

# Linux カーネル互換性テストツールの開発 —リグレッション・テスト・フレームワーク—

## 1. 背景

Linuxカーネルの開発プロセスには、自動的・機械的な互換性の保持メカニズムが存在しない。そのため、バグフィックスやバージョンアップなどの改変による意図しない互換性の問題が生じた場合にも、自動的に発見される仕組みになっていない。Linuxカーネル開発コミュニティのテスト方法は、頻繁なリリースと多数の開発者によるレビューとフィードバックが基本となっている。したがって、利用者やアプリケーション開発者が互換性の問題を発見するのはリリース後となる。

開発者が意図する非互換についても、開発モデルの特徴から、明確に文書化されにくい。「XXX という機能を追加」という記述はあっても、それによって「YYY という非互換が発生」という記述はほとんどない。

しかしながら、今後も増加する Linux 対応アプリケーションの開発者にとって、その動作検証・非互換への対策にかかる工数増は大きな問題となる。ソフトウェア開発による意図しない非互換作りこみを検出するためのテストツールが必要である。

## 2. 目的

本テストツールは、Linuxカーネルのインターフェースについて、互換性をチェックするツールを開発することで、意図しない非互換の作りこみ防止と、非互換の事前抽出によるアプリケーション移植工数の削減を目標とする。なお、本テストツールは Linux Standard Base などの標準仕様への準拠を検証するツールと異なり、特定のカーネルとその更新版カーネルの非互換性についても検出できるものとする。

テストツールは、次のような機能を含む。

- ・Linuxカーネルの振る舞いを自動的に調査し、その結果を保存する機能
- ・保存された結果の差分を検出し、非互換性の発生を指摘する機能
- ・テスト結果および予期する結果を管理(登録、変更、削除など)する機能

本テストツールの開発においては、開発当初より Linuxカーネルコミュニティ、テストコミュニティ、北東アジア OSS 推進フォーラム、ディストリビュータ等々と連携をとり、いわゆるバザールモデル方式によって開発を推進する。

