本スライドは、当日のセミナー資料の一部を抜粋したものです。

UMLとSysMLの違いわかりますか？

～ETロボコンのチャンピオンが語る
開発目線でのモデル記法の考え方・使い分け方～

株式会社デンソー
情報通信技術2部　ソフト開発2室
酒井英子

2.1　こんなことを感じたことは？

- 学習後、いざモデルを作成しようとすると、
  何をどう書けば良いかわからない
- モデルを作ってみたが、
  モデル図をどう使うのかわからない
- モデルを使うと良いと聞いているが、
  何が良くなるのかがわからない

モデルについて誤解がある…

2.2 モデルを使う上での注意

- 定義されているのは表記（フォーマット）のルールのみ
  - ルールに従って記載するだけで、適切な図（モデル）ができるわけではない

モデリングは、
表現したい事を表記ルールを使って図にする

どんな図になるかはモデルを作る人に依存する

4.9 提出モデルでの参考例(過去のモデル)

■目標(コンセプト):コース全域(難所を含む)を安全に走破する

〔Ⅰ〕コンセプト実現の為の要件分析

※コンセプトからコース全域の安全な(=安定した)走行に必要な要件を抽出

全ての難所を安全に走らせる為の走らせ方と、走らせ方以外に必要な要件を抽出

安全に(=安定して)走らせるための走行性能の要件(ピンクの文字)

ソフトウェアの設計品質の要件(水色の文字)

まとめる

やりたい事のために考えなくてはならないことをざっくり抽出(全般的に把握)

〔Ⅱ〕非機能要件の分析

※〔Ⅰ〕の要件分析より、安定して走行させる為の非機能要件を『走行性能の要件』と『ソフトウェア設計品質の要件』の両面からSysMLを使って分析

分析の視点

ソフトウェア設計品質面の分析

走行性能面の分析

拡大

『安全に』関する要件を網羅的に詳細な要件まで落とし込む

※要件を徐々に詳細化し、制約・要件を抽出

要件詳細化

〔Ⅲ〕機能要件

※〔Ⅰ〕の要件分析で抽出した機能要件に、非機能要件の分析結果から機能要件に関するものを反映する

<制約・要件への対応方針>

① 機能追加	→ P.1
② シナリオの代替・例外フローの検討	→ P.1
③ タスクの設計	→ P.3
④ アーキテクチャ設計	→ P.2
⑤ コーディング規約	→ P.5
⑥ 命名規約	→ P.5
⑦ 開発環境	→ P.4

