

短納期でかつ小規模な開発プロジェクトにおけるテスト分析・設計導入による有効性の検証 Efficacy Verification by Introduction of Test Analysis and Test Design in Short Delivery Period's Software Development

リーダー：吉田 千鶴（株式会社ウイングアーク 1st 株式会社）
研 究 員：植木 誠太郎（テックスエンジニアリングソリューションズ株式会社）
岡田 ひろみ（ブラザー工業株式会社）
櫻庭 文吾（ソーバル株式会社）
根本 雄大（株式会社リンクレア）
主 査：秋山 浩一（富士ゼロックス株式会社）
副 主 査：上田 和樹（日本ナレッジ株式会社）
喜多 義弘（東京工科大学）

概要

短納期でかつ小規模な開発プロジェクトでは、たとえ、関心があったとしても、テスト分析・設計の手法を導入することに抵抗が大きい。なぜなら、それらの手法の導入によって想定外の工数が発生し、プロジェクトの納期が守れなくなるのではないかという不安があるためである。

本研究では、短納期でかつ小規模な開発プロジェクトにおけるテスト分析・設計の導入促進を目的として、テスト分析・設計の有効性を検証する。テスト分析・設計の効果把握の手段としては、HAYST 法のテストプロセスを用いた。

テスト分析・設計の有無によって、テスト工数やテストケース数に差が出るかの検証を行った。その結果、テスト分析・設計を行うことでテストの全体工数は増加するものの、テスト担当者ごとの工数ばらつきは減少し、かつ、テスト漏れを防ぐことができることが判明した。テスト分析・設計の有効性の効果を明らかにしたことで、短納期でかつ小規模な開発プロジェクトにおける導入促進が行いやすくなる。

1. はじめに

短納期プロジェクトの場合、アジャイル開発が現場で盛んに活用されている。また超短期開発プロジェクトにおいては、開発とユーザーとツールを信頼し、軽量品質保証を実現する方法が行われている[1]。

典型的なテストレベルである「サブルーチンテスト、単体テスト、新機能テスト、リグレッションテスト、統合テスト、システムテスト」のうち、規模が 1000FP 以下の新規開発プロジェクトでは「サブルーチンテスト」「単体テスト」「新機能テスト」の 3 つのテストレベルしか行われていないケースがある。しかも小規模なテストの場合、少人数の体制となることから、開発者によるテストが一般的である[2]。

研究員内でヒアリングした結果、研究員の 5 名のうち 3 名が小規模・短納期の案件に携わり、その 3 名中 2 名については開発担当者自身がテスト行っている。また 3 人全員がテスト分析やテスト設計に関心はあるものの実際はテスト分析やテスト設計をしておらず、またテスト分析・設計の効果が把握できていないことから、テスト分析・設計の導入が難しいという課題が存在することを確認した。

関心があるにも関わらず、テスト分析・設計の手法を導入することに抵抗が大きい理由について研究員でディスカッションを行った。その結果、手法の導入によって想定外の工数が発生し、プロジェクトの納期が守れなくなるのではないかという漠然とした不安感があるという意見があった。手法の導入が見送られると、テスト分析・設計の効果も分からないため、次のプロジェクトでも導入が見送られるという悪循環が繰り返されて、いつまでたってもテストの作り方が改善されないという問題がある。

このことから我々は、テスト分析・設計の有無による品質およびテスト工数が示されればテスト分析・設計の導入障壁が下がるのではないかと考えた。品質および工数効果の評価観点としては、工数、ケース数、テスト内容のばらつきを比較することとした。

またテスト分析・設計は、既にテストの効率性・品質確保が実証されており、導入しやすい HAYST 法[7]のプロセスを、短納期の小規模プロジェクトに導入することを考えた。

本論文の構成について述べる。2 章では、問題点の現状分析と関連研究について示す。3 章では、HAYST 法のテストプロセスが小規模プロジェクトへの導入に適しているかどうかを評価し、適用時の工夫を提案する。4 章では、テスト分析・設計の有無による有効性の比較実験の結果を示す。5 章はその考察を記載し、6 章はまとめと今後の課題である。

