DENSO
Crafting the Core

SPI Japan 2024

セマンティック技術を用いた エンジニアリングプロセスの構築手法の提案

2024年10月17日

デンソークリエイト　山路 厚

自己紹介

山路 厚（やまじ あつし）

株式会社デンソークリエイト
ソフトウェア現場改善 エバンジェリスト
デザインDX エバンジェリスト

TIME TRACKER NX
Lightning Review
NEXT DESIGN

【略歴】　1986年日本電装株式会社入社
カーナビゲージョン・エンジン制御ECUソフトなど多くの車載組込みソフトウェアの開発の従事。
膨れ上がっていく大規模ソフト開発のプロジェクト・マネジャ経験の中から、
プロセス改善の本質は“人を育てること”であると気づき、トレーニング指向アプローチを提唱。
2007年より、**トレーニング指向アプローチ**による適用事例の社外発表活動を開始。
以降、現在まで人の能力を**最大限**に引き出し伸ばしていくシステムづくりを開発現場の中で取り組む。

【受賞実績】
・2007年 SPES2007:ベスト・プラクティス賞
・2008年 SPES2008: 2 年連続ベスト・プラクティス賞
・2009年 SPES2009: 3 年連続ベスト・プラクティス賞
・2009年 SPI Japan2009:プログラム委員長賞
「トレーニング指向アプローチの土壌作り」
・2010年 SPES2010:ベスト・プレゼンテーション賞
・2014年 ソフトウェア品質シンポジウム:SQiP Best Paper Future Award

【社外向け研修】
・Star研修
・ピアレビュー研修

【その他社外活動】
・SPES 企画WG委員
・JDMF 企画WG委員
・SIS 経験報告査読委員
・システム開発文書品質研究会（ASDoQ）会員

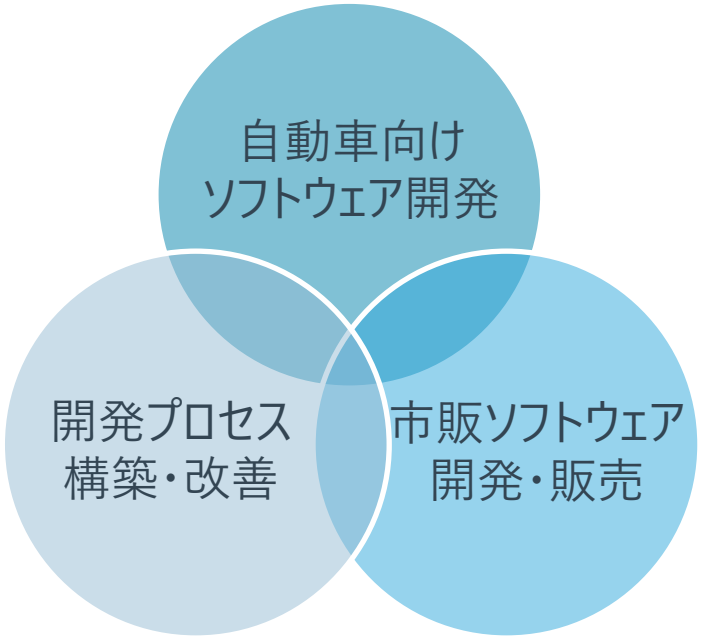
【書籍】
・ソフトウェアドキュメンテーション
進め方ガイド

会社紹介：創立34年目

社名	株式会社デンソークリエイト DENSO CREATE INC.
設立	1991年2月14日
事業内容	●自動車向けソフトウェア開発 ・先行研究開発 ・基盤ソフトウェア開発 ・開発支援ソフトウェア開発 ●開発プロセス構築・改善 ●市販ソフトウェア開発・販売
資本金	9,500万円
従業員数	約300名
株主	株式会社デンソー 100%

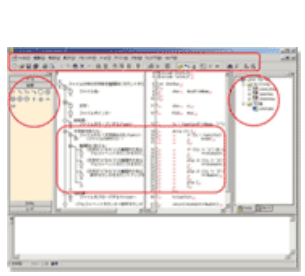
基本理念

時代に聞き、時代を拓き、
最高の品質を商品とし、
お客様と喜びを分かち合う。

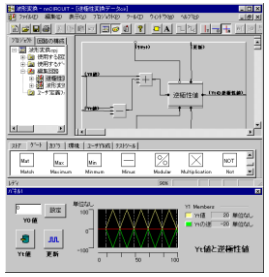


ソフトウェア開発の専門会社、ソフト生産技術をもってDN製品開発を支える

会社紹介：開発現場の改善活動



nnHEADWAY/P
構造化設計ツール



nnCIRCUIT
制御モデル設計ツール

工数管理
プロジェクト管理ツール
TIME TRACKER NX

設計レビュー支援ツール
Lightning Review

デザインDXツール
NEXT DESIGN

1994 1995 1998 2004 2017 2020 2024

トレーニング指向アプローチによる開発現場改善

SPI-Japan発表

2007 2008 2009 2010 2011 2012 2013 2014 2018 2019 2020 2021 2023 2024

組込みソフト開発現場の知見をコンテンツに反映、世の中に発信・洗練

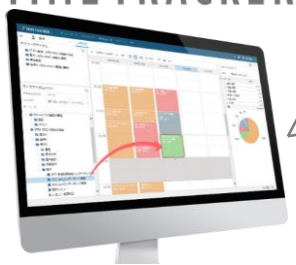
会社紹介：Identity。私たちが大切にしてきたもの

トレーニング指向アプローチ：仕事を通して人を育て続けたい

Lightning Review

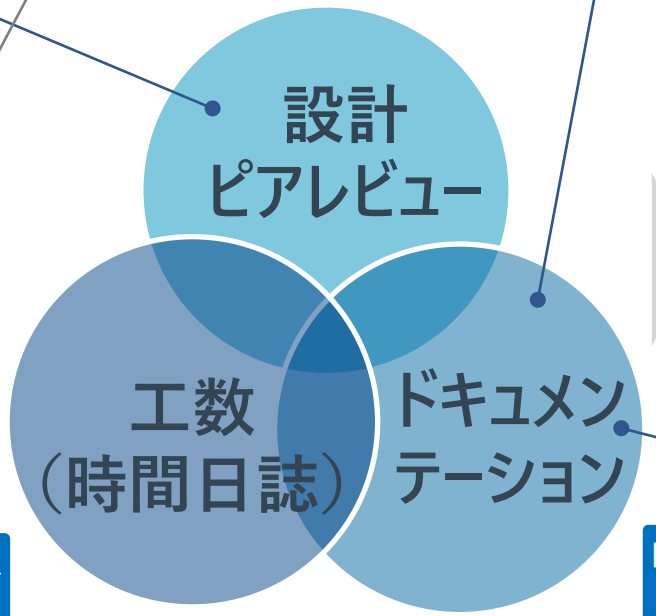
ピアレビュー研修

TIME TRACKER NX

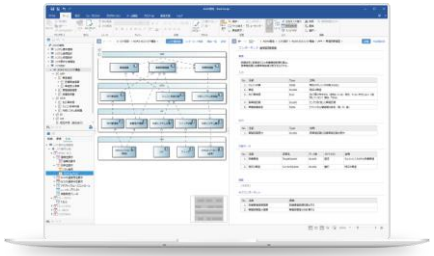


工数・進捗管理研修

①設計重視
②プロセス重視



NEXT DESIGN
nnHEADWAY/P
nnCIRCUIT



③カイゼン文化
SUCCESS 活動

ドキュメンテーション研修

自分達のために作り、自分達の開発の中で使い続けている
業務の成果と人材育成を同時達成させるために培ってきた “Identity”

