組み込み機器のテストプログラムのパターン

近藤　理
(株)富士通コンピュータテクノロジーズ
kondo.tadashi@jp.fujitsu.com

坂本　一憲
国立情報学研究所
exkazuu@nii.ac.jp

ABSTRACT

組み込みシステム等の装置開発において、装置の機能や振る舞いを試験するテスト工程がある。我々は、テスト工程において装置に対するデータアクセスや操作を自動化するテストプログラムを開発している。テストプログラムは、テスト対象の装置ごとに操作のインタフェースやテストの内容が異なり、別々のプログラムとして開発することが多いため、開発手法やノウハウが共有されにくい傾向にある。

本論文では、テストプログラムの開発経験から共有可能な知見をパターンとして抽出して、組み込みシステムのテストにおけるパターンランゲージを提案する。

Categories and Subject Descriptors

D.2.10 [Software Engineering]: Design

General Terms

Design

Keywords

Architecture Pattern, Design Pattern, Test Pattern, Testing, Embedded System

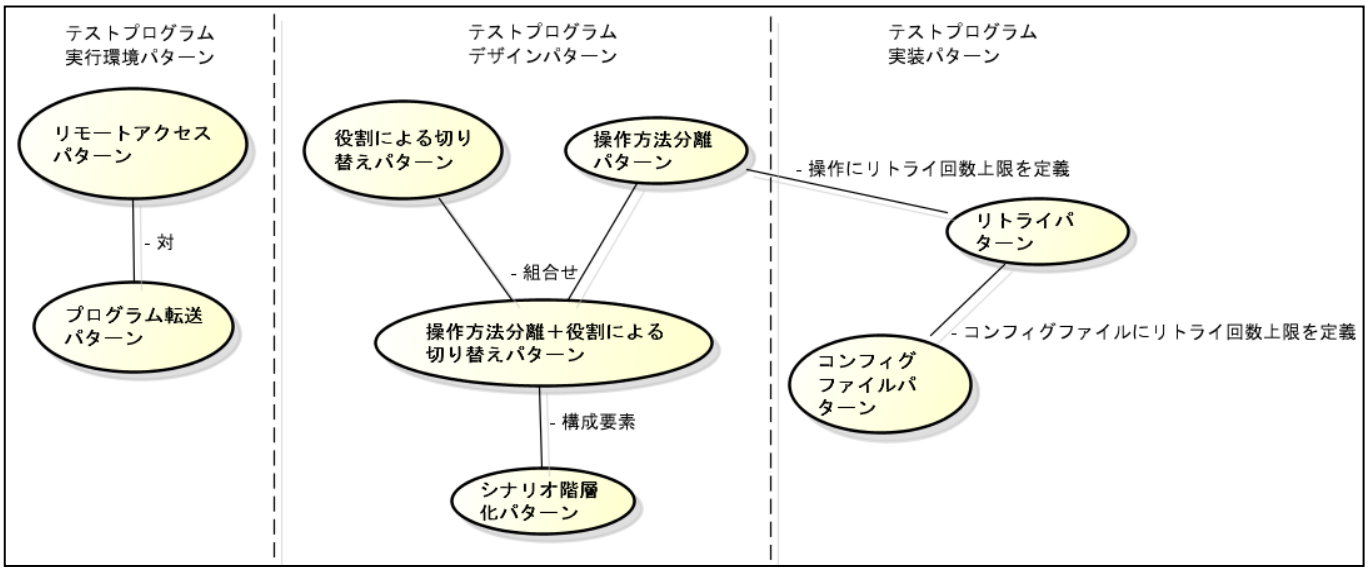
1. はじめに

現代社会において、デジタル家電や生活インフラなどあらゆる製品で組み込みシステムが利用されており、組み込みシステム技術はますます重要となっている。

組み込みシステムの品質を確保するために様々なテストを実施する必要がある。我々は、テスト工程を効率化するために、データアクセスや操作をテストの手順として定義して、テストの実施を自動化するテストプログラムを開発している。

ネットワーク機器やディスクアレイ装置等の組み込みシステムでは、装置ごとに機能や操作のインタフェースが異なるため、装置ごとにテストの目的にあったテストプログラムを開発しており、開発手法やノウハウが共有されにくい傾向がある。本論文では、これまで共有されていなかったノウハウを共有するために、テストプログラムの開発経験から得られた知見をいくつかのパターンとして形式化して、パターンランゲージとしてまとめ上げる。

