見通しのよいテストの段階的詳細化の手法
－テストの網羅性確保の提案－

吉岡 克浩† 水野 昇幸‡ 西 康晴†††

†三菱電機株式会社 設計システム技術センター 〒661-8661 兵庫県尼崎市塚口本町 8-1-1
‡三菱電機株式会社 通信機製作所 〒661-8661 兵庫県尼崎市塚口本町 8-1-1
†††電気通信大学 情報理工学研究科 〒182-8585 東京都調布市調布ヶ丘 1-5-1
E-mail: †Yoshioka.Katsuhiro@ab.MitsubishiElectric.co.jp ‡Mizuno.Noriyuki@ah.MitsubishiElectric.co.jp

あらまし 段階的にテストを詳細化する「テスト開発プロセス」を構築した。テストの網羅性向上のためにマトリクス形式を導入したが、テストケースのばらつきや成果物が巨大化し作業性が悪化する課題があった。そこで、モデル化したテスト対象とテスト目的の組合せでテストアーキテクチャを決定し、組合せ毎にテストを詳細化する手法を作成した。手法の適用により、テスト密度向上と作業効率化を両立し、テストケースのばらつきを抑制した。

キーワード テストアーキテクチャ、テスト対象、テスト目的、テスト観点、マトリクス、デシジョンテーブル

Test Design by Stepwise Refinement
－Approach to Test Coverage Improvement－

Katsuhiro Yoshioka† Noriyuki Mizuno‡ Yasuharu Nishi†††

† Design Engineering Center, Mitsubishi Electric, Ltd., ‡ Communication Network Center, Mitsubishi Electric, Ltd.,
††† Department of Informatics,University of Electro-Communications
E-mail: †Yoshioka.Katsuhiro@ab.MitsubishiElectric.co.jp ‡Mizuno.Noriyuki@ah.MitsubishiElectric.co.jp

Abstract We were established to refine the contents of the test in a stepwise manner "Test Development Process."
By applying this process, the test design work can be done by dividing the unit of test context, we have improved the deterioration of working with the large test matrix for the conventional process. And we have improved both efficiency of test design and test density.

Keyword Test Architecture, Test Item, Test Context, Test Viewpoint, Test Matrix, Decision Table

1. はじめに

情報家電、自動車、産業機器などの工業製品のほとんどは機能や性能の多くを、機器に組み込まれたソフトウェアで実現している。市場競争の激化に伴う製品の高機能化により、組込みソフトウェアはより大規模で複雑なものとなってきている。一方で、製品投入間隔の短縮により、開発期間は短縮してきている。

このような状況で弊社は、納期遅延、コスト超過によるプロジェクト崩れを抑制しつつ市場に高品質な製品を提供するために、設計段階で品質作り込みを行う活動とテストの改善を推進してきた。

1.1. 従来のテスト設計手法の課題

従来のシステムテスト設計は、仕様書から直接テスト手順書を作成し、レビューでテストケースの妥当性を確認する作業であった(第１世代)。テスト手順書は、階層化（主に大中小）したテスト対象機能に対し、テスト手順、期待値を記述したものであり以下のような課題があった。

(1)テストケースの作成基準、網羅基準がなく、テストケースの粒度、記載レベルのばらつきがでる。

(2)テストケースの羅列からは全体像が見えず、何を網羅できているか分からないため、レビューで漏れを検出できない。

(3)テストケースの数が多く限られた時間でレビューしきれない。

これらの課題に対し、テスト手順書作成の前に、機能(行) x テスト観点(列)のマトリクスでテストケースの洗い出し、抜け・漏れのチェックを行う、ゆもつよメソッド[1]を参考にした手法を導入した（第２世代)。マトリクス手法の導入により、大きなテスト観点の抜け・漏れを抑制することはできたが、新たな課題も見つかった。本論文ではそれらの課題を明らかにし、課題への対策（第３世代）について記述する。

