pure·systems

プロダクトライン開発の実践事例

MBSEからHW設計まで

富士設備工業(株)電子機器事業部 浅野 義雄

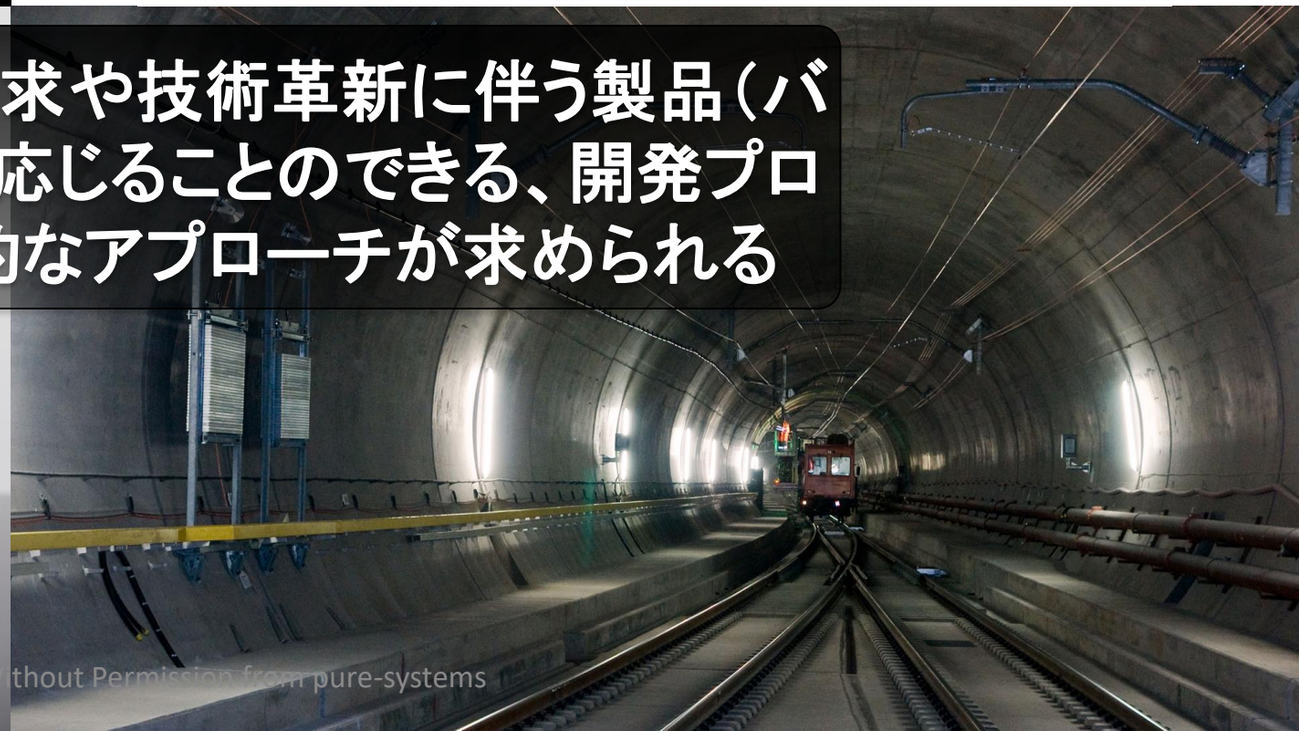
第5回 オートモーティブ・ソフトウェア・フロンティア 2020
2/7(金)13:05～13:35　A会場

