SysML- The Systems Modeling Language

SysMLモデリング言語

システムエンジニアリングの支援する言語　　概要

数学博士　春芽利　楼蘭

日本IBM基礎研究所、大和

メール balmelli@us.ibm.com

IBM Research Division

今日の概要

3つのシステムモデルの役割

要件モデリング

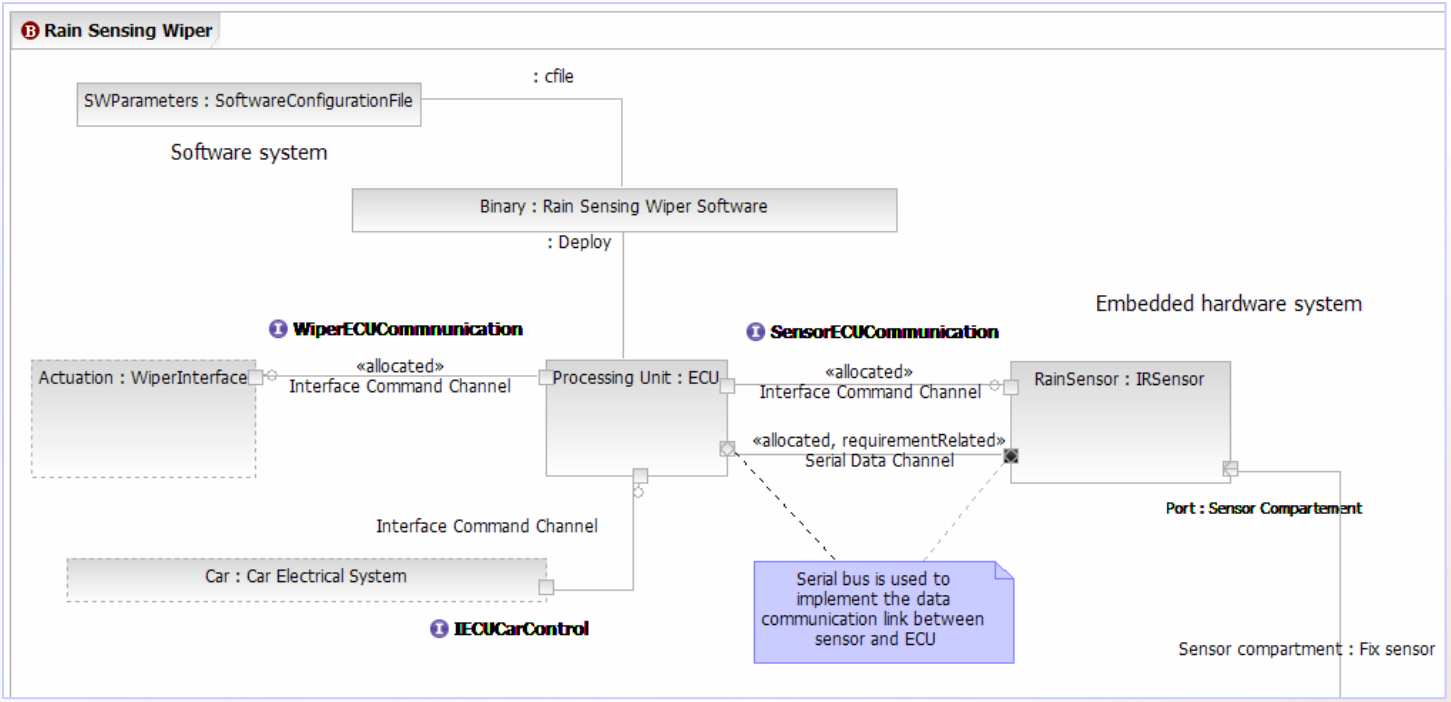
構成モデリング

動作モデリング

結論

システムモデルの第1の役割→コミュニケーション

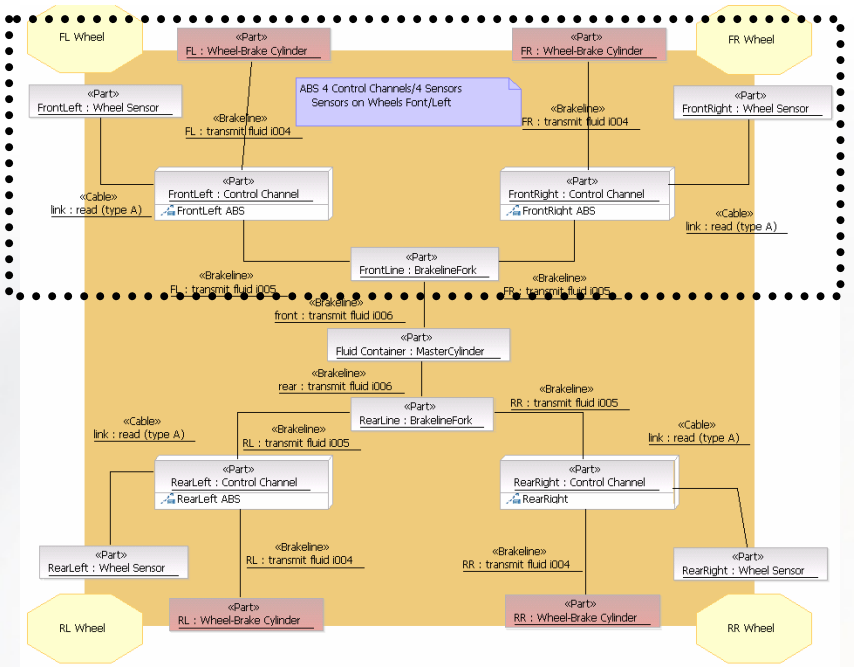
- システムモデル
  - 開発に携わる異なった分野のエンジニア間で、コミュニケーションするための役割
  - 製品の構成や機能や制約などを共通理解するための記述
  - 学際的に協力して開発する部品に関するリスクを共有するためのモデル



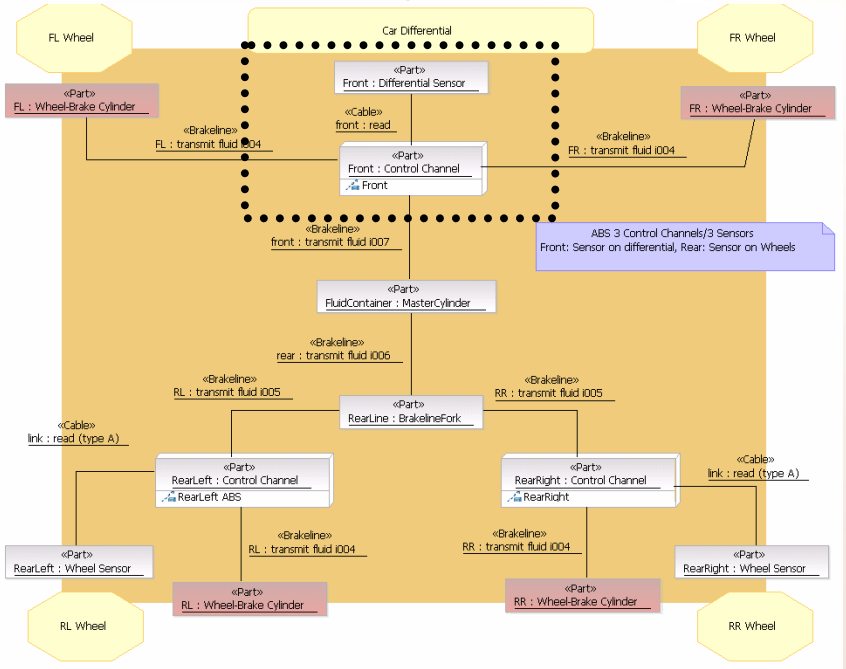
システムモデルの第2の役割→比較・評価

- 測定基準に基づいて製品のコンフィグレーションを比較する役割
- 初期設計しながら、できるだけ早く製品の性能を評価する役割

二つの感知器のモデル

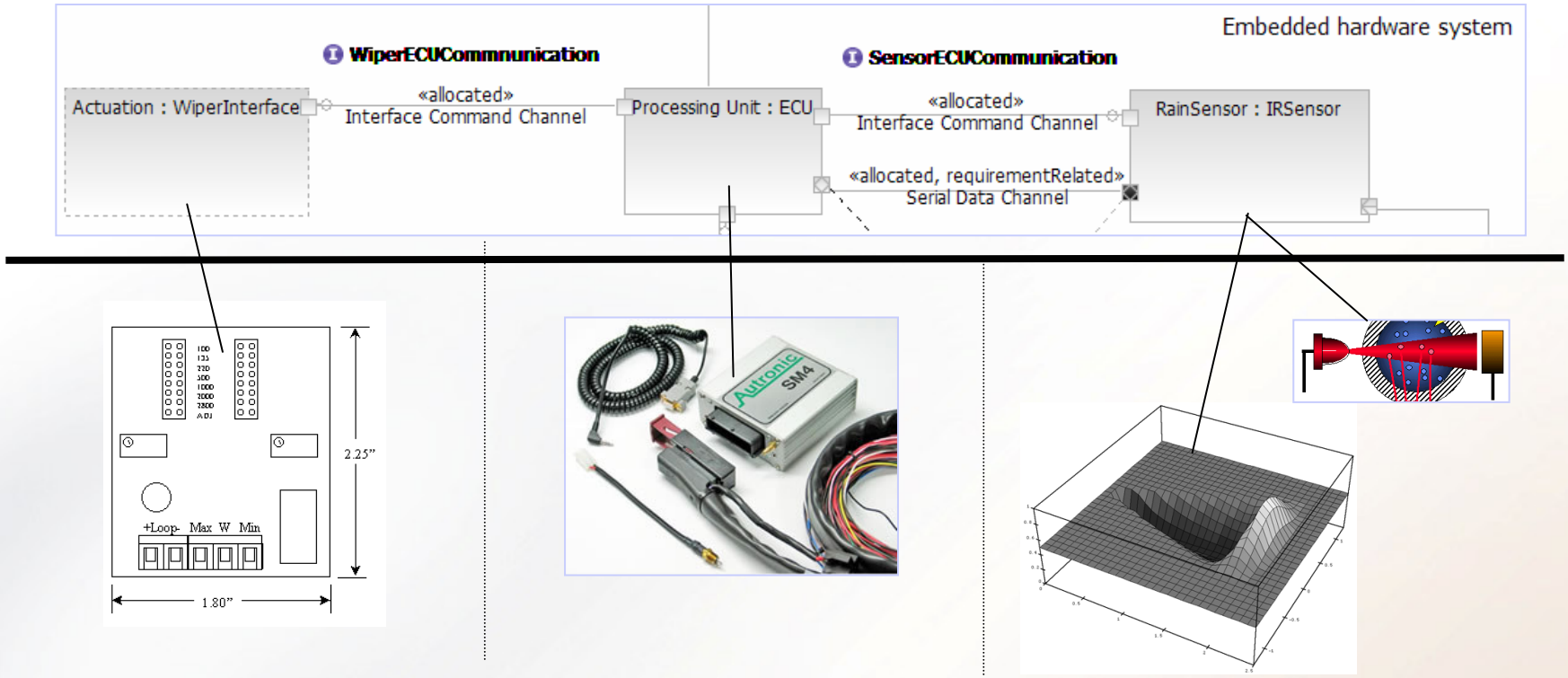


一つの感知器のモデル



システムモデルの第3の役割→管理・統合

- システムモデルは、「CADやソフトのためのUML」専門固有のモデルを取り替えるものではない
- システムモデルは、詳細設計のための成果物を抽象的に記述して管理する



今日の概要

3つのシステムモデルの役割

要件モデリング

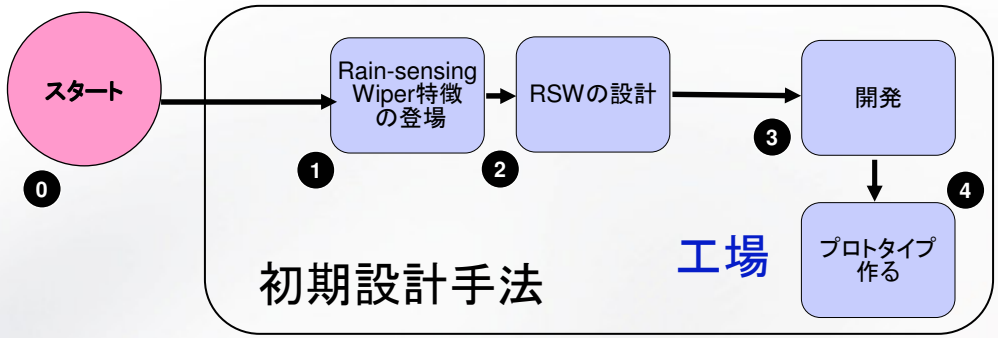
構成モデリング

動作モデリング

結論

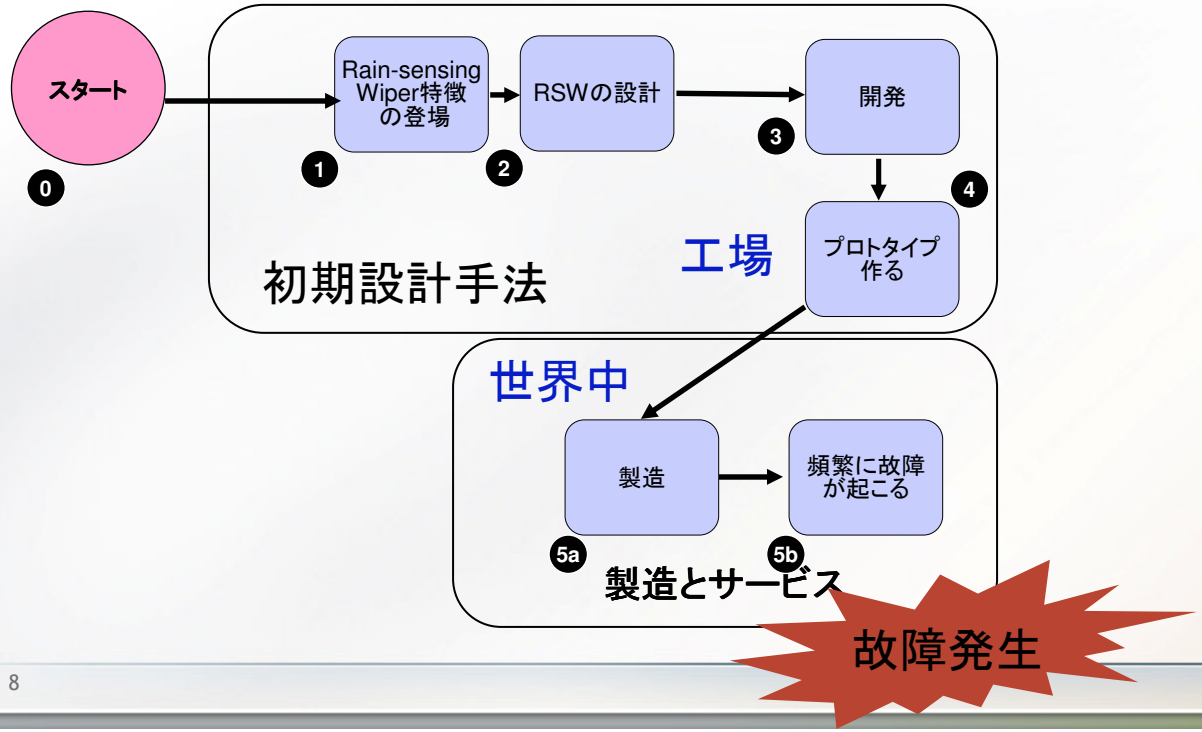
Rain Sensing Wiperの話

- 実際に自動車会社で起こった設計手法に関する不足
- 開発し終わったら、システムの要件の検証によって、正しくシステムが動いている。
- 平均故障間隔（MTBF）・故障が起こる平均の期間