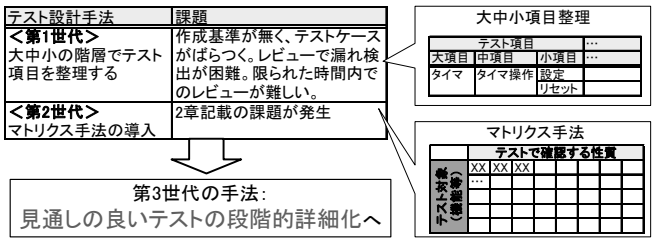


図 1　テスト設計手法の変遷

2. マトリクス手法（第２世代）導入後の課題

2.1. 巨大なマトリクスになり作業性が悪い

ある工事に適用した結果では、階層化した機能：200 行 x テスト観点：15 列の巨大マトリクスとなった。その結果、全体を俯瞰することが困難となりレビューや作業分担がし辛い状況となった。

2.2. テスト観点の定義が不十分

初期に導入したマトリクスでは標準のテスト観点とその説明文書を定義して展開したが、ドメイン共通で使用する狙いとしたためテスト観点の抽象度が高く、作業担当者により解釈が異なる場合があり、抽出結果にばらつきが出た。また、記載の詳細度が個人任せであったため、詳細に記載してテスト手順書と内容が重複したり、逆に一部のキーワードのみの記載で中身が良く分からない資料になってしまうケースがあった。

2.3. マトリクス作成後の作業が不明確

マトリクスは仕様書からテストのポイントを抽出する作業に使用し、テストケース作成のインプットとなる資料であるが、テストケースに落とすまでに組合せやテスト準備手順を追加する方法は個人任せであったため、テストケースの記載内容、粒度が一定でなかった。

3. 課題に対する対策

3.1. 課題の対策方針

これら３つの課題に対し以下の対策方針を立てて解決に取り組んだ。

表 1　課題の対策方針

	課題	対策方針	対策
1	巨大なマトリクスになり作業性が悪い	マトリクスを見通しのよいサイズに分割する	3.2項に記載
2	テスト観点の定義が不十分	テスト観点を具体化する	3.3項に記載
3	マトリクス作成後の作業が不明確	マトリクスからテストケースに落とす作業を具体化する	3.4項に記載

3.2. テスト対象とテスト目的によるテスト全体の分割

(1)テスト対象

テストの全体を考えるには、テストの対象と非対象を明確にする必要があった。マトリクス手法で扱っていた「機能」もその１つであるが、実際にシステムテストで扱っていた対象は、「ユースケース」「実運用シナリオ」「システム全体」といった粒度の大きなものもある。これらテスト対象の例を表 2に示す。システムテストで扱うテスト対象は、インプット資料に記載があるものがベースであり、客先 RFP、システム要求仕様書からは、ユースケースや実運用シナリオに似た粒度の情報が抽出でき、機能仕様書からは機能を抽出できることが多い。しかし、インプット資料の質・量は客先、プロジェクトごとに一様ではなくテスト設計時にすべてが開示されていないケースもある。そのためテスト対象として必要であれば上位仕様書作成者に提示を求めたり、テスト設計者自らユースケース一覧や運用を想定したシナリオを作成する。テスト対象は、プロジェクトによって要不要や重み付けを判断するだけでなく、新しい項目を追加するといったテーラリングを可能としている。

表 2 テスト対象の例

	テスト対象	概要
1	機能	S/W要求仕様書の機能
2	ユースケース	S/W要求仕様書のユースケース（複数機能の連携）
3	実運用シナリオ	運用で想定されるまとまった作業（複数機能、ユースケースの連携）
4	システム全体	システム全体としての動作
5	利用シーン	網羅的な条件で検証したい特定の状況
6	データ	データ保守、システム移行時の対象
7	プロトコル（シーケンス、信号）	シーケンス、信号
8	状態	状態定義、状態遷移

(2)テスト目的

テスト目的は、テストで保証すべき性質の分類であり、本手法ではテスト観点の上位に位置するテストの「狙い」を示すものである。表 3にテスト目的の例を示す。手法の確立においてはテスト目的の漏れ＝手法によるテスト漏れに繋がるため、テスト目的の議論に時間をかけた。テスト目的に挙げた項目はマトリクス手法で用いていた項目をベースに、不具合事例からシステムテストで検出すべき不具合を「狙う」テスト目的を追加していった。