2. 現状分析

短納期の小規模開発における現状分析、および、テスト分析・設計プロセスの導入に対する先行研究を調査した。以下に調査結果を示す。

2.1 短納期の小規模開発におけるテスト工程の現状分析

表 1 は、開発プロジェクト内の工数別の実績月数の比率を示した表である[4]。全体と小規模を比較した場合、新規開発テスト工程にかける割合は 27.3%-33.4%=-6.1%、派生開発テスト工程にかける割合は 33.40%-36.20%=-2.8%と、小規模の方が低いことが読み取れる。

小規模開発の短納期プロジェクトにおいては、テストフェーズの工数は少ない。その結果、テスト設計をせずにテスト実装、または新機能テストや探索型テストなど、特定のテストのみになっているケースがあると考ええる。

表 1 工程別の実績月数の比率の基本統計量（平均値）

		基本設計	詳細設計	コーディング	結合&総合 テスト
新規開 発	全体	21.6%	18.9%	26.2%	33.4%
	小規模	22.6%	20.8%	29.3%	27.3%
派生開 発	全体	20.4%	17.5%	25.9%	36.2%
	小規模	20.1%	20.7%	25.9%	33.4%

2.2 先行研究

短納期の小規模開発におけるテスト分析・設計の有効性の検証を行うにあたり、どのようなテストプロセスを適用するかの検討を行った。

まずは適用するソフトウェアテストプロセスのモデルについて選定を行った。ソフトウェアテストに関する活動を開発活動と同等とみなして、「テスト開発プロセス」を定義している研究がある[5][6]。本論文ではそこで述べられている TDLC (Test Design Life Cycle) を参考とした。すなわち、上流からのテスト対象の分析漏れや、テストケースの抜けを防止するため「テスト要求分析」「テスト設計」「テスト実装」「テスト実施」のプロセスで行うことを考えた。

次に、テスト工数の割合が低いなかでも効率よくテスト分析・設計をすることを目指し、具体的に適用するテストプロセスについては HAYST 法に着目した[7][8][9]。しかし小規模プロジェクトで HAYST 法を利用した研究はない。だが HAYST 法の「テスト分析」では、“10 年以上の実務経験があるテスト担当者が HAYST 法のテスト分析資料を作成するに辺り、開発規模にもよるが 1 日程度で実施が可能である”ことが報告されている[10]。そのため「テスト分析」を導入するにあたり、作業効率の点で HAYST 法プロセスの有効性が確認できていると考えた。また“10 年以上の実務経験があるテスト担当者”ではなく、テスト分析や設計の経験が浅いメンバーでも適切なテスト設計が行えるよう、導入が行いやすいことを考慮した上での解決策を検討する。

3. 提案

本章では、既存のテスト分析・設計手法である HATST 法のテストプロセスが、小規模プロジェクトへの導入に適しているかを評価し、さらに導入時における工夫を提案する。

3.1 HAYST 法プロセス導入における習得時間の評価

テスト分析・設計を行っていないプロジェクトに、HAYST 法のテスト分析・設計を導入すると、図 1 の HAYST 法のプロセスを追加することとなる。テスト分析・設計経験の有無は半々である研究員で、参考文献[9]を元に HAYST 法のテスト分析・設計の各プロセスで定められている資料を作成し、導入のしやすさを評価した。

図 2 に「テスト分析」のための 6W2H, FV 表 (Function Verification Table), 「テスト設計」のためのラルフチャート, FL 表 (Factor Level Table) の習得時間結果を示す。

その結果、HAYST 法のプロセスでは、テスト要求分析やテスト設計の手順にそれぞれ対応したテスト分析・設計用のフォーマット[9]に従って作成することができるため、テスト分析や設計を行ったことがないメンバーであっても習得にかかる時間は 2 日程度であることが確認できた。また資料ごとにレビューを実施し進めることで、対象の理解が十分かのチェックを並行して行えることがわかった。

