PictMaster

ユーザーズマニュアル

２０１０年4月１２日　　第４．１版　　Ｖ４．０対応

岩通ソフトシステム株式会社

本マニュアルの読み方

１．とにかく早く使ってみたい方へ

詳しいことは後でよいからとにかく早く使ってみたいという方は、少なくとも第０章、第 3 章そして第４章を読んでから使ってください。デフォルトの PictMaster の設定は必要最小限の機能しか動作しない設定になっています。PictMaster を有効に使うためには、後でもよいですから本マニュアル全体を読まれることをお勧めします。

２．生成結果の例について

説明の例として示されている組み合わせ生成結果の表および組み合わせ数は、PictMaster の最少テストケース生成の機能を用いて生成したものです。最少テストケース生成の機能は、通常の設定では生成結果の組み合わせ再現性を保証していません。したがってユーザの方が同じモデルで最少テストケース生成を行った場合、本マニュアルで示されている生成結果とは異なる生成結果となる場合があります。

３．各テスト技法の説明について

PictMaster は組み合わせテストのほかに、デシジョンテーブルテスト、状態遷移テスト、制御パステストに適用することができます。本マニュアルの性格上、ツールの使用方法は説明していますが、テスト技法については説明していません。テスト技法について知りたい場合はそれらを解説した書籍を読んでください。

４．PictMaster を業務で使う前に

PictMaster は組み合わせテストをはじめ、複数の種類のテストに対応したテストケースを生成するツールです。本ツールを使いこなすには基礎的なテスト技術を一通り習得していることが必須条件です。

現在は、テスト技術を解説した書籍が何種類か出版されています。基礎的なテスト技術が不足していると感じられた方は、まずこうした書籍などを通じて幅広いテスト技術の知識を学んでから PictMaster を業務で使用されることをお勧めします。

更新履歴

版数	更新日	対応 Ver.	更新内容
1.0	2008.02.06	2.0	新規作成
1.1	2008.02.18	2.1	多くの誤記修正。 5.1 でシートの並びの例を追加。 5.3 でウインドウ分割の説明を追加。
1.2	2008.3.11	2.2	3.PictMaster の使い方の章全般を修正。 4.3 制約条件と制約対象の指定方法の章に AND 条件で複数のパラメータを指定する場合の説明を追記。 4.4 使用できる演算子の一覧の章を追加。
1.3	2008.4.22	2.3	0.インストール方法を変更 5.結果表への記入のしかたの章を追加。以降の章番号を訂正。 6.4 デシジョンテーブルテストと組み合わせテストの統合の章を追加。
1.4	2008.5.20	2.4	3.4 値の並び欄の記入のしかたを変更 3.4.1 エイリアス、3.4.2 無効値テスト、3.4.3 重み付け、付録A 仕様の章を追加。 目次をハイパーリンクからページ番号付きに変更
1.5	2008.6.2	2.5	4 章を条件付き制約と無条件制約の章に分けた。 6.5 テスト実施中に組み合わせを修正する の章を追加。
1.6	2008.6.11	2.6	PictMaster 使用規定を変更した。 3.3.1 原型シートの使い方、3.4.3 重み付け に記述を追加。 重み付けの最大値を 5 から 10 に変更。
1.7	2008.7.7	2.7	PictMaster 使用規定で矛盾する記述を訂正。 3.3 pict.exe の強制終了のしかたを追記。 5.2 表 5.3 の説明内容を修正。
1.8	2008.7.13	2.7	3.4.1 エイリアスを制約表で指定する場合は先頭の値のみ指定するとの説明を追記。
1.9	2008.10.21	2.7.3	3.5 パラメータの重み付けの章を全面変更。 4.制約表への記入のしかたで制約欄を制約欄に変更。 5.確認表を結果表に変更。 6.4 デシジョンテーブルテストと組み合わせテストの統合の章を全面修正。
2.0	2009.2.16	2.8	4.6 制約表の編集方法の章を全面変更
3.0	2009.5.10	3.0	ツールの名称を MTG に変更したことに対応。 MTG 使用規定を一部変更。 7 デシジョンテーブルテストへの適用、8 状態遷移テストへの適用、9 制御パステストへの適用、の章を追記。
3.1	2009.8.16	3.1	3.3.2 制約式の最適化とは を追記。 6.6 効率のよい無効値テストを行なうには を追記。 10 困ったときは を追記。
3.1.1	2009.8.19	3.1	19 ページ目したから4行目の「5個以上の場合で」を「8個以上の場合で」に訂正。 56 ページ目したから4行目「例えばa2 とb2 の組み合わせ」を「例えばc2 とd1 の組み合わせ」に訂正。 3.5.2 特定のパラメータのみ3パラメータ間の組み合わせとするの章を記述内容訂正。
3.2	2009.12.14	3.2	「生成条件」という用語を「シード値」に訂正。 6.2.1 矛盾した制約をすばやく見つけるには の章の記述を簡略化。 3.5 パラメータの重み付け の章の内容修正と追記。

更新履歴の「対応 Ver.」はこのマニュアルが対応している PictMaster の最初のバージョンを表しています。

版数	更新日	対応 Ver.	更新内容
4.0	2010.3.1	4.0	ツールの名称を PictMaster に変更したことに対応した。 3.5 パラメータの重み付け の章を サブモデル と名称を変更し必要最小限度の記述に簡略化した。 サブモデルの効果的な使い方を 6.7 パラメータの重み付け の章で追記した。 4.7 ワイルドカードの使用 の章を追記した。
4.1	2010.4.12	4.0	セル形式がテキスト形式に変わったため、セル内容が数値であるとエラーマークが表示される場合の対処法を ０．インストール の章と１０．困ったときは の章に追記した。

PictMaster 使用規定

以下の使用規定にすべて同意される場合のみ PictMaster を使用することを許可します。

1．PictMaster（以後 本ソフトと表記）はフリーソフトで自由に使用することができますが、著作権は岩通ソフトシステム株式会社にあります。
2．本ソフトはオープン・ソフトウェア・ライセンス v2.0 に基づき、本ソフト（オリジナル成果物）をもとに派生成果物を作成し、配布することができます。ただし、PICTを使用する限り、Microsoftのライセンス条項により、派生成果物を販売することは禁じられています。同様にPICTを同梱しての配布も禁じられています。
3．本ソフトを販売して収益を得る行為を禁じます。
4．本ソフトの著作権表示（© IWATSU System & Software Co., Ltd.）を読めないようにすることを禁じます。ただし、本ソフトをもとに派生成果物を作成し、配布する場合はこの限りではありません。
5．本ソフトを使用したことによるいかなる損害に対しても著作権所有者は一切の責任を負いません。
6．この「PictMaster 使用規定」は予告なく変更を行なうことがあります。

目　次

０．PictMasterのインストール

【PictMaster を使う上で用意するもの】

(1) **PICT**そのものは　http://msdn.microsoft.com/en-us/testing/bb980925.aspx のサイトで入手できます。**あらかじめダウンロードし、インストールしておいてください**。インストール先は必ず以下のデフォルトのフォルダ内にインストールする必要があります。

C:¥Program Files

(2) Excel2000 以降の Excel。

【インストール方法】

(1) PictMaster.zip の圧縮ファイルを開き PictMaster.xls を PC 内の任意の場所に置きます。サーバー上に置くこともできます。ただしネットワークドライブの割り当てを行なっていないサーバーに置いた場合、生成されたテストケースファイル“**a.xls**”、モデルファイル“**a.txt**”などは PICT があるフォルダ内に作成されます。

PictMaster.xls というブック名は変更してかまいません。
Sheet1 というシート名も変更してかまいません。

(2) Excel のセットアップを行ないます。

Excel2007 より前のバージョンでは、ツール → オプション → セキュリティ → マクロのセキュリティ で「中」を選択してください。

Excel2007 では以下の手順を行なってください。

Office ボタン → Excel のオプション → セキュリティセンター →
セキュリティセンターの設定 → 信頼できる場所 → 新しい場所の追加 →
参照 → 任意の PictMaster の保存場所を指定してＯＫをクリックします。
このとき、サブフォルダも含めて指定できます。

(3) PictMaster.zip の圧縮ファイルに同梱されている **nkf.exe** を PICT がインストールされたフォルダ内にコピーします。

以上でインストール作業は終了です。

【重要な注意点】

PictMaster は日本語が使用できますが、スペースについては、**必ず半角スペースを使用してください。**全角スペースはエラーとなります。

PictMaster 4.0 以降では、生成結果はテキスト形式のセルに格納されるようになりました。そのため、Excel のデフォルトの設定では、セルがテキスト形式で内容が数値形式であると型が一致しないとして、**数値のセルにエラーマークが表示**されます。Excel で生成結果が数値のセルにエラーマークが表示されないようにするには、以下の設定を行なってください。

Excel2007の場合

Office ボタン → Excel のオプション → 「数式」 → エラー チェック ルール のカテゴリの 「文字列形式の数値、またはアポストロフィで始まる数値(H)」 のチェックを外す。

Excel2003の場合

ツール → オプション → エラーチェック のタブ → 「文字列として保存されている数値」のチェックを外す。

１．はじめに

PictMaster は Pairwise 法(All-Pairs 法ともいう)を採用した組み合わせテストケース生成を行なう Microsoft のフリーソフトである **PICT**（Pairwise Independent Combinatorial Testing Tool）をより使いやすく、より高機能にした Excel ベースのフリーソフトです。

PictMaster を公開する目的は PICT という非常に優れた組み合わせテストケース生成ツールを Excel 上で簡単に使用することができるようにすることによって、完全に無償のツールとして多くの人に使ってもらいたいからです。直交表をベースにした非常に優れたツールは存在しますが、公開されておらず、自作することはコスト的にも技術的にもかなり困難です。

このような状況を少しでも改善することを目的として PictMaster を公開するものです。

２．PictMasterの仕組み

PICT そのものはコマンドプロンプト上で動作する CUI（キャラクタユーザインターフェース）ベースのアプリケーションです。今となってはコマンドプロンプトになじみのない人が大半です。コマンドプロンプト上で動作する PICT に抵抗感を感じる方も少なくないと思います。コマンドプロンプト上で動作させて、テスト仕様書にテストケースとして組み込むまでに文字コードを２回変更し、Excel でファイルを読み込むなどいろいろな作業をしなければなりません。このように、PICT そのものだけでテストケースを作成しようとすると、かかる手間が無視できません。

Excel でテスト仕様書を作成しているのなら、Excel 上で組み合わせテストケースも生成できたらとても便利になります。これを実現したのが Excel の Book である **PictMaster** です。PictMaster は、CUI ベースの PICT に Excel の GUI（グラフィカルユーザインターフェース）ベースの皮をかぶせます。イメージ的には**図２－１**のようになります。

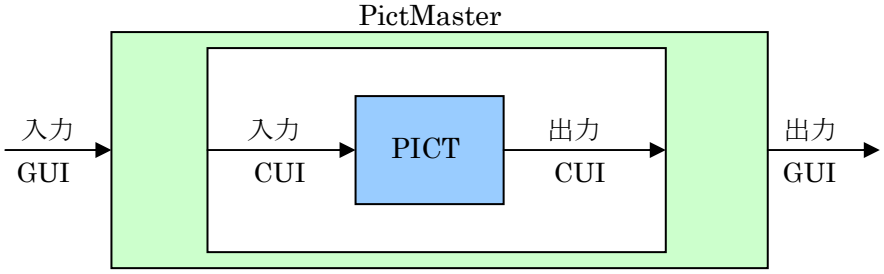

図２－１　PictMaster のイメージ

図２－１に示すように、ユーザからは PICT の存在はまったく見えません。GUI ベースですべての作業を行なうことができます。

PictMaster は次に示す５つのソフトの連携で動作します。

（１）　Excel の VBA
（２）コマンドプロンプト
（３）　バッチファイル
（４）　nkf
（５）PICT

VBA（Visual Basic for Application）は、Excel 用のプログラミング言語（Visual Basic）です。PictMaster では VBA を使用することで Excel のさまざまな GUI をコントロールします。またモデルファイルの作成、バッチファイルの作成、コマンドプロンプトの起動およびバッチファイルの実行も行ないます。さらにユーザの指定に応じて生成結果の並び替え、罫線を描くなどの処理も行ないます。

コマンドプロンプトのバッチファイルは、nkf の実行と PICT の実行を行ないます。

nkf は、モデルファイルと PICT が出力したファイルの文字コードの変換を実行します。nkf はオープンソースのソフトウェアを扱うサイトである sourceforge.jp で公開されているフリーソフトです。nkf の URL は以下のとおりです。

https://sourceforge.jp/projects/nkf/

PICT はモデルファイルの構文解析と組み合わせ生成エンジンの役割を果たします。

以上、述べた内容を含めた PictMaster でテストケースを生成するイメージを**図２－２**に示します。

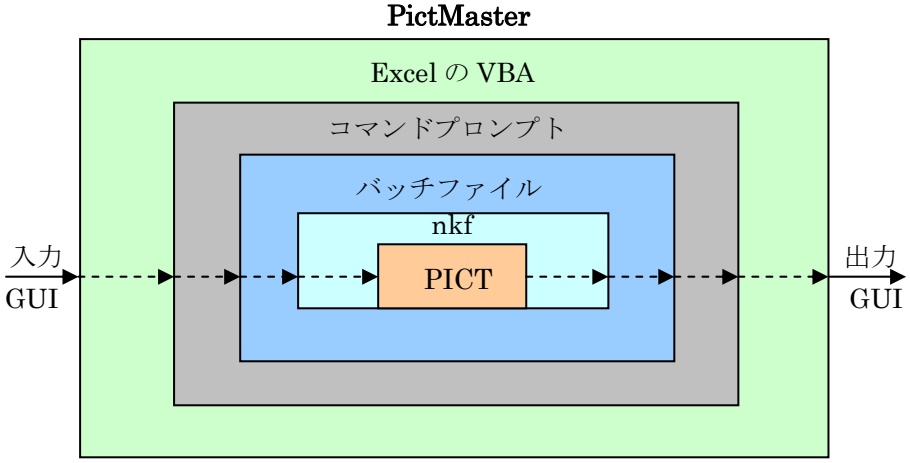

図２－２　より詳しい PictMaster でテストケースを生成するイメージ

図２－２はイメージであり、実際にコマンドプロンプト上で PICT と nkf を制御しているのはバッチファイルです。

PictMaster は PICT に GUI を持たせて使いやすくするだけでなく、PICT にはない機能を追加して機能強化も行なっています。

３．PictMasterの使い方

PictMaster は、Excel 2000 以降の Excel で動作します。Windows Vista、Windows XP、Windows 2000 での動作を確認しています。PictMaster を使用するためには以下のものを用意します。

（１）PICT そのもの
（２）Excel 2000 以降の Excel

PICT は以下のサイトからダウンロードすることができます。
http://msdn.microsoft.com/en-us/testing/bb980925.aspx