## 3. 開発の内容

カーネル互換性テストツールは大きく分けて3つの部分から構成される

- ・リグレッション・テスト・フレームワーク

- ・リグレッション・テスト
- ・テストカバレッジ測定ツール

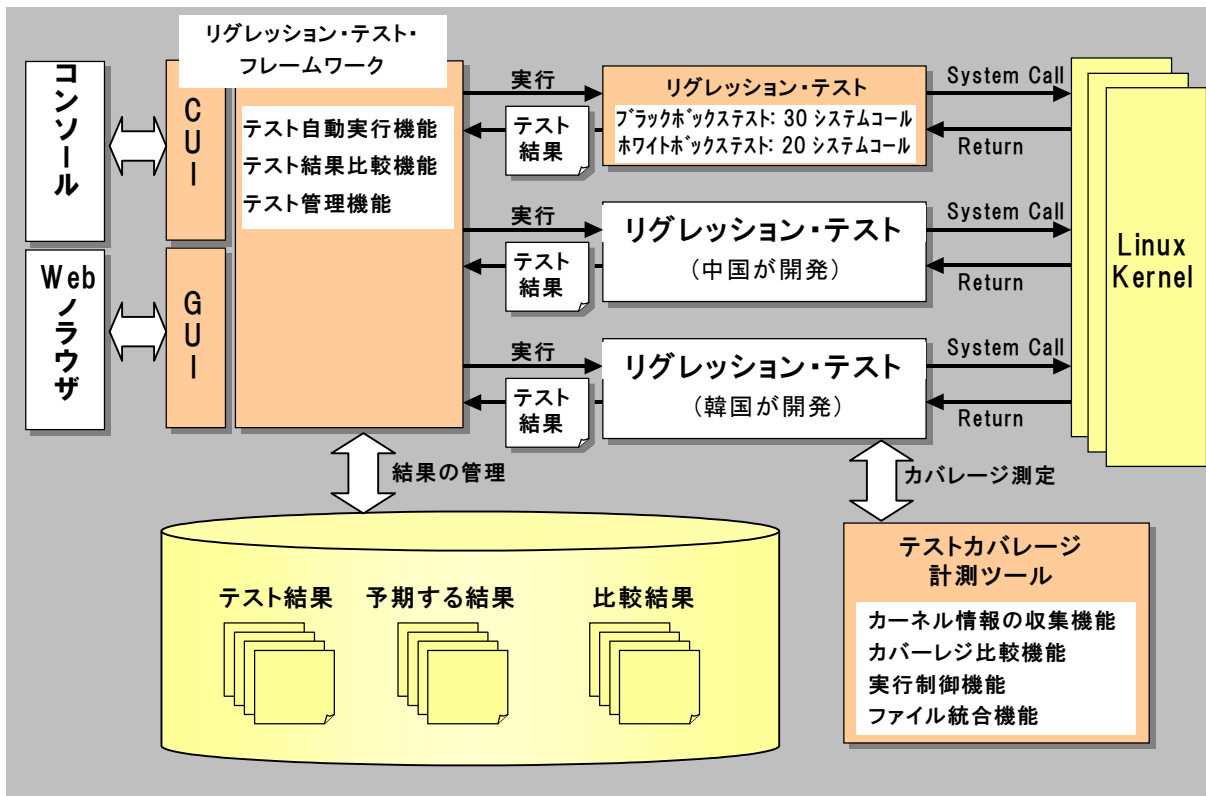


図1 リグレッション・テスト・ツールの構成

### (1) リグレッション・テスト・フレームワーク

CUI、GUIにより以下機能を実装

- ・テスト管理機能
- ・テスト自動実行機能
- ・テスト結果比較機能

### (2) リグレッション・テスト

50個のシステムコールについてテストプログラムを開発

- ・ブラックボックステスト (30 システムコール)
  - man ページ等で定義された機能についてのテストを行うもの
- ・ホワイトボックステスト (20 システムコール)

ソースコードレベルの調査、あるいはテストカバレッジ分析((3)テストカバレッジ測定ツールで実施)により、テストケースの追加修正などを行ったもの

### (3) テストカバレッジ測定ツール

#### A) カーネル情報の収集機能

システムコール実行時の分岐情報の取得

システムコールの実行情報(実行の有無など)の取得

B) カバレッジ比較機能

- ・ バージョンの異なるディストリビューションでの、同じベンチマークの実行結果比較
  - システムコール実行時の、カーネル全体/関数毎のカバレッジ率算出
  - 実行の有無を示す関数のリスト表示
  - システムコール実行時の、カーネル全体/関数毎のソースコード上でのカバレッジ表示
- ・ 同じカーネルでの 2 種類のベンチマーク実行結果の比較
  - 上記と同等の機能
  - カバレッジの差分部と共通部の割合算出、可視化

C) 実行制御機能(高速化の一環)

いくつかの実行条件(PID, システムコール番号など)で分岐情報取得機能の実行を制御情報を絞ることで高速化を図る。

D) ファイル統合機能

カーネルバージョン等は同じ条件で、採取する場所を変えて採取した複数の分岐情報 ファイルを、1つにマージする機能。

4. 従来の技術(または機能)との相違

従来からLinuxカーネルの開発プロセスには、自動的・機械的な互換性の保持メカニズムが存在しなかった。そのため、バグフィックスやバージョンアップなどの改変による意図しない互換性の問題が生じた場合にも、自動的に発見される仕組みになっていなかった。

5. 期待される効果

(1)Linuxカーネル開発者

- ・ 既存カーネルのバグフィックスの場合。そのバグフィックスが、従来の機能を非互換に破壊していないか確認する。当該バグについてもテストプログラムを追加し、既存カーネルではバグが発生し、バグフィックス版では当該バグが発生しないことを確認する。
- ・ 新機能開発の場合。その新機能が従来の機能を非互換に破壊していないかの確認をする。非互換な拡張の場合、その非互換の範囲を確認する。また、新機能についてのテストプログラムも追加し、今後その機能についても互換性の維持をはかる。

(2)ミドルウェア開発者

- ・ カーネルの新バージョンについてリグレーションテストを実行し、非互換の確認、もし非互換があればその範囲を特定する。ミドルウェアのリグレーションテストも用意しておけば、そのテストを流すことによって容易にカーネルの非互換を発見

できる。

## 6. 普及(または活用)の見通し

本プロジェクトで開発するツールおよびテストプログラムは開発初期段階より、広くカーネルコミュニティ、ディストリビュータ、アプリケーションベンダ等に公開し、開発そのものもバザールモデルを意識したものにす。また、国際的な連携として The Linux Foundation、FSG 他カーネルテストコミュニティとも連携する。

OSS Test Tools Summit にも参加し国際的に認知され、各 OSS テストコミュニティとの連携も視野に入れている。

Sourceforge 等 OSS の定番ポータルサイトを利用し、開発の初期の段階から広く OSS コミュニティと連携しバザールモデルで開発する。Wiki やメーリングリストを積極的に利用し開発プロセスそのものをオープンにする。

国際的な共同開発体制をしき、カーネルコミュニティ、ディストリビュータ、アプリケーションベンダー等に認知されるように試みる。

## 7. 開発者名(所属)

- \* 杉田由美子((株)日立製作所 システム開発研究所)
- \* 安井隆宏((株)日立製作所 ソフトウェア事業部)
- \* 藤原哲((株)日立製作所 システム開発研究所[日立情報通信エンジニアリング(株)])
- \* 柳谷良介(ミラクル・リナックス(株) コアテクノロジー部)
- \* 八木和生(ミラクル・リナックス(株) コアテクノロジー部)
- \* 樽石将人(レッドハット(株) グローバルサービス本部)

### (参考)開発者URL

(株)日立製作所

<http://www.hitachi.co.jp/>

ミラクル・リナックス(株)

<http://www.miraclelinux.com/>

レッドハット(株)

<http://www.jp.redhat.com/>