

3 組合せテストの設計

秋山浩一

富士ゼロックス(株)

本稿は、ソフトウェアテストの動向の中で「組合せテストの設計」に焦点をあてたものである。はじめに、要因間の性質に着目した3つのタイプの「組合せテスト技法」について紹介する。続いて大規模ソフトウェア製品へ適用する際の「組合せテスト設計」の実際について述べる。

組合せテストの問題点と対策

SWEBOK 2004¹⁾では、ソフトウェアテストを「期待されるビヘイビア（イベントに対するソフトウェアの動作や振る舞い）を求めて、通常は無限に大きいと考えられる実行ドメインから適正に選定された有限のテストケース集合を用いて、プログラムビヘイビアの動的検証を行うことである」と定義している。ここで、「実行ドメインが無限に大きい」とされる第1の理由は、入力のコムセによりテストケース数が各要因に含まれる選択肢数の掛け算で指数的に増加するためである。

一般にシステムの出力結果や動作を変更するための入力条件のことを要因やパラメータと呼び、各要因に含まれる選択肢のことを値またはバリューと呼んでいる。本稿ではテストに取り上げた要因のことを「因子」、因子の中の選択肢のことを「水準」と表記する。

わずか2つしか水準がない因子であっても、それが10個存在すれば、単純な総当たりの組合せの場合は、

$$\text{テストケース数} = 2^{10} = 1024$$

となり20個あれば1,048,576と100万を超える。

このようなテストケース数の指数増加問題を解決するために、組合せテストの入力にあたる因子・水準の選択方法と、それら因子を合わせた組合せ表の作成テクニックが考案されてきた。本章では、因子間に関連性がある場合、因子間の関連性が不明な場合の3ケースに分けて効率的に組合せ表を作成する方法について説明する。

〔因子間に関連性がないケース〕

1つ目のパターンとして、因子間に関連性がほぼないと考えられる場合の組合せテストについてWindows

のファイル検索機能（図-1）を例に考える。

ファイル検索ツールを開くと、次の入力が可能となっている。なお、括弧内は各因子について同値分割を実施し、代表値として選択した水準の数を表している。

- ファイル名のすべてまたは一部（4水準）
- ファイルに含まれる単語または句（4水準）
- 探す場所（2水準）
- ファイル更新日（8水準）
- ファイルサイズ（8水準）
- ファイルの種類（4水準）
- システムフォルダの検索（2水準）
- 隠しファイルとフォルダの検索（2水準）
- サブフォルダの検索（2水準）
- 大文字と小文字の区別（2水準）
- テープバックアップの検索（2水準）

これらの因子（入力項目）は、ファイル名とファイルサイズの組合せなど、システムのプログラミングコードを想像すると因子間の関連性はなく、組み合わせたときに不具合が生じることもほとんどないことが予測され



図-1 検索ウィンドウ

3 組合せテストの設計

ファイル名	ファイル内の単語	探す場所	変更日	サイズ	ファイルの種類	システムフォルダの検索	隠しファイルとフォルダの検索	サブフォルダの検索	大文字と小文字の区別	テーブルバックアップの検索
sample1	sample1	ローカル	不明	不明	type1	する	する	する	しない	しない
sample2	sample2	ネット	1週間以内	小	type2	しない	しない	しない	する	する
sample3	sample3	ローカル	1か月以内	中	type3	する	する	する	しない	しない
sample4	sample4	ネット	1年以内	大	type4	しない	しない	しない	する	する
sample1	sample1	ローカル	日付指定1	サイズ指定1	type1	する	する	する	しない	しない
sample2	sample2	ネット	日付指定2	サイズ指定2	type2	しない	しない	しない	する	する
sample3	sample3	ローカル	日付指定3	サイズ指定3	type3	する	する	する	しない	しない
sample4	sample4	ネット	日付指定4	サイズ指定4	type4	しない	しない	しない	する	する

