

組み込みソフトウェア開発課題への挑戦 ～網羅度～

渡辺 政彦[†]

組み込みシステムに占めるソフトウェアの割合が高くなるにつれて、モデルやコードを網羅的に検査、テスト、検証することが重要になっている。しかしながら全てを網羅することは、巨大化・複雑化する組み込みソフトウェアでは容易なことではない。そこで、ある部分に限定して100%網羅する技法やより最適解の近似値を求める技法が用いられるようになってきた。そこで本稿では最近よく見かける3つの技法、「直交表」、「MC/DC」、そして「GA」についての入門的な解説をする。

Challenge to embedded software development problem -coverage-

MASAHIKO WATANABE[†]

In an embedded system, due to the increase of software involved in it nowadays, it has become important to inspect, test and verify its model and code exhaustively. It is not easy, however, to cover all the test cases in the model or code in a large-scale complicated embedded software. Therefore recently those methods that target exhausting test cases of a restricted coverage or finding a better approximation of the optimal solution are commonly well used. In this paper, we explain three such methods, called orthogonal tables, MC/DC and GA.

1. はじめに

ソフトウェアの特徴として、同じものを作ることは極めてたやすく、また、ソフトウェアそのものに物理的な経年変化は起こらないことはよく知られている。つまり、ソフトウェアの問題は製造上の問題、物理的現象上の問題ではなく、設計上の問題かつ論理的現象上の問題である。設計かつ論理上の問題であれば、「正しく」設計し、「正しく」テストすれば良い。しかしながら、ソフトウェアにおける問題は、「正しく」設計し、「正しく」テストすることが非常に困難なことである。

Myersは『“完全な”テストは不可能であるから、いかなるプログラムのテストも不完全にならざるをえない』と述べている [1]。そこで本稿では、「正しい」もしくは「完全な」テストに近づける方法を議論する。

さらにDavisは『網羅的でとりこぼしのないテストは不可能である』と述べている [2]。特に、機器に組み込まれるソフトウェア（組み込みソフト

ウェア）では、データの値の他に、制御信号の値、またその値を更新する時間を網羅的にテストすることが要求されるため、網羅すべき範囲が大きくなる。そこで本稿では、網羅すべき範囲をどのように絞り込むかについて議論する。

2. 技法紹介

本稿では、網羅度に関して以下の3つの技法を紹介する。

1. 直交表
2. MC/DC (Modified Condition / Decision Coverage)
3. GA (Genetic Algorithm)

直交表は「実験計画法」の1つであり、1920年代から農業実験で用いられてきた。農業実験では、例えば収穫を最大限にするために、品種、肥料、温度、雨量、そして土壌など数多くの因子を組み合わせて、最適な配合や条件を見つける実験を行う。しかしながら、現実には全ての実験を行うことは時間的、費用的に困難であるため、実験の組合せを削減するために直交表を用いる。3章では、農業におけるこの実験項目削減

[†] キャッツ組み込みソフトウェア研究所、CATS Embedded Software Laboratory

に倣い、組込みシステムにおけるテスト項目を削減することについて議論する。

MC/DCカバレッジは、航空無線技術委員会(RTCA)によって作られたソフトウェアの開発用ガイドラインDO-178Bにおいて採用され注目を集めている。全ての条件の組合せを確認するためのテストケースは前述のように困難であるから、他の条件を固定化することでテストケースを削減するMC/DCについて4章で議論する。

GAとは、「発見的手法」の1つである。発見的手法とは、「完全」であるとは言い切れないが、現実的な時間内で、それに近い解を求める手法である。遺伝子が優秀な子孫を残すことを模倣し、遺伝子を解の候補とすることで、最適に近い解を求めるGAについて5章で議論する。

3. 直交表

3.1. 考え方(交互作用)

