Webアプリケーションの クラス設計仕様に対する モデル化と検証

独立行政法人 産業技術総合研究所
システム検証研究センター
崔 銀惠　渡邊 宏
http://unit.aist.go.jp/cvs/

JaSST'05　2005.1.24

背景と目的

- Webアプリケーションの普及に伴ってその品質保証のための検証技術の重要性はますます高まってきた.
- 本研究では，オブジェクト指向のWebアプリケーション設計における**クラス設計仕様に注目したモデル化と検証法**を提案する.
  - クラス仕様：クラス図，メソッド仕様書
  - クラス仕様の検証
    → 実装に最も近い仕様を検査し，実装前に不具合発見
  - ソフトウェアの設計段階の検証
    → 実装後の不具合改修による手戻りを削減

設計仕様間の整合性

Webアプリケーションの設計仕様

上位仕様　画面遷移仕様

動作仕様：
アクティビティ図，
フローチャートなど…

下位仕様

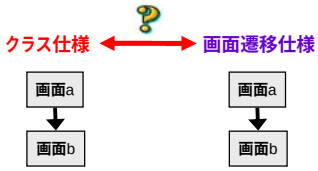
クラス仕様：
クラス図，メソッド仕様

問題点
- ✓ クラス仕様が画面遷移仕様や動作仕様を満たしていない.
- ✓ クラス仕様の変更を画面遷移仕様や動作仕様へフィードバックしていない.

クラス仕様と他の設計仕様間の整合性検査が必要

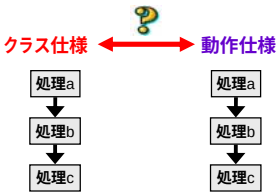
クラス仕様と画面遷移仕様の整合性とは

- クラス仕様と画面遷移仕様が整合しているとは，
  - 条件1. 画面遷移仕様の画面遷移がクラス仕様にある.
  - 条件2. クラス仕様の画面遷移が画面遷移仕様にある.



クラス仕様と動作仕様の整合性とは

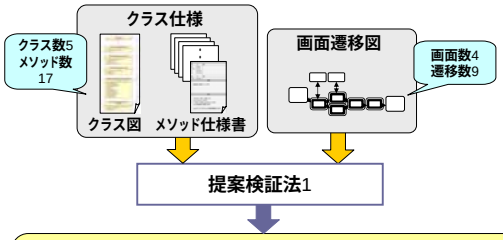
- クラス仕様と動作仕様が整合しているとは，
  - 条件1. 動作仕様の処理の流れがクラス仕様にある.
  - 条件2. クラス仕様の処理の流れが動作仕様にある.



研究の概要

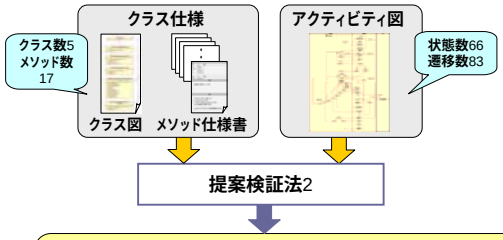
- クラス仕様に対する2つの検証法を提案
  - 検証1: クラス仕様 vs. 画面遷移仕様 の整合性検査
  - 検証2: クラス仕様 vs. 動作仕様 の整合性検査
- ある企業のWebアプリケーション設計開発で用いられた設計仕様に提案法を適用
  - Java+Strutsで開発した業務処理アプリケーション
  - 数十個のモジュールの内の1つに関する設計仕様を検査
    - 上位仕様：画面遷移図，UMLアクティビティ図
    - 下位仕様：クラス図，メソッド仕様書

実験結果1 - クラス仕様と画面遷移図の整合性



検査結果：画面遷移に関する不整合を発見
- ✓ 画面遷移図の画面Aから画面Bへの遷移がクラス仕様にない．
- ✓ クラス仕様の画面Cから画面Dへの遷移が画面遷移図にない．