2. パターンの全体像



3. テストプログラム実行環境パターン

3.1 リモートアクセスパターン

- 状況

テスト対象が外部からの操作に対応する入出力のインタフェース（外部操作 I/F）を持っている。

テスト対象装置が複数ある、もしくは、テスト対象装置内でプログラムを実行できない。

- 問題

テストプログラムの実行環境を、対象装置ごとに構築しなければならず、手間がかかる。

複数の実行環境が存在すると、それぞれの環境でテストプログラムの動作確認をしなければならない。

- フォース

テストプログラムの実行環境を一元化したい。

一つの実行環境から複数の装置に対してテストを実施したい。

- 解決策

テストプログラムの実行環境を一元化して、外部操作 I/F を用いることで各対象装置のテストを実施する。

- 結果

実行環境が一元化されるため、環境構築が簡単になる。

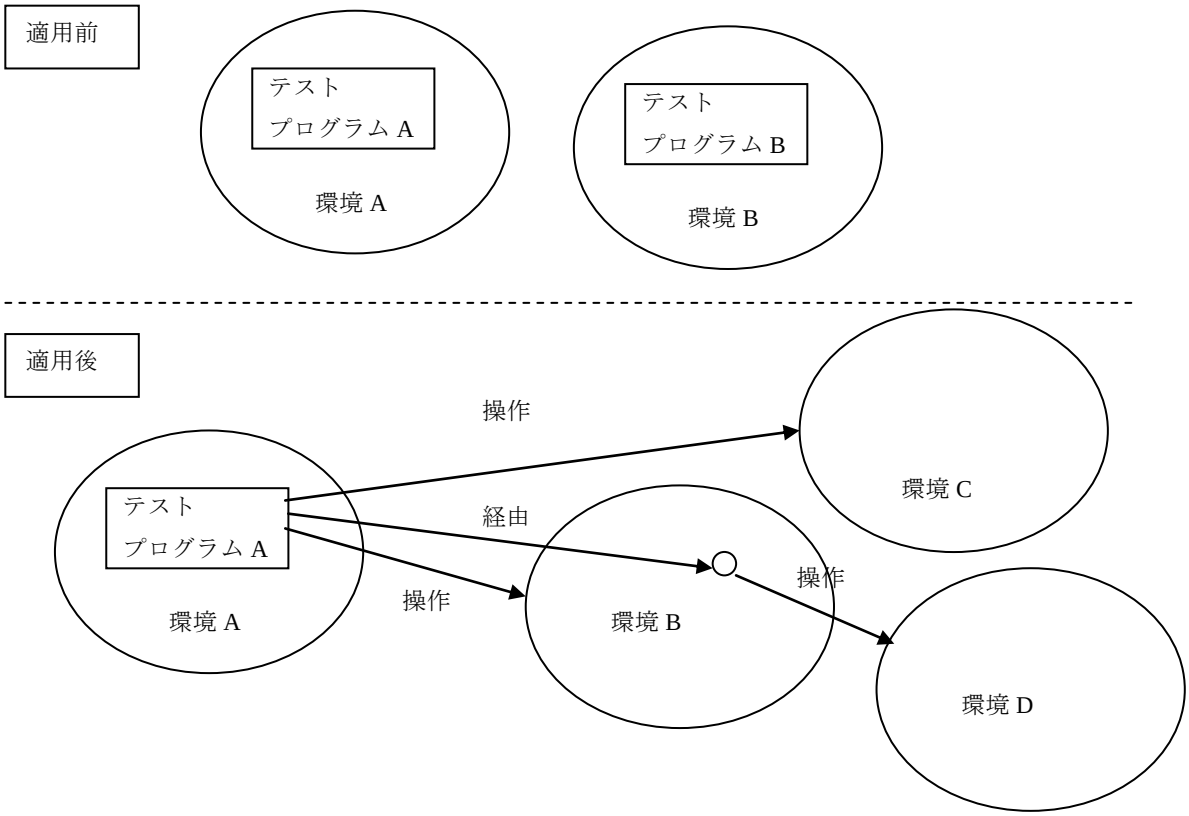
一つのテストプログラムから複数の装置に対してテストを実施できるようになる。

- 関連パターン

プログラム転送パターンと排他的な関係にあり、どちらか一方のみを適用可能である。

- 利用例

サーバ、ネットワーク機器、ディスクアレイ等で構成されたシステムに対して、システム全体の振る舞いをテストする。



3.2 プログラム転送パターン

- 状況

性能測定や負荷テストを目的としており、レスポンス時間を最小限に抑えた対象装置の操作が必要である。

- 問題

リモートアクセスパターンでは、外部操作 I/F 経由なので、実行速度が低下して、性能測定や負荷テストには不向きである。

- フォース

性能測定や負荷テストを目的としたテストを実施したい。