PictMaster のデフォルトの画面イメージの例を**図３－１**に示します。

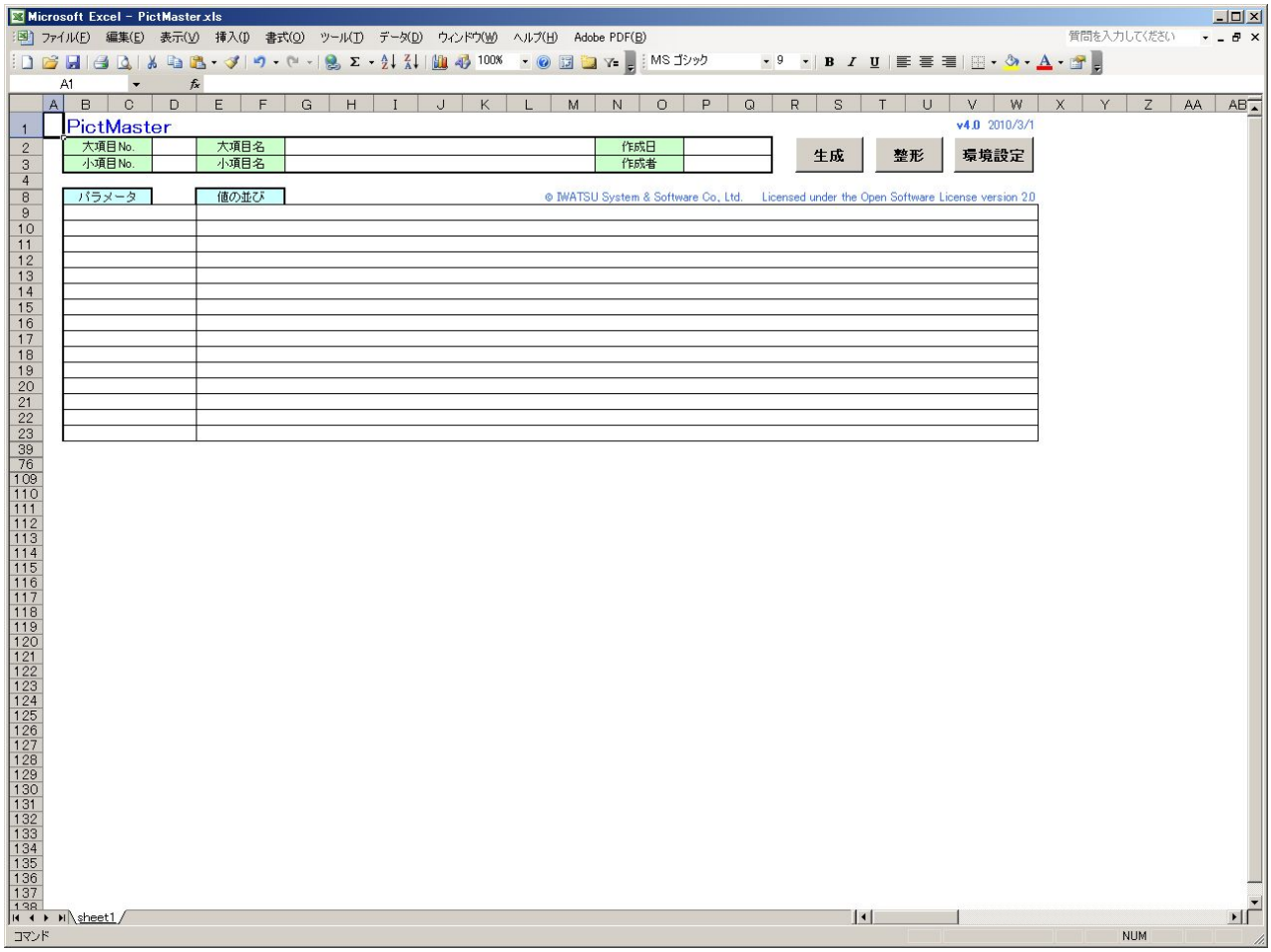

図３－１　PictMaster のデフォルトの画面イメージ

PictMaster は以下の各部分からなっています。

１～７行目　フリーエリア

ユーザが任意にレイアウト可能なエリアです。テスト大項目番号、小項目番号、作成日、作成者など、実際にユーザが使いやすいようにレイアウトを決めてください。なお５～７行目は非表示になっているため、そのエリアを使いたい場合は書式メニューから行の再表示を行なってください。
デフォルトのフリーエリアのレイアウトを**図３－２**に示します。

1	PictMaster					
2	大項目No.		大項目名		作成日	
3	小項目No.		小項目名		作成者	
4						

図３－２　デフォルトのフリーエリアのレイアウト

９～３８行目　パラメータ欄と値の並び欄

パラメータと、値の並びをカンマ（ **,** ）で区切って記入します。パラメータの末尾にコロン（ **:** ）は不要です。パラメータと値の並び欄は３０行で固定です。デフォルトでは１６行目以降は非表示となっています。値の並び欄には３０個までの値を記入することができます。いずれの欄も**行を開けずに詰めて記入**してください。値にはエイリアス記号（|)、無効値記号（~）および重みづけ指定の（n）を付加することができます。

この欄を編集する際の注意点があります。行の削除、挿入は行なわないで下さい。かわりに行のクリア、コピーと貼り付けで対応してください。コピー元の行と貼り付け先の行が重なると正しく貼り付けされません。コピー先が重ならないようあらかじめ間をあけておくようにしてください。第１列目の部分を右クリックすると**編集専用のショートカットメニュー**が表示され、行の挿入、行の削除、元に戻す、を行なうことができます。パラメータ、値の並び欄の例を**図３－３**に示します。

パラメータ	値の並び
A	a1, a2, a3
B	b1, b2, b3, b4
C	c1, c2, c3
D	d1, d2, d3, d4
E	e1, e2, e3

図３－３　パラメータ、値の並び欄の例

※パラメータ欄の最初の行（９行目）には半角大文字の“ID”で始まる名称は使用しないでください。

４１～４２行目　サブモデル欄

デフォルトの状態では非表示となっており、記入内容は無効です。サブモデル欄は後述する環境設定フォームで**「サブモデルを使用」**をチェックすることで表示され、記入内容が有効となります。任意の数のサブモデルを記入することができます。サブモデル欄の例を**図３－４**に示します。

サブモデルについては３．５章で説明します。

サブモデル
B, C, 2

図３－４　サブモデル欄の例

４６～７５行目　制約表欄

デフォルトの状態では非表示となっており、記入内容は無効です。制約表欄は後述する環境設定フォームで**「制約表を使用」**をチェックすることで表示され、記入内容が有効となります。制約の内容を表形式で記入します。３０行まで用意されています。デフォルトでは１６行目以降は非表示となっています。この欄を編集する際は、パラメータ、値の並び欄と同じ注意事項があります。

制約表欄の例を**図３－５**に示します。

制約表

パラメータ	制約1	制約2	制約3
A	a1	a2, a3	
B	b1, b2	b3, b4	
C			c1
D			d2
E			e2

図３－５　制約表の例

制約表への記入のしかたは次の第４章で説明します。

７９～１０８行目　結果表欄

デフォルトの状態では非表示となっており、記入内容は無効です。結果表欄は後述する環境設定フォームで**「結果表を使用」**をチェックすることで表示され、記入内容が有効となります。結果表は組み合わせ内容に応じてあらかじめ期待する結果（確認内容）を記入する表です。３０行まで用意されています。デフォルトでは１６行目以降は非表示となっています。結果表を使用するとテストケース生成後、自動的にテストケースごとの結果内容欄に期待する結果が設定されます。

結果表の例を**図３－６**に示します。

結果表

結果内容	A	B	C
aaaとなる	a1	b1	
aaaとなる		b3, b4	c1
bbbとなる	♯a1		♯c1
cccとなる			

図３－６　結果表の例

結果表への記入のしかたは第５章で説明します。

２～３行目　「生成」、「整形」、「環境設定」ボタン

デフォルトのレイアウトでは、２～３行目の右端に**図３－７**に示す３つのボタンがあります。

図３－７　３つのボタン

３．１「生成」ボタン

パラメータ、値の並び欄などに必要な記入を行なった後に、このボタンを押すことでテストケースが**“a.xls”**という Book 名で作成されます。どのような条件でテストケースを生成するかを「環境設定」ボタンで指定します。

３．２「整形」ボタン

テストケースが生成された後で、行の並び替え、罫線を描く、など指定した条件でテストケースの形を整えることができます。「整形」ボタンをクリックすると、**図３－８**の例のようなフォームが表示されます。

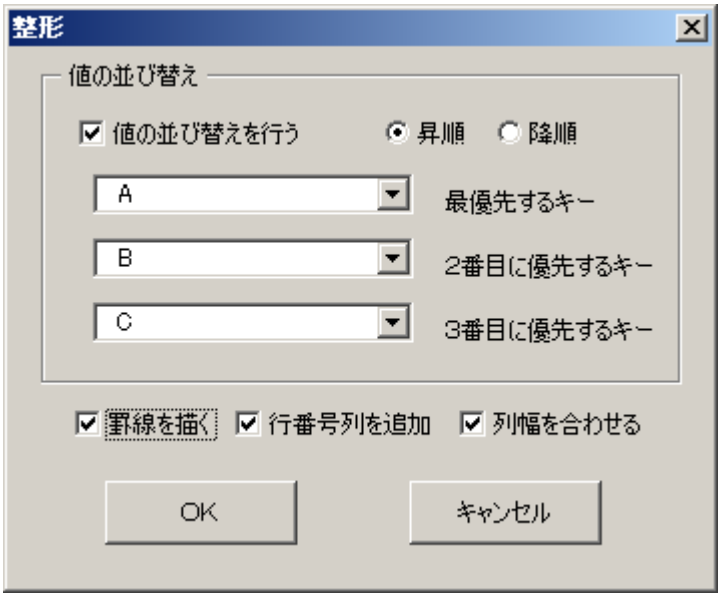

図３－８　「整形」ボタンのクリックで表示されるフォーム

並べ替えのキーには先頭から３つのパラメータが選択され、罫線を描く、行番号列を追加、列幅を合わせる、にすべてチェックが入れられます。すでに行番号列が追加されている場合は「行番号列を追加」にはチェックが入りません。もちろん、ユーザが好きなように手直しすることもできます。「OK」ボタンのクリックで処理が行なわれます。

生成結果を整形ボタンで整形した結果の例を**表３－１**に示します。

表３－１　整形されたテストケースの例

No.	A	B	C	D	E
1	a1	b1	c2	d1	e2
2	a1	b2	c1	d2	e2
3	a1	b2	c1	d1	e3
4	a1	b3	c2	d1	e1
5	a1	b4	c1	d1	e3
6	a2	b1	c1	d2	e3
7	a2	b2	c2	d1	e2
8	a2	b3	c1	d2	e1
9	a2	b3	c1	d2	e3
10	a2	b4	c2	d2	e2
11	a3	b1	c2	d1	e2
12	a3	b2	c2	d2	e2
13	a3	b3	c2	d2	e2
14	a3	b4	c1	d2	e1
15	a3	b4	c2	d1	e1

３．３「環境設定」ボタン

このボタンをクリックすると**図３－９**のフォームが表示されます。

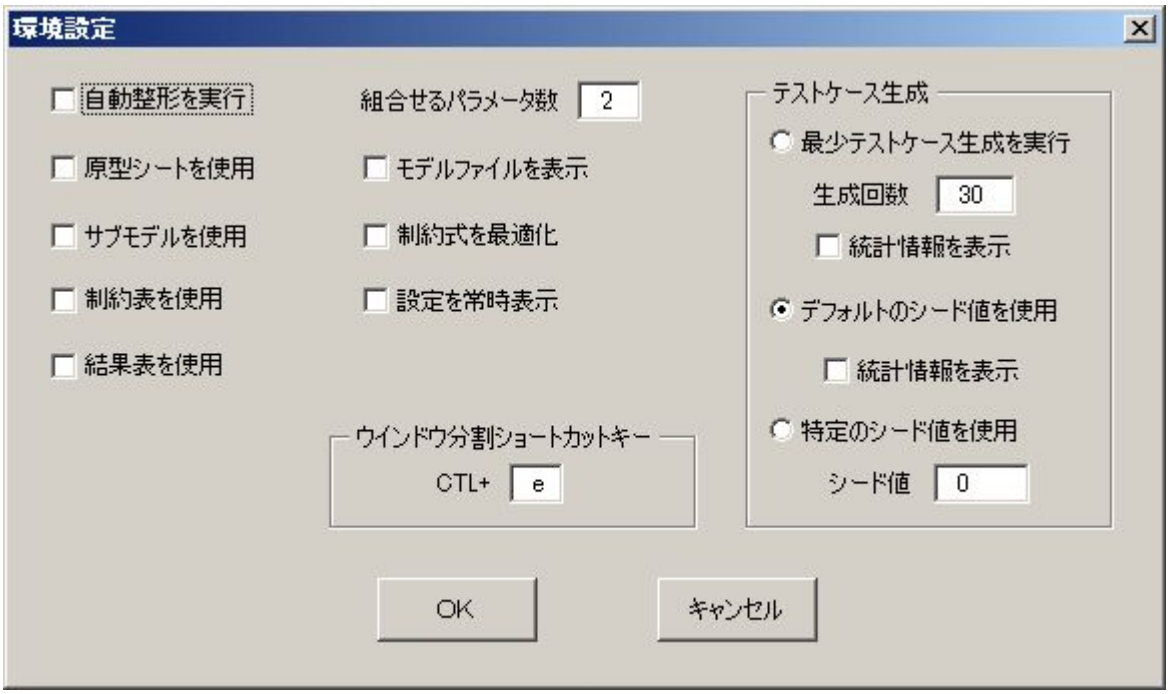

図３－９　環境設定ボタンのクリックで表示されるフォーム

「自動整形を実行」にチェックを入れて、テストケースの生成を行なうと、テストケースが生成された後、自動的にテストケースの整形が行なわれます。この際の整形の条件は、**図３－８**と同様な条件で行なわれます。異なる条件でテストケースの整形を行ないたい場合は、チェックを外し、テストケースが生成されてから「整形」ボタンをクリックして任意の条件を指定してください。

「原型シートを使用」にチェックを入れてテストケースの生成を行なうと、すぐ**右隣りのシートを原型シートとして扱い**ます。詳細は次の**３．３．１章**で説明します。
すぐ右隣りにシートがない、あったとしても原型ファイルの形式と異なる形式の場合はエラーメッセージが表示されます。原型シートの１桁目が**ダラーマーク（$）**の場合その行は**コメント行**として扱われます。原型シートは空白行を含まず詰めて記入してください。PictMaster の制限事項のため**原型シート上の各パラメータは同じ数の値が定義されている必要があります**。原型シート上の値のチェックは PictMaster では行なっていません。原型シートには 10000 個までのテストケースを記入できます。

「サブモデルを使用」にチェックを入れるとサブモデル欄が表示され、記入されたサブモデルが有効となります。詳細は**３．５章**で説明します。

「制約表を使用」にチェックを入れると先頭から１５行について、５０制約の制約表欄が表示され、記入された制約内容が有効となります。制約表欄は３０行設けていますので１５行で不足する場合は残りの行を再表示させてください。詳細は**第４章**で説明します。

「結果表を使用」にチェックを入れると１５行の結果表欄が表示され、記入された条件が有効となります。結果表欄は３０行設けていますので１５行で不足する場合は残りの行を再表示させてください。詳細は**第５章**で説明します。

「組合せるパラメータ数」には１～３０までの数字を指定します。「生成」ボタンのクリックで、指定されたパラメータ数での組み合わせが生成されます。「生成」ボタンのクリック時にパラメータ欄に記入されたパラメータ数を超えて指定されている場合、エラーメッセージが表示されます。

「モデルファイルを表示」にチェックを入れて、テストケースの生成を行なうと、テストケースが生成された後、PictMaster がパラメータ欄、値の並び欄、制約表などをもとに生成し、PICT に渡したモデルファイル“**a.txt**”がメモ帳によって表示されます。

「制約式を最適化」にチェックを入れてテストケースの生成を行なうと、パラメータの値の数が多い場合に制約指定が適切でないため、テストケース生成に時間がかかると判断されると、プログラム内部で制約指定の記入形式を最適化し、より短時間で生成が完了するようにします。詳細は**３．３．２章**で説明します。

「設定を常時表示」にチェックを入れると環境設定フォームの主な設定内容が、画面右上に常時表示されるようになります。これで設定内容を確認するためにいちいち環境設定フォームを開かなくて済みます。この表示内容のフォントの色やサイズ、太字にするなどは自由にカスタマイズすることができます。

「ウインドウ分割ショートカットキー」の入力欄に任意の半角１文字を入力しておくと、コントロールキーと入力したキーを押すことでPictMasterのウインドウが２つ開かれ、上下に整列され、下側のウインドウはパラメータ欄と制約欄との間で分割されます。この機能は多くの制約がある場合に制約表への記入をやりやすくするためのものです。結果表への記入時にも使えます。詳細は**６．３章**で説明します。