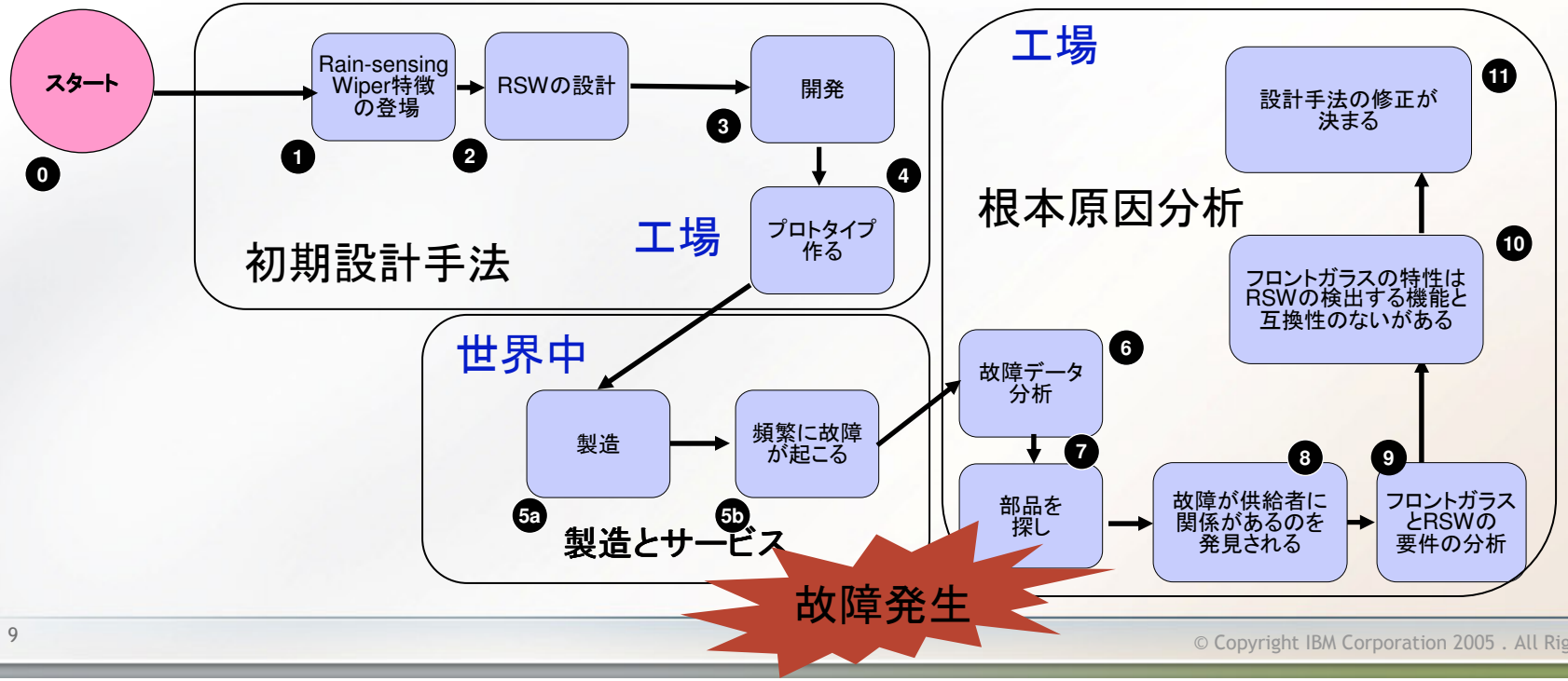
Rain Sensing Wiperの話

- 少しずつ、RSWに関する故障があるので、お客様は自動車を修理工場に運びます。
- 実際の故障する期間を、故障が起こる平均の期間に比べて、もっと短ければ、根本原因分析を行う。



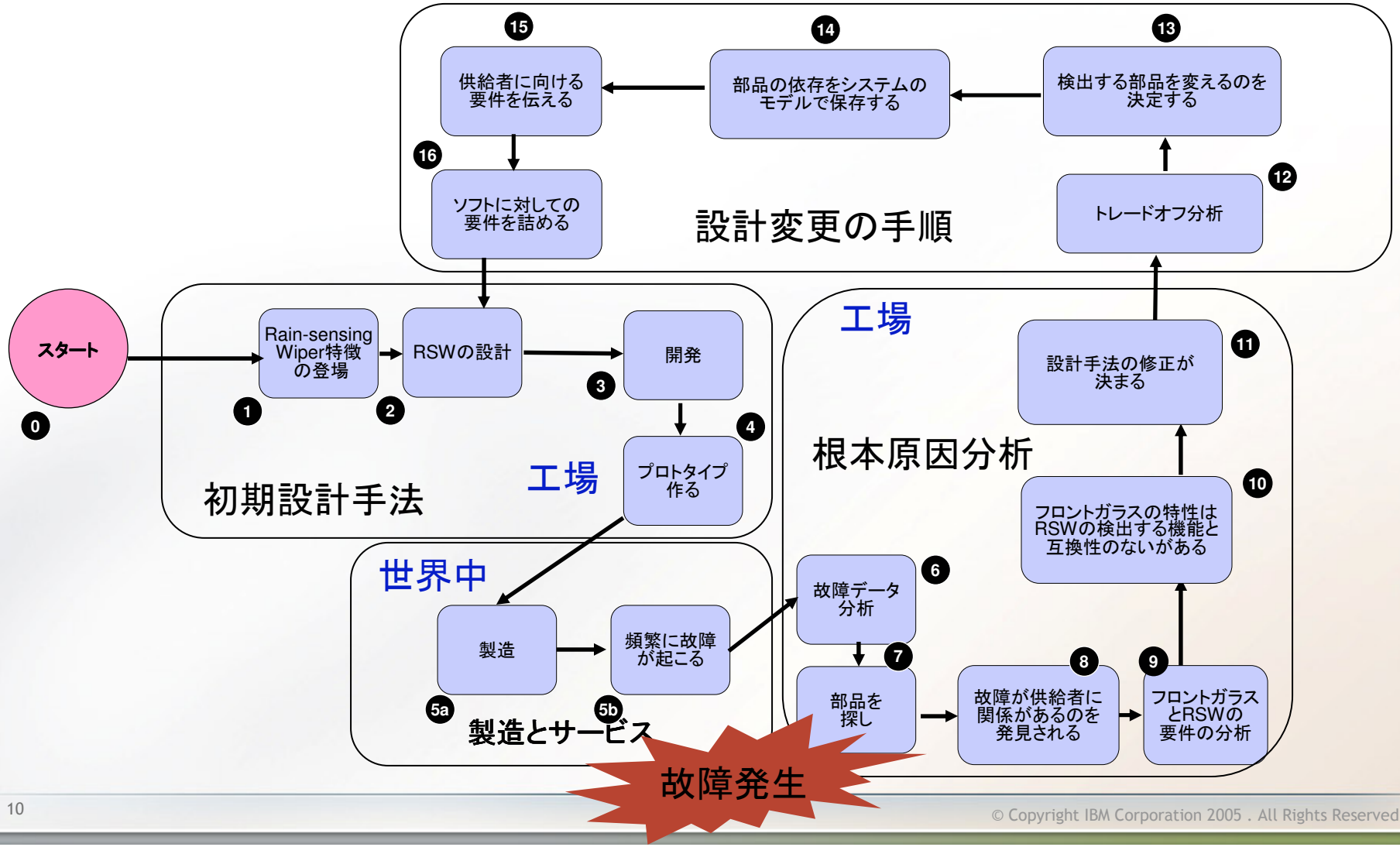
Rain Sensing Wiperの話

- それぞれの部品は正しく動いたけど、システムに対する要件が満たさないから、システムが動かなかい。
- 故障の理由→ローカルの供給者が異なった特性があるフロントガラスを配っているから、このフロントガラスはセの特性はンサーの特性と不整合です。