- 解決策

テスト対象装置にプログラムの一部を転送し、対象装置上にて直接テストを実施し、その結果を受け取る。

- 結果

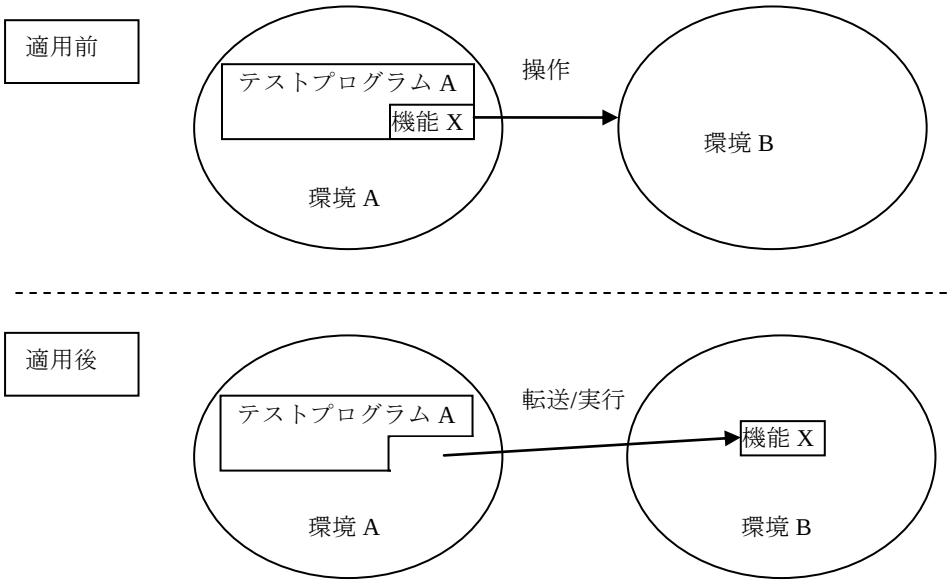
対象装置を直接操作することで、操作の命令に対するレスポンス時間の増大を抑える。

- 関連パターン

リモートアクセスパターンと排他的な関係にあり、どちらか一方のみを適用可能である。

- 利用例

ディスクアレイに接続されたホストサーバ上で性能測定用のプログラムを実行し、測定結果をテスト用 PC で取得する。



4. テストプログラムデザインパターン

4.1 操作方法分離パターン

- 状況

テストプログラムを開発する際、様々なテスト対象装置に対して、同じ手順のテストを行う。ただし、対象装置ごとに具体的な操作方法が異なり、各手順を構成する操作内容を変更する必要性がある。

- 問題

同じテスト手順を異なる対象装置で実施する場合、操作方法が異なるので、類似性の高いプログラムを別々に用意しなければならない。

- フォース

開発工数を低く抑えたい。また、操作方法やテスト対象装置の仕様が変更されても、早期にテストできるようにしたい。

- 解決策

テストの目的を実現するテスト手順と、手段となる操作方法を分離する。テスト手順はテストのシナリオとして共通に定義しておき、対象装置に対して各操作内容を定義する。

テスト実施においては、具体的な対象装置を与えて、対象装置に対応する操作方法を用いてテスト手順を実施する。

- 結果

別のテスト対象装置をテストする場合でも、操作方法の部分だけを開発することで、開発工数を抑えられる。開発工数を抑えることで、テスト対象装置の操作方法が変更されても、早期に対応できる。

