機能競合を考慮したソフトウェアテスト手法

6H−4

水野 浩之、山本 徹也、田村 一賢、平山 雅之
（株）東芝 研究開発センター システム技術ラボラトリー

1. はじめに

複数機能の同時動作が起こる複雑なソフトウェアとして、ハードウェアに組み込まれる制御ソフトウェアが考えられる。制御ソフトウェアでは、制御対象となる機器をフラグや共有データを利用して制御しているものが多い。このようなソフトウェアでは、複数の機能が同時に動作する際に矛盾する動きを引き起こすフラグが同時にセットされたり、同一の共有データへ同時にアクセスしたりすることがある。このような状況を機能競合とよぶ。本稿では、各機能の状態遷移図からフラグや共有データの使用タイミングに注目することで、機能競合を検査するためのテストケースを効率よく作成する手法を提案する。

2. 現状の課題

機能競合不具合

機能競合時には、フラグのセット／リセットや共有データのアクセスのタイミングによっては想定外の動作が起こり、不具合が発生することがある。例えば、モーター制御用のソフトウェアで順方向へ回転させるフラグに対し、逆方向へ回転させるフラグが同時にセットされるとモーターが回転できなくなることが考えられる。通常、こうした相反する関係を持つフラグや共有データへの同時アクセスに対して、どちらの処理を優先させるかが明確に設計されている。しかし、様々なタイミングを考慮したシステムの振る舞いを設計段階で全て検討することは難しい。そのため、設計を基に実装したソフトウェアが本当に意図した通りに動作するかは、テストして調べる必要がある。

機能競合不具合のテストに関する課題点

機能競合に関する不具合を検出するためのテストケースを作成については次のような課題がある。

① 全ての機能に対して、各機能における状態、各状態におけるフラグの組み合わせを考慮して、競合が起こるタイミングを特定しなければならない。
② 全ての組み合わせを網羅的に考慮するのは、膨大な数となるため非常に時間がかかる。

3. 機能競合に対するテスト手法

3.1 テスト手法の概要

当社では、上記課題の解決を目標に、機能競合に関するテストケースを効率的に作成する手法の開発を進めている。

この手法では、組み込みソフトウェアで使われているフラグ, 共有データに着目する。そして、各機能において使われるフラグ, 共有データの中から、相反するフラグ（以下、相反フラグ）や同一の共有データを特定し、競合が起こり得る機能の組み合わせを絞り込む。その後、競合が起こり得る機能間で状態遷移図を比較し、相反フラグの同時セット、共有データへの同時アクセスが起こる状態を、機能競合が発生する状態として特定するものである。

3.2 テスト手法の流れ

機能競合に対するテスト手法の流れを図 1 に示す。以下に、手法の概要を述べる。

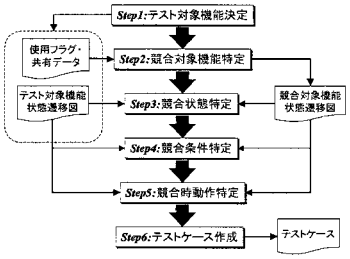

図1　機能競合に対するテスト手法の流れ

Step1: テスト対象機能決定

一般に機能競合不具合は、既存システムに新規の機能を追加したり、既存機能を一部変更した場合に、既存機能との間で発生することが多い。このため機能競合のテスト対象機能としては新規に変更や追加が行われた機能をテストすることが効果的と考えられる。

ここでは、テスト対象機能の仕様から状態遷移図を作成し、特に内部で使用するフラグや共有データをアクションとして明記する。

Step2: 競合対象機能特定

テスト対象機能のフラグと相反フラグ、同一の共有データを使用するものを競合対象機能として特定する。競合対象機能に対しても、仕様から状態遷移図を作成する。

Step3: 競合状態特定

テスト対象機能と競合対象機能の状態遷移図を比較し、相反フラグや同一の共有データが同時に使用される状態を特定する。

Step4: 競合条件特定

競合状態の成立に必要な前提条件と操作を特定する。
前提条件:競合状態の前の状態に至るまでに必要な条件をテスト対象機能と競合対象機能の両方に対して特定する。
操作:競合前の状態から競合状態への遷移条件を特定する。

Step5: 競合時動作特定

競合状態での動作は、基本的にフラグ間の優先度関係や共有データへのアクセス順位に従って決定される。その他、競合状態になるタイミングや抜けるタイミングを考慮することで特定する。

Step6: テストケース作成

これまでの処理で特定した競合状態になるための前提条件と操作、競合状態において期待される動作を基に、実際にテストできる形にしてテストケースを作成する。

4. 適用評価

The Software Test Technique in Functional Competition: Hiroyuki Mizuno, Tetsuya Yamamoto, Kazuyoshi Tamura, and Masayuki Hirayama, Research and Development Center, Toshiba Corporation.