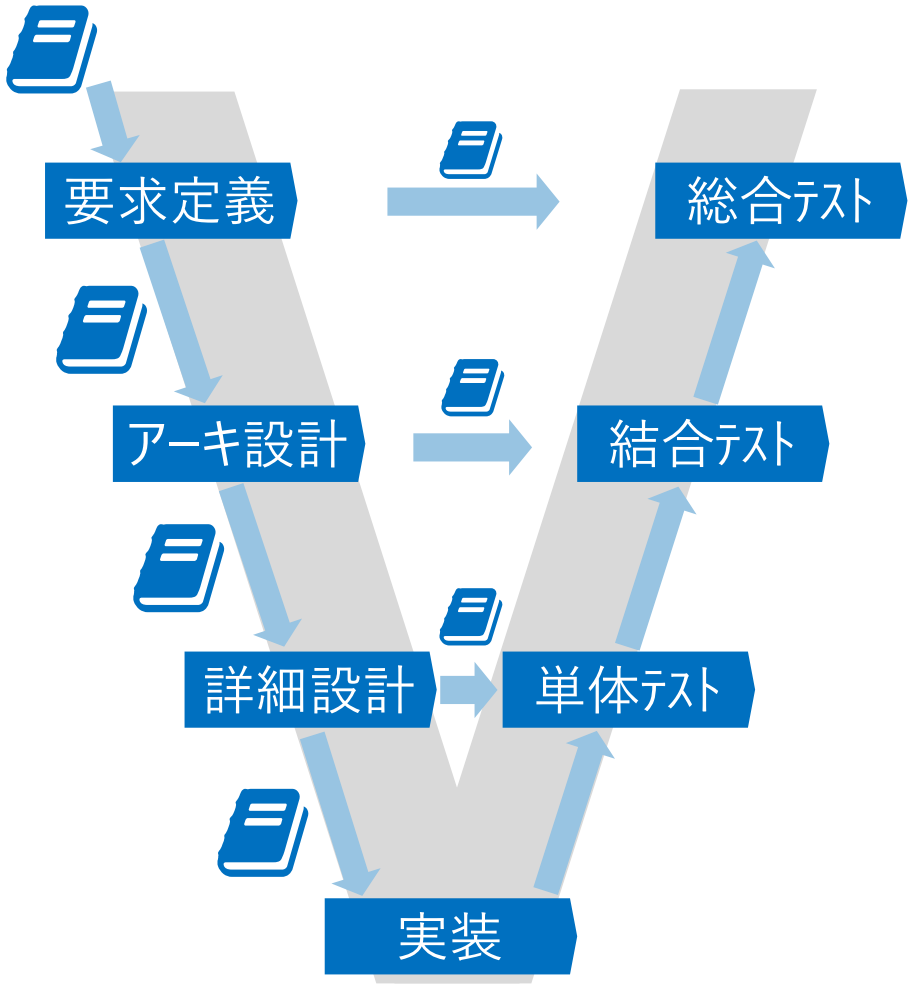
アジェンダ

1. 背景
2. 観察
3. 気づき
4. 着想
5. 提案方式
6. 期待効果
7. 効果確認
8. まとめ

1.背景：エンジニアリングプロセス

エンジニアリングプロセスって、“どうやって”決めていますか？

1.背景：エンジニアリングプロセス



【エンジニアリングプロセス】
入力文書をもとに
何を設計するかを**出力文書**で規定する

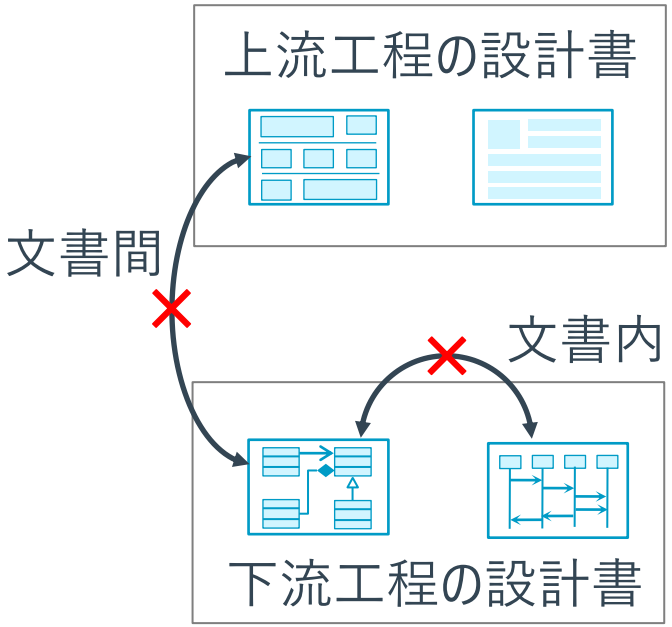
ここで行う作業を
出力文書で表現

・既存の成果物
・標準テンプレート
＋ガイド

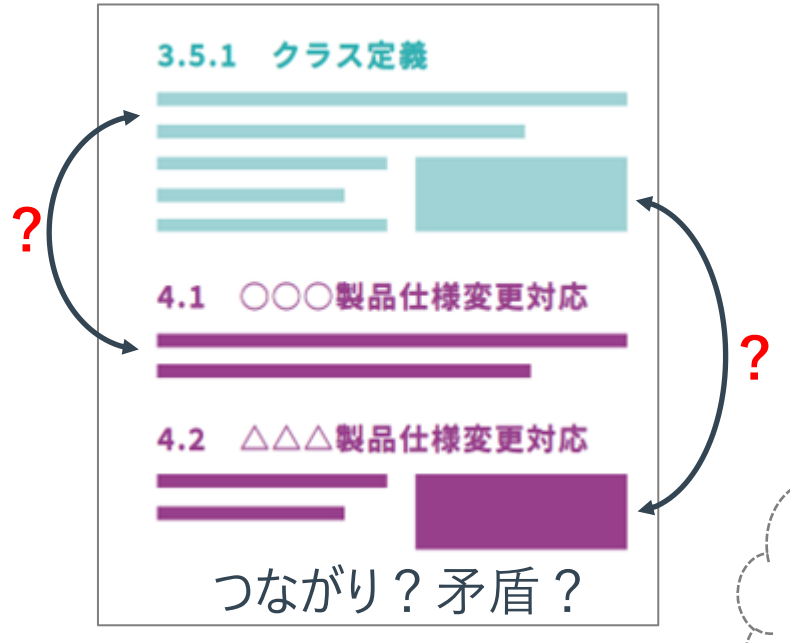
ドメインに特化した技術文書。**既存の成果物**や**標準テンプレート**を利用することが多い

1.背景：設計文書にかかわる問題

不整合の発生



変更設計書の作成



後追いトレーサビリティ

要求仕様	基本設計	詳細設計	ソースコード	単体検査仕様	結合検査
ドライバのスイッチ操作からクルコンON要求の有無を判断する	クルコン制御状態	スイッチ状態判定			
		クルコン状態判			
		制御状態取得			
	スイッチ状態	スイッチ信号A			
先行車の有無をドライバに通知する	先行車有無	先行車検出			
	先行車情報	車間距離計算			
	外部システム情報	状態取得			
		表示情報変換			

トレーサビリティマトリクス

開発後に作成、ムダに感じる

そもそも
何かが
おかしい？

文書間や文書内で設計情報の**不整合**や**重複作業**が発生しやすい
このような状況に対し、**継続的なプロセス改善**を施してきた

2.観察（1/4）：設計行為は、段階的に詳細化

コンポーネント分割

1. レイヤ APP

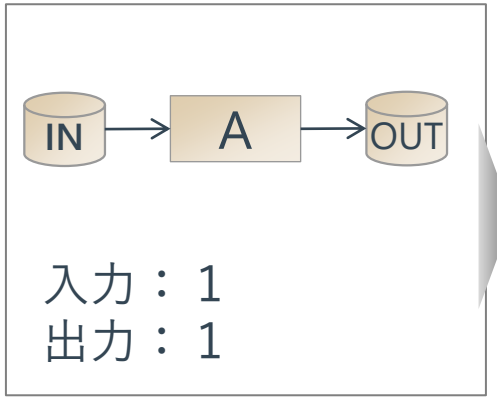
責務

ユーザに対するサービスを実現する。
本システムでは、速度と車間距離の差分から、目標制御量（加減速量）の算出を行う。

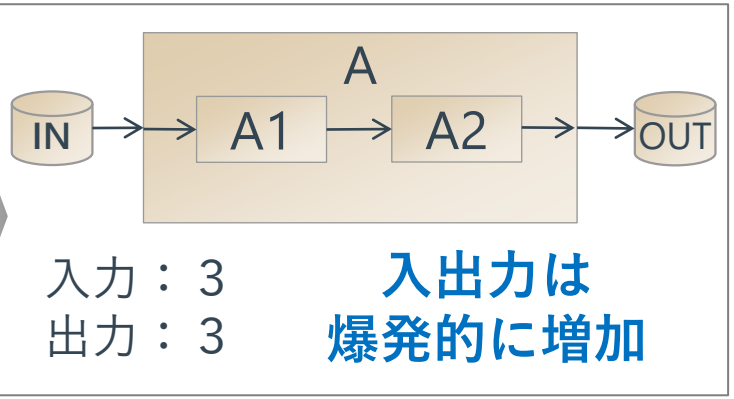
コンポーネント

ブレイクダウン

No.	名前	責務
1.	車速偏差	ドライバの要求を元に目標車速を算出。 実際の車速と目標車速の差分を算出する。
2.	車間距離偏差	距離設定と実車速を元に目標車間距離を算出。 実車間距離と目標車間距離の差分を出力する。
3.	目標制御量	車速の差分、車間距離の差分を元に、目標値に近づくための加速度、減速度を算出する。



入力：1
出力：1



入力：3
出力：3

入出力は爆発的に増加

コンポーネント詳細定義

コンポーネント 車間距離偏差

責務

距離設定と実車速を元に目標車間距離を算出。
実車間距離と目標車間距離の差分を出力する。

入力

No.	名前	Type	説明
1.	DISTANCEスイッチ操作	bool	車間距離設定SWの操作有無
2.	クルコン状態	State	現在のクルコンの状態(State)
3.	車速	double	現在の車速
4.	先行車有無	bool	先行車が存在する（感知している）場合、True。存在しない（感知していない）場合、False。
5.	実車間距離	double	センサが計測した車間距離

出力

No.	名前	Type	説明
1.	車間距離差分	double	実車間距離と目標車間距離の差分

内部データ

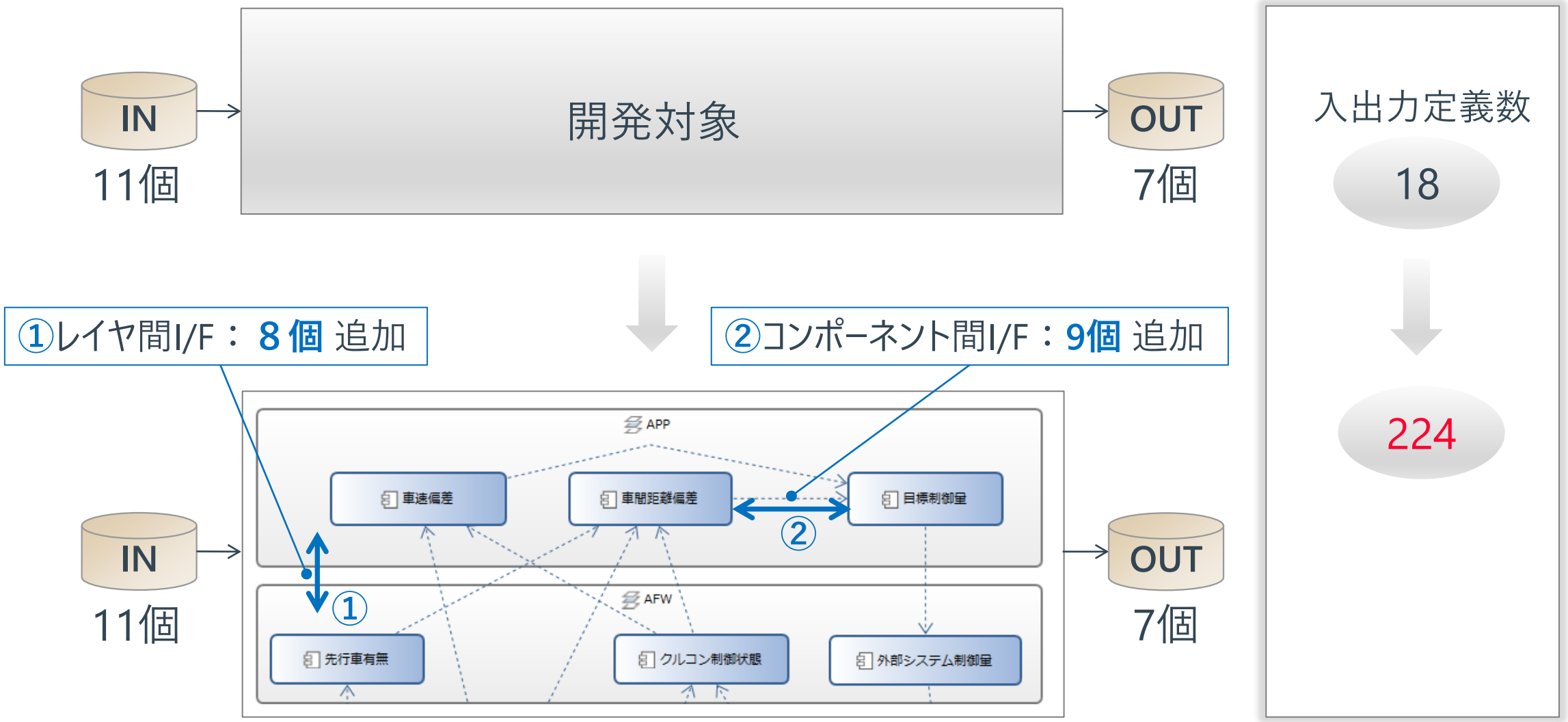
No.	名前	変数名	データ型	ダイナミクス	説明
1.	車間距離偏差	DistanceDifference	double	動的	目標とする車間距離と実際の車間距離の差分

サブコンポーネント

No.	名前	責務
1.	目標車間距離演算	目標車間距離を算出する
2.	車間距離差分演算	車間距離差分を計算する

人間の思考：**段階的に詳細化**（ブレイクダウン）により**理解**する。**同じ記述**を繰り返す

（参考）ブレイクダウンの副作用



たかだか17個のI/Fの追加 ⇒ **200個以上の定義**が必要

2.観察（2/4）：関心事（視点）によって、見た目を変える

1. レイヤ APP

責務

ユーザに対するサービスを実現する。
本システムでは、速度と車間距離の差分から、目標制御量（加減速量）の算出を行う。

コンポーネント

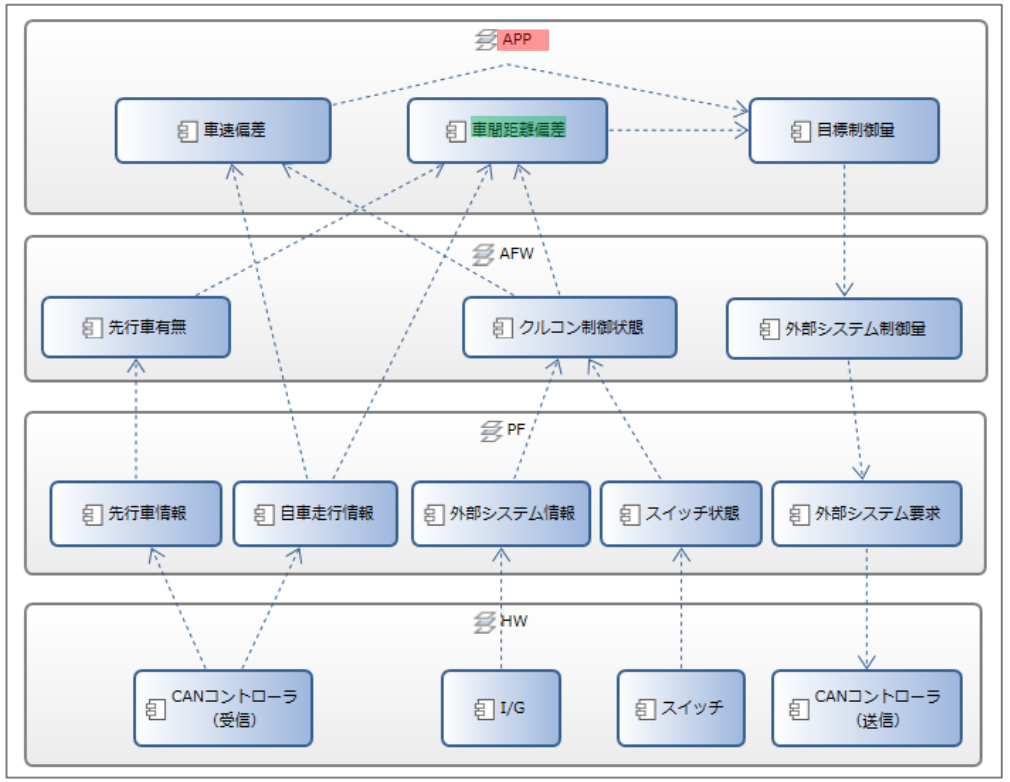
No.	名前	責務
1.	車速偏差	ドライバの要求を元に目標車速を算出。 実際の車速と目標車速の差分を算出する。
2.	車間距離偏差	距離設定と実車速を元に目標車間距離を算出。 実車間距離と目標車間距離の差分を出力する。
3.	目標制御量	車速の差分、車間距離の差分を元に、目標値に近づくための加速度、減速度を算出する。