内容

- **プロダクトライン開発のバリアント管理について**
- **バリアント管理ツールの機能**
- **プロダクトライン開発の実践事例**

Product Line Engineering（PLE）は、
再利用資産を運用する技術的な取り組み

継続的に変化する市場要求や技術革新に伴う製品（バリアント）の進化に柔軟に応じることのできる、開発プロセスや手法を伴う全体的なアプローチが求められる



Product Line Engineering Variation Dimensions

製品間の違い
Technical Dimension

プロダクトライン開発の課題として同時に対応すべき2つのバリエーション（変化）がある

この例では、一番上のみ右ハンドルで、下の3つは同様に見えるが違いもある

ここでは製品間の違いのみ

Product Line Engineering Variation Dimensions

製品間の違い
Technical Dimension

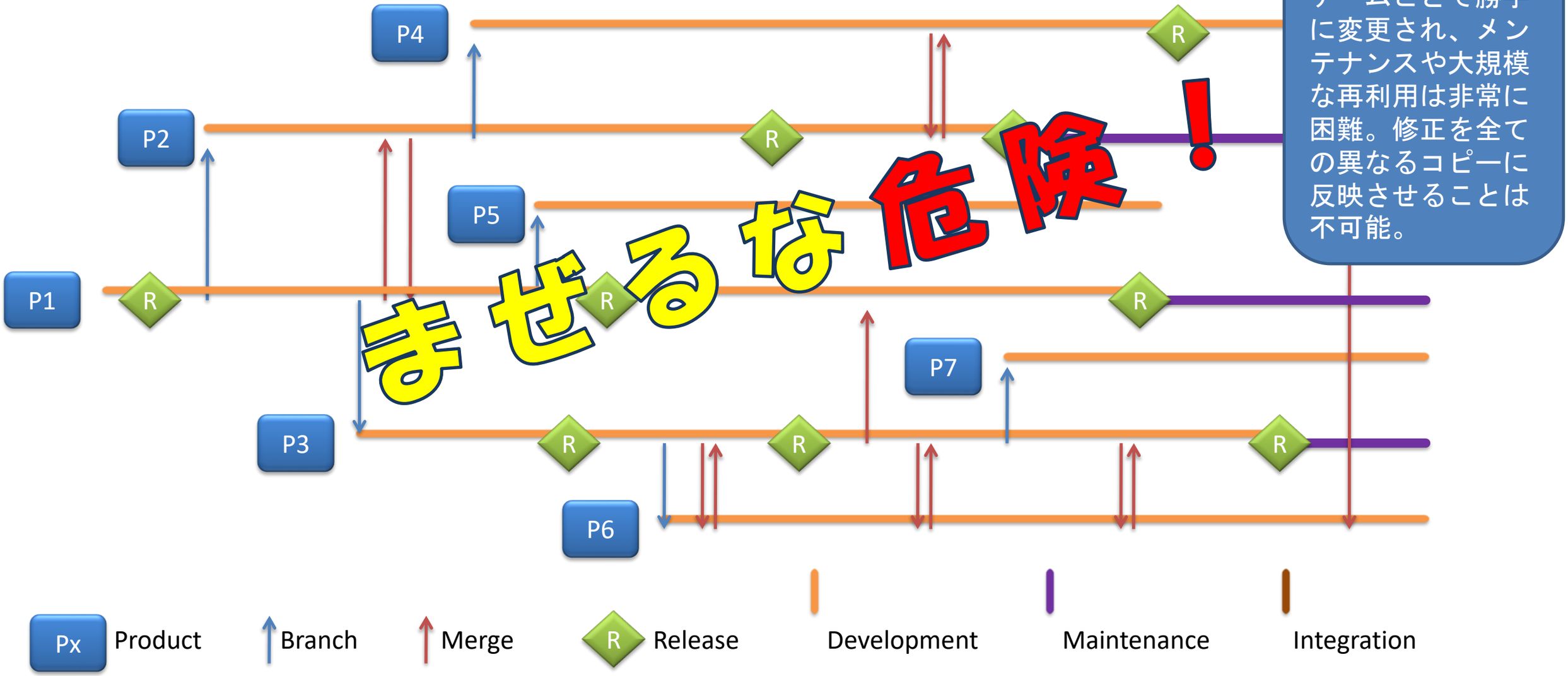
これら製品（バリアント）は、継続的に変化する市場要求や技術革新を受けて進化する

備えはいるが、どうなるか予測はできない

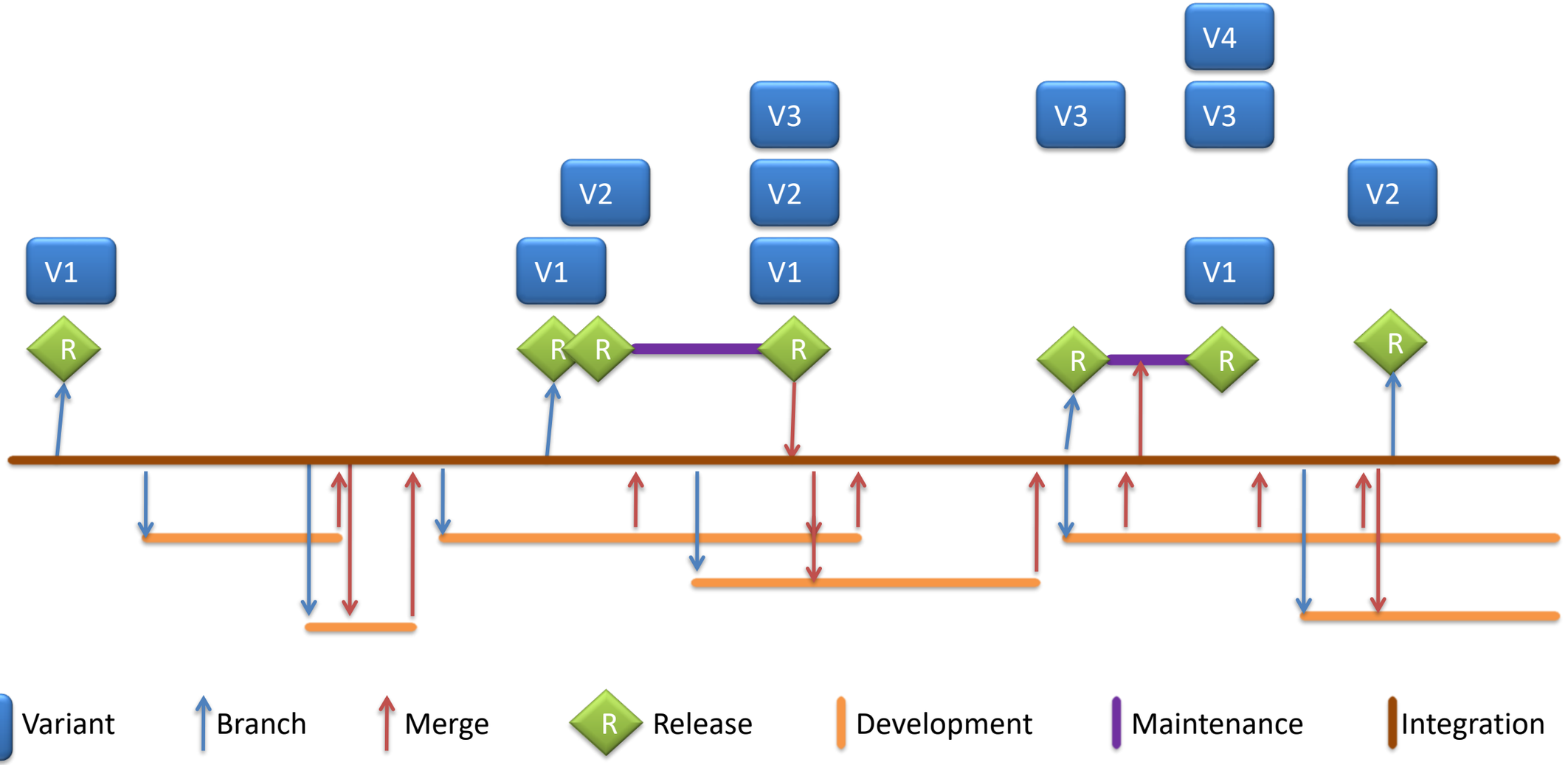
同時進行で発生するバリエーション（製品間の違い）とバージョンの両方の管理が必要

バージョン
Time Dimension

バージョン管理とバリエーション管理を混同すると、、



https://www.fuji-setsu.co.jp/products/purevariants/Danilo_Blog.html#5

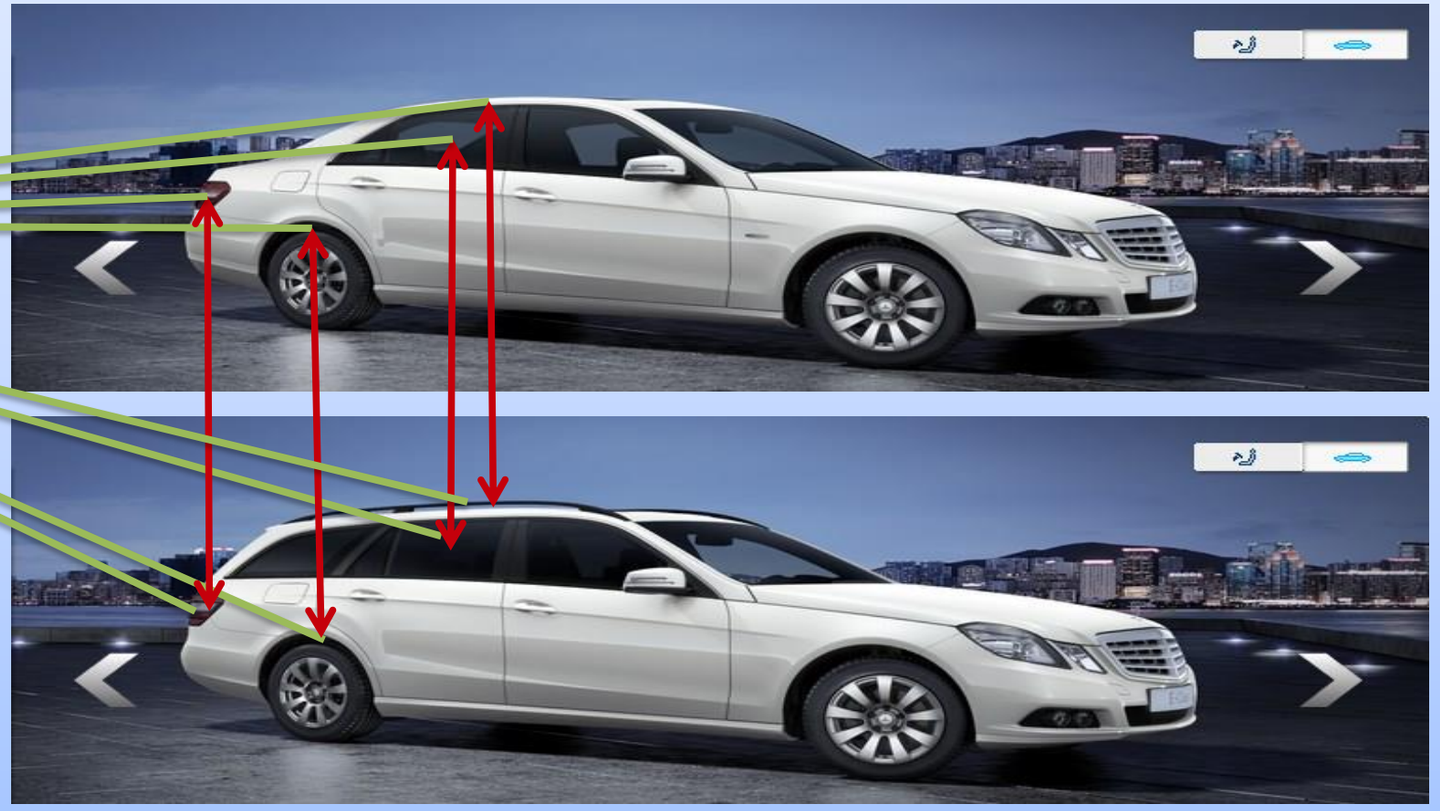


バージョン管理に適正なバリアント管理ツールを統合することで
プロダクトラインの持続的な進化と保守を支援して体系的な再利用を実現できる

バリエーションポイント

問題空間

解決空間

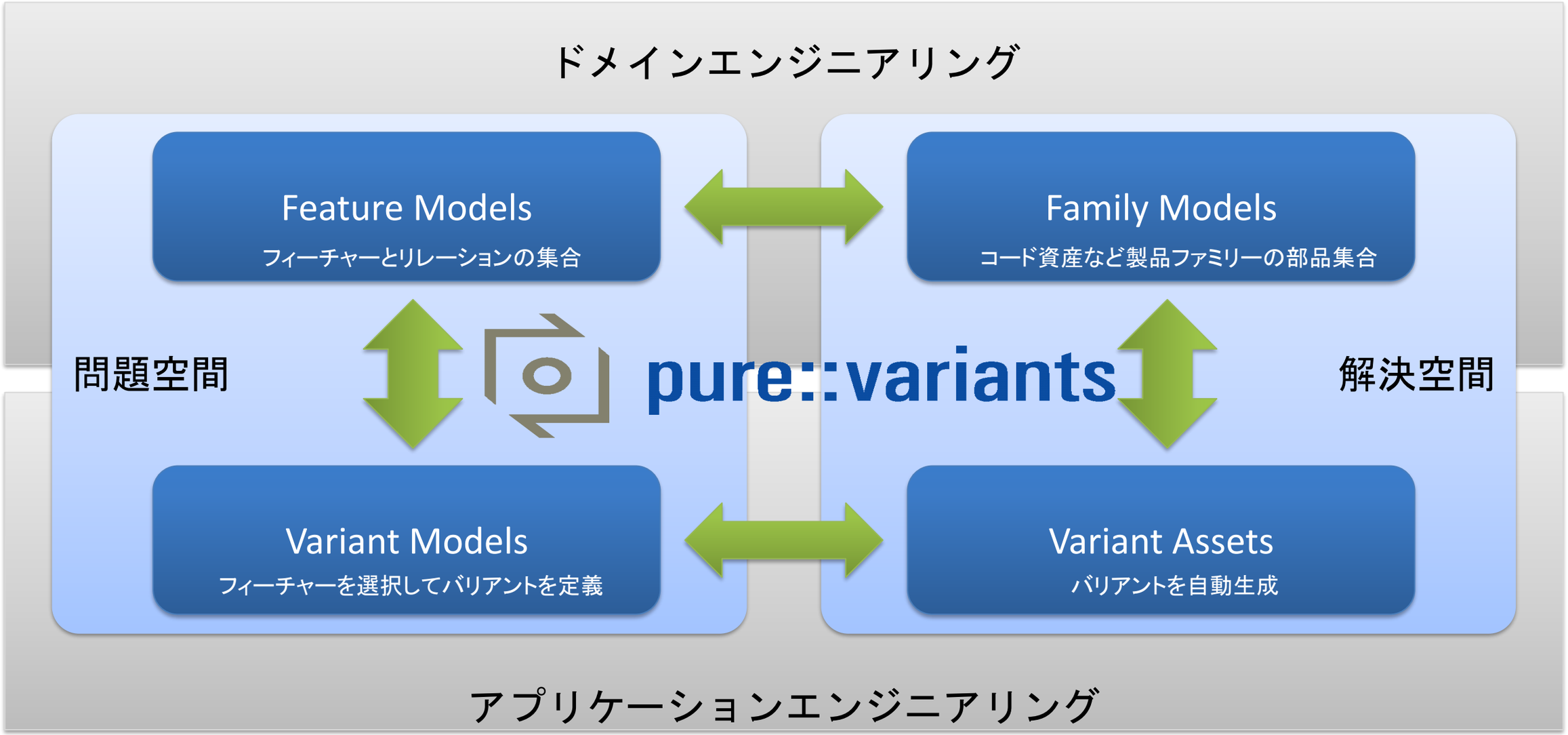


問題空間上のバリエーションポイントは、解決空間上のテクニカルなそれと結びつき、バリエーションの複雑さを軽減してバリアントの決定項目を削減できる

Variant Management Solution for Systems & Software Engineering

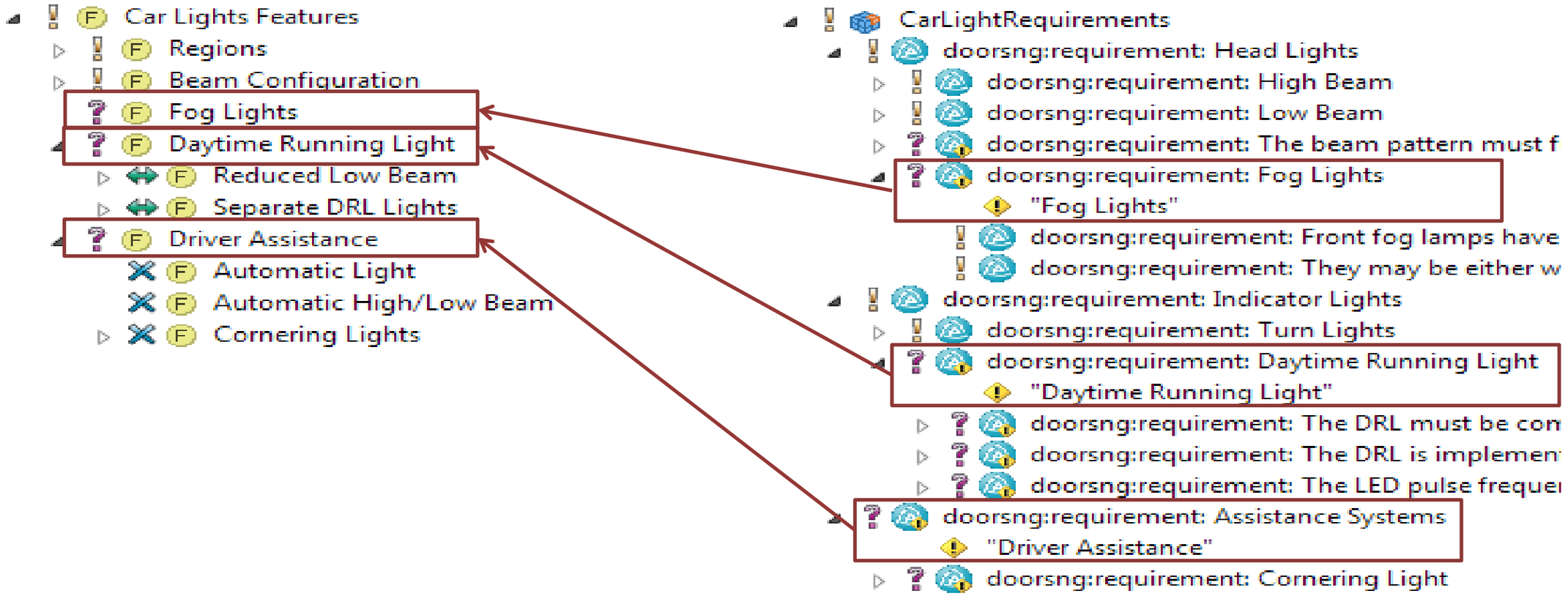
pure::variants

PLEの戦略＝関心事の分離



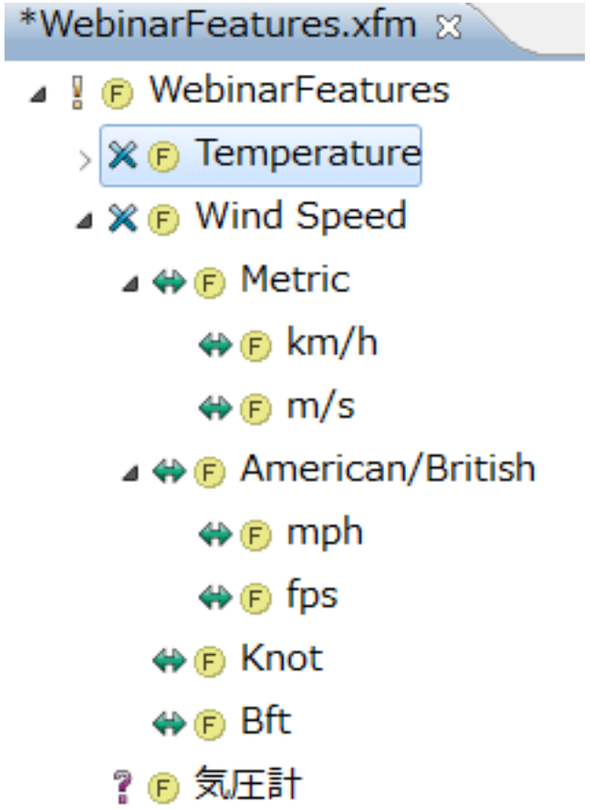
Feature Models

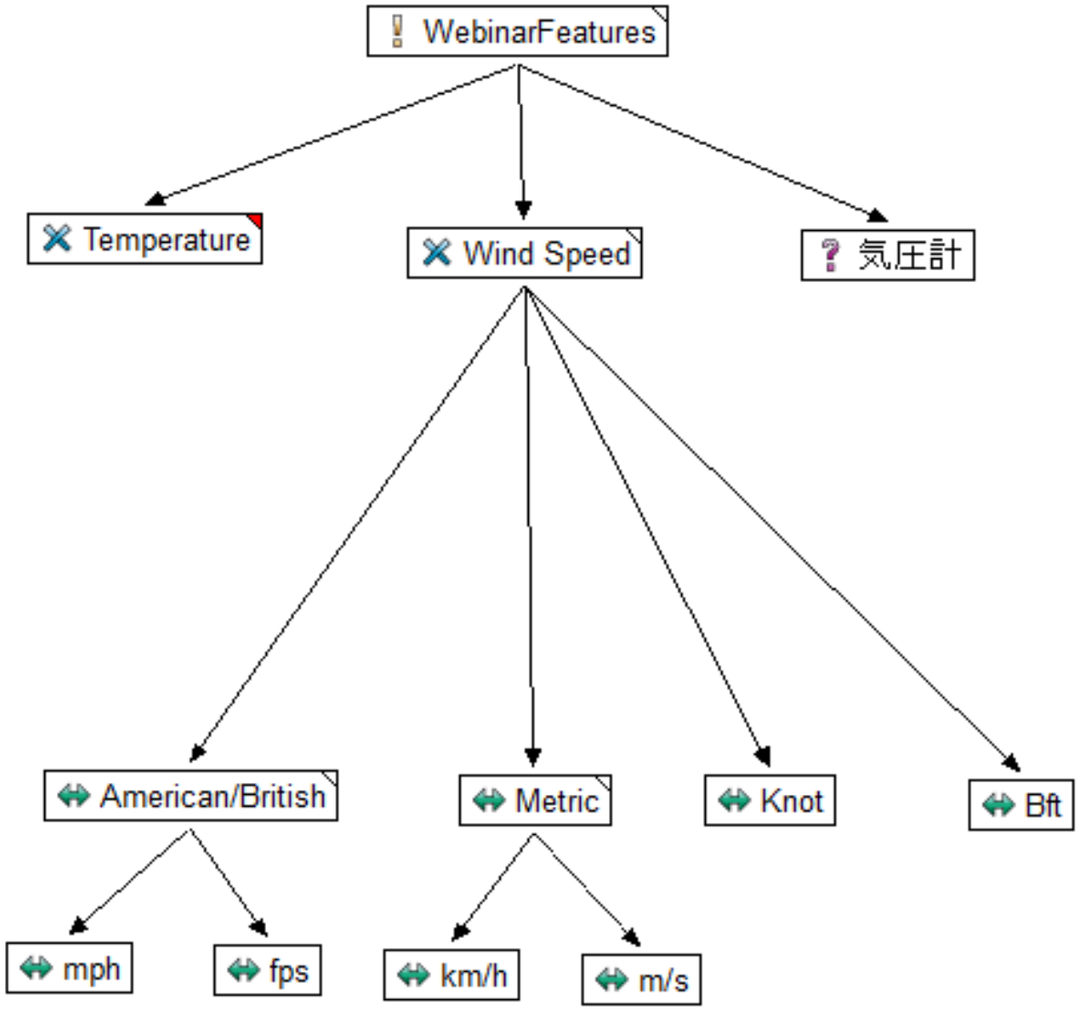
Family Models



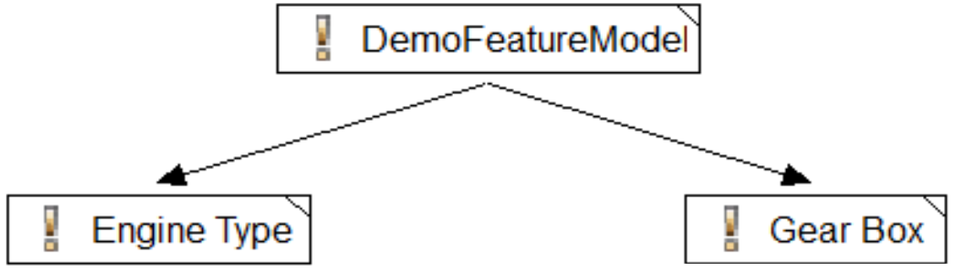
問題空間上の各フィーチャに解決空間内のバリエーションポイントを紐付ける

フィーチャモデル：ツリー構造とグラフ表示



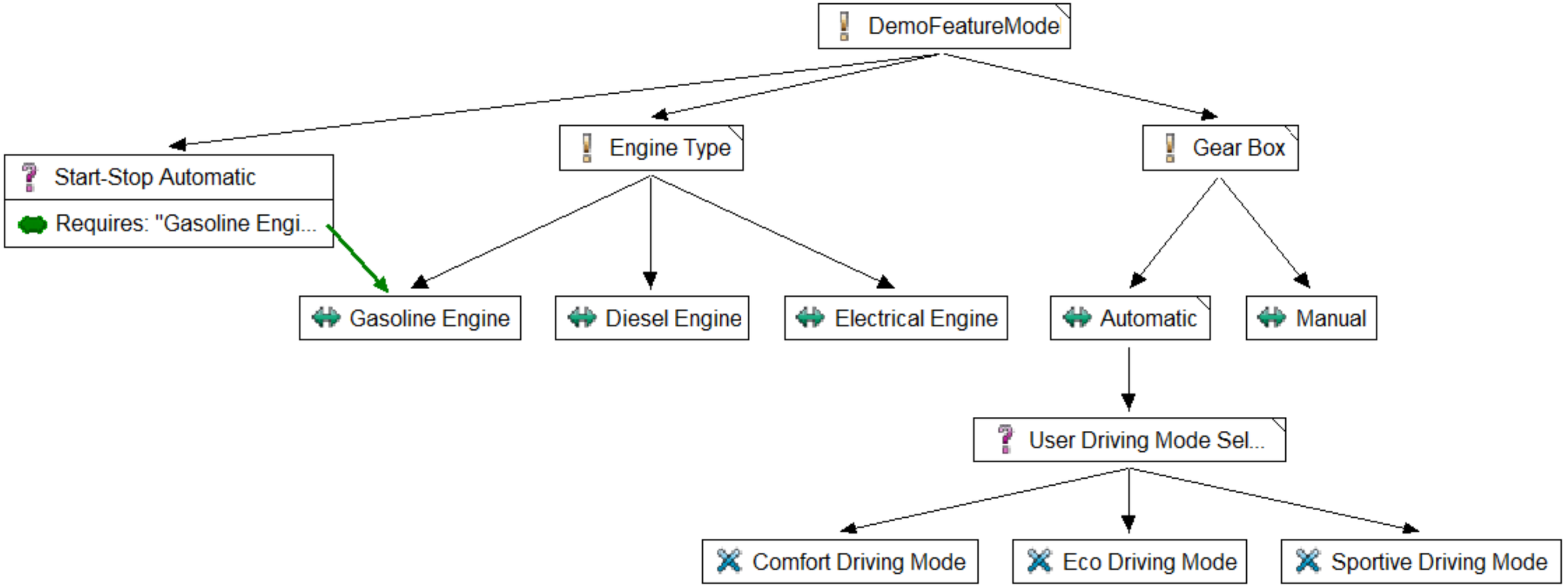


Feature Model



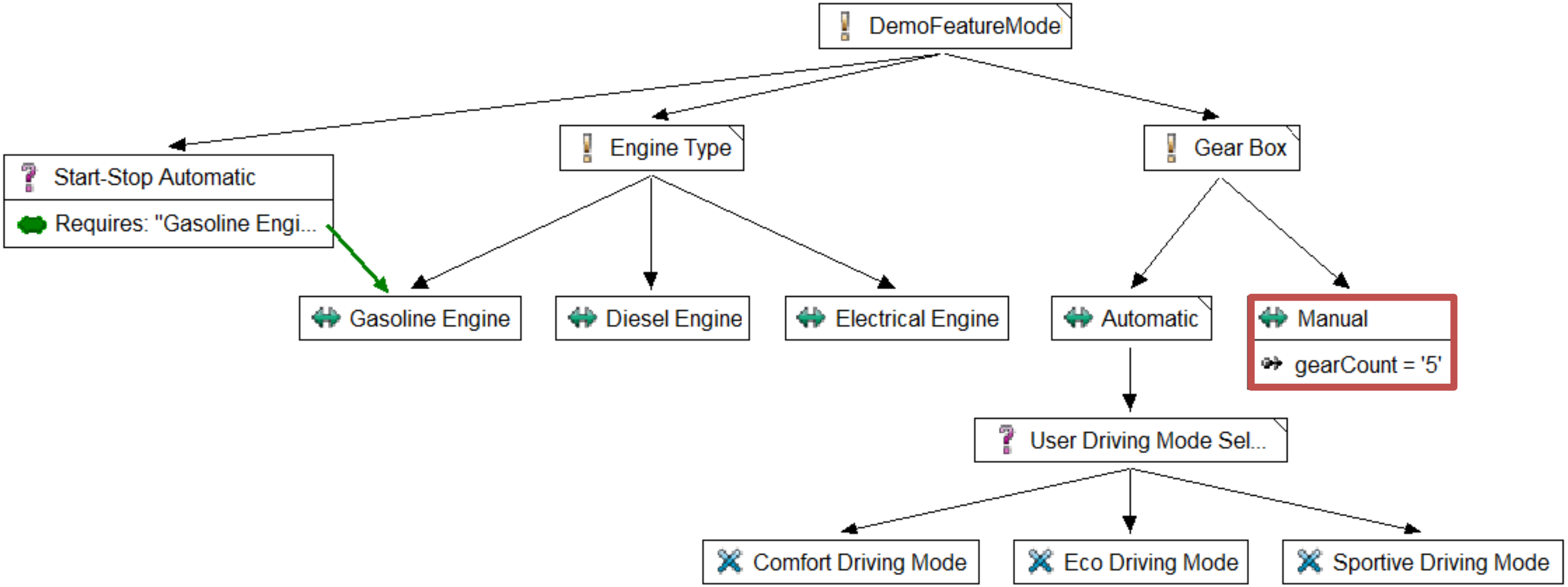
Legend: = Mandatory（必須） = Optional（選択自由） = Alternative（どれか一つ） = Or（少なくとも一つ）

Feature Model - Dependencies



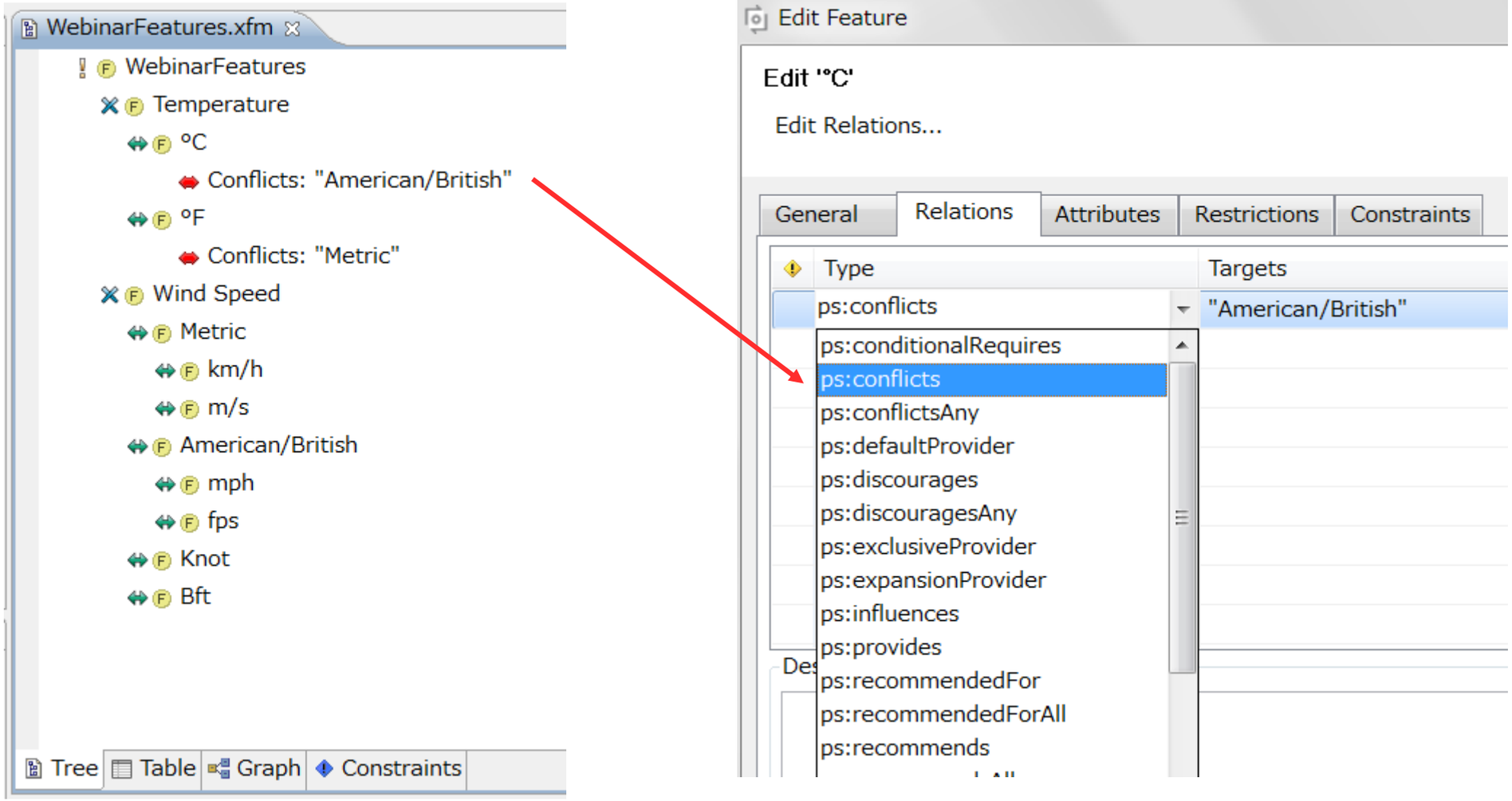
Legend: = Mandatory（必須） = Optional（選択自由） = Alternative（どれか一つ） = Or（少なくとも一つ）