直交表は実験計画法の1つである。実験計画法は実験を効率よく行うための方法で、1920年代に農業での実験に使われていた。

例えば、収穫を最大限にするために品種、温度、水をどのように組み合わせればよいかを実験する。品種、温度、水の3つの因子にそれぞれに2つの水準があるとすると、品種(東・西)、温度(高・低)、水(硬・軟)である。全ての因子の全ての水準の組合せ実験をしようとする $2^3=8$ 通りの実験が必要となる(表1)。

直交表では、因子間で「交互作用」がないことを前提として、組合せ数を削減することができる。「交互作用」について図1に示す。図1には因子(品種、温度)の収穫高への影響を実験した結果が示されている。

図1-(a)は、品種、温度は収穫効果なし(「主効果なし」と言う)の場合を示している。

図1-(b)は、品種が西より東の場合に収穫効果あり、温度は高いほど収穫効果あり(「主効果あり」と言う)の場合を示している。

図1-(c)は、品種は西が良いとも、東が良いとも言えず、温度は高ければ良いとも、低ければ良いとも言えない。しかし、西の品種は温度が高いほど収穫効果があり、東の品種は温度が低いほど収穫効果がある。

図1-(c)では、どの品種で収穫が高いのかそれ単独ではわからず、また温度においても低い方がよいのか、高い方がよいのかが分らないので、両因子に主効果なしとなる。西の品種は温度が高くなれば収穫が良くなり、東の品種は温度が低くなれば収穫が良くなる。この依存関係を交互作用と呼ぶ。

図1-(a)では、品種と温度に交互作用はない。図1-(b)では、どの品種にも温度が高ければ同じ比率で収穫が良くなるため、品種と温度に交互作用はない。あくまで温度という主効果があるということになる。品種の線が平行になれば、交互作用があるとなる。

表1 農業実験

		品 温 水		
列 No.		1	2	3
1	東	高	硬	
2	東	高	軟	
3	東	低	硬	
4	東	低	軟	
5	西	高	硬	
6	西	高	軟	
7	西	低	硬	
8	西	低	軟	

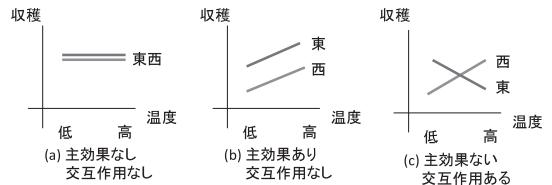


図1 交互作用

交互作用がなければ、因子間全ての組合せをとらずに、2つの因子間で全ての水準の組合せをすれば良いという考え方が直交表である。

『網羅的でとりこぼしのないテストは不可能である』が、全因子のうち直交する因子成分に関しては、網羅的でとりこぼしのないテストができる可能性はある。

3.2. 組合せ削減方法(L4)

組み合わせる因子の中で最大水準数を持つ2つの因子の各水準を2のべき乗に切り上げてそれらの積をとり、それを最少の直交表サイズとする。

因子数は直交表のサイズより少ない。

表1を例にすると最も大きい水準数は2なので $2*2=4$ となり、直交表L4を適用すれば良いことが分かる(表2)。LはLatin squareの頭文字を示し、続く数字は実験数(行数)を示す。

L4は2因子間で2水準の全ての組合せを示す。表2の例では、(11)、(12)、(21)、(22)の組合せは、どの2列でもともに1回ずつあり、直交である。「AとB」、「AとC」、そして「BとC」で全ての水準が組合せられている。水準値が1か2の2種類しかもたないことを2水準系と呼ぶ。

表2 L4

L4		A	B	C
列No		1	2	3
1		1	1	1
2		1	2	2
3		2	1	2
4		2	2	1

表3に、表2の全ての組合せからL4が適用された組合せと削減された組合せを示す。No2,3,5,8が削減された組合せである。

表3 全組合せ

		A	B	C
列No		1	2	3
1		1	1	1
2		1	1	2
3		1	2	1
4		1	2	2
5		2	1	1
6		2	1	2
7		2	2	1
8		2	2	2

