

AtScopeを使って 図面化してみよう

2024年6月5日

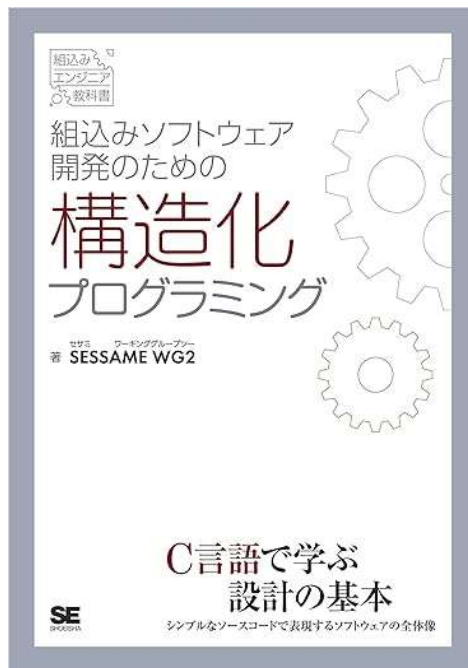
ビースラッシュ株式会社



- 解析対象ソースコードの入手
- コンポーネント構造図とファイル構造図
 - A2Gスタイル
- ソースコードの資産レベル
 - 戦略資産／組織資産／属人資産／在庫／不良在庫
- 関数構造図（関数起点と変数起点）
- 「配置を保存」で図面化
 - 配置は一度整えることで、次からはその配置で図面化

図面化対象のソースコードの入手

- 構造化プログラミング本のソースコードを図面化してみましよう
 - 書籍はこちら
 - <https://www.shoeisha.co.jp/book/detail/9784798147611>
 - ソースコードはこちら
 - https://www.bslash.co.jp/books/str_prg

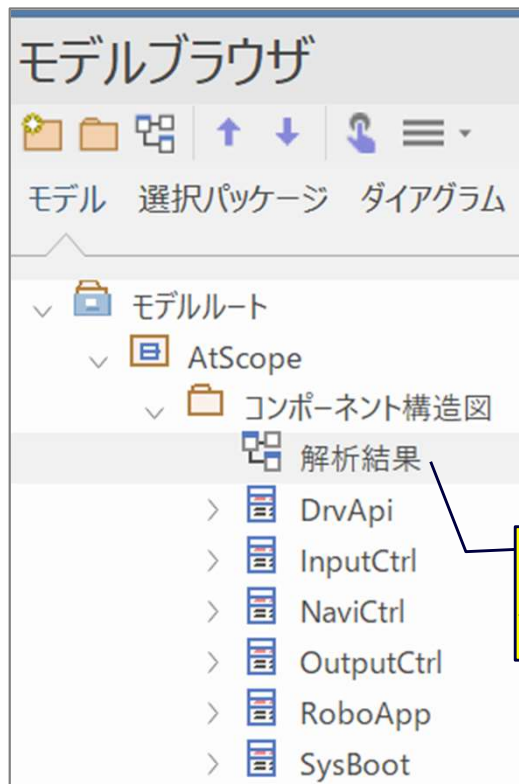


TraceBaseフォルダ内の図面化

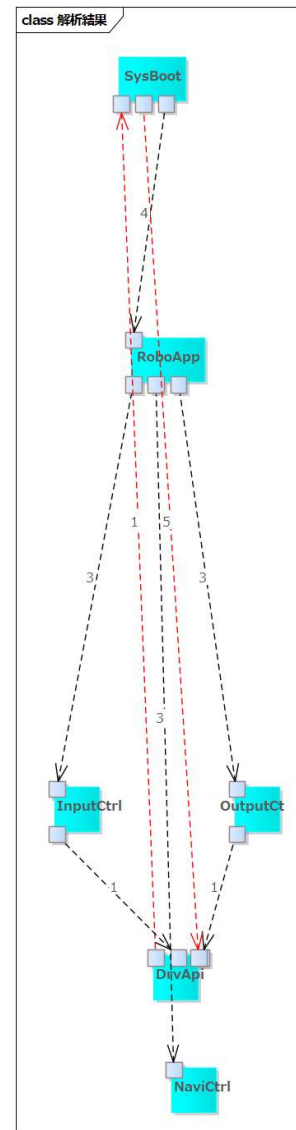
- AtScopeを起動します
- TraceBaseのProjフォルダを選択してください
- 実行ボタン押下で、ファイル単位のコンポーネント構造図が出力されます