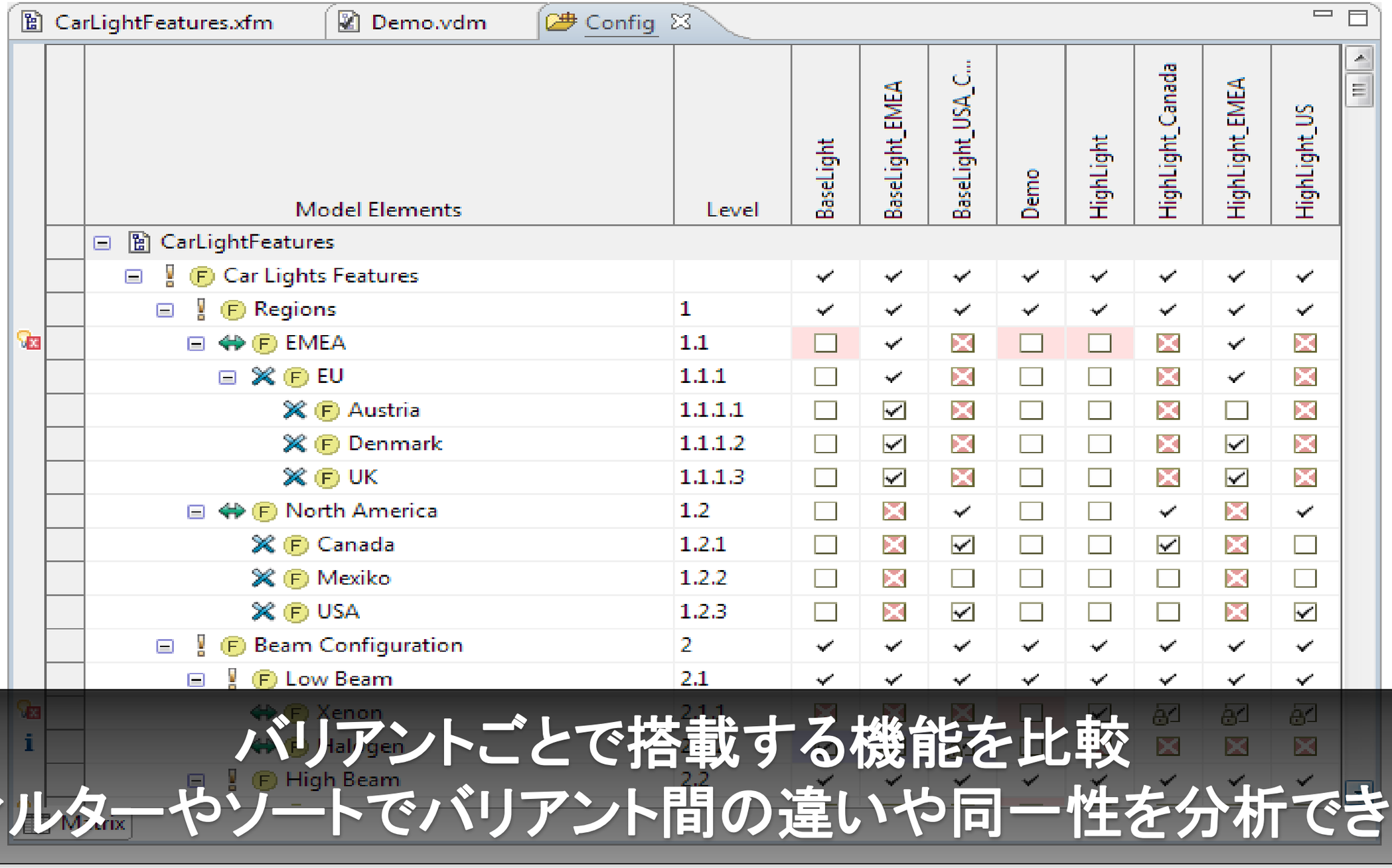
Feature Model - attribute



Legend: = Mandatory（必須） = Optional（選択自由） = Alternative（どれか一つ） = Or（少なくとも一つ）

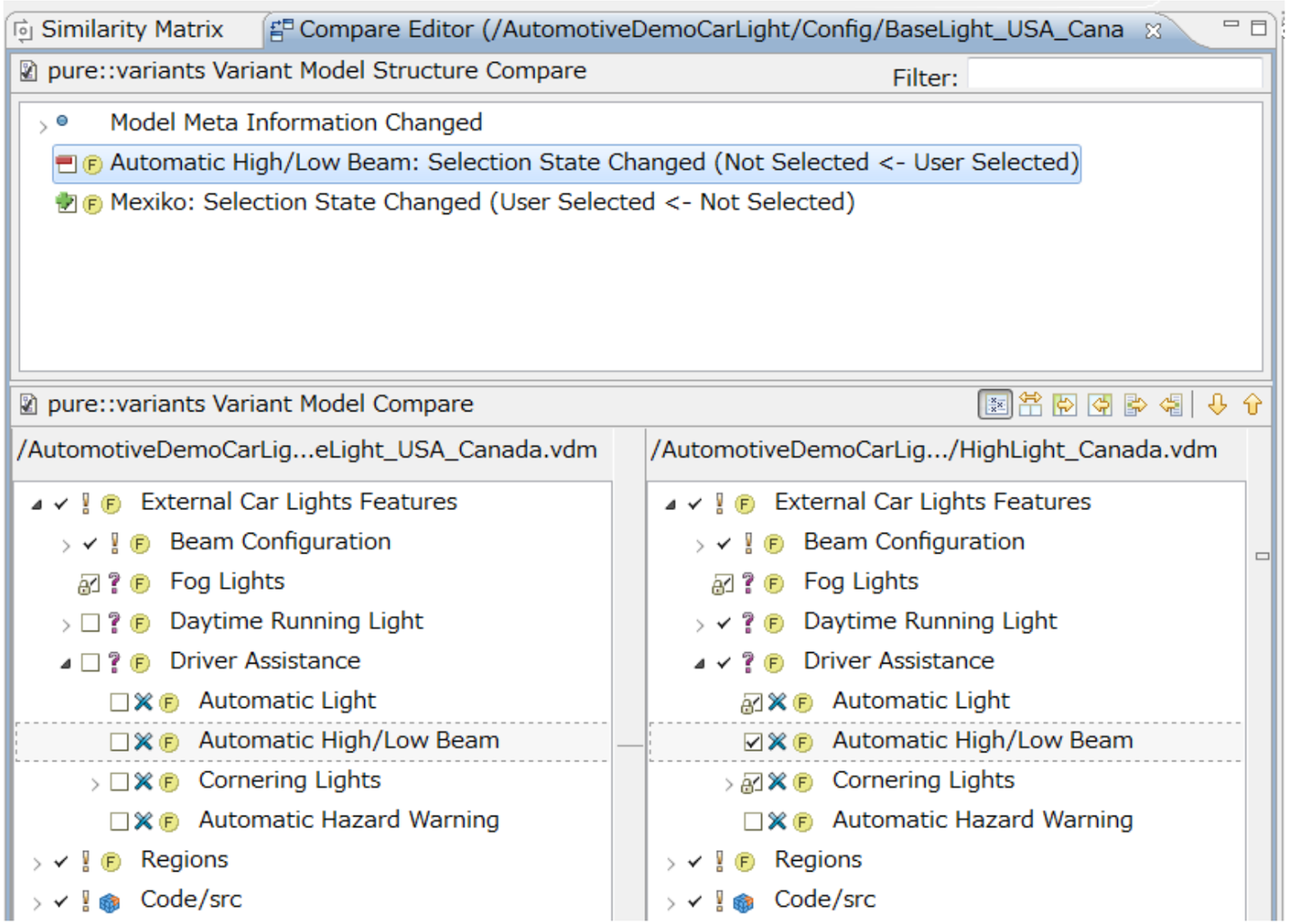
フィーチャ間の依存・排他関係





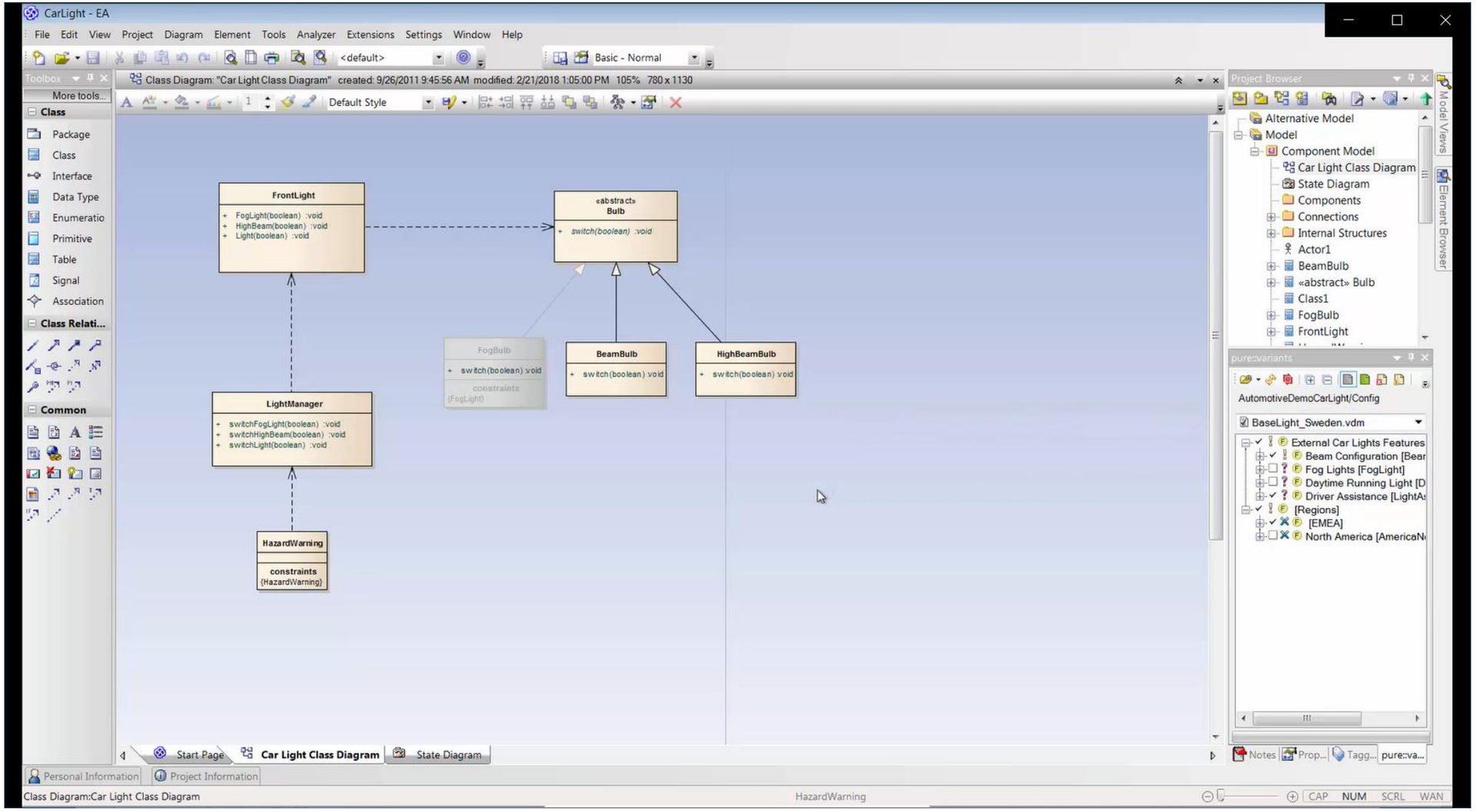
バリアントごとで搭載する機能を比較
フィルターやソートでバリアント間の違いや同一性を分析できる

バリアント間の比較や旧バージョンとの比較もできる



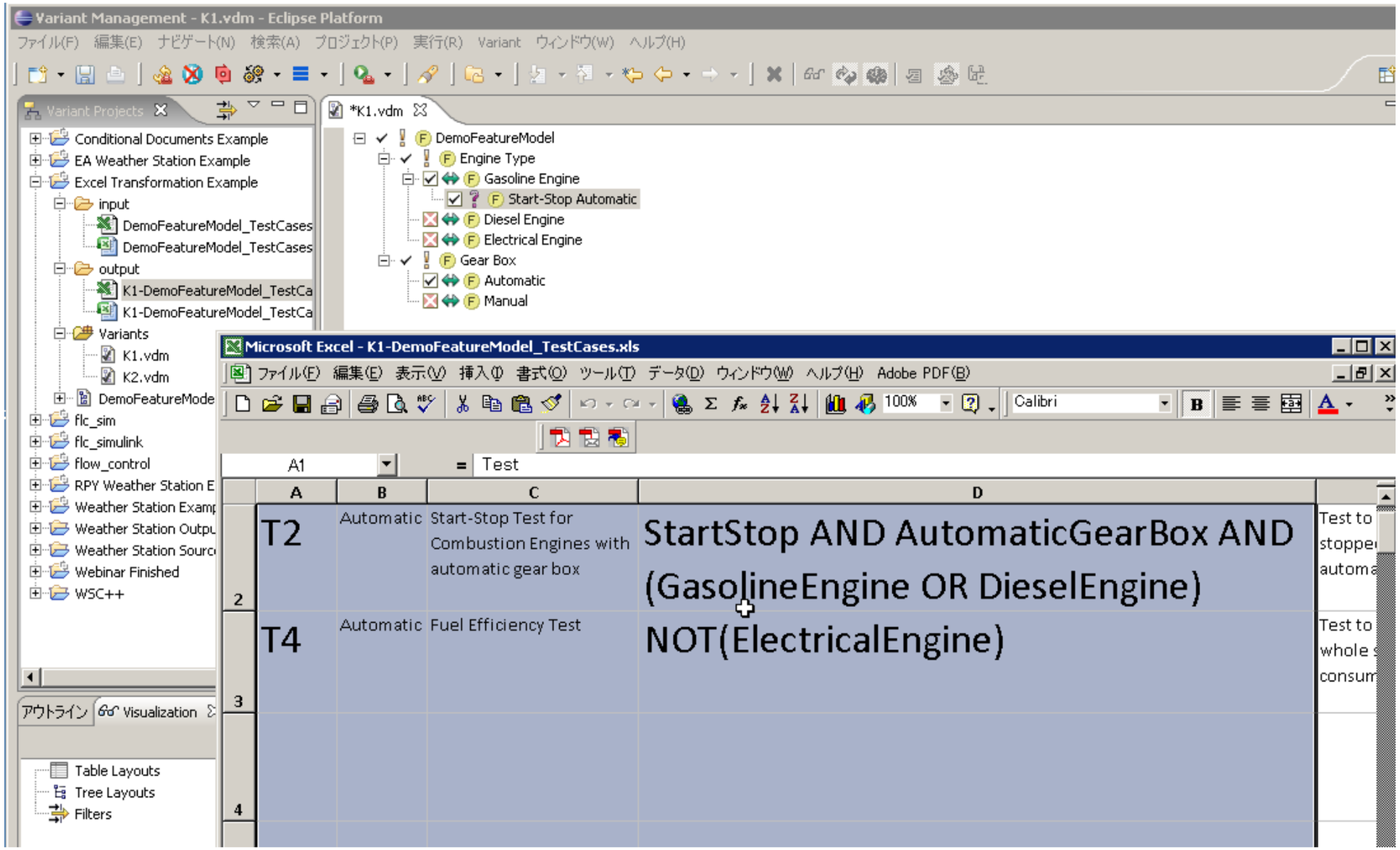
http://www.fuji-setsu.co.jp/demo/pvIBM/pv3CompareMatrix.wmv

UML連携：Enterprise Architect　動画デモ



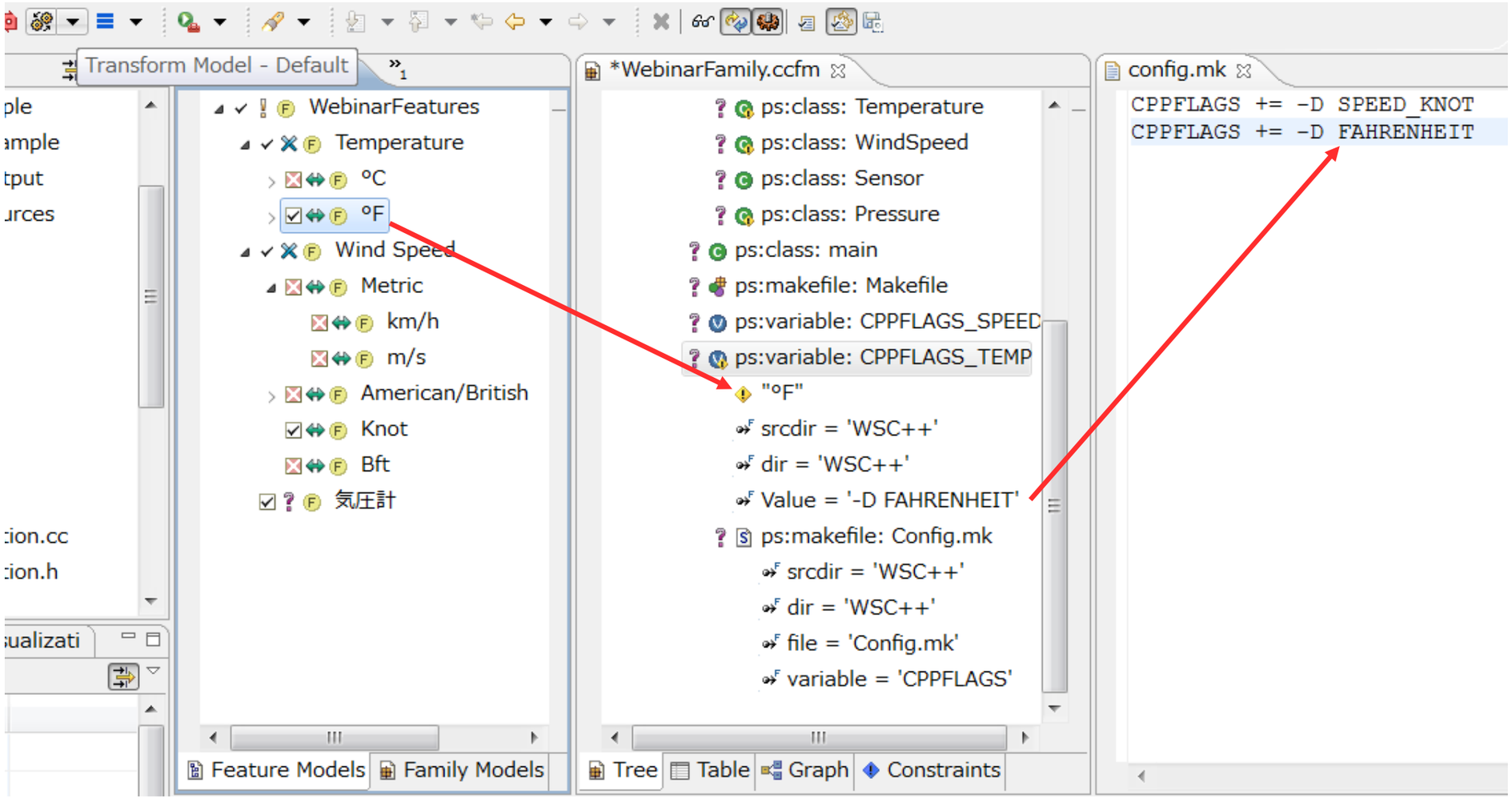
https://www.fuji-setsu.co.jp/products/purevariants/tutorials.html#EA

Excel連携：仕様・ケースの管理など

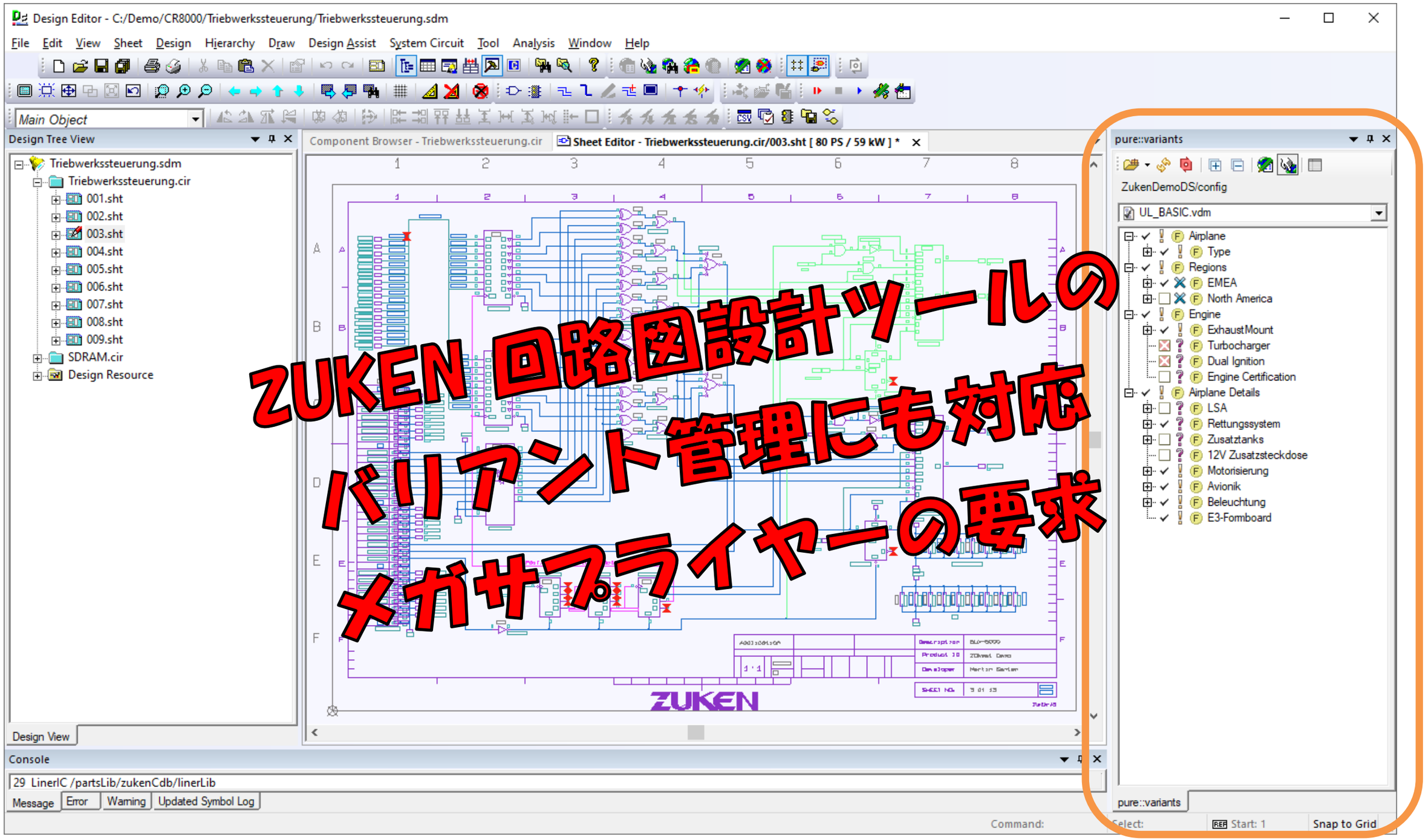


https://www.fuji-setsu.co.jp/products/purevariants/tutorials.html#office

フィーチャ選択によりmakeファイルにフラグを設定

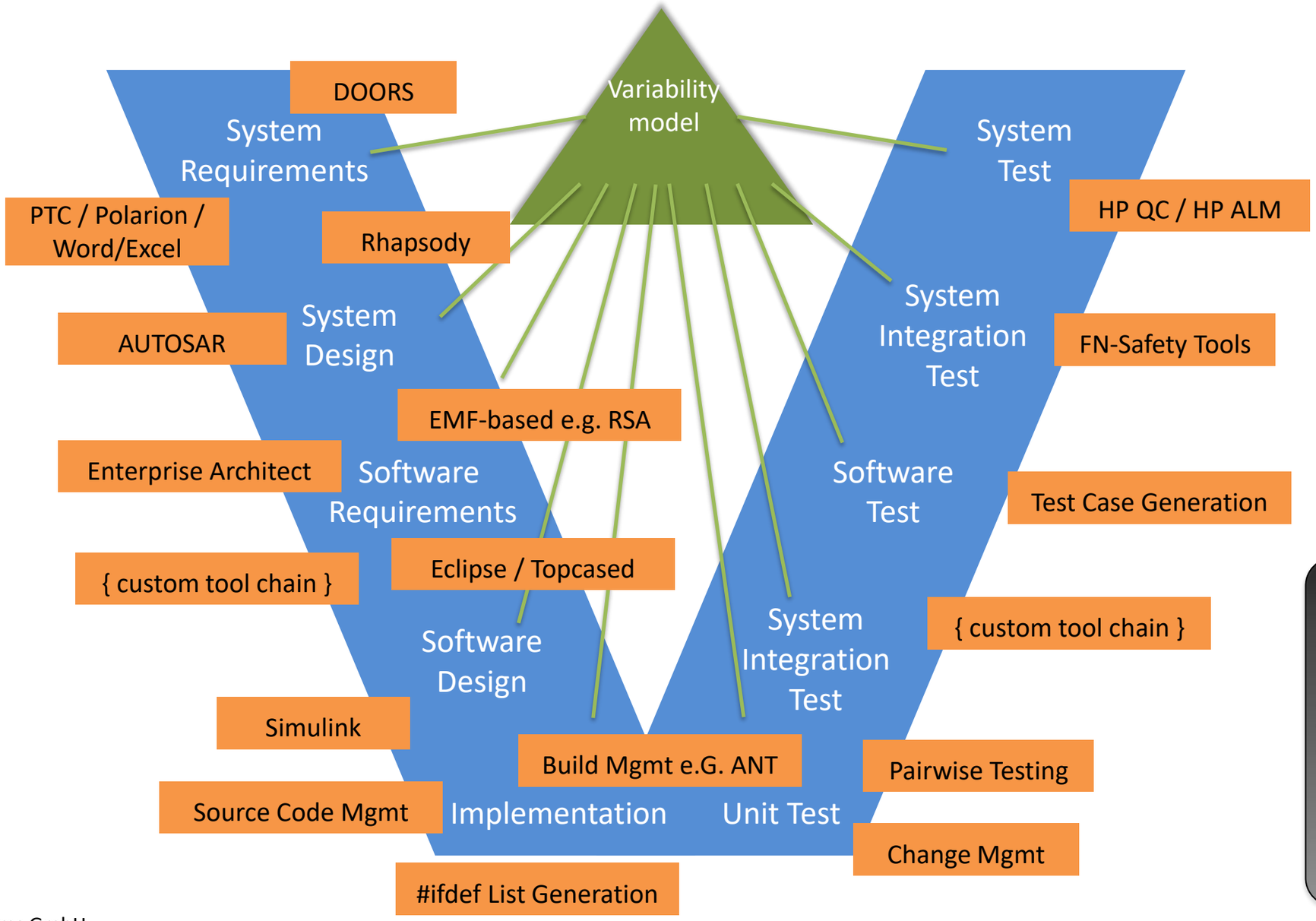


https://www.fuji-setsu.co.jp/products/purevariants/tutorials.html#SourceCode



ZUKEN 回路図設計ツールの バリアント管理にも対応 メガサプライヤーの要求

pure::variants Enterprise Integrations



フィーチャモデルにプロダクトラインの問題空間のバリエーションを表現して、解決空間上のあらゆる資産と紐付けられる

pure::variants プロダクトラインライフサイクルをサポート

DOORS 9 | DOORS Next | Polarion Requirements

Rhapsody | RTC | Rational Quality Manager | C/C++/Java

MS Word,Excel | AUTOSAR | EMF | Enterprise Architect

Simulink | MagicDraw | … | Zuken | medini analyze

Made with pure::variants

プロダクトライン開発とバリアント管理 / 成功事例

Prof. Dr. Danilo Beuche

Customer Example

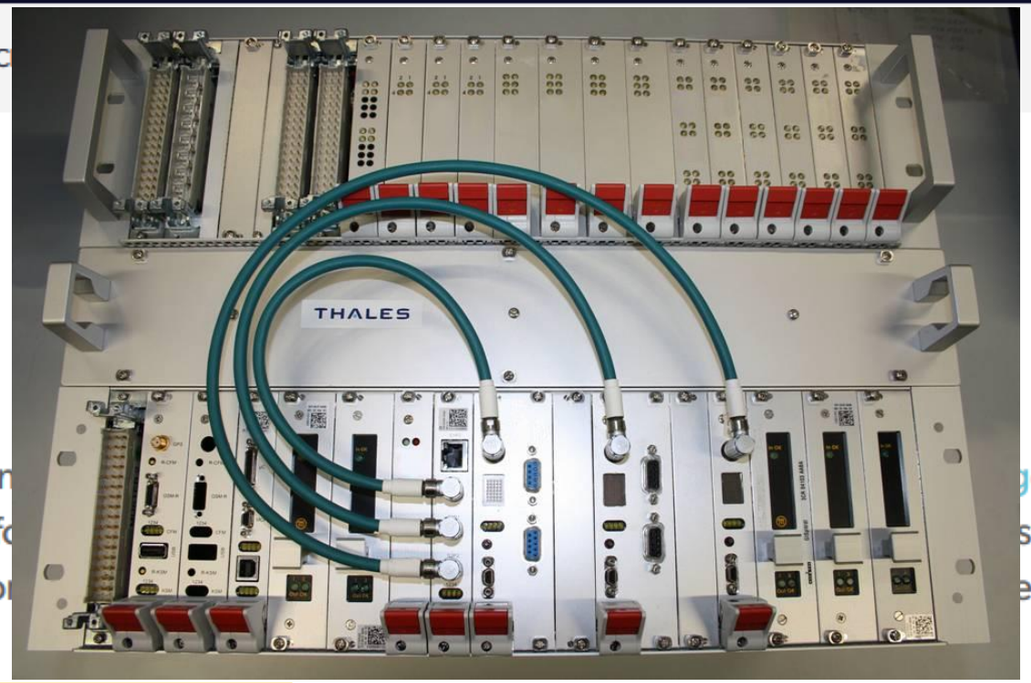
THALES

Home

European Train Control System (ETCS)

欧州鉄道の統一列車制御システム

TECHNOLOGY FUTURE SOLUTIONS REFERENC



背景：ヨーロッパの鉄道は国ごとの共通性がなく、
一つの国に複数の信号システムが存在することもあり、相互運用性の向上のため、統一された信号システムが必須

ETCS is the core signalling and train gement System. ETCS continuously calculates a safe maximum speed fo stems that take control if the permissible speed is exceeded. For e standardized according to the different ETCS levels.