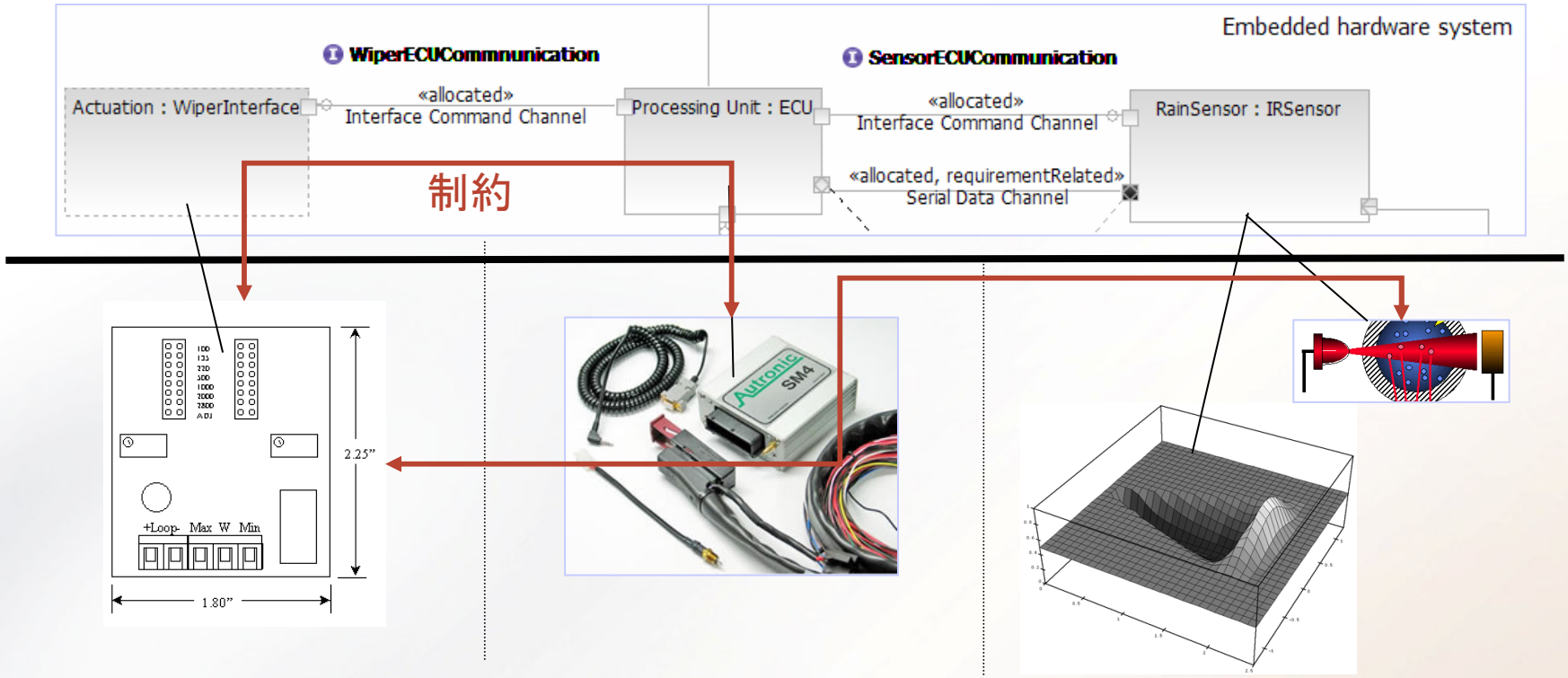
Rain Sensing Wiperの話

- 不整合の分品を、異なった分野で設計しているから、その部品の特性の間に、制約があったら、システムレベルで記述できるしかない。



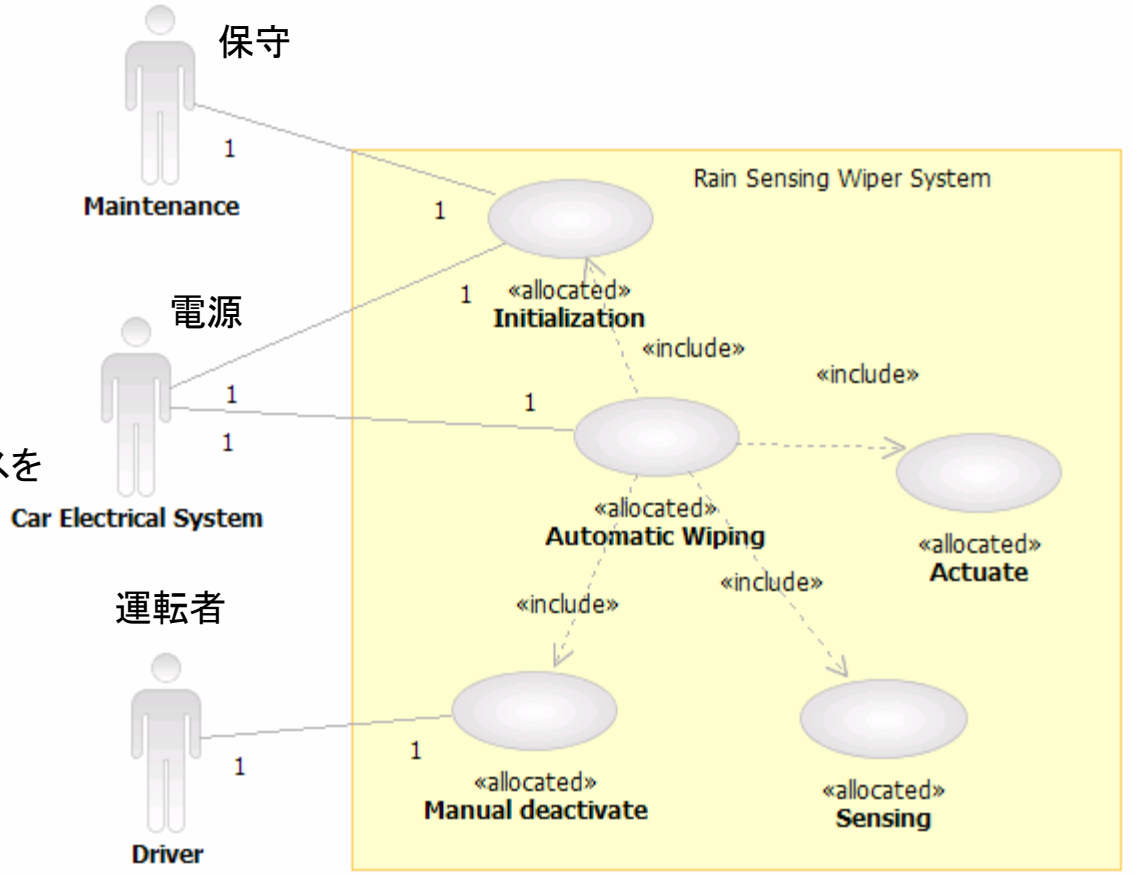
自動車会社の設計手法の不足の解決

- 異なった分野で設計した部品間に制約を述べる
- システムレベルで記述できるしかない→学際的な製品の理解



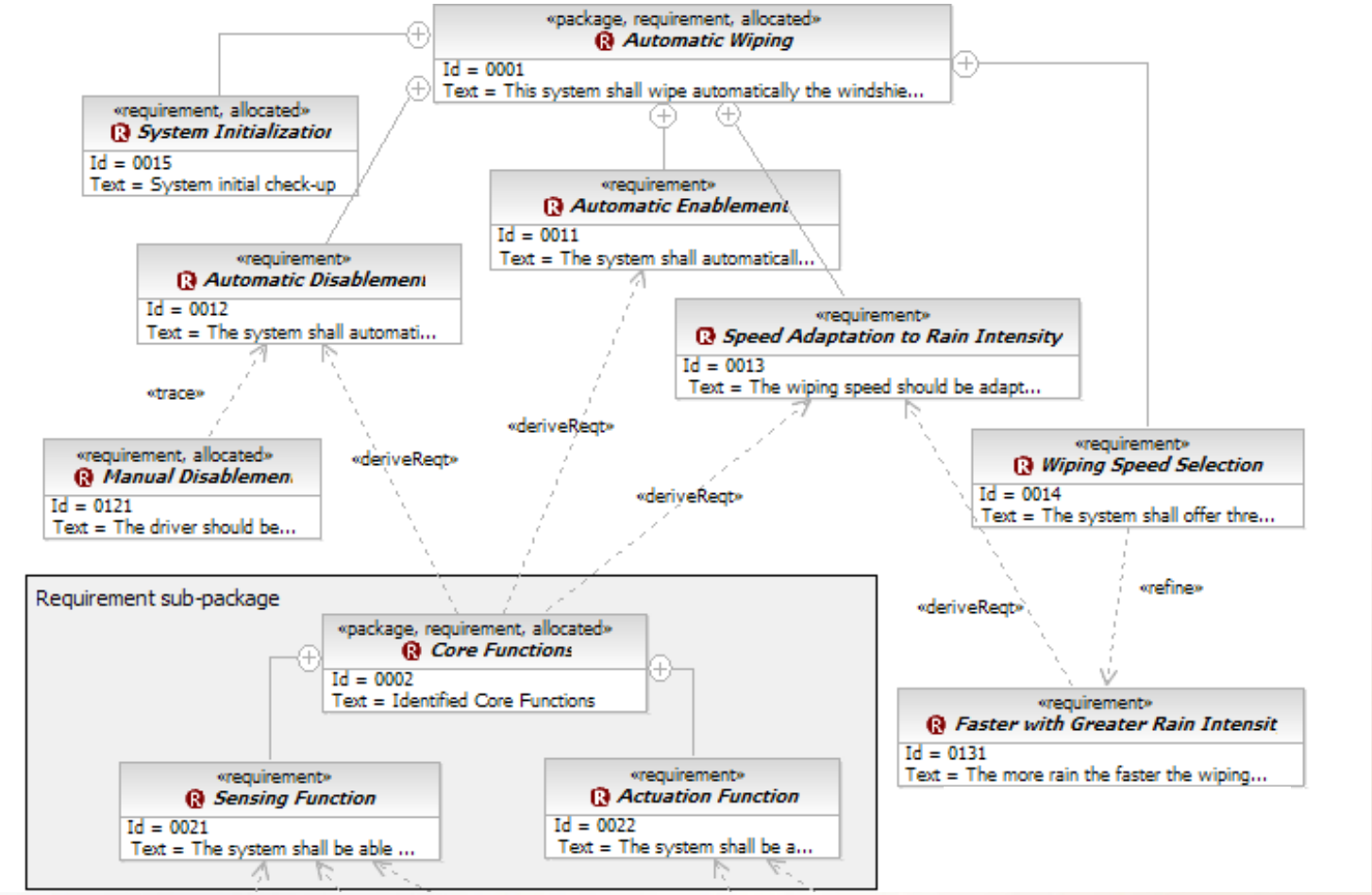
SysMLユースケース図

- 機能→製品に関するユースケース
- システムの範囲
- 外部のシステム・アクター
- ユースケースをオーナに設ける
- アクッタとシステムの連絡
- ユースケース間の関係
  - Include「埋め込んでユースケースを表す」
  - Extend 「例外的な活動を表す」



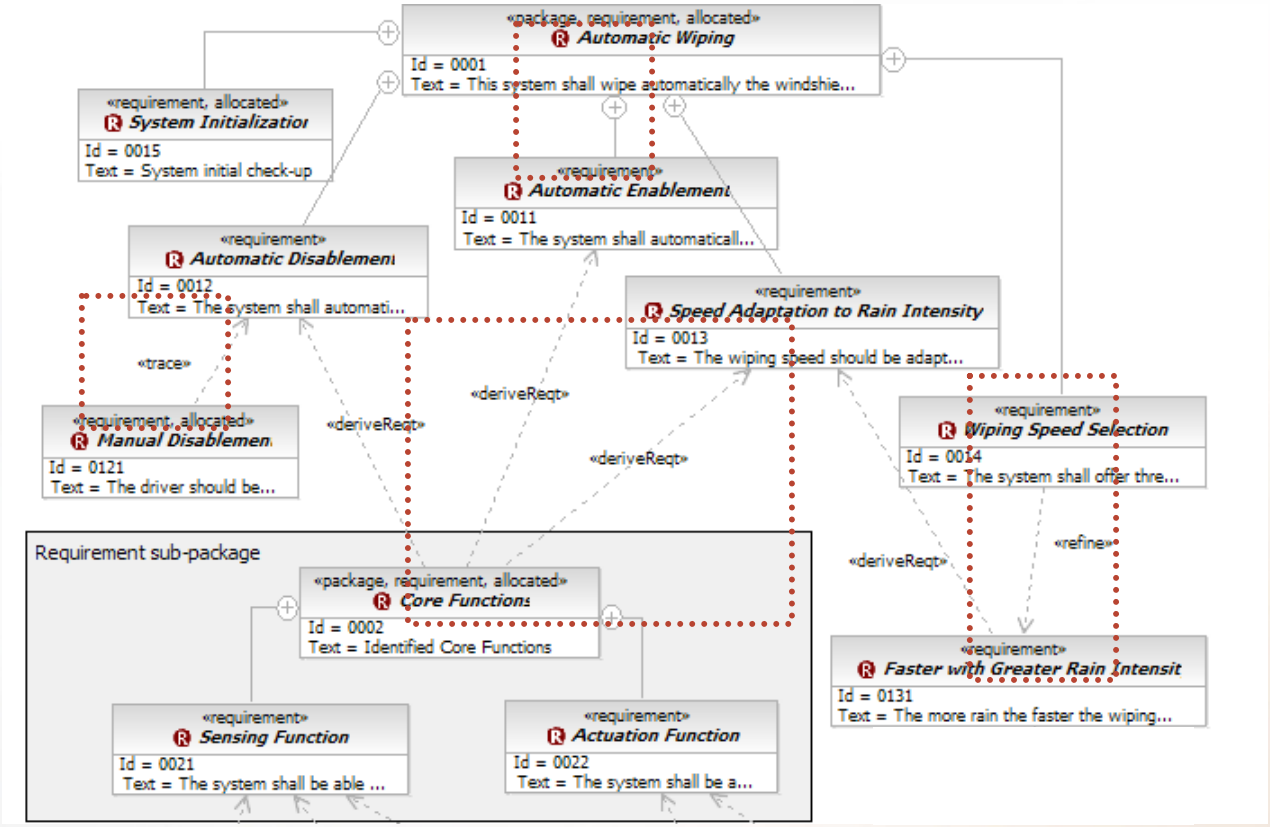
SysML要件図　整理

- SysML要件モデルは、パッケージで整理する.



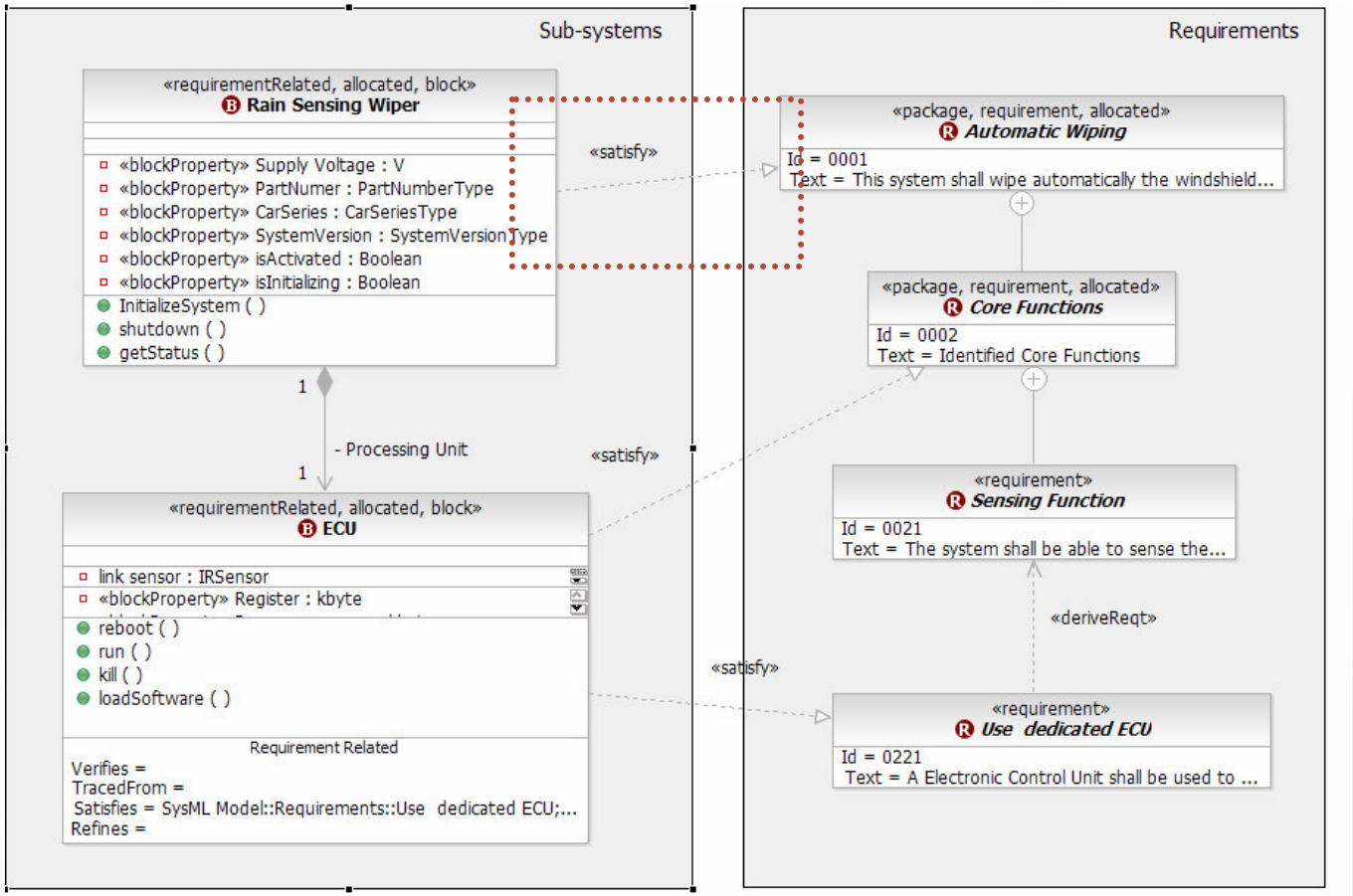
SysML要件図　関係

- 規格化した要件の関係
  - **Containment**: この関係を使うと、ある要件の定義が、他の要件の定義の中に、埋め込まれているのを表す
  - **<<trace>>**: この関係の使うと、ある要件の定義の間に、あるつながりがあるのを表す 「抽象的な関係」
  - **<<refine>>**: この関係の使うと、ある要件の定義は、他の要件の定義で、詰められるのを表す
  - **<<deriveReqt>>**: この関係の使うと、ある要件の定義で、他の要件の定義が派生されるのを表す