「最少テストケース生成を実行」を選択すると、テストケース生成の際、ランダムな初期条件（シード値）で PICT を実行し、最もテストケース数の少ないテストケースを生成結果として出力します。**「生成回数」**で何回テストケース生成を行なうかを指定します。２から９９９９回まで指定できます。　デフォルトは３０回です。

PICT は内部で固有のシード値（デフォルトは０）を使用してテストケースの生成を行なっています。このシード値を変えることより、生成される組み合わせ数が若干違ってきます。

最少テストケース生成実行中は、**図３－１０**の例に示すプログレスバーが表示されます。

図３－１０　最少テストケース生成中のプログレスバーの例

「統計情報を表示する」にチェックを入れると、最少テストケース生成が完了した時点で**図３－１１**の例に示す生成回数、最少数、最多数、初期数、最少時シード値、制約式最適化の実施有無およびテストケース生成にかかった経過時間が表示されます。こうした表示はデフォルトのシード値を指定して１回だけ生成する場合も行なうことが可能です。

図３－１１　表示される統計情報の例

初期数の値は、PICTのデフォルトの生成結果を表します。最少時シード値は、最少テストケースを生成したシード値の数値を表します。 この数値は後で説明する**「シード値」**の欄に自動的に設定されます。この状態で**「特定のシード値を使用」**を選択して生成を行なうと、１回の生成で最少テストケースと同じ生成結果を得ることができます。制約式最適化の表示は、最適化を行なった場合にTrue、行なわなかった場合にFalseとなります。環境設定で制約式の最適化を指定しても、モデルの値の個数が少ない場合、最適化は行なわれず、Falseが表示されます。詳しくは３．３．２章を参照してください。

ランダムな条件で生成した場合、生成されるテストケース数には最大で１０％程度のバラツキが発生し、多くの場合、最少テストケース生成を行なうことにより５％程度テストケース数を減らすことができます。生成回数を増やすほど、テストケース数を減らせる確率が高くなりますが、ほとんどの場合、３０回程度行なえば充分のようです。１件でもテストケース数を減らしたい場合は、生成回数に１００～３００程度の値を入力して最少テストケース生成を行なってみてください。ただし、パラメータと値の数が極端に多

く複雑な制約がある場合、あるいは多くの値を持つパラメータをサブモデルに指定した場合などは**１回のテストケース生成に非常に長い時間がかかる場合があります**ので注意してください。このような状態になった場合に、実行を中止したいときは**タスクマネージャで pict.exe を強制終了させてください**。この際、Excel のエラーメッセージが表示されますが「終了」ボタンをクリックしてください。

デフォルトのシード値（０）で生成したテストケース数が最も少なかった場合は、最少シード値には０が表示されます。

最少テストケース生成にかかる時間はどれくらいでしょうか。例として以下に示すスペックのノートパソコンでパラメータ数が５、パラメータあたり５個の値をもつモデルを実行した結果を**表３－２**に示します。

ＯＳ：　Windows XP　Professional Version 2002 SP2
CPU：　Intel Pentium M　1.6GHz
メモリ：512MB
Excel：　Excel 2000

表３－２　最少テストケース生成にかかった時間の例

生成回数	経過時間
１００	１２秒
３００	３１秒
９００	８９秒

最新の Excel 2007 を含めても Excel のバージョンの違いによる経過時間の変化は認められませんでした。

「デフォルトのシード値を使用」を選択すると、PICT のデフォルトのシード値（０）を使用してテストケースの生成を行なうことになります。

「特定のシード値を使用」を選択すると、**「シード値」**欄に設定された数値を用いてテストケースの生成を行なうことになります。「シード値」欄には０から６５５３５までの数値を設定することができます。

３．３．１　原型シートの使い方

原型シートには２つの使用方法があります。

(1) 前に使用したモデルを変更する必要がある場合、以前のモデルで作成した**テストケースを再利用**して、できるだけ少ない変更で新しいテストケースを作成します。
(2) 生成されるテストケースに必ず含まれるべきである**重要な組み合わせを指定**します。指定された組み合わせで出力が初期化され、次に残りの組み合わせが生成されます。

環境設定で「原型シートを使用」にチェックを入れてテストケースの生成を行なうと、すぐ右隣りのシートを原型シートとして扱います。すぐ右隣りにシートがない、あったとしても原型シートの形式と異なる形式の場合はエラーメッセージが表示されます。原型シートの１桁目が**ダラーマーク（$）**の場合その行は**コメント行**として扱われます。原型シートは空白行を含まず詰めて記入してください。原型シート上の値のチェックは PictMaster では行なっていません。原型シートには 10000 個までのテストケースを記入できます。

原型シートは PICT によって生成されたテストケースと同じフォーマットを使用します。コメント行を除いた最初の行にはパラメータ名を記入します。２行目以降の行は組み合わせられた値の並びを記入します。

【重要な注意】

(a) 原型シートが現在のモデルにないパラメータを含んでいる場合、そのパラメータの列全体が削除されます。
(b) 原型シートが現在のモデルにない値を含んでいる場合、そのパラメータの値の組み合わせは行

われません。現在のモデルに含まれるパラメータの値の組み合わせだけが生成されます。無効とされた値の列は不完全であるか部分的な列になります。

(c) 原型シートの値の列が現在の制約指定のどれかに違反する場合、その列は無視されます。

(d) 原型シートを使用する場合、**値の重み付け**（３．４．３章）**は正常に動作しません**。値の重み付けを行なって生成した結果を原型シートとして使用する場合は、PictMasterの値の並び欄から値の重み付けを取り除いてテストケースの生成を行なうことを推奨します。

原型シートの使用例を以下に示します。

例１：新しい値を追加する

	A	B	C
1	A	B	C
2	a1	b1	c2
3	a1	b2	c1
4	a2	b1	c1
5	a2	b2	c2
6	a3	b1	c1
7	a3	b2	c2

図３－１２　原型シートの例（その１）

パラメータ	値の並び
A	a1, a2, a3
B	b1, b2, b3
C	c1, c2

図３－１３　新しいモデルの例

新しいモデルでは、パラメータ B に値 b3 が追加されています。

表３－４　新しい値が追加されたテストケース

No.	A	B	C
1	a1	b1	c2
2	a1	b2	c1
3	a2	b1	c1
4	a2	b2	c2
5	a3	b1	c1
6	a3	b2	c2
7	a1	b3	c2
8	a3	b3	c1
9	a2	b3	c1

新しく生成されたテストケースには、原型シートの内容がそのまま流用され、追加した値 b3 に関する組み合わせが追加されています。これにより既存のテストケースでテストした後で仕様変更などにより、新しい値を追加してテストを行う必要が生じた場合、追加された値に関する組み合わせだけテストすればよい場合もあります。（注：**表３－４**は値の並べ替えを行っていません）

例２：必ず含まれなければならない組み合わせを指定する

	A	B	C
1	A	B	C
2	a1	b1	c1
3	a1	b1	c2
4	a1	b2	c1
5	a1	b2	c2

図３－１４　原型シートの例（その２）

パラメータ	値の並び
A	a1, a2, a3
B	b1, b2
C	c1, c2

図３－１５　モデルの例

原型シートでは値 a1 について、他のパラメータ B と C のすべての値と組み合わされています。このような、あるパラメータの特定の値についてのみ、ほかのすべてのパラメータが持つ値と組み合わせることは、PICT の柔軟な制約条件指定機能をもってしても指定することができません。したがって、このような**特殊または複雑な組み合わせは原型シートを使って指定します**。

表３－４　指定した組み合わせが含まれたテストケース

No.	A	B	C
1	a1	b1	c1
2	a1	b1	c2
3	a1	b2	c1
4	a1	b2	c2
5	a2	b1	c2
6	a2	b2	c1
7	a3	b1	c1
8	a3	b2	c2

新しく生成されたテストケースには、原型シートの内容がそのまま流用され、その他のパラメータの組み合わせが追加されています。この例のように、制約条件では指定できない複雑で特殊な組み合わせでも、原型シートで指定することにより、簡単に扱うことができます。

３．３．２　制約式の最適化

組み合わせ生成エンジンの PICT は、組み合わせできない組み合わせを指定するのに、IF 文に似たスクリプト言語（制約式）を用います。PictMaster では、制約表への記入を PICT が理解できる制約式に変換します。PICT の特性として、値の個数が多いパラメータに制約の指定を行なうと、制約の指定方法によって組み合わせの生成時間に大きな差が出ます。「制約式の最適化」を指定すると、PictMaster は制約指定が行なわれているパラメータの値の個数を調べ、８個以上の場合で、制約式の最適化の処理が有効な条件である場合、制約指定をプログラム内部で変更し、最も生成時間が短くなるような制約式を生成します。

制約指定の違いによる生成時間の違いを示すために**図３－１６**のモデルを使用することにします。使用した PC 環境 は、CPU が PentiumD　2.8GHz、RAM　1 GB、OS は Windows XP Professional SP3 です。

パラメータ	値の並び
A	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
B	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
C	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
D	1, 2, 3, 4, 5, 6, 7, 8, 9, 10
E	1, 2, 3, 4, 5, 6, 7, 8, 9, 10

図３－１６ 生成時間比較用のモデルの例

このモデルに次の制約指定を行なった場合の生成時間を示します。これらの制約指定は異なる記述ですがいずれも等価な制約指定です。生成時間は最少テストケース生成を３０回行なった場合の経過時間です。

（１）制約指定その１

制約表

パラメータ	制約1	制約2
A	1, 2, 3, 4, 5	
B	1, 2, 3, 4, 5	1, 2, 3, 4, 5
C		1, 2, 3, 4, 5
D		
E		

制約式最適化＝False　　生成時間＝７秒
制約式最適化＝True　　生成時間＝４秒

（２）制約指定その２

制約表

パラメータ	制約1	制約2
A	1, 2, 3, 4, 5	
B	#6, 7, 8, 9, 10	1, 2, 3, 4, 5
C		#6, 7, 8, 9, 10
D		
E		

制約式最適化＝False　　生成時間＝３秒
制約式最適化＝True とはならない。（すでに最適化の形式となっている）

（３）制約指定その３

制約表

パラメータ	制約1	制約2
A	#6, 7, 8, 9, 10	
B	1, 2, 3, 4, 5	#6, 7, 8, 9, 10
C		1, 2, 3, 4, 5
D		
E		

制約式最適化＝False　　生成時間＝７分２０秒
制約式最適化＝True　　生成時間＝４秒

（４）制約指定その４

制約表

パラメータ	制約1	制約2
A	#6, 7, 8, 9, 10	
B	#6, 7, 8, 9, 10	#6, 7, 8, 9, 10
C		#6, 7, 8, 9, 10
D		
E		

制約式最適化＝False　　生成時間＝７秒
制約式最適化＝True　　生成時間＝３秒

以上の結果から、制約指定が（３）の場合に制約式最適化の効果が最も顕著であることが分かります。それ以外では（２）の場合、最適化が行なわれないことがわかります。この制約指定では最適化が不要なためです。制約式最適化の処理が行なわれるのは以下の条件をすべて満たす場合です。

① 環境設定で「制約式を最適化」が指定されている。
② 制約指定のパラメータの値の個数が８個以上である。
③ 制約対象で指定された値の個数が２個以上である。
④ 制約条件が逆制約か、または制約対象が順制約である。

PictMaster は制約式の最適化が指定されると、①から④の条件をすべて満たす制約欄がある場合に、プログラム内部でその制約欄の記述形式を（２）の 制約指定その２ の形式に変換します。したがって、環境設定で「**制約式を最適化**」が指定されていても、実際には最適化が行なわれない場合があります。また制約式の最適化が行なわれたとしても、多くの制約指定の一部についてだけ行なわれた場合は、生成時間が短縮されない場合もあります。

パラメータとパラメータの比較を行なう制約指定は最適化の対象外です。

制約式の最適化が行なわれた場合は、制約表の制約指定とは異なる制約式が生成されることになるため、万が一、PICT 内部でエラーを検出した場合、エラーメッセージとして表示される制約式と、記述されている制約指定が対応しないことがあることに注意してください。

３．４　値の並び欄への記入のしかた

値の並びを半角のカンマ（ **,** ）で区切って記入します。値の前に特定の記号を付加することで特殊な処理を行なうことができます。特定の記号には**パイプ（|）**と**チルダ（~）**があります。パイプはひとつの値に複数の名称を与える**エイリアス**で使用します。チルダは相互に組み合わせ不可の値を指定する**無効値テスト**で使用します。値の後ろに**半角の括弧（ ）で囲まれた数字**を付加することで、その値が他の値より多く組み合わせに出現するようになる**値の重み付け**を行なうことができます。

これらの記号はパラメータおよび値の名称には使用できません。値の名称に半角の括弧を使いたい場合は代わりに半角の**ブラケット（ [] ）**または**全角の括弧**などを使ってください。

３．４．１　１つの値に複数の名称を与えるエイリアス

エイリアスは、１つの値に複数の異なる名称を与える機能です。まったく同一とは言えないがほぼ同一と考えることのできる複数の値に対して、同値分割の考え方を適用し、エイリアス機能を用いることができます。これにより組み合わせ生成の際に 1 つの値として扱われ、組み合わせ完成後、１つの値はエイリアスで与えた複数の名称に戻されます。

エイリアスを使用することによって生成される組み合わせの数を減らすことが可能です。Pairwise 法で生成される組み合わせの数は最も多くの値を持つパラメータ P1 と、P1 と同じか次に多くの値を持つパラメータ P2 の、それぞれの値の個数を V1n、V2n とすると、これらを積算した値か、それよりもやや多い値になります。生成される組み合わせ（テストケース）のおおよその数 TCn は次式で表されます。

$$TCn = (V1n * V2n + \alpha) * m$$

ここでαの値は、パラメータ P2 の次に多くの値を持つパラメータ P3 以降が持つ値の個数 P3n, P4n･･･によって異なります。この数が P2n と等しいかほとんど同じ場合は、多くの場合αは比較的大きな値になります。逆に P2n より小さい場合は、αは小さな値になります。この場合多くはαの値は 0 になります。このことから、エイリアスを適用するパラメータは、**最も多くの個数の値を持つパラメータを対象**にしたほうが、組み合わせ数を削減する効果が最も大きくなります。逆に、比較的少ない個数の値しか持たないパラメータに適用しても効果がありません。mは制約の数が 0 の場合 1 です。制約の数が増えるに従って徐々に増加しますが、増加の程度は緩やかです。

エイリアスを使用したモデルの例を**図３－１７**に示します。この例では OS 種別と HD 容量および HD インターフェースの組み合わせをテストします。このモデルでは OS 種別の 5 個の値から、Windows 系 3 個をエイリアスの記号“｜”で 1 つにまとめ、全体で 3 個にしています。エイリアスでまとめた場合、先頭の値の名称を使用することで制約などを記述することができます。

パラメータ	値の並び
OS種別	Windows Vista | Windows XP | Windows 2000, Lunux, Mac OS X
HD容量	250GB, 500GB, 750GB
HDインターフェース	USB2.0, IEEE1394, eSATA

図３－１７　エイリアスを使用したモデルの例

エイリアスを使用した**図３－１７**の組み合わせ生成結果を**表３－５**に、エイリアスを使用しなかった場合の組み合わせ生成結果を**表３－６**に示します。

表３－５　エイリアスを使用した場合

No.	OS 種別	HD 容量	HD インターフェース
1	Linux	250GB	IEEE1394
2	Linux	500GB	USB2.0
3	Linux	750GB	eSATA
4	Mac OS X	250GB	USB2.0
5	Mac OS X	500GB	eSATA
6	Mac OS X	750GB	IEEE1394
7	Windows 2000	250GB	eSATA
8	Windows Vista	750GB	USB2.0
9	Windows XP	500GB	IEEE1394

