural and legal constraints
- business opportunities
  - price range
  - time to market

The marketing strategy may initially include:

- an outline of product delivery methods : how the products will be delivered to customers

Elements of MPP — ***Product Line Initiation***

Marketing Plan (Business Concerns)

Market analysis

- Market segment
  - Needs assessment
  - User profile
  - Cultural and legal constraints
- Business opportunities
  - Time to market,
  - Price range, etc.

Marketing strategy

- Product delivery methods
- Other business considerations

Product Plan (Engineering Concerns)

Product features

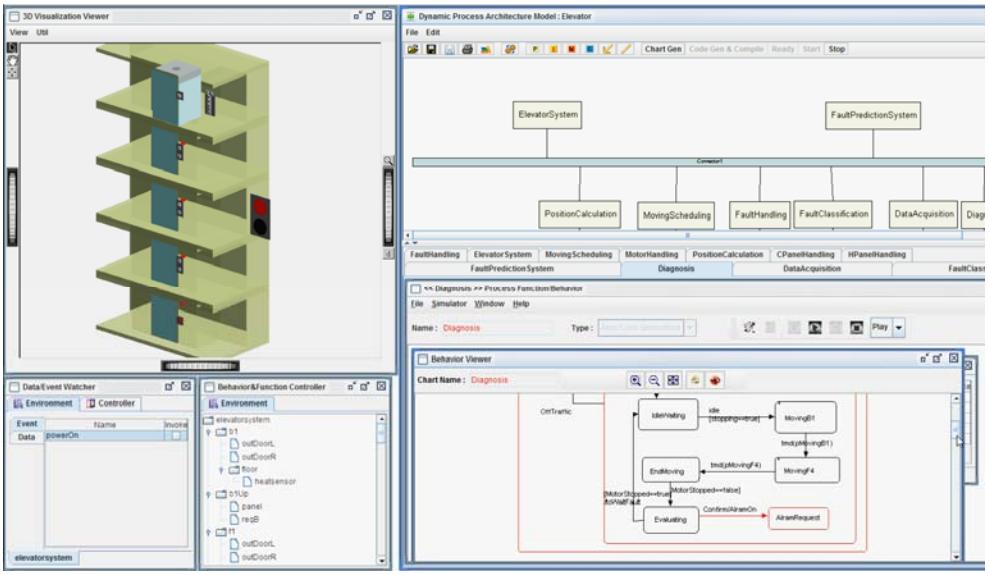
- Product functional features
  - Feature lists
  - Feature description
- Non-functional features
  - Usability, scalability, etc.

Product features delivery methods

- Feature coverage
- Feature binding time
- Feature binding techniques

Software Product Line Engineering

FORM Application: Elevator Control System (ECS) Product Line



Product Plan

Once the marketing plan has been defined, it is important to spend some effort on identifying the characteristics of products in a product line in terms of features and developing a plan for incorporating features.

The *product plan* includes

- Product features:
- Product feature-delivery methods:
- Feature release plan for future:

Software Product Line Engineering

MPP for ECS Product Line

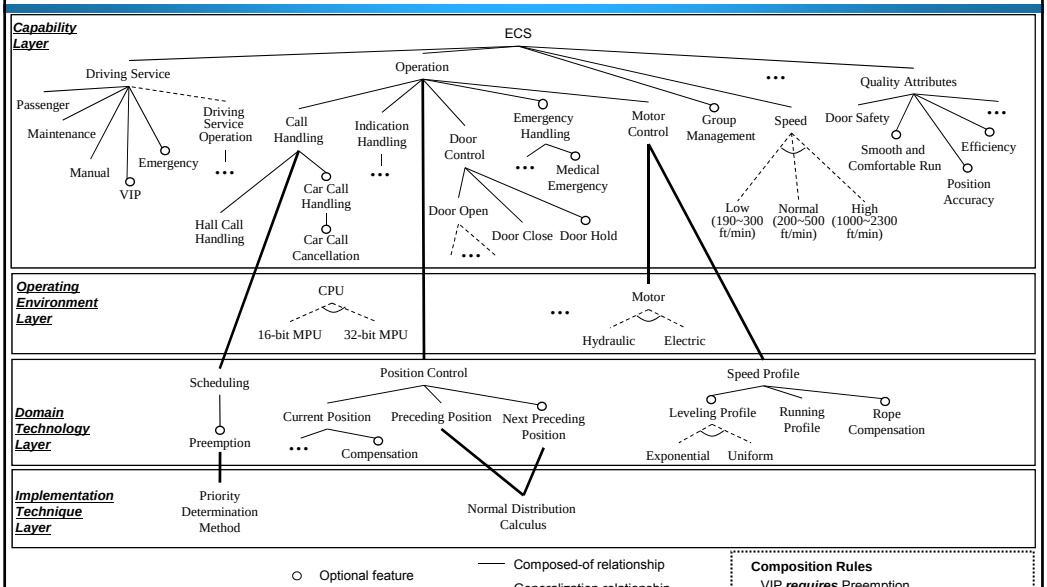
Product Line Initiation

Marketing and Product Plan for ECS Product Line			
Market Segments	**High-Rise Office Building**	**General Hospital**	**Apartment**
User/Maintainer Profile	Dedicated engineers with computer science backgrounds.	No computer knowledge is assumed.	No computer knowledge is assumed.
Legal Constraints	Because an elevator is part of a building, it must comply with standards relating to earthquake resilience, fire standards, electrical wiring rule, and etc.		
Feature Delivery Method	Feature selection from a predefined set of features (Feature Selection Method)	Prepackaged Method	Prepackaged Method
Product Features	Call Handling, Indication Handling, Door Control, Motor Control, Car Call Cancellation, Group Management, etc.	Call Handling, Indication Handling, Door Control, Motor Control, Emergency Driving, Hospital Emergency, etc.	Call Handling, Indication Handling, Door Control, Motor Control, etc.
Quality Attributes	Door Safety, Usability, Efficiency	Door Safety, Usability, Smooth and Comfortable Run, Position Accuracy	Door Safety, Usability
Product Feature Binding Time	Product Delivery Time	Product Build Time	Product Build Time