表 3 テスト目的の定義

テスト目的カテゴリ	テスト目的	狙う不具合
品質特性	1 論理性	仕様不適合の動作。仕様通りの結果にならない。条件通りに動作が変化しない。条件の裏(else)で不正な動作をする。
	2 ユーザビリティ	仕様不適合のユーザビリティ。使いにくいUI。ユーザ操作時のレスポンスが悪い。表示の一貫性がない。
	3 性能	仕様不適合の性能。速度性能未達。リソースの枯渇。
	4 保守性	仕様不適合の保守性。解析しにくいログ。膨大な量のログ。短時間で消えるログ。更新しにくいマスタ。更新時間がかかる。装置交換が困難。
	5 信頼性	オーバースペックな条件で発生する不具合。
	6 移行	仕様不適合のデータ移行性。旧バージョン・別環境からのデータ移行が困難。データの追加・変更・削除が困難。
依存性	7 組合せ	仕様上影響がないはずの機能・データ・状態の組合せによる不具合。(直交表、HAYST法の適用範囲)
	8 環境	H/W、対向装置、気温、地域性など環境・構成を変更して発生する不具合。仕様不適合のもの(環境の違いに実装で対応しているもの)。仕様上影響がないはずのもの。
非ロバスト	9 通常負荷	通常負荷での動作不具合。速度性能未達。
ロバスト性	10 意地悪	意地悪操作による動作不具合。ボタン連打、同時押し、同時実行、回線切断などによる想定外の不具合。
	11 設計最大負荷	設計最大負荷での動作不具合。負荷に耐えられず動作が中断。想定以上の性能劣化。
	12 過負荷	過負荷での動作不具合。リソース枯渇によるクラッシュ。無応答。フェールセーフ機能の動作不具合。
	13 連続運転	連続運転での動作不具合。想定以上のメモリ使用、リソース使用。性能劣化。

(3)テスト対象とテスト目的の組合せ

テスト対象にテスト目的を組合せることでどのテスト対象で何の性質を保証するテストをする、というテストの構成を示す。本手法ではこの組合せ表を「テスト設計サマリ表」と呼び、その構成要素である1つ1つの組合せを「フレーム」と呼ぶ。表 4はテスト設計サマリ表の例である。この表は組合せを「○」の有無で表示する単純な表であり全体を容易に俯瞰することができる。テスト設計サマリ表で示す内容は、テストの専門家でなくとも短時間の説明で理解可能である。そのため、システム設計者、S/W 設計者など関係者を交えてテストの範囲を議論して合意するツールとして適している。

表 4 テスト設計サマリ表の例

テスト目的 / テスト対象	品質特性				
	論理性	ユーザビリティ	性能	保守性	….
1 機能	○		○		
2 ユースケース	○	○		○	○
3 実運用シナリオ	○			○	
4 …	○		○	○	○

フレーム

3.3. テスト観点の具体化と因子の抽出

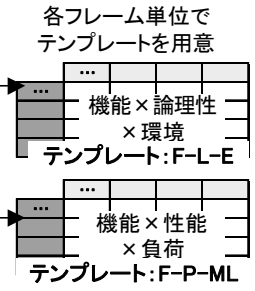
テスト構成が決定したら、次は具体的にインプット仕様書から入力値、イベントなどの「テスト条件」と振る舞い、判定基準などの「チェックポイント」を抽出する。また、テスト条件、チェックポイントになり得る仕様書に記載された項目を「因子」、因子の取り得る選択肢を「水準」と呼ぶ。因子と水準の考え方はHAYST 法[2]を参考にした。

(1)フレームごとのマトリクス定義

従来のマトリクス手法ではテスト条件およびチェックポイントの抽出結果を1つ巨大マトリクスに記載していた。本手法では、作業者の分担やレビューをしやすくするために、フレームごとにマトリクステンプレートを分割して定義した。表 5はフレームごとのマトリクステンプレートの対応イメージである。