表３－６　エイリアスを使用しなかった場合

No.	OS 種別	HD 容量	HD インターフェース
1	Linux	250GB	eSATA
2	Linux	500GB	USB2.0
3	Linux	750GB	IEEE1394
4	Mac OS X	250GB	eSATA
5	Mac OS X	500GB	IEEE1394
6	Mac OS X	750GB	USB2.0
7	Windows 2000	250GB	USB2.0
8	Windows 2000	500GB	IEEE1394
9	Windows 2000	750GB	eSATA
10	Windows Vista	250GB	USB2.0
11	Windows Vista	500GB	eSATA
12	Windows Vista	750GB	IEEE1394
13	Windows XP	250GB	IEEE1394
14	Windows XP	500GB	eSATA
15	Windows XP	750GB	USB2.0

この例ではエイリアスを使用した場合は、使用しなかった場合に比べてテストケース数が 3 分の 2 未満に減少しています。**値の数の多いパラメータで同値と考えることのできる値がある場合は、エイリアスを積極的に使ったほうがよいでしょう。**

エイリアスで 1 つにまとめた複数の値のうち、後述する制約表や結果表で指定できるのは先頭の値の名称のみです。

※ 注意事項

エイリアスとして 1 つにまとめた値の数が多い場合は、**生成結果に一度も現れない値が存在する**可能性があります。これはエイリアスを含まないパラメータに属する値が少ない場合に発生します。エイリアスの値すべてを組み合わせに使わなくてもすべての組み合わせが網羅されてしまうからです。本来、エイリアスは同値とみなせるものを１つにまとめたものですから出現しない値があっても問題とは言えませんが、こうした性質があることは知っておいたほうがよいでしょう。

３．４．２　機能が動作しない無効値テストの方法

組み合わせテストでは通常、機能が動作しない無効値を含まないようにする必要があります。1 つのテストケースに 1 つの無効値を含むと機能が動作しなくなり、他のパラメータと組み合わせた意味がなくなります。

ここでは無効値を含むテストの実行を目的としている場合について説明します。1 つのテストケースに無効値を 2 つ以上含むと 1 つの無効値についてのテストとなり、残りの無効値についてのテストが行われなくなり、不完全なテストとなります。機能が動作しない場合の組み合わせでは、無効値どうしの組み合わせが行われないようにする必要があります。

PICT には無効値テストという機能があり、**無効値どうしの組み合わせが生成されない**ようにすることができます。値の並び欄で**値の前に半角の記号“ ~ ”を置く**ことで無効値を指定します。

図３－１８に無効値を含むモデルの例を示します。このモデルでは FAX と通信回線の組み合わせをテストします。FAX とは通信できない無効値である電話機が含まれています。さらに FAX 使用に制限がある IP 外線も無効値としています（実際には多くの場合、支障なく使用できます)。

パラメータ	値の並び
発信端末	ＦＡＸ，~電話機
通信回線	アナログ，ＩＳＤＮ，~ＩＰ外線
着信端末	ＦＡＸ，~電話機

図３－１８　無効値を含むモデルの例

表３－７　無効値テストの生成結果

No.	発信端末	通信回線	着信端末
1	~電話機	ISDN	FAX
2	~電話機	アナログ	FAX
3	FAX	~IP外線	FAX
4	FAX	ISDN	~電話機
5	FAX	ISDN	FAX
6	FAX	アナログ	~電話機
7	FAX	アナログ	FAX

表３－７が生成結果です。この生成結果には無効値どうしの組み合わせがありません。この例では無効値の場合でも無効値以外のすべてのペアを組み合わせています。そこまでの徹底したテストが不要な場合は、制約定義で各無効値１つに１つのテストケースのみが生成されるように指定することもできます。

値が複数の名前を持つエイリアスの場合、最初の名前に記号“ ~ ”を付けます。制約の記述で無効値を指定する場合は、記号“ ~ ”は省略します。

ここで示した方法は、PICTの無効値機能を使用する方法ですが、この場合、生成されるテストケースには無効値を１つも含まないテストケースも生成されてしまいます。PICTの無効値機能を使わずに無効値だけを含むテストケースを生成する方法を ６．６　効率のよい無効値テストを行なうには の章で説明しています。

３．４．３　値の重み付け

特定の値を重点的にテストしたい場合、重み付けの機能が役に立ちます。**重み付けを使用すると指定された値がより多くテストケースに現れるようになり、３要因間の網羅率が向上します**。重み付けは重点的にテストしたい値の右側に**半角の括弧（ ）**で数値を付加します。**図３－１９**に重み付けの例を示します。

パラメータ	値の並び
A	a1(2), a2, a3
B	b1, b2, b3, b4
C	c1, c2, c3(3)

図３－１９　重み付けの例

括弧内に**記入できる数値は２～１０**です。基本的には記入された数値をかけた分だけ他の値より多く組み合わせに出現するようになります。例えば**c3(3)**を記入した場合、値 **c3** は他の値 **c1**、**c2** より３倍多く出現するようになります。ただし、生成された組み合わせで重複する組み合わせが存在する場合、重複する組み合わせは１つを残し削除されて最終的な生成結果となります。

図３－１９で重複する組み合わせを含んだ生成結果を**表３－８**に示します。

表３－８　重複する組み合わせを含んだ生成結果

No.	A	B	C
1	a1	b1	c3
2	a1	b1	c3
3	a1	b1	c3
4	a1	b2	c1
5	a1	b2	c2
6	a1	b3	c3
7	a1	b3	c3
8	a1	b4	c1
9	a1	b4	c2
10	a1	b4	c3
11	a2	b1	c1
12	a2	b2	c3
13	a2	b3	c2
14	a2	b3	c3
15	a2	b4	c3
16	a3	b1	c2
17	a3	b2	c3
18	a3	b2	c3
19	a3	b3	c1
20	a3	b4	c3

表３－８でパラメータＣの各値の個数は、**c1**が４個、**c2**が４個、**c3**が１２個です。この時点で**c3**は他の値より３倍多く出現しています。これは重み付けの指定どおりです。しかし、網掛けした組み合わせが重複しています。組み合わせの重複が発生するのは、１つの値を重み付けにより３倍多く用いて組み合わせを生成しているため、他のパラメータの値の数が少ないと、同じ組み合わせが生成されるためです。

PictMaster は、重複した組み合わせのうち１つを残して他の組み合わせを削除します。そのため重複した組み合わせが存在する場合は、結果的に重み付けで指定した数値より少ない重み付けとなります。

最終的な生成結果を**表３－９**に示します。なお、**表３－８**とは異なる生成実行のため、組み合わせ内容は一部異なります。

表３－９　重複分を取り除いた最終的な生成結果

No.	A	B	C
1	a1	b1	c1
2	a1	b1	c3
3	a1	b2	c3
4	a1	b3	c2
5	a1	b3	c3
6	a1	b4	c1
7	a1	b4	c2
8	a2	b1	c3
9	a2	b2	c2
10	a2	b3	c1
11	a2	b3	c3
12	a2	b4	c3
13	a3	b1	c2
14	a3	b2	c1
15	a3	b3	c3
16	a3	b4	c3

最終的な生成結果では、重複していた４個の組み合わせが削除され、１６個の組み合わせとなりました。この結果では、パラメータＣの各値の個数は、**c1** が４個、**c2** が４個、**c3** が８個です。重み付けを行なって最少テストケース生成を行なうと、環境設定フォームで「統計情報を表示する」がチェックされていれば、**図３－２０**の統計情報が表示され、重複した組み合わせがいくつあったかが分ります。

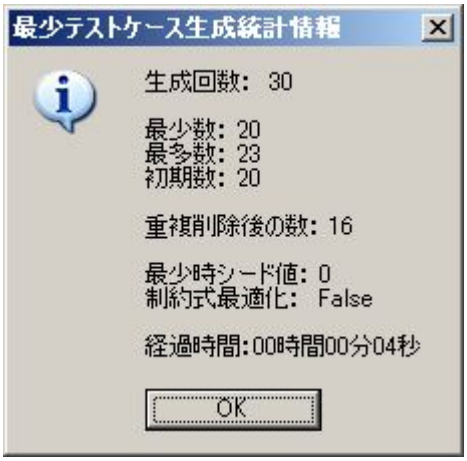

図３－２０　重み付けのあるモデルの生成統計情報の例

重み付けの数値が正確に反映された生成結果が得られるのは、他のパラメータの数が多い場合か、他のパラメータで値の数が多い場合です。この場合は重複が起こらず、生成されるテストケース数も重み付けを行なわない場合と比べてそれほど増加することはありません。そうでない場合、重み付けの数値は目安的な意味合いを持ちます。

重み付けを行なうと値の数が実質的に多くなります。入力できる値の個数は３０個までです。重み付けを行なう場合は実質的な値の個数が３０個を超えないようにしてください。

【重要な注意】

・値の重み付けと原型シート（３．３．１章）の機能を同時に使用すると重み付けは正常に動作しません。
原型シートの機能を使用する場合は値の重み付けは行わないようにすることを推奨します。

・重み付けを行なった値はエイリアスとして使用することはできません。

※　PICT 自体が備えている重み付けの機能は使用できません。

３．５　サブモデル

すべてのパラメータがすべて同じ重要度を持つというケースはそれほど多くはありません。PICT では**サブモデル定義**を使用することで、特に重要と考えられる限定したパラメータについて異なるパラメータ組み合わせ数を指定することができます。サブモデル欄を表示させるには環境設定フォームの「サブモデルを使用する」にチェックを入れます。

サブモデルの記入形式を**図３－２１**に示します。

＜パラメータ１＞,＜パラメータ２＞，･･･ ,＜組み合わせるパラメータ数＞

図３－２１　サブモデルの記入形式

サブモデルの対象としたい複数のパラメータ名を半角のカンマで区切って記入します。最後に組み合わせるパラメータ数を半角で記入します。この数は１から、指定したパラメータの数までの任意の値が記入できます。

サブモデルで指定された複数のパラメータは、“組み合わせるパラメータ数”の数のパラメータ間の組み合わせが生成されます。例えば、この数が３の場合、指定された複数（３個以上）のパラメータのみ３パラメータ間の組み合わせが生成されます。その後で、サブモデルで指定されなかった残りのパラメータと、２パラメータ間の組み合わせとして統合されます。最終結果として、サブモデルで指定された複数のパラメータどうしでは３パラメータ間の組み合わせとなりますが、サブモデルで指定されなかった残りのパラメータとの間では、１つ多い４パラメータ間の組み合わせとなります。そしてサブモデルで指定されなかった残りのパラメータどうしの間ではデフォルトの２パラメータ間の組み合わせとなります。

サブモデルは使い方が難しい面がありますが、使い方を工夫することによって非常に有効に使うことができます。詳細は ６．７　パラメータの重み付け の章を参照してください。

４．制約表への記入のしかた

４．１　制約に関する用語の定義

制約表の説明をする前に制約に関する用語の定義を明確にしておきます。

制約とは、組み合わせることのできないパラメータ（因子）と値（水準）のペアがあることを言います。

この制約を if 関係式 then 関係式 else 関係式 の形式で記述した式を**制約式**といいます。else 以降は省略される場合があります。**関係式**は、パラメータと値、またはパラメータとパラメータを比較するために = 、<> などを**演算子**として用います。PICT はこの制約式を入力として制約に対応した組み合わせを生成します。

if と then との間の関係式を**制約条件**といい、then 以降の関係式を**制約対象**と言います。then に続く関係式を**順制約**といい、else に続く関係式を**逆制約**といいます。制約条件にも順制約（=）と逆制約（<>）があります。

以上の用語を使って制約式を記述すると以下のとおりとなります。

if **制約条件** then **順制約** else **逆制約**

制約対象

以上の制約式を**条件付き制約式**といいます。制約条件のある制約式はすべて条件付き制約式です。これに対して、制約条件がなく、制約対象のみからなる制約式を**無条件制約式**といいます。無条件制約式ではパラメータのみが対象となります。基本的な無条件制約式の記述を以下に示します。

パラメータ　演算子　パラメータ

無条件制約はパラメータの値の如何にかかわらず、**常に成立する制約**です。

４．２　制約表の構成

未記入の制約表を**表４－１**に示します。

表４－１　未記入の制約表

制約表				
パラメータ	制約1	制約2	制約3	

表４－１でパラメータの列には９行目からのパラメータ欄の内容をコピーして貼り付けます。ただし**制約がない場合はパラメータ欄のコピーは不要です**。制約１、２、３、～ は、基本的には１つの制約式が１つの制約に対応します。パラメータ欄で指定されたパラメータの値のうち、任意の値を制約欄に記入します。また別のパラメータそのものを記入することもできます。

パラメータは３０個まで、制約は５０個まで記入できます。パラメータ欄は間に空白行を置かずに詰めて記入してください。同様に制約の各列も空白列を置かずに詰めて記入してください。たとえ空白列があっても無視されるだけで処理上の問題は起きませんが、画面に表示されていない列に制約が記入されていて、意図した結果が得られない事態も想定されますので、詰めて記入することを推奨します。

PictMaster は、パラメータ欄、値の並び欄および制約表欄の記述内容を、a.txt というモデルファイルに出力し、PICT に入力データとして渡します。この際、PictMaster は制約表の記入内容を PICT が理解できる制約式に変換します。

４．３　制約条件と制約対象の指定方法

制約条件と制約対象で、記述できる形式に差はありません。以降の説明で制約条件と制約対象の例として挙げられている記述は、制約条件と制約対象のどちらでも記述可能です。

４．３．１　条件付き制約

制約条件とする制約欄は白色以外の**任意の色で塗りつぶし**てください。塗りつぶされた制約欄に記入された値またはパラメータが制約条件となります。

制約対象とする制約欄は塗りつぶさないでください。白色で塗りつぶしても塗りつぶしなしとして扱われます。制約欄に記入された値またはパラメータが制約対象となります。

制約欄に複数の値を記入する場合はそれぞれの値を半角のカンマ(,) で区切ります。

先頭に**シャープ(#)** をつけると逆制約となり、記入した値以外の値を意味します。この場合もカンマで区切って複数の値を記入することができます。複数の値を記入した場合は、任意の１つ値の先頭に記入することができます。

先頭に**グレーターザン（>)、レスザン（<)** をつけることで値の大小比較ができます。あるパラメータの１つの値に“>”と“<”をつけることで任意の範囲の値を指定することができます。この場合はカンマ(,) で区切られた二つの値のＡＮＤ条件となります。値が数字だけの場合は問題ありませんが、数字と文字が混在する場合は全体が文字と見なされ、文字コードでの大小比較になります。

エイリアスの値を指定する場合は、先頭の値の名称を使用します。

パラメータそのものを記入する場合は、順制約の場合は先頭に**イコール(=)** をつけます。逆制約の場合は**エクスクラメーション(!)** を付けます。複数のパラメータを指定する場合はそれぞれのパラメータを半角のカンマ(,) で区切り、それぞれのパラメータにイコール(=) またはエクスクラメーション(!) を付けます。値を記入する場合と異なりますので注意してください。パラメータそのものを記入する場合はパラメータ欄と制約欄のパラメータ双方で少なくとも一部が同じ値を含んでいる必要があります。

制約条件でもなく制約対象でもない場合、制約欄は空白とします。

制約が１つもない場合は制約表のパラメータ欄は空白でもかまいません。

４．３．１．１　パラメータと値との制約

この章では、パラメータの値がある特定の値の場合に発生する制約の指定方法を説明します。
モデルが**表４－２**で、制約表が**表４－３**の場合、生成されるモデルファイル“**a.txt**”は**リスト４－１**となります。モデルファイルは、モデルそのものの部分を省略してあります。