表-1 設定集約による組合せ表

		ファイル名		ファイル内の単語		探す場所		変更日		サイズ		ファイルの種類		システムフォルダの検索		隠しファイルとフォルダの検索		サブフォルダの検索		大文字と小文字の区別		テーブルバックアップの検索																		
		ファイル名	ファイル内の単語	探す場所	変更日	サイズ	ファイルの種類	システムフォルダの検索	隠しファイルとフォルダの検索	サブフォルダの検索	大文字と小文字の区別	テーブルバックアップの検索																												
		sample1	sample2	sample3	sample4	ローカル	ネット	不明	1週間以内	1か月以内	1年以内	日付指定1	日付指定2	日付指定3	日付指定4	不明	小	中	大	サイズ指定1	サイズ指定2	サイズ指定3	サイズ指定4	type1	type2	type3	type4	する	しない	する	しない	する	しない	する	しない	する	しない	する	しない	
ファイル名																																								
ファイル内の単語																																								
探す場所																																								
変更日																																								
サイズ																																								
ファイルの種類																																								
システムフォルダの検索																																								
隠しファイルとフォルダの検索																																								
サブフォルダの検索																																								
大文字と小文字の区別																																								
テーブルバックアップの検索																																								

表-2 設定集約による総当たり表

る。ここで因子間の関連性がなく組み合わせても特別な結果が生じないことを因子間に直交性がある、または無則（むそく）²⁾、あるいは内部独立³⁾の関係と呼ぶ。

このようなケースで、先に述べた総当たりの組合せテストを実施することは、非常に効率が悪い。しかし、一方では、万が一のことを考えて多少は組合せが現れるテストをしておきたいという現場の要求がある。そこで、各因子を単独でテストするのではなく、組み合わせでテストする方法が考案された。ここでは、設定集約と水準集約法について説明する。

■ 設定集約

設定集約という手法は、最も大きな因子の選択項目の数と同数の組合せテストを実施するという方法であ

る³⁾。設定集約では小さな因子については水準をローテーションさせる方法をとる。本ケースでいえば、表の大きさは、最大の因子であるファイル更新日とファイルサイズの8水準から8行と決まる。ファイル名のような小さな因子は、Sample1からSample4までの4水準を割り付けた後、Sample1に戻り表の行が埋まるまで繰り返して割り付ける。設定集約を使用して組み合わせたテストマトリクスを表-1に示す。

表-2はそれぞれの水準の組合せが現れている状況を示す総当たり表である。総当たり表とは2因子間の水準組合せの出現状況を示す表でグレイ部分に組合せが現れていないことを示している。

設定集約を使用した場合のテストケースは最大因子の水準数と同数の8項目、組合せ網羅率は2パラメータ

テスト No.	A	B	C
1	1	1	1
2	1	2	2
3	2	1	2
4	2	2	1

表-3 直交表 L4

間で 38.0%, 3 パラメータ間で 13.3% である.

■ 水準集約法

設定集約に直交表の考えを組み入れた手法を水準集約法と呼ぶ⁴⁾。直交表については後ほど詳細を述べるが、任意の2因子間の全水準の組合せが同数回現れる表である。表-3に最も小さな「直交表L4」を示す。

因子 A と B, 因子 B と C, 因子 C と A との間で (1, 1),

(1, 2), (2, 1), (2, 2) の組合せがすべて出現している.

水準集約法は小さな因子について水準をローテーションさせるのではなく、最も大きな因子の選択項目数と同数の大きさの直交表を横並びにして、そこに因子を割り付けることにより因子間の水準の組合せがより多く出現するようにしたものである。