実験結果2 - クラス仕様とアクティビティ図の整合性

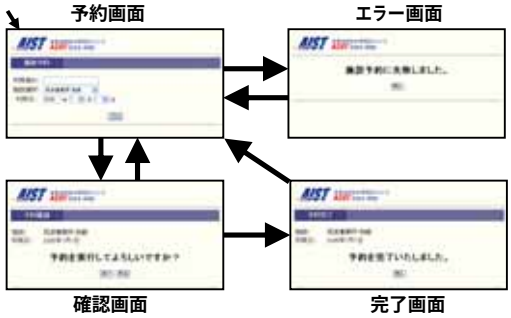


検査結果：両仕様間の不整合を発見
- ✓ クラス仕様にだけエラー画面へ遷移するところがある．
- ✓ 両仕様でボタン押下時の処理が異なる．

発表のながれ

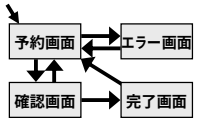
- Webアプリケーションと設計仕様の例
- 検証法1: クラス仕様 vs. 画面遷移仕様 の整合性検査
- 検証法2: クラス仕様 vs. 動作仕様 の整合性検査
- まとめ

Webアプリケーションの例

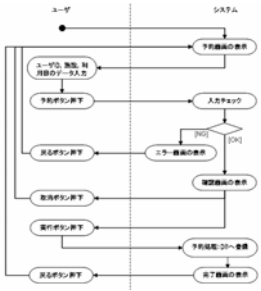


画面遷移図，アクティビティ図

画面遷移図

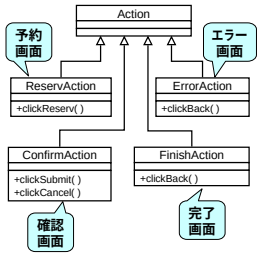


アクティビティ図



クラス仕様

クラス図

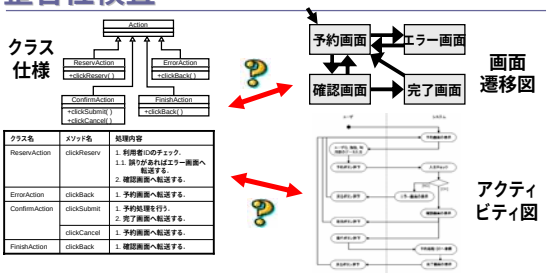


メソッド仕様書

予約ボタン押下時の処理

クラス名	メソッド名	処理内容
ReservAction	clickReserv	1. 利用者IDのチェックを行う． 1.1. 誤りがあればエラー画面へ転送する． 2. 確認画面へ転送する．
ErrorAction	clickBack	1. 予約画面へ転送する．
ConfirmAction	clickSubmit	1. 予約処理を行う． 2. 完了画面へ転送する．
	clickCancel	1. 予約画面へ転送する．
FinishAction	clickBack	1. 確認画面へ転送する．

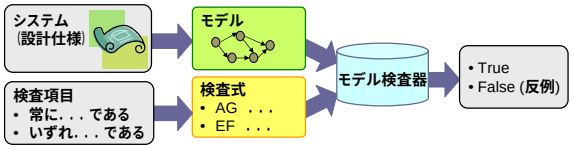
整合性検査



クラス名	メソッド名	処理内容
ReservAction	clickReserv	1. 利用者IDのチェック. 1.1. 誤りがあればエラー画面へ転送する. 2. 確認画面へ転送する.
ErrorAction	clickBack	1. 予約画面へ転送する.
ConfirmAction	clickSubmit	1. 予約処理を行う. 2. 完了画面へ転送する.
	clickCancel	1. 予約画面へ転送する.
FinishAction	clickBack	1. 確認画面へ転送する.