個別に詳細定義

名前	
ADAS ECUソフト構造	
APP	ユーザに対するサービスを実現する。
車速偏差	ドライバの要求を元に目標車速を算出。
車間距離偏差	距離設定と実車速を元に目標車間距離を算出。
目標制御量	車速の差分、車間距離の差分を元に、目標値に近づくための加速度、減速度を算出…
AFW	APP層が共通で使用する情報をPF層の情報を元に合成する、APP層が算出した要求…
クルコン制御状態	ドライバの各種スイッチ操作と外部システムの状態からクルコンの制御状態を判定し、他…
先行車有無	車間距離や舵角から先行車の有無を判定する。
外部システム制御量	目標制御量をエンジン、ブレーキそれぞれに分配する責務を持つ。
PF	デバイスに依存した入出力の処理を行う。
スイッチ状態	各種スイッチの操作判定をまとめたコンポーネント。
自車走行情報	自車の状態（今回は車速のみ）を公開する。
外部システム情報	他のECUから情報取得/公開する責務を集約したコンポーネント。

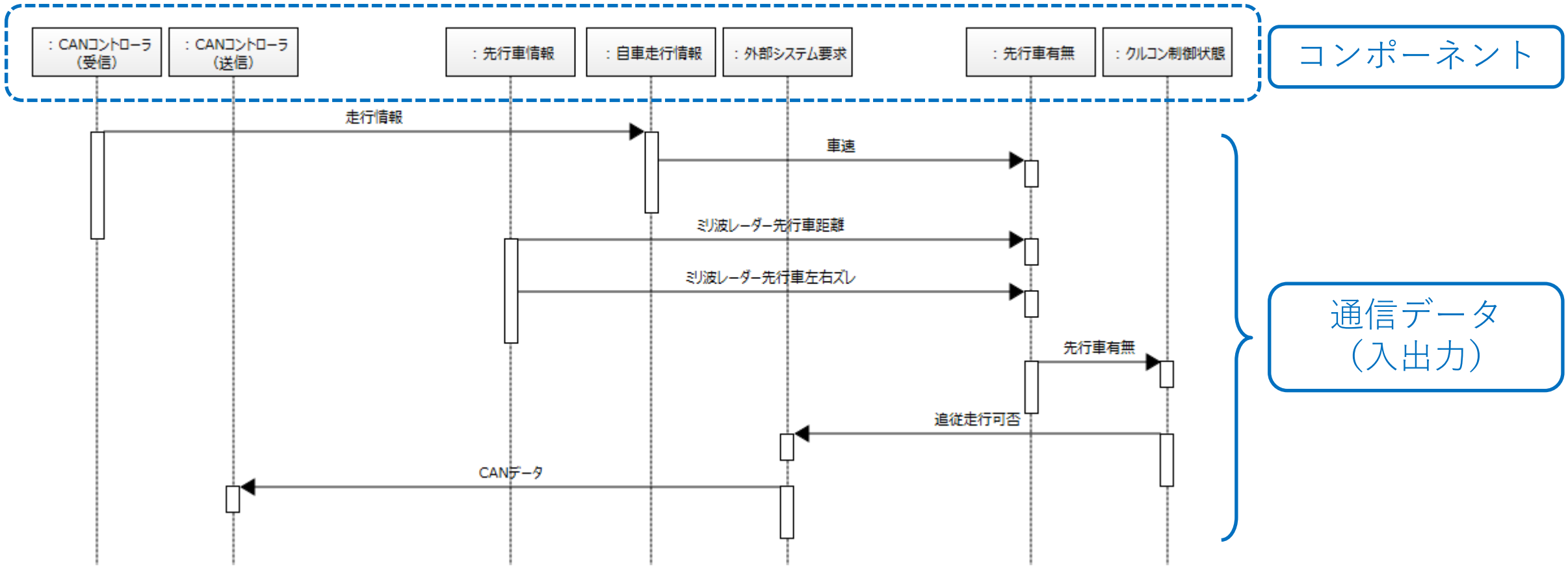
一覧で確認

構造図（静的関係）



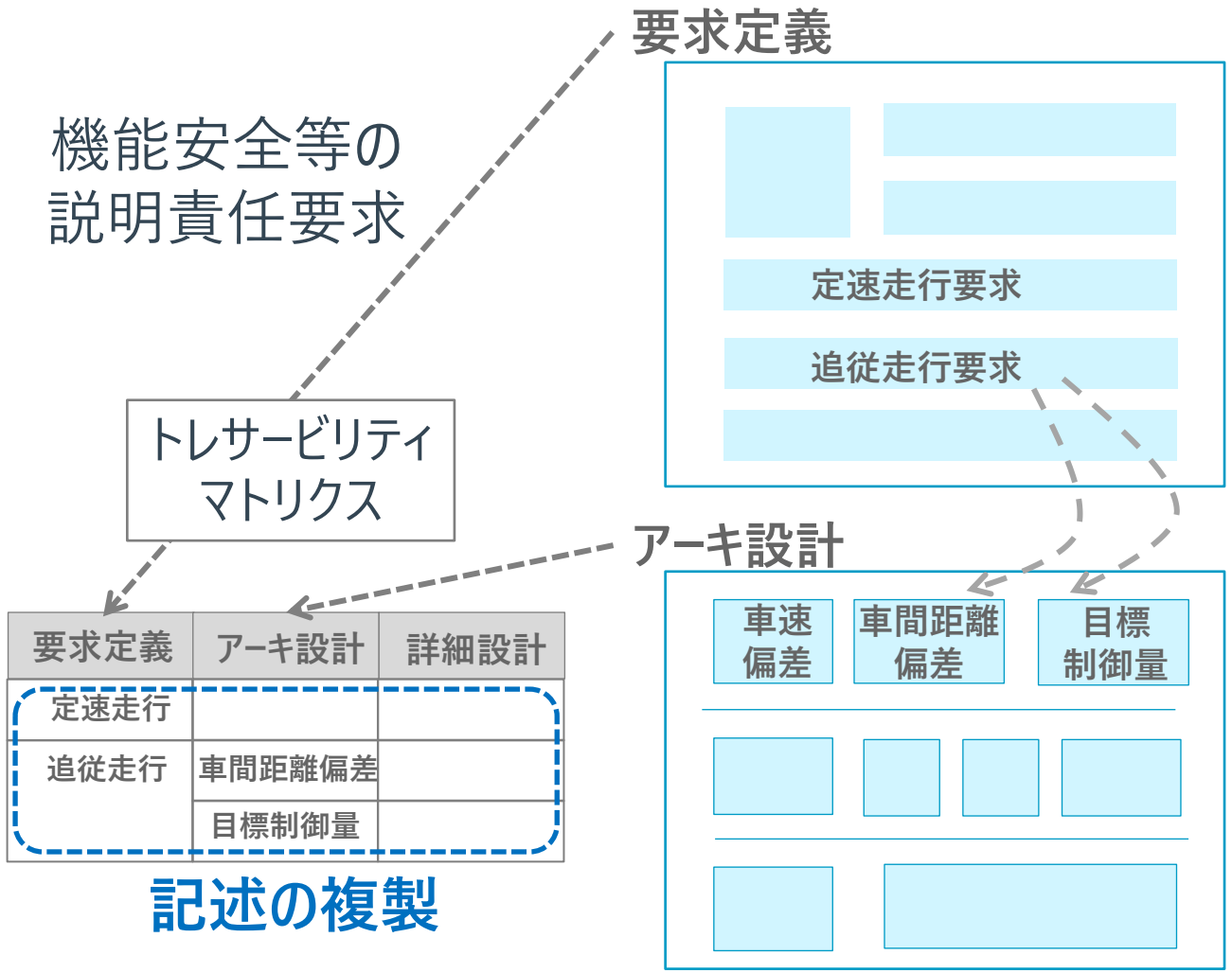
関心事によって異なる**見た目（ビュー）**が必要。同じ設計情報を**繰り返し**使用する

2.観察（3/4）：動的関係の表現が設計の理解を助ける



動的な関係を示すことで、動作イメージ（**振る舞い**）を理解する

2.観察（4/4）：後追いとなりやすいトレーサビリティ記録



この要求は、この２つの
コンポーネントで実現しよう

と、紐づけて設計している
だが、実際には記録に残りにくい（欠落）

評価日程
優先

納期
厳守

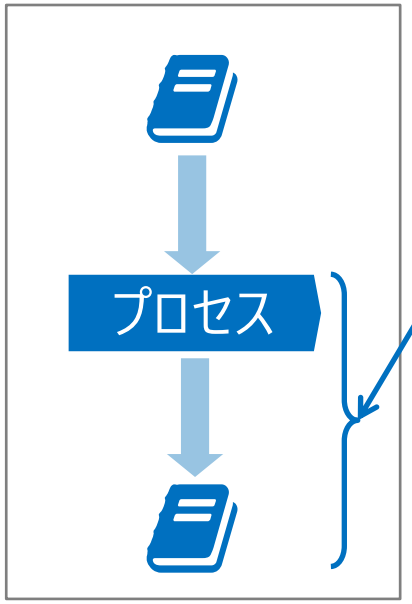
面倒
くさい

“**開発効率**“という理由で**後回し**
この時点では、“紐づけ”記録の価値が低い

通常の開発の中でやっていることだが**記録に残りにくい**。文書に残すと**記述が重複**となる

3.気づき

エンジニアリングプロセス



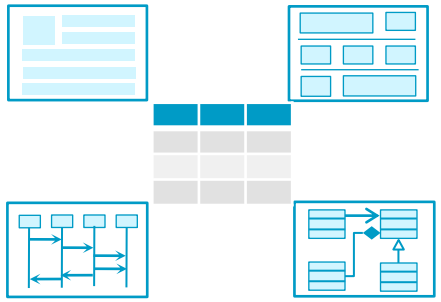
エンジニアリングを**出力文書**で規定する
様々な**関心事（視点）**に対し、**特定の見た目**で定義

