大学共同利用機関法人 情報・システム研究機構
国立情報学研究所
National Institute of Informatics

2013年度ソフトウェア工学分野の先導的研究支援事業

形式仕様とテスト生成の部分的・段階的な活用

～探索を通したコード中心インクリメンタル型開発の支援

情報・システム研究機構
国立情報学研究所（NII）
研究責任者　石川　冬樹

研究概要：要は・・・

- 仕様やテスト設計（の部分的な記述）に対し，例（テストケース）を提示
  - 例： 三角形の判別問題（マイヤーズ）

（部分的に）事前条件や場合分けを記述

```
pre { a >= 0, b >= 0, c >= 0 }
partition "正三角形" { a == b && b == c }
partition "二等辺三角形"
  { a == b && b != c || b == c && …}
partition "不等辺三角形“
  { a != b && b != c }
```

c != a 抜け

例の提示

a	b	c	
3	3	3	正三角形
4	2	1	不等辺三角形
6	4	6	二等辺三角形 不等辺三角形

検証・妥当性確認，フィードバック

- 「仕様」を成果物とするタスクにも，「テストケースリスト」を成果物とするタスクにも使える
- 「人手で・暗黙に」 から「厳密，十分な記述による完璧な自動生成」へ段階的に，その間でもよい

研究概要：動機

現場の課題

仕様や制約の記述・検証
→ 各自思い思いになりがちで
不十分・不正確・不整合，
以降の工程につけがくる

単体・統合テストの設計
→ 比較的単純なのに
不十分・不徹底＆大量の人手対応，
結果のみ残り保守時に再利用難

プログラムやテストケースという
「結果」だけではなく
その「出所」こそ「きちんと」書く
※ ツールにも渡せる厳密な言語で

形式手法

テスト自動生成

仕様を・テスト設計を
厳密に書けば…？？

踏み込んだ課題

- ありうる理想だが現状から一段ジャンプはできない
- 意図に合った必要十分な宣言的記述は難しい
- 自明なこと・対話や経験で共有されていることも記述（形式化・形式知化）する手間が余計に感じる
- 「出所」から「結果」の自動生成は完璧ではなく，結局「結果」もいじることになり手間が重複

アジャイル

相反する
関係ない

でも思想と「優れた多能工に任せる」だけでは…

2013年度ソフトウェア工学分野の先導的研究支援事業

研究概要：目的

- **仕様・テスト設計の部分的な形式記述言語**
  - 必要な仕様や設計を一部与え，十分でなくてよい
  - 具体例（ケース）を与え，一般式を与えなくてよい
- **この記述を入力とするモデル発見ツール（具体例となるテストケースリストの出力）**
  - 記述を少しでも加えたらすぐフィードバック
  - インクリメンタル・反復的な利用

コード以前が軽量
現在一般的
暗黙的，属人的
人の作業が中心

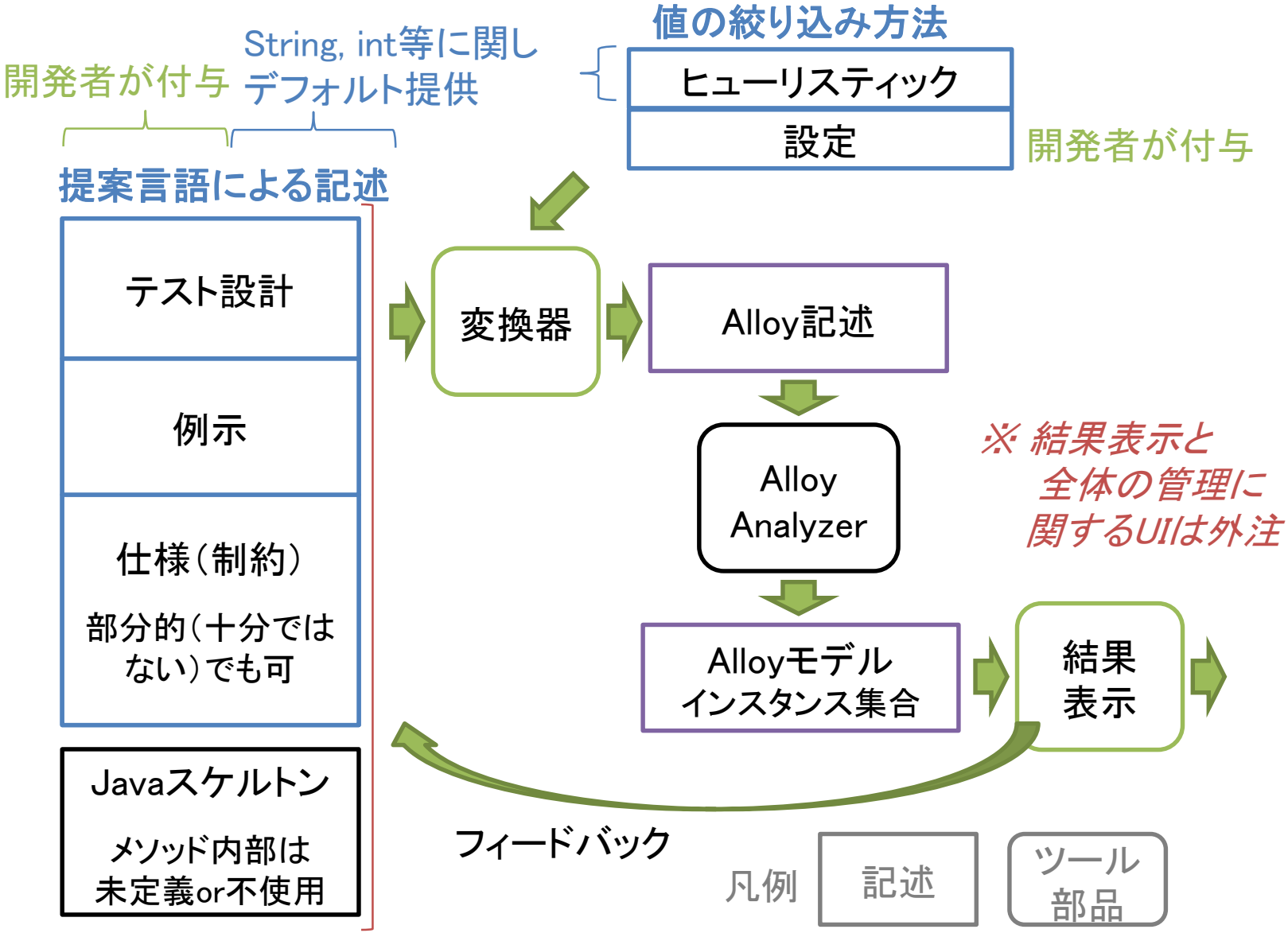
必要・有用なら厳密に書き
ツールを使う
人手・暗黙で十分なら
それが早くそれでよい