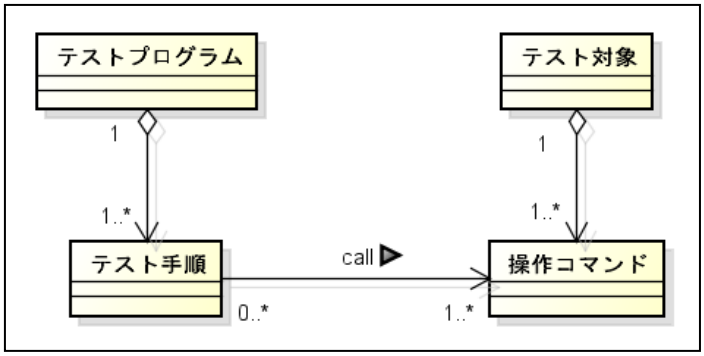
- 関連パターン

役割による切り替えパターンと組み合わせることで、操作方法分離＋役割による切り替えパターンを実現できる。

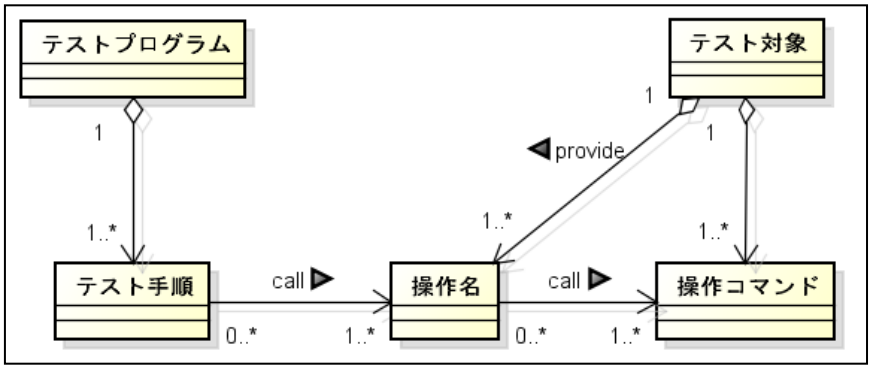
- 利用例

サーバ OS 上の実行中プロセスを取得する際に、対象 OS によりコマンドが異なる。この場合に、テスト手順としては、操作名を"get_process"としておき、対象 OS が Linux の場合は"ps"を操作コマンドとして定義する。

適用前



適用後



4.2 役割名による切り替えパターン

- 状況

同じテスト対象装置が複数存在していて、それぞれ異なる役割を担っている。もしくは、同じ役割であっても異なるテスト対象装置を利用する。

- 問題

同じテスト対象装置であっても担っている役割が異なり、対象装置の製品名だけでは役割を区別できない。また、テスト手順を定義する際に、テスト手順がどの装置をテスト対象とするか定義する必要がある。そのため、具体的な対象装置を変えるために、テスト手順が変化しないのにも関わらず、テスト手順と対象装置の組み合わせを再定義する必要がある。