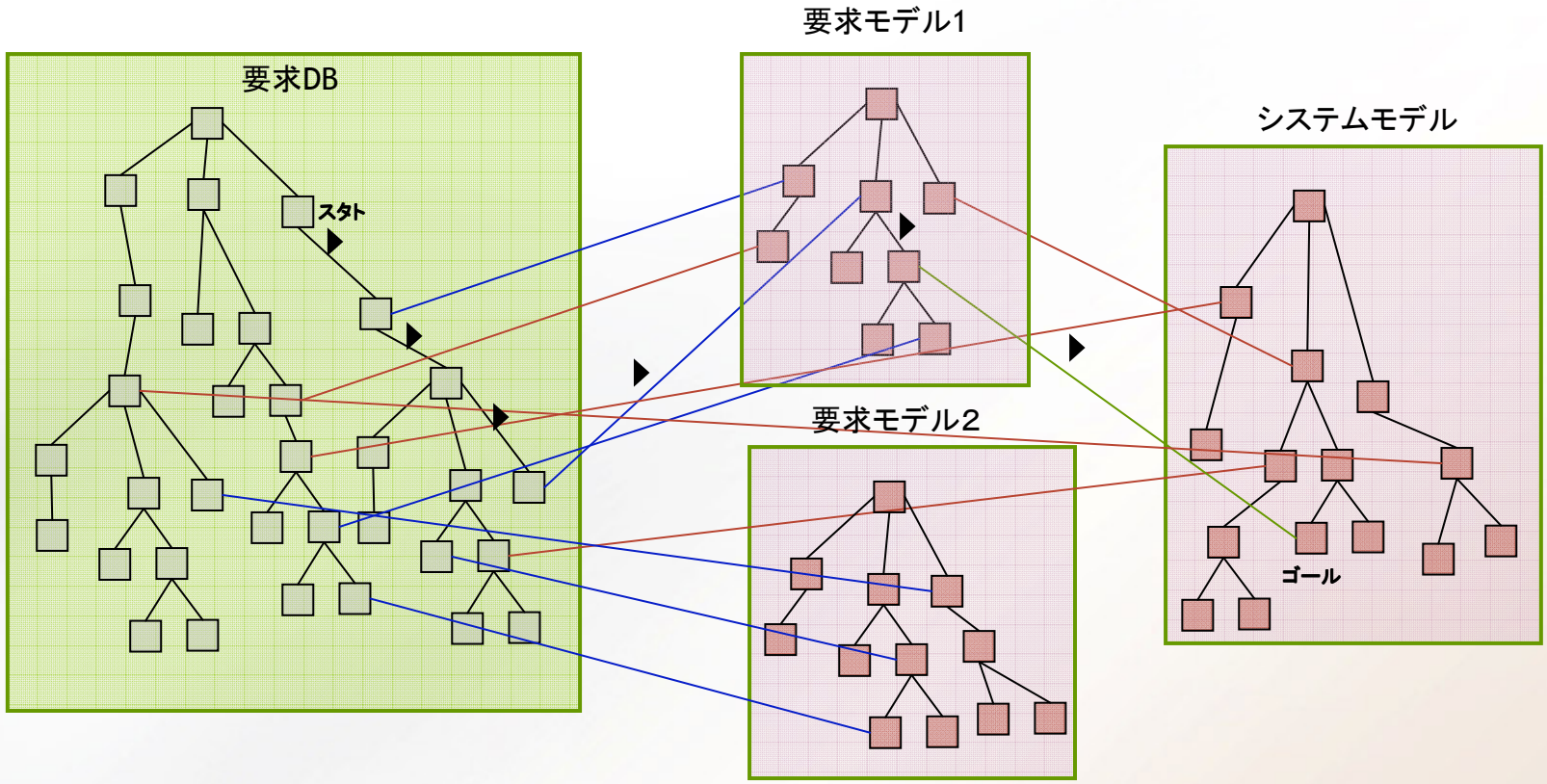
要件の定義をシステムの要素に設けろ

- システムの要素と要件の規格化した関係
  - **<<satisfy>>: その関係を使うと、システムの要素に要件の定義を設ける**



要素の間のトレーサビリティー

- モデルベースアプローチの基礎→トレーサビリティーのインフラを作成する
- 要素の間に、関係を描いておく理由
  - モデルを分析する



今日の概要

3つのシステムモデルの役割

要件モデリング

構成モデリング

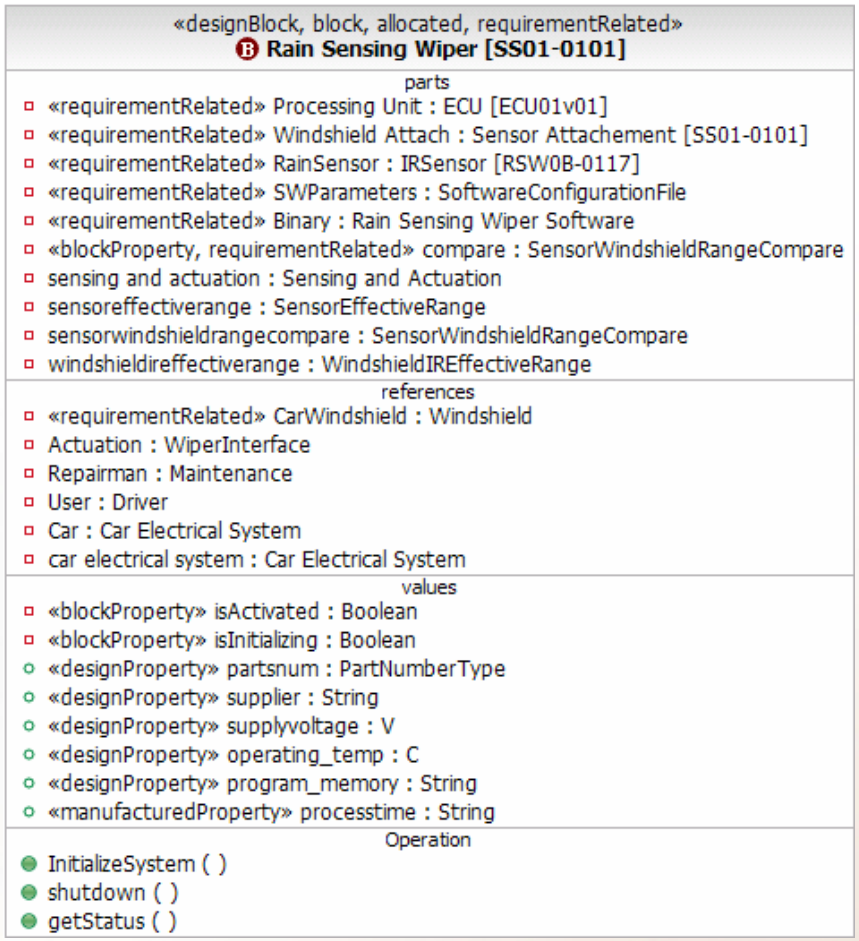
動作モデリング

結論

基本的な製品構成

Block Definition Diagramという図式

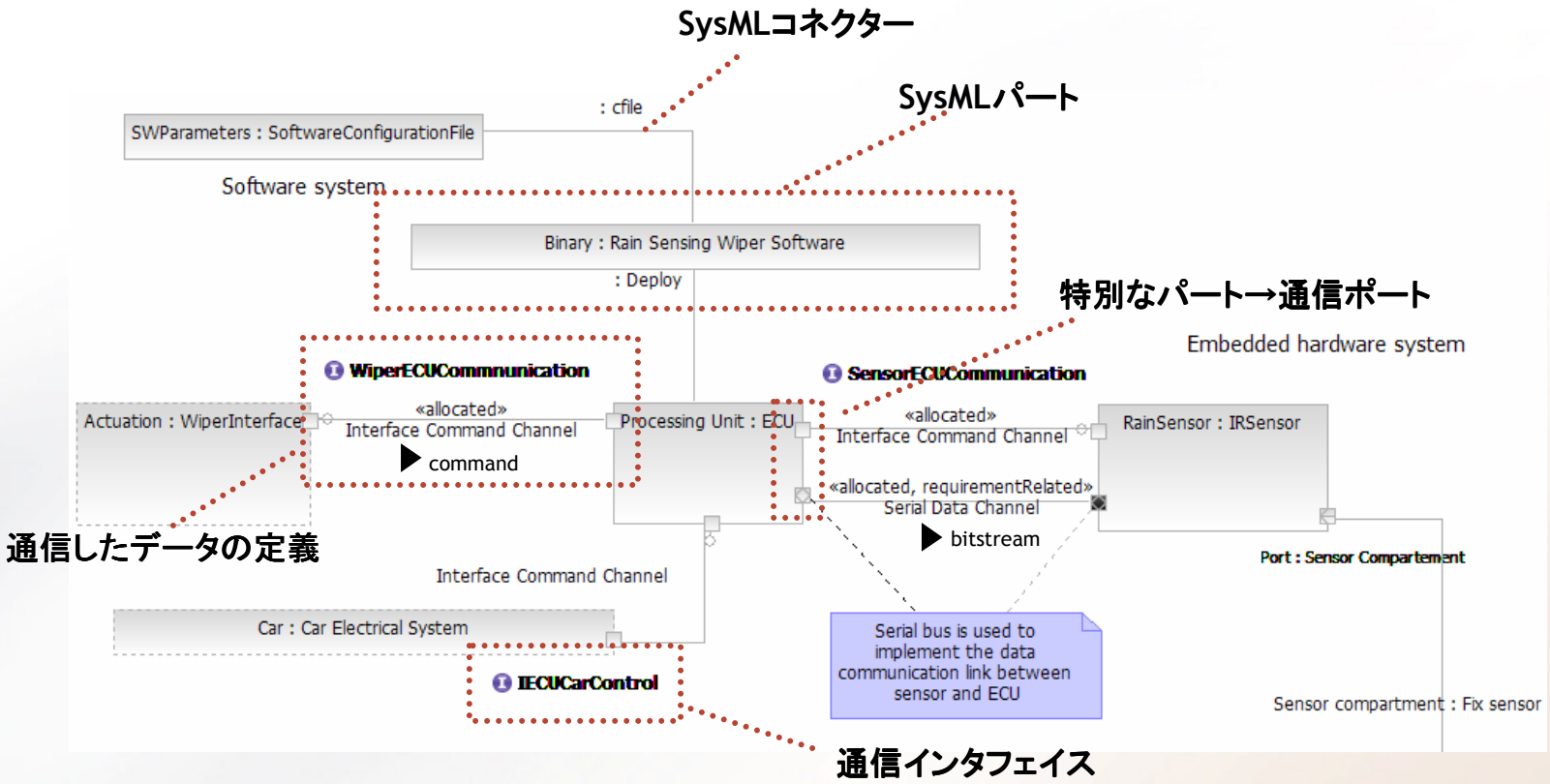
- SysMLブロック→基本的な製品構成の定義
- SysMLブロックに含まれる情報:
  - オペレージョン　「操作」
  - 属性
- 属性には三つの種類がある
  - パート (埋め込んでいる)
  - レファレンス・参考　(埋め込んでいない)
  - 属性の値



インスタンス生成

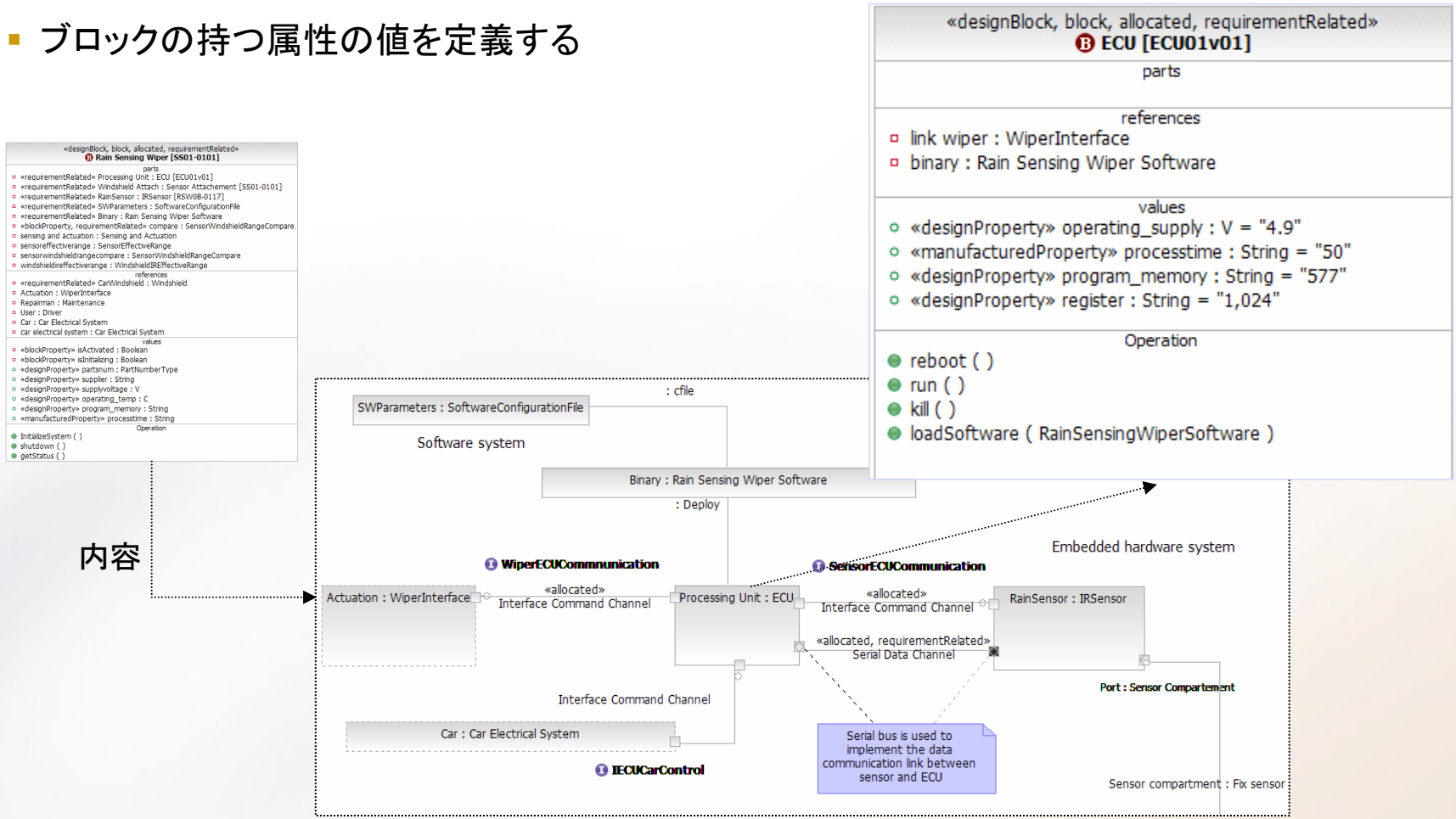
Internal Block Diagramという図式

- 非機能による制約に合わせて、製品の機能を提供する要素を組み合わせる製品構成が描かれる
- 製品構成で、要素の組み合わせが描かれる、だから、動作の記述ではない。



