5A-07

ビジネスルールエンジンによる自動生成プログラムの単体テスト効率化に対する提案

小野　裕美† 小池　賢一† 大松　史生†
三菱電機株式会社　情報技術総合研究所†

1. はじめに

従来から業務アプリケーションにおいて，ビジネスルールを処理するためのミドルウェアとしてビジネスルールエンジン BRE（Business Rules Engine）やビジネスルールマネージメントシステム BRMS（Business Rules Management System）が広く利用されている．ビジネスルールは，専用のルール記述言語を使用してルールを記述するため，新たに言語を習得する必要があった．その問題点を解決するために，ビジネスルールをスプレッドシート上に日本語で記述する方式を提案した［1］．［1］では，スプレッドシート上に記述したルールを Java 言語に自動変換する．提案したルールエンジンでは，プログラムの生産性は向上するが，自動生成したプログラムは，別途単体テストを行う必要がある．

本稿では，自動生成したプログラムの単体テスト実施におけるテストの品質向上とコード生成の自動化を実現する方式を検討する．

2. 先行開発[1]の課題と解決策

先行開発であるスプレッドシートを利用したビジネスルールエンジンの実装方式を示す．

ビジネスルールで使用する項目は，多数の項目を使用してルールを記述することが多い．以下に，４つの項目を使用した場合の例を示す．

支払方法	= “カード”	and
カード種別	= “SMART”	and
配達先	= “自宅”	and
一括払い	= True	and
金額	＞ 10000	

図 1　ルールの例

このルールを単純に Java のプログラムに変換すると，if 文の記述が長くなり，各項目について毎回処理を行うため，処理の負荷が高くなる．

この問題点を解決するために，図 1 の場合は「支払方法」と「カード種別」と「配達先」を１つの文字列にアペンドした文字列「カードSMART 自宅」をハッシュのキーとするハッシュ表を作り，ハッシュ値では，残りの数値の条件が記述されたルールクラス名が得られるようにする．

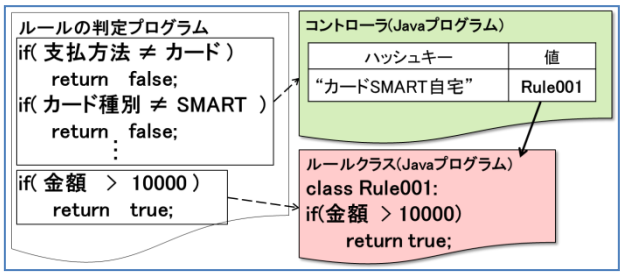

図 2　ルールの一部をハッシュ化した例

ルールエンジンで生成される Java プログラムは，コントローラとルールクラスが自動生成され，プログラムの生産性は向上するが，これらのプログラムは自動生成後，別途単体テストを行う必要があり，次のテスト工数が必要となる．

① 要領書作成
② 単体テストの項目作成
③ テストコード作成
④ テストコード実行
⑤ テスト結果の評価書作成

単体テストを行うためのテストコードは，それぞれコントローラとルールクラスのテストコードが必要となる．コントローラの単体テストでは，ルールクラスが適用または非適用となることを検証する必要がある．具体的には，正しいルールの場合は，ルールクラスが適用となることを確認し，間違ったルールの場合は，ルールクラスが非適用となることを確認する．生成するテストコードのテストケースの入力は，ルールの項目の値の組み合わせからなる条件となる．項目数i の場合，項目k_1の選択肢はn_1個，項目k_2はn_2個，…，項目k_iの選択肢はn_i個とする．１件のルールに対するテストケースの入力の組み合わせ数は，

The Approach for Efficient Unit Test of Auto-Generation Program by Business Rule Engine
†Hiromi Ono, Kenichi Koike, Fumio Omatsu
Information Technology R&D Center, Mitsubishi Electric Corporation

組み合わせ数＝ ${}_{n_i}C_1 \times {}_{n_2}C_1 \times \ldots \times {}_{n_i}C_1$ となり，さらにルール数に対して，上記の組み合わせのテストを行わなければならず，膨大な数となってしまう．それらに対して，使用するテストケースを手動で正確な選定を行うことは困難である．

