

繰り返し型ソフトウェア開発における試験効率化に関する一考察

榎田 隆行[†] 安井 照昌[†] 甲斐 啓文[†]

[†]三菱電機 (株) 先端技術総合研究所

1. はじめに

近年のソフトウェア開発における高機能化、短納期、仕様変更に対応するため、繰り返し型開発プロセスによるソフトウェア開発を採用するプロジェクトが増えている。繰り返し型開発プロセスによるソフトウェア開発は、新しい機能の追加時などに、実装済みの機能に影響がないことを回帰試験で確認する必要がある。一方で、ソフトウェアが大規模になると試験も大規模になり、試験の維持・管理が困難になる。そのため、低コストで試験を維持・管理する方法が求められている。

本稿では、大規模なソフトウェア回帰試験の維持・管理効率化方法の提案と、提案方法の実現確認として、GUI アプリケーションへの適用例を述べる。

2. 大規模試験の維持・管理効率化のポイント

大規模なソフトウェアは、回帰試験も大規模になるため、一般には試験をプログラムで記述し、自動実行することで、全件の回帰試験を確保する。しかし、仕様変更などにより試験自体の変更が頻発することがあり、試験プログラムを逐一記述する方法では試験の管理、全件自動回帰試験の維持が難しい。

我々は、作業が複雑で高コストになりやすい以下の 3 つの作業を単純化することで、大規模な自動回帰試験の実現が図れると考えた。

- (1) 試験シナリオの記述
- (2) 試験の事前条件の成立
- (3) 検証用の正解データの準備

なお、本稿の試験対象ソフトウェアは、操作前の状態が同じ場合に同じ操作を行うと、操作後の状態が同一になるソフトウェアとする。

3. 試験管理の効率化

2. 章に挙げた 3 つの作業を単純化するための方法を述べる。

3.1 試験シナリオの形式化

試験とは、1)事前条件の成立、2)対象の操作、3)状態の検証、の 3 つから成る (図 1)。これらを個別に機械可読なよう形式化すれば自動試験を構成することが可能になる。そこで、1)~3)をシナリオ要素として管理し、さらにシナリオ要素の連続で試験シナリオを構成する。これにより、シナリオ要素を別の試験シナリオに再利用でき、試験シナリオ作成を効率化できる。

1) 事前条件	試験を開始する前のソフトウェアの状態
2) 対象の操作	ソフトウェアに入力する操作
3) 状態の検証	ソフトウェアの状態を検証

図 1 形式化した試験シナリオ

3.2 試験シナリオ実行による事前条件の成立

試験を開始する前に事前条件を成立させる必要がある。この事前条件は、試験対象の機能で作り出すことが可能である。つまり、対象となる試験シナリオの事前条件は、別の試験シナリオを実行して作り出す。

3.3 試験実行による正解データの取得

検証に必要な正解データは、検証シナリオごとに準備する必要がある。しかし、試験が膨大になれば検証シナリオと正解データの数も膨大になるため、正解データの準備を簡素化しなければ、試験の大規模化は難しい。そこで、以下の検証方法を提案する。

- ・ 検証シナリオで得た出力データを保持する。
- ・ 保持した出力データと仕様を照らし、目視による正解判定を行い、正解データとして登録する。
- ・ 正解データ登録後に行う回帰試験では、検証シナリオで得た出力データと正解データを比較して検証する。

A study on method of effective test management
for iterative software development