表 5 フレーム毎のマトリクステンプレート

NO	テスト対象	テスト目的		ID
1.1	機能	論理性 L	環境 E	F-L-E
1.2	F	性能 P	設計最大負荷 ML	F-P-ML
2.1	ユースケース UC	論理性 L		UC-L
2.2		ユーザビリティ U		UC-U
2.3		保守性 MA		UC-MA



(2)テスト観点マトリクス

マトリクステンプレートは、行見出しに具体化したテスト対象、列見出しにテスト目的を具体化したテスト条件、チェックポイントを持つ。本手法ではこのマトリクスをテスト観点マトリクスと呼ぶ。表 6はテスト対象：機能、テスト目的：論理性・環境、のテスト観点マトリクスの例である。

表 6 機能 x 論理性 x 環境のテスト観点マトリクスの例（テンプレート：F-L-E）

列見出し：テスト条件、チェックポイント

テスト観点番号	テスト対象	テスト条件				チェックポイント			
		環境	論理			論理			
	機能	環境	入力/参照値	入力/参照状態	入力イベント	中間値、中間状態	結果値	振る舞い	事後状態
F-L-E-01	給湯機能	ポットシリーズ(ポットA、ポットB)	優先度	ロック状態(ロック、解除) 水量(空、適量、満水超え) 蓋センサ状態(ON、OFF) *1 温度制御状態(保温、沸騰、アイドル) エラー状態	給湯ボタン押下	#1 給湯不可能状態(L1、L2、L3、L4)	表示コード(L1、L2、L3、L4)	給湯(実施/非実施) エラーコード表示(L1、L2、L3、L4)	元の状態であること。
F-L-E-02	タイマセット機能								
F-L-E-03	タイマリセット機能								
F-L-E-04	タイマアラーム機能								
F-L-E-05	沸騰機能								
F-L-E-06	カルキ抜き機能								
F-L-E-07	保温機能	ポットシリーズ(ポットA、ポットB)	なし 水温	保温状態	なし	なし	なし	保温ランプ点灯 沸騰ランプ消灯	保温状態
F-L-E-08	モード設定機能	ポットシリーズ(ポットA) 外気温(低温、高温)	なし	設定モード(高温、節約、ミルク)	なし	なし	水温(高温、節約、ミルク)	なし	なし