- クラス仕様と画面遷移仕様，動作仕様の整合性を網羅的に検査することは難しい．(人手によるレビューでは見落としやすい．)
- 本研究では，モデル検査を用いた検証法を提案する．

モデル検査



- 有限状態遷移系としてモデル化したシステムが時相論理の検査式を満たすかを全状態探索で網羅的に検査
- 既存のモデル検査器(UPPAAL, SMV, SPIN...)を使用して全自動的な検査が可能
- 本研究では，UPPAALを使用
  - モデル：オートマトンの集まり，検査式：CTLの一部

発表のながれ

- Webアプリケーションと設計仕様の例
- 検証法1: クラス仕様 vs. 画面遷移仕様 の整合性検査
- 検証法2: クラス仕様 vs. 動作仕様 の整合性検査
- まとめ

モデル作成：クラス仕様のモデル化

- クラス図とメソッド仕様書から，クラスの動的な振舞いを表すモデル(クラスモデル)を次の手順で作成する．

Step 1. 個々のクラスのモデル化
- 各クラスに対して，メソッド仕様書に記述された処理の流れを遷移システムとしてモデル化

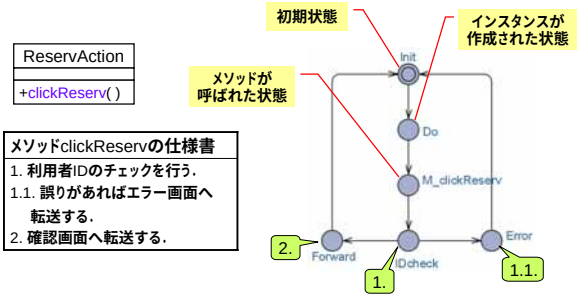
Step 2. モデルの合成
- Step1で作成した遷移システムのクラス連携部分の遷移に同期のラベルをつけて並列合成

クラス呼び出し関係

- クラスモデルはクラス全体の振舞いを模擬する．

モデル作成例 — 1.個々のクラスのモデル化

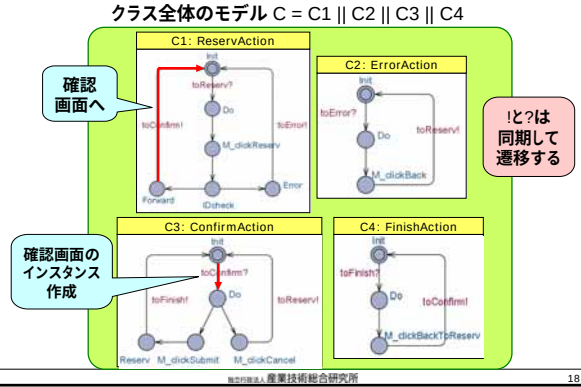
- 例：ReservActionクラスのモデル



メソッドclickReservの仕様書
1. 利用者IDのチェックを行う．
1.1. 誤りがあればエラー画面へ転送する．
2. 確認画面へ転送する．

モデル作成例 — 2.モデルの合成

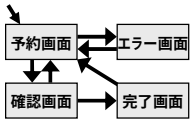
クラス全体のモデル C = C1 || C2 || C3 || C4



検査式作成

- 画面遷移仕様から検査式を作成する．

検査式の作成例



画面遷移図

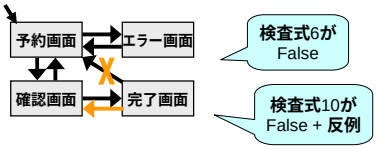
検査項目	検査式
画面遷移図の画面遷移がクラスモデルにある	1. EF (予約画面 ∨ EX エラー画面) ... 6. EF (完了画面 ∨ EX 予約画面)
クラスモデルの画面遷移が画面遷移図にある	7. AG (予約画面 → AX (予約画面 ∨ エラー画面 ∨ 確認画面)) ... 10. AG (完了画面 → AX (完了画面 ∨ 予約画面))

モデル検査結果の例

- クラスモデルで作成した検査式が成り立つことを検査する．

	検査式	検査結果
6	EF (完了画面 ∨ EX 予約画面)	False
10	AG (完了画面 → AX (完了画面 ∨ 予約画面))	False

クラス仕様と画面遷移図の不整合発見



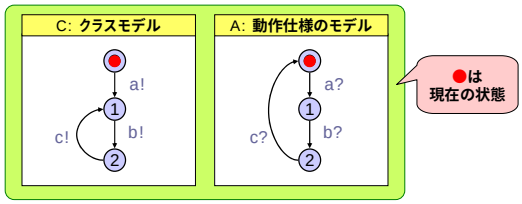
発表のながれ

- Webアプリケーションと設計仕様の例
- 検証法1: クラス仕様 vs. 画面遷移仕様 の整合性検査
- 検証法2: クラス仕様 vs. 動作仕様 の整合性検査
- まとめ