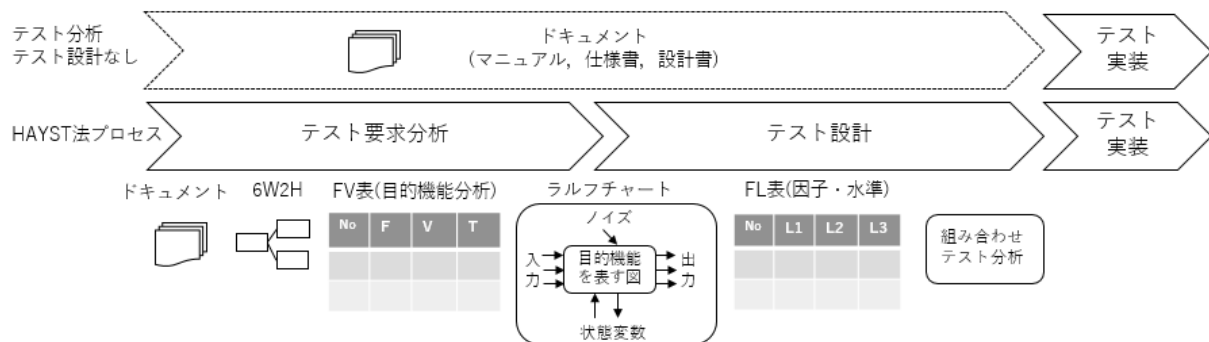


図 1 テスト分析・設計がない場合と、HAYST 法テストプロセスの違い

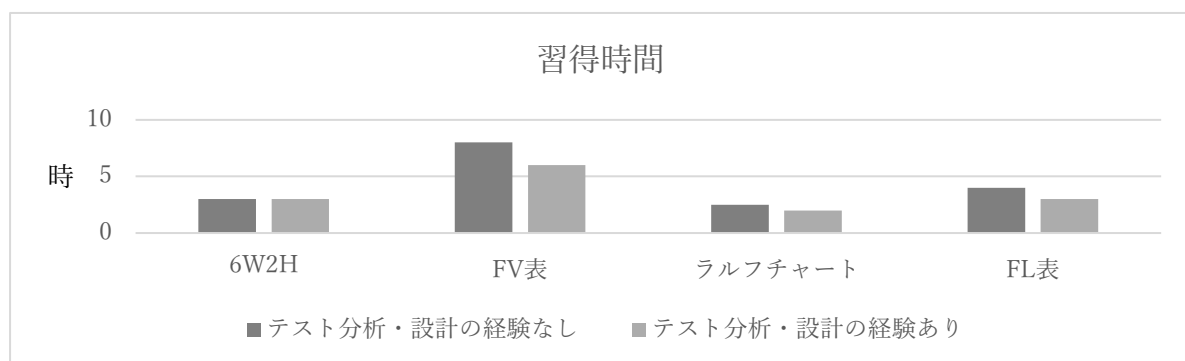


図 2 HAYST 法のテスト分析・設計手法の習得時間

3.2 HAYST 法の FV 表を導入するための工夫

FV 表は HAYST 法によるテスト要求分析の手法であり「システムを機能が果たす目的の切り口で分割した単位 (以降、目的機能と表記)」で分解・整理を行うことで、目的機能(F)ごとに検証すべき条件(V)を一覧にまとめた表である。この FV 表について小さなサンプルを用いて試用してみたところ、FV 表を初めて記入した研究員 5 名中 4 名が「目的機能」を記載していたはずが、「要求」を誤って記載するという問題があった。そこで原因分析のため研究員が書き出した内容で誤った箇所を洗い出し、問題点についてのヒアリングを行った。

- 製品が果たすべき役割が意識できておらず、そのまま「要求」を書き出していた
- 書き出している内に、意識が機能仕様書の記述に引きずられ「要求」に変化した

FV 表を記載するためには「機能」を取り出し、次にその機能が持っている「目的」について考え、その機能について製品が果たすべき役割を考える必要があるが、「要求」はリリース時に満たすべきもので、「目的」は利用者が使い始めてサポートが終了するまで満足させるべきものである[8]。その機能が果たすべき目的を予測し、「目的機能」として見落とさないように全て記載する必要があるが、経験が浅いと「目的機能」を記入していたつもりであっても「要求」を記載してしまうケースがあると判明した。

FV 表について、研究員全員で記載した内容についての確認と、記載した内容についての認識合わせが必要であると考えた。そこで表 2 の通り FV 表に「対象可否（記述した目的機能が、内容に相違ないかを判断する列）」を追加し、レビュー時にチェックを行うことで、記載したテスト対象物の整理が行いやすいようにした。