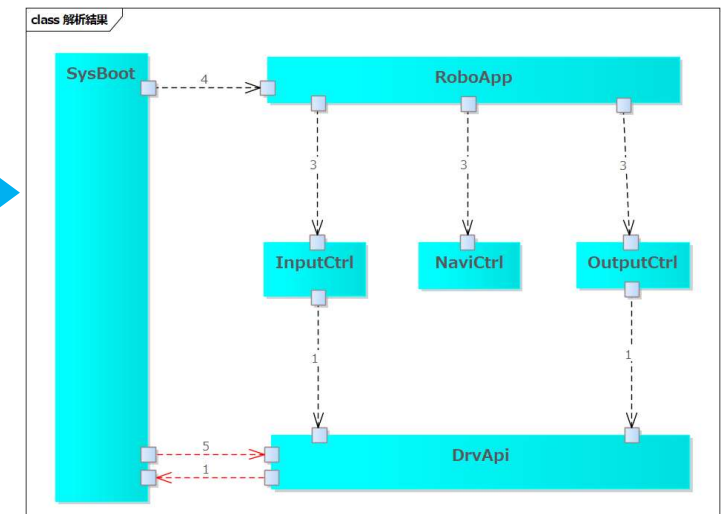
コンポーネント構造図



解析結果を
ダブルクリック



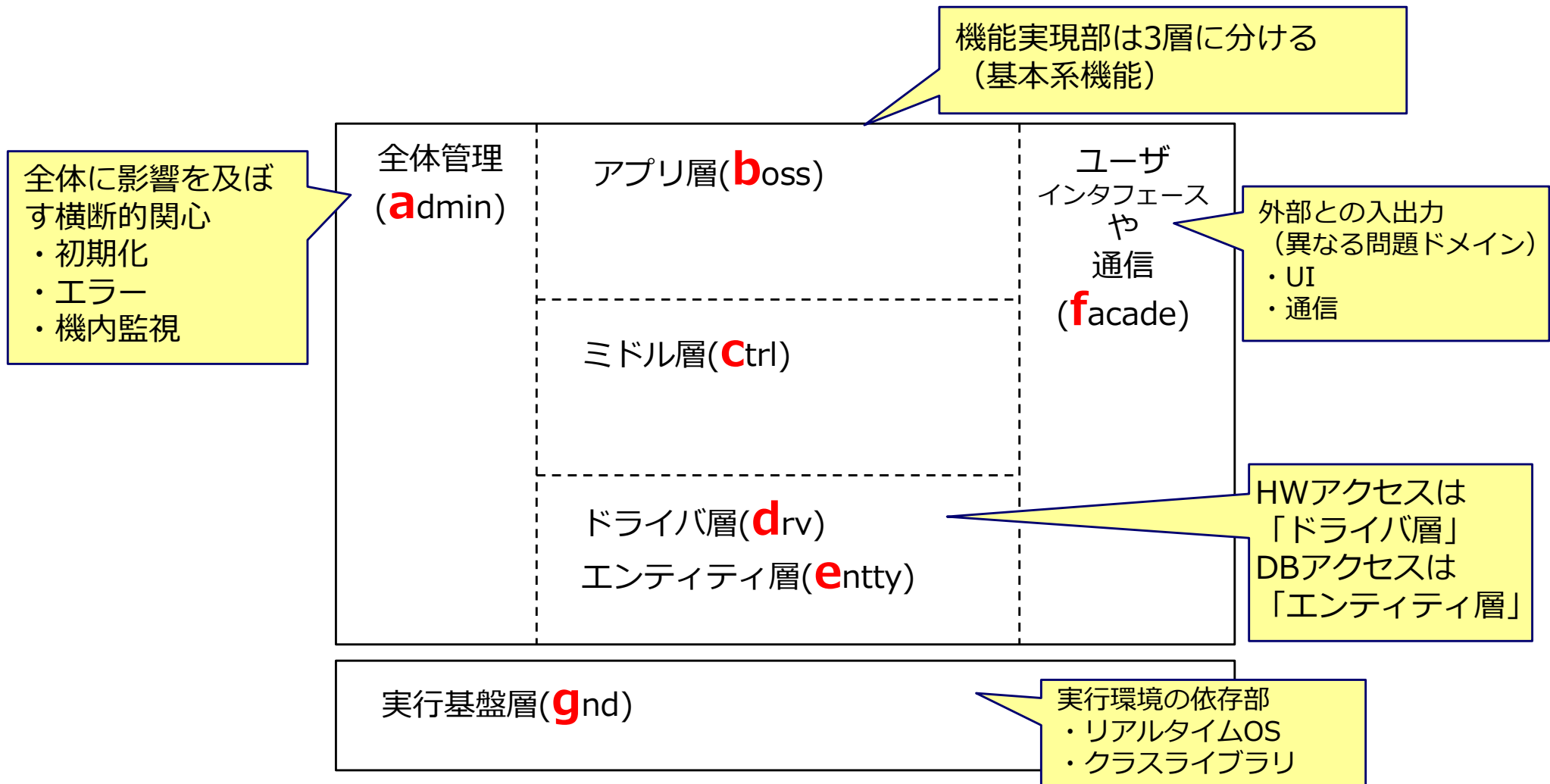
配置を変えることで設計意図
が見えてきます



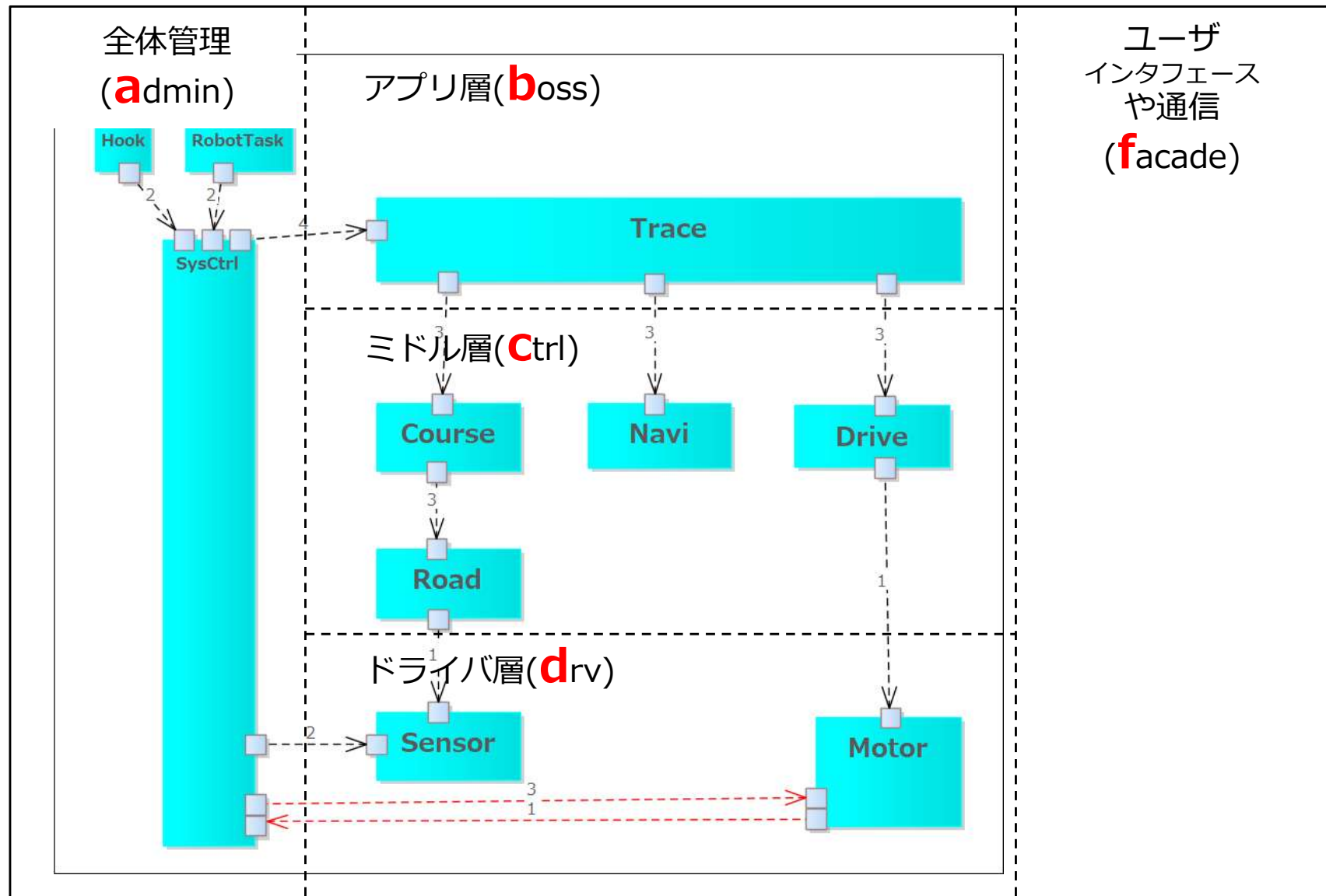