Thales offers a complete portfolio of ETCS solutions or turnkey projects in the same way as for green field or brown field

THALES 課題：鉄道システムのバリアビリティ

banedanmark

ATC
デンマーク

DB NETZE
LZB
ドイツ

SBB
Integra-Signum
スイス

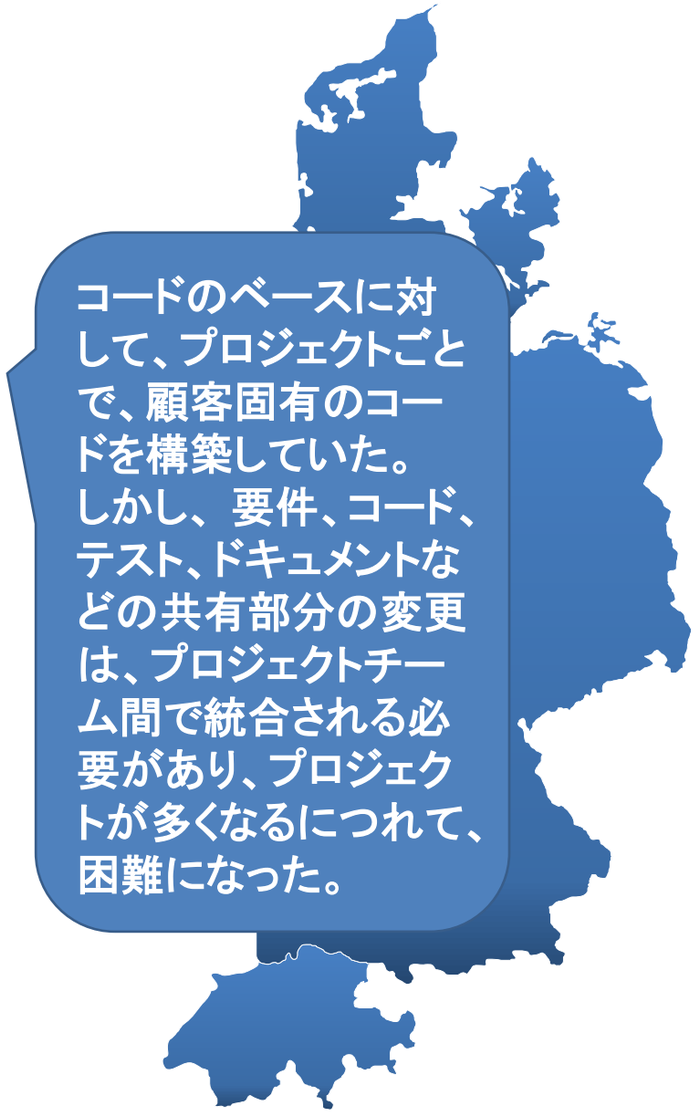
ETCS L2
欧州統一
列車制御システム
Level 2

ETCS L2 + ATC

ETCS L2 + LZB

ETCS L2 + Integra-Signum

THALES 課題：鉄道システムのバリアビリティ



コードのベースに対して、プロジェクトごとで、顧客固有のコードを構築していた。
しかし、要件、コード、テスト、ドキュメントなどの共有部分の変更は、プロジェクトチーム間で統合される必要があり、プロジェクトが多くなるにつれて、困難になった。

従来の取り組みは、

banedanmark

ETCS L2

ATC

ETCS L2 + ATC

DB NETZE

LZB

ETCS L2 + LZB

SBB

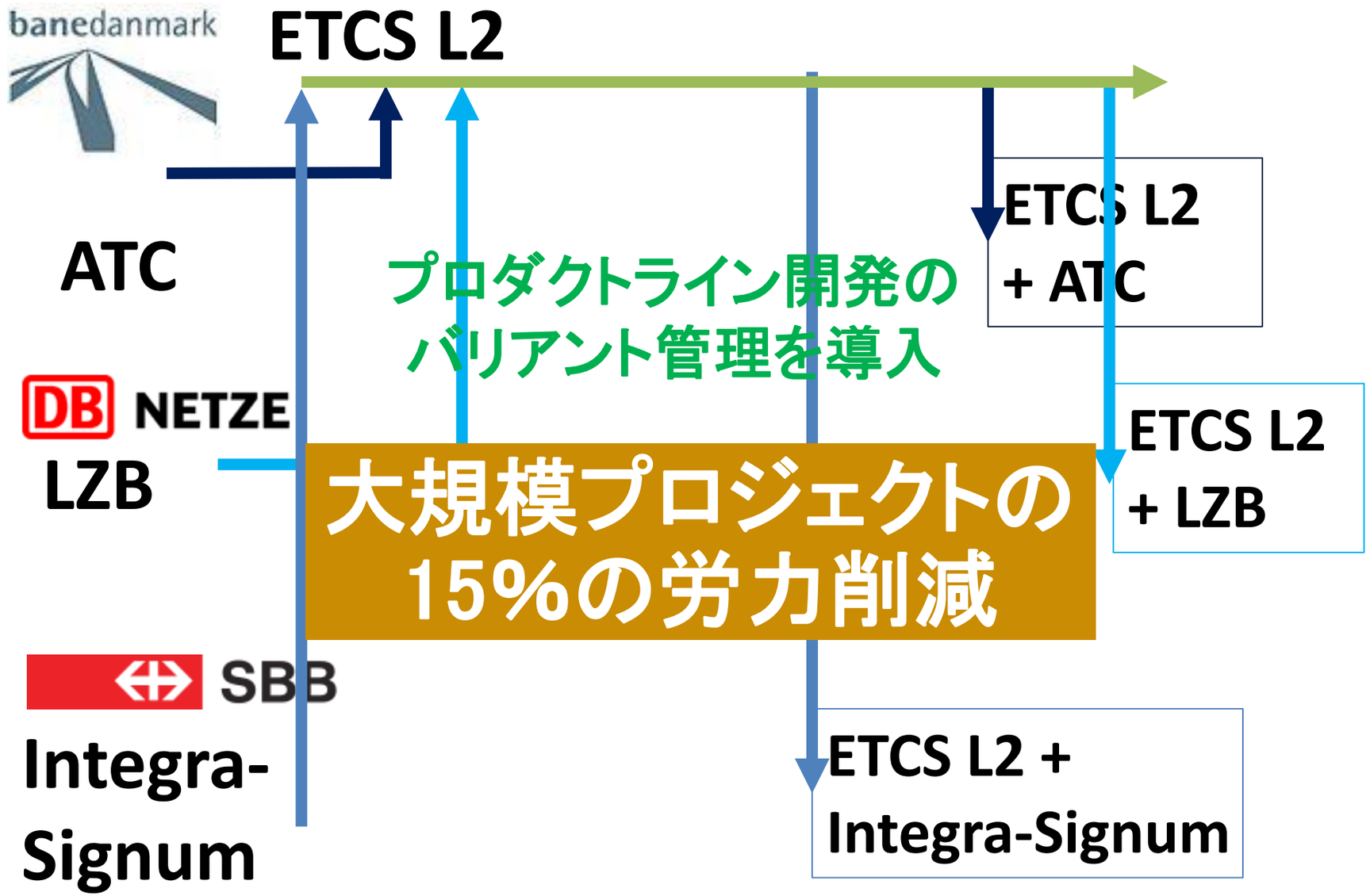
Integra-Signum

ETCS L2 + Integra-Signum

THALES 課題：鉄道システムのバリアビリティ

製品ファミリの資産から、顧客プロジェクトに応じたバリアントを自動生成して、契約ごとではなく、プロダクトラインをベースに整合性のあるプロセスで開発できるようになった。

2012年、2つのプロジェクトのDoorsで管理された要件から始めて、直ぐにコード、テスト、およびドキュメントなど他の資産にも範囲を拡大。システムの性質上、当初は少数のプロジェクトでも多くのフィーチャ(x600～700)でしたが、既存の機能がドメインの変動性を非常にうまく捉えていたため、以降のプロジェクト数の増加に対して、フィーチャ数は比例していない。過去5年で10件のプロジェクトでフィーチャ数＝1000でした。

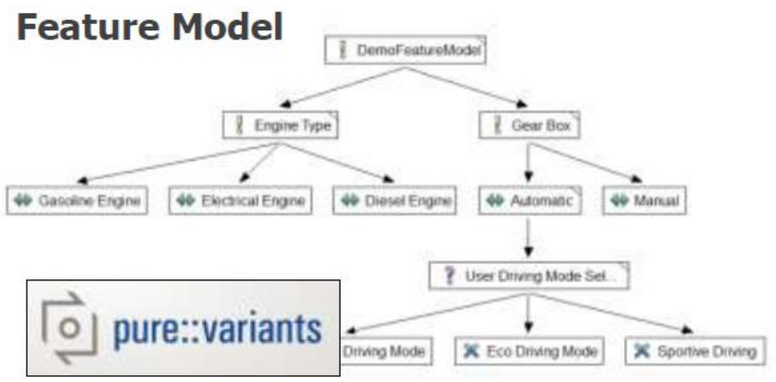


Real world examples

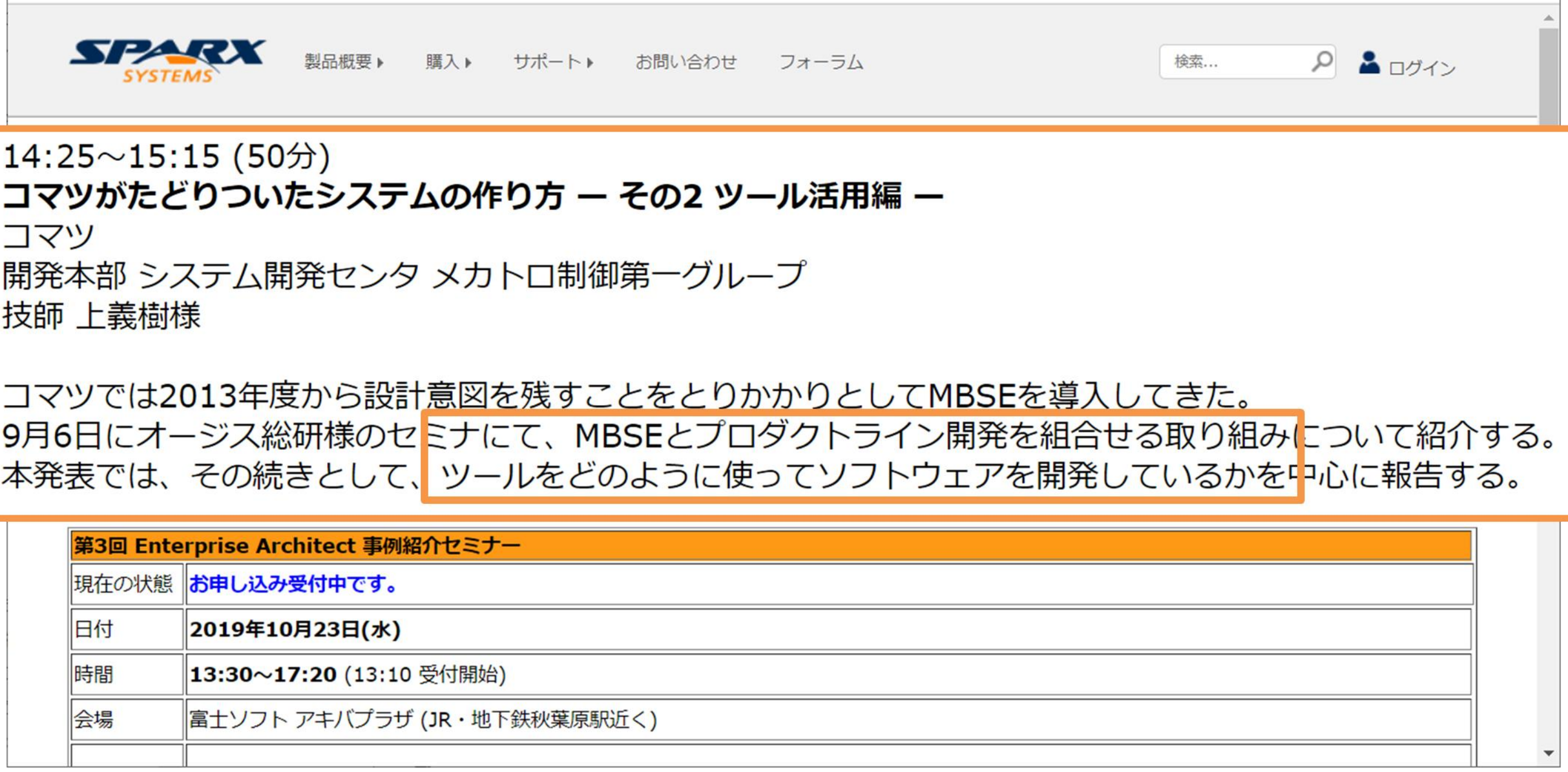
THALES

THALES は全社のPLE に pure::variants を採用

参考データ:
22000人のR&Dエンジニア



Source: MBSE Symposium,
“**MBSE, Backbone of the Thales Engineering Manifesto**”,
Olivier Flous
VP Engineering THALES,
October 27th, 2014 Canberra, Australia
http://download.polarsys.org/capella/publis/MBSE_Canberra-Backbone_of_the_Thales_Engineering_Manifesto.pdf



「gihyo.jp」に記事公開 => http://gihyo.jp/news/report/2019/11/1112

MBSE，プロダクトライン開発を実践し要求を正しく満たすソフトウェアをスピード感をもって実装できる体制を整備

Customer Example

The Product Line Family








ダンフォス社（デンマークの大手企業）

Frequency converters / Drives　周波数コンバータ

業界2位

- Determine motor speed and torque by controlling the frequency of the power supply

電源の周波数を制御して
モーターの速度とトルクを決定する

Complete product range up to 690 V

- From 200 V up to 690 V
- From 0.18 kW to 1.2 MW

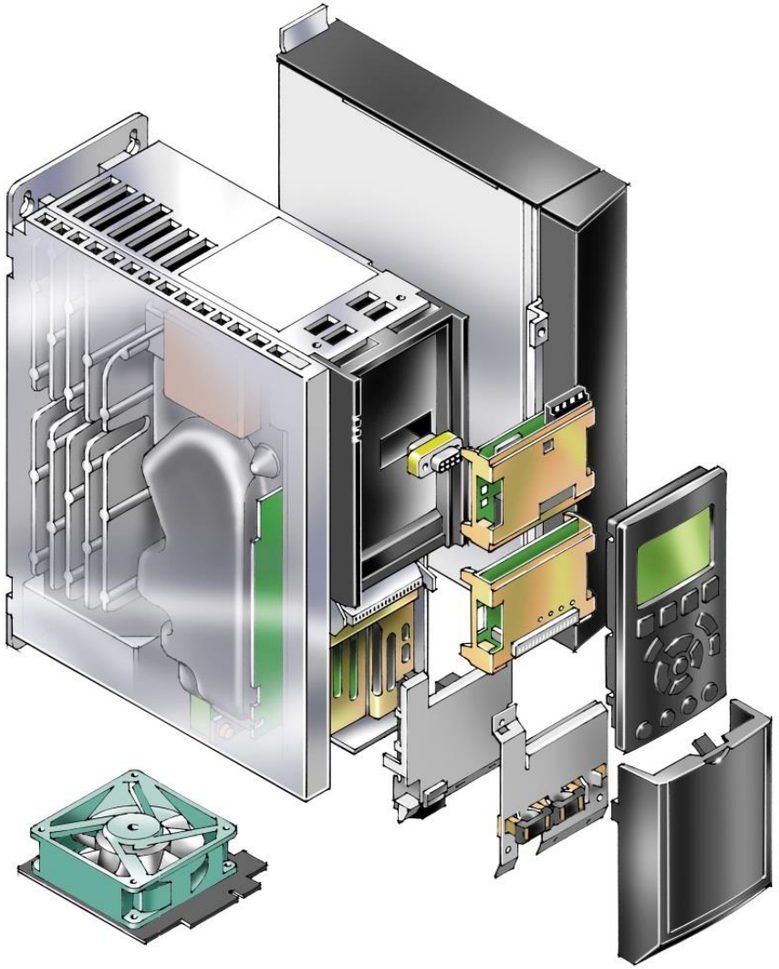
異なる市場向けの製品群
小型～大規模システム向けまで
様々なバリアントが存在する

Drives for different application areas

- VLT® AutomationDrive
- VLT® AQUA Drive
- VLT® HVAC Drive
- Customized drives

工業用、水のポンプ、ビルの空調、特定用途向けの周波数コンバータ

なぜプロダクトライン開発？



主な動機：

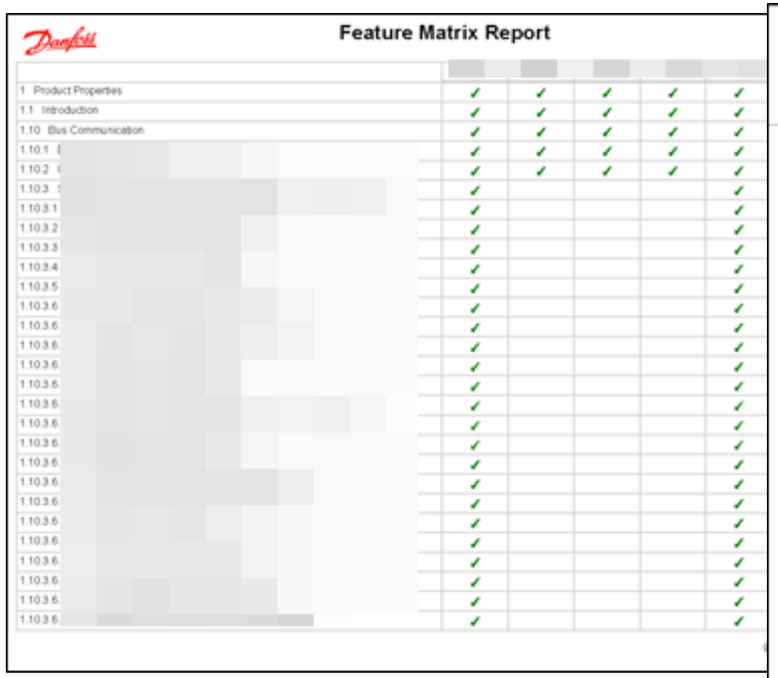
- 類似製品の増加
- ソフトウェア資産とその要件のより良い管理を模索
- 資産を再利用して品質を向上させる

ゴール：

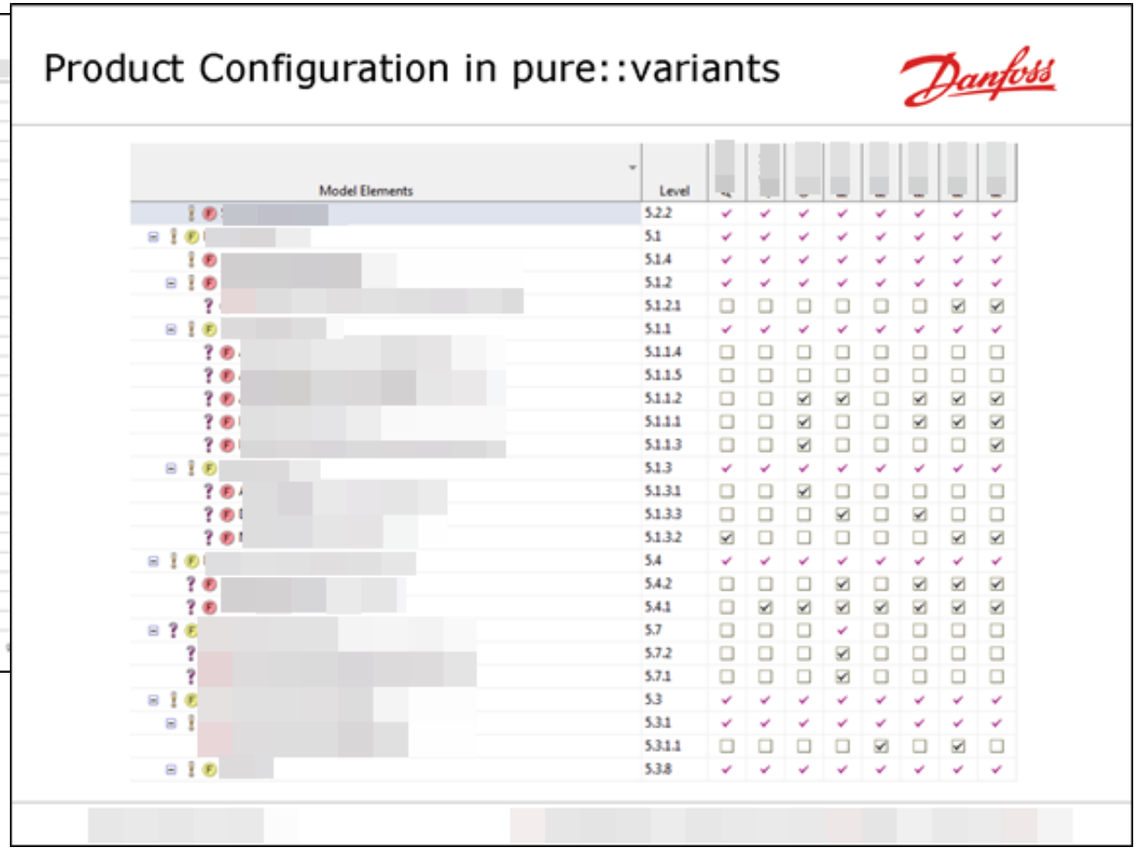
- CRS（顧客要求仕様書）およびPRS（製品要求仕様書）レベルで製品を構成できること
- 3つのレベル間のモデルをバインドさせることにより、設計からソースコード構成までのCRSからPRSへのトレーサビリティを確保