水準集約法を使用して組み合わせたテストマトリクスを表-4に示す。本例ではL8という直交表を2つ横連結したものに因子・水準を割り付けている。

表-5は上記水準集約法を使用して作成された組合せ表に対する総当たり表である。

水準集約法を使用した場合のテストケースは8項目、組合せ網羅率は2パラメータ間で52.0%, 3パラメータ間で18.1%であり、本ケースにおいては設定集約よ

ファイル名	ファイル内の単語	探す場所	変更日	サイズ	ファイルの種類	システムフォルダの検索	隠しファイルとフォルダの検索	サブフォルダの検索	大文字と小文字の区別	テーブルバックアップの検索
sample1	sample1	ローカル	不明	不明	type1	する	する	する	しない	しない
sample1	sample1	ネット	1 週間以内	小	type1	しない	しない	しない	する	する
sample2	sample2	ローカル	1 カ月以内	中	type2	する	しない	しない	しない	しない
sample2	sample2	ネット	1 年以内	大	type2	しない	する	する	する	する
sample3	sample3	ローカル	日付指定 1	サイズ指定 1	type3	しない	する	しない	しない	する
sample3	sample3	ネット	日付指定 2	サイズ指定 2	type3	する	しない	する	する	しない
sample4	sample4	ローカル	日付指定 3	サイズ指定 3	type4	しない	しない	する	しない	する
sample4	sample4	ネット	日付指定 4	サイズ指定 4	type4	する	する	しない	する	しない

表-4 水準集約による組合せ表

[illegible]

表-5 水準集約法による総当たり表

3 組合せテストの設計

直交表	テスト回数	2 因子間網羅率	3 因子間網羅率
設定集約	8	38.0%	13.3%
L8	8	52.0%	18.1%
L16	16	75.4%	31.8%
L32	32	92.0%	53.1%
L64	64	100.0%	71.8%
L128	128	100.0%	86.3%
L256	256	100.0%	94.4%

表-6 水準集約法と網羅率（検索ツールの例）

	個人	団体
一般	1,200 円	1,000 円
小学生	600 円	500 円
ただし 6 歳未満と 65 歳以上、および県内在住の小学生は無料		

表-7 遊園地の入場料

り 2 パラメータ間で 14.0%，3 パラメータ間で 4.8% 網羅率が向上している。

このように水準集約法ではテスト項目数を設定集約のときと変えずに網羅率を向上させることができる。特に小さな因子間の組合せが 50% から 100% へ増えていることが総当たり表から読み取れる。

さらに、水準集約法は、選択する直交表をより大きなものにすることでテスト回数の増加と引き換えに、網羅率を向上させることができる。L8, L16, L32, L64, L128, L256 を用いた場合の結果を表-6 に示す。

このように、因子間に関連性がほぼないと考えられる場合の組合せテストにおいても、単機能テストだけではなく、設定集約、あるいは水準集約法を使用して組合せテストを実施することにより、テストケース数をきわめて少数に抑えながら、デフォルト以外との組合せを多数実施することができる（組合せ網羅率：40%～50%）ので非常に有効なテストとなる。

【因子間に強い関連性があるケース】

次に、因子間に強い関連性があるケースについて、表-7 に示す遊園地の入場料問題⁵⁾を例に考える。

表-7 の条件から因子・水準をリストアップすると、

- ・個人（Yes, No）
- ・団体（Yes, No）
- ・一般（Yes, No）
- ・小学生（Yes, No）
- ・県内在住（Yes, No）
- ・6 歳未満（Yes, No）
- ・65 歳以上（Yes, No）

の 7 つの因子となる。これらが深く関係して入場料金が決まっている。ここで因子間の関連性が強く、組み合わせた場合に特別な結果が生じることを因子間に直交性

	TEST #1	TEST #2	TEST #3	TEST #4	TEST #5	TEST #6	TEST #7	TEST #8
Causes:								
個人		T	T	F	F	F	T	F
団体		F	F	T	T	T	F	T
一般		F	F	F	F	F	T	T
6 歳未満		F	F	F	F	F	T	F
65 歳以上		F	F	F	T	F	F	F
小学生		T	T	T	F	F	F	F
県内在住		T	F	T	F	F	F	T
Effects:								
県内在住の小学生		T	F	T	F	F	F	F
年齢割引		F	F	F	T	F	T	F
1,200 円 {obs}		F	F	f	f	f	T	F
1,000 円 {obs}		f	f	F	F	F	f	T
600 円 {obs}		F	T	f	f	F	f	f
500 円 {obs}		f	F	F	T	f	f	f
無料 {obs}		T	F	T	T	F	T	F

表-8 遊園地の入場料問題のデシジョンテーブル

がない、あるいは有則（うそく）²⁾の関係があるという。このような場合に、先に説明した設定集約や水準集約法を使用すると動作検証が必要な論理関係が十分にテストされない。