表 2 レビュー版 FV 表

No.	対象可否	目的機能(F)	検証内容(V)	テスト技法(T)

3.3 HAYST 法プロセスの適用手順／レビュー版 FV 表の適用手順

HAYST 法プロセスのテスト分析・設計に関する適用手順は以下の通りである。なお①～⑥のステップで作成した各成果物については FV 表の[No.]を元に、トレーサビリティを確保するように番号を一致させて作成を行う。

- ① 機能仕様書などのテストベースとなる資料を元に、「6W2H」のツリーを作成する
- ② 機能仕様書などのテストベースとなるドキュメントを元に、レビュー版 FV 表(表 3)を作成し、テストする部分とテストしない部分を明らかにする。
- ③ レビュー版 FV 表に記述した目的機能に対して、関係者でレビューを行い、「目的機能」として記載に相違がないか、記載した内容をテストするかどうかを判断する。その結果を「対象可否」の列にてチェックをつけ、内容の合意をとる
- ④ レビュー版 FV 表でテストすると決定した目的機能に対し、テストする因子・水準の抜けがないかをレビューするために「ラルフチャート」を作成する。
 - 小規模の機能で目的機能が複雑ではない場合は、作成対象外としても良い
 - ラルフチャートを作成後、FV 表の変更があれば再度レビューを行う
- ⑤ 組み合わせのテスト技法を適用すると決めた目的機能に対して、その入力に当たる因子・水準を確定するための「FL 表 (Factor Level Table)」を作成する
- ⑥ 作成した FL 表を元に、テストケースの実装を行う

4. テスト分析・設計の有無の有効性の評価

短納期の小規模開発においてテスト分析・テスト設計の有無によるテストの有効性の差異を比較するため、テスト分析・設計を行っていない場合のテスト実装と、HAYST 法のプロセスにてテスト分析・設計を行った場合との有効性の差異を比較する実験を行った。実験を始めるにあたり評価対象物の選定を行った。短納期の小規模開発を想定し、機能が複雑でなく、組み合わせ評価も必要となることを考慮し、タスク管理や連絡先ツールでの検討を行った。また利用経験を同一とするため研究員全員が利用したことがないアプリケーションであり、拠点が異なる研究員全員がアクセス可能であることを条件とした結果、「Google 連絡先」の「新しい連絡先登録画面」(図 3)を評価対象物に決定した。「Google 連絡先」は Gmail, カレンダー, ドライブなどの Google サービスに統合されているオンラインのアドレス帳である。

「Google 連絡先」は世の中に出回っているツールであるため、今回は障害の発見ではなく、

テスト分析・設計の有無によるテスト実装数の比較と所要時間の計測を主目的とした。

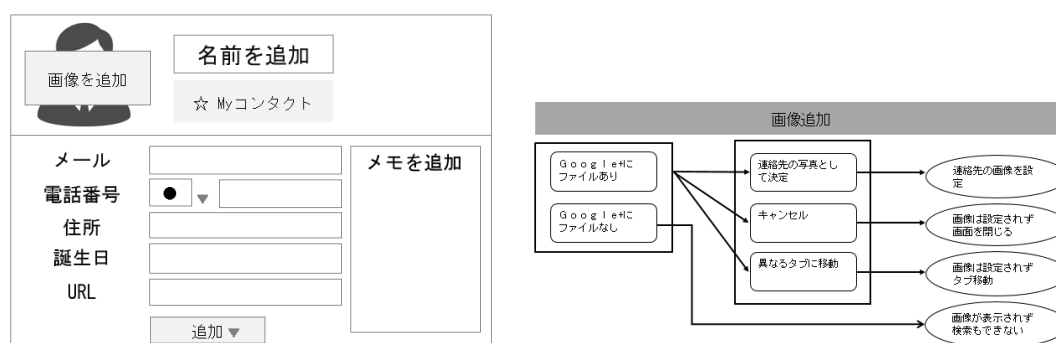


図3 「Google 連絡先」の画面イメージ

4.1 評価対象

実験を行うにあたり、評価対象を下記に限定した。

- 「Google 連絡先」は評価対象が膨大なため、新規登録に関する機能動作のみを評価対象とする。パフォーマンスやセキュリティなどの非機能、環境に関連する動作(インストール、環境条件、アクセス時間、言語設定)は評価対象外とする。
- 小規模の新規開発として「Google 連絡先の新しい連絡先」を対象とする。