3. システム構成図

本稿では，テストコードの品質向上を実現するために，コントローラに対するテストコード作成の作業を自動化するテストコード自動生成ツールの開発を行う．以下に，システムの構成を示す．

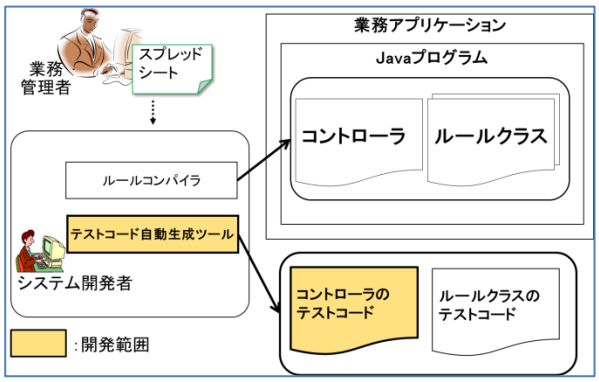

図 3　システム構成図

コントローラとルールクラスのテストコードが必要となるが，本稿では，コントローラのテストコードについて検討する．ルールクラスのテストコード自動生成法については，今後の課題とする．

4. テストコード自動生成ツールの開発

コントローラは，ルールクラスを呼び出す処理を行うため，本稿では，コントローラのテストコードとルールクラスのスタブを自動生成するテストコード自動生成ツールを開発する．

また，膨大なテストケースの絞り込みを行うために，判定条件に関わる条件文処理の動作を確認するために境界の値をテストデータとして選ぶ手法である境界値分析法を用いて，コントローラのテストケースを自動的に絞り込む．今回は、一つの項目のみが適用でない場合を境界値とし、テストケースの作成を行う。

5. 評価

表 1 において，すべての組み合わせ数と，境界値を用いた組み合わせ数の比較を行い，本手法の机上評価を実施した．

表 1　テストケースの組み合わせの一例

	NO. 1	NO. 2	NO. 3	NO. 4	NO. 5	
項目の種類数	5 (3)	4 (1)	7 (5)	2 (1)	3 (2)	840 (30)

表 1 は，ルールの項目数が 5 個あり，それぞれの項目の種類数を示す．括弧内の数字は，その項目が適用となる場合の種類数を示す．この場合，ルールの全体の組合せ数が 840 通りあり，その内，30 通りが，ルールが適用される場合の組み合わせ数となる．

ルールが非適用となり境界値となる組合せの数は，項目 No. 1 については，ルールが適用となる種類数が 3 種類のため，それ以外の 2 種類を用いてテストケースを作成する．その場合は，2 種類と項目 No. 2～No. 5 のルールが適用となる種類数を掛け合わせるので，2 × 1 × 5 × 1 × 2 = 20 通りとなる．その他の項目についても，同様の計算を行い合算した結果，167 通りとなった．その結果，境界値としてテストを行う個数は，適用となるルール数と非適用となるルール数を合わせた個数である 197 回となった．ルールの項目の組み合わせ最大数である 840 通りから考えると，実際にテストを行う個数は約 20%程度となる．

また，テストコード自動生成ツールにより，コントローラのテストコード作成時の人為的ミスを防ぐことができる。

6. おわりに

本稿では，ビジネスルールによって自動生成されたプログラムの単体テストの品質向上とコード自動生成の自動化を実現方式について検討した．本方式の特長は，境界値分析法を用いてテストケースの絞り込みを行っていることと，自動化によってテストの品質向上を実施している点である．今後は，ルールクラスのテストの自動化を検討していく．

参考文献

[1] 小池賢一，大松史生，谷屋直隆，小清水麻衣．スプレッドシートを利用したビジネスルールエンジンの実装．情報処理学会　ソフトウェア工学研究会，　2014.

[2] 町田欣史，高橋和也，小堀一雄，飯山教史．現場で使えるソフトウェアテスト Java 編．翔泳社，　2010.