そこで、入力条件の組合せに対して、ソフトウェアがどのようなアクションを起こすかの条件表を作成しその表に従ってテストを実施する。この条件—アクション表のことを決定表、または、デシジョンテーブルと呼ぶ⁶⁾。遊園地の入場料問題のデシジョンテーブルを表-8 に示す。表中の大文字の T と F は明示的に True と False を設定することを意味し、小文字の t と f は結果として論理的に True と False になることを示している。また {obs} は、その結果ノードが出力として現れ、テスト実施者にとって観測可能であることを表している。

実際にテストを行うときには、因子が数値の場合は最低限、境界値の 2 点で確認する必要がある。たとえば TEST#6 は、6 歳未満が True なので境界値である 5 歳と 4 歳の 2 つの条件で 2 回テストを実施する。したがって、テスト回数はデシジョンテーブルの列数よりも多くなる。

ここで、複雑な仕様になると、デシジョンテーブルをミスなく作成することが困難になる。そこで、原因結果グラフや、CFD（Cause Flow Diagram）法⁷⁾、機能図式法⁸⁾が仕様書とデシジョンテーブルをつなぐ役割として使用されている。

原因結果グラフは、図-2 のように、原因と結果の関

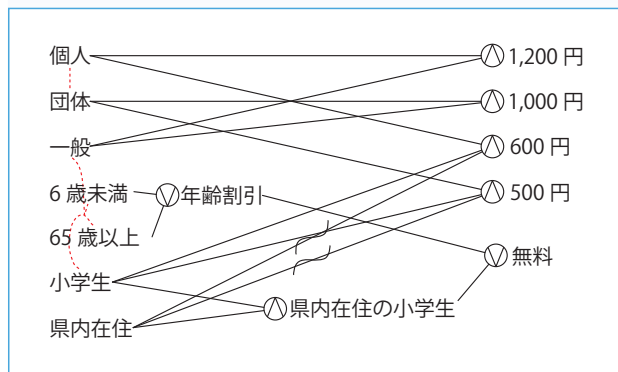


図-2 遊園地の入場料問題の原因結果グラフ

係をブールグラフにまとめたものである。

この原因結果グラフの全ノードの True/False が確実に確認できる論理展開式一覧を作成し、次に、それぞれの論理展開式がテストケースに含まれているかチェックしながらデシジョンテーブルを作成していく⁵⁾。

つまり、中間ノードを含め全ノードの True/False に対してどこまでテストできたかが「因子間に強い関連性があるケース」のテスト網羅性ということになる。

なお、原因結果グラフやデシジョンテーブルを設計段階で作成すると、プログラミングロジックを明確にすることができるため、バグの少ないソフトウェア開発につながる。

【因子間の関連性が不明なケース】

最後に、因子間の関連性が不明なケースにおける組合せテスト技法について説明する。本ケースがソフトウェアの組合せテストのなかで、最も大きい割合を占める。その理由はソフトウェアがその構造を目で確認することが非常に困難であるという性質を持つため「因子間に関連性がない」と言い切れるケースが少ないからである。

また、「因子間に強い関連性がある」ケースにおいては、その因子数が15を超えるケースはまれである。逆に15を超えるようであればシステム分割を適切に行い強い関連性を持つ因子は1グループあたり15個以内に抑えるようにすることが望ましい。一般的に人間は同時に10個程度の要因の組合せまでしか考慮できないからである。それと比較し、因子間の関連性が不明なケースにおいて筆者は200程度までの因子を組み合わせたテストを実施した経験がある。

さて、「因子間の関連性が不明なケース」への対策として、従来はランダムテストと呼ばれるテスト担当者の経験と勘に頼った方法がとられる場合が多かった。しかし、近年、網羅的にテスト項目を作成する方法として直交表の活用⁴⁾と、ペアワイズテスト⁹⁾が注目を集めている。