The Product Line Family

- 2000要件に対してソフトウエアに800のバリエーションポイントを持つ

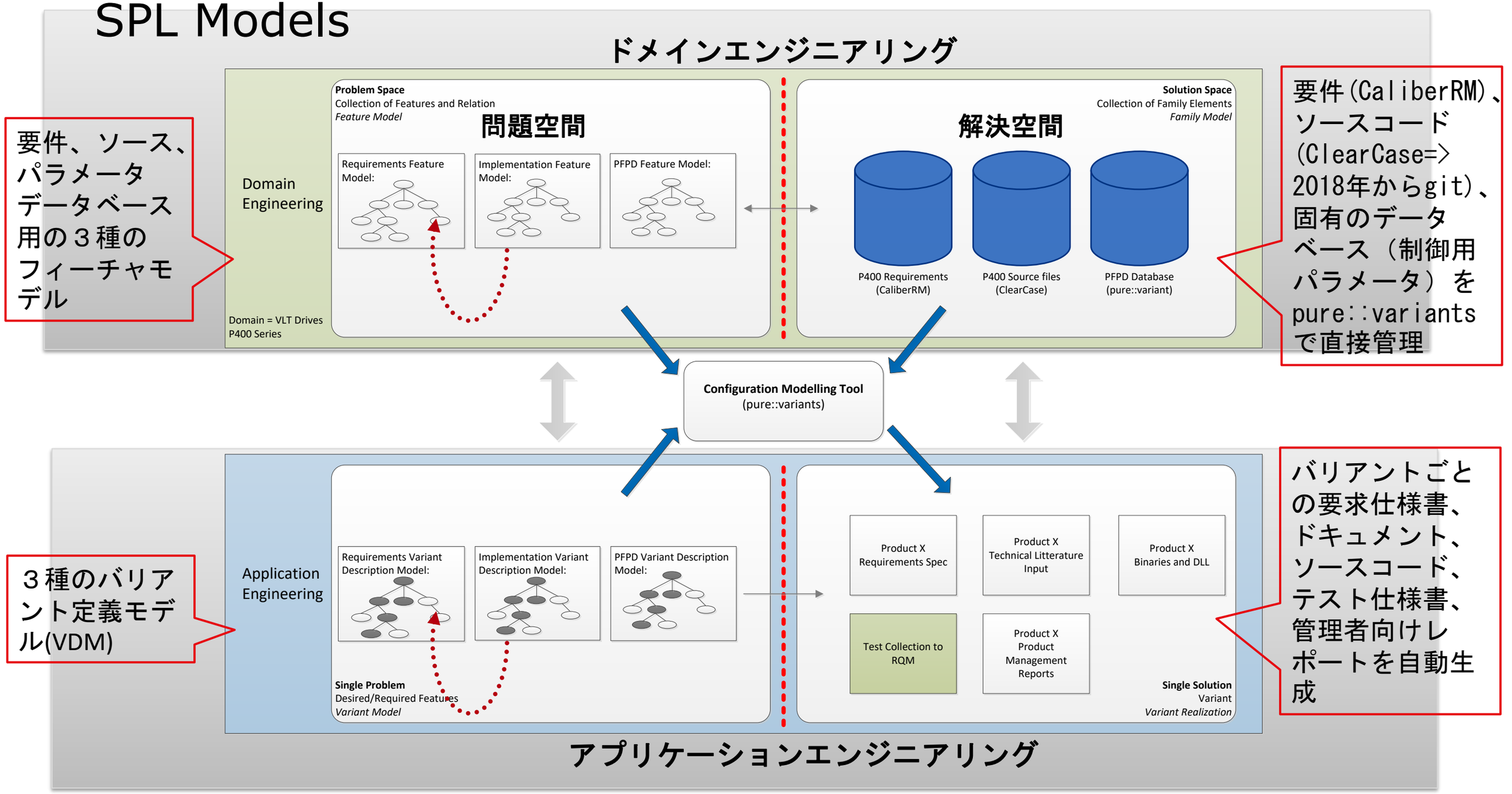


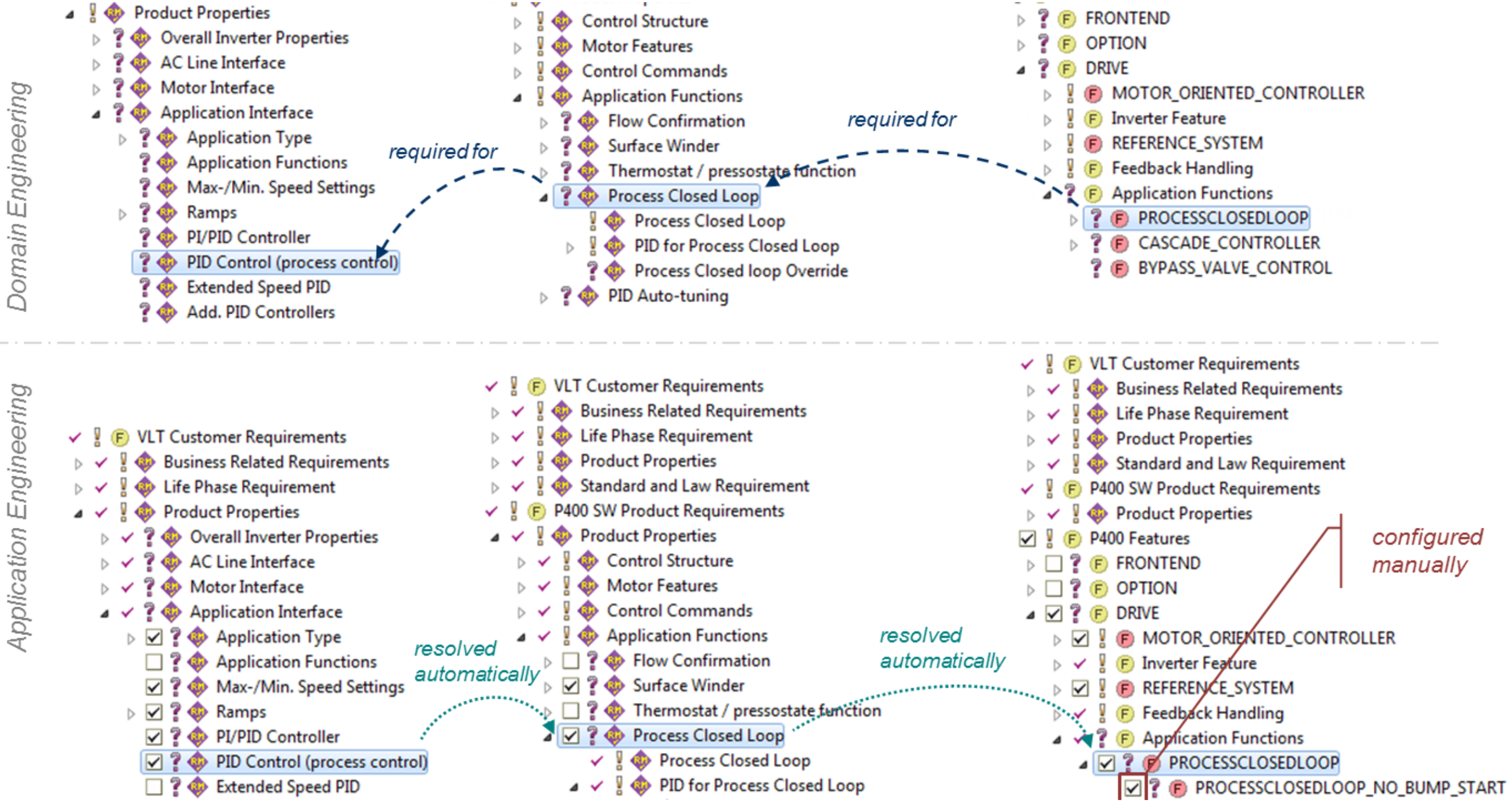
フィーチャのマトリックス



バリアントモデル一覧

SPL Models


ドメインエンジニアリング
要件、ソース、パラメータデータベース用の３種のフィーチャモデル
Domain Engineering
Problem Space
Collection of Features and Relation
Feature Model
問題空間
Requirements Feature Model:
Implementation Feature Model:
PFPD Feature Model:
Domain = VLT Drives P400 Series
Solution Space
Collection of Family Elements
Family Model
解決空間
P400 Requirements (CaliberRM)
P400 Source files (ClearCase)
PFPD Database (pure::variant)
要件(CaliberRM)、ソースコード(ClearCase=>2018年からgit)、固有のデータベース（制御用パラメータ）をpure::variantsで直接管理
Configuration Modelling Tool
(pure::variants)
３種のバリアント定義モデル(VDM)
Application Engineering
Requirements Variant Description Model:
Implementation Variant Description Model:
PFPD Variant Description Model:
Single Problem
Desired/Required Features
Variant Model
Product X Requirements Spec
Product X Technical Litterature Input
Product X Binaries and DLL
Test Collection to RQM
Product X Product Management Reports
Single Solution
Variant
Variant Realization
バリアントごとの要求仕様書、ドキュメント、ソースコード、テスト仕様書、管理者向けレポートを自動生成
アプリケーションエンジニアリング

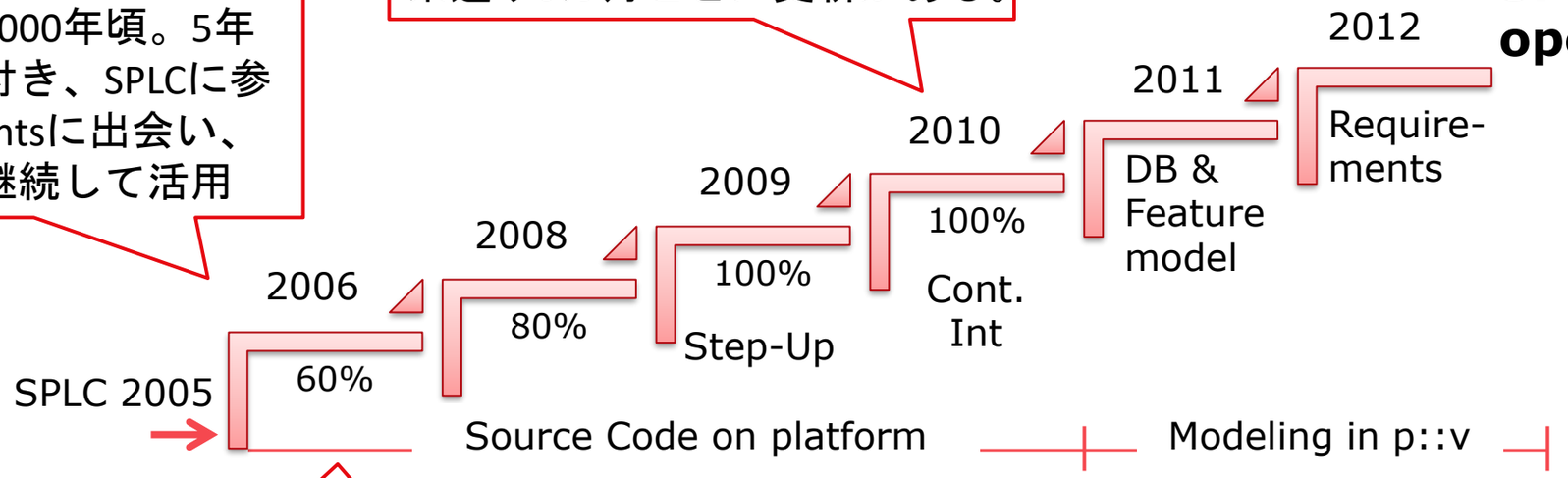

Domain Engineering
Product Properties
Overall Inverter Properties
AC Line Interface
Motor Interface
Application Interface
Application Type
Application Functions
Max-/Min. Speed Settings
Ramps
PI/PID Controller
PID Control (process control)
Extended Speed PID
Add. PID Controllers
required for
Control Structure
Motor Features
Control Commands
Application Functions
Flow Confirmation
Surface Winder
Thermostat / pressostate function
Process Closed Loop
Process Closed Loop
PID for Process Closed Loop
Process Closed loop Override
PID Auto-tuning
required for
FRONTEND
OPTION
DRIVE
MOTOR_ORIENTED_CONTROLLER
Inverter Feature
REFERENCE_SYSTEM
Feedback Handling
Application Functions
PROCESSCLOSEDLOOP
CASCADE_CONTROLLER
BYPASS_VALVE_CONTROL
Application Engineering
VLT Customer Requirements
Business Related Requirements
Life Phase Requirement
Product Properties
Overall Inverter Properties
AC Line Interface
Motor Interface
Application Interface
Application Type
Application Functions
Max-/Min. Speed Settings
Ramps
PI/PID Controller
PID Control (process control)
Extended Speed PID
resolved automatically
VLT Customer Requirements
Business Related Requirements
Life Phase Requirement
Product Properties
Standard and Law Requirement
P400 SW Product Requirements
Product Properties
Control Structure
Motor Features
Control Commands
Application Functions
Flow Confirmation
Surface Winder
Thermostat / pressostate function
Process Closed Loop
Process Closed Loop
PID for Process Closed Loop
resolved automatically
VLT Customer Requirements
Business Related Requirements
Life Phase Requirement
Product Properties
Standard and Law Requirement
P400 SW Product Requirements
Product Properties
P400 Features
FRONTEND
OPTION
DRIVE
MOTOR_ORIENTED_CONTROLLER
Inverter Feature
REFERENCE_SYSTEM
Feedback Handling
Application Functions
PROCESSCLOSEDLOOP
PROCESSCLOSEDLOOP_NO_BUMP_START
configured manually

The Journey

2006年にPLEを開始。プラットーフォームは2000年頃。5年たって課題に気付き、SPLCに参加してpure::variantsに出会い、現在に至るまで継続して活用

100%までの期間は長く見えるが、全ての既存の製品群を継続開発しながら、PLEのアプローチにマイグレートする必要があった。また各製品は従来通り6カ月ごとに更新がある。

その後範囲も拡大した。製品ごとのパラメーターデータベースを1つの共通パラメーターファミリーデータベースにして、最後の段階で要件も対象に。通常は要件から始めることが多いが、当初要件管理のマネージャがPLEに賛同しなかった。



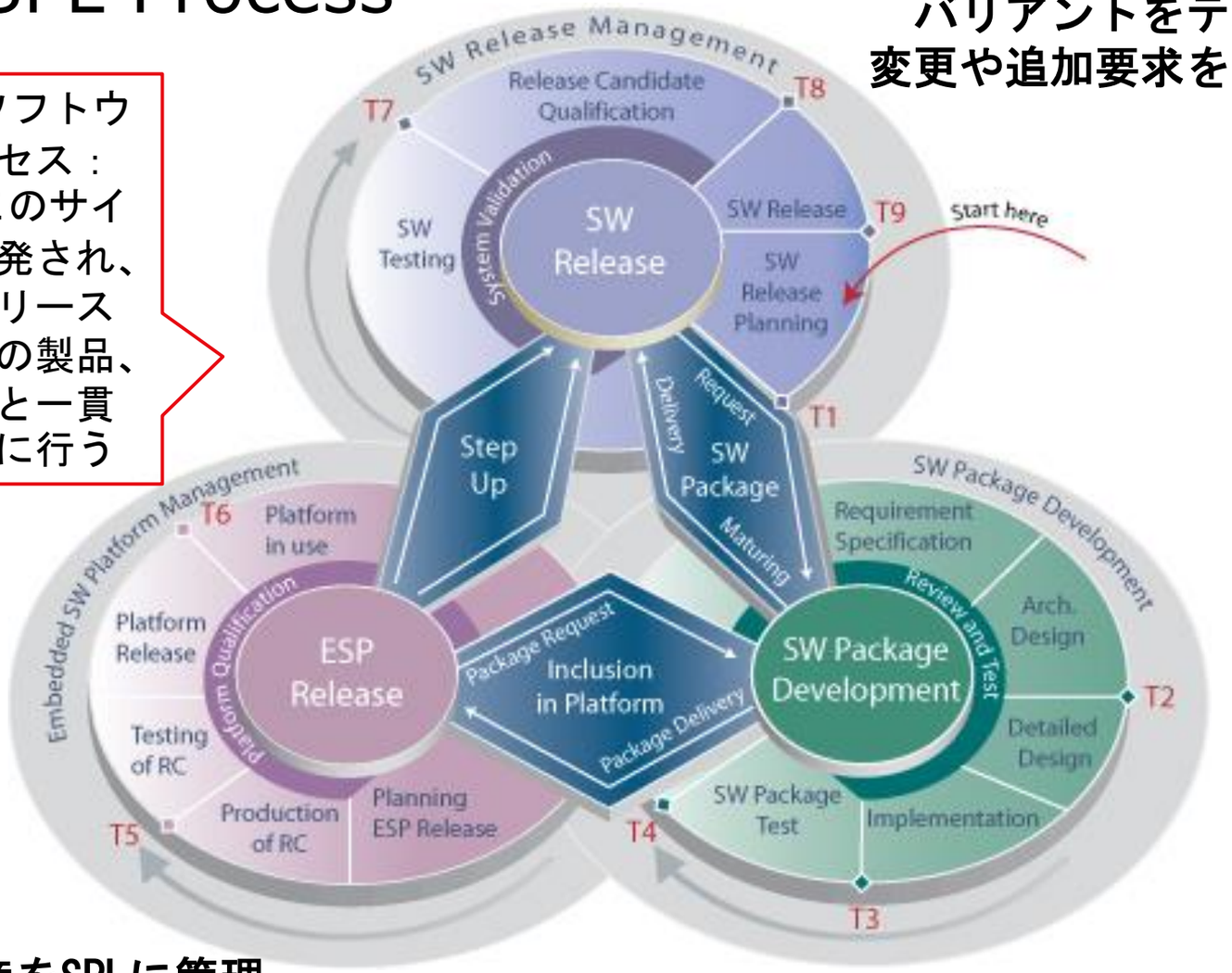
当初はSWだけに。再利用の範囲は60%（2カ月で達成、残りの40%は各プロジェクトに固有）、80%（2年後）、100%（また1年後）と段階的に拡げた。

2つの製品を選び、これらのマージから始めた。これが上手くいくことを見届けてから、残りの１０製品とのギャップを調査。マージされたものと同等のコードが有った場合は、それをマージされた方のコードに置き換えた。これで最初の60%の達成は比較的容易であった。複雑なものは手を付けないようにした。ただそのあとは複雑なコードにも対処したので、時間も費やした。一度に100%を目指さないことが良かったといえる。

Drives SPL-Process

SPLベースのソフトウエア開発プロセス：
全てのSWがこのサイクルの中で開発され、管理され、リリースされる。全ての製品、コードの品質と一貫性を保つために行う

各プロジェクトで利用して
バリアントをテスト
変更や追加要求を出す



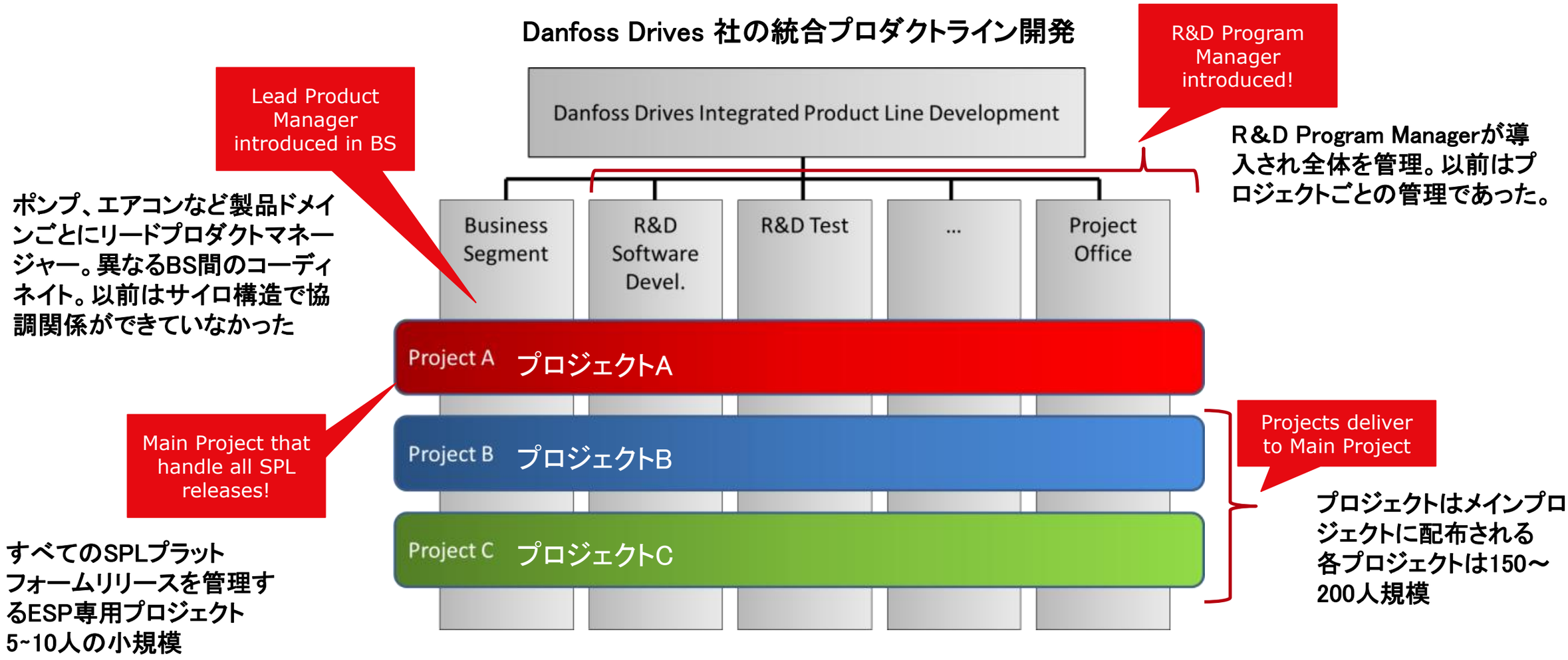
The 3 Circles
- SW Release cycle
- SW Package cycle
- ESP release cycle

The Toll Gates
- T1 – Release Planning is in place
- T2 – Arch design acceptable
- T3 – Implementation acceptable
- T4 – Test & platform elements acceptable
- T5 – ESP Release Candidate available
- T6 – ESP Official Release available
- T7 – Planned SW Testing completed
- T8 – Final Release Candidate Approved
- T9 – Official SW Release is completed

資産をSPLに管理
矛盾や一貫性など品質面も評価
ESP = Embedded Software Platform

SPLの資産を開発
新たなパッケージなどもここで開発

既存プロセスに新しく４つのロールが導入された



商業的成功

- プロダクトライン開発は、Danfoss Drivesが6番目に大きいVFDメーカから2番目に大きいVFDメーカに押し上げた要因の一つです。
- プロダクトラインは2004年に開始され、当初期待されていた寿命は10年でしたが、2016年現在、このプロダクトラインはさらに10年間使用される予定。
- バリアビリティを通じて顧客に専心したサポートができる、プロダクトラインを備えたプラットフォーム全体のコンセプトが、この成功の重要な要素。
- 製品数10倍増（20の主要製品と60のオプション）対して担当者数は5倍増

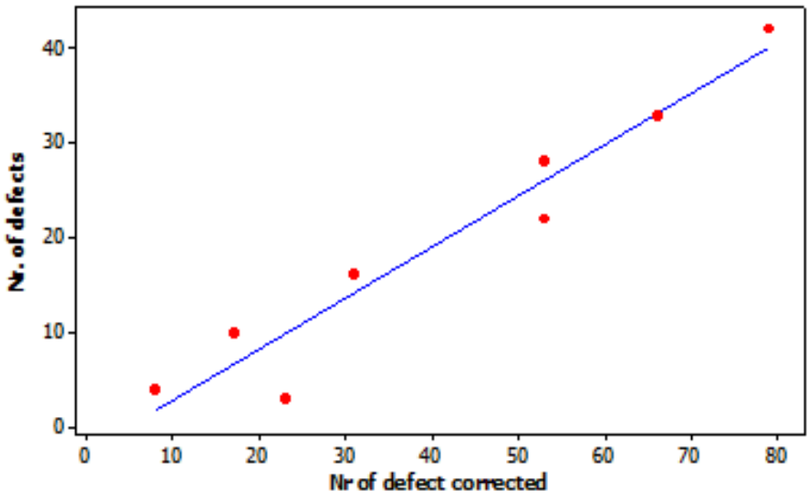
論文：Ten Years of Product Line Engineering at Danfoss: Lessons Learned and Way Ahead

https://www.researchgate.net/publication/308816044_Ten_years_of_product_line_engineering_at_Danfoss_lessons_learned_and_way_ahead