- フォース

複数あるテスト対象装置を区別して、かつ、役割を分かりやすくしたい。同じ役割を担うテスト対象装置を切り替えたい。

- 解決策

テスト対象装置に役割の名前を与える。テスト手順と役割名の組み合わせを定義することで、役割名が指す具体的なテスト対象装置を切り替えるようにする。

- 結果

テスト対象装置に役割の名前を与えることで、役割が分かりやすくなる。役割に対して手順を定義できるので、具体的なテスト対象装置に依存しない。

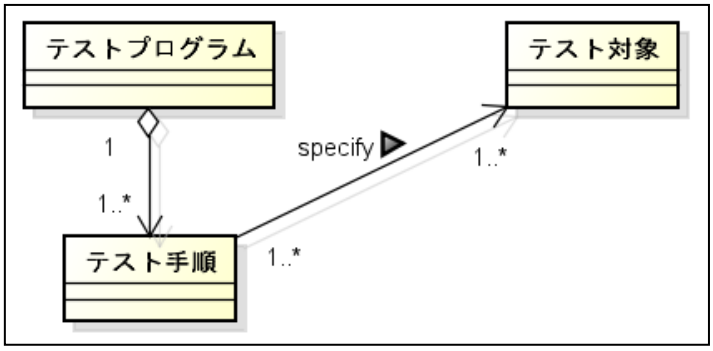
- 関連パターン

操作方法分離パターンと組み合わせることで、操作方法分離＋役割による切り替えパターンを実現できる。

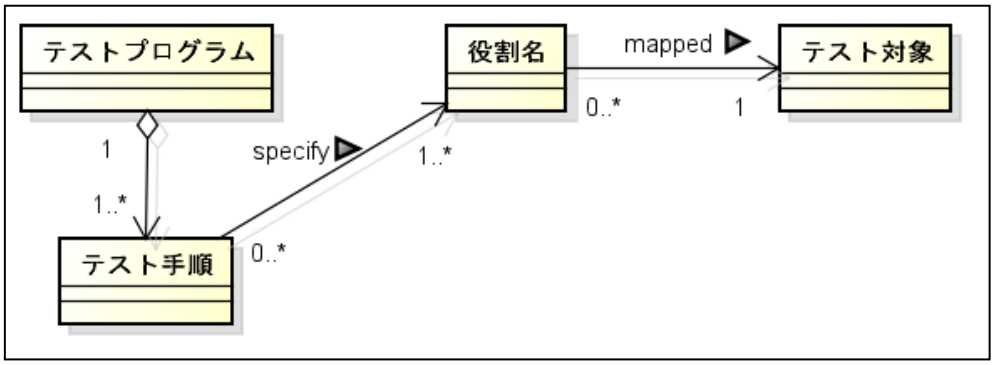
- 利用例

ツリー上に構成されるネットワーク機器に対して、ノードの親子関係がある場合、"親"に行う操作と"子"に行う操作をそれぞれ手順に記述する。

適用前



適用後



4.3 操作方法分離＋役割名による切り替えパターン

- 状況

複数の装置で構成されたシステムを試験する。それぞれの装置に対して操作を行い、検証する。また、システムの構成によって対象装置種別の組み合わせが複数ある。

- 問題

同じテスト対象装置であっても担っている役割が異なり、対象装置の製品名だけでは役割を区別できない。また、テスト手順を定義する際に、テスト手順がどの装置をテスト対象とするか定義する必要がある。そのため、具体的な対象装置を変えるために、テスト手順が変化しないのにも関わらず、テスト手順と対象装置の組み合わせを再定義する必要がある。

また、同じテスト手順を異なる対象装置で実施する場合、操作方法が異なるので、類似性の高いプログラムを別々に用意しなければならない。

- フォース

さまざまなテスト対象に同じテスト手順を適用したい。

テスト手順をテスト対象の依存関係をなくしたい。

- 解決策

4.1と4.2を組合せる。役割に対して、実行して欲しい目的の操作名を指示することにより、テスト手順とテスト対象・操作コマンドの依存関係を排除する。役割名はテスト対象が持っている操作名の内、実行可能な操作名を公開する。