A2Gスタイル（次ページ）に
従って配置

■ 大局的な設計テンプレート

- ファイルを配置することで、設計意図が明確になる
- アーキテクチャとしてのシンプルさを維持できる



- ファイル構造図をA2Gスタイルに従って配置



ソースコードの資産レベルが分かります

コンポーネント構造図とファイル構造図で、おおよその資産レベルが分かります

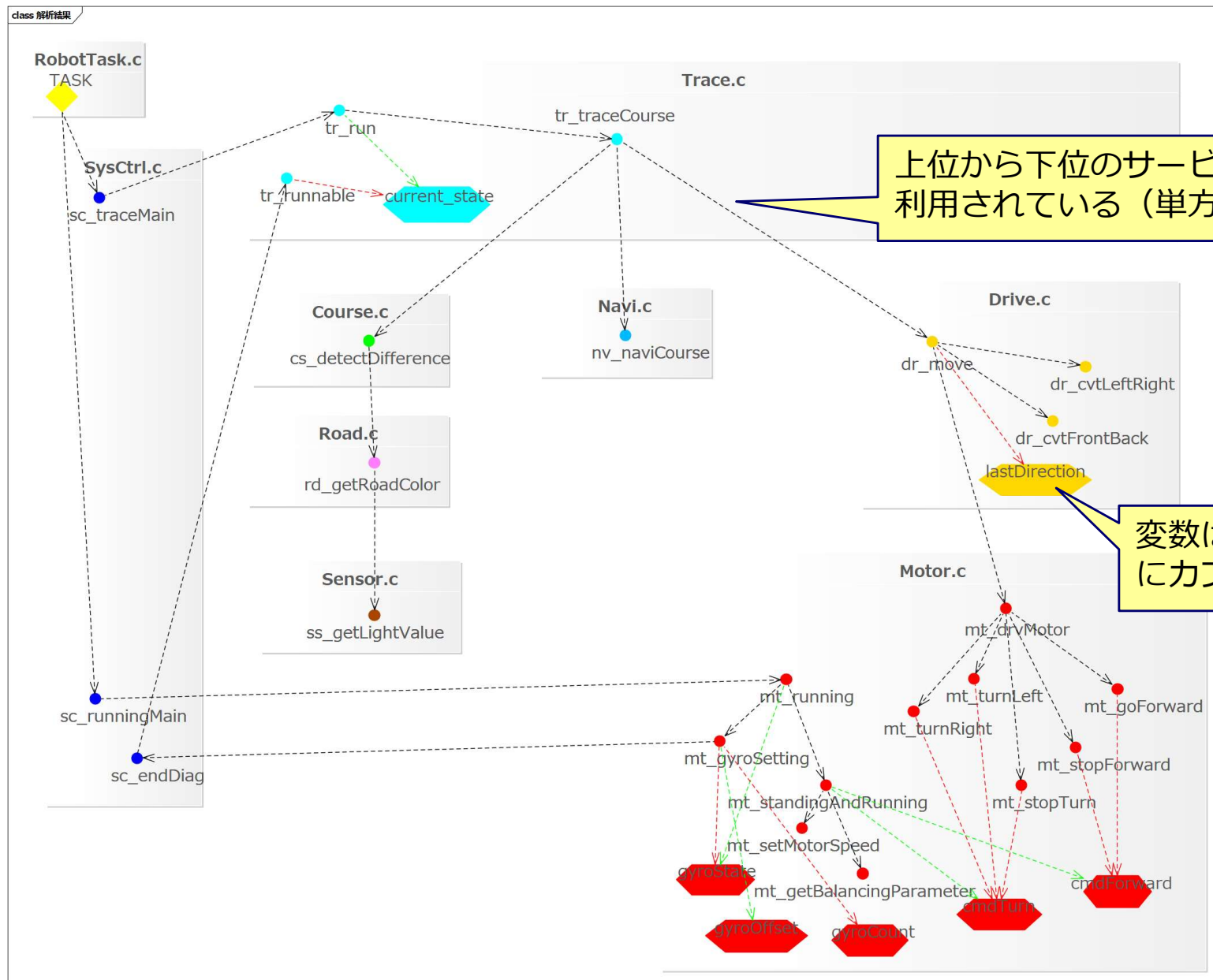
資産レベル		コンポーネント 構造図	ファイル構造図	状況	想定原因
1	戦略資産	○	○	シンプルで美しく 成長しやすいコード	アーキテクトと担当者の 連携がうまくいっている
2	組織資産	○	×	局所的な設計が崩れ ている	アーキテクトは活躍して 担当者の設計スキル不足
3	属人資産	×	○	大局的な設計が崩れ ている	アーキテクトの関与不足
4	在庫	×	×	全体が絡み合ってい る一枚岩	構造レビュー欠如
5	不良在庫	××	××	そもそも構造設計さ れていない	動くコードを積み上げて いく文化