どちらの手法も、その有効性の根拠は「バグの多くが

テスト No.	A	B	C	D	E
1	1	1	1	1	1
2	1	2	2	2	2
3	2	1	1	2	2
4	2	2	2	1	1
5	3	1	2	1	2
6	3	2	1	2	1
7	4	1	2	2	1
8	4	2	1	1	2

表-9 直交表の例 (L8)

テスト No.	配置	太字	斜体	下線	色	期待結果
1	左詰め	OFF	OFF	OFF	白地に黒	Test
2	左詰め	ON	ON	ON	黒地に白	Test
3	右詰め	OFF	OFF	ON	黒地に白	Test
4	右詰め	ON	ON	OFF	白地に黒	Test
5	中央揃	OFF	ON	OFF	黒地に白	Test
6	中央揃	ON	OFF	ON	白地に黒	Test
7	均等割	OFF	ON	ON	白地に黒	Test
8	均等割	ON	OFF	OFF	黒地に白	Test

表-10 直交表の割り付け例 (L8)

単一の要因、あるいは2～3の少数の要因の組合せが原因で発生している」というバグ解析結果に基づいている¹⁰⁾。

■直交表を使用するテスト

水準集約法の節で触れたように、直交表とは一般には任意の2因子間の全水準の組合せが同数回現れる表のことを指すが、2因子間に限らず、より強度が高い3因子以上の間での全水準の組合せが同数回現れる直交表を作成することもできる。表-9にL8直交表を示す。

表-9に次の因子・水準を割り付けてみると表-10のようになる。ここで、最右列はテスト期待結果を示している。

- 配置 (左詰め, 右詰め, 中央揃, 均等割)
- 太字 (OFF, ON)
- 斜体 (OFF, ON)
- 下線 (OFF, ON)
- 色 (白地に黒, 黒地に白)

表-10を見ると、4水準の因子(配置)と、2水準の因子(太字・斜体・下線・色)は、それぞれの水準の組合せが各1回ずつ出現している。また、2水準同士の組合せは、たとえば太字×斜体の組合せで見ると「OFF×OFF, OFF×ON, ON×OFF, ON×OFF」の4通りの組合せが各2回ずつ出現している。直交表を使用した場合のテストケースは8項目、組合せ網羅率は2パラメータ間で100.0%, 3パラメータ間で62.5%である。

組み合わせる因子間に同時に設定できない水準の組合

3 組合せテストの設計

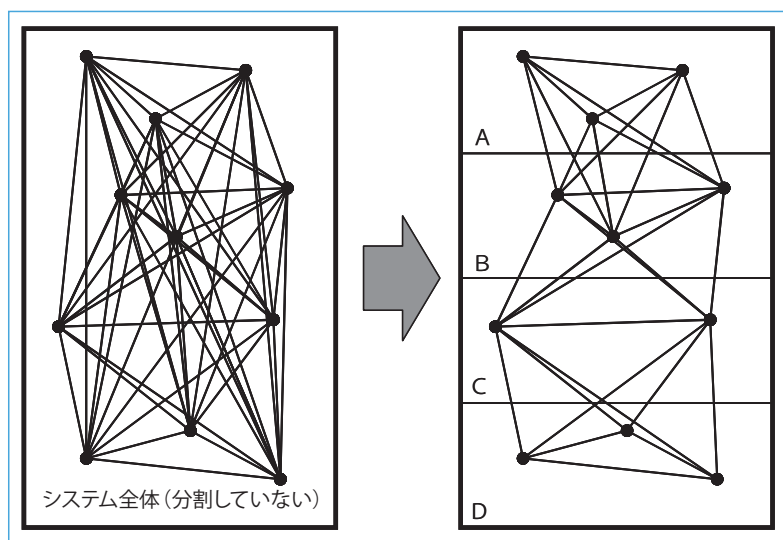


図-3 システムの階層化による組合せの減少

せが存在する場合がある。これを禁則あるいは制約という。直交表に割り付けた後に禁則を回避しながらテストをすることは非常に困難がともなうとともにテストの網羅性について計画できないため、HAYST（ヘイスト）などの禁則を回避する手法が考案され使用されている⁴⁾。