何を

・段階的に詳細化、入出力
・個別、一覧、構成・構造
・振る舞い
・トレーサビリティ（導出関係）

複数の関係性

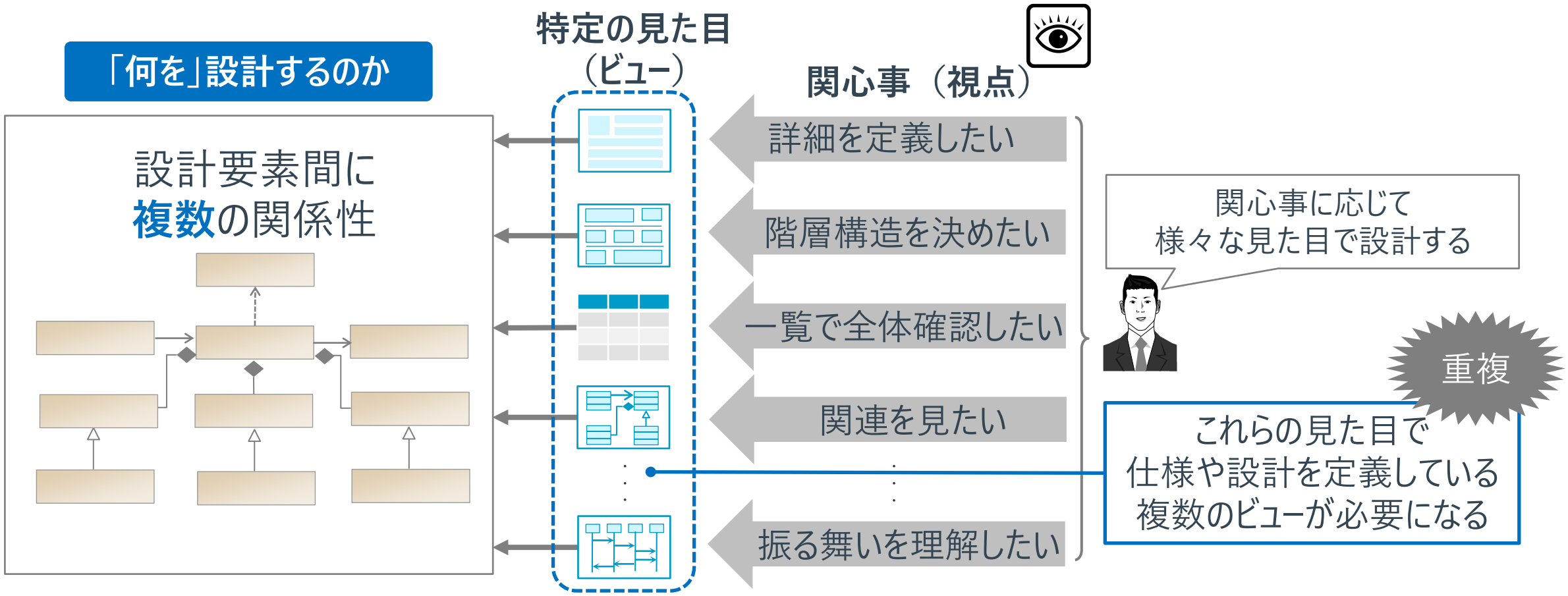
どうやって



人間が理解しやすい見た目

特定の見た目ごとに設計情報を記述 ⇒ **重複**する原因、**不整合や欠落**を引き起こす
“見た目を使って定義する” **既存の方法**：「**何を**」を「**どうやって**」で定義している

4.着想：見た目に影響を受けない方式



見た目の制約により、**定義が制限**される。見た目で表現できることしか定義できない
見た目に影響を受けない定義方法 ⇒ **セマンティック技術**

4 .着想：セマンティック技術

【セマンティック技術】

「**情報の意味**を**コンピュータが理解できる**形で表現し、コンピュータに処理を行わせる技術」

つまり、

人間が理解するために必要な**見た目を持たず**に、
設計情報をコンピュータが理解できる形で表現する技術

⇒ **複数の関係性**を扱える（意味的に表現する）
加えて、コンピュータが解釈できるので、**デジタル検証可能**

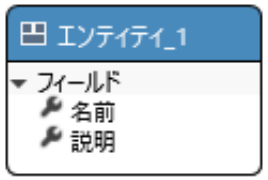
実現には、**メタモデル言語**（モデルを記述するためのモデル）を用いる

4 .着想：メタモデル

メタモデル言語

設計成果の情報構造体を表現するための言語
（MOFのM2層に相当）

設計要素：



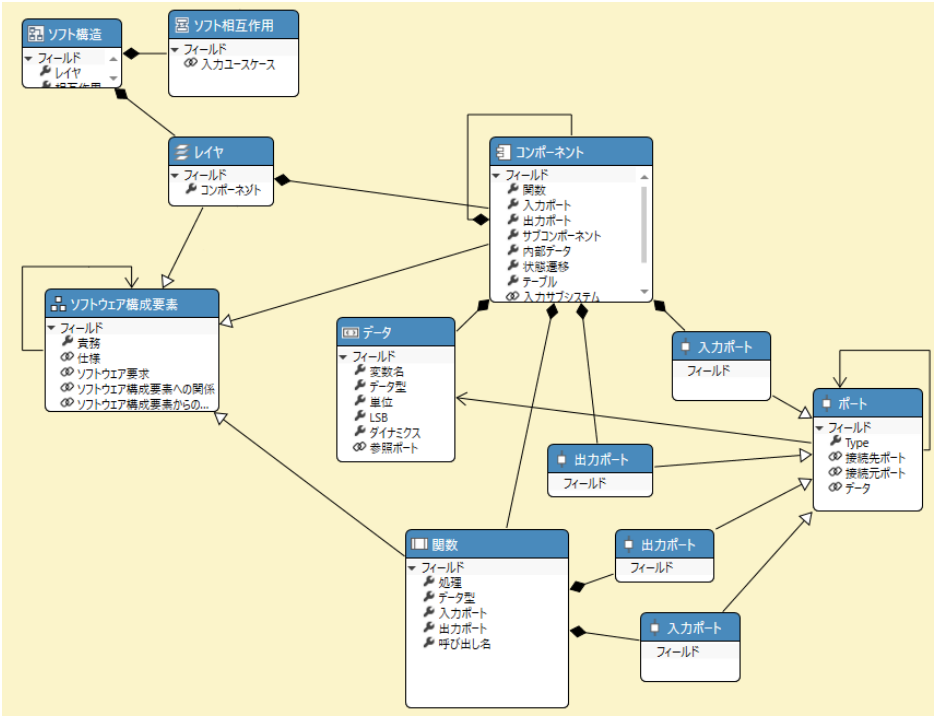
関係種類：

種類	説明
所有	親子関係や集約を表現
参照	利用や依存の関係を表現
導出	起源とのトレース関係を表現
継承	特性を継承することを表現

表現

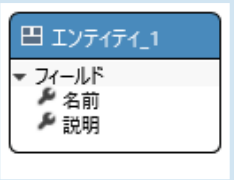
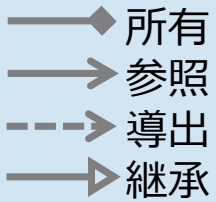
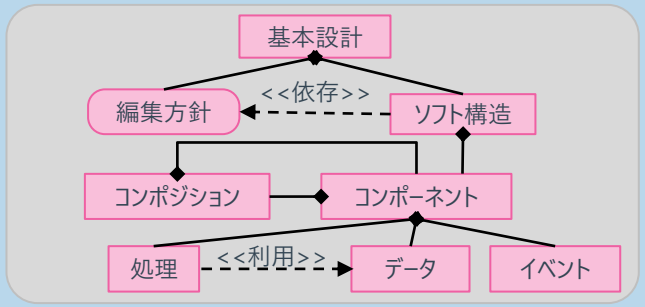
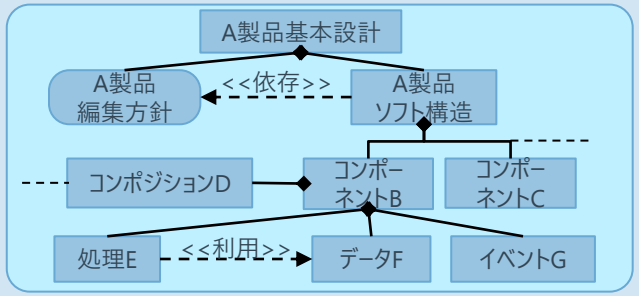
設計情報メタモデル

メタモデル言語を使い、ドメイン固有の
設計情報メタモデルを定義する
（MOFのM1層に相当）



設計要素間の**複数の関係性**をメタモデル言語を用いて定義する

（参考）モデル・メタモデルの階層概念図

階層	例	提案手法	内容
M3	MOF	なし	なし
M2	UML	メタモデル言語 || 設計情報メタモデルを記述するための言語体系	設計要素：  
M1	クラス図	設計情報メタモデル || 設計文書に記載すべき内容の構造を抽象化したもの	
M0	インスタンス	設計情報モデル || 設計文書に記載すべき内容を構造化したもの	

5 .提案方式：プロセス

従来のプロセス構築方式

(1)設計作業の洗い出し
(2)必要となるシーン毎に
出力形式を決める
関心事に応じた見た目を
使用する

重複

「何を」を「どうやって」で定義している

提案するのプロセス構築方式

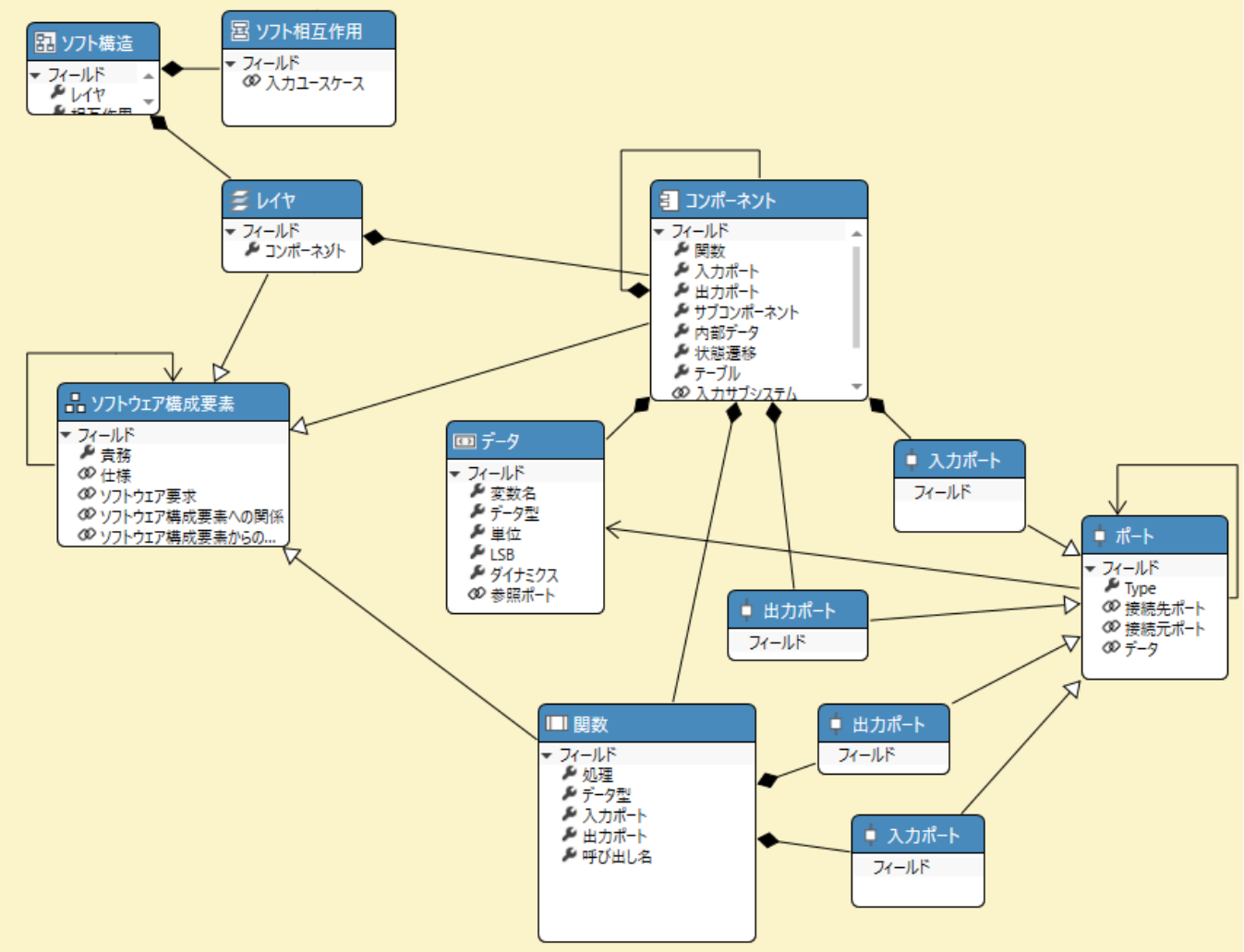
Step1：**何を設計するか**
①設計要素の洗い出しと整理
②設計要素間の関係の洗い出しと整理
③メタモデル言語による定義
（設計の情報構造体の定義）