○：相互依存が限定的で、かつ関数複雑度も低い

×：相互依存が多く、関数複雑度が10を超えている関数が多い

××：構造が見えない（配置を工夫しても設計意図が見えてこない）

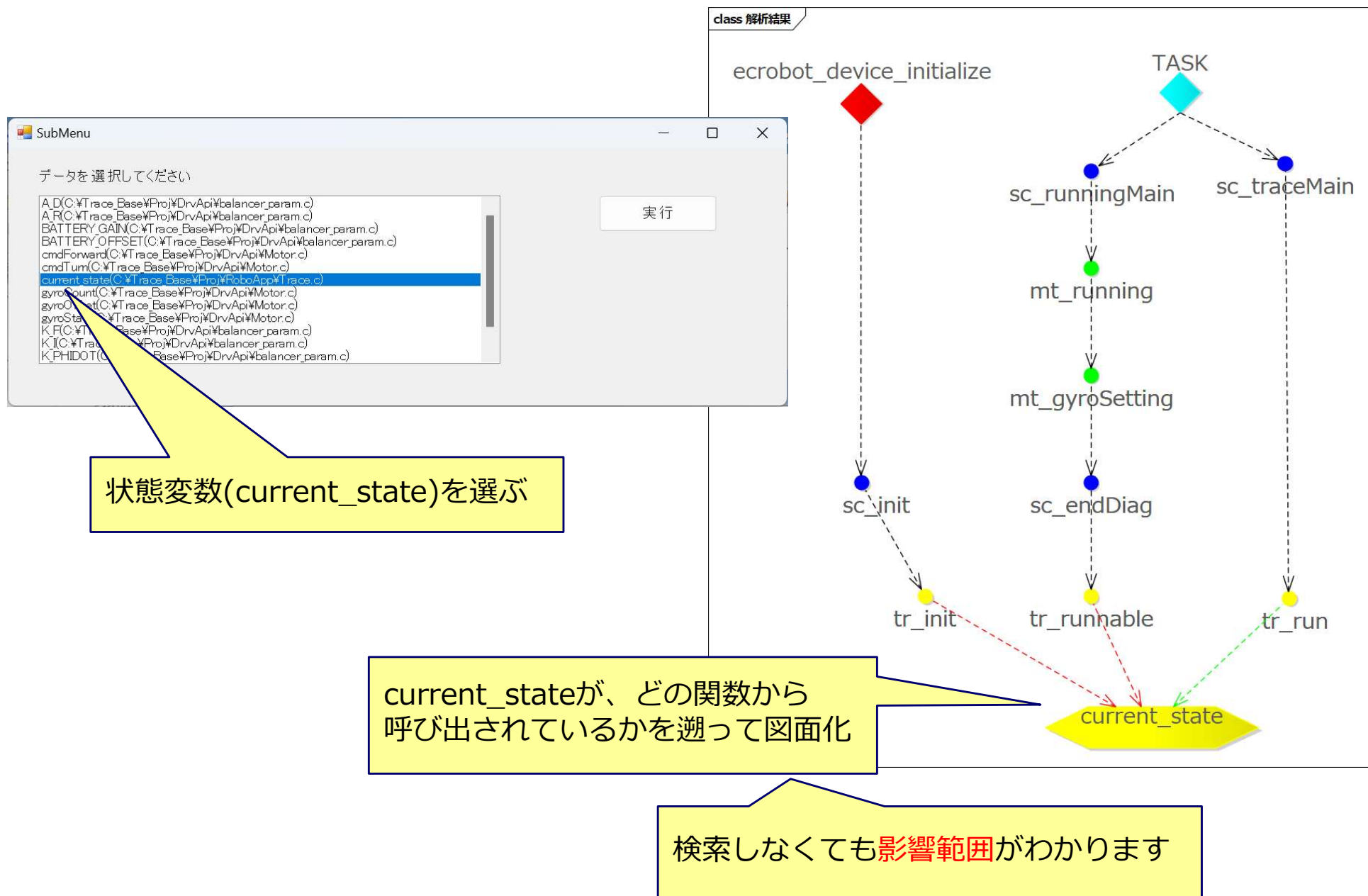
関数構造図（関数起点） 入れ子自動



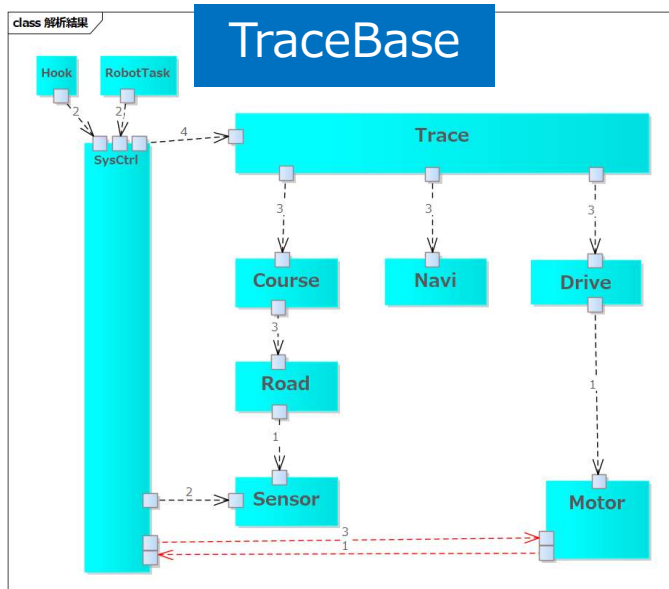
上位から下位のサービスが
利用されている（単方向）

変数はファイル内
にカプセル化

関数構造図 (変数起点)