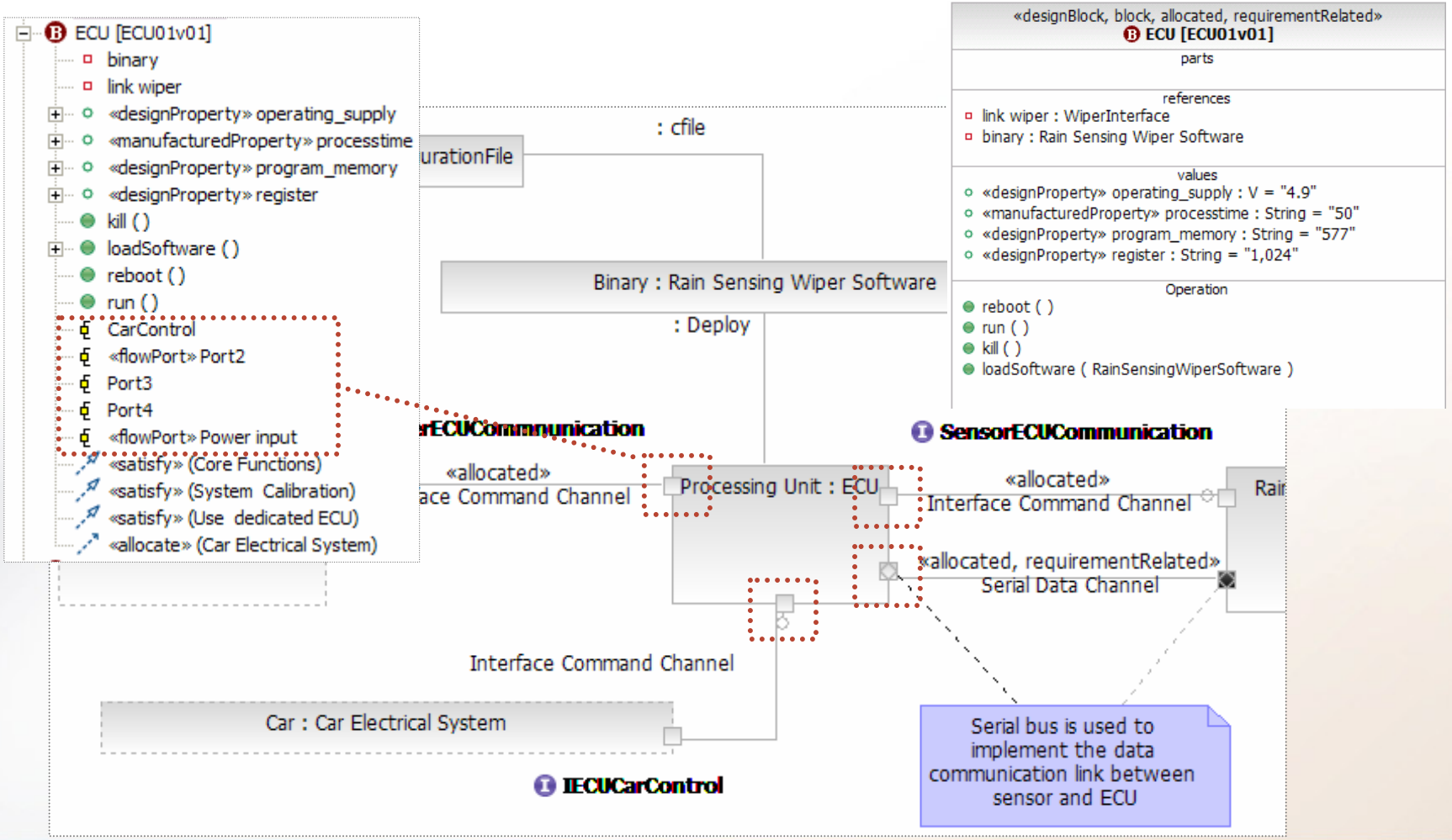
定義から、インスタンス生成

- 製品の構成のため、二つの記述の種類→定義と使い方
- 定義を作成した後、ブロックの中に、インスタンスとして使う
- ブロックの持つ属性の値を定義する



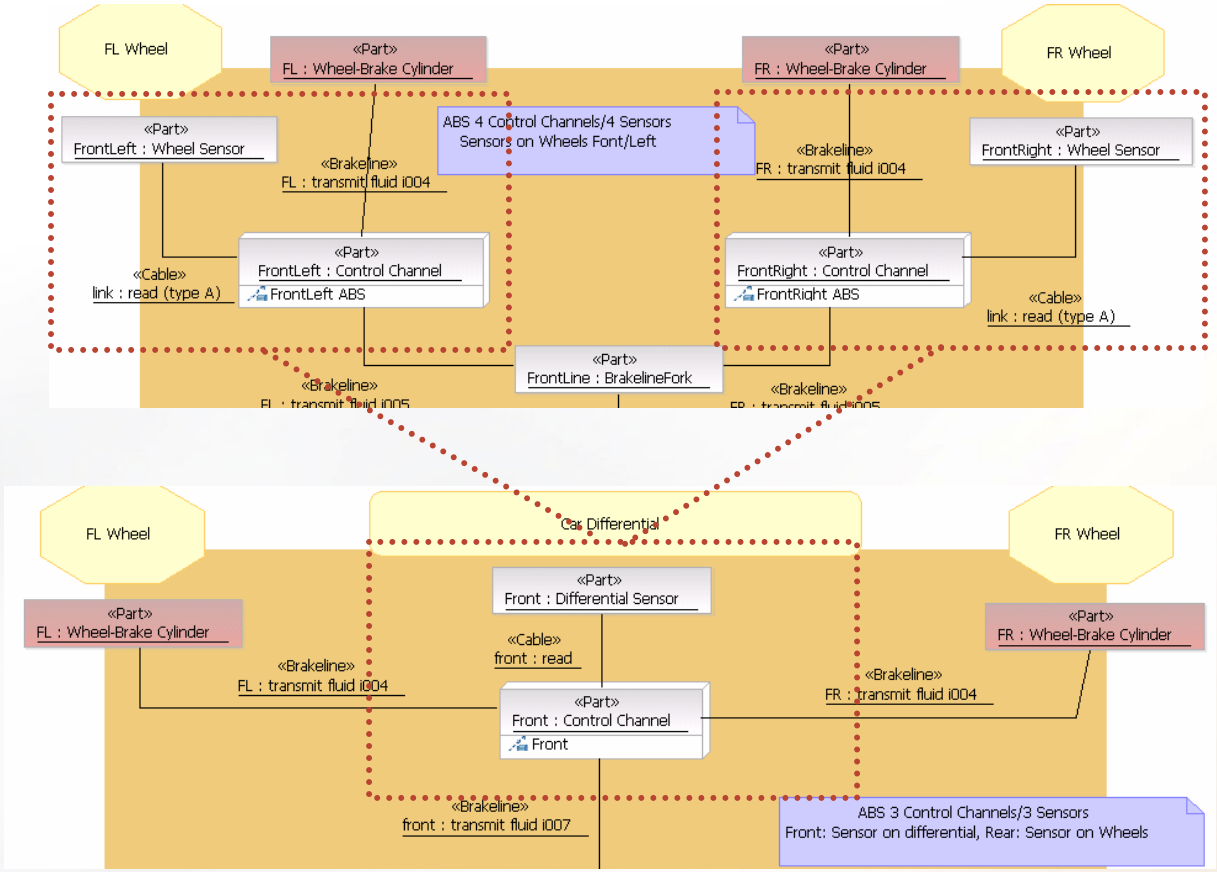
製品の構成のモデリング→通信ポート

- ポートは、特別なパートとして、ブロックの外部の要素につなげる要素である。
- ポートの機能は、インテフェスによって、定義される



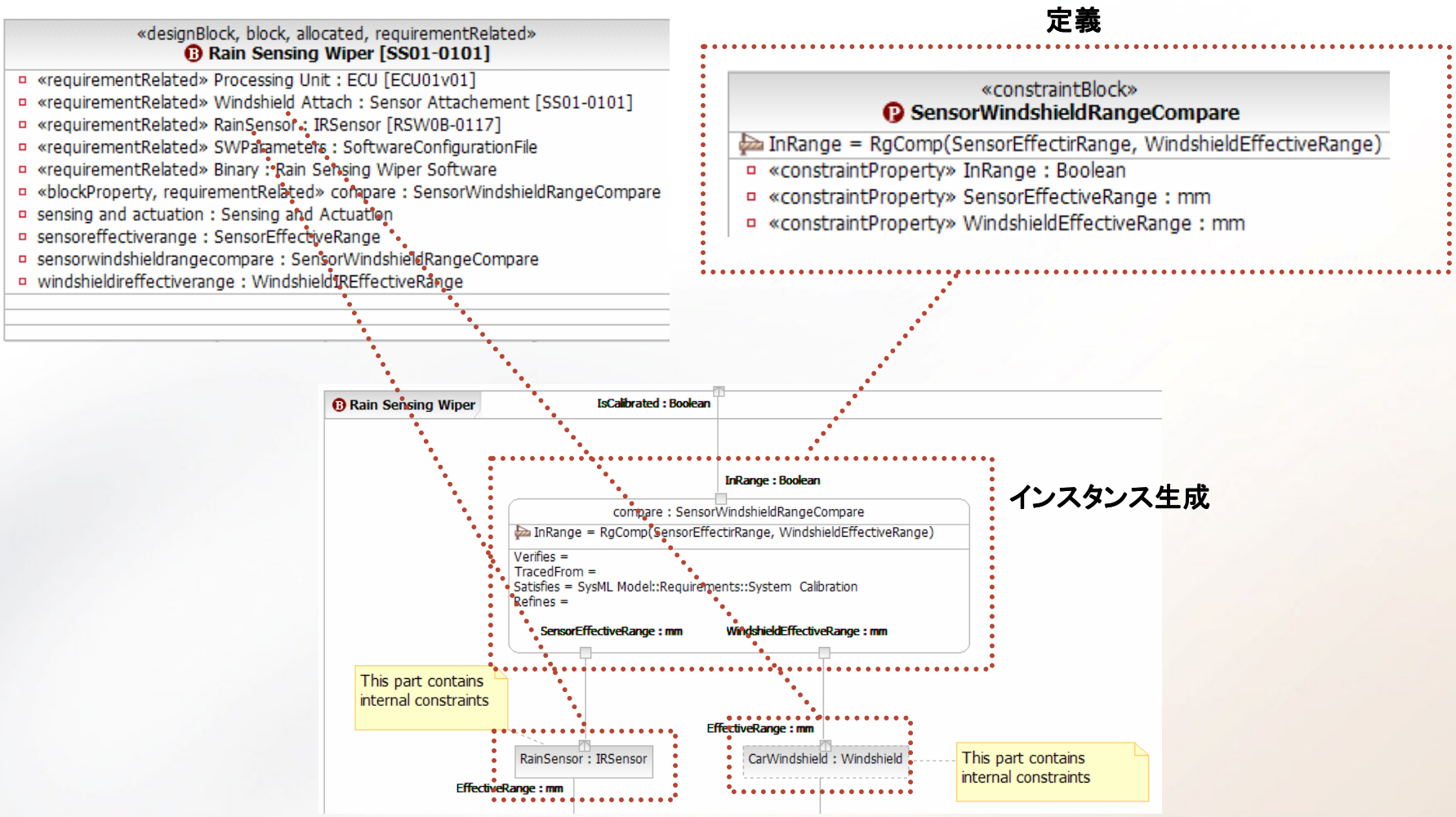
製品の構成の使い方

- 製品の構成は、異なった製品のコンフィグレーションの中に、選択する
- 要素の種類は、異なった種類を使えますから、機能的に、物理的になど、コンフィグレーションを評価したり最適化したりできる



