トップエスイー修了制作
VDM++ → Event-B変換

川俣 洋次郎
東京大学

発表の流れ

- 形式仕様記述とは
- 手法間のトレードオフ
  - 記述・習得のしやすさ vs 検証の厳密さ
- 変換を用いた解決
- 検証実験
- 結果と考察
- まとめ

形式仕様記述

- 「ソフトウェアの静的構造に関する仕様を形式的な言語を用いて記述・検証する手法」
  - 厳密な意味論、言語構造

- 何が嬉しい？
  - 仕様の曖昧さを排除
    - 曖昧な仕様 → 下流における仕様上のトラブルの原因
  - 開発初期におけるツールを用いた検証支援
    - 下流で仕様の不整合を発見 → 多大な修正コスト
    - 人手による仕様の整合性検証 → 見落としリスク

講座に登場した形式仕様記述言語

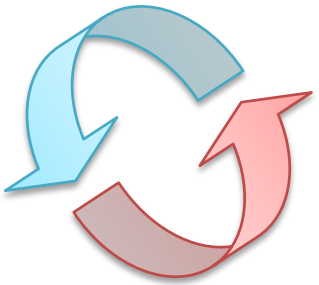
	VDM-SL	VDM++	Z	B	Event-B
用いたツール	VDM-SL Toolbox	VDM++ Toolbox	Z/EVES	Click'n'prove / B4free	RODIN
検証方法	テスト	テスト	証明	証明	証明
詳細化	陰/陽仕様	陰/陽仕様	Data refinement	段階的詳細化	段階的詳細化
記述のしやすさ	○	○	△	△	△
オープンなツール	×	Overture	CZT	Click'n'prove	RODIN

書きやすいが
検証の厳密さに難あり

検証は厳密だが
書きやすさに難あり

変換によるいいとこ取り

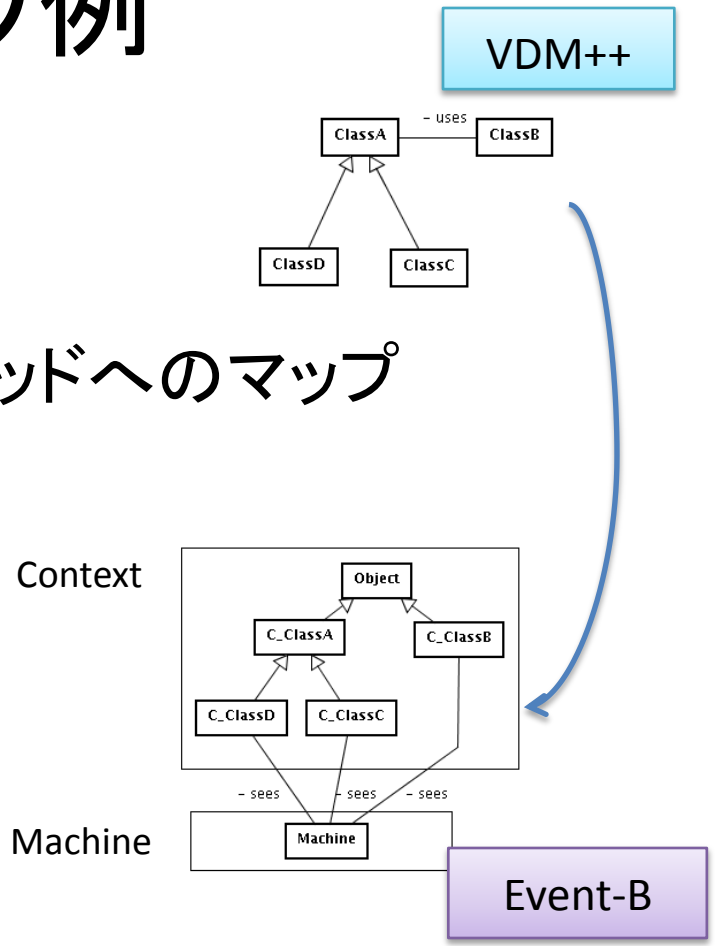
- VDM++で記述 → （変換）→ Event-Bで検証
  - 書きやすく、かつ厳密な検証が可能に
- 問題点：言語間のギャップ
  - 様々な表現が可能なVDM++
  - 利用できる構文が限定的なEvent-B



- 言語の理論的基礎は同じ（集合論と述語論理）
  → 本質的には同等の表現が可能なはず
  → **表現ノウハウ**の導入による解決

表現ノウハウ例

- オブジェクト指向
  - インスタンス集合の包含関係
  - インスタンスからフィールド・メソッドへのマップ
- 返り値、メソッド呼び出し
  - 関数型のアクセサ変数
- 事後条件
  - 恒真定数による検証の代替
- 型
  - ライブラリの提供
  - 基本型の組み合わせによる表現