表４－２　モデルの例（その１）

パラメータ	値の並び
A	a1, a2, a3
B	b1, b2, b3
C	c1, c2, c3

表４－３　制約表の例（その１）

制約表

パラメータ	制約1	制約2
A	a1,a2	a3
B	b1	#b1
C		

リスト４－１　制約式の例（その１）

```
if ([A] = "a1" or [A] = "a2" )
    then ([B] = "b1" ) ;
if ([A] = "a3" )
    then ([B] <> "b1" ) ;
```

表４－２のモデルで制約表が表４－４の場合、生成されるモデルファイル“a.txt”はリスト４－２となります。

表４－４　制約表の例（その２）

制約表

パラメータ	制約1	制約2
A	a1,a2	a3
B	b1	#b1
C	#c2, c3	c1

リスト４－２　制約式の例（その２）

```
if ([A] = "a1" or [A] = "a2" ) and ([B] = "b1" )
    then ([C] <> "c2" and [C] <> "c3" ) ;
if ([A] = "a3" )
    then ([B] <> "b1" ) and ([C] = "c1" ) ;
```

これまでの例で分かるようにある制約の１つの欄に複数の値が記入されている場合は、それぞれの値は基本的には OR 条件になります。これに対してある制約の異なる行に値が記入されている場合は、それぞれの値は AND 条件になります。これはパラメータそのものが記入された場合も同様です。

制約条件として異なるパラメータの値を OR 条件で指定したい場合は、異なる制約に制約条件として異なるパラメータの値を記入し、制約対象は同じにします。

制約対象として異なるパラメータの値を OR 条件で指定したい場合は、すぐ右側の制約に制約条件として同じパラメータの値を記入し、制約対象に異なるパラメータの値を記入します。表４－５にこの場合の制約表を示します。このとき生成される制約式をリスト４－３に示します。複数の制約条件がある場合はすべての制約条件が同一である必要があります。

表４－５　制約表の例（その３）

制約表

パラメータ	制約1	制約2	制約3
A			a1
B	b1	b3	b3
C	c1	c2	

リスト４－３　制約式の例（その３）

```
if ([B] = "b1" )
    then ([C] = "c1" ) ;
if ([B] = "b3" )
    then ([C] = "c2" ) or ([A] = "a1" ) ;
```

この例では制約２と制約３の制約条件が１つの制約式として統合され、制約対象が OR 条件となっています。制約対象がいくつあっても同じ制約では AND 条件となり、異なる制約間では OR 条件となります。ただし**隣り合う制約の制約条件の欄を異なる色にすると制約式の統合は行われず、２つの異なる制約式となります。**

４．３．１．２　パラメータとパラメータとの制約

次に**制約欄にパラメータを指定する場合**を示します。このときのモデルを**表４－６**、制約表を**表４－７**、生成される制約式を**リスト４－４**に示します。

表４－６　モデルの例（その２）

パラメータ	値の並び
A	a1, a2, a3
B	1, 2, 3
C	1, 2, 3
D	1, 2, 3

表４－７　制約表の例（その４）

制約表

パラメータ	制約1	制約2
A	a1	a3
B		=C
C	!B	
D		

リスト４－４　制約式の例（その４）

```
if ([A] = "a1" )
    then ([C] <> [B] ) ;
if ([B] = [C] )
    then ([A] = "a3" ) ;
```

パラメータを指定する場合は、指定する側と指定される側のパラメータの値に一部でも**同じ値が含まれている必要があります**。また、指定する側と指定される側の値の種類（文字列か数値）が一致しなければなりません。１つの欄に複数のパラメータを記入することもできます。この場合はそれぞれのパラメータの前にイコール（=）またはエクスクラメーション（!）を付加し、カンマ（,）で区切ります。各パラメータはＯＲ条件での指定となります。

１つの欄にＡＮＤ条件で複数のパラメータを指定する場合は演算子の前に**アンパサンド（&）**を付加します。
このときのモデルを**表４－６**、制約表を**表４－８**、生成される制約式を**リスト４－５**、生成されるテストケースを**表４－９**に示します。

表４－８　制約表の例（その５）

制約表

パラメータ	制約1	制約2
A	a1	#a1
B	=C, &=D	!C, !D
C		
D		

リスト４－５　制約式の例（その５）

```
if ([B] = [C] and [B] = [D] )
    then ([A] = "a1" ) ;
if ([B] <> [C] or [B] <> [D] )
    then ([A] <> "a1" ) ;
```

表４－９　テストケースの例（その１）

No.	A	B	C	D
1	a1	1	1	1
2	a1	2	2	2
3	a1	3	3	3
4	a2	1	2	3
5	a2	2	3	1
6	a2	3	1	2
7	a3	1	3	2
8	a3	2	1	3
9	a3	3	2	1

４．３．２　無条件制約

無条件制約には制約条件がありません。そのため、**無条件制約は常に成立する制約です**。制約表に記入できるのはパラメータのみです。制約欄に演算子付きで記入することで、２つのパラメータ間の制約を指定します。

使用できる演算子には、**イコール（=）**、イコールと逆の意味を表す**エクスクラメーション（!）**があります。１つの制約欄に**カンマ（,）**で区切って複数のパラメータを指定することができます。この場合、それぞれの OR 条件となりますが、２つめ以降のパラメータの先頭に**アンパサンド（&）**を置くことで AND 条件とすることができます。

モデルを**表４－１０**、ＡＮＤ条件で逆制約の制約表を**表４－１１**、生成される制約式を**リスト４－６**、生成されるテストケースを**表４－１２**に示します。

表４－１０　モデルの例（その３）

パラメータ	値の並び
A	a1, a2, a3
B	1, 2, 3
C	1, 2, 3
D	1, 2, 3

表４－１１　制約表の例（その６）

制約表

パラメータ	制約1
A	
B	!C
C	!D
D	!B

リスト４－６　制約式の例（その６）

```
([B] <> [C] ) and ([C] <> [D] ) and ([D] <> [B] ) ;
```

表４－１２　テストケースの例（その２）

No.	A	B	C	D
1	a1	1	3	2
2	a1	2	1	3
3	a1	3	2	1
4	a2	1	2	3
5	a2	2	3	1
6	a2	3	1	2
7	a3	1	3	2
8	a3	2	1	3
9	a3	3	2	1

この例では１つの式になっていますが、記入する制約を別にすると、異なる独立した式になります。その場合でも意味的には同一の制約です。

ＯＲ条件で順制約の制約表を**表４－１３**、生成される制約式を**リスト４－７**、生成されるテストケースを**表４－１４**に示します。**ＯＲ条件を指定するには無条件制約の右側の制約で、先頭にプラス（+）を置きます**。

表４－１３　制約表の例（その７）

制約表

パラメータ	制約1	制約2	制約3
A			
B	=C		
C		+=D	
D			+=B

リスト４－７　制約式の例（その７）

```
([B] = [C] ) or ([C] = [D] ) or ([D] = [B] ) ;
```

表４－１４　テストケースの例（その３）

No.	A	B	C	D
1	a1	1	1	2
2	a1	2	2	3
3	a1	3	3	1
4	a2	1	2	1
5	a2	2	3	2
6	a2	3	1	3
7	a3	1	3	3
8	a3	2	1	1
9	a3	3	2	2

１つの制約欄に複数のパラメータを記入した例として、モデルを**表４－１５**、制約表を**表４－１６**、生成される制約式を**リスト４－８**、生成されるテストケースを**表４－１７**に示します。

表４－１５　モデルの例（その４）

パラメータ	値の並び
A	a1, a2, a3
B	1, 2, 3
C	1, 2, 3
D	1, 2, 3
E	1, 2, 3
F	1, 2, 3

表４－１６　制約表の例（その８）

制約表

パラメータ	制約1
A	
B	=C, !D
C	
D	!E, &=F
E	
F	

リスト４－８　制約式の例（その８）

```
([B] = [C] or [B] <> [D] ) and ([D] <> [E] and [D] = [F] ) ;
```

表４－１７　テストケースの例（その４）

No.	A	B	C	D	E	F
1	a1	1	1	3	1	3
2	a1	1	3	2	1	2
3	a1	2	2	2	3	2
4	a1	3	2	1	2	1
5	a2	1	3	3	2	3
6	a2	2	2	3	1	3
7	a2	2	3	1	3	1
8	a2	3	1	2	3	2
9	a3	1	1	1	3	1
10	a3	1	2	2	3	2
11	a3	2	1	1	2	1
12	a3	3	3	3	1	3

４．４　使用できる演算子の一覧

関係式で使用できる演算子はパラメータと値、パラメータとパラメータで異なります。**表４－１８**に使用できる演算子を示します。#を除いていずれの演算子もカンマ（,）で区切って複数回記述することができます。条件の欄は１つの制約欄に値またはパラメータを複数記述した場合にＯＲ条件となるのかＡＮＤ条件となるのかを表します。

表４－１８　使用できる演算子

関係式の対象	演算子	条件	説明
パラメータと値	（指定しない）	OR	パラメータに属する値を表します。= と同じ意味です。
	#	AND	記述した値を除いた残りの値を表します。任意の１つの値について先頭に記述しますが値は複数記述することができます。
	>, <	AND	値の大小関係を表します。
パラメータとパラメータ	=	OR	パラメータ間の値が等しいことを表します。
	!	OR	パラメータ間の値が異なることを表します。
	&=	AND	パラメータ間の値が等しいことをAND条件で表します。パラメータを複数記述した場合に、２つめ以降のパラメータの前に記述します。
	&!	AND	パラメータ間の値が異なることをAND条件で表します。パラメータを複数記述した場合に、２つめ以降のパラメータの前に記述します。
	+	(OR)	無条件制約で異なる制約で OR 条件となることを表します。無条件制約の右側の制約で無条件制約の最初のパラメータの前に記述します。

表中の演算子（+）の条件欄が(OR）となっているのは、異なる制約間の条件を表しているためです。

これらの**演算子と同じ記号をパラメータおよび値の名称の先頭に使用することはできません**。

４．５　ダミーの値について

モデルの制約によってはダミーの値が必要となる場合があります。**ダミーの値とは、そのパラメータが意味をなさなくなることを表す値です**。ダミーの値としては**ハイフン（－）**などの一見してダミーと分かる値にします。

ダミーの値が必要となる例として、ビジネスホンシステムでの**３者会議通話**の組み合わせテストの例を以下に示します。このときのモデルを**表４－１９**、制約表を**表４－２０**、生成された制約式を**リスト４－９**、生成結果を**表４－２１**に示します。この例でパラメータの通話種別は外線側の回線数とシステム内の内線端末の台数を表しています。パラメータの端末種別２と端末種別３の値にダミーの“－”が含まれています。

表４－１９　モデルの例（その３）

パラメータ	値の並び
回線種別	アナログ，ＩＳＤＮ，ＩＰ外線，内線
通話種別	外線１内線２，外線２内線１，内線３
端末種別１	ＫＴ，ＤＣＬ，ＳＬＴ
端末種別２	ＫＴ，ＤＣＬ，ＳＬＴ，－
端末種別３	ＫＴ，ＤＣＬ，ＳＬＴ，－

表４－２０　制約表の例（その５）

制約表

パラメータ	制約1	制約2	制約3
回線種別	＃内線	＃内線	内線
通話種別	外線１内線２	外線２内線１	内線３
端末種別１			
端末種別２	＃－	－	＃－
端末種別３	－	－	＃－

リスト４－９　制約式の例（その６）

```
if ([通話種別] = "外線１内線２" )
    then ([回線種別] <> "内線" ) and ([端末種別２] <> "－" ) and ([端末種別３] = "－" ) ;
if ([通話種別] = "外線２内線１" )
    then ([回線種別] <> "内線" ) and ([端末種別２] = "－" ) and ([端末種別３] = "－" ) ;
if ([通話種別] = "内線３" )
    then ([回線種別] = "内線" ) and ([端末種別２] <> "－" ) and ([端末種別３] <> "－" ) ;
```

表４－２０の制約表で通話種別が内線端末を２台必要とする場合は、端末種別３のみ“－”とし、１台のみ必要な場合は端末種別２と端末種別３を“－”とし、内線端末を３台必要とする場合は端末種別２と端末種別３を“－”とは組み合わせないように指定しています。

※ ある制約欄に多くの値が記入された場合はすべての値が見やすいように行の高さを変えてみてください。

表４－２１　テストケース生成結果

No.	回線種別	通話種別	端末種別1	端末種別2	端末種別3
1	IP外線	外線1内線2	KT	DCL	－
2	IP外線	外線1内線2	SLT	KT	－
3	IP外線	外線1内線2	SLT	SLT	－
4	IP外線	外線2内線1	DCL	－	－
5	ISDN	外線1内線2	DCL	KT	－
6	ISDN	外線1内線2	KT	SLT	－
7	ISDN	外線1内線2	SLT	DCL	－
8	ISDN	外線2内線1	KT	－	－
9	アナログ	外線1内線2	DCL	KT	－
10	アナログ	外線1内線2	KT	DCL	－
11	アナログ	外線1内線2	SLT	SLT	－
12	アナログ	外線2内線1	SLT	－	－
13	内線	内線3	DCL	DCL	DCL
14	内線	内線3	DCL	SLT	SLT
15	内線	内線3	DCL	SLT	KT
16	内線	内線3	KT	KT	DCL
17	内線	内線3	KT	KT	SLT
18	内線	内線3	KT	KT	KT
19	内線	内線3	SLT	DCL	KT
20	内線	内線3	SLT	DCL	SLT
21	内線	内線3	SLT	SLT	DCL

表４－２１の生成結果を見ると、端末種別のパラメータが不要なケースでは不要な台数分だけ各パラメータにダミーの値が割り当てられていることが分かります。

この例で示したように、モデル作成時にダミーの値が必要かどうか検討を行ないます。必要であれば値としてダミーを追加します。そして制約表に記入する際に、ダミーの値と組み合わせ可能なパラメータか不可能なパラメータかをすべてのパラメータについて検討する必要があります。その検討結果を制約表に記入します。

４．６　制約表の編集方法

制約表の編集方法について説明します。

Excel のメニューからの**制約表の行の削除、挿入は行なわないでください。**制約表の行数が変化すると次に続く結果表欄の行位置がずれてしまいます。

行の削除や挿入を行ないたい場合はその１つの**行の１桁目を右クリック**してください。追加、削除、元へ戻すなどの**編集専用のコンテキストメニュー**が表示され、メニューを選択することで行の削除や挿入を行なうことができます。

制約の削除や挿入を行ないたい場合は**制約のタイトル欄を右クリック**してください。追加、削除、元へ戻すなどの編集専用のコンテキストメニューが表示され、メニューを選択することで制約の削除や挿入を行なうことができます。

いずれも最後の行や列では無効です。**この機能は結果表でも有効です。**

４．７　ワイルドカードの使用

制約表に記入する値には**ワイルドカード**を使用することができます。ワイルドカードとは、「任意の文字」を意味する特殊な文字です。「*」は０個以上の任意の文字を、「?」は１個の任意の文字を意味します。

例えば「A*」は名称が「A」で始まるすべての値を意味します。「A」だけも含まれます。「???A」は４文字で最後が「A」で終わるすべての値を意味します。

同様に結果表の一致条件の値にもワイルドカードを使用することができます。

ワイルドカードは、制約表では条件付制約の値の名称のみに使用することができます。値のタイプが数値の場合はワイルドカードを使用することはできません。あるパラメータの値が数字の値と文字の値が混在している場合、値のタイプは文字となります。

ワイルドカードの使用例と PictMaster が生成する制約式を以下に示します。

表４－２２　ワイルドカードの使用例

制約表

パラメータ	制約１
A	
B	*B*
C	#C??