■ ペアワイズテスト

直交表は、因子間の水準組合せが「同数回」出現するが、この「同数回」という条件を外すことでテストケース数を減らす手法をペアワイズテストまたはk-wayテストと呼び、特に2因子間の全水準組合せを100%出現させる方式をAll-pair法と呼ぶ⁹⁾。

ペアワイズテストでは、与えられた因子・水準からできるだけテスト回数が少なくなるように代数的手法や探索的手法で組合せ表を生成する。ところが代数的手法は数学における未解決問題の1つである任意の条件における最小のカバリングアレイ（Covering Array）を探す問題と同等であり最小を保証する決定的な方式がないため、近似的なアルゴリズムが多数提案されている。

ペアワイズテストにおいても直交表同様に禁則の取り扱いが必要であり、こちらもさまざまな方法が提案され実用化されているが、代数的手法とはなじまないため探索的手法とセットになっていることが多い。

直交表とペアワイズテストのどちらが良いかの実測例は乏しいがFeliCa ICチップファームウェアのテストにおいてはほぼ同等という報告がある¹¹⁾。本議論は、多因子間のバランスがとれた網羅率を優先するか、テスト項目数（工数）を優先するかのトレードオフに行き着くためテスト対象製品の特性に合わせて総合的に判断し選択すべきと考える。

組合せテスト設計の実際

これまで、因子間の関連性に着目し、大きく3種類の組合せテストがあることを説明してきた。ここでは実際にこれらの組合せテスト技法を開発プロセスに組み込む場合の注意点について述べる。

【組合せの観点のレビュー】

効率の良い組合せテストを実施するためには、開発プロセスのすべての段階で組合せの観点でレビューすることが重要になる。

■ 要件レビュー

顧客要求を要件リストに変換する際に、組合せの観点で検証可能性をレビューすることにより対象市場のスコープを明確にする。具体的には、最小単位のアトミック要件に対してFV表⁴⁾を用いてそれぞれの要件をどのようにして検証するか明確にする。

FV表はFunction Verification Tableの略で、機能に対して検証方法をセットで書き出した表のことである。FV表を作成しレビューする際に、顧客の使用条件であるノイズについて十分議論しておくことによりビジネス対象スコープを明確にすることができる。

■ アーキテクチャレビュー

2水準の因子が20個あった場合の総組合せ数は100万（ 2^{20} ）に達する。しかし、それを10個+10個の2つのグループに分割しグループ間の独立性を確保できれば検証すべき総組合せ数は、2千強（ $2^{10} + 2^{10}$ ）に減少する²⁾。図-3のように、システムをいくつかの階層に分けて同じ階層と隣接する階層の因子のみにアクセスするように仕様を制限する（アーキテクトする）ことによって全体として因子間の組合せ数を減らし組合せバグの発生を減らすことができる。

また、ソフトウェアプロダクトラインアーキテクチャの考え方を取り入れ、製品群に対して共通部と変動要素を分け、変動要素を容易に切り替え可能としておくことも独立性を高めることも、ソフトウェアの検証空間を小さくする方法の1つである。

■ テスト設計レビュー

テスト設計においては、テスト対象として取り上げた因子・水準の過不足の確認が中心となる。因子・水準の抽出は6W2H（Why, What, How, When, Where, Who, Whom, How much）を用いて実施する。

因子はWhat（仕様書）からの抽出が中心になるが

まとめ

これまで、「組合せテスト技法」と「組合せテスト設計の実践」について説明してきた。無限に大きいといわれる実行ドメインに対して論理的な観点とバグ発生要因数の観点から効果と効率のバランスをとりながらテストケース数が爆発しないように組合せ表を作成し、網羅率（論理接点の True/False 網羅率、因子間の組合せ網羅率、状態遷移の 2 イベントパス網羅率）という観点で品質を評価する方法である。また、テストのみならず、要件レビューやアーキテクチャに遡って対策を組み込んでいくことの有効性について強調した。

しかしながら、状態遷移や並行処理を含めた組合せについては未解決の問題も多く残っておりモデルチェッキングとの併用など、テスト技術の開発とともに開発プロセス全般に渡る対応が必要であると考えている。