セルの中身：テスト条件、チェックポイントの因子と水準

行見出し：具体化したテスト対象

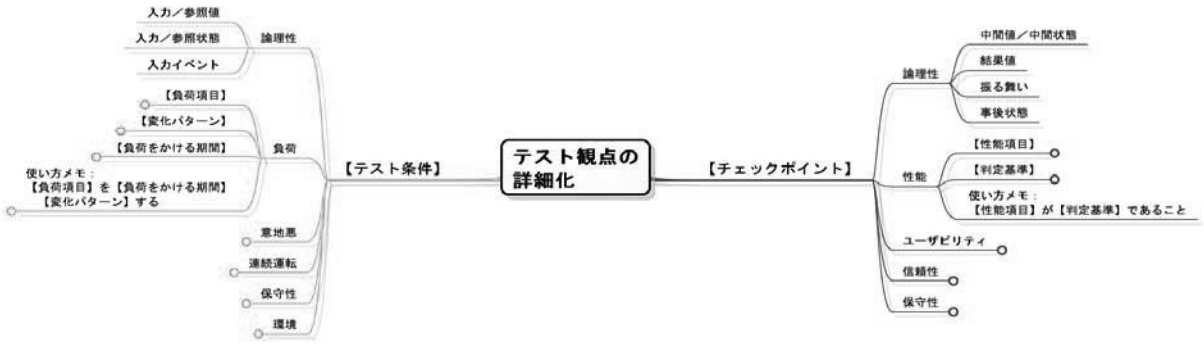


図 2テスト条件、チェックポイントの詳細化の例

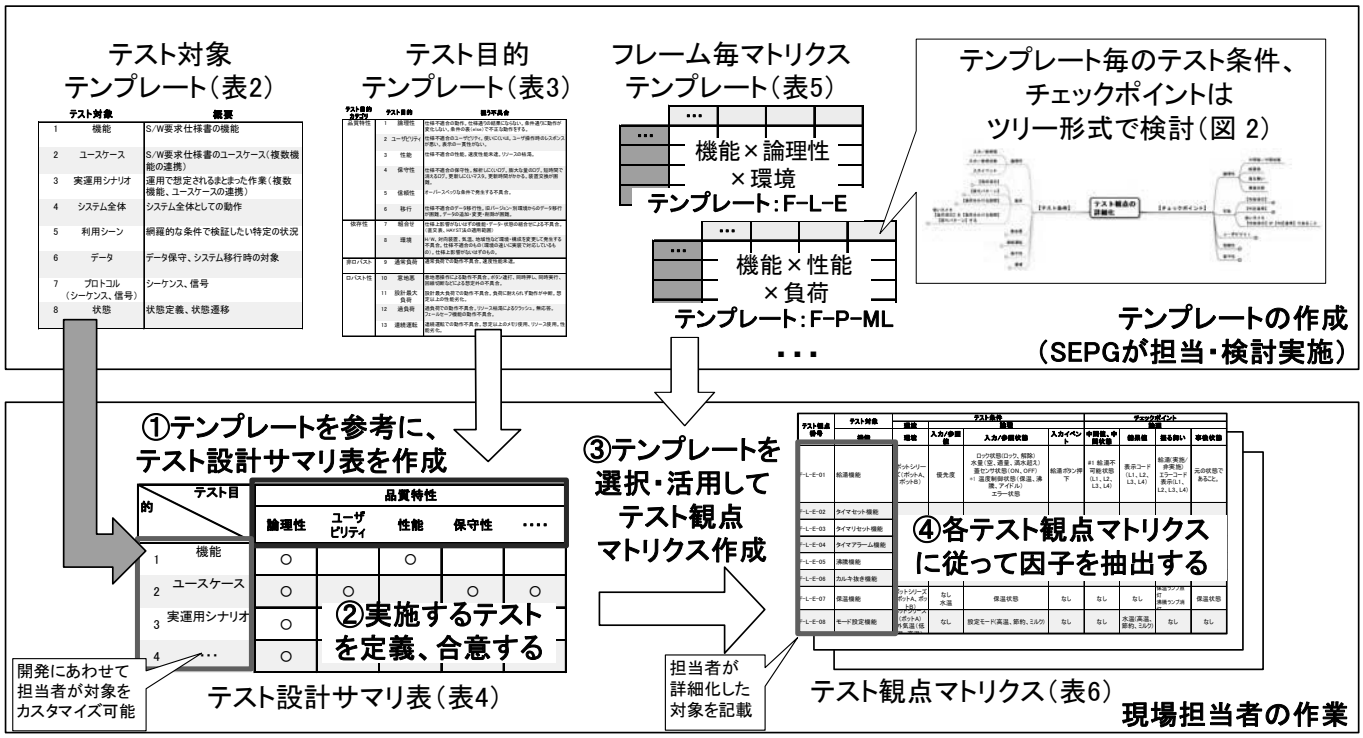


図 3　テスト範囲の決定～テストマトリクス作成の流れ（テンプレートと実作業の関係）

テスト観点マトリクスのテスト条件、チェックポイントの定義は、テスト目的ごとに狙う不具合を検出するために何を入力（テスト条件）として、何を確認する（チェックポイント）かを作業者に分かりやすく示す必要がある。そこで、NGT/VSTeP[3]におけるテスト観点抽出方法を参考に、論理性テスト、性能テスト、ユーザビリティテストといったテスト目的ごとにテスト条件とチェックポイントを有識者で洗い出してマインドマップで整理し、枝の関係は何か（has-a／is-a）、漏れはないかをチェックした。マトリクスの段階で、因子をテスト条件、チェックポイントに分類することで、テストケース作成のインプットとして両方がそろっていることを確認する。

また、テスト観点マトリクスは因子の抽出に特化し、因子の組合せは次の作業で行う。これには、抽出作業と組合せ作業で頭の切り替えによる作業効率の低下を防ぎ、組合せで数が増える前なので類似パターンの抽出・レビューを容易化する狙いがある。