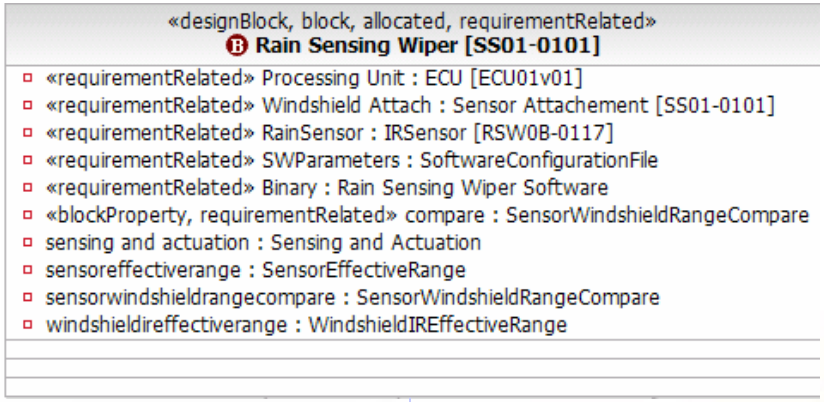
システムの制約

- 属性の間に、制約を記述してシステムの特性の整合性を検証する
- ブロックを定義するように、制約の状態が二つある→定義とインスタンス生成

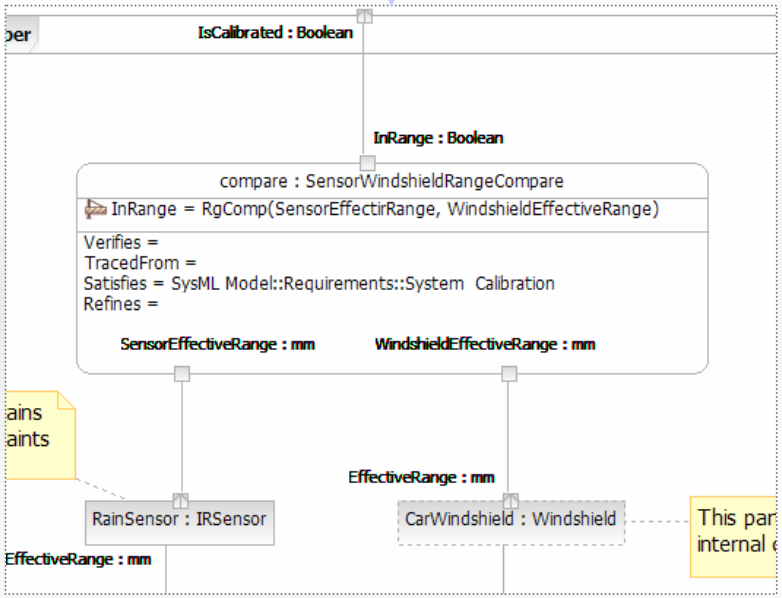


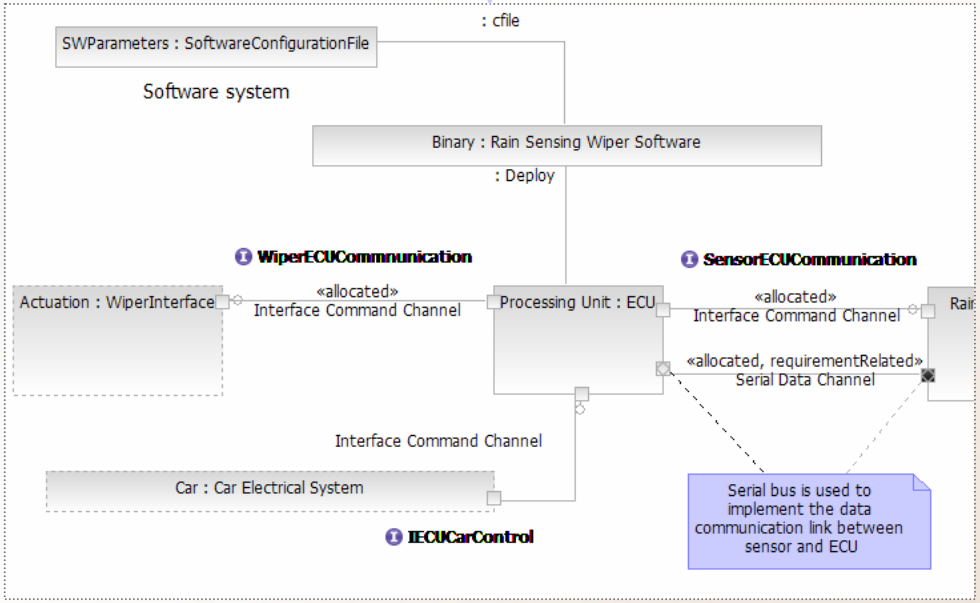
構成モデリングの概要

- SysML言語によって、製品の構成は、三つの手段で現れる
  - 定義→ ブロックの定義図 （BDD)
  - 使う状態（インスタンス）→ブロックの内部図 （IBD)
  - 制約を記述する状態→パラメトリック図

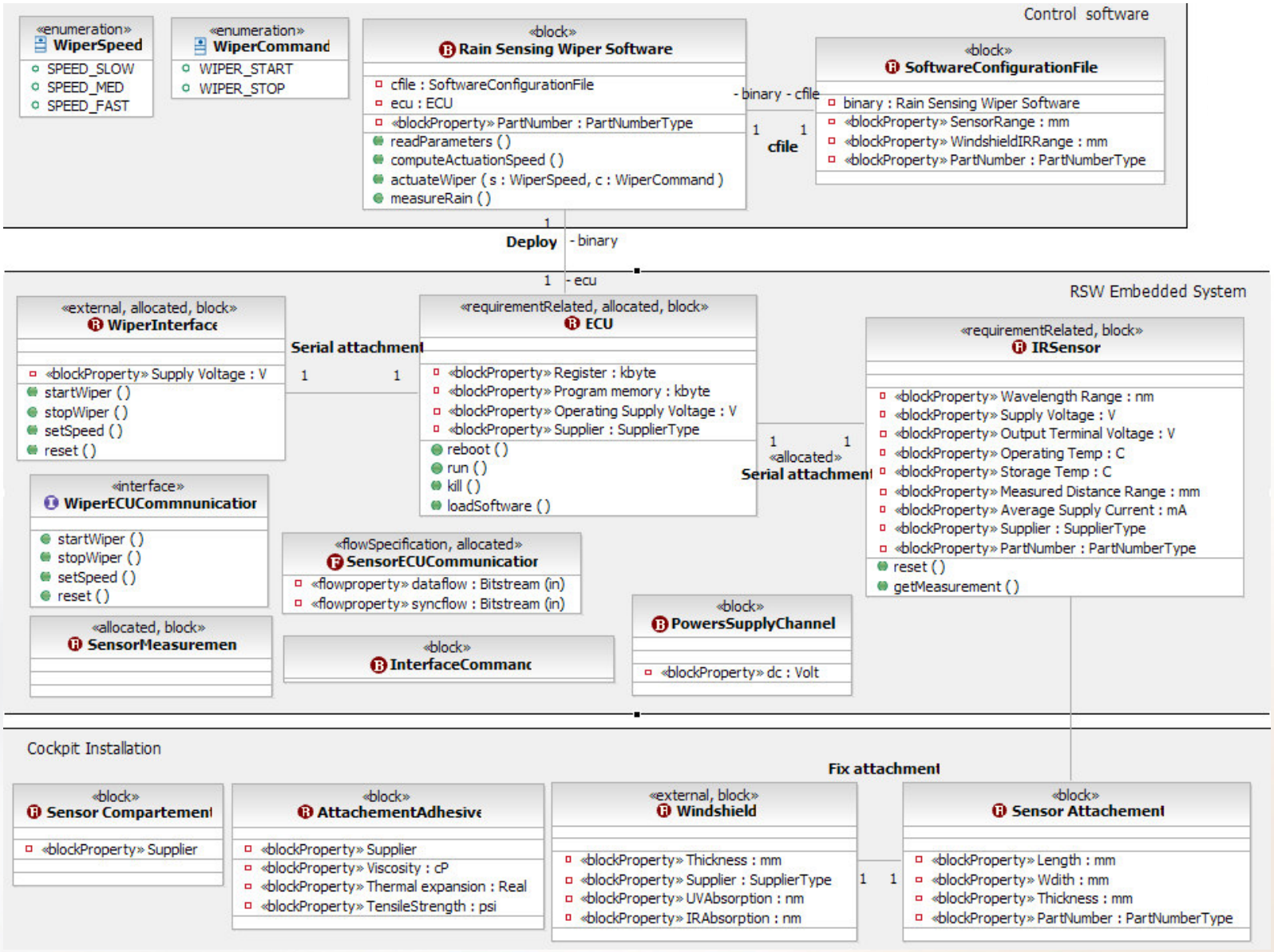


ブロックの定義の中





複雑な製品の構成の定義



今日の概要

3つのシステムモデルの役割

要件モデリング

構成モデリング

動作モデリング

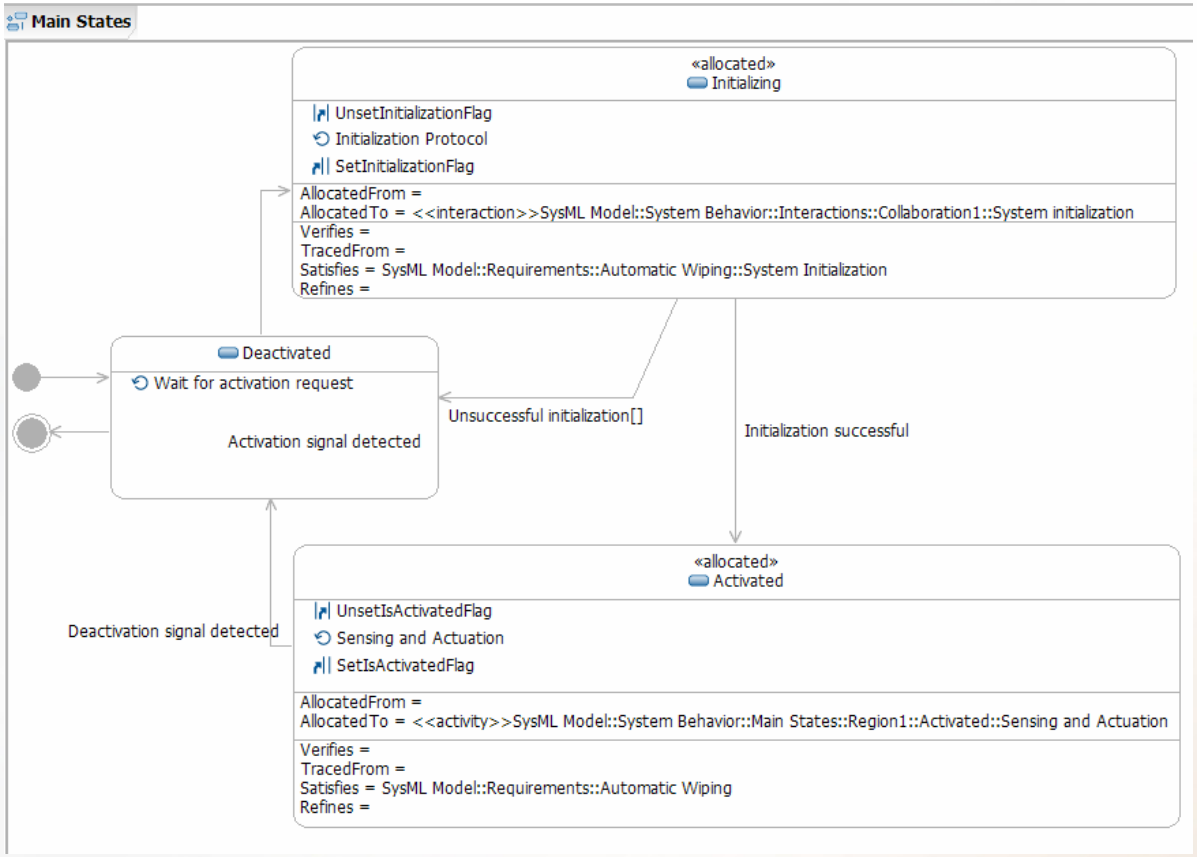
結論

SysMLによる動作モデリング

- SysML言語は、動作をモデリングする仕組みが三つある
  - アクティビティ図→非同期動作の活動を表現します
  - インタラクション図→同期動作の通信を表現します
  - ステート図→製品の状態遷移を表現します
- 以上の動作モデルは、独立しないモデル→モデルを組み合わせられる
- 動作モデルの選択は、システムの表現したい部分に応じて、使う
  - 例えば、ある基準において、モデルの選択
    - 「時間によらない、独立した活動、活動の順番、データ通信」→アクティビティ図
    - 「時間による、メッセージの流れ、データ通信」→インタラクション図
    - 「時間によらない、状態の順番」→ステート図