コード前の工数
現在敷居が高い
明示的，系統的
様々なツール支援

研究概要：本事業での範囲

- 本事業での範囲
  - Javaスケルトン（変数・メソッドシグネチャ）を対象とした実現手段
    （仕様記述言語や他プログラミング言語への適合は期間内には行わない）
  - Alloy Analyzer（SATソルバー）を内部で用い，比較的単純な論理で実現できるツール機能を実現
    （ソルバー使い分けなどは期間内には行わない）
- 1年目の技術構築の後，2年目はセミナー実施
  - 教育・スキル評価・意識向上の手段としても評価
  - アジャイル，形式手法，品質保証テストの，分断されがちな”Camp”を俯瞰する技術として産業界に提示

研究成果：技術全体像


値の絞り込み方法
String, int等に関し
デフォルト提供
開発者が付与
ヒューリスティック
設定
開発者が付与
提案言語による記述
テスト設計
例示
仕様（制約）
部分的（十分ではない）でも可
Javaスケルトン
メソッド内部は
未定義or不使用
変換器
Alloy記述
Alloy
Analyzer
※ 結果表示と
全体の管理に
関するUIは外注
Alloyモデル
インスタンス集合
結果
表示
フィードバック
凡例
記述
ツール
部品

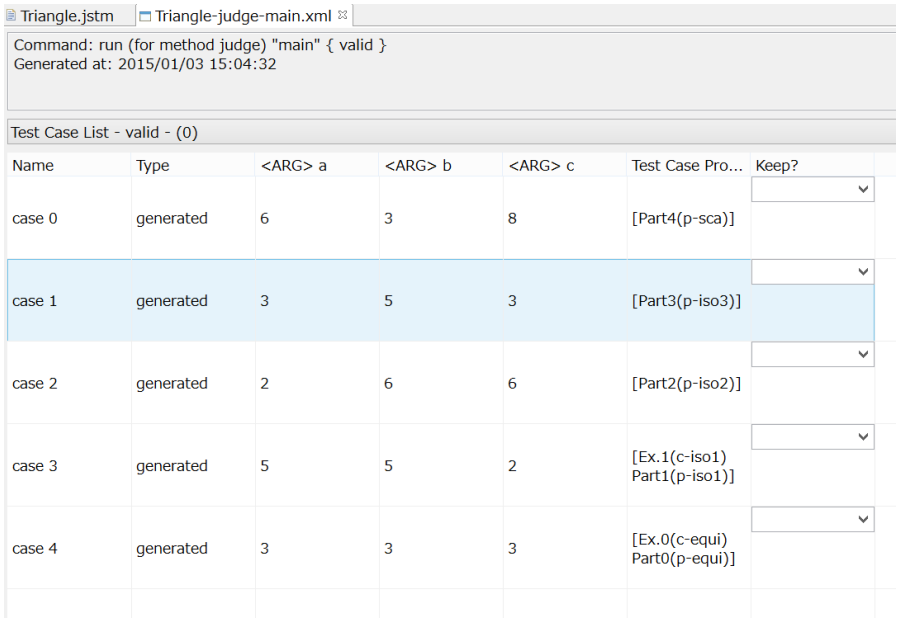
研究成果： 言語

以下3つが記述可能な言語

- 仕様記述
  - 事前条件，事後条件，不変条件
- テスト設計
  - 同値クラス（境界値，ペアワイズなども表現可能）
  - 単純な場合の同値クラス生成（境界値，列挙型やブール型，違反値同士の分離など）
  - テストケース数の最小化あるいは冗長な生成など
- テストケース
  - 仕様記述のアサーション固有に必要となる，反例や，決定性の指定も可能