- 結果

役割名と操作名をテスト手順とテスト対象の間に入れることにより、直接的な依存関係を排除した。

テスト対象の置き換えやテスト手順の変更が容易になった。

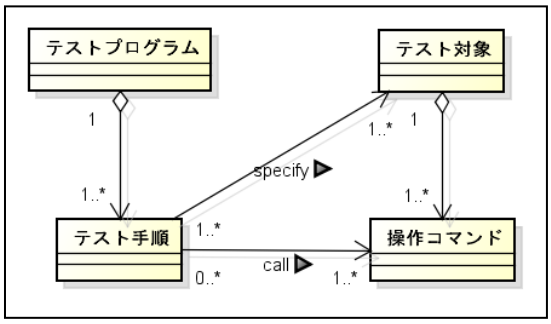
- 関連パターン

役割による切り替えパターンと操作方法分離パターンを組み合わせることで実現できる。

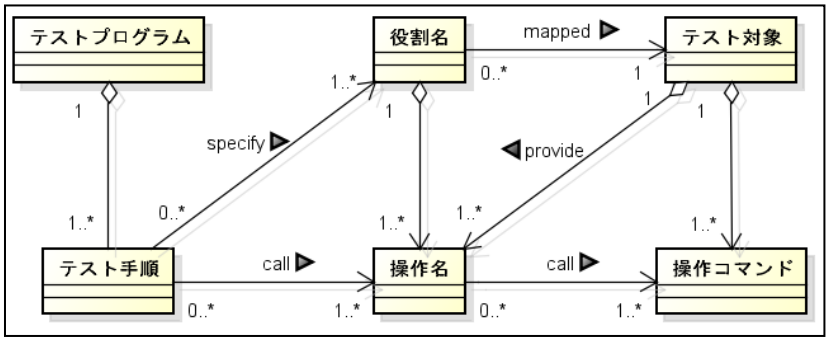
- 利用例

ディスクアレイに接続されるホストが Windows サーバと Linux サーバがある場合に、役割としてそれぞれ"Host1", "Host2"として定義し、ディスクアレイに対する Write/Read や OS 再起動といった操作名をテスト手順に記述する。"Host1", "Host2"は、それぞれの OS に対応した操作コマンドを実行する。

適用前



適用後



4.4 シナリオ階層化パターン

- 状況

テストの手順（処理の順番や内容）を変更することがある。テストの手順の中に、同じような処理が複数存在する。

- 問題

テスト手順に同じ処理が複数回現れるので、コピー&ペーストにより記述しがちであるが、テストプログラムにコードクローンが作りこまれるため、当該する処理を変更する際に、全てのクローンの修正が必要になる。手順の処理内容がフラットに記述されていると、手順の順番の入れ替えが困難である。

- フォース

現場で使用者自身が手順の変更を行い、すぐに試したい。テスト手順を別にテストにも簡単に流用したい。

- 解決策

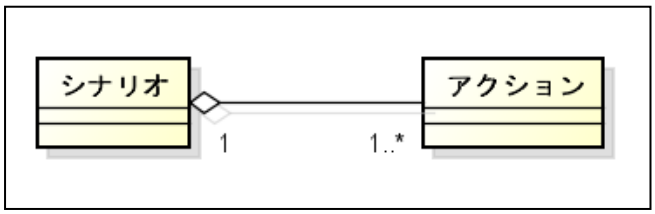
手順をシーンとアクションによって階層化する。シナリオは、シーンとアクションから成る。アクションは、テスト手順を構成する最小要素となるテスト対象装置の操作内容を示す。シーンは、複数のアクションまたはシーンからなり、アクション単体よりも抽象度の高い手順として定義する。共通の手順はシーンとしてまとめておくことで、様々な場面で繰り返し現れる手順の内容を局所化できる。また、あるシーンを他のシーンで構成できるようにすることで、テスト手順の抽象度を任意に設定できる。

- 結果

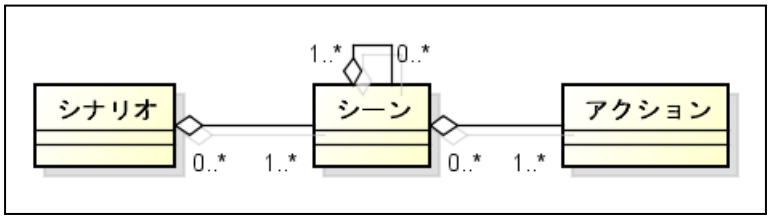
手順が抽象化されたシーンで記述できるので、どんなテスト手順か概要がすぐに理解できる。だれでもテスト手順の変更が容易にできる。別のテストにも手順を流用しやすくなる。

操作方法分離＋役割による切り替えパターンを用いた際に、手順を構成する要素の記述で利用できる。

適用前



適用後



- 利用例

ディスクアレイのテスト手順記述

シナリオ：RAID 作成テスト｛
　シーン：装置状態確認
　シーン：RAID 作成
　シーン：装置状態確認
｝

シーン：装置状態確認｛
　アクション：装置状態取得
　アクション：装置状態が正常か確認
｝

シーン：RAID 作成｛
　アクション：利用可能 Disk 取得
　アクション：RAID 作成
　アクション：RAID 作成確認
｝

5. テストプログラム実装パターン

5.1 コンフィグファイルパターン

- 状況

テスト対象や実行環境によってテストパラメータを変更する。

テストパラメータが多数ある。

- 問題

テストに必要なパラメータをテストプログラムの引数として与えると、引数が長く、複雑になる。

パラメータの追加や削除があると引数処理全体への影響が大きい。

- フォース

テストの入力パラメータを簡単に設定したい。

テストプログラム実行ごとにパラメータを設定する面倒をなくしたい。

- 解決策

テストに必要なパラメータは、コンフィグファイルに記述し、テストプログラムは実行時にコンフィグファイルを読み込むことで、パラメータを取得する。パラメータの記述は、パラメータ名と値の関係で記述し、分かりやすく箇条書きにする。