Step2：**どうやって設計し、確認するか**
①必要となる見た目（ビュー）を決める
・個別、一覧、構成、動的関係・・・
人間が利用するために必要なビュー
②必要なビューを準備する
（IT活用：自社開発ツールを利用）

「何を」と「どうやって」を分離してプロセスを構築する

見た目を切り離し、設計すべき情報を定義する（**データとビューの分離**）
メタモデル言語をコンピュータが解釈し、必要なビューを実現

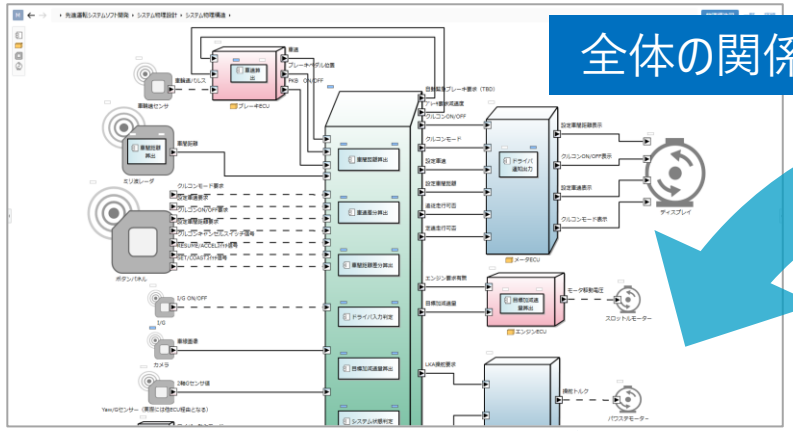
5 .提案方式：Step1（メタモデルによる設計情報の定義）



何を設計するか：設計要素と要素間の関連を整理し、メタモデル言語で定義する

5 .提案方式：Step2（見た目を決める）

①システム全体を俯瞰し、責務を把握する

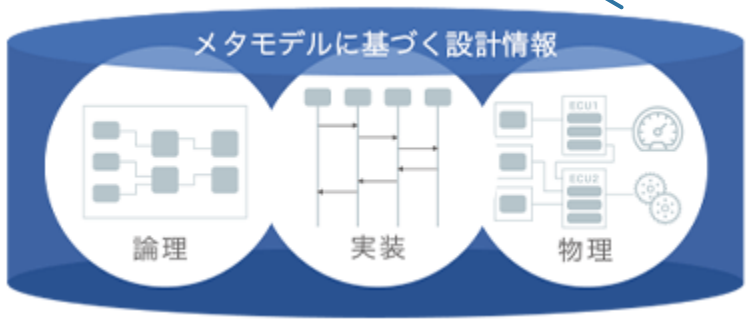


全体の関係性のビュー

切り替え

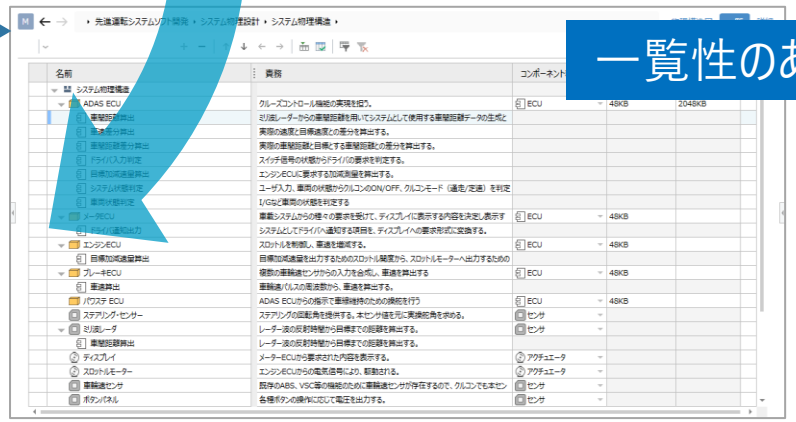
②エンジニアが設計内容を詳細に定義する

個別の設計のビュー



Step1：何を設計するか（設計の情報構造体）

③リーダーが網羅性や一貫性を確認する



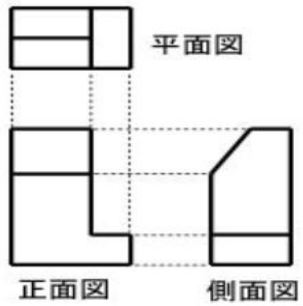
一覧性のある表の

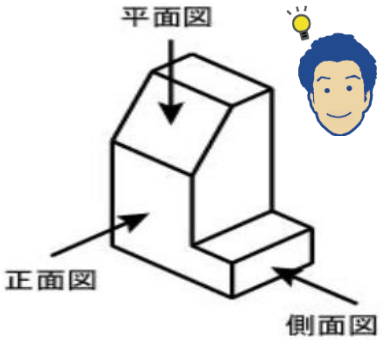
現実的な解法：既存の標準文書で使われている見た目をベースに、不十分な点を補う

5.提案方式：設計情報のデジタル化、一元化

設計情報のデジタル化（3D-CADとの対比）

3面図

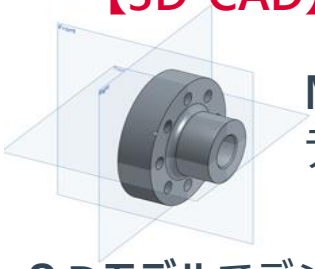




頭の中で
立体物を想像

デジタル化

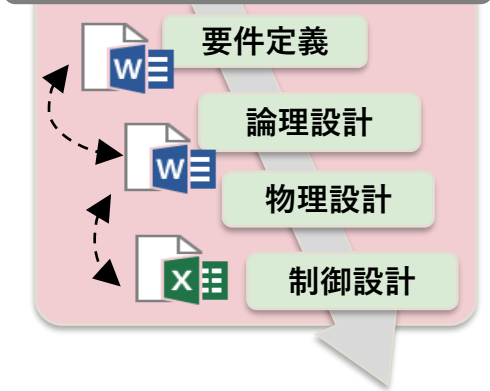
【3D-CAD】



Multi View
デジタル検証

3Dモデルでデジタル化

デジタル・ペーパー

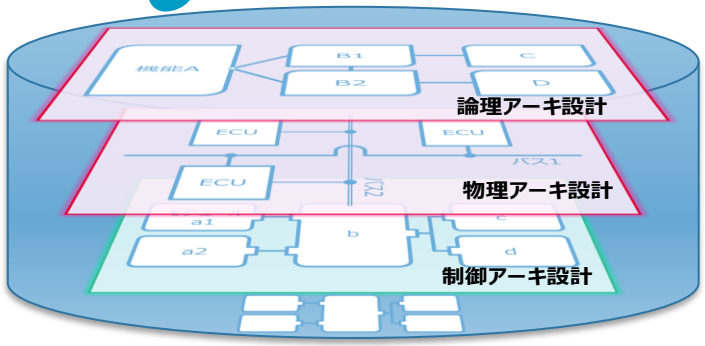


頭の中で考えるに
は限界あり

デジタル化

特定の見た目で定義している
情報は**バラバラ**、**不整合**？

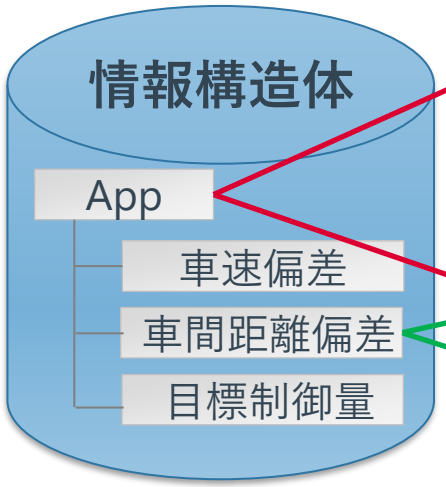
デザインDXツール
NEXT DESIGN



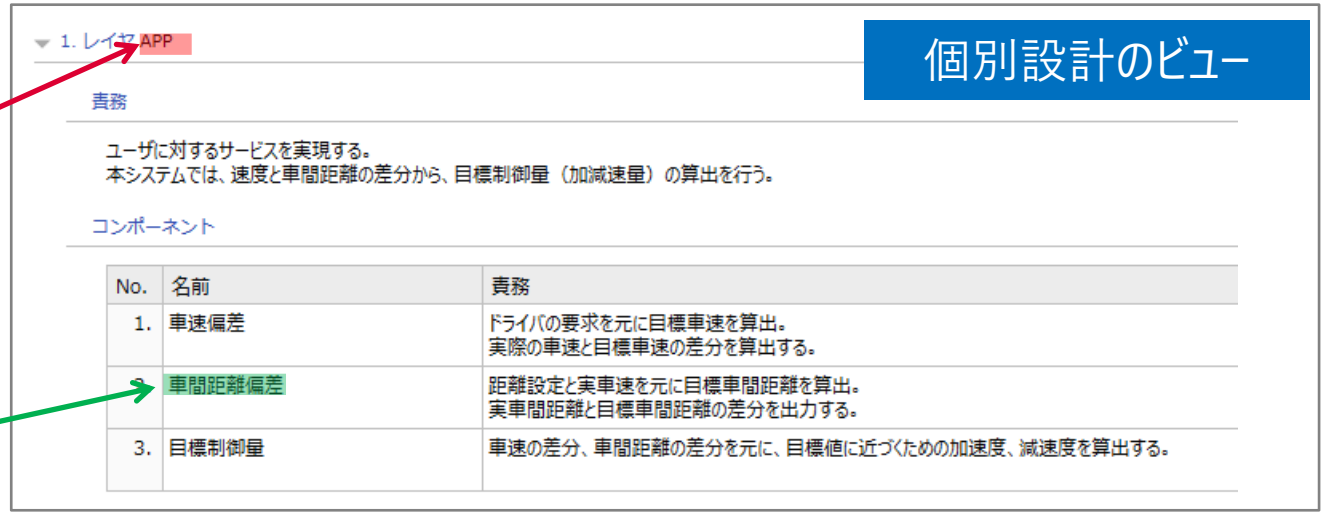
情報構造体としてデジタル化

人の力でつなげていた設計情報 ⇒ コンピュータが解釈、つなぎ合わせる（デジタル化）
多様な関心事をデジタル検証する（Multi View）