全組合せ $2*2*2=8$ 通り

これらの組合せを削減する。

繰り返しになるが、表4に、表1へL4を適用した場合の組合せを示す。

表4 L4適用

L4		品	温	水
列No		1	2	3
1		東	高	硬
2		東	低	軟
3		西	高	軟
4		西	低	硬

3.3. テストへの応用 (L16)

直交表をテストに適用する。組込み機器には、複写機、自動車、そして家電製品などユーザインタフェースを持つものは多い。ユーザインタフェースを持つということは、ユーザにどのような操作をされても問題ないことをテストしなければならない。しかしながら、全てのユーザインタフェースの組合せをテストすることは難しい。

図2にテスト対象のカーオーディオ機器を示す。操作ボタン3種類(A, B, C)ある。Aボタンには2種類の操作、Bボタンには4種類の操作、そしてCボタンには4種類の操作がある。ボタンの種類が因子、ボタン操作種類が水準として直交表を適用する。全てのボタン操作の組合せをテストするには $2*4*4=32$ 通りのテストケースが必要となる。

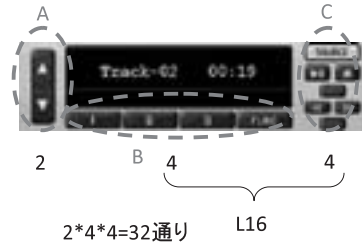


図2 カーオーディオ機器

図2において最も大きい水準数は4なので $4*4=16$ となり、直交表L16を適用すれば良い。直交表から因子と水準を選択する場合、不用意に割り当てすると直交性が実現しないので注意が必要である。

表5 L16 割り当て失敗例

L16	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	2	2	2	2	2	2	2	2	2
3	1	1	1	2	2	2	2	1	1	1	1	2	2	2	2
4	1	1	2	2	2	2	2	2	2	2	2	1	1	1	1
5	1	2	2	1	1	2	2	1	1	2	2	1	1	2	2
6	1	2	2	1	1	2	2	2	2	1	1	2	2	1	1
7	1	2	2	2	2	1	1	1	1	2	2	2	2	1	1
8	1	2	2	2	2	1	1	2	2	1	1	1	1	2	2
9	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
10	2	1	2	1	2	1	2	2	1	2	1	2	1	2	1
11	2	1	2	2	1	2	1	1	2	1	2	2	1	2	1
12	2	1	2	2	1	2	2	1	2	1	2	1	1	2	2
13	2	2	1	1	2	2	1	1	2	2	1	1	2	2	1
14	2	2	1	1	2	2	1	2	1	1	2	2	1	1	2
15	2	2	1	2	1	1	2	1	2	2	1	2	1	1	2
16	2	2	1	2	1	1	2	2	1	1	2	1	2	2	1

B C A

表5のL16の列1-3にボタンBを割り当て、列4-6にボタンCを割り当てる。ボタンAは残りの列7-15のどこか1列を割り当てれば良い。ボタンBに割り当てた列1-3では、4水準が実現できている。しかしながら、列4-6に割り当てられたボタンCでは、4水準がボタンBに直交していない。

割り当てを正しく行うには点線図を用いる。表6では点線図から組み合わせて使用できない列(5,10,15) (6,11,13) (7,9,14)を除外して、列4,8,12にボタンCを割り当てることで、2つの因子間で全ての水準の組合せのテストケースを導出することができる。直交表を用いることで、全ての組合せでは32のテストケースを半分の16のテストケースにできる。

表6 L16 割り当て成功例

no.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	2	2	2	2	2	2	2	2	2
3	1	1	1	2	2	2	2	1	1	1	1	2	2	2	2
4	1	1	1	2	2	2	2	2	2	2	2	1	1	1	1
5	1	2	2	1	1	2	2	1	1	2	2	1	1	2	2
6	1	2	2	1	1	2	2	2	2	1	1	2	2	1	1
7	1	2	2	2	2	1	1	1	1	2	2	2	2	2	1
8	1	2	2	2	2	1	1	2	2	1	1	1	1	2	2
9	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
10	2	1	2	1	2	1	2	1	2	1	2	1	2	1	2
11	2	1	2	2	1	2	1	1	2	1	2	2	1	2	1
12	2	1	2	2	1	2	1	2	1	2	1	1	2	1	2
13	2	2	1	1	2	2	1	1	2	2	1	1	2	2	1
14	2	2	1	1	2	2	1	2	1	1	2	2	1	1	2
15	2	2	1	2	1	1	2	1	2	2	1	2	1	1	2
16	2	2	1	2	1	1	2	2	1	2	1	2	1	2	2