SPLのビフォーアフター/欠陥相関

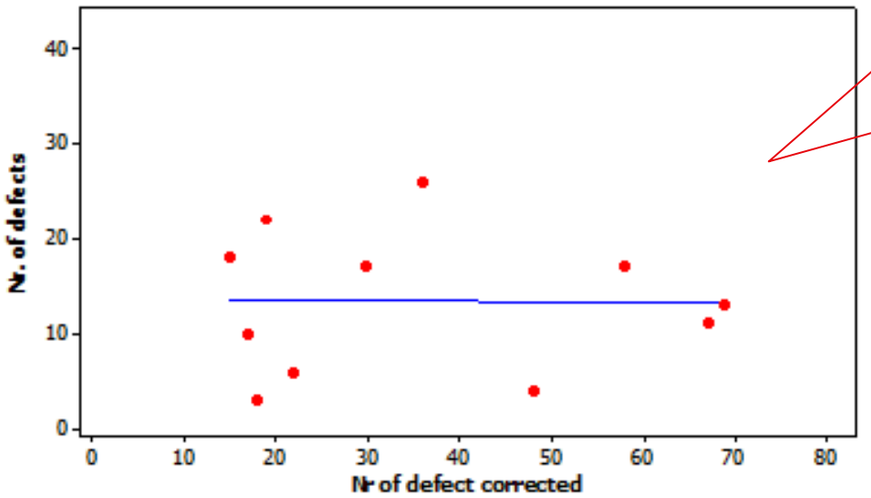
新たに検出されたバグの数

Before SPL Process



バグ修正の数

After SPL Process



バグ修正の数

SPL後はバグの修正によって新たなバグが出ない傾向であることがわかった！

SixSigma Study

Customer Examples Data

Starting and Running Success Product Lines – Real World Data

業界 /製品 プロジェクト内容	始める前の取り組み	PLの第一段階で管理できた資産	完全な適応までの期間	バリアント数	Tools
Industry Automation Frequency Converters	既存製品のコピーを修正 (Code)	2 ヶ月でソースコードの60%を再利用資産化	4–5 年 (Req, Code, Parameter Database)	~10-20 at any point in time	ClearCase, Git, Caliber, BuildForge, Inhouse, C++, pure::variants
Transportation Railway Signaling Systems	Requirements (Catalog Approach)	3–6 ヶ月 (Requirements)	18 ヶ月 (Req, Code, Tests, other assets)	初期段階は 2つで、後から追加	Doors, Inhouse, C/C++, pure::variants
Automotive Transmission Systems	部品の選択と #ifdef	3 ヶ月	6 ヶ月	50以上	ClearCase, ClearQuest, make, C, pure::variants
Automotive Airbag Systems	既存製品のコピーを修正 (Req), #ifdef (Code)	9 ヶ月 (まずReqから)	12ヶ月	100以上	Doors, RTC, Inhouse tools, pure::variants

フィーチャ数は40くらいから数千まで様々な事例がある

成功事例間の共通性は？」：これら製品は特定の成熟があり、またプロジェクトは数名ではなく50名以上。PLEで得られる効果と対する負担は共通。失敗例もある。その違いは、成功要素＝＞受け入れられるか、他のチームと上手くやれるか？変えることに痛みも伴うが、それに耐えられるか？

そもそもは従来式に痛みがあったので取り組みを考えたはず。エンジニアだけでなくマネージャを含めた組織内で皆が共通の認識を持てることが重要。ただいつでもみんなの賛同が得られるというわけでは無い。組織が十分な柔軟性を持つことができることが大きな成功要素。

また作業を継続する中で学びもあるはず。それを生かして改善を加えていけることも大切。

現状に満足できているか？がモチベーション。

FUJI SETSUBI

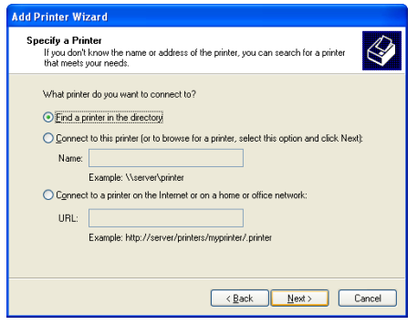
pure::variants について：
https://www.fuji-setsu.co.jp/products/purevariants/

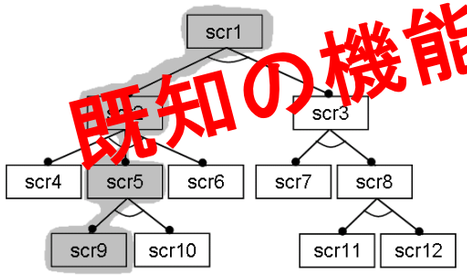
Dr.Danilo の実践的なプロダクトライン開発について
https://www.fuji-setsu.co.jp/products/purevariants/Danilo_Blog.html#PLE

Spectrum of variability mechanism*

Routine configuration → Creative construction

Wizards

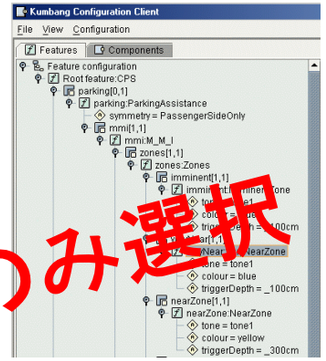


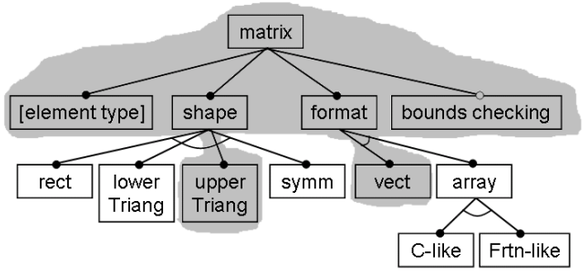


既知の機能の中からのみ選択

Path through a decision tree
All choices known
Implementation available

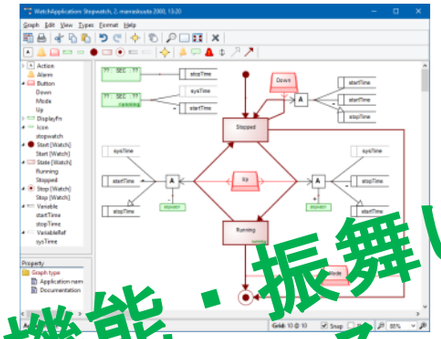
Feature-based configuration





Subtree of a feature tree
All features known
Feature implementations available

Domain-Specific Language



新たな機能・振舞いを追加出来る

Subgraph of an infinite graph
Variant space known, variants not
New features can be implemented

Czarnecki&Eisenecker 2000

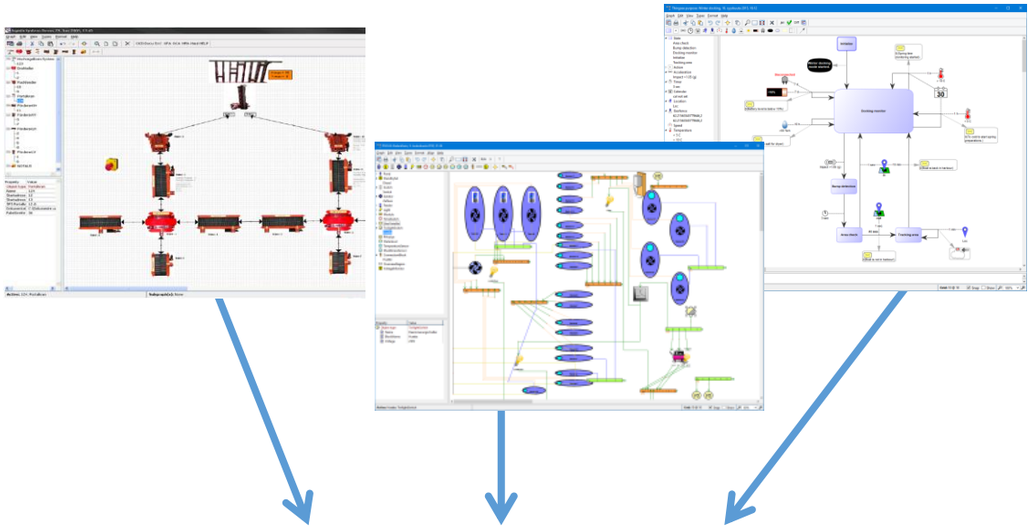
https://www.fuji-setsu.co.jp/files/jp-how2start.pdf

MetaCase

ドメインスペシフィックモデリング（DSM）言語による プロダクトライン開発事例

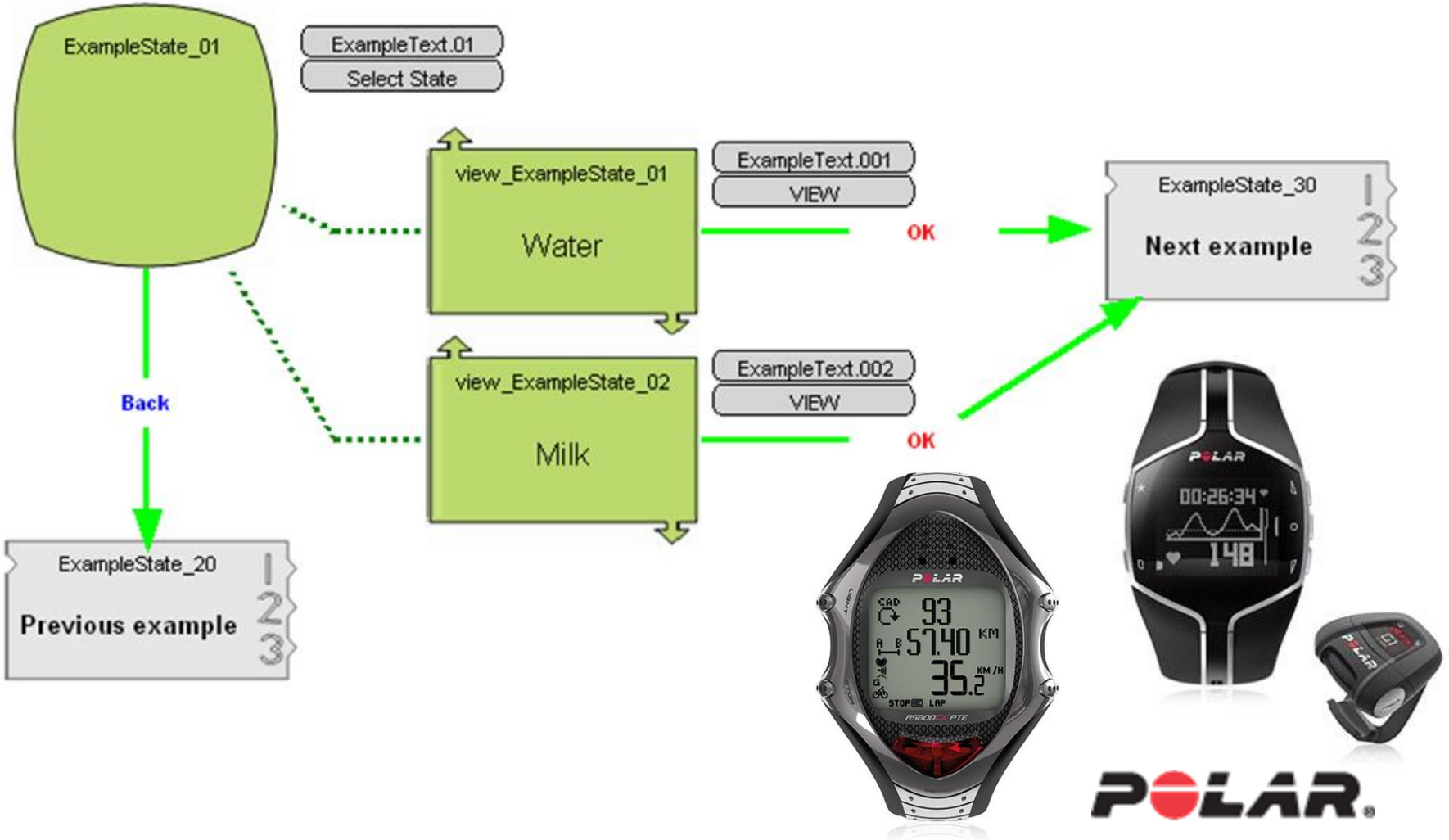
ドメインスペシフィックモデリング言語？

- **ドメインの知識を用いる**
  - ドメインのコンセプトやルールを採用
  - 開発抽象度をコードレベルから上げる
- **慣れ親しんだ用語・表現で開発できる**
- **自動生成：コードや様々な成果物に変換**



- 完全なコード
- コンフィギュレーション
- HW へのマップ
- シミュレーション
- テストケース
- ドキュメント
- その他

事例1: スマートウォッチのUIアプリ開発



・メニューから次のメニューや表示への遷移の選択肢を仕様化

・数十のエレメントがモデル図内に存在して、ひとつのアプリケーションは数十のモデル図があり、何十ものアプリケーションで製品が構成される

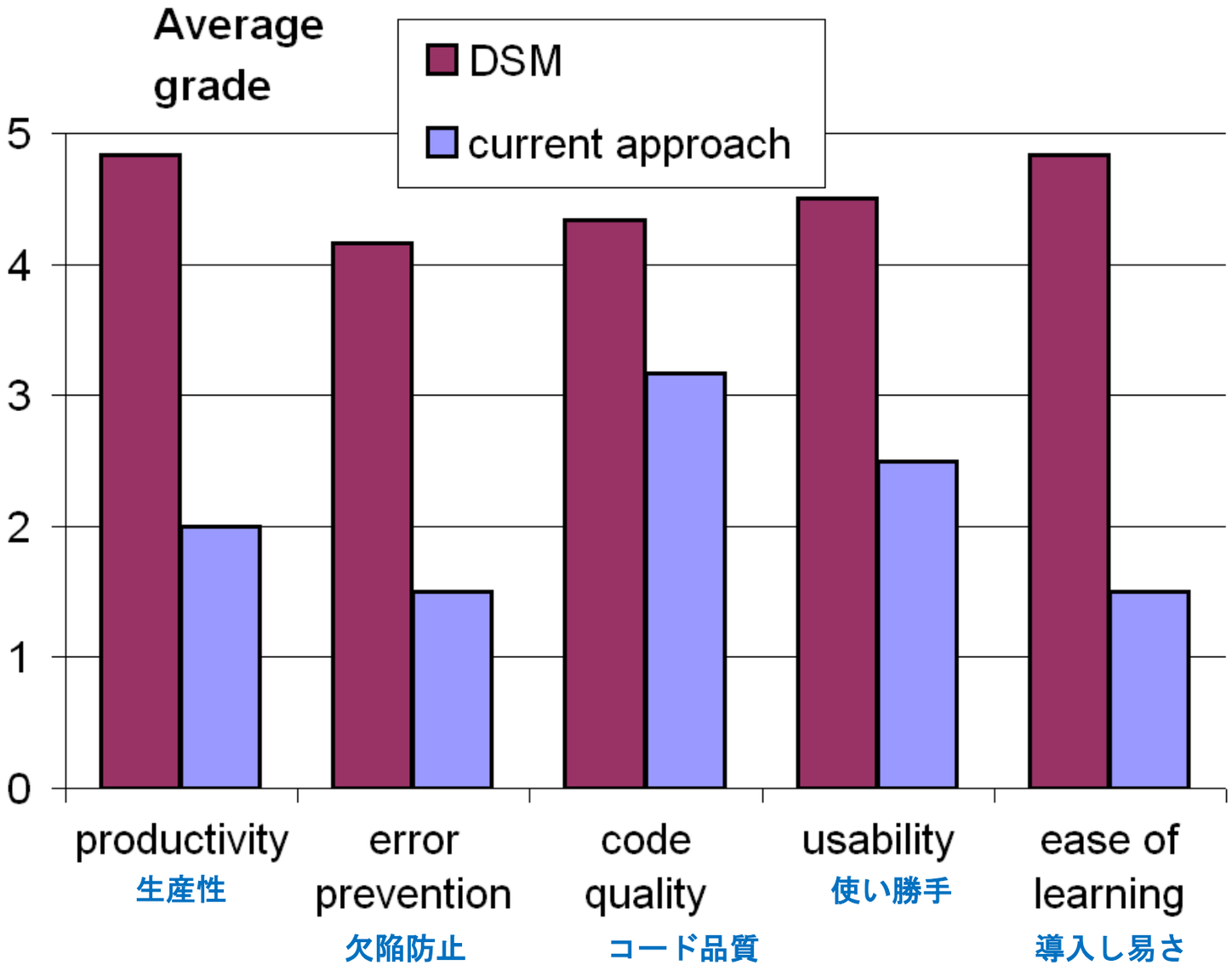
・単一のモデル図にある、ひとつのエレメントは、他のモデル図にリンク、参照され、また再利用される