4.2 被験者の詳細

被験者の年齢、テスト業務歴、テスト戦略経験、テスト設計経験、テスト実装経験
主業務は表3の通りである。全員がテスト実装経験者であるが、テスト設計経験者は2名であった。

表3 被験者の詳細

	年齢	テスト 業務歴	テスト 戦略経験	テスト 設計経験	テスト 実装経験	主業務
被験者1	30代	10年	あり	あり	あり	品質保証
被験者2	40代	14年	なし	なし	あり	運用保守
被験者3	40代	15年	あり	あり	あり	品質保証
被験者4	30代	9年	なし	なし	あり	運用保守
被験者5	30代	8年	なし	なし	あり	品質保証

4.3 実験内容

[A. テスト分析・設計なし]の場合、テストベースとなるマニュアル、「Google 連絡先」の画面UIを確認後、機能の目的を考えずにテスト項目になるであろう機能を仕様書からテストケースに転記し作成を行った。一方、[B. テスト分析・設計あり]の場合は、テストベースを確認後、レビュー版FV表を使用して機能の目的を考えた。その上で目的の認識合わせを研究員全員で実施し、その後3.3に記載した手順でテストケースを作成した。

4.4 実験結果

実験結果を表4,5に示す。

表の項目の説明は以下の通りである。

- [分析]とはレビュー版FV表にて、目的機能の洗い出し作業を行う時間である。
- [設計]とはレビュー版FV表をもとにテスト設計した時間である。
- [実装]とはテスト設計で検討したテストケース作成方針に従ってテストケースを作成する時間である。

表 4 ではテスト分析から実装までにかかった平均時間と、標準偏差を比較データとして示す。[A. テスト分析・設計なし]と [B. テスト分析・設計あり]を比べると、[B. テスト分析・設計あり]ではテスト分析・設計のプロセスが増えたため、テスト実装に要する工数以上の時間がテスト分析とテスト設計にかかった。テスト実装の時間は[A. テスト分析・設計なし]が 13.7h, [B. テスト分析・設計あり]が 29.6h とテスト設計を行っていない場合の方が少ないことが読み取れる。しかし被験者の標準偏差については [A. テスト分析・設計なし]は 7.1, [B. テスト分析・設計あり]は 2.3 と、テスト分析・設計を行った方が工数のばらつきの幅は小さい結果となった。

表 5 では目的機能別に、被験者別のテスト実装数の平均個数と、標準偏差を示す。[A. テスト分析・設計なし]では、目的機能の一つである[全体機能 3]の実装数が、被験者全員 0 件となった。また[住所]や[追加項目 2]の目的機能では、テスト実装数の標準偏差は 50 以上と、各被験者が作成した実装件数で大きく差異が発生した。[B. テスト分析・設計あり]ではテスト実装数が 0 件となる機能は存在せず、標準偏差についても差異の幅が小さい結果となった。テスト分析・設計の有無による平均実装数の差異では、13 機能中の 12 機能で [A. テスト分析・設計なし] よりも [B. テスト分析・設計あり]のテスト実装数が多く、さらに 5 機能で 40 件以上もの実装数の差異が発生した。

表 4 工数の比較データ(被験者の平均時間・標準偏差)単位:時

	分析	設計	実装	合計工数	標準偏差
A. 分析・設計なし	0	0	13.7	13.7	7.1
B. 分析・設計あり	8	8.2	13.4	29.6	2.3

表 5 目的機能別のテスト実装数 (被験者の平均実装数・標準偏差)単位:個

機能	A. 分析・設計なし		B. 分析・設計あり		平均実装数の 差異[B]-[A]
	平均値	標準偏差	平均値	標準偏差	
全体機能 1	1.6	3.2	78.6	2.3	77
全体機能 2	18.2	29.2	78.2	5.4	60
全体機能 3	0	0	27	6.2	27
画像	18.8	13.6	24.2	3.1	5.4
名前	27.4	15.6	97	4.7	69.6
メール	15.4	5.9	23.2	1.9	7.8
電話番号	35.4	23.2	44.2	2.2	8.8
住所	53	50.8	58.6	1.2	5.6
誕生日	9.4	8.2	11	1.7	1.6
URL	18.8	11.4	84.6	1.2	65.8
メモ	4.4	2.9	47	2.8	42.6
追加項目 1	2.4	4.8	22.4	1.6	20
追加項目 2	131	112.4	97.6	14.2	-33.4