58

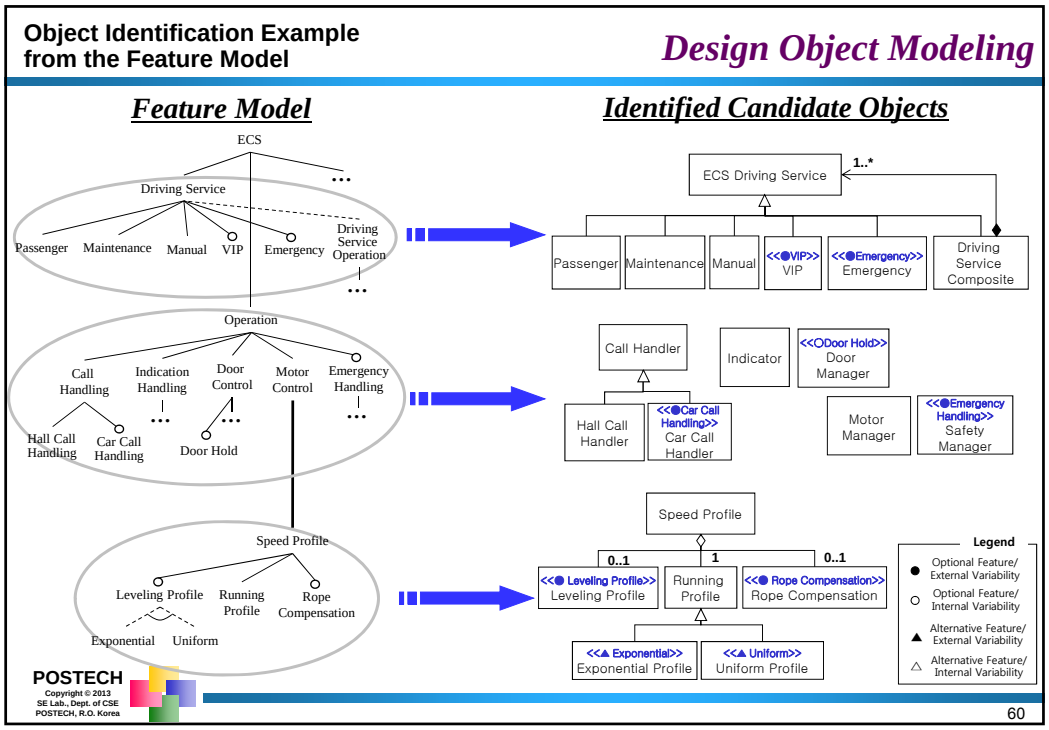
Feature Model Example: ECS Product Line

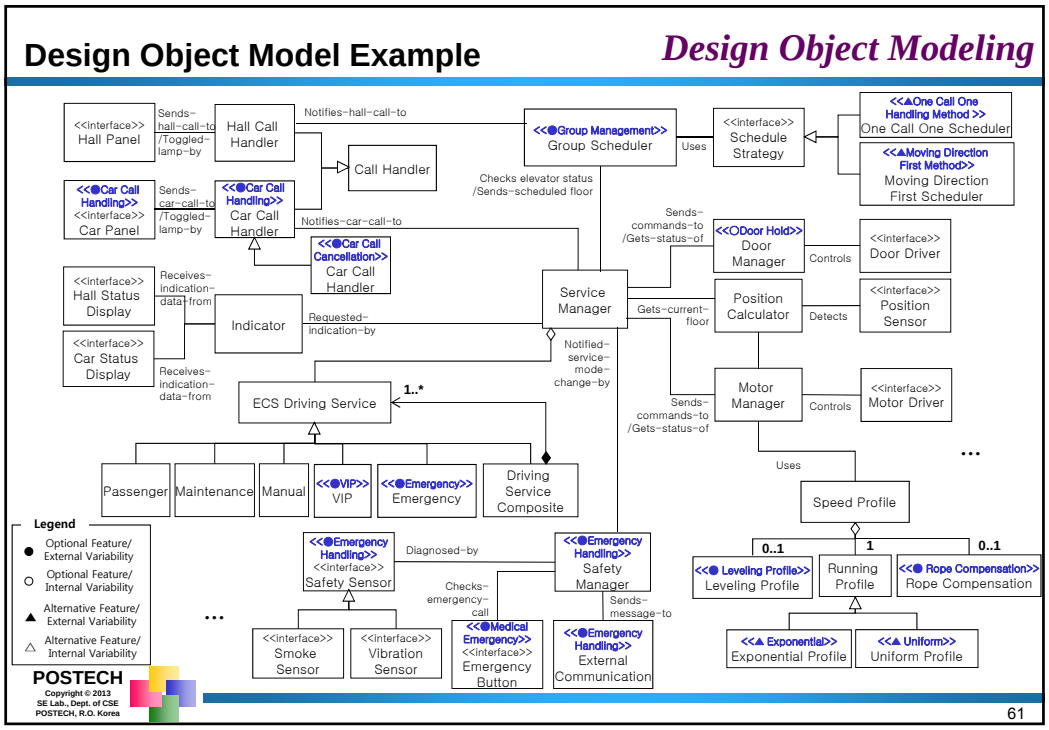
Product Line Analysis



59

Software Product Line Engineering





Software Product Line Engineering

Feature Binding Analysis

- Feature binding: When and how features are included to products and delivered to customers.

- Asset or product development time

- Pre-operation time (delivery, installation, etc.)