クラス仕様と動作仕様の整合性検査法

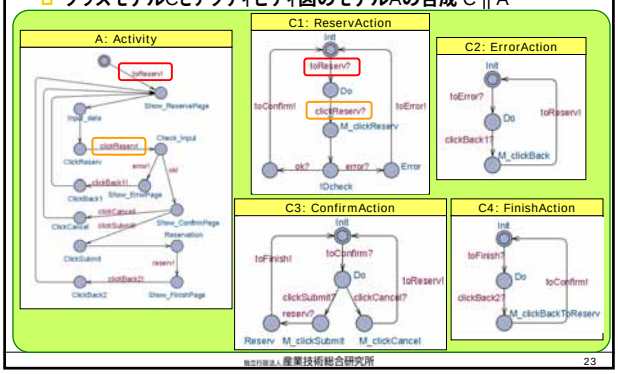
- クラスモデルCと動作仕様のモデルAを合成してモデルC||Aを作る．
- モデルC||Aでデッドロック検査を行う．
  - もしデッドロックがあれば不整合が見つかる．

例
C: 初期状態 → 処理1 → 処理2 → 処理1
A: 初期状態 → 処理1 → 処理2 → 初期状態



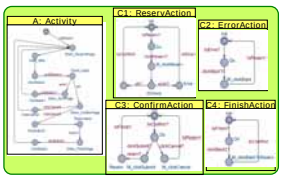
モデル作成例

- クラスモデルCとアクティビティ図のモデルAの合成 C || A



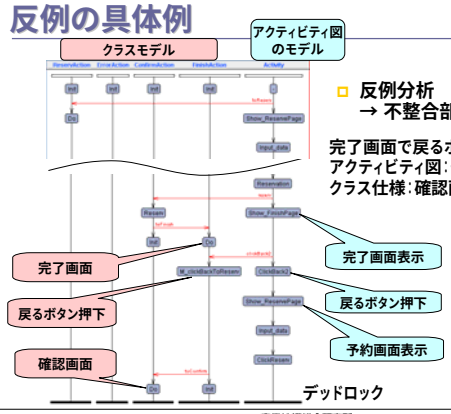
モデル検査例

- 入力
  - モデル:



  - 検査式: AG(not deadlock) --- デッドロックがない
- 検査結果: False ＋ 反例

反例の具体例



- 反例分析
  → 不整合部分特定

完了画面で戻るボタン押下時，
アクティビティ図：予約画面を表示
クラス仕様：確認画面へ転送

発表のながれ

- Webアプリケーションと設計仕様の例
- 検証法1: クラス仕様 vs. 画面遷移仕様 の整合性検査
- 検証法2: クラス仕様 vs. 動作仕様 の整合性検査
- まとめ

まとめ

- Webアプリケーションのクラス設計仕様に対するモデル化と検証法を提案した．
  - クラス図とメソッド仕様書からクラスモデルを作成し，
  - クラスモデルとモデル検査法を用いて，1) クラス仕様と画面遷移仕様の整合性検査，2) クラス仕様と動作仕様の整合性検査を行う．
- 実際のWebアプリケーションの設計仕様に対して適用実験を行い，いくつかの不具合を発見した．
- 今後の課題
  - データフローを考慮した整合性検査
  - ソースコードの検証

システム検証研究センターの研究活動

① システム検証技術の基礎研究
- 抽象化技法
- モデル検査技法
- 定理証明技法

パートナー募集中！

② システム検証の「フィールドワーク」
- 企業との共同による事例研究
- 技術者向けモデル検査研修コースの開発・実施

受講者募集中♪

システム検証研究センター

- ホームページ　http://unit.aist.go.jp/cvs/
- Eメイル　informatics-inquiry@m.aist.go.jp
- 活動拠点
  - 千里中央（大阪府豊中市）
  - 尼崎（兵庫県尼崎市）
  - つくば
  - 臨海副都心（お台場）