6.期待効果（1）直接的効果：①重複データの削減



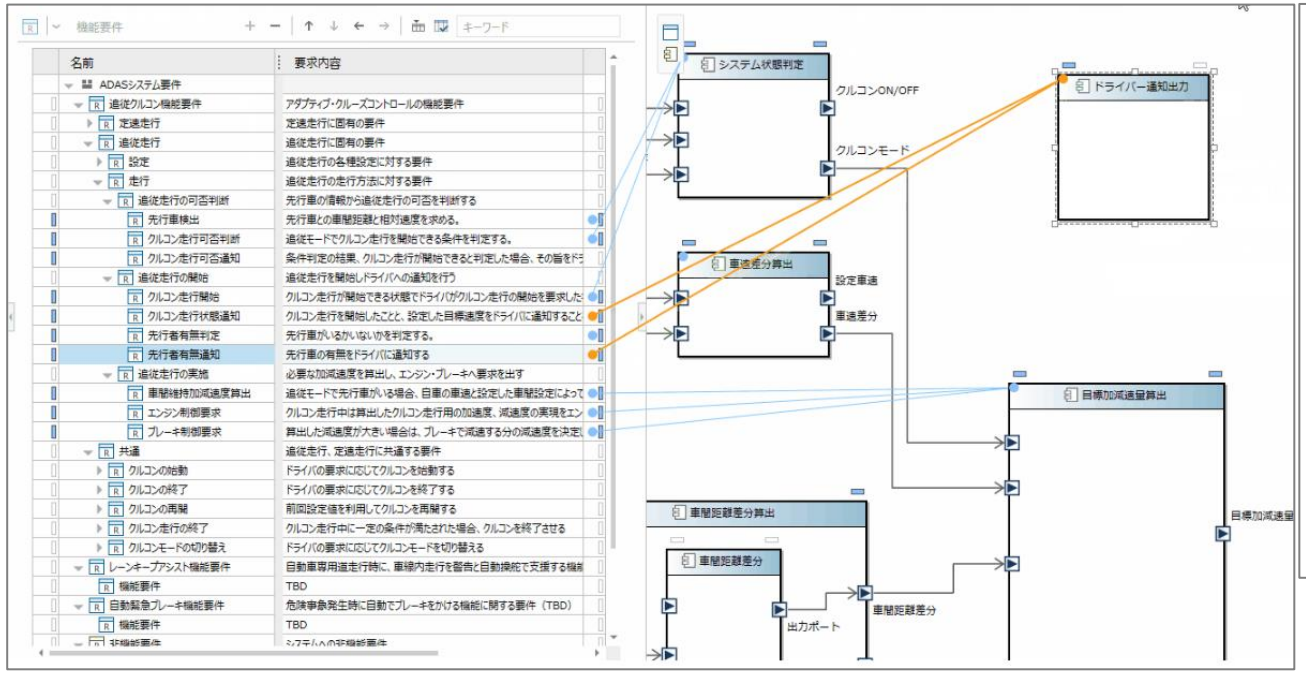
設計情報の**一元化**



それぞれのビューで利用、重複する
⇒ 一元化により重複データが削減

6.期待効果（2）間接的効果：①トレーサビリティ記録

要求　　機能構造図



要求　　機能一覧図

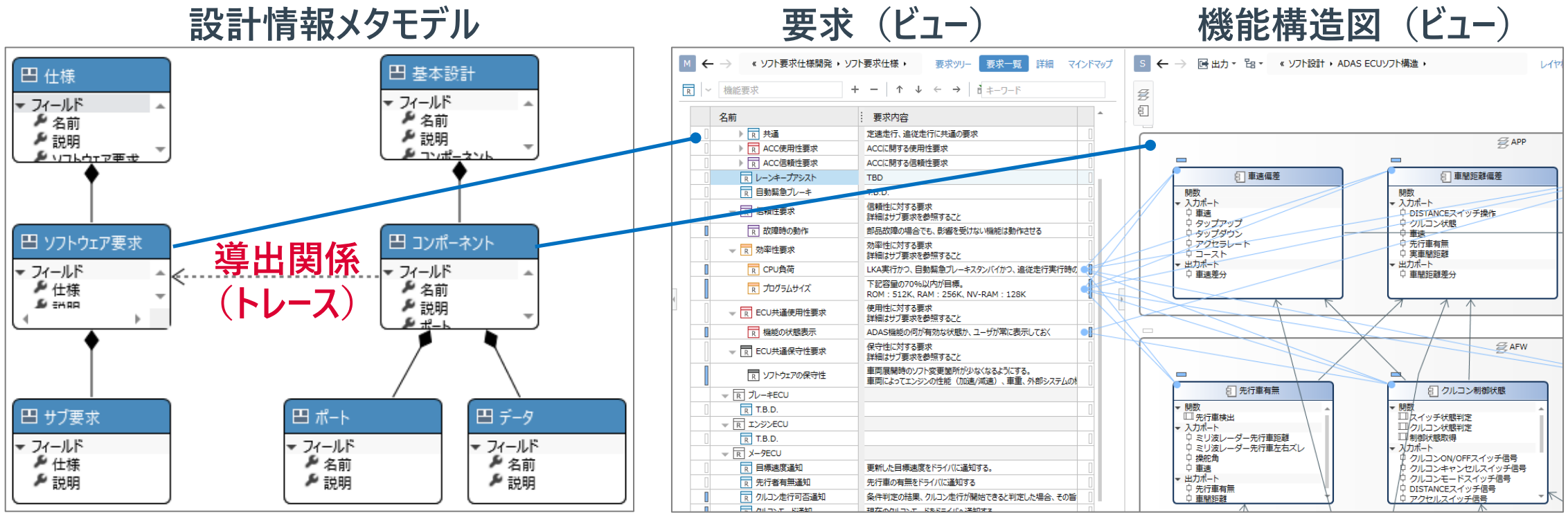
一元化 ⇒ 機能一覧図にも反映

要求をどこで実現するかを設計しながら、
トレーサビリティを記録していく

この要求とこの要求は、
この機能ブロックで実現しよう

が、**そのまま**記録に残る（後追い、欠落の歯止め）

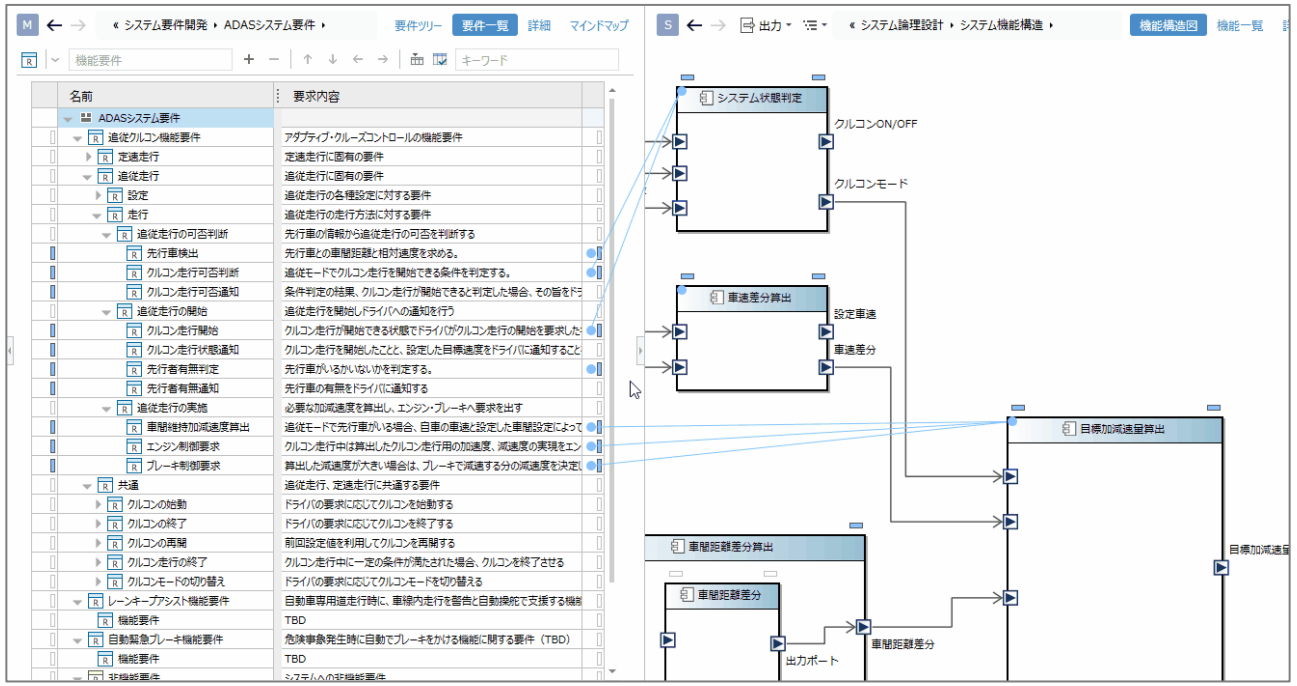
（参考）メタモデルで導出関係を定義



6.期待効果（2）間接的効果：①トレーサビリティ記録

詳細

要求　　機能構造図



要求　　機能一覧図

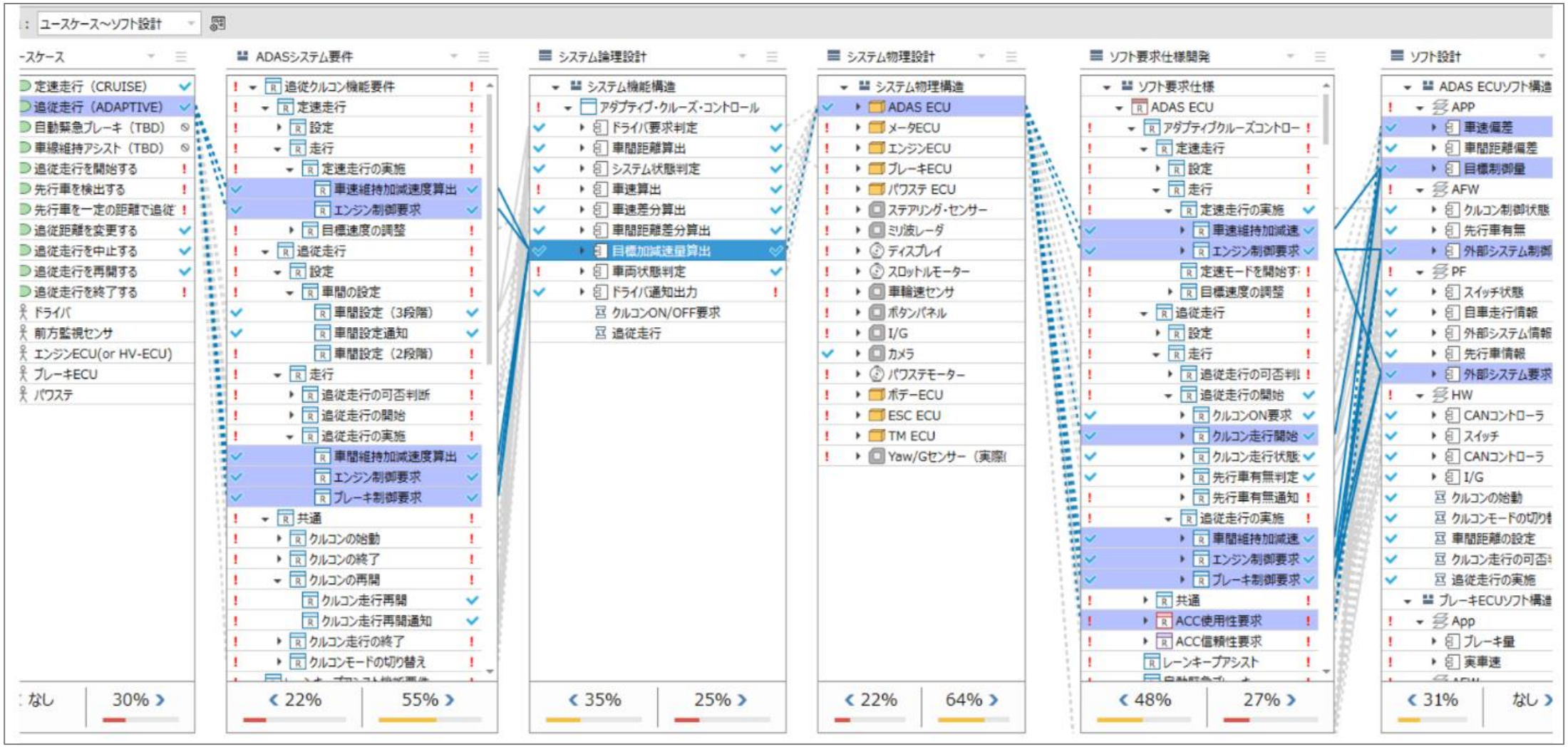
一元化 ⇒ 機能一覧図にも反映

要求をどこで実現するかを設計しながら、
トレーサビリティを記録していく

この要求とこの要求は、
この機能ブロックで実現しよう

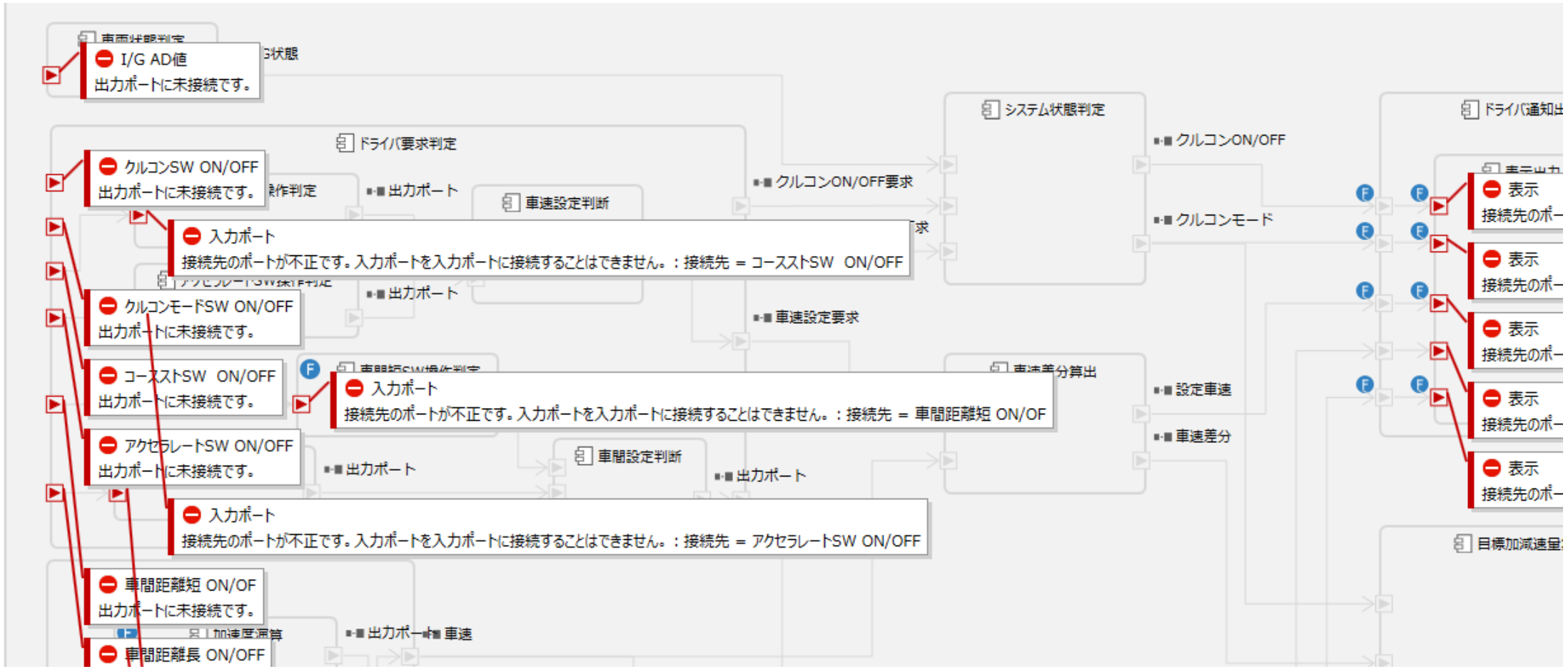
が、**そのまま**記録に残る（後追い、欠落の歯止め）

（参考）トレーサビリティ記録



レーン表示でトレーサビリティの全体把握や影響範囲の確認が可能 ⇒ **利用**できる

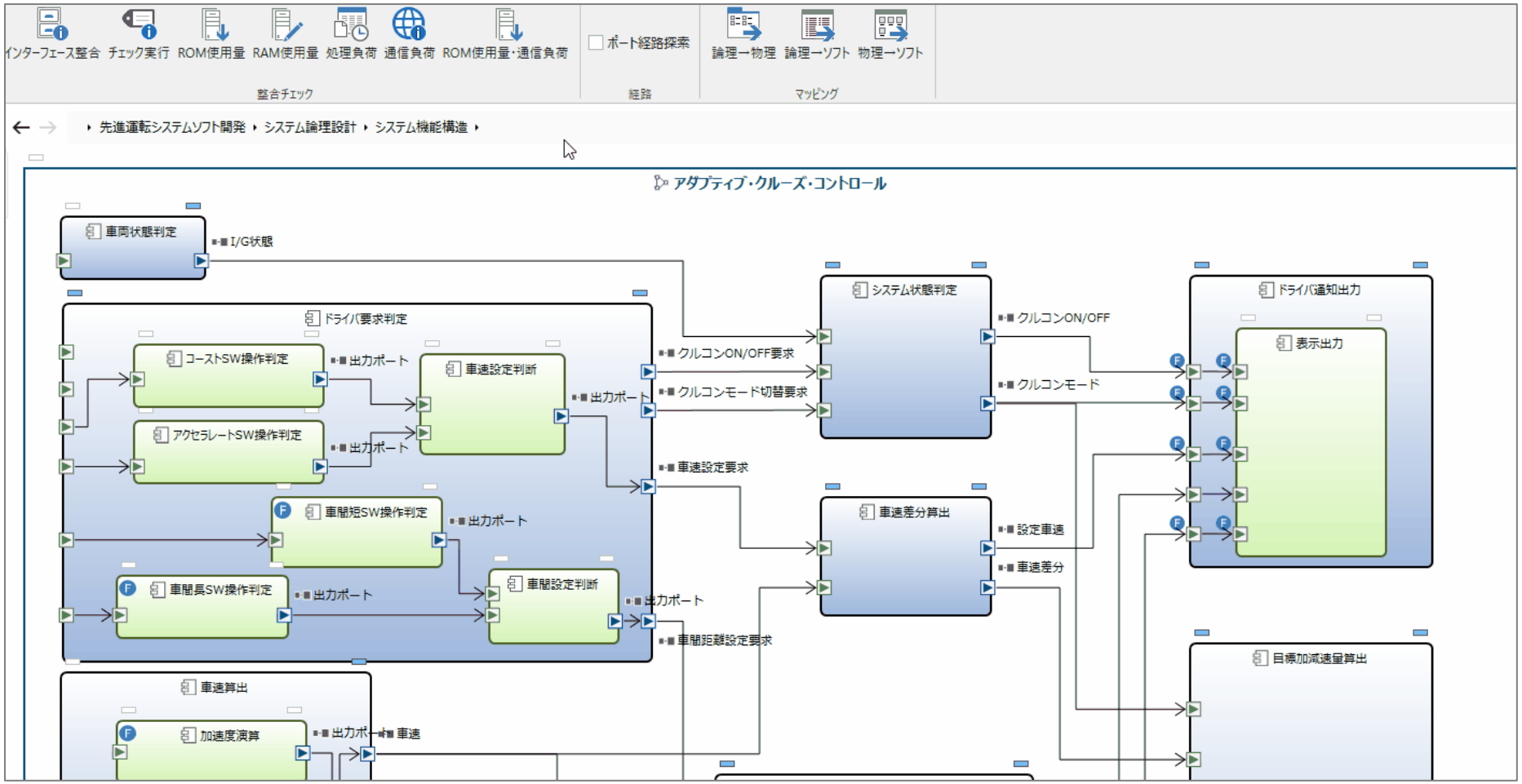
6.期待効果（２）間接的効果：②デジタル検証



コンピュータが解釈できる形式で設計情報を蓄積（**セマンティック技術**）
不整合、未定義、ルール違反など**デジタル検証**が可能となる

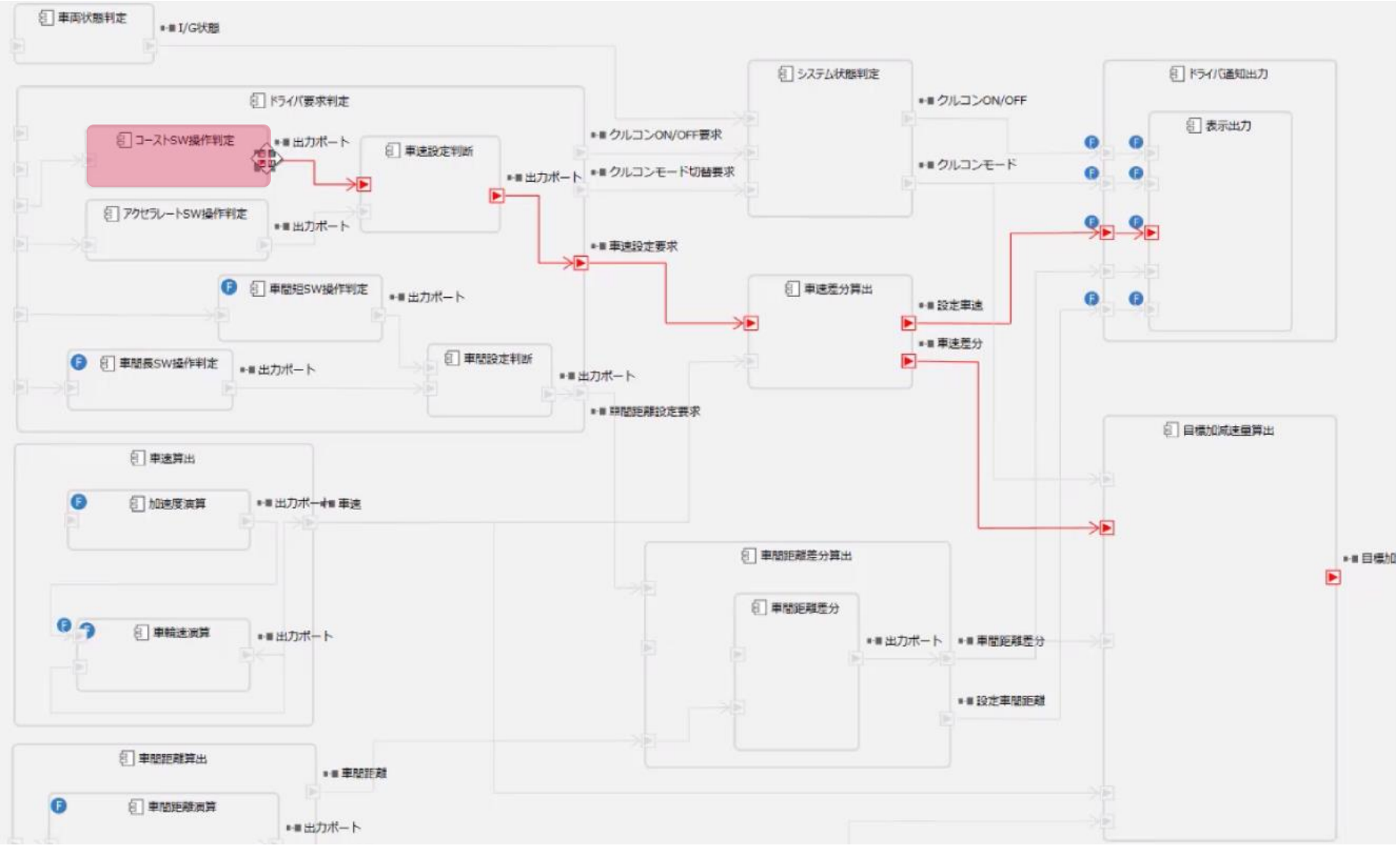
6.期待効果（2）間接的効果：②デジタル検証

詳細



コンピュータが解釈できる形式で設計情報を蓄積（**セマンティック技術**）
不整合、未定義、ルール違反など**デジタル検証**が可能となる

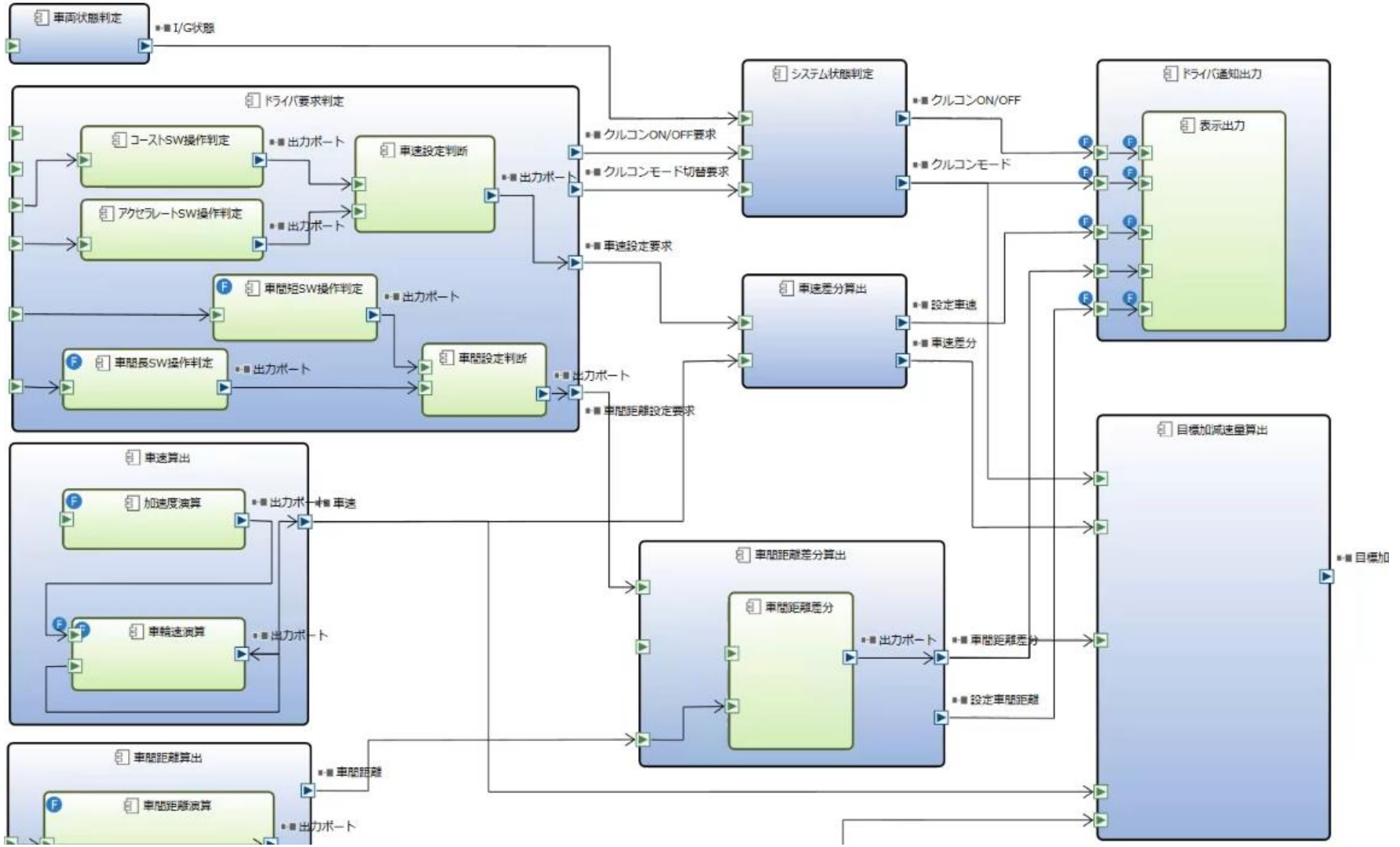
6.期待効果（2）間接的効果：③影響範囲の確認



“どこからどこへ”つながっているか、コンピュータが調べることが可能

6.期待効果（2）間接的効果：③影響範囲の確認

詳細



“どこからどこへ”つながっているか、コンピュータが調べることが可能

7 .効果確認

【評価項目と評価方法】