5. 考察

テスト分析・設計の有無による実験結果にて、大きく以下 2 点の違いが判明した。

1. [テスト実装数・テスト工数のばらつき]の差異について

表 4, 5 の通り [A. テスト分析・設計なし]は被験者によるばらつきが大きかった。とくに[住所]や[追加項目 2]については標準偏差が 50 以上と、大きな差異が発生した結果となった。詳細情報や追加入力をどの範囲まで実装するか、テスト分析・設計に該当する部分を個人の判断で行ったため、テスト内容や粒度に差が出たことが原因と言え

る。実際には不足している観点がある場合でも個人で完成したと判断してしまっている。つまり、[A. テスト分析・設計なし]の場合のテスト内容はテスト実装を行う人の経験やスキルに依存すると言える。

[B. テスト分析・設計あり]では合計工数、目的機能別のテスト実装数とも差が少なかった。また表 5 の通り、[A. テスト分析・設計なし]よりも[B. テスト分析・設計あり]の方が[追加項目 2]以外の目的機能のテスト実装数の方が多く、かつ全ての目的機能別のばらつきが小さい結果となった。これはテスト分析を行い、共通のレビュー版 FV 表を元にテスト設計を行ったため、テスト内容や粒度の差が少なかったと考えられる。さらに被験者の経験に関わらず、ばらつきが少なかったためテスト設計工数やテスト実装数の見積もりもしやすいと言える。

2. [テストケースの内容]の差異について

テストケースの内容については、他の被験者から以下の意見が挙がった。

- [A. テスト分析・設計なし]では利用目的を検討していなかったため影響しない項目を組み合わせて 1 つの連絡先として利用する、というテストが漏れた
 - [A. テスト分析・設計なし]では見た目の操作、動作にとらわれすぎて利用者視点に立てていなかった
 - [A. テスト分析・設計なし]では「なぜその機能が必要か」を考えていなかった
- 表 5 の結果と被験者の意見を照らし合わせると、テスト実装数だけでなくテスト内容にも大きく差が出たことがわかる。[A. テスト分析・設計なし]では機能の目的や利用者の立場でどう使うか、といったテスト分析段階からの視点が漏れてしまい、結果として[全体機能 3]自体の欠落に繋がった。[B. テスト分析・設計あり]で気づくことができた理由は、レビュー版 FV 表で利用者の目的機能を満たすことを検討したためである。これは HAYST 法を導入した際にレビュー版 FV 表にて、目的機能を要求と混同しないようレビューして洗い出す手順を確保したことが、一番の成功要因であると考えられる。テスト分析段階で製品や機能を果たすべき役割を全員で検討を行い、識合わせを行ったことがテストの漏れ防止に繋がったと考える。

また表 5 の通り[追加項目 2]の目的機能は、[B. テスト分析・設計あり]の方が、[A. テスト分析・設計なし]のテスト平均実装数よりも少なく、ばらつきも少ない。[追加項目 2] は組み合わせが主体となる目的機能である。[B. テスト分析・設計あり]では、FL 表でのテスト設計にて因子・水準を洗い出し、組み合わせ内容を検討して実装を行った結果、ばらつきを抑えつつも、[A. テスト分析・設計なし]より効率的にケースの絞り込みが行えたと考える。

6. まとめ

6.1 に実験結果と考察による結論を、6.2 に今後の課題を示す。

6.1 結論

短納期の小規模開発プロジェクトに関わる研究員全員がテスト分析やテスト設計に関心はあるものの、実際にはテスト分析・設計を行っておらず、テスト分析・設計の効果が把握できていないことから導入が難しいという課題があった。このことから我々は、短納期の小規模開発を想定したテスト分析・設計の有無によるテストプロセスを比較し、品質確保の有効性を測定する研究を行うこととした。HAYST 法のプロセスを適用したケースと、テスト分析・設計を行わずテスト実装した場合のケースと工数やテストケース数の比較を行い、テスト項目のばらつきの差異による結果から、テスト分析・設計を行ったことでテスト漏れを防ぐことができることが判明した。またテストの全体工数の差異による結果からは、同時にテスト設計工数やテスト実装数の見積もりが行いやすくなることが判明した。