研究成果： ツール

- Eclipseプラグイン
  - 右図は結果表示画面



- 何種類かのモード
  - 適切な入力に対し適切な出力のテストケース（例）
  - 不適切な入力の（例外発生の）テストケース（例）
  - 適切な入力に対し不適切な出力の例
  - 与えたテストケースがテスト設計を満たすかのチェック（自動生成をオフに）

研究成果： セミナー

- 12/5，12/7の2日間実施（同じ内容）
  - 参加者計60名強，うち35名ほどアンケートに回答

  ※ 場所と時間配分の都合上，
  アンケートは後日回答可とした
  - 調査したところ，「アジャイル開発またはテスト駆動開発は業務で用いており，形式手法は勉強の経験がある」参加者が最も多かった（詳細略）
- イントロ，形式手法，テスト駆動開発，品質保証テストの4セッション
  - 3分野それぞれの概要を示した後，提案ツールで演習を行う

研究成果：セミナー結果抜粋（１）

- 今回用いた言語・ツールのよい点

（複数回答可）

テスト駆動・形式手法・テスト技法の考え方を1つのツールで扱える点	22
何か部分的にでも書くとすぐにツールを動かせる点	22
形式手法やソルバーツールとしては取っつきやすい点	17
様々なアプローチを織り込み視点を広げる・学ぶよい機会になる点	16
入力内容を変え様々なタスクをある程度こなせる汎用的なツールである点	14
「テスト一部自動生成」など軽い導入がしやすい点	14
その他	1

研究成果：セミナー結果抜粋（２）

- 今回用いた言語・ツールの使い道
  （複数回答可）

中堅者の視点を広げる教育・概念整理・意識向上	15
テスト駆動開発におけるテストケースの整理や議論	14
初心者の基礎的な考え方・技術への入門・教育	12
ドメイン分析や仕様化における制約の洗い出しや確認	11
品質保証におけるテスト設計の補助	10
特定なタスクによらず個人でのロジック整理・確認	9
プログラムに対する（自動処理できる）定型コメント	7
既存の形式手法を使う準備としての記述の厳密化や確認	7
論理的にややこしい問題への手軽なソルバー適用	4
その他	1

研究成果：セミナー結果抜粋（３）

- 今回用いた言語・ツールの特に改善すべき点
  （複数回答可）

実行速度の改善	18
UIの洗練(記述の自動補完・情報の表示など)	14
UML・Excelなど広く使われている表記との相互変換	8
その他	8
直接扱える問題の拡張(操作列や部品の統合など)	3
特定の利用法に絞って機能・UIの特化	3
出力結果の数や多様性などのチューニング	2

「その他」はドキュメント，モデル検査やランダムテストとの連携など

成果の活用：期待される効果

- 形式手法や，品質保証のテスト生成ツールで求められる厳密，十分な記述が段階的に，フィードバック付で行えるようになり容易となる
- （受け入れ）テスト駆動開発において，科学・工学・技術的な拠り所も促進する仕組みも持った上でテストケースの設定ができるようになる
- アジャイル開発，形式手法，品質保証テストの分野にまたがって，仕様やテストケース間の関係，基礎技術について考えるきっかけを提供する
- 手続き型プログラミングパラダイムで覆うことで，ドメイン分析や仕様化などにソルバー技術をより手軽に活用できる

想定との差異や限界

- サブタイトル「コード中心開発」
  - 実行速度などから，実コードよりも，概念モデルを対象とする方が向いている
  - ただし，MVCのM部分など，本質を取り出した実コードであれば使える状況はある
- 実行時間に課題が残る
  - 計画にはなかった単純なシンボリック化などは実装したが，まだ不十分で，期間内では対象外とした実装上の様々な工夫に取り組む必要がある
- セミナーによる意識向上やスキルに関する評価
  - 参加者は皆ポジティブで経験も多く，「思い込みからの脱出」や「スキル差」は今回現れなかった

まとめ

- 形式仕様とテスト生成の部分的・段階的な活用
  - 仕様，テスト設計，テストケースを与える言語
  - テストケース（具体例）を生成，提示するツール
    （JSTN: Java Specification and Testing Note）
- 仕様とテストを行き来する
  （Spec-Test-Go-Round）
  - 仕様，テスト設計（テスト仕様），テストケースいずれを導くタスクでも活用できる
- セミナーでの実証
  - 形式手法，テスト駆動開発，品質保証テストそれぞれの分野への一定の貢献
  - 分野を横断した理解・原則・議論・施策の促進