テスト観点マトリクス作成までの作業におけるテンプレートと実際の関係を図 3 に記載する。

テンプレートは SEPG により事前に作成及び準備された状況で、担当者はテンプレートを活用しテーラリング検討を行いながら、テスト観点マトリクスを作成する。

3.4. テスト観点マトリクス作成後の作業

(1)テストケースの作成

テスト観点マトリクスでテスト条件とチェックポイントの因子を抽出した後は、因子毎に同値分割表を用いて具体的な入力値と期待値を決定する。次に、原因結果グラフ、デシジョンテーブル等を用いて入力値と期待値の組合せを作成し､テストケースとする｡表 7 はデシジョンテーブルの作成例である。テスト観点マトリクスの１行のデータが１つのデシジョンテーブルに対応する。

表 7 デシジョンテーブルの作成例

入力条件	#1	#2	#3	#4	#5	#6	#7
1. ロック状態							
1 ロック	○	-	-	-	-	-	-
2 解除	-	○	○	○	○	○	○
2. 水量							
1 適量(水位センサ1～4がONの状態)	○	-	-	○	○	○	○
2 空	-	○	-	-	-	-	-
3 満水(満水センサがONの状態)	-	-	○	-	-	-	-
3. 蓋センサ							
1 ON(閉じる状態)	○	○	○	-	○	○	○
2 OFF(開く状態)	-	-	-	○	-	-	-
4. 状態							
1 保温状態	○	○	○	○	-	-	○
2 沸騰状態	-	-	-	-	○	-	-
3 アイドル状態	-	-	-	-	-	○	-
期待結果	**#1**	**#2**	**#3**	**#4**	**#5**	**#6**	**#7**
1. 振る舞い							
1 給湯実施	-	-	-	-	-	-	○
2 給湯非実施	○	○	○	○	○	○	-
2. エラーコード表示							
1 L1(ロック中)	○	-	-	-	-	-	-
2 L2(給湯不可水量)	-	○	○	-	-	-	-
3 L3(蓋空き)	-	-	-	○	-	-	-
4 L4(給湯不可状態)	-	-	-	-	○	○	-
5 なし	-	-	-	-	-	-	○

(2)テスト手順の作成

テストケースにテスト実施における準備作業などのテスト手順を追加しテスト手順書にまとめる(表 8)。このとき、同じ入力条件や事後条件に着目しテスト手順の集約を図る。

表 8 テスト手順書の作成例

テスト項目番号	テスト項目	開始状態	テスト手順	テスト規格	備考
共通手順-01	**共通手順　保温状態にする**	水なし コンセント抜け 蓋空き	1)ポットに水を適量注ぐ 2)コンセントを挿す 3)蓋を閉める 4)沸騰するまで待つ 5)カルキ抜き完了まで待つ 6)温度が安定(保温温度±2℃)するまで待つ	6)保温状態であること	
F-L-E-01-#1-01	給湯ボタン　ロック状態給湯非実施	保温状態 ロック解除	**1)共通手順-01を実施する** 2)ロックボタンを押してロック状態にする 3)給湯ボタンを押す	2)ロック状態であること 3)給湯非実施 エラーコード表示L1	
F-L-E-01-#2-01	水量空状態給湯非実施	保温状態 ロック解除	**1)共通手順-01を実施する** 2)水量を空にする 3)給湯ボタンを押す	3)給湯非実施 エラーコード表示L2	水量の変化はデバッガで状態変化させる必要ありもしくは、ポットの蓋を空けずに水量を減らす治具が必要

4. テスト開発プロセスの構築

マトリクスを抽出とした一連のテスト作業を現場に適用するために、テスト開発プロセスとしてガイドライン化した。図 4にテスト開発プロセスの全体像を示す。ガイドラインに則って作業を実施することで、テスト対象・テスト目的の組合せからテストケースまで定型作業で段階的にテストを詳細化することができる。またテスト設計サマリ表が示す全体像から段階的に成果物をレビューすることにより、ポイントが絞れ効率的に漏れ・誤りを検出できる。