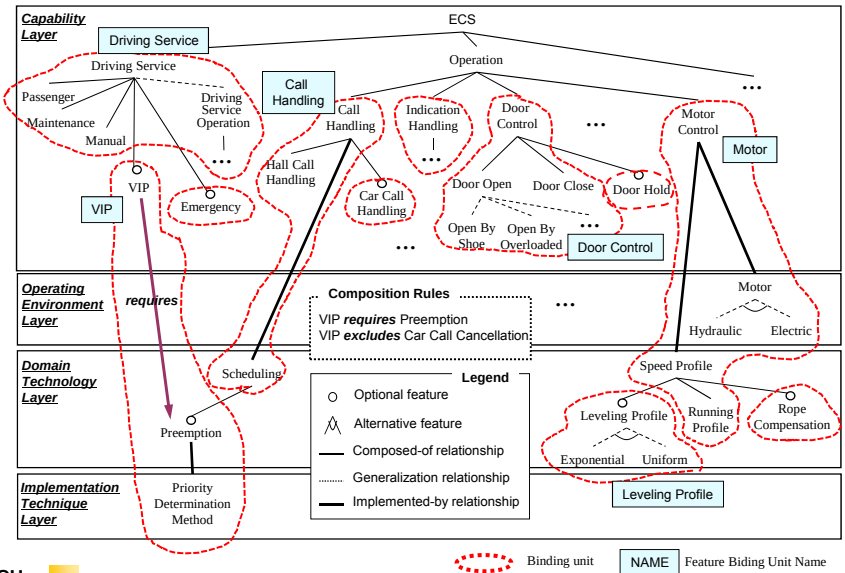
A Design Driver

PL Engineer

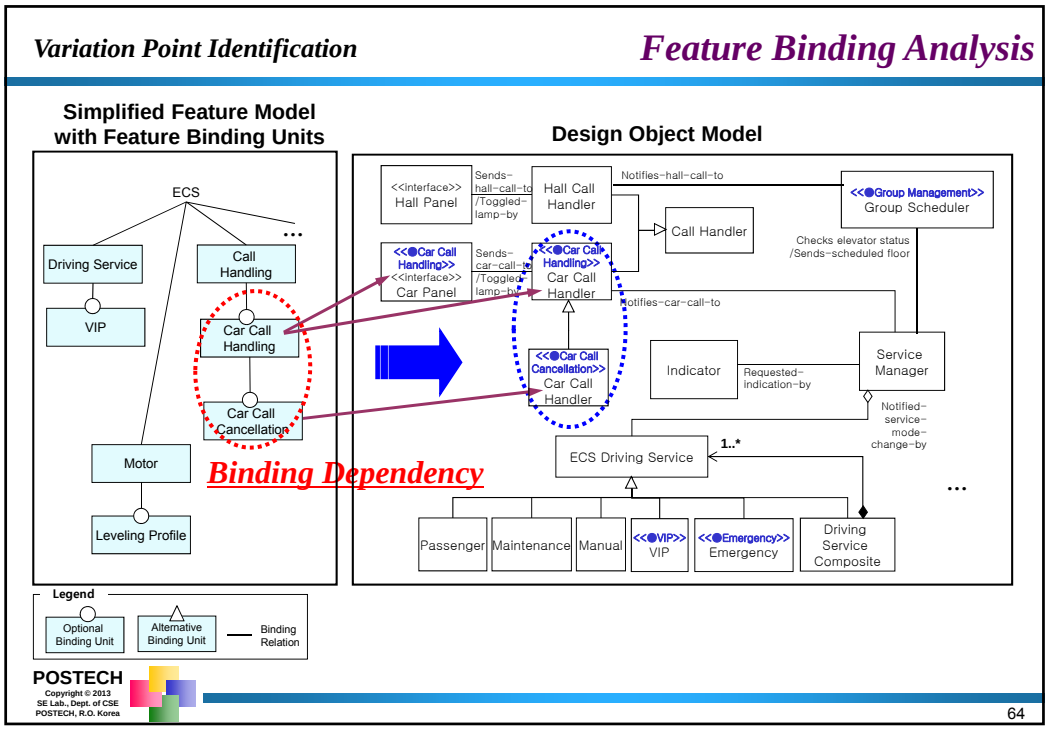
- Operation time

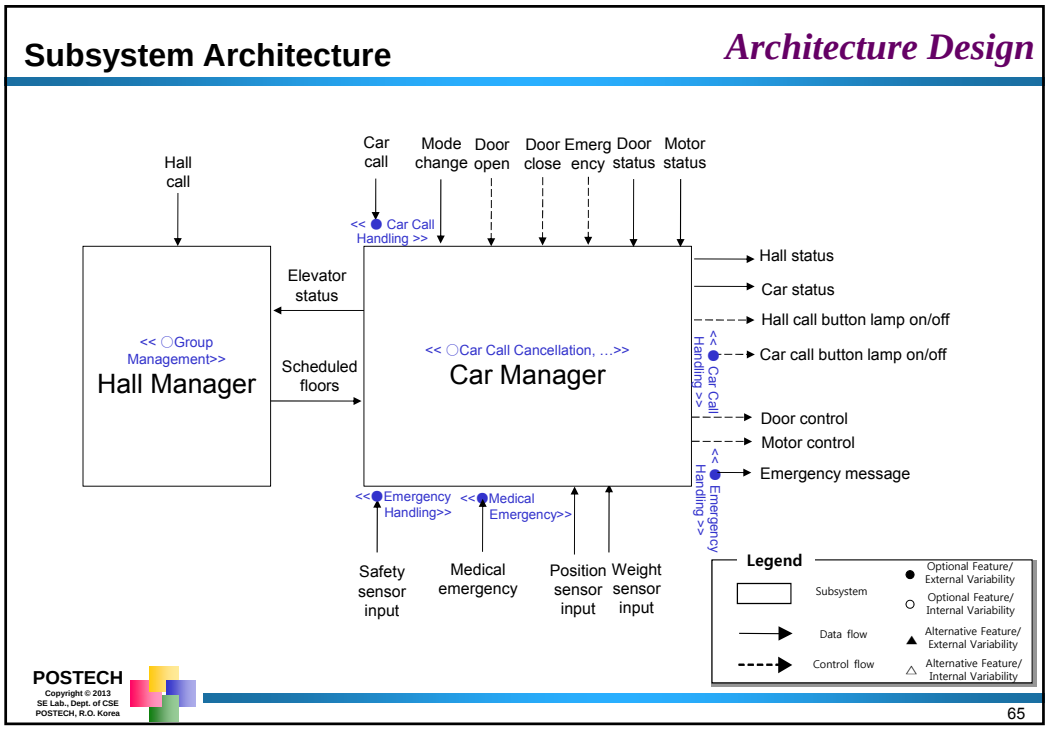
Feature Binding Units