(1)直接的効果：

①重複データの削減度合い

【方法】文書の重複データの削減状況を観察する

【対象】アーキ設計～コンポーネント設計（全87頁）

(2)間接的効果：

②トレーサビリティ記録の効率化

【方法】後追いでタグを埋め込む方式との比較実験

【対象】テスト仕様設計（要求仕様とテスト仕様とのトレーサビリティ記録）

③不整合の削減（不整合発生状況、不整合解消状況）

【方法】前プロジェクトとの比較

【対象】大規模組込みシステム開発

7 .効果確認：①重複データの削減度合い

①重複データの削減度合い

【方法】文書の重複データの削減状況を観察する
【対象】アーキ設計～コンポーネント設計（全87頁）

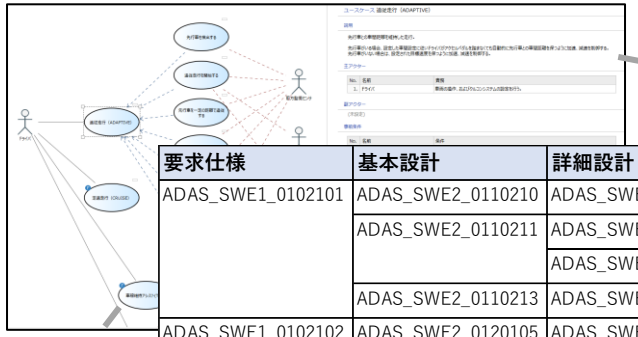
状態遷移図、シーケンス図など振る舞い関連は計測から除外した

設計要素種別	要素数	出現数	
		従来手法	提案手法
レイヤ	4	16	4
コンポーネント（含サブコンポーネント）	20	112	20
関数	23	120	23

狙い通り、重複データの削減が削減されていることが確認できた

7 .効果確認：②トレーサビリティ記録の効率化

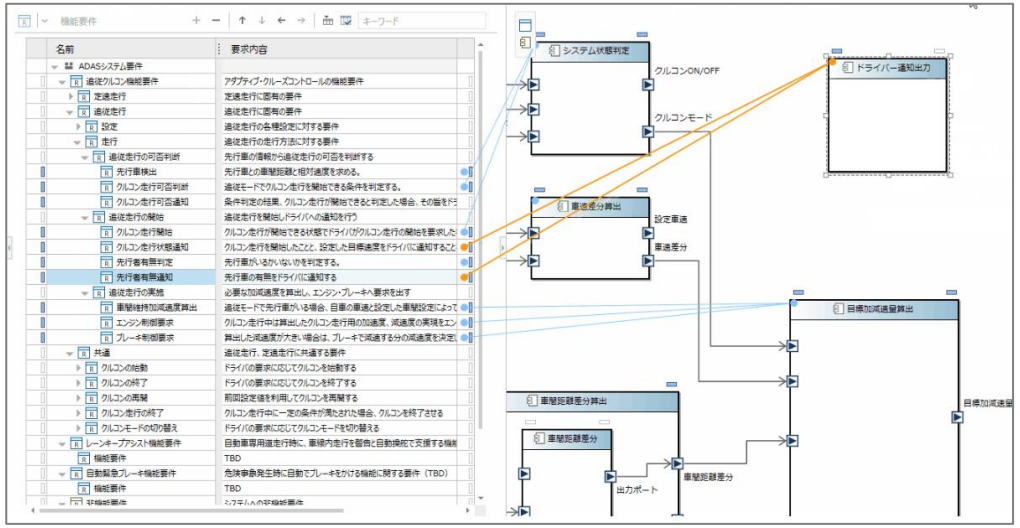
【従来手法：タグ埋め込み方式】



要求仕様	基本設計	詳細設計
ADAS_SWE1_0102101	ADAS_SWE2_0110210	ADAS_SWE3_0205001
	ADAS_SWE2_0110211	ADAS_SWE3_0205002
		ADAS_SWE3_0205003
	ADAS_SWE2_0110213	ADAS_SWE3_0205004
ADAS_SWE1_0102102	ADAS_SWE2_0120105	ADAS_SWE3_0205010
	ADAS_SWE2_0120106	ADAS_SWE3_0205011
		ADAS_SWE3_0205012
ADAS_SWE1_0102103	ADAS_SWE2_0180650	ADAS_SWE3_0205013
	ADAS_SWE2_0180651	ADAS_SWE3_0205014
	ADAS_SWE2_0180652	ADAS_SWE3_0205015

トレサービリティ
マトリクス

【提案手法：セマンティック技術利用】



トレーサビリティ記録方式	テストケース数(項目)	テスト仕様工数(H)	トレース記録工数(H)	不具合件数(件)	テスト生産性(項目/H)	不具合密度(件/項目)
従来手法	472	84.5	4.5	7	5.3	0.015
提案手法	345	45.5	0.0	0	7.6	0.000