リスト４－１０　ワイルドカードを含む制約式

```
if ([B] LIKE "*B*" )
    then ([C] NOT LIKE "C??" ) ;
```

値の名称に適切な文字列を付与することで、ワイルドカードを使用して制約指定を簡潔に記述することができるようになります。

５．結果表への記入のしかた

５．１　結果表の構成

テストケースの組み合わせによってテスト結果が異なる場合があります。テストケースの数が多い場合は、期待する結果（確認内容）を記入するのに手間がかかりますが、結果表を使用することでこの手間をなくすことができます。

未記入の結果表を**表５－１**に示します。

表５－１　未記入の結果表

結果表		
結果内容	パラメータ１	パラメータ2

結果表の**結果内容欄**には、右側の各パラメータの、値の組み合わせの場合にどのような結果となるべきかを記入します。記入内容が長くなる場合は、結果内容欄に **(*1) (*2) …** などの番号を記入し、生成されたテストケースが書かれたワークシートの欄外で番号ごとに具体的な確認内容を記述すればよいでしょう。結果内容欄は３０行設けられており、デフォルトでは１５行分表示されています。不足する場合は残りの行を再表示してください。

結果内容欄は１行目から**間を空けずに詰めて記入**してください。

結果表の**パラメータ欄**の列には、左側の結果内容欄の結果となる値の組み合わせ（**一致条件**）を制約表の記述と同じ方法で記入します。ただしパラメータそのものを記入することはできません。

パラメータ欄の列は３０列設けられています。

５．２　一致条件の指定方法

一致条件は制約表で値を指定する場合と同じ方法で指定します。左側の結果内容欄に書かれている結果となる値の組み合わせを必要なパラメータごとに記入します。１つの欄には値をカンマ（**,**）で区切って複数記入することができます。先頭にシャープ（**#**）をつけると逆条件となり、指定した値以外の値を意味します。色の塗りつぶしは不要です。

環境設定フォームで「結果表を使用」にチェックを入れて OK をクリックすると、モデルのパラメータ欄に記述されているパラメータ名が結果表のパラメータ欄に自動的に記入されます。

一致条件欄の記入は制約表の記述に違反しないようにします。**結果内容欄に記入があり、すべてのパラメータの一致条件欄になにも記入がない場合**は、他の結果内容欄の一致条件にいずれにも一致しなかったテストケースについて結果内容欄にその記述内容が記入されます。

表５－２にモデルの例、および**表５－３**に結果表の例を示します。

表５－２　モデルの例

パラメータ	値の並び
A	a1, a2, a3, a4
B	b1, b2, b3, b4
C	c1, c2, c3

表５－３　結果表の例

結果表

結果内容	A	B	C
1111となる	a1	b1, b2	
1111となる	a2	b3, b4	
2222となる	#a1, a2		c1, c2
3333となる			

結果表の記入を行なう際は、**CTL-e**（デフォルト）を押してウインドウの分割を行ない、モデルと結果表がともに見えるようにすると記入しやすくなります。

表５－３では結果内容欄には４つの確認内容が記入されており、最後の確認内容には一致条件が何も記入されていません。それまでの３つの条件に一致しなかったテストケースに最後の確認内容が記入されます。最初の２つの結果内容欄は同じ内容です。このような指定も可能です。

一致条件はパラメータ間の AND 条件となります。記入されたすべての条件が一致するテストケースのみに左側の確認内容が記入されます。記入された値がエイリアスの場合はそのすべての値を意味します。

表５－２から**表５－３**の条件で生成されたテストケースの例を**表５－４**に示します。

表５－４　生成されたテストケース

No.	A	B	C	結果内容
1	a1	b1	c3	1111 となる
2	a1	b2	c1	1111 となる
3	a1	b3	c1	3333 となる
4	a1	b4	c2	3333 となる
5	a2	b1	c1	3333 となる
6	a2	b2	c3	3333 となる
7	a2	b3	c2	1111 となる
8	a2	b4	c3	1111 となる
9	a3	b1	c3	3333 となる
10	a3	b2	c3	3333 となる
11	a3	b3	c2	2222 となる
12	a3	b4	c1	2222 となる
13	a4	b1	c2	2222 となる
14	a4	b2	c2	2222 となる
15	a4	b3	c3	3333 となる
16	a4	b4	c1	2222 となる

生成されたテストケースには右端に結果内容の欄が追加され、一致条件に一致したテストケースについて結果表の結果内容欄の記述が記入されています。テストケース数が数十個以上と多い場合は結果表機能があると時間の節約になります。生成を繰り返した場合は時間短縮効果が大変大きくなります。

エイリアスの値を指定する場合は、先頭の値の名称を使用します。

５．３　使用できる演算子の一覧

一致条件で使用できる演算子を**表５－５**に示します。#を除いていずれの演算子も１つの値をカンマ（,）で区切って複数回記述することができます。条件の欄は１つの一致条件欄に複数の値を記述した場合に OR 条件となるのか AND 条件となるのかを表します。

表５－５　使用できる演算子

関係式の対象	演算子	条件	説明
パラメータと値	（指定しない）	OR	パラメータに属する値を表します。= と同じ意味です。カンマ（,）で区切って複数記述することができます。
	#	AND	記述した値を除いた残りの値を表します。値の先頭に１つだけ記述しますが値は複数記述することができます。

５．４　記入上の注意事項

結果表の記入に関する注意事項を説明します。

(1) 一致条件の重複

複数の確認内容を定義した場合、同じ組み合わせに２つの異なる確認内容が一致することがあります。これは一致条件が重複する条件を含んでいるためです。この状態となった場合、PictMaster は**図５－１**の例のようなエラーメッセージを表示し、そこで処理を中止します。このエラーとなった場合は、条件が重複しないように「#」を使った逆条件を追加するとうまくいきます。

図５－１　一致条件重複のエラーメッセージの例

(2) 一致条件の不一致

一致条件に一致する組み合わせが１つもない場合があり得ます。たとえば組み合わせるパラメータ数を２としている場合に一致条件として３つ以上のパラメータの値について指定した場合、起きる可能性が高くなります。また制約表の記述に違反した組み合わせを指定する一致条件の場合もこのエラーとなります。この状態となった場合、PictMaster は**図５－２**の例のようなエラーメッセージを表示し、そこで処理を中止します。

図５－２　一致条件不一致のエラーメッセージの例

６．より便利な使い方

６．１　PictMasterのカスタマイズ

PictMaster は、Excel の Book であることから使いやすいようにカスタマイズすることが可能です。１～７行目は自由にレイアウトしてかまいません。PictMaster のファイル名、シート名は任意の名前に変更してかまいません。

PictMaster で**編集メニューから「シートの移動またはコピー」を選択し、表示されたフォームの「コピーを作成する」にチェックを入れ「ＯＫ」ボタンをクリックすることで異なるテストケースを生成するシートを任意の枚数設けることができます**。これに対して、挿入メニューからワークシートを選択して新しいシートを作成し、PictMaster のシートをコピーして貼り付けてもそのシートでは正常に動作しませんので注意してください。

テスト対象の大きな機能ごとに PictMaster の Book を設け、そのいくつかの組み合わせテストケースのモデルを複数のシートに分けて管理するという方法がよいかもしれません。またテスト仕様書を Excel で作成している場合は､PictMaster を使用してテストケースを作成したテスト仕様書を PictMaster の別シート上に記述し、**テスト仕様書と PictMaster を１つの Book に統合することも可能です**。その場合、「PictMaster」という Book 名はテスト対象を表す機能名などの名称に変更することになるでしょう。このような例でのシート名の並びを**図６－１**に示します。

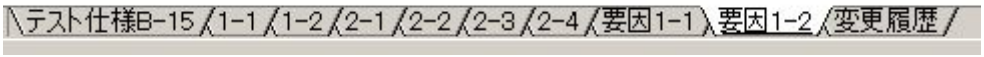

図６－１　シート名の並びの例

この例ではシート名「テスト仕様 B-15」がテストの記号名であり Book 名でもあります。1-1～2-4 のテストケースの操作方法、確認内容、データ設定内容などを記述した**テスト設計仕様書(*1)**のシートです。1-1～2-4 のシートは個々の確認内容に応じた**テストケース仕様書(*1)**です。これらのテストケースのうち、“要因”がシート名についているシートが PictMaster を使用して組み合わせテストケースの作成に使用した PictMaster のシートです。

カスタマイズする際、パラメータ、値の並び、サブモデル、制約表および結果表の文字が記入されたセルの行番号と列番号は変えないで下さい。VBA がモデルなどの位置を認識できず実行できなくなります。

***1: IEEE 829 標準規格に準拠したドキュメント。テスト手順書に該当する内容は、テスト設計仕様書またはテストケース仕様書で包含しているので設けていない。**

６．２　エラー／警告メッセージが表示された場合

PictMaster では制約表から制約式に変換する過程で多くのチェックを行なっており、PICT 自体からエラー/警告メッセージが表示されることはまれだと思います。

エラー/警告メッセージが表示されている間はそのメッセージ内容が記述されているファイル “**e.txt**” が存在するので、メモ帳などで e.txt のファイルを開くことができます。ファイルを開いた後で、エラー（警告）メッセージの OK ボタンをクリックすれ ば、メモ帳などでエラー/警告メッセージを見ながら PictMaster のパラメータ定義や制約表の間違いを調べることができます。また警告メッセージの場合は、警告メッセージの OK ボタンをクリックしても、テストケースは生成されているので“**a.xls**”のファイルを開いて生成結果を確認することができます。

６．２．１　矛盾した制約をすばやく見つけるには

PICT から表示されるメッセージで多いものとして**図６－２**に示すようなメッセージがあります。

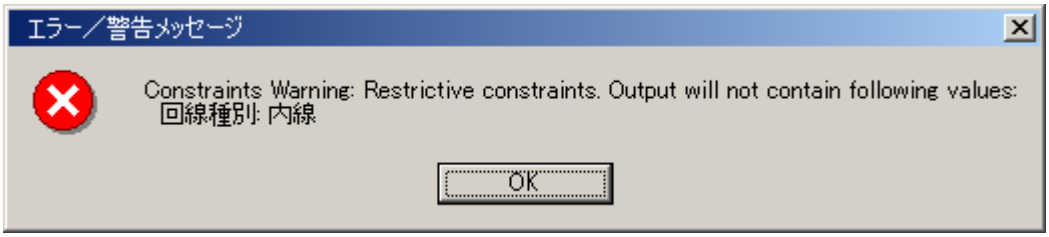

図６－２　PICT が表示するメッセージの例

この例はパラメータ“回線種別”で値“内線”が組み合わせに１つも含まれていないことを警告するメッセージです。この例では１つだけですが場合によっては５～６個の指摘がなされることもあります。

こうしたメッセージが出る原因は制約の指定に誤って相互に矛盾する複数の制約を指定したためです。間違った制約を突き止めるには指摘されたパラメータの値が組み合わせに表れないような矛盾した制約の指定を行なっていないか制約表で各制約の関係を見直すことです。いくつも指摘された場合はどれか１つに的を絞って調べます。

このような警告が現れる最も単純な制約指定を以下に示します。

制約表

パラメータ	制約1	制約2
A	a1	a2
B	b1	b1

図６－３－１　矛盾した制約指定の例（**その１**）

この制約指定は制約条件の値とそれに対応する制約対象の値が不一致であるため値 a1 が組み合わせに現れないことになっています。この制約指定の誤りは一目見れば分かりますが、実際には多くの制約が関係した結果として、こうした制約条件と制約対象との矛盾が起こります。

基本として**矛盾した制約かどうかは、同じ行に位置する２つのセルの値が異なる場合、値が同じでなくても矛盾しない関係にあるかどうかを調べる**ことにあります。

６．３　画面を分割し制約表を記入しやすくする

制約の数が多くなると右に横スクロールしなければならないため、各パラメータの値の名称が見えなくなります。そのため左に横スクロールして名称を確認しなければなりません。また結果表の記入を行なう際もパラメータの値が見えず面倒です。これを頻繁に行なうことは煩わしいので簡単に記入できる方法を紹介します。

環境設定ボタンを押すと表示される「ウインドウ分割ショートカットキー」に任意の１文字（デフォルトは“e”）を入力しておくと、コントロールキーを押しながらショートカットキーを押すことで PictMaster のウインドウが２つ開かれ、上下に整列され、下側のウインドウはパラメータ欄と制約欄との間で分割されます。なおこの機能は複数の PictMaster の Book が開かれた状態では**最初に開いた Book でのみ有効**です。分割したい Book が最初に開いた Book でない場合は、一旦すべての Book を閉じて、最初に Book を開いてください。

図６－５にウインドウ分割ショートカットキーを押した後の画面例を示します。

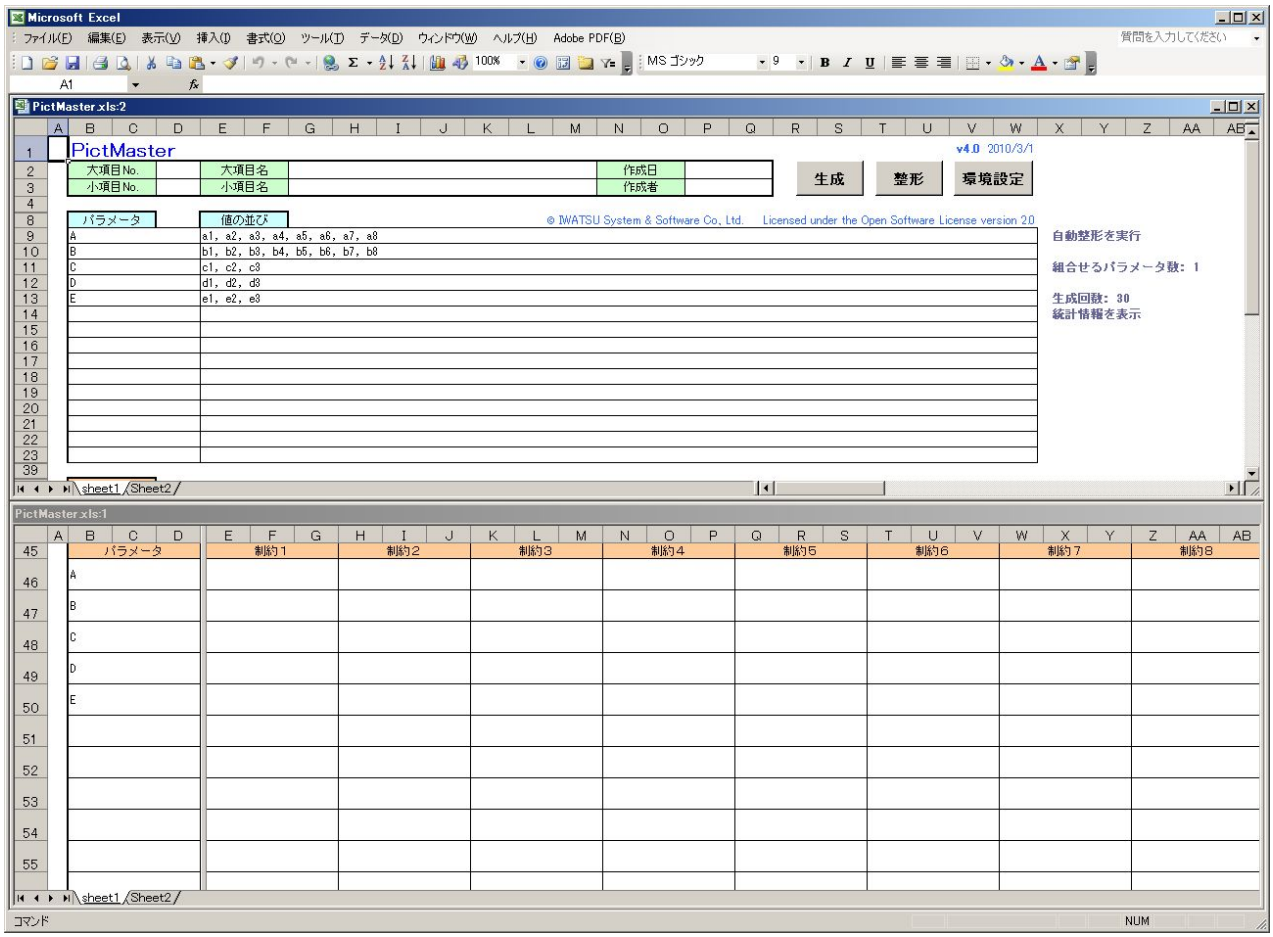

図６－５　ウインドウ分割の画面例

この状態の画面では制約表を横スクロールしてもパラメータと値が常に見えているので多くの制約を必要とする場合に制約表への記入がしやすくなります。

この状態で再度コントロールキーを押しながらショートカットキーを押すと分割前の画面に戻ります。

なお他のシートや Book に一旦切り替えた場合、元の画面に戻るとウインドウ分割の状態が変化しています。ここで再度ウインドウ分割のショートカットキーを２～３回押すことで正しい分割画面とすることができます。他のシートを表示したい場合は**画面の下のウインドウからシートを選択することで、選択したシートの本来のズーム（倍率）で表示させることができます**。画面の上のウインドウから他のシートを選択すると常に 100%のズームで表示されます。