Takayuki Kushida[†], Terumasa Yasui[†], Hirofumi Kai[†],

[†]Advanced Technology R&D Center,
Mitsubishi Electric Corporation

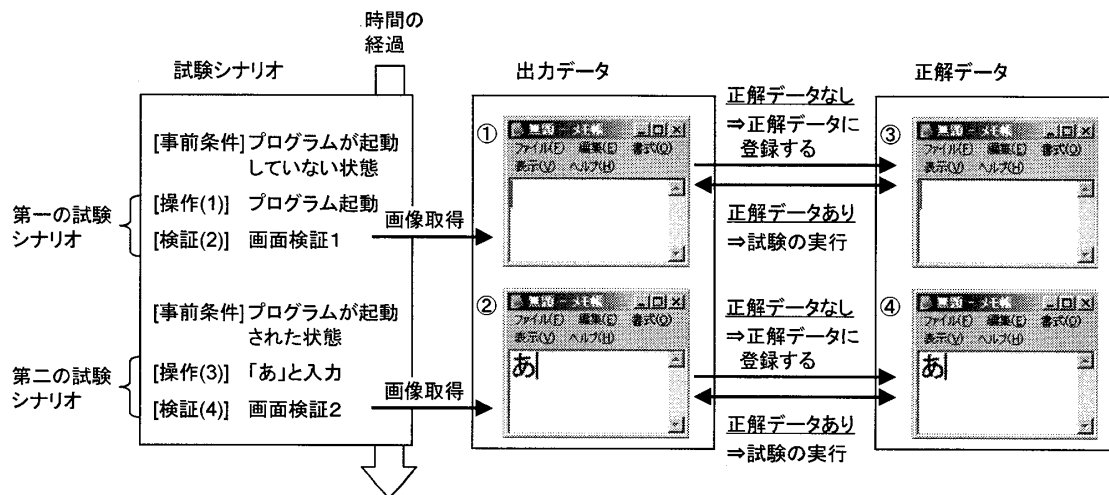


図2 GUIアプリケーション（Windows の notepad.exe）への適用例

4. 実践—GUIアプリケーションへの適用

3. 章で述べた試験管理方法をGUIアプリケーションに適用し、実現性を確認した（図2）。図2に示す試験シナリオは、それぞれ単独で存在し得る二つの試験シナリオで構成されるが、3.2 章で述べたように、第二の試験シナリオの事前条件は第一の試験シナリオを実行して作り出している。

試験シナリオは、4 つのシナリオ要素から構成され、操作シナリオ(1),(3)と、検証シナリオ(2),(4)を持つ。具体的な試験手順は(A)～(C)になる。

(A) 出力データの取得

試験シナリオの初回実行時は正解データが存在しないため、検証シナリオ(2)で出力データ①を取得し、検証シナリオ(4)で出力データ②を取得する。試験が終了すると、検証シナリオ数の出力データが得られる。

(B) 正解データの登録

検証シナリオ(2),(4)で取得した出力データ①,②を目視で確認し、操作シナリオ(1),(3)の操作を行った場合の出力データとして、正しければその画像を正解データとして登録する（③,④は登録した正解データを示す）。誤っていれば正しい出力データを出力するまで試験対象を修正する。

(C) 試験の実行と結果判定

回帰試験を行うには、再び試験シナリオを実行す

る。検証シナリオ(2),(4)で得られる画像は、前回と同一のデータが得られるはずである。そこで、検証シナリオ(2)で得られる出力データ①と、登録した正解データ③の画像比較、同様に、出力データ②と正解データ④の画像比較を行い、一致すれば試験成功と判定し、一致しない場合は、原因別に対処する。

同じ検証シナリオで同じデータが得られないのには主に二つの場合があり、以下に対処例を示す。

- 1) 試験対象に問題が発生している、つまりデグレードが起きている場合で、試験対象の修正が必要である。
- 2) 正解が二つ以上ある場合、例えばテキストエディタでカーソルがブリンクして場合で、どちらの画像も正解データとして登録すればよく、試験対象の修正は不要である。

5. まとめ

大規模なソフトウェア回帰試験の維持・管理効率化方法として、以下を提案した。

- (1) 試験をシナリオ要素として記述
- (2) 別の試験シナリオ実行による事前条件の成立
- (3) 試験を実行して得られた出力データを正解データとして登録し、以降の試験は出力データと正解データの比較から試験結果を判定

今後は、本方法を実開発に適用し、開発効率を計測する予定である。