Feature Binding Analysis



Software Product Line Engineering

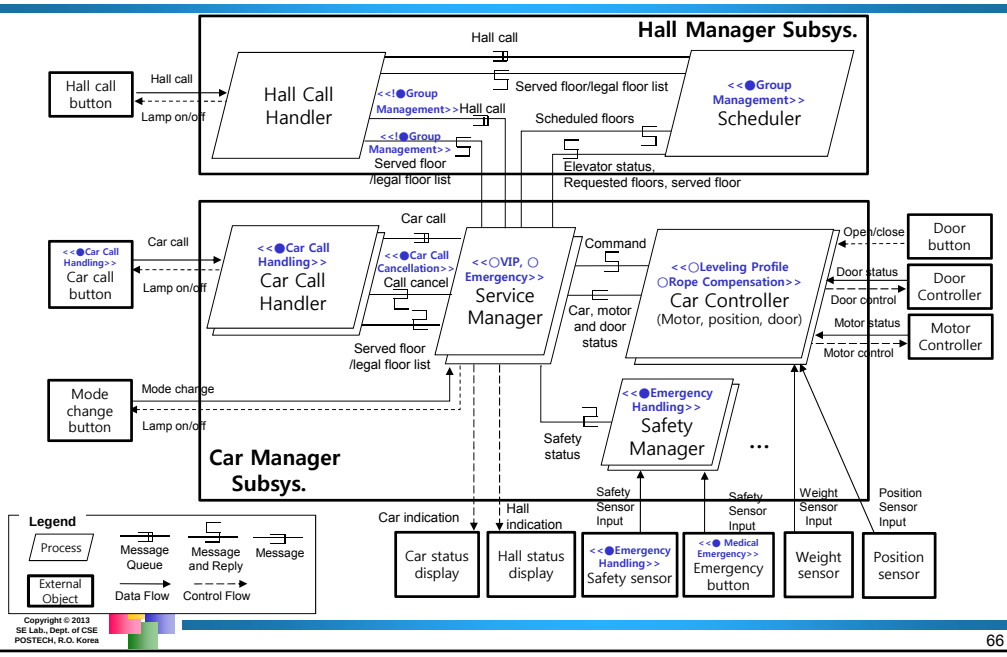




Software Product Line Engineering

Process Architecture

Architecture Design



66

What is Product Development?