参考文献

- 1) 松本吉弘：ソフトウェアエンジニアリング基礎知識体系，オーム社（2005）。
- 2) 松尾谷徹：探索理論に基づく合理的なテストとは，JaSST 札幌 基調講演（2007）。
- 3) 藤江祐二，三宅正二，西 康晴：組込みソフトウェアに対する開発上流からのテスト設計の試行，JaSST 東京（2007）。
- 4) 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，日科技連出版（2007）。
- 5) 松尾谷徹，秋山浩一，鈴木三紀夫：三賢者，テストを語る，JaSST 東京（2007）。
- 6) Hughes, M. L., Shank, R. M. and Stein, E. S. (石尾登監訳)：デシジョン・テーブル テーブル化による思考整理学，日本能率協会（1972）。
- 7) 松本正雄，小山田正史，松尾谷徹：ソフトウェア開発・検証技法，電子情報通信学会（1997）。
- 8) 古川善吾：機能テストのためのテスト項目作成手法について，情報処理学会ソフトウェア工学研究会報告，No.16-2（1980）。
- 9) 土屋達弘，菊野 亨：ペアワイズテスト，電子通信学会論文誌 D，Vol.J90-D, No.10（2007）。
- 10) Kuhn, D. R., Wallace, D. R. and Gallo, A. M. Jr.：Software Fault Interactions and Implications for Software Testing, IEEE Transactions on Software Engineering, Vol.30, No.6（2004）。
- 11) 太田豊一，栗田太郎，松尾谷徹：All-pair 法を応用した携帯電話組み込み用モバイル FeliCa IC チップファームウェアの評価に関する報告，JaSST 大阪（2006）。

（平成 19 年 12 月 26 日受付）

Why（システムの目的＝お客様の課題）に立ち返ることと How much（セールスポイント）を忘れずに，Who（使用者）について When と Where を頭に浮かべながらリストアップするとよい。また，Whom すなわち，使用者が誰のために使う機能なのかを考えると機能の真の目的や使用条件が明確になる。

なお，水準については，グレーボックステスト的に How（設計やコード）に立ち入り，仕様書に現れない境界値や特異値がないかどうか調査する必要がある。

■ 組合せ表(テストマトリクス)の作成と実施

抽出された因子・水準から組合せ表を作成するためのツールが存在するのでそれを利用するか，手作業で実施する。因子数が 10 程度の原因結果グラフからデシジョンテーブルを作成する作業は制約条件の複雑度にもよるが半日から 1 日程度である。また，因子数が 30 から 50 の直交表への割り付けは禁則の複雑さにもよるが 1～2 日程度かかる。富士ゼロックス社においては社内ツールを開発し開発部門を含め広く活用している。

なお，テスト実施においては組合せテストを開始する前にそれぞれの水準が機能することを単機能テストで確認しておくことが重要である。もし，単機能テストを怠ると 1 テストケースに対して複数のバグが発生する可能性が高くなり，バグの解析に時間がかかるとともにテストのやり直しが発生するため効率が極端に落ちる。

■ 状態遷移の考慮

システムが状態遷移をとまなう場合の組合せテストについては，状態遷移グラフから関係行列を作成し，2 つのイベントによる遷移パターンを行列演算によりすべて求め，それらに対して直交表を用いて組合せを作成する方法が提案されている⁴⁾。また，状態自身を要因の 1 つと見て CFD 法を用いてテストケースを作成する方法もある。どちらも現場で使われ始めた段階で有効性の検証については今後の課題と考えている。

■ 並列処理の考慮

並列処理システムに対して網羅的にテストケースを作成することはきわめて難しい。したがって，論理的網羅，組合せ網羅，状態遷移網羅をとったテストケースを並列処理のそれぞれに対して実施した後に，統計的テスト技法を用いて，システムの使用方法に近いテストケース群を流して評価する。

秋山浩一

Kouichi.Akiyama@fujixerox.co.jp

1985 年，富士ゼロックス（株）へ入社。1997 年からテスト手法およびツール開発を担当。NPO 法人ソフトウェアテスト技術振興協会理事。