トレース情報の記録コストは**十分に小さい**（工数0H）
従来方式と比較して、テスト生産性が**43％向上**し、**不具合密度も低減**した

7 .効果確認：③不整合の削減

当日のみ

大規模組込みシステム開発に適用、大きな効果が確認され製品化できた

8.まとめ

【背景・課題】
・エンジニアリングプロセスは、**既存の成果物や標準テンプレート**を利用して定義している
・文書間や文書内で**不整合や欠落**が発生している

【気づき・着想】
・設計には、様々な**関心事（視点）**があり、関心事に応じて**特定の見た目（ビュー）**を用いる
　⇒ 設計情報が重複する原因であり、不整合や欠落を引き起こす
・見た目に影響を受けない定義方法として、**セマンティック技術**を用いる（メタモデル言語の適用）

【提案方式】
・**「何を」と「どうやって」を分離**してプロセスを構築する
　見た目を切り離し、設計すべき情報を定義する（**データとビューの分離**）

【評価・効果確認】
・本方式により重複データを削減
・トレーサビリティ記録の効率化を実現
・大規模組込みシステム開発プロジェクトに導入、不整合問題に対し大きな効果を確認

「何を」設計するのかに立ち戻った**本質的なアプローチ**！ “**Identity**”から生まれた構築手法

参考文献

- 山路厚．“五月雨式な”大規模組込みソフト開発におけるメトリクス活用事例　－トレーニング指向アプローチによる“考え使う”業務スタイルへの挑戦．ソフトウェアプロセス改善カンファレンス2008 4C3 発表資料．2008
- 堀内一，大林正晴，藤川泰之．「メタモデル標準化の意義と最新動向，前編：-基本的概念と歴史的経過-，後編：-MOF，XMI 仕様と応用-」．情報処理 Vol. 43 No.11,No.12．（社）情報処理学会．2002
- 西村隆，山路厚，伊藤善博，原健三．メタモデルによる設計情報定義とマルチビューを活用したトレーサビリティ記録方式の提案．SQiPシンポジム2021 A2-1 発表資料．2021
- 山路厚，栗山順次，西村隆．～CASE時代に求められるデザインDXアプローチ～ デジタライゼーションによる大規模で高速な“すり合わせ”開発の実現．ソフトウェアプロセス改善カンファレンス2021 3-3 発表資料．2021
- 不破慎之介，小林展英．Next Design を用いたDX推進ツールの構築プロセスの提案．人工知能学会第二種研究会資料 2023巻KSN-033号．2023
- システム・ソフトウェア設計ツール Next Design: https://www.nextdesign.app/

DENSO

Crafting the Core