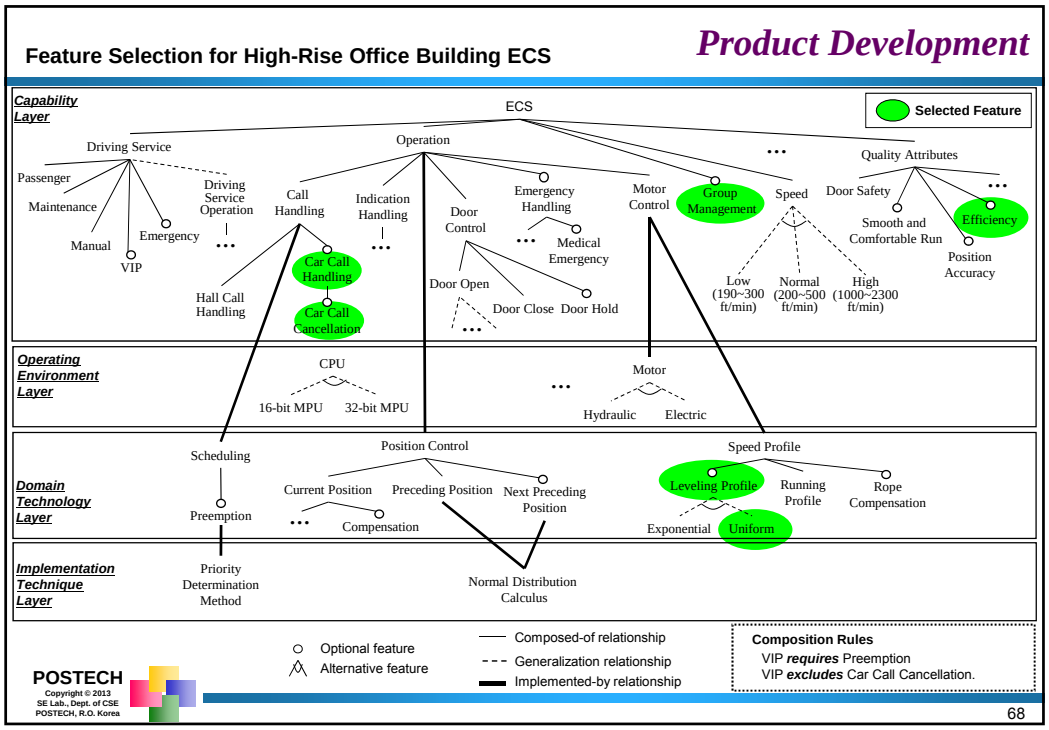
Product Development

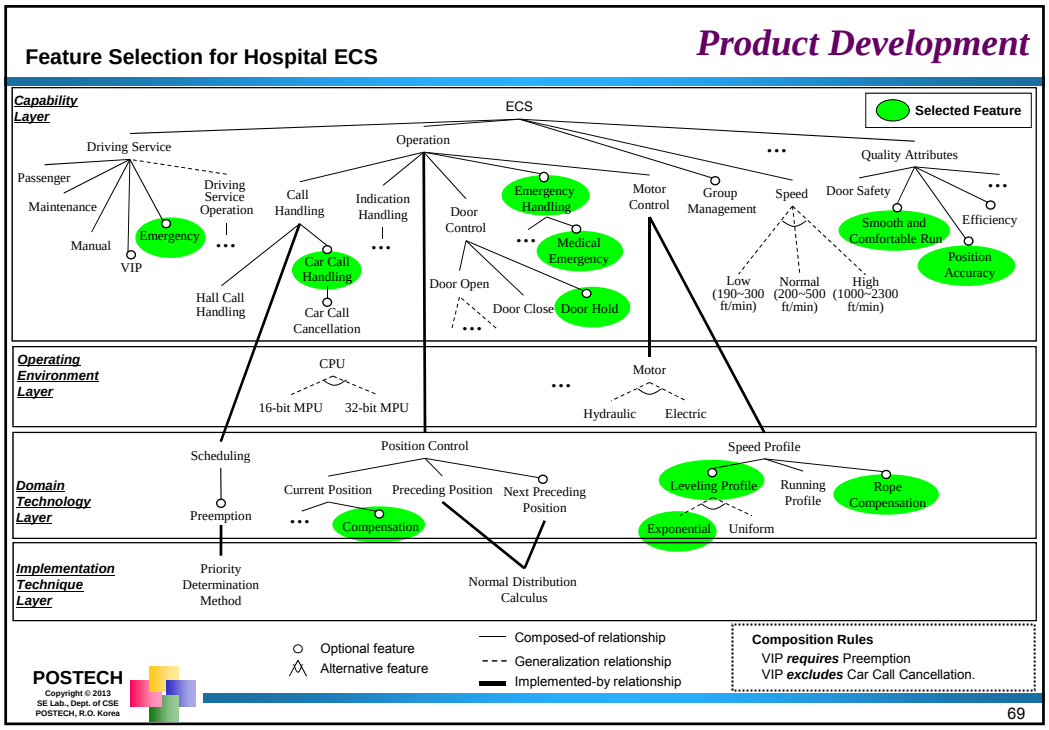
- Product development is a process of developing a specific product making use of the product line asset developed during the product line asset development process.
- Product development proceeds by:
  - analyzing user's requirements,
  - selecting appropriate and valid product line features from the feature model,
  - identifying the corresponding architecture models,
  - completing the product development by reusing software components, and
  - adapting components as needed.

