トップエスイー修了制作

並行性設計における不具合検出のためのICONIXの拡張

日立ソリューションズ東日本　國松 冬樹　fuyuki.kunimatsu.wg@hitachi-solutions.com

ICONIXにおける問題点

ICONIXでは，複数ユースケース並行実行時にシステムの振る舞いが仕様を満たすか分からない．そこでICONIXでは，設計段階にて並行性を考慮した設計を行う(並行性設計)．しかし，明確な手順がないため不具合が混入し得る可能性があり，その不具合を設計段階で検出することは難しい．

手順拡張の提案による解決

ICONIXの設計段階で，並行性設計の不具合を検出するための手順拡張を行う．具体的には，モデル検査技術SPINを導入し，モデル検査の実施に必要となるステートマシン図とPromelaコードの作成手順を提案する．これにより，ICONIXの設計段階で並行性設計の不具合を検出することが可能となる．

提案する設計プロセス

ドメイン分析 → 要求分析 → システム分析 → **設計** → 実装

アーキテクチャ設計 → **ステートマシン図作成** → **Promelaコード作成** → **モデル検査実施**

既存手順

入力
ロバストネス図
分析クラス図

出力
シーケンス図

並行性設計

拡張手順

手順提案

入力
ロバストネス図
ユースケース記述
分析クラス図
シーケンス図

出力
ステートマシン図

入力
ステートマシン図

出力
Promelaコード

手順提案

入力
Promelaコード

出力
モデル検査結果

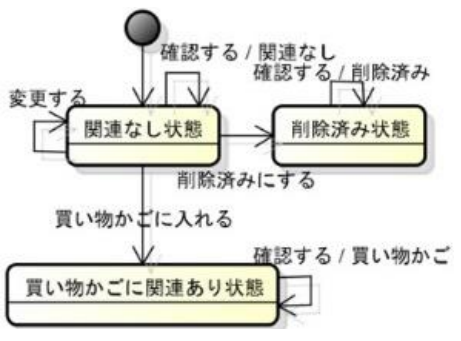
反例検出

ロバストネス図活用
状態抽出　オブジェクト間関連特定

ICONIXの特徴

提案手順適用結果

ステートマシン図



Promelaコード

```
/* GOODS Process */
proctype GOODS(){
  do
    :: (Goods_state == noRelation) ->
       Goods_ch?Goods_event;
       if
         :: (Goods_event == checkGoodsReq_mgp) ->
            ManagementGoodsProcess_ch!noRelation;
                              ⋮
```

モデル検査実施

反例あり

```
pa                                    s
To replay the error-trail, goto
Simulate/Replay and select "Run"
```

OR

反例なし

```
No errors found -- did you verif
y all claims?
```