B C A 点線図 (5,10,15) (6,11,13) (7,9,14)

3.4. 適用事例

富士ゼロックスにおける適用結果では、組合せ網羅率が従来（経験と勘）と比較して2.7倍、組合せ市場不具合ほぼ0件との報告が公表されている [3,4]。

3.5. 詳細

点線図を含めさらに詳細な直交表については、別稿で述べる。→リンク先：川崎

4. MC/DC

(Modified Condition / Decision Coverage)

4.1. 考え方 (条件と判定)

MC/DCカバレッジは、航空無線技術委員会 (RTCA) によって作られたソフトウェアの開発用ガイドラインDO-178Bで規定される。

MC/DCはModified Condition / Decision Coverageの略語である。判定 (Decision) は条件判定文の判定であり、条件 (Condition) は条件判定文を構成する条件である (図3)。

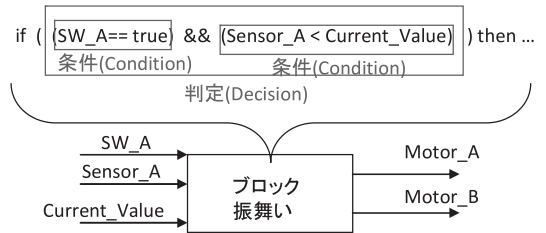


図3 条件と判定

MC/DCはDC (Decision Coverage) やCC (Condition Coverage) よりも網羅度が高く、MCC (Multiple Condition Coverage) よりも網羅度が低い。

『網羅的でとりこぼしのないテストは不可能である』が、MC/DCに関して網羅的でとりこぼしのないテストができる可能性はある。

4.2. DC

DCは、全ての判定に関してtrue/falseの全ての組合せを網羅する。

表7 DC

テストケース			条件		判定
SW_A	Sensor_A	Current_Value	SW_A == true	Sensor_A < Current_Value	
T	50	80	T	T	T
F	50	80	F	T	F

表7に図3のブロック図からDCを用いたテストケースを示す。判定がtrue/falseの全ての組合せを網羅するようにテストケースを作成する。

4.3. CC

CCは、全ての条件に関してtrue/falseの全ての組合せを網羅する。

表 8 CC

テストケース			条件		判定
SW_A	Sensor_A	Current_Value	SW_A == true	Sensor_A < Current_Value	
T	90	80	T	F	F
F	50	80	F	T	F

表8に図3のブロック図からCCを用いたテストケースを示す。条件がtrue/falseの全ての組合せを網羅するようにテストケースを作成する。判定は考慮しない。表8では判定はFFとなっている。

4.4. MCC

MCCは全ての条件に関してtrueとfalseの全ての組合せと判定のtrue/falseの全ての組合せを網羅する。

表 9 MCC

No	テストケース			条件		判定
	SW_A	Sensor_A	Current_Value	SW_A == true	Sensor_A < Current_Value	
①	T	50	80	T	T	T
②	T	90	80	T	F	F
③	F	50	80	F	T	F
④	F	90	80	F	F	F

表9に図3のブロック図からMCCを用いたテストケースを示す。条件がtrueとfalseの全ての組合せと、判定のtrue/falseの全ての組合せを網羅するようにテストケースを作成する。