POSTECH
Copyright © 2013 SE Lab., Dept. of CSE POSTECH, R.O. Korea

67

Software Product Line Engineering





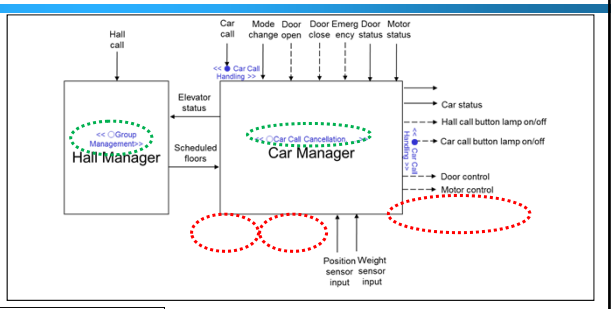
Software Product Line Engineering

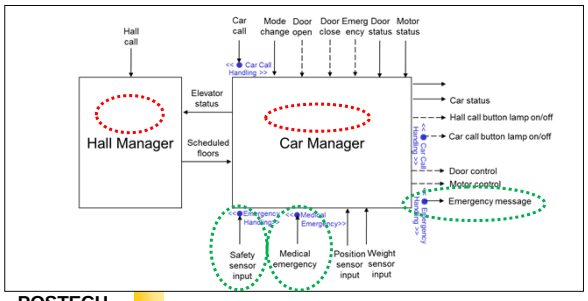
Subsystem Architecture Selection

Product Development

With the **Group Management** and **Car Cancellation** features,

without the **Emergency Handling** and **Medical Emergency** features





With the **Emergency Handling** and **Medical Emergency** features,

without the **Group Management** and **Car Cancellation** features

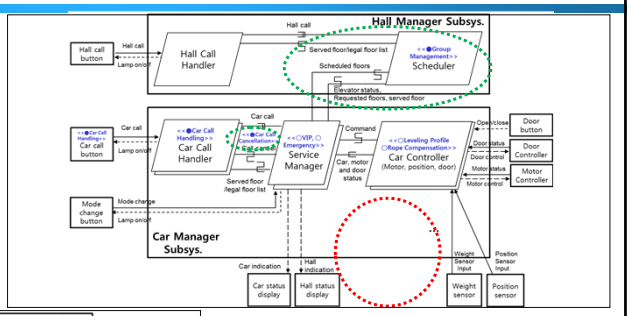
POSTECH

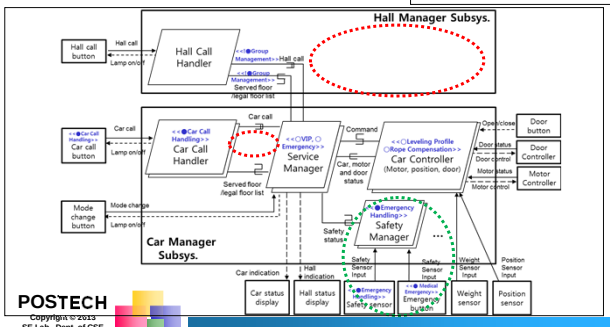
Process Architecture Selection

Product Development

With the **Group Management** and **Car Cancellation** features,

without the **Emergency Handling** and **Medical Emergency** features





With the **Emergency Handling** and **Medical Emergency** features,

without the **Group Management** and **Car Cancellation** features

POSTECH

Software Product Line Engineering

Agenda

- Maintainability and Reusability
- Product Line Engineering and Variability Management
- Feature-Oriented Product Line Engineering
- **Product Line Adoption Issues**

Process — Adoption Issues

- How to change to PL-based organization
  - How to evolve: staged process model for reuse adoption
  - Key process areas
    - Best practices
  - Metrics
    - Key indicators: cost of production, time to market, project completion time, etc.
    - Relationship between reuse, quality, and productivity
    - Relationship between reuse and ROI for sustainability of a reuse program
- Process models
  - Proactive vs. reactive vs. extractive models
    - Best practices
  - PL process vs. agile methods

Software Product Line Engineering

Management ***Adoption Issues***

- ROI analysis
  - Estimating ROI from a reuse program
  - Estimating benefits from strategic market position
- Asset management (How to make PL-based development happen in an organization)
  - Who should develop assets (with variation points)
  - Who should maintain assets (variation management)
  - Who will be responsible for quality assurance
  - Who should enforce the use of assets
  - Models (best practices)
    - Centralized vs. distributed

74

Variability Management ***Adoption Issues***

- Discovery and Modeling
  - Functions
  - Non-functional attributes
  - Usage and operating contexts
- Configuration
  - Decision model
  - Rationales
  - Goals and Issues
- Management
  - Centralized
  - Distributed

75

Software Product Line Engineering

Epilogue

- For good software engineering, maintainability and reusability must be built into software while designing!
- For softness , design must base on commonalities and variations of a product line
  - Think about what may change!
    - Look across similar applications
    - Look ahead for anticipated changes
    - Look at product contexts as well as functionality

Epilogue

Thank you!