正確な記述や知識ポインターは「報告書」をご覧下さい！

形式手法の紹介
演習コースIIの紹介

副主査　石川 冬樹
国立情報学研究所 コンテンツ科学研究系 准教授
@fyufyu / f-ishikawa@nii.ac.jp

ごく簡単な「検証」

- 例: 定員付イベントへの参加予約機能

```
reserve(User u, Event e){
   予約操作
}
```

「イベントeをユーザuが予約している
という情報が登録される」
（結果，イベントeの予約済み人数が1増える）

操作実行後に必ず成り立つのだと保証したいこと
「イベントeの予約済み人数は，イベントeの定員以下」

保証できている？

ごく簡単な「検証」

- 例：定員付イベントへの参加予約機能

呼び出し元の責任で保証すべきこと
（入力チェックすべきこと，でもある）
「イベントeの予約済み人数は，イベントeの定員未満」

```
reserve(User u, Event e){
   予約操作
}
```

「イベントeをユーザuが予約している
という情報が登録される」
（結果，イベントeの予約済み人数が1増える）

操作実行後に必ず成り立つのだと保証したいこと
「イベントeの予約済み人数は，イベントeの定員以下」

ごく簡単な「検証」

- 例：定員付イベントへの参加予約機能

呼び出し元の責任で保証すべきこと
（入力チェックすべきこと，でもある）
「イベントeの予約済み人数は，イベントeの定員未満」

```
reserve(User u, Event e){
   予約操作
}
```

「イベントeをユーザuが予約している
という情報が登録される」
（結果，イベントeの予約済み人数が1増える）

操作実行後に必ず成り立つのだと保証したいこと
「イベントeの予約済み人数は，イベントeの定員以下」

保証できている！（ことが理屈で説明できる！）

今の話を「形式」とやらにすると

まずは形式「記述」

- 例: 定員付イベントへの参加予約機能

pre e.registered < e.capacity

呼び出し元の責任で保証すべきこと
（入力チェックすべきこと，でもある）
「イベントeの予約済み人数 ＜ イベントeの定員」

```
reserve(User u, Event e){
   予約操作
}
```

e.registered := e.registered ∪ { u }

「イベントeをユーザuが予約している」
という情報が登録される
（結果，イベントeの予約済み人数が1増える）

操作実行後に必ず成り立つのだと保証したいこと
「イベントeの予約済み人数 <= イベントeの定員」

post e.registered <= e.capacity

まずは形式「記述」

- 厳密に文法・意味が定義された言語で記述
  - 例：「実行後に必ず成り立つこと」を書く欄がある
    - その保証に十分であるかという観点から，入力チェックの内容を系統的に導く・検査するというノウハウ
    - 「定員オーバーさせないこと」と書いていない仕様書に対し，新たな予約操作を追加する派生があったら・・・
  - 例：「・・・を満たすものをすべて抜き出す」，「・・・までは・・・である」といった語彙が典型として用意されている
    - プログラマが “or” や “xor” を典型として活用し，日本語の「または」の曖昧さに注意するのと同様

皆さんの日常から遠い，別の世界の話でしょうか？

ところでここまでの話って・・・

- プログラムの話？
  - Howの話（RDBならでは，Javaならでは，といった話）は一切していない
  - この「誤り」（必要な制約の抜け）が発生し埋め込まれるのは，いつ？
    （その時点で検出されるべき）

ところでここまでの話って・・・

- プログラムの話？
  - Howの話（RDBならでは，Javaならでは，といった話）は一切していない
  - この「誤り」（必要な制約の抜け）が発生し埋め込まれるのは，いつ？
    （その時点で検出されるべき）

コース名「形式手法と仕様記述」

先の例では「形式仕様記述」の言語を利用

形式「手法」の中身は多種多様

- 厳密な記述を様々な作業に活用する

  ツール化・自動化されている，人手でもぶれにくい

  – 動かしてみる（シミュレーション上の各種テスト）
  – 「ユーザ1」など名前を適当に付けて，データや実行列の例をいろいろ洗い出してみて確認する
  – （先の例のように）理屈による保証（証明）を行う
  – 起きうる状態変化を網羅的に探索し検証する
  – アサーションや，入力などの因子やその境界値などを抜き出し，テストケースを生成する
  – ・・・

  皆さんの日常から遠い，別の世界の話でしょうか？

演習コースⅡの進め方

まずは全員，手法の一つ（VDM）を学習・演習

- 「学習」組（さらにグループに分かれる）
  - 自分たちの問題意識にあった手法・ツールを選んで勉強し，自分たちのアウトプットも時間上可能な範囲で検討する

「勉強する」が目的で身につくなら
皆さん英語ペラペラでしょう

- 「研究」組（今年はたまたま1グループ）

※ 前述のような議論の「横展開」は常に発生

取り組み全体像

「学習」組

形式仕様基礎（VDM） 「動かしてみる」～共通基礎学習	小田部
モデル検査（NuSMV） 「網羅的に探索，検証する」～教育コース作成	有賀・羽田・和田
実時間モデル検査（UPPAAL） 「時間制約を・・・（同上）」～入門例題集作成	平野
モデル発見（Alloy） 「例を洗い出す」～テスト生成問題に試用	水野

「研究」組

「日常」（USDM）との融合を模索する	日下部・宮本