5. 実施結果

本論文執筆時点('12/9 月)において､2 件の通信 S/W に本手法を適用した。適用工事 2 件はテスト実装を終えたところで、テスト実施はこれからであるため今回はテスト成果物について、組織標準との比較を行い良好な結果を得た（表 9)。

表 9 手法適用効果（組織標準比）

	工事1	工事2	平均
(1)テスト密度 （テストケース数/開発量KL）	252%	150%	**201%**
(2)テスト成果物作成時間 （作業時間/開発量KL）	92%	79%	**86%**
(3)テストケース1件あたりの 作成時間	37%	53%	**45%**
(4)テストケース1件あたりの レビュー時間	36%	85%	**60%**

(1)テスト密度（テストケース数/開発量 KL）

テスト密度は 2 工事平均で組織標準比で 201%となり増加した。

(2)テスト成果物作成時間（作業時間/開発量 KL）

テスト成果物作成時間は組織標準比 86%となり短縮した。

(3)テストケース１件あたりの作成時間

テストケース作成時間は組織標準比 45%となり大きく短縮した。

(4)テストケース１件あたりのレビュー時間

テストケースレビュー時間は組織標準比 60%となり短縮した。

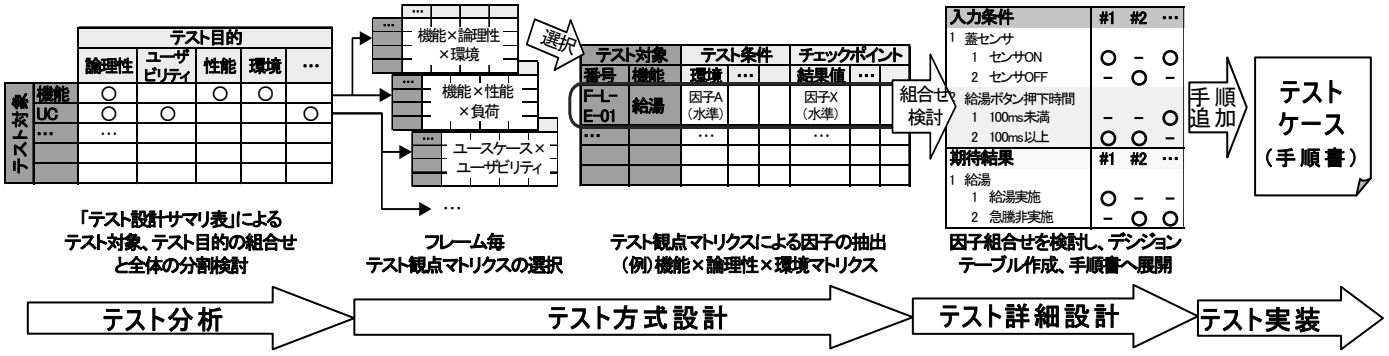


図 4　テスト開発プロセス全体像

(5)テストケースの質

抽出したテストケースの質を評価するために、同一の機能に、以前の設計手法と今回の設計手法を適用し、結果を比較して論理的な組合せの抜け・重複の確認を行った。

表 10 以前の設計手法と今回の手法の比較

テスト対象	ケースA:テスト数（以前の設計手法）	ケースB:テスト数（今回の設計手法）
機能1	10 ⇒評価:7件の抜け	16 ⇒評価:1件の抜け
機能2	2 ⇒評価:2件の抜け	5 ⇒評価:1件冗長あり
機能3	20 ⇒評価:3件の抜け 1件冗長あり	18 ⇒評価:抜け無し
合計	32 ⇒評価:12件の抜け	39 ⇒評価:1件の抜け