６．４　デシジョンテーブルテストと組み合わせテストの統合

テスト対象によってはデシジョンテーブル（以降 DT と記述）と組み合わせテストを合わせて行いたい場合があります。DT を組み合わせテストと統合したい場合、組み合わせテストケースごとに DT のすべてのルールを組み合わせると全体のテストケース数が非常に多くなり実用的でなくなる場合があります。DT 全体を組み合わせテストのパラメータの１つとして扱い、各ルールをパラメータの値として扱うことで DT と組み合わせテストを合理的に統合することが可能となります。

方法を順番に述べると以下の通りとなります。ここでの DT の例は JaSST'07 Tokyo での「三賢者、テストを語る　（DTvsCEGvsCFD）」**http://www.jasst.jp/archives/jasst07e/pdf/A5.pdf** の DT の「入場料問題」（34 ページ目）を使っています。

（1）　最初にデシジョンテーブルそのものを作成します。各ルールが列ではなく、行となっていることに注意してください。

表６－１　作成したデシジョンテーブル

	6歳未満	小学生	一般	65歳以上	県内在住	個人	料金
ルール1	Y	N	N	N	－	－	無料
ルール2	N	N	N	Y	－	－	無料
ルール3	N	Y	N	N	Y	－	無料
ルール4	N	Y	N	N	N	Y	600
ルール5	N	Y	N	N	N	N	500
ルール6	N	N	Y	N	－	Y	1200
ルール7	N	N	Y	N	－	N	1000

（2）　例として組み合わせるパラメータが**図６－６**とします。

パラメータ	値の並び
A	a1, a2, a3, a4, a5
B	b1, b2, b3, b4, b5
C	c1, c2, c3, c4, c5

図６－６　組み合わせるパラメータ

（3）　**図６－６**のモデルに、DT 全体を１つのパラメータとして追加します。（**図６－７**）。

パラメータ	値の並び
A	a1, a2, a3, a4, a5
B	b1, b2, b3, b4, b5
C	c1, c2, c3, c4, c5
DT	ルール１, ルール２, ルール３, ルール４, ルール５, ルール６, ルール７

図６－７　DT をパラメータとして追加したモデル

（4）**図６－７**のモデルで組み合わせを生成します。生成結果を**表６－２**に示します。この結果、DT の各ルールは、他のパラメータと２パラメータ間の組み合わせを網羅したものとなります。

表６－２　DT を統合した生成結果

No.	A	B	C	DT
1	a1	b1	c3	ルール6
2	a1	b2	c1	ルール7
3	a1	b3	c5	ルール2
4	a1	b4	c1	ルール4
5	a1	b4	c3	ルール1
6	a1	b5	c2	ルール3
7	a1	b5	c4	ルール5
8	a2	b1	c2	ルール1
9	a2	b2	c3	ルール3
10	a2	b3	c2	ルール7
11	a2	b3	c3	ルール4
12	a2	b3	c3	ルール5
13	a2	b4	c4	ルール2
14	a2	b5	c1	ルール6
15	a2	b5	c5	ルール1
16	a3	b1	c5	ルール7
17	a3	b1	c5	ルール4
18	a3	b2	c1	ルール1
19	a3	b3	c4	ルール6
20	a3	b4	c2	ルール5
21	a3	b4	c5	ルール3
22	a3	b5	c3	ルール2
23	a4	b1	c1	ルール2
24	a4	b1	c4	ルール1
25	a4	b2	c2	ルール4
26	a4	b2	c4	ルール7
27	a4	b2	c5	ルール5
28	a4	b3	c1	ルール3
29	a4	b4	c2	ルール6
30	a4	b5	c3	ルール7
31	a5	b1	c1	ルール5
32	a5	b1	c4	ルール3
33	a5	b2	c2	ルール2
34	a5	b2	c5	ルール6
35	a5	b3	c5	ルール1
36	a5	b4	c3	ルール7
37	a5	b5	c4	ルール4

最後に生成結果の各ルールの値を実際の DT のルールと置き換えて完成です。

６．５　テスト実施中に組み合わせを修正する

テスト担当者が組み合わせテストを行なっている最中に、**実施できないテストケースを見つける**ことがあります。組み合わせが制約のため実施できない組み合わせとなっている場合です。テストケースを生成する際は、すべての制約を考慮して実施できない組み合わせが生成されないようにしますが、テスト仕様書作成の基となる機能仕様書に不備がある、組み合わせ生成後のチェックでの見落とし、などの理由で**実施できない組み合わせのテストケースがまぎれこんでしまう**ことがあります。

制約を修正しただけで再度テストケースを生成すると、組み合わせが全く異なるテストケースとなります。テストの初めの時点で見つけた場合は、制約を修正し、再度テストを実施すれば済みますが、テストの半分以上を実施してから見つけたような場合は、最初からテストをやり直さざるを得ず、それまでの**多くの工数が無駄になります**。これは組み合わせテストを実施する上での大きな問題です。

こうした問題にその場で対処できる方法があります。**原型シート**の機能を用いることで、それまで実施したテストを無駄にすることなく、残ったテストケースを実施することができます。

手順を以下に示します。

（１）PictMaster のワークシートの右隣りに原型シート用として新しいワークシートを挿入します。
（２）問題の見つかったテストケースのワークシートから、PictMaster が生成した部分（パラメータ名、値）をすべてコピーします。
（３）PictMaster の原型シートの左上（A1）のセルをクリックし、右クリックで「形式を選択して貼り付け」を選び、「貼り付け」のグループから「値」を選択して「ＯＫ」をクリックします。
（４）原型シートに貼り付けたテストケースの中から、実施できない組み合わせの原因となる値だけを削除します。
（５）実施できないテストケースが生成されないように PictMaster の制約表を修正します。
（６）環境設定フォームで「原型シートを使用する」をチェックし、「ＯＫ」ボタンをクリックしてからテストケースの生成を行ないます。
（７）生成されたテストケースを元のテストケースのワークシートに貼り付けます。

後は基本的には、残ったテストケースについてテストを再開するだけです。以下に例を用いて説明します。なお例で用いているテストケースの組み合わせは最少テストケース生成で生成したため、必ずしも各ユーザで同じ結果になるとは限らない点に留意してください。

図６－８のモデル、**図６－９**の制約表、**表６－３**のテストケースがあります。

パラメータ	値の並び
A	a1, a2, a3, a4, a5
B	b1, b2, b3, b4
C	c1, c2, c3

図６－８　モデルの例

制約表

パラメータ	セット1
A	a5
B	b4
C	

図６－９　制約表の例（その１）

表６－３　テストケースの例（その１）

No.	A	B	C
1	a1	b1	c3
2	a1	b2	c3
3	a1	b3	c2
4	a1	b4	c1
5	a2	b1	c1
6	a2	b2	c2
7	a2	b3	c1
8	a2	b4	c3
9	a3	b1	c2
10	a3	b2	c1
11	a3	b3	c2
12	a3	b4	c3
13	a4	b1	c2
14	a4	b2	c1
15	a4	b3	c3
16	a4	b4	c2
17	a5	b4	c1
18	a5	b4	c2
19	a5	b4	c3

この例では、No.13 までテストを実施したところでこの組み合わせでテストできないことが分かったとします。理由は、「**値 a4 は b4 とのみ組み合わせ可能であるのに、b4 以外と組み合わされている**」とします。

PictMaster で新規ワークシートを右隣りに挿入し原型シートとします。テスト仕様書から**表６－３**のテストケースのパラメータＡ、Ｂ、Ｃの列をコピーし、原型シートに貼り付けます。

次に原型シートから、誤った組み合わせである a4 と組み合わされているパラメータＢの値を削除します（**図６－１０**）。

	A	B	C
1	A	B	C
2	a1	b1	c2
3	a1	b2	c1
4	a1	b3	c2
5	a1	b4	c3
6	a2	b1	c2
7	a2	b2	c3
8	a2	b3	c1
9	a2	b4	c1
10	a3	b1	c1
11	a3	b2	c2
12	a3	b3	c1
13	a3	b4	c3
14	a4		c3
15	a4		c2
16	a4		c3
17	a4	b4	c1
18	a5	b4	c1
19	a5	b4	c2
20	a5	b4	c3

図６－１０　誤った組み合わせを削除した原型シート

続いて環境設定フォームから「**原型シートを使用**」にチェックを入れ、「ＯＫ」をクリックします。
次に制約表を**図６－１１**に修正します。制約条件に **a4** を追加し修正しました。

制約表

パラメータ	セット1
A	a5, a4
B	b4
C	

図６－１１　修正後の制約表

そしてテストケースの生成を行ないます。修正前のテストケースを以下に示します。

表６－４　修正前のテストケース

No.	A	B	C
1	a1	b1	c2
2	a1	b2	c1
3	a1	b3	c2
4	a1	b4	c3
5	a2	b1	c2
6	a2	b2	c3
7	a2	b3	c1
8	a2	b4	c1
9	a3	b1	c1
10	a3	b2	c2
11	a3	b3	c1
12	a3	b4	c3
13	a4	b1	c3
14	a4	b2	c2
15	a4	b3	c3
16	a4	b4	c1
17	a5	b4	c1
18	a5	b4	c2
19	a5	b4	c3

表６－５　修正後のテストケース

No.	A	B	C
1	a1	b1	c2
2	a1	b2	c1
3	a1	b3	c2
4	a1	b4	c3
5	a2	b1	c2
6	a2	b2	c3
7	a2	b3	c1
8	a2	b4	c1
9	a3	b1	c1
10	a3	b1	c3
11	a3	b2	c2
12	a3	b3	c1
13	a3	b3	c3
14	a3	b4	c3
15	a4	b4	c1
16	a4	b4	c2
17	a4	b4	c3
18	a5	b4	c1
19	a5	b4	c2
20	a5	b4	c3

修正前のテストケースでは、誤った組み合わせの部分を網掛けにしています。原型シートを使用し、制約を追加して生成した修正後のテストケースでは、修正前に比べて No.13 の網掛けしたテストケースが追加されています。それ以外では No.15、No.16 と No.17 で修正が反映されたほかは**修正前の内容とまったく同一**です。このことから、No.13 で止まったテストは、修正後のテストケースで No.13 からテストを再開することができ、**それまでに行ったテストが無駄にならずに済みます**。
修正後に No.13 が追加されているのは、制約が変わったことによる影響のためと考えられます。

【重要な注意】
値の重み付けと原型シートの機能を同時に使用すると値の重み付けが正常に動作しません。修正後のテストケースの生成は必ず値の重み付けがない状態にしてから行なってください。

６．６　効率のよい無効値テストを行なうには

PICT の無効値テストの機能を使用せずに、制約表で制約指定を行なうことで、１つのテストケースには１つの無効値だけが含まれるようにすることができます。

例えば**図６－１２**のモデルがあるとき、値 a1、b2、c3 が無効値である場合、PICT の無効値機能を使用すると**表６－６**のテストケースが得られます。

パラメータ	値の並び
A	~a1, a2, a3
B	b1, ~b2, b3
C	c1, c2, ~c3
D	d1, d2, d3
E	e1, e2, e3

図６－１２　モデルの例

表６－６　PICT の無効値機能を使用したテストケース

No.	A	B	C	D	E
1	~a1	b1	c1	d3	e1
2	~a1	b3	c1	d1	e3
3	~a1	b3	c2	d2	e2
4	a2	~b2	c1	d3	e3
5	a2	b1	c2	d2	e2
6	a2	b3	~c3	d3	e2
7	a2	b3	c1	d1	e2
8	a2	b3	c2	d1	e1
9	a2	b3	c2	d3	e3
10	a3	~b2	c1	d1	e2
11	a3	~b2	c2	d2	e1
12	a3	b1	~c3	d2	e3
13	a3	b1	c1	d3	e1
14	a3	b1	c2	d1	e3
15	a3	b1	c2	d2	e1
16	a3	b3	~c3	d1	e1
17	a3	b3	c1	d2	e3
18	a3	b3	c2	d3	e2

このテストケースで、色がついているテストケースが無効値テストに相当します。一方、色のついていないテストケースは無効値テストではなく通常のテストケースです。本来、無効値テストを行ないたいのですが、PICT の無効値テストの機能では、無効値テスト以外のテストケースも生成されてしまいます。それでは生成されたテストケースのうち、無効値テストのテストケースだけ実行すればよいかというと、そうではありません。色のついた無効値テストのテストケースだけテストを実施した場合、組み合わせに網羅されない値が出てしまいます。例えばc2 とd1 の組み合わせが含まれていません。

PICT の無効値テストでは無駄なテストケースが多く生成されてしまいます。PICT の無効値テストの機能は使用せずに、制約表を用いて無駄のないテストケースを生成することができます。この例での制約表を用いた方法での制約表の記述内容を**図６－１３**に示します。そしてその生成結果を**表６－７**に示します。

制約表

パラメータ	制約1	制約2	制約3	制約4	制約5
A	a1	#a1	#a1	#a1	a1
B	#b2	b2		b2	#b2
C	#c3		c3	#c3	
D					
E					

制約表

パラメータ	制約6	制約7	制約8	制約9	制約10
A		#a1	a1		
B	#b2	#b2		b2	
C	c3	c3	#c3	#c3	
D					
E					

図６－１３ 制約表による無効値テストの制約指定の例

この制約表では、制約１、制約４、制約７で無効値同士の組み合わせが生成されないよう指定しています。残りの６個の制約指定で、無効値でない場合に組み合わせ可能な値をOR 条件で指定しています。

この制約表で生成したテストケースを次に示します。

表６－７ 制約表による無効値テストケース

No.	A	B	C	D	E
1	a1	b1	c1	d3	e1
2	a1	b1	c2	d2	e3
3	a1	b3	c1	d2	e2
4	a1	b3	c2	d1	e1
5	a2	b1	c3	d1	e2
6	a2	b2	c1	d3	e1
7	a2	b2	c2	d2	e1
8	a2	b3	c3	d3	e3
9	a3	b1	c3	d3	e1
10	a3	b2	c1	d1	e3
11	a3	b2	c2	d3	e2
12	a3	b3	c3	d2	e2

このテストケースには、無駄なテストケースは含まれていません。すべてが無効値テストのテストケースとなっています。その結果、テストケース数はPICT の無効値機能を使用した場合の１８個に対して１２個で済んでいます。この方法は無効値を含むパラメータの値の個数が多くない場合に有効な方法です。

６．７　パラメータの重み付け

サブモデルの機能は使い方を工夫することによってきわめて有効な機能として働きます。

サブモデルを使用することで重要なパラメータのみ選択的に組合せ数を３としたり、重要でないパラメータのみ組合せ数を１としたりすることによって、生成されるテストケース数を合理的に減らすことが可能となります。

６．７．１　３つ以上の任意のパラメータのみ３パラメータの組み合わせとする

指定した３つ以上のパラメータのみ３パラメータ間の組み合わせとする方法を紹介します。すべてを３パラメータ間の組合せとするよりも少ないテストケースとすることができます。