その他もろもろ・・・

変換の適用範囲

- VDM++：**What**の記述 ＋ **How**の記述（251個）
  - **What**：静的構造の仕様（204個）
  - **How**　：振る舞いの仕様（47個）

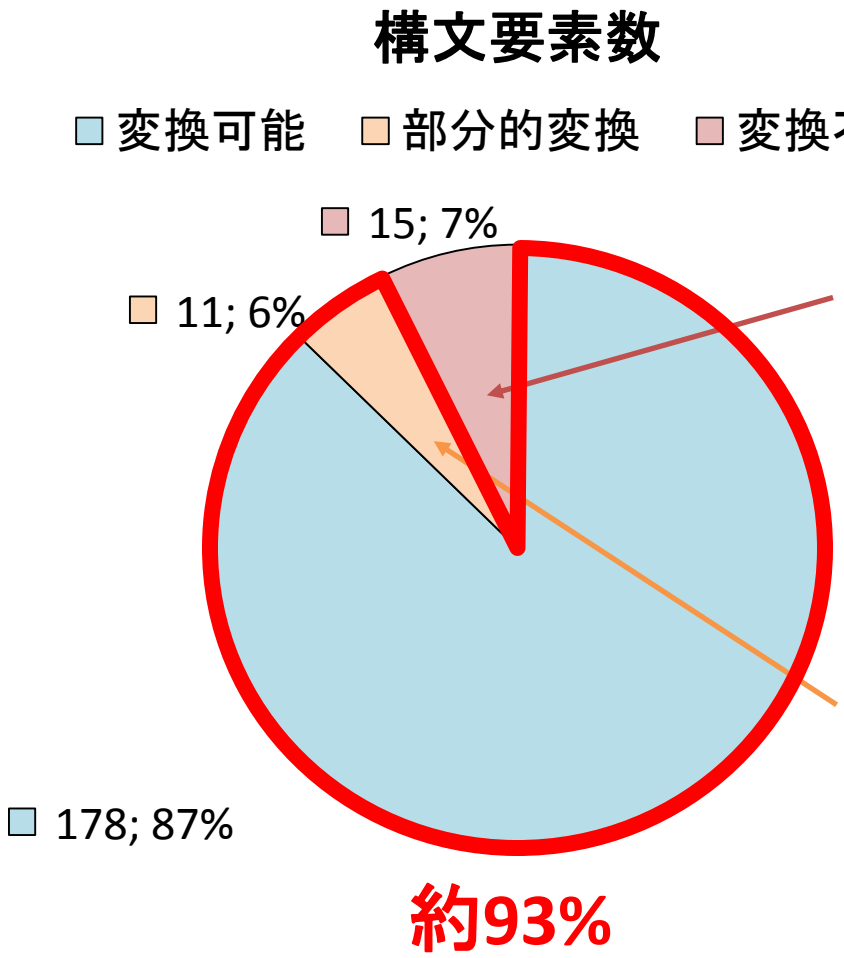
クラス、内部変数
操作、関数、条件
etc…

実際の処理記述
スレッド、例外処理
etc…

- Event-B：**What**の記述のみ
  - **How**の記述は表現ノウハウではカバーしきれない

→ **What**の記述のみを変換対象

変換率



- 型 (実数型、有理数型、選択型)
- 演算子 (整数部分)
- iota式
- etc…

型の制約
- 型の違う値を同じ集合に代入できない
- 特定の型のみ適用可能な演算子がある

検証実験

95.6%

	VDM++の行数（入力）	Event-Bの行数（出力）	自動証明成功証明責務数	自動証明失敗証明責務数
蔵書管理システム	119	387	95	**4**
ファイルシステム	104	338	85	**11**
ワークフローシステム	136	473	145	0

- 記述された条件間の不整合（蔵書管理システム）
  - 削除操作で、引数に指定した本が蔵書内に複数存在する場合を考えていない（1種類の書籍を複数冊扱うかが明示されていない）
- 証明器の実行時間不足、補題不足（ファイルシステム）
  - より抽象的な仕様で証明された証明責務を追加 → 全て証明された

考察

- 不整合の原因の特定方法
  - 証明責務の生成元条件まではトレース可能
  - 「どの条件がどう悪いか」→ レビューが必要
- 検証対象
  - 条件間の不整合のみ（事前・事後条件、不変条件）
  - 実際の処理が条件を遵守しているかは対象外
    → テスティングと併用することが望ましい
- 変換の正当性
  - 厳密には示していない → 今後の課題

まとめ

- 書きやすさと検証の厳密さの両立
  - VDM++ → Event-B変換器による実現
- 結果
  - 証明失敗により仕様の不整合（の可能性）が検出できた
- 現場での使いかた
  - 検証者に「気づき」を与えるツール（？）

ご静聴ありがとうございました