ケース A が以前の設計手法を用いた結果、ケース B が今回の設計手法を用いた結果である。

ケース A において、組合せによる抜けが発生しているという状況が確認できた。

上記のように、今回の設計手法を用いることで論理的な検討が改善できることが判断できる。

表 11 プロセス単位のヒアリング結果

対象プロセス/担当者ヒアリング結果
テスト分析
全体の俯瞰性が良い。
関係者で合意を取りながら作業が出来た。
大きな範囲で、テストを行う対象の抜けを見つけることが出来た。
早期に目標とする品質を考えることが出来た。
テスト方式設計
レビュー観点がシンプル。
抽出すべき項目が分かりやすく比較的単純で作業に集中できる。
因子を抜き出すまでが目標となるため、作業の終了判断基準が明確だった。
フレーム単位で担当者を分担することが出来た。結果として、マネジメントがやりやすかった。
全体が見えやすいことから抽象化するポイントを発見しやすく、テスト実装～実施時の作業効率化につながった。
テスト詳細設計、テスト実装
入出力のレビュー、組合せのレビューに分割して考えることが出来たため、レビューがやりやすい。
全てのデシジョンテーブルをレビュー出来た。
事前に因子を抜き出していたことにより、CFDによる組合せ検討がやりやすかった。

また、担当者に手法に対するヒアリングを行った結果においても、手法の狙いに適合する結果が得られていることを確認出来た（表 11）。

6. 考察

次に 2 項に挙げた課題に対する対策結果を記載する。

（課題１）巨大なマトリクスになり作業性が悪い

マトリクスをフレーム毎に分割したため、最も大きいマトリクスの列数で従来のものより半減（15⇒8）した。ヒアリングでは、全体の見通しがよくなり因子の抽出作業が実施しやすくなったという結果が出ており、定量データではテスト成果物作成時間の低減に表れているものと思われる。

（課題２）テスト観点の定義が不十分

テスト目的毎に抽出すべき項目を細分化し、マトリクスの列見出しに定義した。ヒアリングでは、抽出すべき因子が分かりやすく作業しやすいという結果が出ており、定量データのテスト成果物作成時間の低減と因子の網羅度向上によるテスト密度向上に表れている。表 10 の比較結果からテスト設計漏れが低減したことも確認できた。また、レビューにおいてもポイントが明確になりレビューがしやくなったという意見があり、レビュー効率向上に繋がったと思われる。

（課題３）マトリクス作成後の作業が不明確

マトリクスに記載する内容を因子の列挙（テスト方式設計）とし、以降の因子組合せ作業（テスト詳細設計)､テスト手順を足してテスト手順書を作成する作業（テスト実装）を定義した。これにより、各成果物はガイドラインで定めた手法で一貫性を持って作成され、テストケースにおける粒度のばらつきは見られなかった。成果物作成効率化の面では、テスト方式設計において因子が具体的な入力値を含まない粒度であるため、類似パターンの流用が多く行え貢献が大きかった。また、方式設計の類似パターンはテスト詳細設計も流用でき、更に効率化できた。レビュー効率化の面では、従来はテスト手順書で因子、因子組合せ、テスト手順のすべてをレビューしていたのに対し、段階的に詳細化する成果物毎にポイントを絞ってレビューすることができレビュー効率が向上したことが、ヒアリング・定量データの両方で確認できた。また、従来テストケース全件のレビューが実施できていない工事もあったが、今回の 2 工事はいずれも 100%レビューを実施できた。

7. 今後の取り組み

(1)テスト実施結果のフィードバック

テスト開発プロセスの適用は 1 順目の途中であり、テスト実施結果の不具合情報を分析し手法へのフィードバックを行う。

(2)テストの網羅度の定義

テスト設計結果の中身の良否を判断するために、何らかの基準に対する網羅度などの定義が必要である。

(3)テスト実装～実施における効率化の実施

今回のプロセス上では改善対象外となっている、テスト実装からテスト実施フェーズに対して効率的な方法を検討、実践する。

(4)展開拡大

手法の検討メンバを中心に主要工事での展開拡大を進めていく。

文　献

[1] 湯本剛，“テストの上流設計”，ソフトウェアテスト PRESS Vol.10, pp.1-25, Sep.2010.

[2] 秋山浩一，ソフトウェアテスト技法ドリル，日科技連出版社,2010.

[3] 西康晴，JaSST’09 東京テスト観点に着目したテスト開発プロセス(VSTeP)の概要