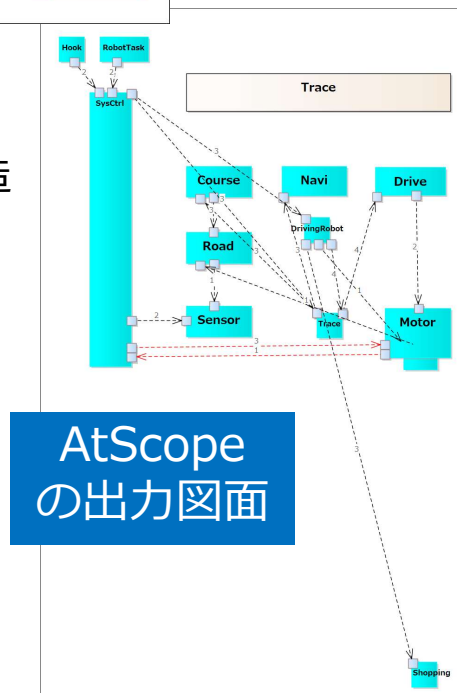


Shoppingの図面化

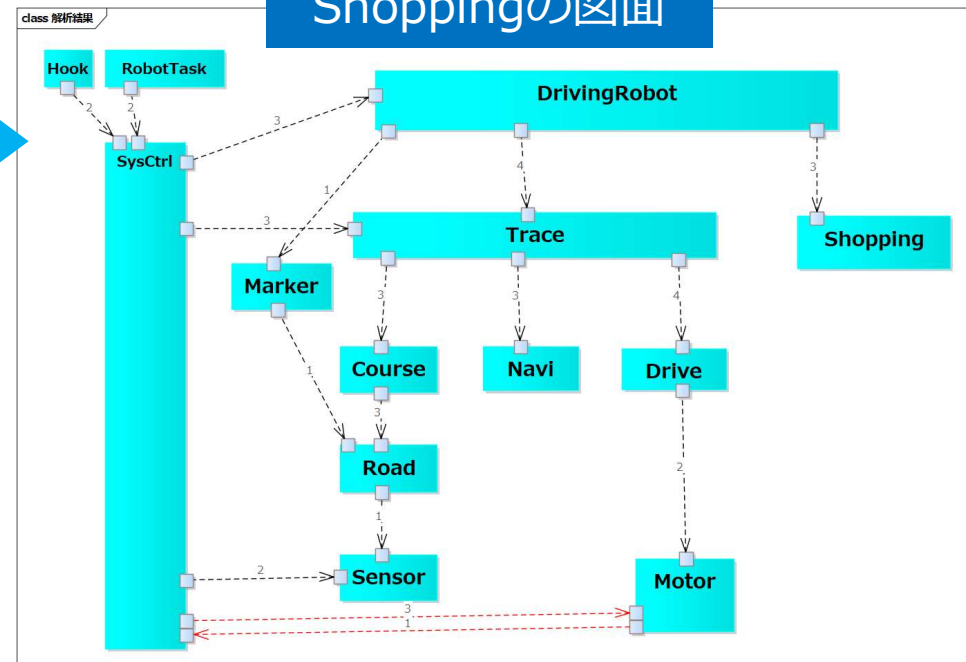


配置を保存
をチェック

TraceBaseのファイル構造
図を開いて
「配置を保存」で
Shoppingを図面化する



配置を整えた
Shoppingの図面



TraceBaseからShoppingへの設計変更

- 新たな責務を持つファイルを追加
 - 関数内部にアドホックに処理をついかない