さらにテスト分析・設計を行わずテスト実装した場合に、テスト分析段階から目的機能

自体が欠落するケースがあった。目的機能自体の欠落は、利用目的を達成できないというバグに繋がり、重大な手戻りとなる可能性が想定される。事前にテスト分析・設計を行うことで気づけなかったテストに気づくことができ、テスト漏れを防ぐことで手戻り工数が減ることが期待される。

テスト分析・設計を行うことでテストの全体工数は増加するものの、機能の目的や利用者の立場でどう使うかという視点でのテスト項目の漏れ防止や、工数予測を行いやすくなった結果から、テスト分析・設計の有効性が明らかになった。本結果により、短納期でかつ小規模な開発プロジェクトにおける導入促進が行いやすくなると考える。

6.2 今後の課題

本研究によりテスト分析・設計を実施することの有効性を確認することができたが、今後の課題として以下の3点を挙げる。継続して取り組むことで更なる効果が期待できる。

1) 合計工数の改善

テスト分析・設計を行うことでトータルの工数が多くなっている。短納期でテストを完了させなければならないためかけられる工数は限られている。新規開発時に作成したテスト成果物を利用した派生開発では工数が削減となるかなど、テスト対象の規模に対する工数の妥当性について、さらに検討する必要がある。

2) 短納期開発の粒度にあわせたテスト分析・設計の検討

例えば2週間の短納期と3週間の短納期では、テスト分析・設計の導入にかかるインパクトが異なってくる。またテストプロセスの初期導入時と、初期導入後の工数は異なってくると考える。短納期開発の粒度にあわせたテスト分析・設計の導入を検討する必要がある。

3) 利用目的のユーザー定義について

今回の実験対象である Google 連絡先は、機能面に注力する形での実施であり、複数の環境など複雑な要因は考慮されていない。またシステムの利用者であるユーザーは自身の1名だけとなることから、今回の研究ではユーザーの定義まで行っていない。ユーザーの定義が異なるとその機能の使われ方、つまりテストの目的も異なってしまう。そのためユーザーの定義をテスト関係者間で事前に合意しておく必要がある。

参考文献

- [1] 松木 晋祐, 超短期開発プロジェクト/ 非クリティカルシステムに向けた軽量品質保証 概念とその最上流プロセスの提案, SQiP シンポジウム 2013, pp.1-20, 2013.
- [2] Capers Jones, Olivier Bonsignour, ソフトウェア品質の経済的側面, 構造計画研究所, 共立出版社, 2013.
- [3] 清水吉男, 「派生開発」を成功させるプロセス改善の技術と極意, 技術評論社, 2007.
- [4] 独立行政法人情報処理推進機構 (IPA) 技術本部ソフトウェア高信頼化センター (SEC), ソフトウェア開発データ白書 2016-2017, 独立行政法人情報処理推進機構, 2016.
- [5] 西康晴, テスト観点に基づくテスト開発方法論 VSTeP の概要, JaSST' 13 Tokyo, pp. 17-20, 2013.
- [6] 水野 昇幸, テスト詳細設計プロセスの手順定義によるテストケースの抜け防止, ソフトウェア品質シンポジウム 2017, pp. 1-8, 2017.
- [7] 吉澤正孝, 秋山浩一, 仙石太郎, ソフトウェアテスト HAYST 法入門 品質と生産性がアップする直交表の使い方, 日科技連出版社, 2007.
- [8] 秋山 浩一, 事例とツールで学ぶ HAYST 法—ソフトウェアテストの考え方と上達のポイント, 日科技連出版社, 2014.
- [9] 秋山 浩一, 効率的にソフトウェアの品質向上を実現する HAYST 法の適用, 2015.
- [10] SQiP25 期第5分科会 品質チーム, 派生開発におけるテスト分析手法の提案 — ラルフチャートを用い, 変更前後の差分に着目するテスト分析 —, SQiP 研究会, pp. 1-8, 2016.