ここではある制御用ソフトウェアに対して本手法の適用した事例を紹介する。

4.1 適用対象

適用したシステムは、レール上を上下に移動し、要求のあった位置まで来るとランプを点灯する機器を制御するソフトウェアであり、約 110 種類の機能を持つ。

4.2 適用状況

この機器の機能 A と機能 B の 2 つの機能について、機能競合のテストケース生成の過程を示す。

機能 A : 制御対象をあらかじめ定められたホームポジションまで戻す機能

機能 B : 制御対象をレールの下端に一旦停止させ、その後上端まで引き上げる機能

競合状態特定(step-3)

これらの機能動作の状態遷移図は図 2 で表される。ここで矛盾する関係にあるフラグとしては、「点灯フラグ」と「消灯フラグ」が挙げられる。これらのフラグは、「消灯フラグ」の方の優先度が高く設定されており、要求位置に停止したとき、同時にこの 2 つのフラグがセットされると、ランプは消えたままになる。

この 2 つのフラグが同時にセットされる状態(**競合状態**)としては、機能 A の「ホームポジション、停止中」と機能 B の「最下端、停止中」が同時に成り立ったときに絞り込むことができる。

競合条件特定(Step-4)

競合状態になる前提条件は、ホームポジションの設定が最下端になっていること(以下「ホームポジション＝最下端」と示す)である。また、競合前の状態は[ホームポジション＝最下階移動要求応答、減速中]で、この状態に至るためには、[機能 A と機能 B のどちらかが動作している]ことが必要となる。

競合状態が成立するための操作は、競合前状態から競合状態への遷移条件から、どちらかの機能が動作していれば、そのまま競合状態まで遷移することがわかる。よって、競合前状態において、どちらかの機能が動作しており、まだ動作の始まっていない方の機能を開始させることで競合状態にすることができる。

競合時動作特定(Step-5)

競合状態時動作については、競合状態になるタイミングから、先に機能 A が動作していて後から機能 B が動作する場合とその逆が考えられる。

機能 A が先に動作していた場合

「ホームポジション＝最下端、停止中」にてランプが点灯する前と点灯した後でさらに 2 つのタイミングに分けることができる。

ランプ点灯前に機能 B が開始した場合

「点灯フラグ」よりも「消灯フラグ」の優先度が高いので、ランプは消えたままであり、ランプ点灯中に機能 B が開始した場合、点灯していたランプがすぐに消灯する。一方、機能 B が先に動作しており、後から機能 A が動作した場合、ランプは消灯したままになる。

競合状態から抜けるタイミング

競合状態の「ホームポジション＝最下端、停止中」になると、機能 A はそのまま終了する。その後は、機能 B による動作に従って、「最上端移動要求応答、移動中」になる。

生成された機能競合に関するテストケース

以上の情報から機能競合になるタイミングとして、機能 A が先に動作おり、ランプ点灯前と点灯中の場合、機能 B が先に動作していた場合の 3 つのケースが考えられる。各ケースについて特定した前提条件、操作、期待動作から、図 3 に示すテストケースを得る。

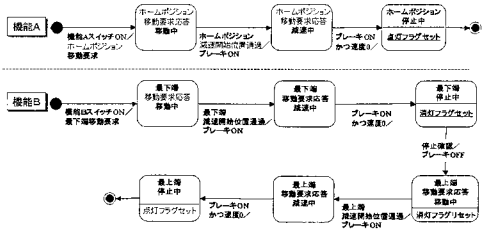

図 2 手法説明に用いる状態遷移図

前提条件：［機能A運転中。ホームポジション＝最下端移動要求応答、減速中］
操作：［機能B開始スイッチON］
期待される動作：［ホームポジション＝最下端、停止中になる。ランプは消灯したまま。その後、最上端への移動要求に応答し、最上端移動要求応答、移動中になる］

前提条件：［機能A運転中。ホームポジション＝最下端、停止中。ランプ点灯中のとき］
操作：［機能B開始スイッチON］
期待される動作：［ホームポジション＝最下端、停止中で点灯していたランプがすぐに消灯する。その後、最上端への移動要求に応答し、最上端移動要求応答、移動中になる］

前提条件：［機能B運転中。ホームポジション＝最下端移動要求応答中、減速中］
操作：［機能A開始スイッチON］
期待される動作：［ホームポジション＝最下端、停止中になる。ランプは消灯したまま。その後、最上階への移動要求に応答し、最下端移動要求応答、移動中になる］

図 3 得られたテストケース例

4.3 評価

ここでは同じく適用対象システムのある機能 C に対する機能競合を例に本手法の適用効果を示す。

機能 C は 17 種類のフラグと 4 つの状態を持ち、本手法 Step-2 の相反フラグによる絞り込みの結果、競合対象機能を 26 種類に絞り込むことができた。ここで 12 種類のフラグと 5 つの状態を持つ機能 D を競合対象機能として選択した。本手法 Step-3 によって機能 C と D の間で相反関係になっているあるフラグに注目し、同時にセットされ、競合状態になる状態を 1 つに絞ることができ, Step-4, 5 によって機能競合のテスト項目を作成することができた。

また、同様にしてこの制御ソフトウェアの機能のうち、実際に機能競合が起こる 3 組の機能に対して本手法を適用したところ、4 つの競合状態を特定することができた。これらについてもテスト仕様を作成したが、実際にこれらのテスト仕様をもとに対象システムを再レビューし、1 件のレアケースの機能競合不具合を検出できた。

5. まとめ

ソフトウェア内の機能競合を対象に、相反フラグや共有データに注目することで、機能競合が起こる状態を効率よく特定し、テストケースを作成する手法について述べた。

今後は、本手法の自動化などを進め、実際の開発プロセスに組み込むことを考えていく。

参考文献

[1]飯塚悦功監訳,実践ソフトウェア工学,日科技連出版
[2] 平山　他,"機能モジュールに対する優先度に基づいた選択的テスト手法提案",信学会01-5,pp1-8、2001