4.5. MC/DC

MCCでは全ての条件の組合せを確認するため、テストケースが増大する。そのため、MC/DCでは、他の条件の組合せの交互作用を無視することで、組合せ数の削減を行う。他条件の組合せの交互作用の無視とは、他条件を固定化し、自条件の全ての組合せをとることである。これを全ての条件で行い、最後に判定の全ての組合せを求める。

表 10 MC/DC

No	テストケース			条件		判定
	SW_A	Sensor_A	Current_Value	SW_A == true	Sensor_A < Current_Value	
①	T	50	80	T	T	T
②	T	90	80	T	F	F
③	F	50	80	F	T	F

表10に図3のブロック図からMC/DCを用いたテストケースを示す。処理判定ロジックのコン

パイラ生成実行順序のため、条件 (condition) の組合せは必要である。つまりDCではなく、CCが必要になる。また、論理積 (&&) と論理和 (| |) の結果 (decision) の違いをテストする必要もあり、DCが必要である。両方を満足するMCCでは組合せ数が増大するので、他条件を固定化、つまり、他条件の組合せが自条件の組合せには影響しないとして、組合せ数を削減するのがMC/DCである。

表10がどのように求められるかの手順を以下に示す。条件「SW_A == true」をAとし、条件「Sensor_A < Current_Value」をBとする。

(1) Aに関する確認

(ここでは、B = trueに固定して確認)

Aがtrueとなるテストケース：①

Aがfalseとなるテストケース：③

(2) Bに関する確認

(ここでは、A = trueに固定して確認)

Bがtrueとなるテストケース：①

Bがfalseとなるテストケース：②

(3) 判定がtrueとなるテストケース：①

判定がfalseとなるテストケース：②、③

(1) から (3) の操作からテストケース①、②、③が導出され、表9-④を削除することができる。

仮にAとBをfalseに固定すると、

(1)' Aに関する確認

(ここでは、B = falseに固定して確認)

Aがtrueとなるテストケース：②

Aがfalseとなるテストケース：④

(2)' Bに関する確認

(ここでは、A = falseに固定して確認)

Bがtrueとなるテストケース：③

Bがfalseとなるテストケース：④

(3)' 判定がtrueとなるテストケース：①

判定がfalseとなるテストケース：②、③、④

この場合、MCCと同じ数のテストケースとなり、削除には失敗となる。

4.6. MC/DC練習問題

MC/DCの理解を深めるため、練習問題を解く[5]。下記If文のMC/DCを求めよ。

$If(A \ \&\& \ (B \ || \ C)) \ then...$

表11に解答を示す。

表 11 MC/DC表

No	A	B	C	判定	A	B	C
1	T	T	T	T	5		
2	T	T	F	T		4	
3	T	F	T	T			4
4	T	F	F	F		2	3
5	F	T	T	F	1		
6	F	T	F	F			
7	F	F	T	F			
8	F	F	F	F			

- (1) Aに関してNo.1でTTT、No.5でFTT
- (2) Bに関してNo.2でTTF、No.4でTFF
- (3) Cに関してNo.3でTFT、No.4でTFF
- (4) 判定はNo.1,2,3でT、No.4,5でF

これから (1,2,3,4,5) がMC/DCテストケースとなる。

Aに関してNo.2のTTFとNo.6のFTFを選択すれば、さらにテストケースを最小化した(2,3,4,6)となる。

4.7. 適用事例

英国のソフトウェア信頼センターでは、MC/DCテストとランダム入力テストと既存のシステムテストにおける不具合検出数の比較を行った [6]。その結果が表12である。

表 12 MC/DCテスト評価

テスト数	不具合検出数		
	MC/DCテスト	ランダム入力	既存システムテスト
10	4	0	?
100	5	0	?
486	6	0	4
1,000	6	0	
3,000	6	1	
9,514	6	4	
10,000	6	4	
100,000	6	6	
1,000,000	6	6	

PLC (Programmable Logic Controller) にMC/DCを適用し、全不具合数6件を他のテスト方法に比べて早期に発見し、その有効性を示した。