・あるいはサブグラフにリンクされて更なる詳細を定義することもできる

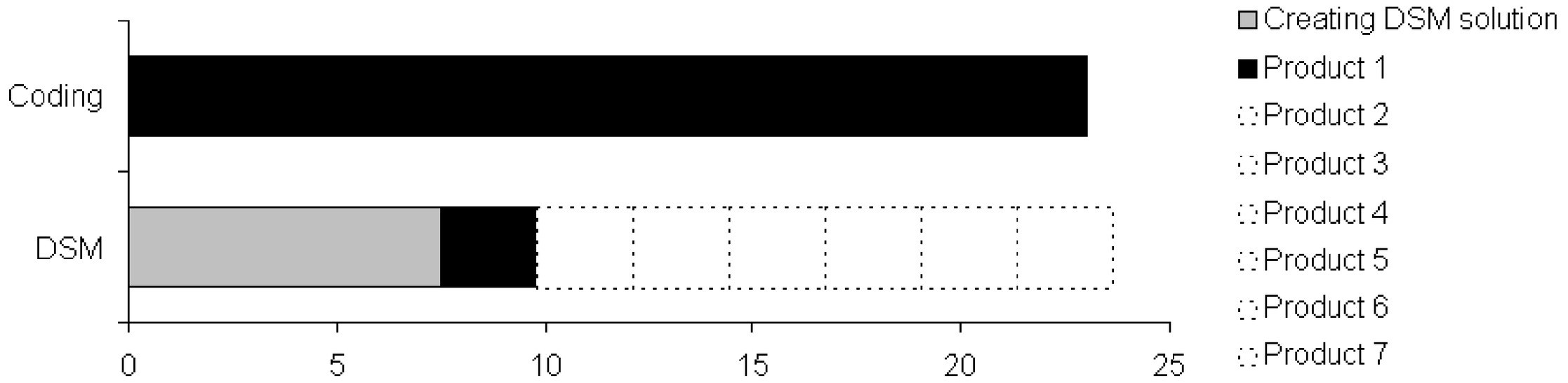
・そしてモデル図に対してジェネレータを実行して、完全なコードが自動生成される。

http://www.dsmforum.org/events/DSM09/Papers/Karna.pdf

DSMと従来手法を比較

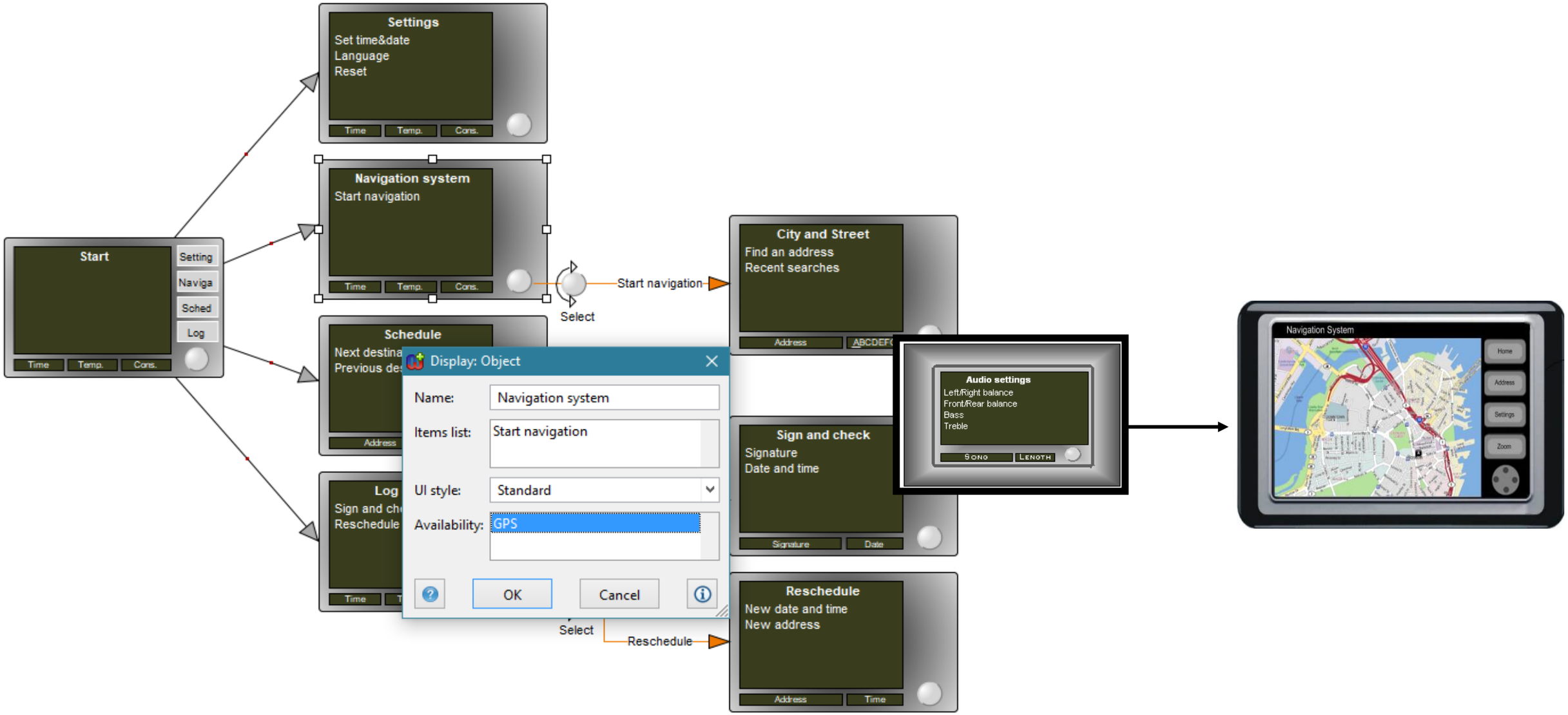


Polar社：投資対効果

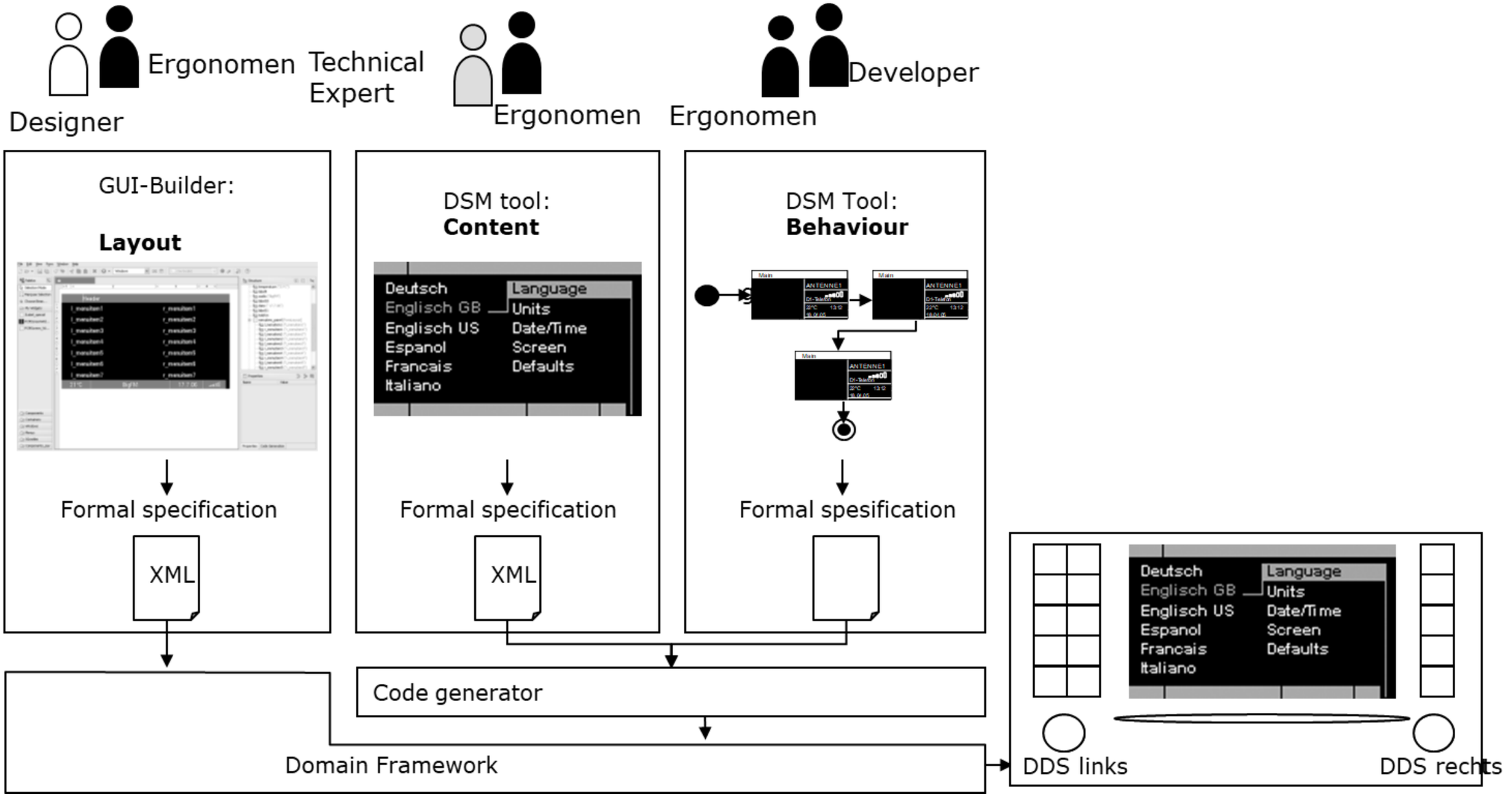


- DSM の開発は 60時間
- Product development with:
  - 従来手法: 23 日
  - DSM: 2～3 日

事例2:HMIの一貫性を改善しバリエーションに対応



マルチ言語、マルチユーザー対応



事例3: HUAWEI

ファーウェイはアウディと協力して、MDCソリューションを使用したL4自動運転に関する共同イノベーションを実施し、その結果は有望でした。中国のHuawei MDCを搭載したAudi Q7のテストは、不明瞭な車線、歩行者、自転車、スクーターが道路を横断するなど、複雑な交通状況下で夜間の薄明かりの都市部および農村部の道路で成功しました。また、高速巡航、他の車両の追跡、信号機と歩行者の認識、地下ガレージでのセルフパーキングを正常に実行しました。

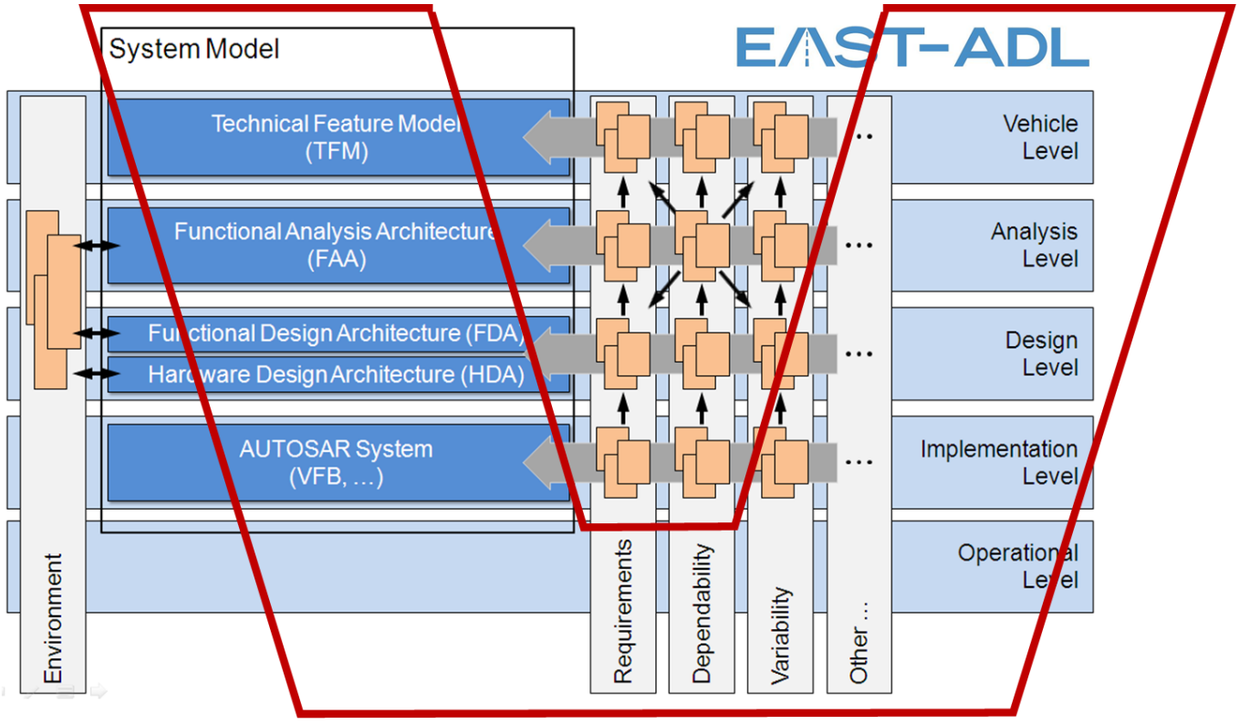
共同イノベーションを通じて開発された自動運転アルゴリズムは、KITTI 2D、3D、およびBEVテストで優れた結果を達成しています。

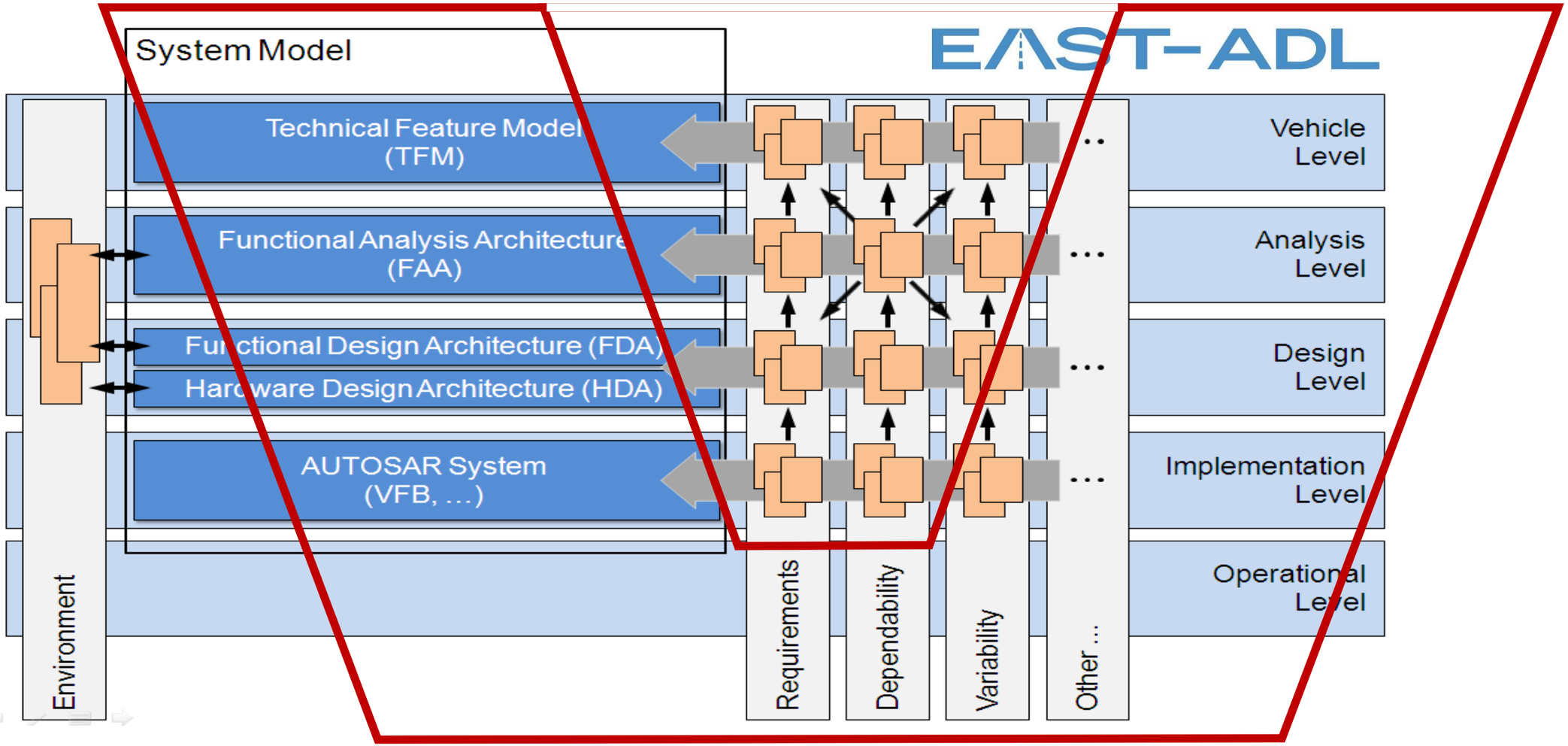
“MetaEdit+ is applied for development of Huawei's intelligent driving system”

https://www-file.huawei.com/-/media/corporate/pdf/publications/communicate/86/09-en.pdf

事例4：MBSE（モデルベースシステムズエンジニアリング）

- MBSEで仕様化したいこと
  - コンポーネントの構造
  - インターフェイス
  - 通信プロトコル
  - SWとHWアーキテクチャのマッピング
  - コンフィグレーション
  - アーキテクチャの分析
  - アプリケーション構成のためのガイドライン
- 単一言語では間に合わない
- 開発工程上で連携されるべき
  - コラボレーション開発
  - トレーサビリティとインパクトの解析
  - コード、メトリクス、テストケース、ドキュメントなどの生成

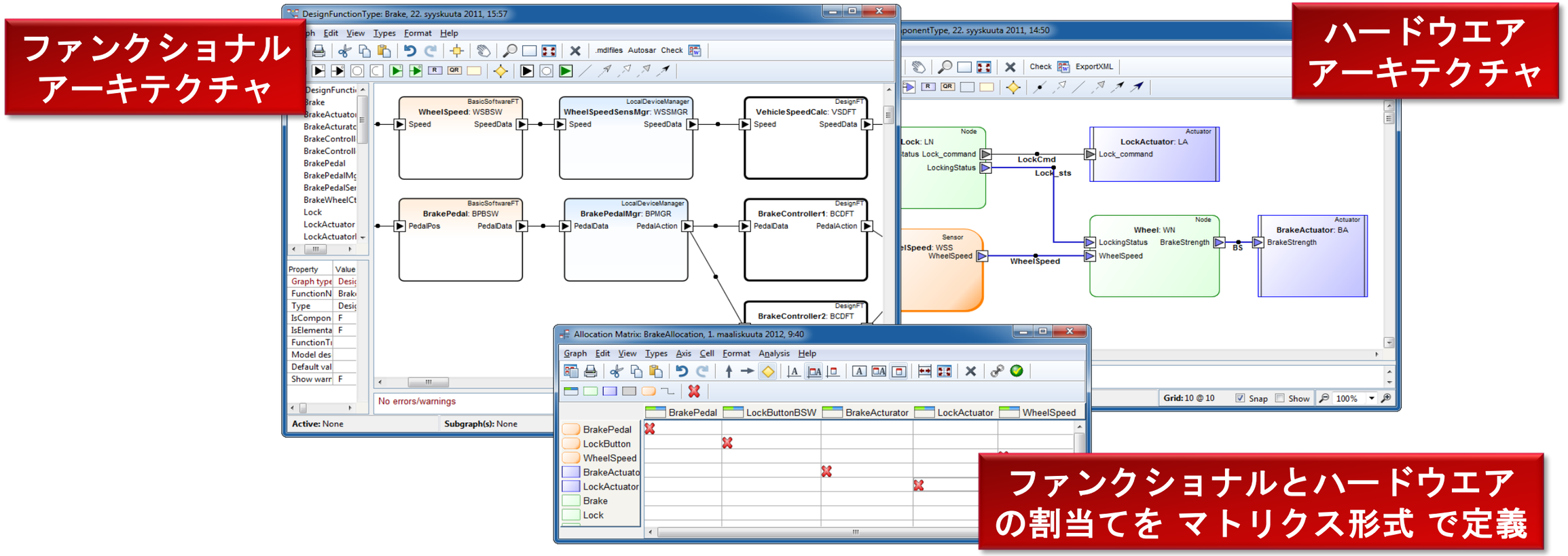




EAST-ADL（Architecture Description Language）は、SysMLのアイデアを採用してAUTOSARを高い抽象レベルで補完する自動車用のドメインスペシフィックモデリング言語。車両レベルの機能、機能構造、要件、変動性、ソフトウエア部品、ハードウエア部品、通信等の側面をカバーする。振舞いの表現に、Simulinkモデルを活用することや、プロダクトライン開発のコンセプトとしてフィーチャモデル、ISO 26262の認証プロセスを支援する、デペンダビリティモデル、エラーモデルなど安全設計（ハザードや障害の影響範囲の分析）をサポートして、開発プロセスを早期段階から実装に至るまで、一貫して支援する。

EAST-ADL モデリング言語をMetaEdit+で実装

Volvo, Fiat, Daimler, Continental, Bosch



https://www.fuji-setsu.co.jp/products/MetaEdit/SystemModeling.html#EAST-ADL

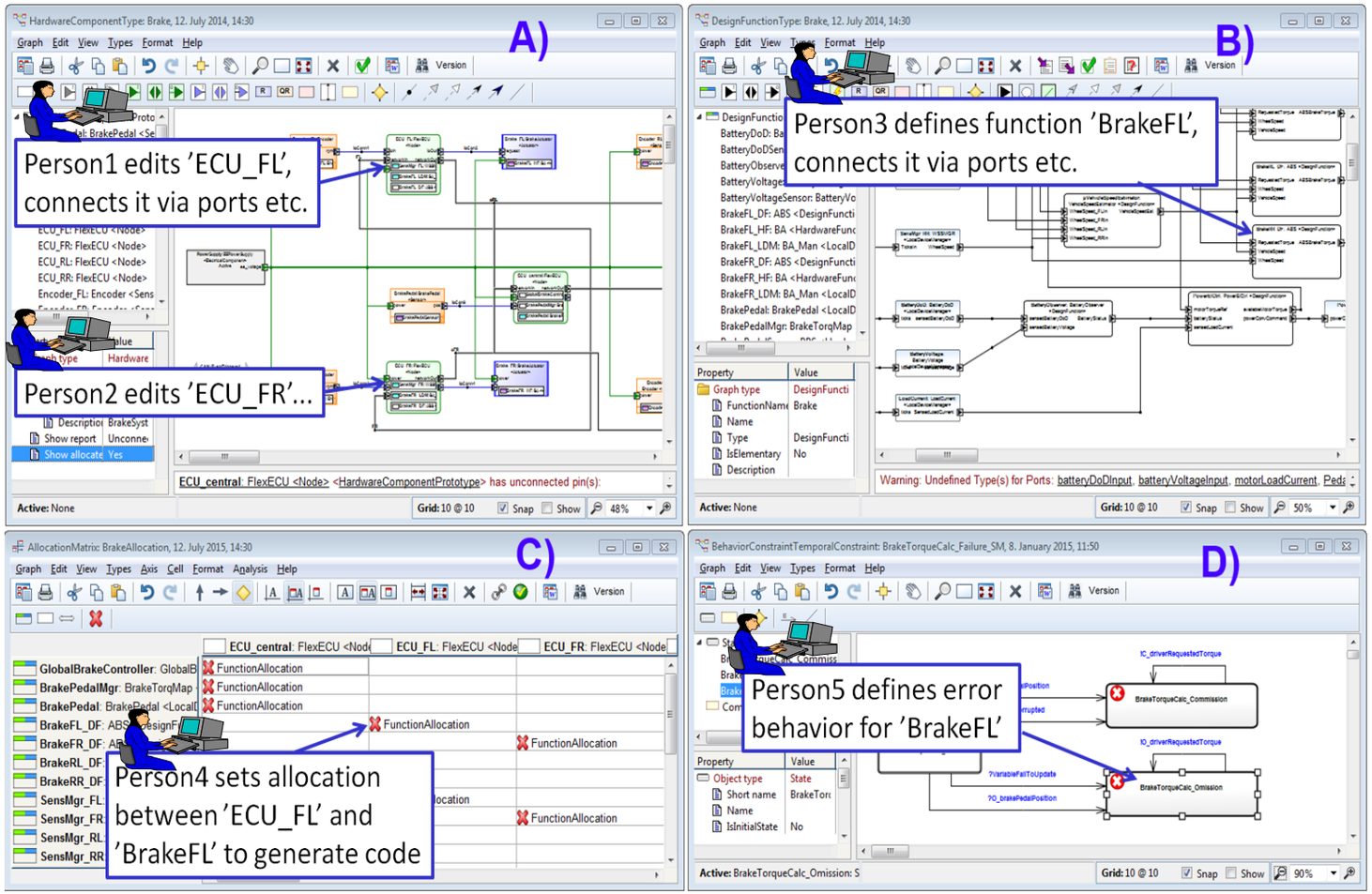
車載システムモデリングのコラボレーション

2人の担当者が同じモデル図 A) の異なるハードウエア部品を編集。

同時にB)ではシステムの論理コンポーネントを別の担当者が定義していて

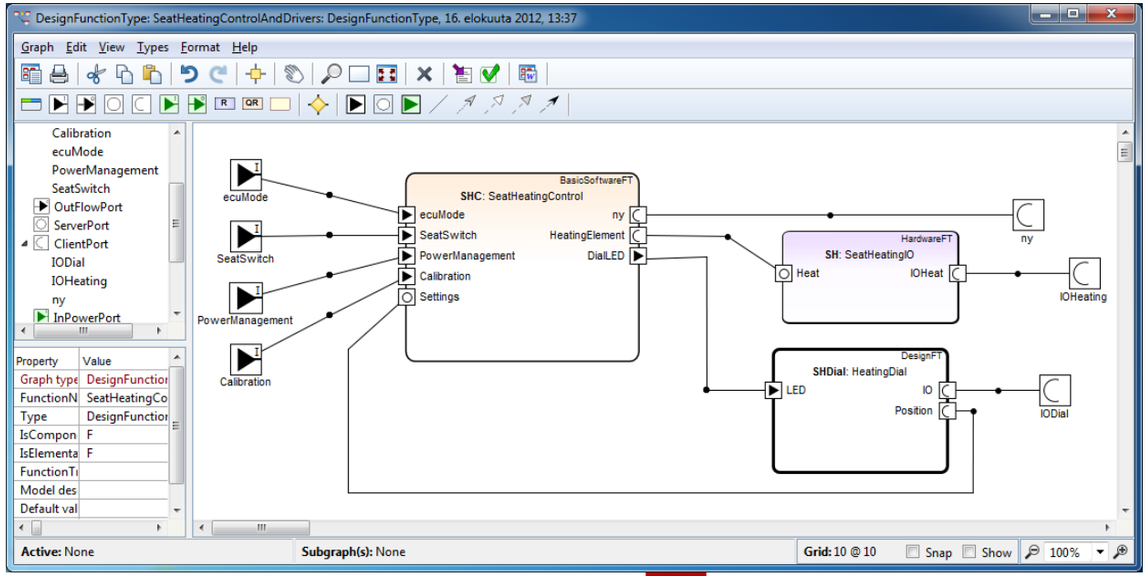
C)ではコードを生成させるために、A)とB)で編集中のハードウエアと論理アーキテクチャ間のアロケーションを定義している。

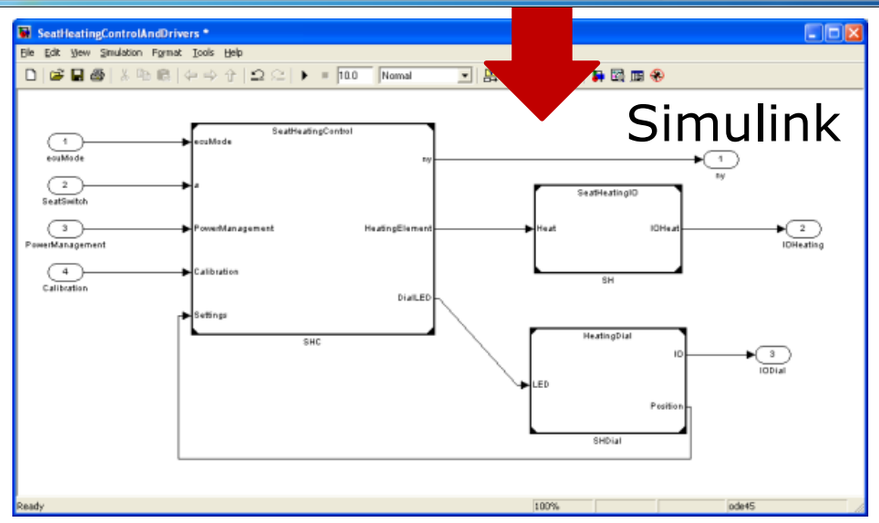
そしてD)では機能安全担当がB)で定義される論理コンポーネントのエラーモデルを定義。