- 結果

コンフィグファイルに記述することで、多数あるパラメータを容易に設定できる。

プログラム自体を変更することなく、パラメータを変更できる。

- 関連パターン

リトライパターンの振る舞いを設定する際にコンフィグファイルパターンを利用できる。

- 利用例

ディスクアレイテスト時のコンフィグファイル

アクセス IP アドレス：192.168.1.1
アクセスポート番号：xx
リトライ回数：10
リトライタイムアウト[sec]：60
テスト番号：xx, yy, zz
ディスク番号：aaa, bbb, ccc

5.2 リトライパターン

- 状況

テスト対象や周辺環境の状態により操作が失敗することがある。

- 問題

テスト項目とは関係無い箇所で操作の失敗が発生した場合、Fail と判定するため、本来のテストを実行できない。

- フォース

テストをできるだけ継続したい。テストで異常が見つかっても可能な限り復旧させたい。

許容範囲を超えるものはエラーとして検出したい。

- 解決策

操作が失敗した場合に操作をリトライする。リトライの上限は、指定回数またはタイムアウトで設定する。

リトライを許容しない操作に対しては、リトライしない設定とする。

リトライした回数やタイムスタンプを記録として残す。

- 結果

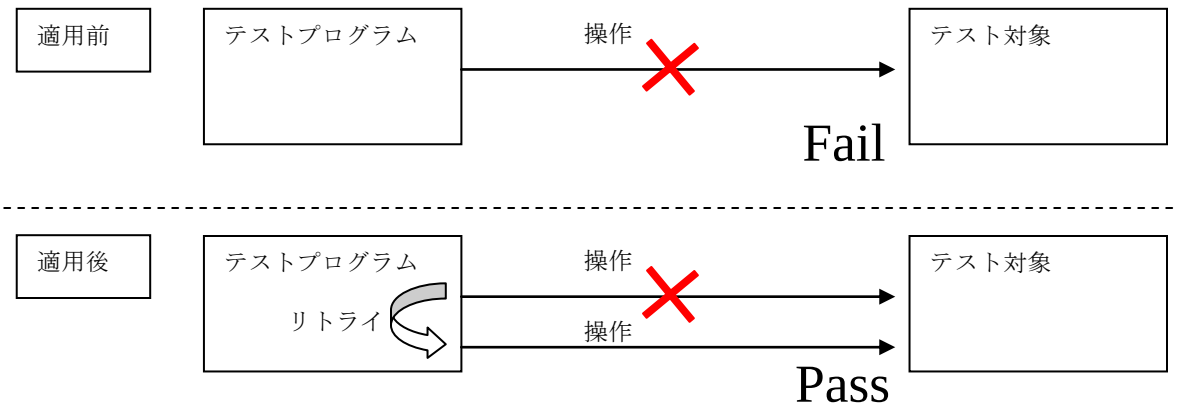
操作が失敗した場合でもリトライによりテストを継続できる。

- 関連パターン

リトライパターンの振る舞いを設定する際にコンフィグファイルパターンを利用できる。

- 利用例

ディスクアレイ装置への操作を行う場合に telnet を使用する。ネットワークの状態により稀に接続エラーする場合があるが、リトライにより接続できれば試験を続行する。



6. まとめ

我々は、組み込みシステムに対するテストプログラムの開発経験から、テストプログラムのパターンランゲージを提案した。また、テストプログラムの実行環境、デザイン、実装の３種類のパターンへと整理した。これらのパターンはテストプログラムの新規設計時だけでなくリファクタリングの際にも活用できると考える。

今後は、これらのパターンの追加および有用性の評価を行い、テストプログラム開発者の役に立つパターンランゲージへと進化させたい。