①,②に対応

異常ケースを検討し、対策を代替フローや例外フローに織り込む

シナリオ

機能や設計方針のどこで対応するかを決める

<特徴>
非機能要件をいかに漏れなく抽出して機能や設計方針に落とし込むかに注力している。

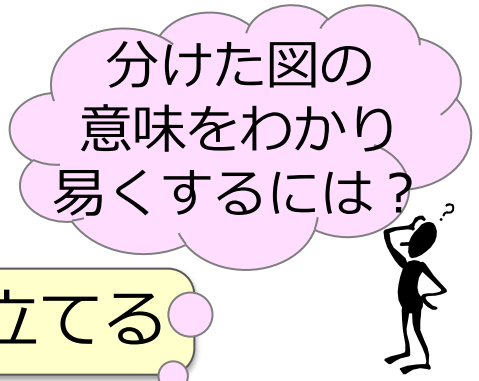
最終的な機能をまとめる

制約・要件への対応方針

対応すべき制約・要件

5.1 事例 ~どのような設計をするかを考える~

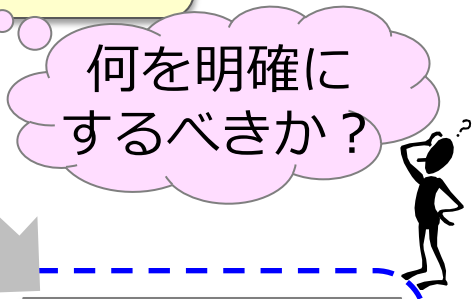
ソフトウェアアーキテクチャの設計を
どのように検討するかを考える



＜設計の大まかなステップの例＞

システムへの要件を踏まえて設計方針を立てる

設計方針に合わせて詳細な設計手順を決める



設計手順に沿って設計する

- 設計方針
- ソフトウェア 全体構造（構造図・振舞い図）
- ソフトウェア 詳細構造（構造図・振舞い図）
- その他（部品組合せetc）

※実装を考える時に必要な成果物

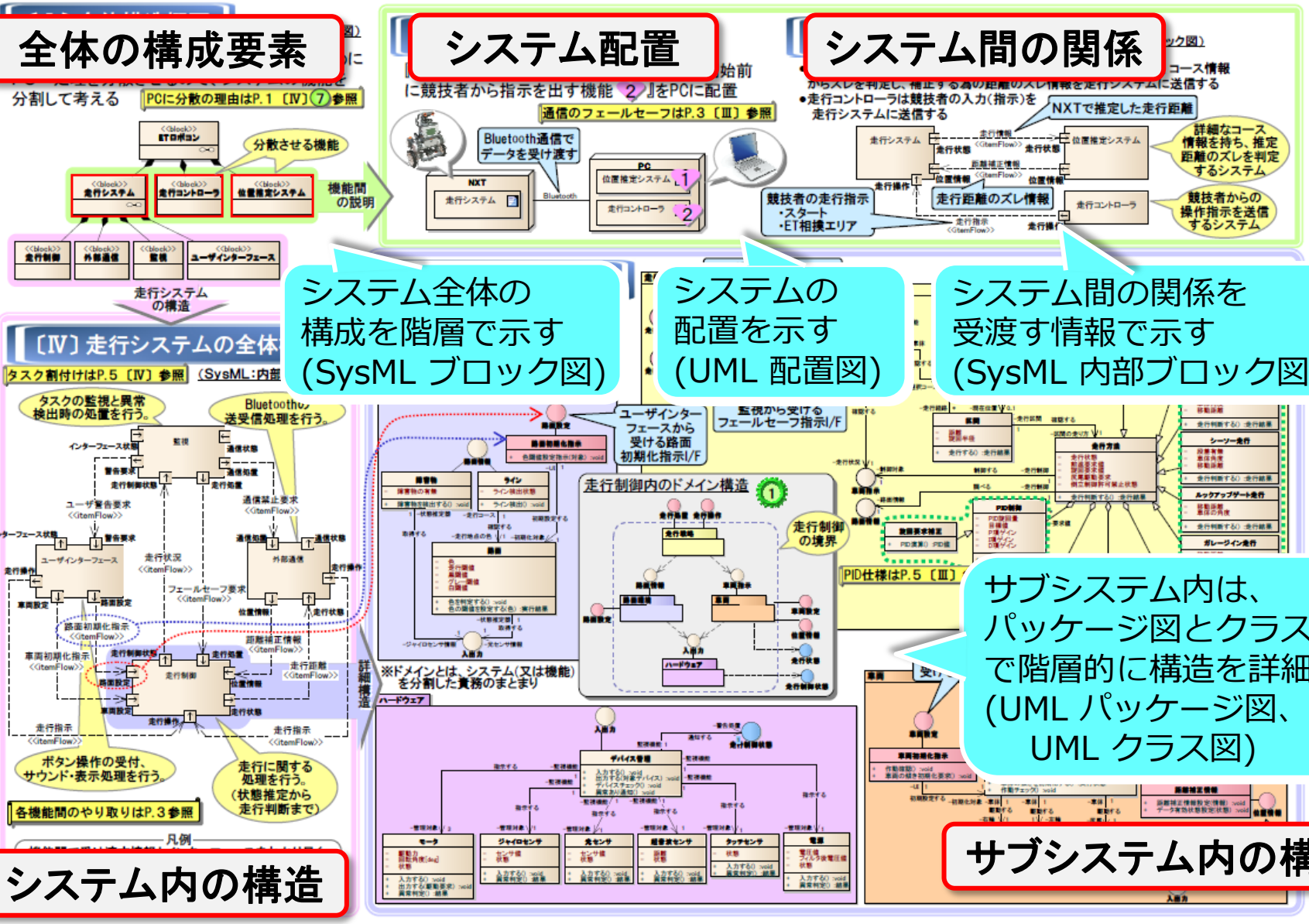
設計の結果

5.9 事例 階層構造の図の使い分け例

全体の構成要素

システム配置

システム間の関係



システム内の構造

サブシステム内の構造