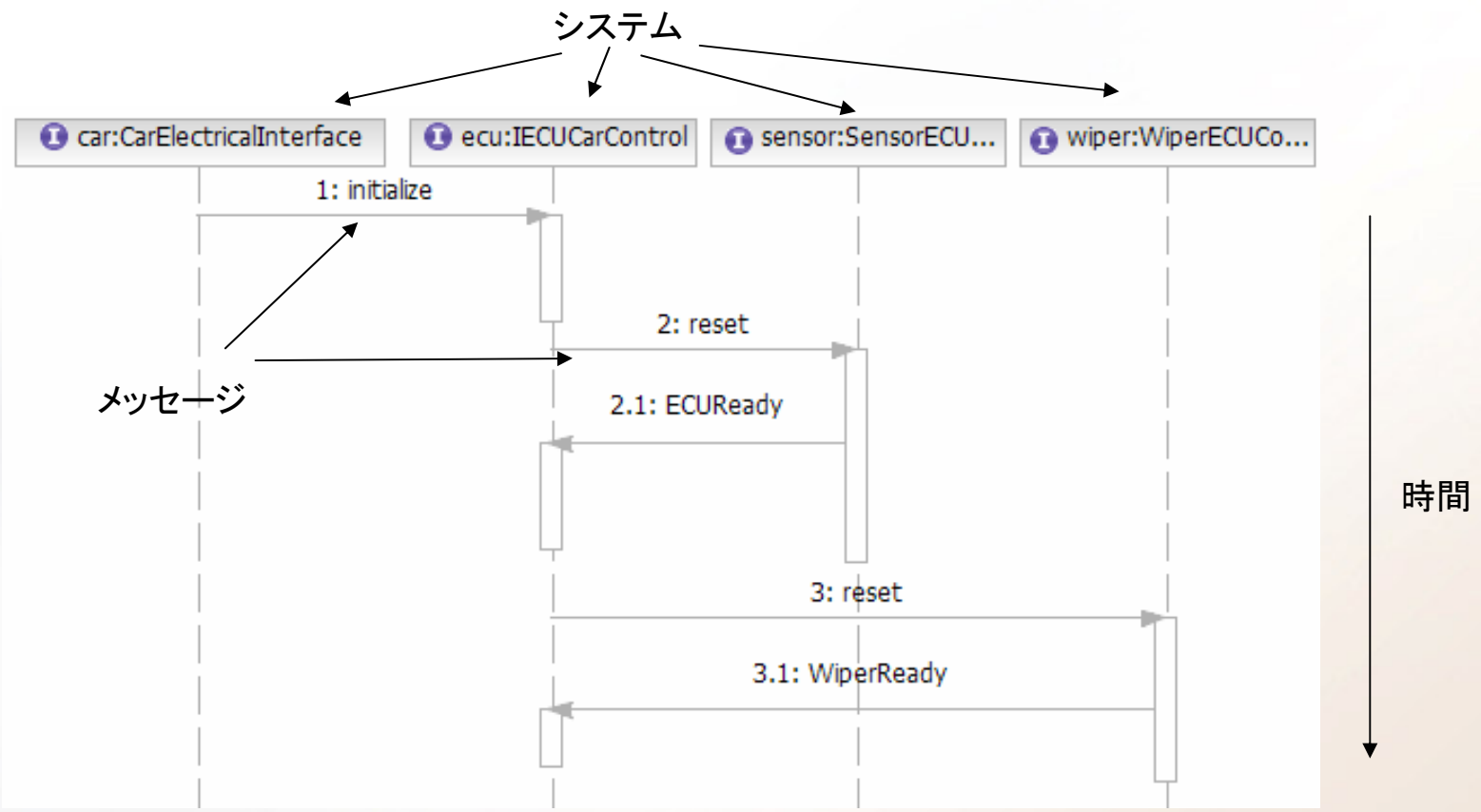
動作モデリング→状態図

- 状態図によって、システムの状態と状態推移を記述する仕組みです。
  - UML2.1に比べて、限定された図式→プロトコルステート図が除いている
- モデルの範囲→状態に入る条件・状態に出る条件・状態で行う活動・状態遷移を定義する



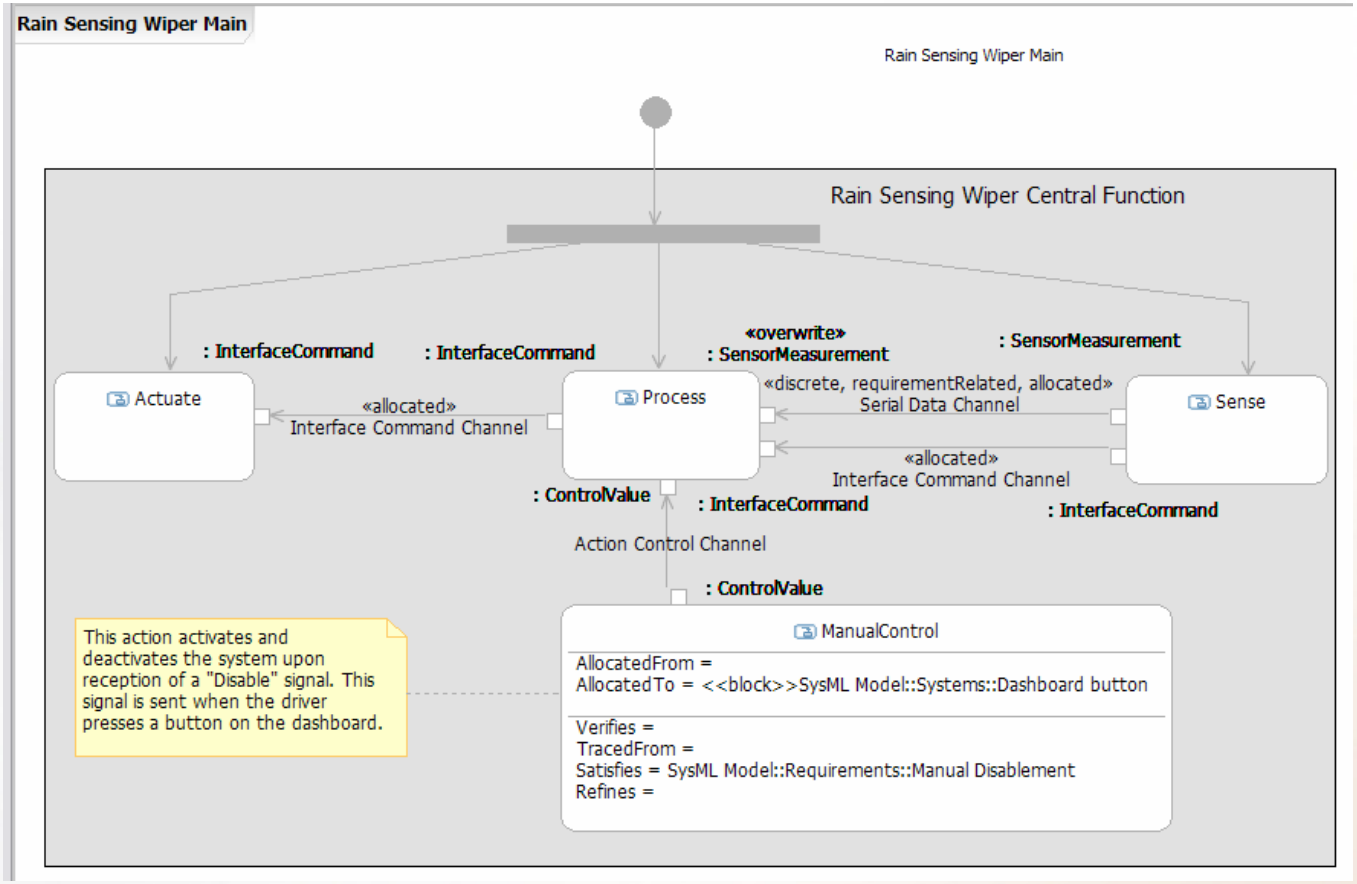
動作モデリング・同期動作→インタラクション

- インタラクション図によって、システムの間に同期的にメッセージの通信を記述し、メッセージは時間順で並んでいます。
  - UML2.1に比べて、限定された図式→コミュニケーション図とタイミング図は除いている。

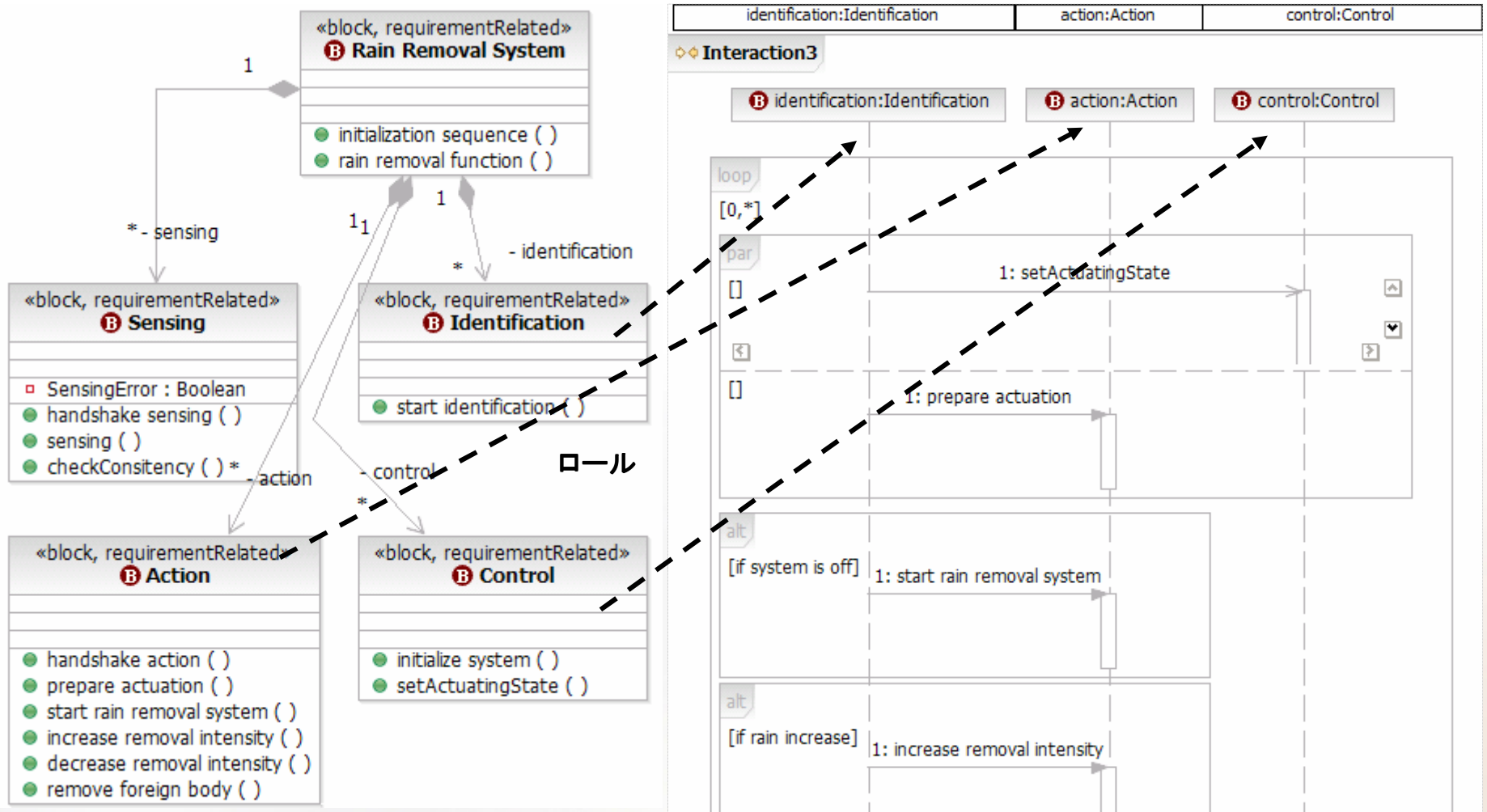


動作モデリング・非同期動作→アクティビティ

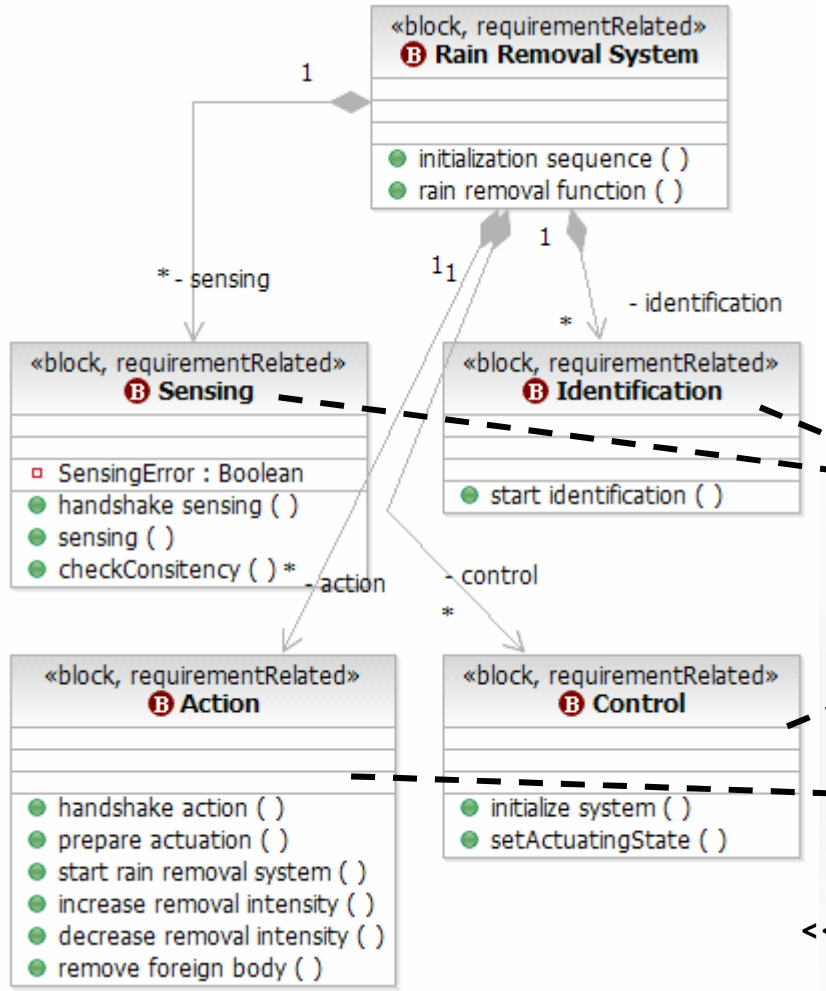
- アクティビティ図によって、独立した活動の流れを描き、活動はデータを交換し、連続的な通信も記述でき、また離散的な通信も記述できる
  - UML2.1に比べて、拡張されたモデル