4.8. 詳細

MC/DCテストケースを最小化する方法、MC/DCランダムテストに確率値誤差逆伝播法(バックプロパゲーション)を用いる方法については、別稿で述べる。→リンク先: 松本

5. GA (Genetic Algorithm)

5.1. 考え方 (遺伝子)

GA (Genetic Algorithm) とは、発見的手法分野におけるアルゴリズムである。発見的手法とは、「完全」であるとは言い切れないが、現実的な時間内で、それに近い解を求める手法である。

GAは解の候補を遺伝子として、選択(結婚)・交叉(生殖)し、優秀な子孫が生き残り、まれに突然変異を発生することで、最適解を発見する。遺伝子である解の候補を何にするかが、GAをどのように適用するかの鍵となる。

『“完全な”テストは不可能であるから、いかなるプログラムのテストも不完全にならざるをえない』が、完全に近い解を生命の営みのメカニズムから得ることができる。

5.2. GA適用 (WCET)

組込みソフトウェアがリアルタイムシステムである場合、最悪実行時間(WCET: Worst-Case Execution Time)を保証することが求められる。RM (Rate Monotonic) などのリアルタイムスケジューリング理論で計算できるシステムであれば良い。しかしながら、ネットワークやバスにより、複数のユニットが並列に動作し、かつユニット間の通信に遅延時間が発生し、かつ、駆動するメカの状態によってユニットの反応が異なるような複雑なシステムにおいて、一定時間内の反応を保証することは難しい。そこで、最適解を求めるGAの遺伝子である解の候補を、最悪実行時間と考えた例を図4に示す。

従来の最悪実行時間を検証するには、ランダムにテストケースを作成し、実行後の反応時間を測定し、最悪実行時間値を守れているかを検証していた。ランダムにテストケースを作成するのではなく、実行したテストケースで時間がかかったテストケースを掛け合わせて、より優

秀な、つまり時間のかかるテストケースを導出し効果的なテストケース生成をGAで実現する。

GAでは突然変異を除き、概して次の6つの操作を行う。

- ①初期集団
- ②結果
- ③適合度計算
- ④選択
- ⑤交叉
- ⑥次世代

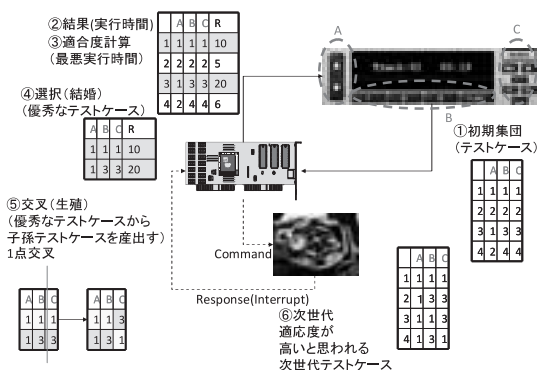


図4 最悪実行時間検証

①初期集団では、最初の親（テストケース）を決める。親の優性遺伝（ここではより実行時間のかかるテストケース）が、子に引き継がれる。このためできるだけ優秀な親を最初に作る事が、GAの効率に大きく影響を与える。解をヒトとした場合、最初の親に魚を与えるのとサルを与えるのでは、求めるまでの時間に大きな差が出るのと同じである。初期集団はテスト実行者が考えて、GAに与えるものである。

②結果では、テストケースを実行した結果を得る。図4では、テストケース1が10ms、テストケース2が5ms、テストケース3が20ms、テストケース4が6ms、実行時間がかかったとしている。

③適合度計算では、結果から優秀（最悪実行時間に近い）な遺伝子（テストケース）がどれかを計算する。

④選択では、適合度計算から優秀なテストケースを選択する。図4では、実行結果が10msかかったテストケース1と20msかかったテストケース3を選択（結婚）する。選択にはルーレット選択、ランキング選択、トーナメント選択があ

るが、ここでは単純に上位から選択するランキング選択としている。

⑤交叉では、テストケース1とテストケース3を「AB」と「C」の1点の箇所で切って、交叉させる。交叉にはここで用いた1点交叉以外にも、2点交叉、多点交叉、一様交叉がある。

