

e-コックピット時代のHMI開発環境

～ HMIの振舞い設計は状態遷移/画面遷移が重要 ～

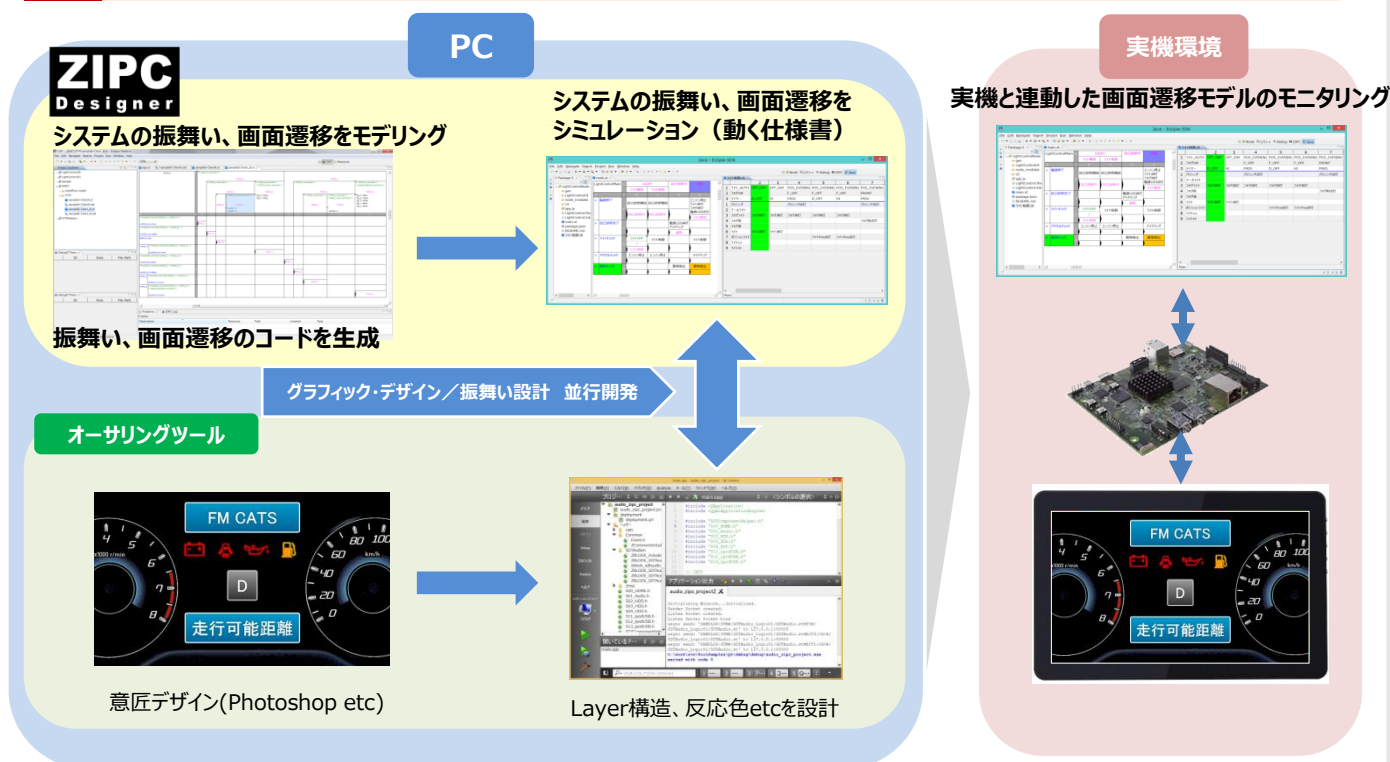
こんな状況はありませんか？

- ・デザイナー作成のグラフィックが制約される。
- ・振舞いの仕様が確定しない。
- ・実装後の手戻りが多く工数、コストが増大する。
- ・より複雑な条件でのグラフィック表示の制御が難しい。

本HMI開発環境で期待される効果

- ・実装開発を並行で進められ、リッチな画面を実現できます。
- ・動く仕様書で明確な振舞いを早期に定義、確定できます。
- ・振舞いのモデルからソースコードを生成し手戻りが減ります。
- ・状態遷移/画面遷移モデル、決定表で複雑な制御も可能です。

グラフィック・デザインと画面遷移設計を並行して開発できる



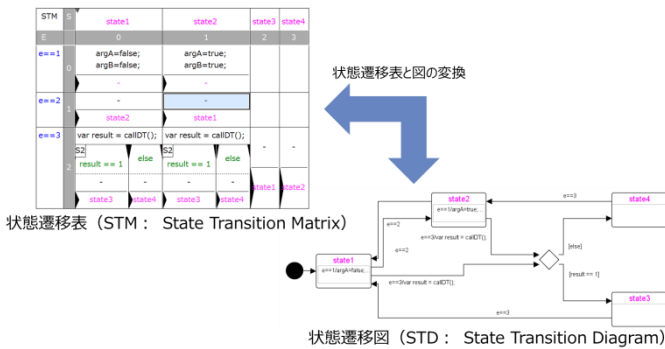
本環境でのHMI開発の流れ

- 1 並行開発でムダがない**
「グラフィック・デザイン」とHMIの「振舞い、画面遷移」のモデリングを並行して実施できる。
- 2 動く仕様書で素早く確認**
グラフィック構造と制御モデルのコードをコンパイルし、PCで実装前に「動く仕様書」を動作確認できる。
- 3 手戻りがなく生産性向上**
動作が正しいと確認されたコードを実装し実機確認するので手戻りが無くなり、生産性が向上する。

e-コックピット時代のHMI開発環境

HMIの振舞い設計は ZIPC Designer で！

特長①：状態（画面）遷移設計



- あいまいな仕様の定義を明確に表現できます。
- 複雑な振舞いも状態と事象の組合せで網羅します。
- 状態モデルがそのまま設計書として資産化できます。



特長②：決定表



- ISO準拠の決定表エディタです。
- 複雑な条件の組合せを整理、定義できます。

条件記述部		条件指定部			
条件部	動作部	1	2	3	4
1	argA	OFF	ON	OFF	ON
2	argB	OFF	OFF	ON	ON
1	SetMessage	setMsg('1')	setMsg('2')	setMsg('3')	setMsg('4')
2	SetEvent	e=1;	e=2;	e=3;	e=4;
3	CallSTM	stm_eventCall();	stm_eventCall();	stm_eventCall();	stm_eventCall();

動作記述部

動作指定部

argA = OFF && argB = OFFの場合
①setMsg('1')処理を実行
②e=1処理を実行
③stm_eventCall();を実行

特長③：様々なオーサリングツールとの連携



- 主要なオーサリングツールとの連携ができます。

これらの特長により HMI開発の生産性、品質が飛躍的に向上します

