serverspec: 宣言的記述でサーバの状態をテスト可能な汎用性の高いテストフレームワーク

宮下 剛輔[†1,†2,a)]　栗林 健太郎[†1,b)]　松本 亮介[†3,c)]

1. サーバの構成管理とテスト手法

本論文で提案するストフレームワークは，サーバ構成を記述したコードのテストをいかに効率よく行うか，という観点から出発している．そこで代表的な構成管理ツールを例に，ツールと言語の特徴，テスト手法ならびに従来手法の問題点について言及する．

1.1 CFEngine から Chef へ

安価な UNIX ライク OS を搭載したサーバが普及し，TCP/IP により異なる OS を搭載したサーバ同士がネットワーク接続可能になった．これによりシステムが大規模化・複雑化し，シェルスクリプトに代わり，サーバの設定を宣言的なコードで扱う構成管理手法が提案された．その実装として CFEngine?が登場した．

2005 年には CFEngine に影響を受け発展させた Puppet?が登場，2009 年には Puppet から影響を受けた Chef?が登場している．CFEngine，Puppet は独自の DSL(Domain Specific Language) でサーバの構成を記述するが，Chef は Ruby による内部 DSL で記述する．その特性ゆえ，Chef はシステム管理者よりも開発者に強く訴求し，Amazon EC2(Elastic Compute Cloud) の様な開発者自身がサーバ構築・運用を行える環境の普及とともに広まりを見せている．

1.2 Chef から Test-Driven Infrastructure へ

CFEngine や Puppet では，記述できることが独自 DSL の範囲内に収まるため，コードは比較的簡易なものとなる．しかし Chef は Ruby でできることはどんなことでも記述できる．そのためシステムが複雑になるにともない，それを記述するコードも複雑になる．そこでサーバ構成を記述したコードに対し，アジャイル開発におけるテスト駆動開発のプロセスを適用する事で，効率よくコードを記述することができる，といった考えが生まれる．Test-Driven Infrastucture という概念は Chef コミュニティ周辺で発生したものであるが，それは偶然ではなく，このような Chef が持つ言語特性ゆえである．

1.3 Test-Driven Infrastructure におけるテスト手法の分類

Test-Driven Infrastructure におけるテストの種類は，テスト駆動開発における以下の 3 つに分類できる．

(1) 単体テスト
(2) 結合テスト
(3) 受け入れテスト

単体テストは構成管理ツールにおける “モジュール” に対するテストで，実際にコードをサーバに適用する前の段階で行うテストである．結合テストはコードをサーバに適用した後に行うテストで，コードが期待通りにサーバの設定を行ったかどうかをテストする．受け入れテストもコードをサーバに適用した後に行うテストだが，結合テストがサーバ内部の状態をテストするホワイトボックステストなのに対し，受け入れテストはサーバの外から見た振る舞いをテストするブラックボックステストであるという違いがある．

1.4 従来テスト手法の問題点

単体テストツールとしては Chef には ChefSpec，Puppet には rspec-puppet というツールが存在する．その名が示すとおり，それぞれ Chef と Puppet 専用のツールであり，単体テストツールとしては十分であるが，実際にコードを

†1 現在，株式会社 paperboy&co.
†2 現在，帝京大学
†3 現在，京都大学
a) gosukenator@gmail.com
b) kentarok@gmail.com
c) matsumoto_r@net.ist.i.kyoto-u.ac.jp

サーバに適用した結果をテストすることはできないため，単体テストツールだけでは不十分である．

結合テストツールとしては，minitest-chef-handler，Test Kitchen，rspec-system が存在する．この内 minitest-chef-handler と Test Kitchen は Chef に依存したツールであり，他の構成管理ツールとともに利用することができない．Test Kitchen や rspec-system は，テスト用 VM の作成，テスト用 VM への構成管理ツールの適用，テストの実行をトータルで行う統合テストスイートであるが，ツール標準のテスト機構は汎用性に乏しく，特定の構成管理ツールに依存していたり，OS やディストリビューション毎の違いを意識したテストコードを書く必要がある．

受け入れテストツールとしては cucumber-chef，leibniz が存在するが，これらは Chef に依存したツールであり，他の構成管理ツールとともに利用することができない．

以上をまとめると表 1.4 のようになる．

ツール名	テスト種別	ツール独立性	OS 汎用性
ChefSpec	単体	×	○
rspec-puppet	単体	×	○
minitest-chef-handler	結合	×	○
Test Kitchen	結合	×	×
rspec-system	結合	○	×
Cucumber-Chef	受入	×	○
leibniz	受入	×	○

表 1 テストツール比較表

参考文献

[1] Mark Burgess, “CFEngine: a site configuration engine”, USENIX Computing systems, Vol8, No. 3 1995, http://cfengine.com/markburgess/papers/paper1.pdf.