⑥次世代では、親のテストケース1とテストケース2と1点交叉によって生まれたテストケース3とテストケース4を実行させ、②の結果へと繰り返す。

この繰り返しテストで最悪実行時間以内に処理が完了することを確認する。

5.3. 適用事例

あるプログラムに対して、ランダムに作成されたテストケースとGAを用いて作成されたテストケースで、MC/DCの網羅率を比較した事例を表13に示す [7]。

表13 MC/DC網羅率比較

プログラム	ランダム	GA
バイナリサーチ Binary search	53.3	66.7
バブルソート1 Bubble sort 1	100	100
バブルソート2 Bubble sort 2	44.4	44.4
日にち計算 Number of days between two dates	35.3	39.2
ユークリッド最大公約数 Euclidean GCD	100	100
挿入ソート Insertion sort	100	100
中央値計算 Computing the median	100	100
解の公式 Quadratic formula	75	75
ワーシャル-フロイド法 Warshall's algorithm	91.7	100
三角形分割 Triangle classification	48.6	84.3

5.4. 詳細

本稿のGAでは非常に単純なランキング選択および1点交叉を用いたが、これら以外にも選択にはルーレット選択やトーナメント選択があり、交叉には2点交叉、多点交叉、一様交叉がある。また突然変異についても別稿で述べる。→リンク先：目時

6. むすび

組込みソフトウェア開発における課題の1つである「網羅度」に対して解を与えるであろう3つの技法について紹介した。最初に「直交表」、次に「MC/DC」、最後に「GA」を解説した。網羅すべき範囲をどのように絞り込むかについ

て、「直交表」と「MC/DC」の技法と適用事例を紹介した。「正しい」もしくは「完全な」テストに近づける方法について「GA」の技法と適用事例を紹介した。

無作為に時間と費用をかけて「網羅度」を上げるより、工学的技法を用いて効率的に「網羅度」を上げる取り組みがこれからますます重要になるであろう。

参考文献

- [1] Glenford J. Myers, 長尾真監訳、「ソフトウェアテストの技法」、近代科学社、1980.
- [2] Alan M. Davis, "201 Principles of Software Development," McGraw-Hill, 1995.
- [3] 秋山浩一、「直交表を活用したソフトウェアテストの効率化－HAYST法の活用－」、<http://www.swtest.jp/jasst05w/S4-1.pdf>
- [4] 小川正樹、「技術者に必要な統計知識」、<http://www.fiberbit.net/user/masa-2ogawa/crmin020.html>
- [5] Christopher Ackermann,"MC/DC in a nutshell," http://www.cs.umd.edu/~atif/Teaching/Fall2006/StudentSlides/MCDC_Coverage_02.ppt
- [6] Peter G Bishop, "MC/DC based estimation and detection of residual faults in PLC logic networks," ISSRE 2003, Fast Abstracts, Supplementary Proceedings, pp. 297-298, 2003.
- [7] Christoph C. Michael, Gary E. McGraw, Michael A. Schatz, and Curtis C. Walton, "Genetic Algorithms for Dynamic Test Data Generation," Proceedings of the 12th International Conference on Automated Software Engineering (ASE'97), pp.307, 1997.

WATCHERS Back Numberで振り返るユーザ会



第13回「ZIPCユーザーズカンファレンス」では、「テクノロジー」「サイエンス」および「エコノミクス」の総合的なビジョンに基づいて、全く新しいコンセプトの組織体である「CATS先端研究所（CATS Advanced Laboratory 略称：CAL）」の開設を記念し、「CALオープン記念セミナー」を同会場にて開催いたしました。

また、恒例の事例発表「ZIPCユーザーズカンファレンス」ではユーザー様自ら事例の発表を賜り、実際の製品開発にZIPCを適用した場合の品質向上、生産性向上に役立つ講演をご聴講頂きました。