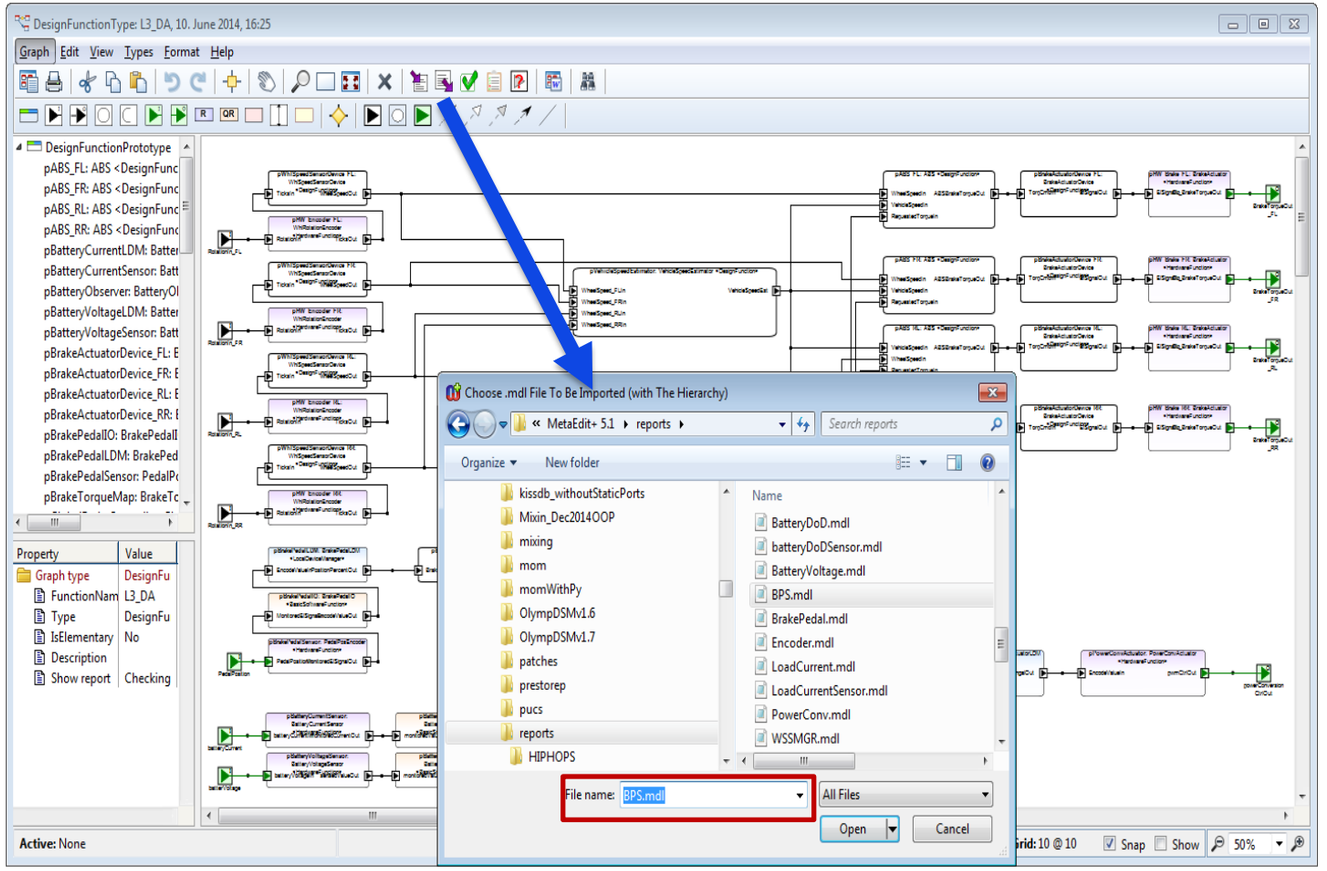


動画デモ： https://www.fuji-setsu.co.jp/demo/EASTADL_HW_Allocation.wmv

Simulinkモデル生成、モデルインポート





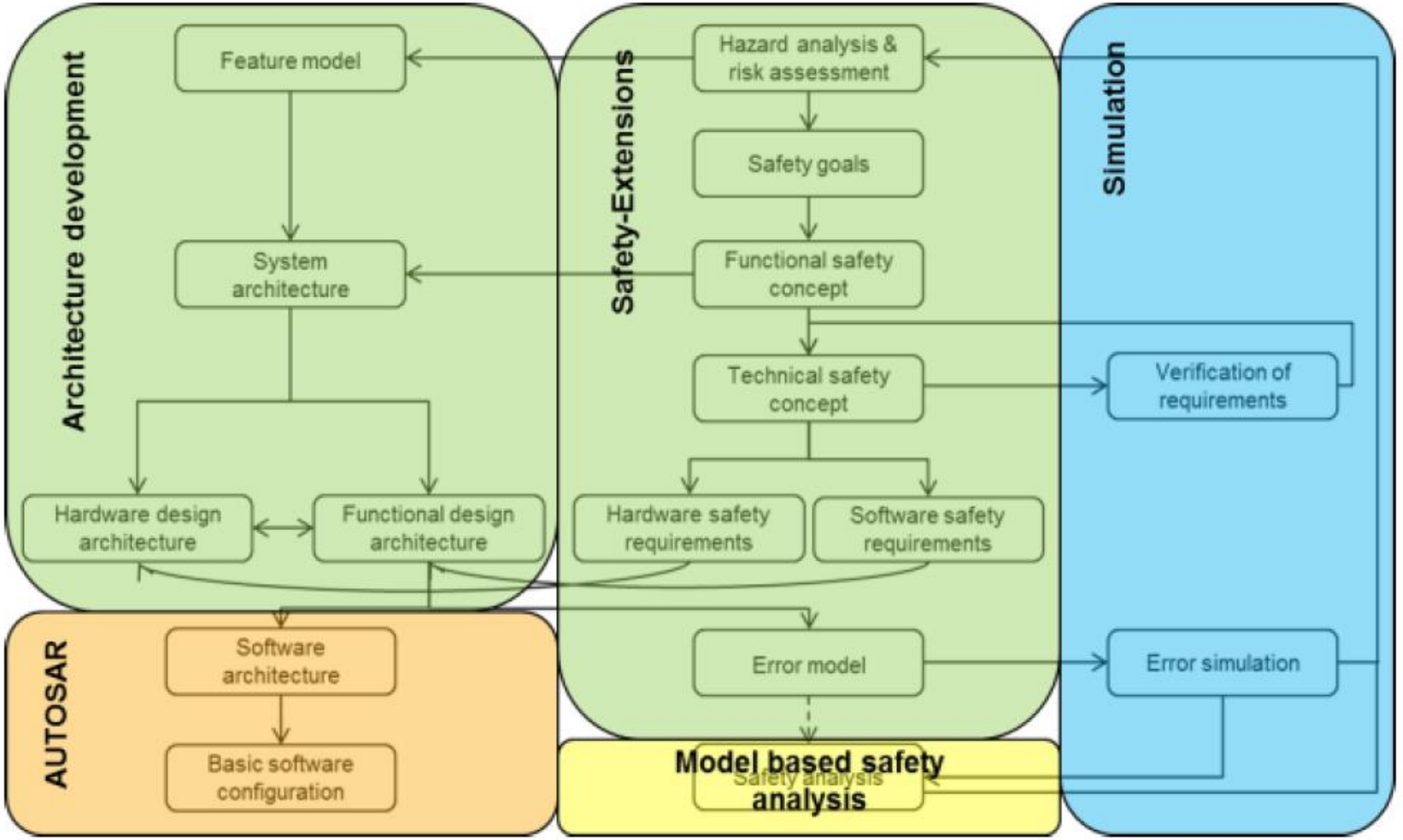


動画デモ： https://www.fuji-setsu.co.jp/demo/EASTADL_Simulink.wmv

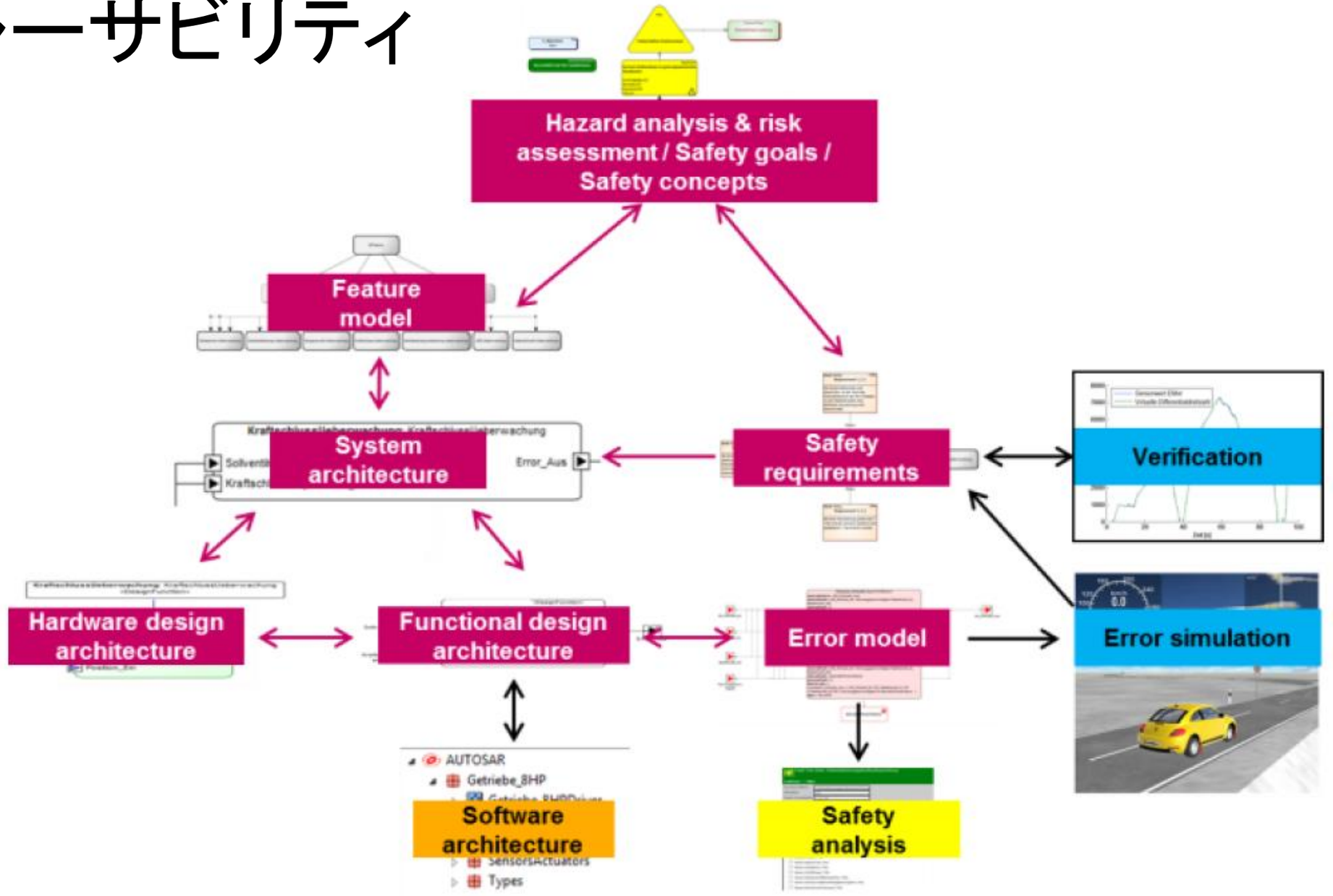
DSM: システムモデリングへの効果

- 重複作業を排除
  - 変更は全ての成果物に同時反映され通達もされる
- コラボレーション開発
  - 比較（diff）とマージの必要無く、共同開発できる
- 全プロセスに渡ってのトレーサビリティ
  - 要件からコードまで全ての成果物でバイディレクショナルに
- モデル変換・コード生成
  - Simulinkモデル、Cコード、各種ドキュメント
- 様々なツールと連携
  - 定理証明、テストベクタ生成など
- 組織ごとの需要に応じてカスタマイズ
  - ルールや制約の追加、各種ジェネレータ、ツール統合など柔軟に対応

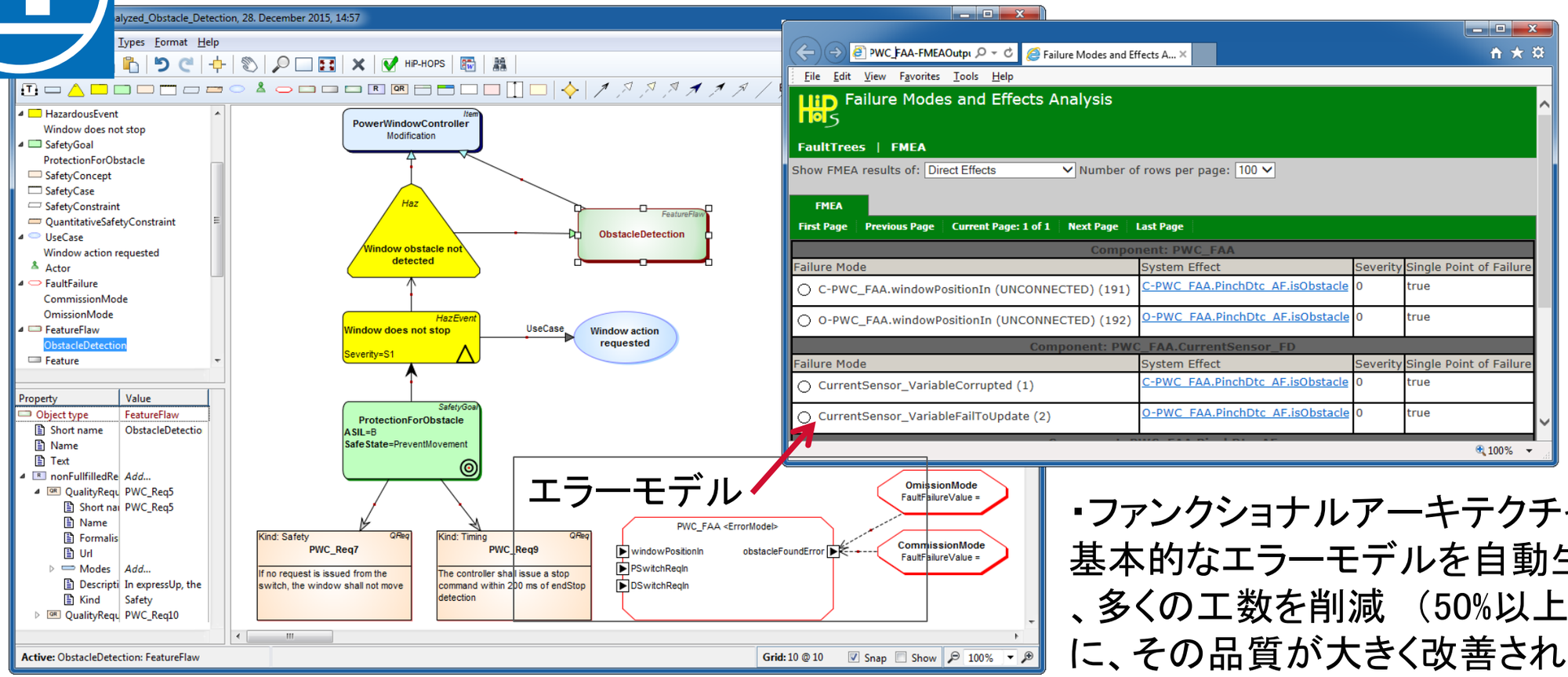
EAST-ADLを拡張してISO 26262に対処



システムおよび安全性設計間で双方向のトレーサビリティ



ISO26262 機能安全設計と解析の自動化



・ファンクショナルアーキテクチャモデルから基本的なエラーモデルを自動生成することで、多くの工数を削減　（50%以上）すると同時に、その品質が大きく改善された

・安全設計技術者は、実際に計画されるアーキテクチャで安全設計の側面を考察できる

・FTA/FMEA解析ツール用のデータもエラーモデルから自動生成

一貫したモデルベース開発を効率化

- システムおよび安全性アーキテクチャの開発をEAST-ADLで統合
- ASILを考慮した安全目標と機能安全のトレーサビリティの自動化
- 機能安全コンセプトと技術安全コンセプトのシミュレーション検証を早期開発段階で実施
- ハザード分析およびリスク評価から安全要件に至る、ISO 26262作業成果物をモデルベースで自動化
- 体系的なエラーおよびエラーのシステムへの影響の早期検出

原文： https://astesj.com/v02/i03/p158/

日本語訳公開： https://www.fuji-setsu.co.jp/products/MetaEdit/SystemModeling.html#EAST-ADL

Industry experiences

Elektrobit

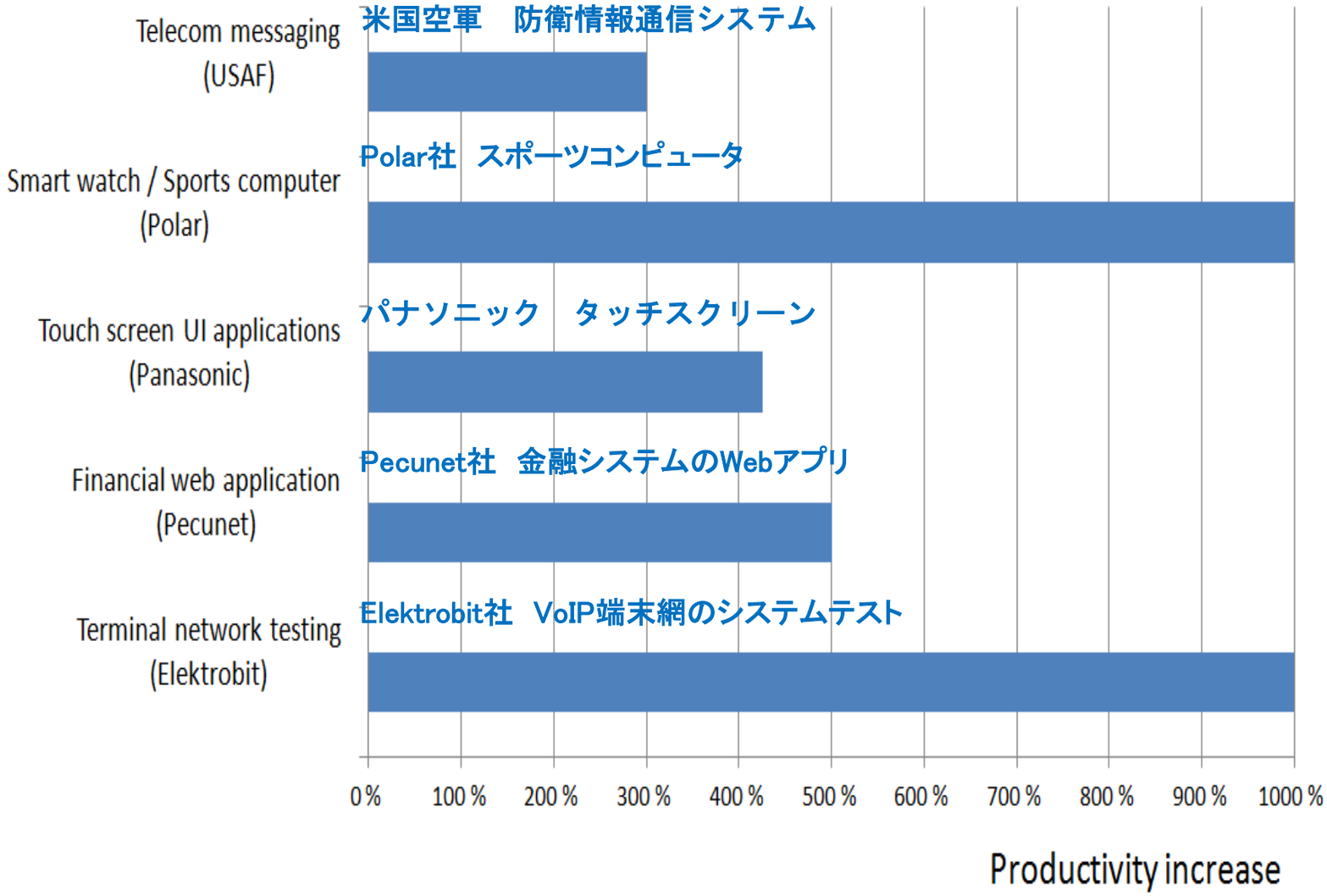
"provided **more than ten times** faster speed in comparison to previous experience"

AIRBUS

"The quality is clearly better, simply because the modeling language **rules out errors**"

POLAR

"**750%** increase in productivity, and **greatly** improved quality in the code and development"



ドメイン固有のモデリング言語がどのように変動性に対処するか：23 の事例調査

How DSM Address Variability: 23 cases

	Domain	Targets	By	Size	Use	Approach
1	Consumer electronics	C, HTML, Docs	1		3	1
2	Industrial automation	PLC, GUI, DB schema, net config, deploy	1	165	5	1
3	Enterprise applications	C#, DB schema	1		6	1
4	Railway signaling	Simulation, XML	1	291	6	1
5	Signal Processing Systems	Matlab, simulation, XML	1		4	1
6	Oil drilling	Cost calculation, documentation	1		3	1
7	Big data applications	Java, JSON, CQL, SPARQL, SQL	2	397	4	1
8	Printing process	Ruby, XML, Docs	3	55	4	1
9	System performance	Gherkin, HTML, Docs	1	145	3	1
10	Consumer electronics	JSON	3	72	2	2
11	Telecom service	XML	1	61	2	2
12	Medical	XML, audit documents, change history	1	63	1	2
13	High-level synthesis	System C	1	450	3	2
14	Radio network	TTNC-3, simulation/animation	1		5	2
15	An automotive system	System specification	3	62	3	2
16	Database applications	Java	3	46	1	2
17	Consumer electronics	C, localization, docs	1	403	6	3
18	Automotive architecture	Simulink, ISO26262 documents, AUTOSAR	3	652	6	3
19	Telecom	C, build automation	2	109	3	3
20	Insurance	Cobol, DB schema	3	234	6	4
21	Aerospace	C#, XSD, JSON, API	2	121	1	5
22	Automotive ECU	Python, JSON, Test document, change history	3	64	5	5
23	Software testing	Propriety format of state machines	3	317	6	6

SPLC 2019講演資料/日本語訳版： https://www.fuji-setsu.co.jp/products/MetaEdit/tutorials.html#PLE

DSM言語：生産性と品質を飛躍的に向上

- 言語のコンセプトを問題空間から得ること
  - コードの視覚化では無く
  - 特定ドメインにのみスコープの範囲を限定する
  - 狭いほど良い（必要に応じて拡張できる）
- 市場の持続的な変化に応じた製品開発の支援には、DSM言語への淀みない進化・変更が必要になる
  - モデルの開発、アップデート、チェックに要する作業は最小限にする
  - モデルチェックやリファクタリングを支える仕組みが必要
  - 利害関係者間の意思疎通をサポートできること

様々な事例： https://www.metacase.com/ja/solution/automotive_transport.html

FUJI SETSUBI

・Domain-Specific Modeling について

https://www.fuji-setsu.co.jp/products/MetaEdit/tutorials.html

・ドメインスペシフィックモデリング（DSM）言語によるシステムモデリング事例

https://www.ipa.go.jp/files/000057615.pdf

富士設備工業（株） 電子機器事業部

https://www.fuji-setsu.co.jp