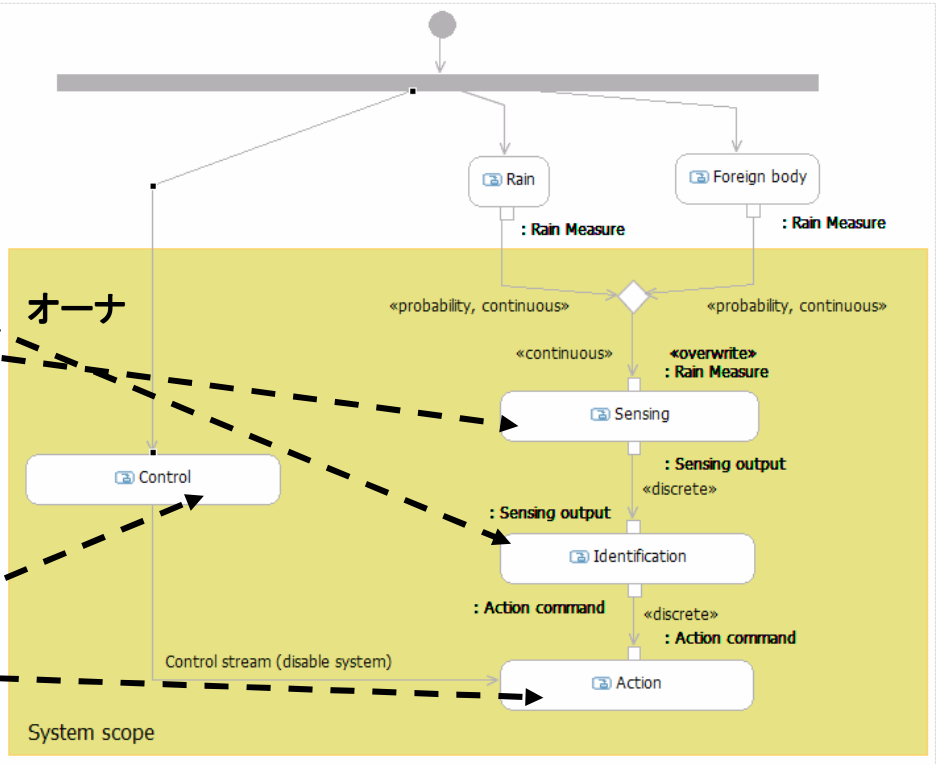


構成と動作の関係 （1）



構成と動作の関係 （2）

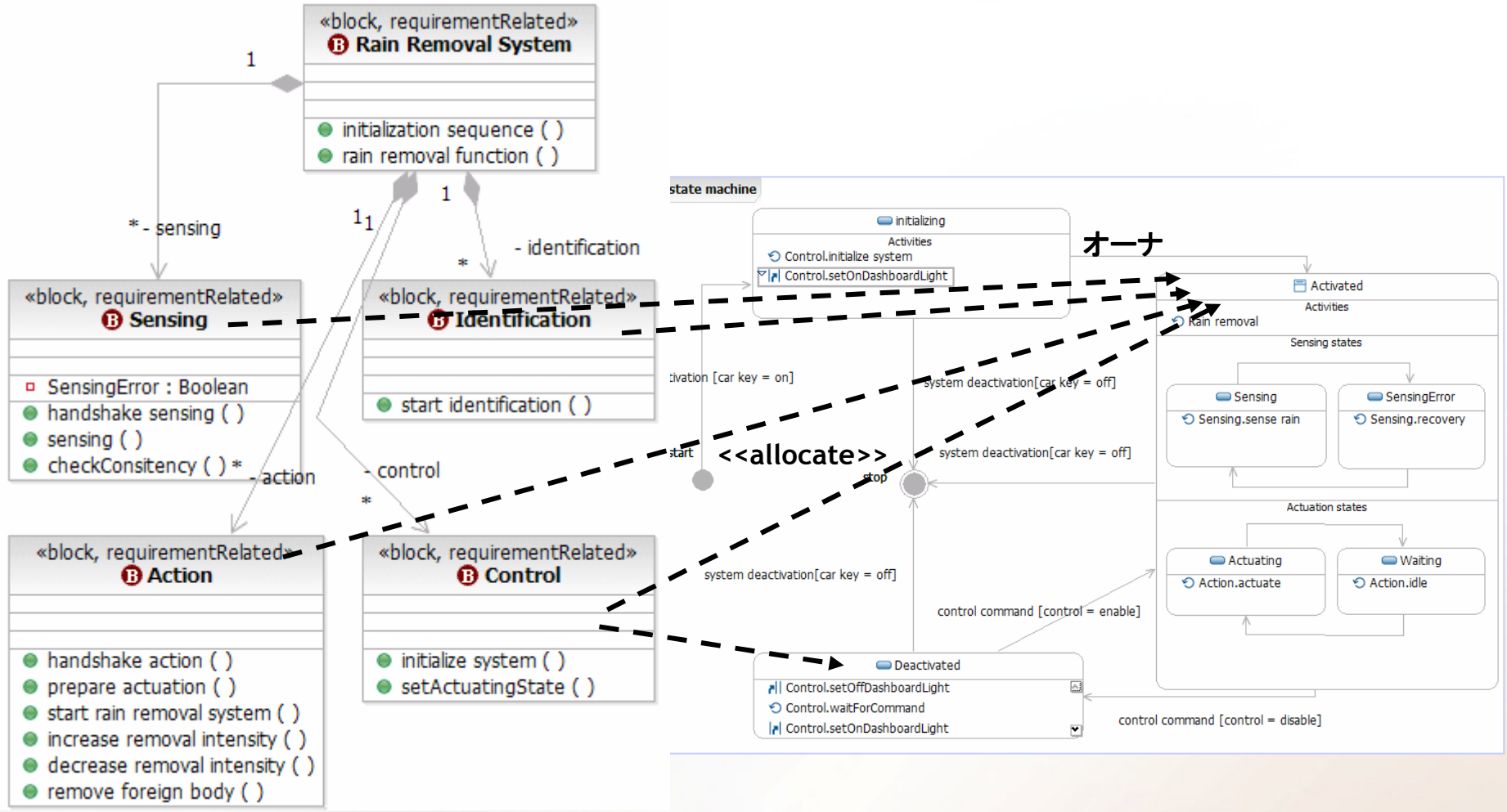




オーナ

<<allocate>>

構成と動作の関係 （3）



今日の概要

3つのシステムモデルの役割

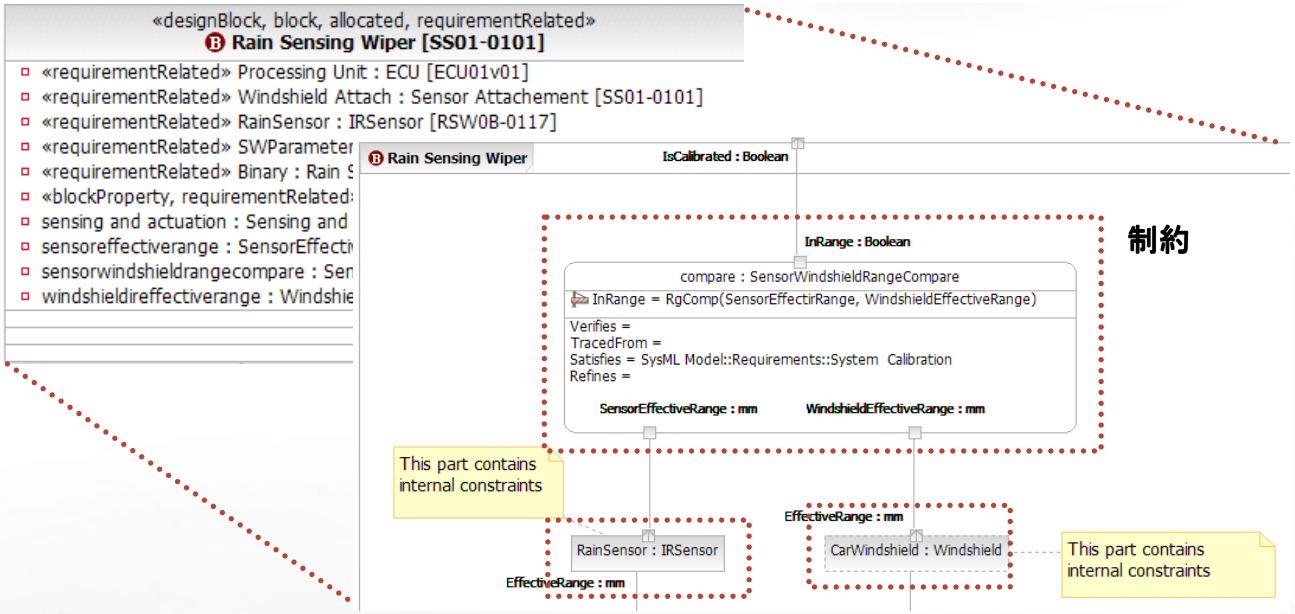
要件モデリング

構成モデリング

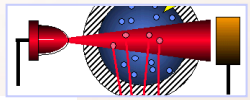
動作モデリング

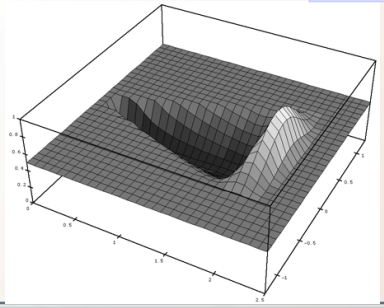
結論

自動車会社の問題の解決



制約

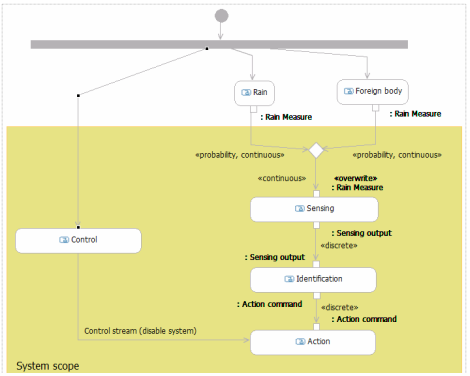




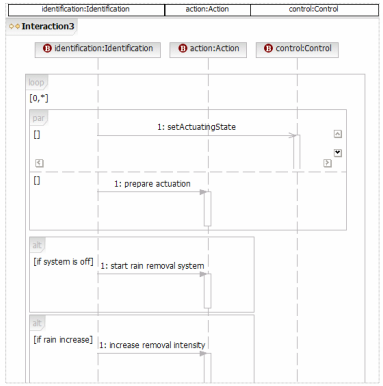
SysML言語の図式の概要

動作

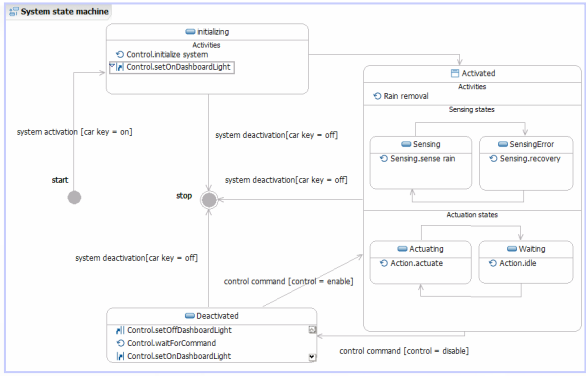
アクティビティ



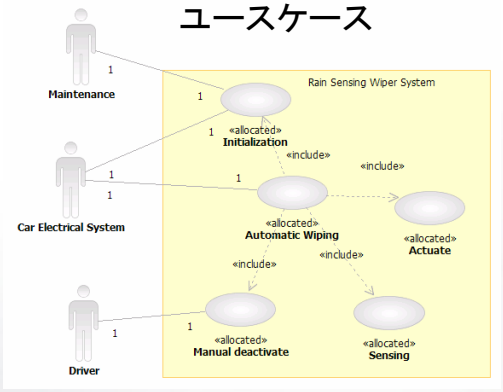
インタラクション

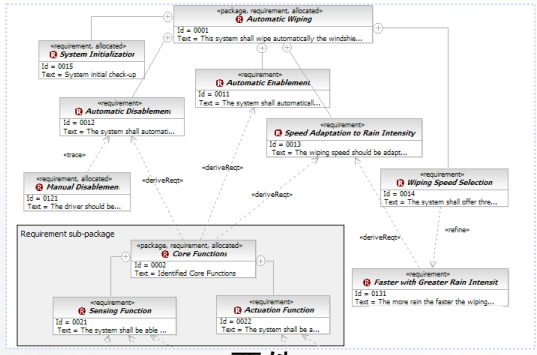


スタート

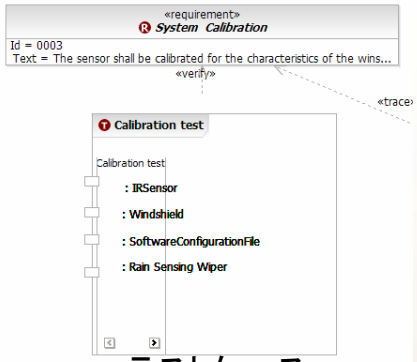


ユースケース





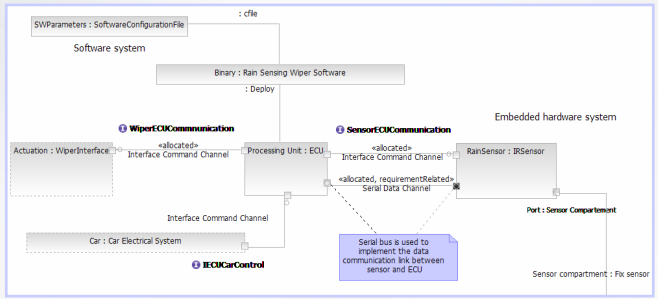
要件



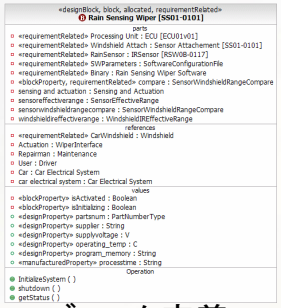
テストケース

要件

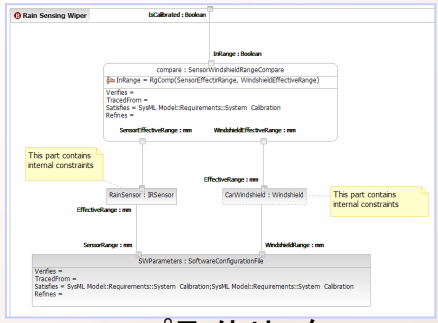
構成



ブロックの内部



ブロック定義



パラメトリック

結論・次の授業

重要なポイント

- 異なった分野で設計した部品があるから、開発に関する学際的な依存関係がある
- 全体的な製品に対する制約を、システムのレベルで記述するしかない

今回紹介した内容

- SysMLによっての構成・動作モデリング

次の授業

- システムエンジニアリングのプロセスと方法論