基本的な手順は以下の通りとなります。

（１）３パラメータの組み合わせとしたい３つ以上のパラメータについてサブモデルで３パラメータの組み合わせを指定します。同時に環境設定で組み合わせるパラメータ数に１を指定します。
（２）以上の条件でテストケースを生成します。このときのテストケース数を控えておきます。
（３）生成されたテストケースを原型シートのワークシートに貼り付けます。
（４）環境設定で組み合わせるパラメータ数に２を指定します。そして原型シートを使用する設定とします。同時にサブモデルを使用しない設定とします。
（５）以上の条件でテストケースを生成します。このときのテストケース数が（２）項でのテストケース数より多いことを確認します。

ここで生成されたテストケースの集合は、指定した任意のパラメータのみが３パラメータの組み合わせとなり、残りのパラメータは２パラメータの組み合わせとなります。このときのテストケース数はすべてを３パラメータの組み合わせとした場合より顕著に少ない数となります。

モデルの構成によっては、この手順で最後の（５）項のテストケース数が（２）項でのテストケース数よりも少ない場合があり得ます。この場合は必要な組み合わせが得られていないので、対処が必要です。

この場合の対処法については次章で詳しく説明しています。

具体的なモデルの例を図６－１４に示します。このモデルではパラメータ A, B, C, D の４つのパラメータのみ３パラメータ間の組み合わせとし、残りのパラメータ E, F, G, H は２パラメータ間の組み合わせとします。

パラメータ	値の並び
A	1, 2, 3
B	1, 2, 3
C	1, 2, 3
D	1, 2, 3
E	1, 2, 3
F	1, 2, 3
G	1, 2, 3
H	1, 2, 3

サブモデル
A, B, C, D, 3

図６－１４　サブモデルで３パラメータの組み合わせを指定する

このとき環境設定で**組み合わせるパラメータ数を１**とします。

以上の条件で生成されたテストケースを次に示します。このときのテストケース数は**２８**となりました。（最少テストケース生成で生成しているため、必ずしも２８となるわけではありません）

表６－８　必要な３パラメータ間の組み合わせを生成する

No.	A	B	C	D	E	F	G	H
1	1	1	1	1	1	1	3	1
2	1	1	2	2	3	3	2	1
3	1	1	3	3	2	3	1	1
4	1	2	1	3	2	3	1	1
5	1	2	2	1	3	1	3	3
6	1	2	3	2	2	1	2	2
7	1	2	3	1	1	3	2	1
8	1	3	1	2	3	2	3	3
9	1	3	2	3	3	1	3	2
10	1	3	3	1	3	1	1	3
11	2	1	1	2	3	1	1	2
12	2	1	2	3	1	1	2	3
13	2	1	3	1	1	1	1	2
14	2	2	1	1	3	3	1	3
15	2	2	2	2	3	1	2	3
16	2	2	3	3	2	2	2	1
17	2	3	1	3	2	3	2	1
18	2	3	2	1	2	3	2	1
19	2	3	3	2	1	3	1	1
20	3	1	1	3	3	1	2	3
21	3	1	2	1	2	2	3	2
22	3	1	3	2	3	2	2	3
23	3	2	1	2	2	2	3	1
24	3	2	2	3	1	3	3	2
25	3	2	3	1	1	1	1	3
26	3	3	1	1	2	2	2	2
27	3	3	2	2	3	2	3	1
28	3	3	3	3	1	2	3	1

次に PictMaster に新規ワークシートを作成し、それを PictMaster のワークシートの右側に移動し、原型シートとします。そして先ほど生成した３パラメータの組み合わせ結果を原型シートに貼り付けます（No. の列は除外します）。

続いて環境設定で**サブモデルを使用しないに設定**し、**原型シートを使用するに設定**します。また**組合せるパラメータ数に２を設定**します。

この条件で生成されたテストケースを次に示します。

表６－９　３つ以上の任意のパラメータのみ３パラメータの組み合わせとしたテストケース

No.	A	B	C	D	E	F	G	H
1	1	1	1	1	1	1	3	1
2	1	1	2	2	3	3	2	1
3	1	1	3	3	2	3	1	1
4	1	2	1	3	2	3	1	1
5	1	2	2	1	3	1	3	3
6	1	2	3	2	2	1	2	2
7	1	2	3	1	1	3	2	1
8	1	3	1	2	3	2	3	3
9	1	3	2	3	3	1	3	2
10	1	3	3	1	3	1	1	3
11	2	1	1	2	3	1	1	2
12	2	1	2	3	1	1	2	3
13	2	1	2	3	1	2	3	2
14	2	1	3	1	1	1	1	2
15	2	2	1	1	3	3	1	3
16	2	2	2	2	3	1	2	3
17	2	2	3	3	2	2	2	1
18	2	3	1	3	2	3	2	1
19	2	3	2	1	2	3	2	1
20	2	3	3	2	1	3	1	1
21	3	1	1	3	3	1	2	3
22	3	1	2	1	2	2	3	2
23	3	1	3	2	3	2	2	3
24	3	2	1	2	2	2	3	1
25	3	2	2	3	1	3	3	2
26	3	2	2	1	2	2	1	3
27	3	2	3	1	1	1	1	3
28	3	3	1	1	2	2	2	2
29	3	3	2	2	3	2	3	1
30	3	3	3	3	1	2	3	1

このときのテストケース数は**３０**となりました。このテストケースはパラメータ A, B, C, D のみ３パラメータの組み合わせとなり、残りのパラメータは２パラメータの組み合わせとなっています。このテストケース数が先ほどの原型シートとした組み合わせのテストケース数より少なくないことを確認します。

こうした方法ではなく、単純に組み合わせるパラメータ数に３を指定した場合のテストケース数は**５６**となり、２倍近くのテストケース数となります。

６．７．２　生成されたテストケース数が原型シートのテストケース数より少ないときは

最後に生成されたテストケース数が、原型シートのテストケース数より少なくなる場合があり得ます。例えば以下のモデルの場合でA、Bそして Cのみを組合せ数３とした場合です。

パラメータ	値の並び
A	1, 2, 3
B	1, 2, 3
C	1, 2, 3
D	1, 2, 3
E	1, 2
F	1, 2
G	1, 2
H	1, 2

サブモデル
A, B, C, D, 3

図６－１４　対処が必要なケースのモデル

このモデルでは、パラメータ A、B、C,そして D のみ３パラメータの組み合わせとし、残りの組合せは２パラメータの組み合わせとしています。

このモデルで組み合わせるパラメータ数に１を指定して生成を行なうとテストケース数は**２９**となりました（**表６－１０**）。このテストケースを原型シートにペーストし、サブモデルを使用せず、組み合わせるパラメータ数に２を指定して生成を行なうと得られるテストケース数は**２０**となります（**表６－１１**）。この数は原型シートのテストケース数である**２９**より少ないので、パラメータ A、B、C,そして D の３パラメータの組み合わせを網羅していません。

こうしたことは３パラメータの組み合わせとしたパラメータの値の数が、それ以外のパラメータの値の数より多い場合に発生する傾向があります。この理由については PICT のマニュアルに記述がありませんが、おそらく PICT の仕様であると思われます。

この場合の対処として、サブモデルで指定しなかった残りのパラメータのうち、最も値の個数が少ないパラメータについて、任意の値を２つ重複して指定します。**図６－１４**のモデルでは、パラメータE～Hの値の数はどれも２個なので、ここではパラメータHの値「２」を**図６－１５**のように２つ重複して定義することにします。この対処は、前章の手順（**４**）で行ないます。

パラメータ	値の並び
A	1, 2, 3
B	1, 2, 3
C	1, 2, 3
D	1, 2, 3
E	1, 2
F	1, 2
G	1, 2
H	1, 2, 2

図６－１５　パラメータ H の値２を２つ定義する

このモデルで生成した正しいテストケースを**表６－１２**に示します。

表６－１０　組み合わせるパラメータ数に１を指定して生成したテストケース

No.	A	B	C	D	E	F	G	H
1	1	1	1	2	2	1	1	2
2	1	1	2	3	2	2	1	1
3	1	1	3	1	1	1	2	2
4	1	2	1	1	2	1	2	2
5	1	2	2	1	1	1	2	1
6	1	2	3	2	2	1	2	1
7	1	2	3	3	2	1	1	2
8	1	3	1	3	2	1	1	1
9	1	3	2	2	2	1	2	1
10	1	3	3	1	1	1	2	2
11	2	1	1	3	2	1	1	1
12	2	1	1	1	1	2	2	1
13	2	1	2	2	1	2	1	1
14	2	1	3	2	1	1	1	2
15	2	2	1	2	1	2	2	1
16	2	2	2	3	2	1	1	1
17	2	2	3	1	2	1	1	1
18	2	3	1	2	2	1	1	2
19	2	3	2	1	2	1	1	1
20	2	3	3	3	1	1	2	2
21	3	1	1	2	2	1	2	1
22	3	1	2	1	2	2	1	2
23	3	1	3	3	1	2	2	2
24	3	2	1	3	1	2	2	2
25	3	2	2	2	2	2	2	2
26	3	2	3	1	1	2	2	2
27	3	3	1	1	2	1	2	1
28	3	3	2	3	1	2	1	1
29	3	3	3	2	2	2	2	1

※パラメータ A、B、C および D の３パラメータの組み合わせを網羅したテストケース

表６－１１　原型シートを使用し組み合わせ数を２としたテストケース

No.	A	B	C	D	E	F	G	H
1	1	1	1	2	1	1	2	2
2	1	1	2	2	2	2	2	2
3	1	1	3	1	2	2	2	1
4	1	2	1	2	1	1	1	1
5	1	2	3	3	1	1	2	1
6	1	3	2	3	1	1	2	2
7	1	3	3	2	1	2	1	2
8	2	1	2	1	2	2	1	2
9	2	2	1	1	1	2	2	2
10	2	2	2	3	1	2	1	2
11	2	2	3	2	1	2	2	1
12	2	3	1	3	1	2	2	1
13	2	3	2	2	2	1	2	1
14	2	3	3	1	2	2	2	2
15	3	1	1	1	2	1	2	1
16	3	1	1	3	2	1	1	1
17	3	2	2	2	2	2	1	1
18	3	2	3	1	2	1	2	1
19	3	3	2	1	2	1	1	2
20	3	3	3	3	1	2	2	1

※パラメータＡ、Ｂ、ＣおよびＤの３パラメータ間の組み合わせを網羅していないテストケース

表６－１２　３パラメータの組み合わせと２パラメータの組み合わせを合成したテストケース

No.	A	B	C	D	E	F	G	H
1	1	1	1	2	1	1	2	2
2	1	1	1	3	2	2	2	2
3	1	1	2	2	2	2	2	2
4	1	1	3	1	2	2	2	1
5	1	2	1	2	1	1	1	1
6	1	2	2	1	2	2	2	1
7	1	2	2	3	2	1	1	2
8	1	2	3	3	1	1	2	1
9	1	3	1	1	1	2	2	1
10	1	3	2	3	1	1	2	2
11	1	3	3	2	1	2	1	2
12	2	1	1	2	1	1	1	2
13	2	1	1	2	1	2	2	2
14	2	1	2	1	2	2	1	2
15	2	1	3	3	1	2	2	1
16	2	2	1	1	1	2	2	2
17	2	2	2	3	1	2	1	2
18	2	2	3	2	1	2	2	1
19	2	3	1	3	1	2	2	1
20	2	3	2	2	2	1	2	1
21	2	3	3	1	2	2	2	2
22	3	1	1	1	2	1	2	1
23	3	1	1	3	2	1	1	1
24	3	1	2	3	1	2	2	2
25	3	1	3	2	1	1	2	2
26	3	2	1	3	1	1	1	1
27	3	2	2	2	2	2	1	1
28	3	2	3	1	2	1	2	1
29	3	3	1	2	2	2	1	1
30	3	3	2	1	2	1	1	2
31	3	3	3	3	1	2	2	1
32	3	3	3	1	1	1	1	2

※このテストケースでは、パラメータ A、B、C および D については３パラメータ間の組み合わせが網羅され、残りのパラメータについては２パラメータ間の組み合わせが網羅されています。

表６－１２のテストケースでは、テストケース数が**３２**となり、**表６－１０**の原型シートのテストケース数**２９**よりも少なくないので、パラメータ A、B、C および D については３パラメータ間の組み合わせが網羅され、残りのパラメータについては２パラメータ間の組み合わせが網羅されていることが保証できます。

ちなみに**図６－１４**でサブモデルを使用せず、組み合わせるパラメータ数に３を指定した場合に生成されるテストケース数は**３７**となります。

６．７．３　要因列挙テストでテストケースを削減する

ソフトウェア開発ではまったくの新規開発は少なく、大部分は既存製品をベースにしたソフトウェアの再利用をともなうソフトウェア開発です。

既存製品をベースにした新製品の開発では製品の各機能は、新製品で新たに追加された「**新規機能**」と、既存製品の機能をそのまま流用した「**既存機能**」に分けることができます。こうした場合、ソフトウェアテストを各機能に対して均等に工数を割り当てるのではなく、ソフトウェアの変更のない既存機能についてのテストは思い切って工数を減らし、新規機能のテストに工数を割り当てることになります。これを組み合わせテストに当てはめれば、新規機能については**要因組み合わせテスト**を多めに行うが、既存機能についてはテストケース数が多くなりがちな要因組み合わせテストではなく、組み合わせを行わない**要因列挙テスト**で済ますということになります。

要因列挙テストのテストケースを作成するには環境設定で**「組み合わせるパラメータ数」に１を設定**して生成を行なうだけです。

パラメータが５つ、各パラメータあたり５つの値を持つモデルでの要因列挙のテストケースの例を以下に示します。

表６－１３　要因列挙のテストケースの例

No.	A	B	C	D	E
1	a1	b2	c1	d3	e1
2	a2	b1	c5	d2	e4
3	a3	b3	c4	d4	e2
4	a4	b5	c3	d5	e3
5	a5	b4	c2	d1	e5

全てを要因列挙にする必要がない場合もあります。それは最も多くの数の値を持つパラメータとそれ以外のパラメータの持つ値の数に大きな開きがある場合です。

パラメータが５つ、８個の値を持つパラメータが２つ、残りは３個のパラメータを持つモデルを例として説明します。（**図６－１５**）

パラメータ	値の並び
A	a1, a2, a3, a4, a5, a6, a7, a8
B	b1, b2, b3, b4, b5, b6, b7, b8
C	c1, c2, c3
D	d1, d2, d3
E	e1, e2, e3

図６－１５　要因組み合わせと要因列挙を合成するのに適したモデル

このモデルのテストケース数は要因列挙テストでは８件となります。この場合、３個の値をもつパラメータが３つありますが、このパラメータについては、要因組み合わせテストとしてもテストケース数は３×３＝９件となり、８件とほとんど変わりありません。

ここで１つのテストケースで要因組み合わせと要因列挙を合成する方法を説明します。

環境設定で**「組み合わせるパラメータ数」に１を設定**します。**「サブモデルを使用する」**にチェックを入れます。サブモデルの記入欄に、要因組み合わせを行なうパラメータを記入し、**組み合わせ数に２を記入**します。（**図６－１６**）

図６－１６　サブモデルで要因組み合わせを行なうパラメータを指定する。

このモデルでの生成結果を以下に示します。このテストケースではパラメータＡとＢが要因列挙であり、パラメータＣ、ＤおよびＥが２パラメータ間の要因組み合わせとなっています。

表６－１４　要因組み合わせと要因列挙を合成したテストケース

No.	A	B	C	D	E
1	a1	b3	c1	d2	e1
2	a2	b7	c3	d2	e3
3	a3	b5	c2	d3	e1
4	a3	b7	c1	d1	e2
5	a4	b1	c2	d2	e2
6	a5	b8	c3	d1	e1
7	a6	b4	c2	d1	e3
8	a7	b6	c1	d3	e3
9	a8	b2	c3	d3	e2

このように多くの値を持つパラメータだけ選択して要因列挙とし、それ以外のパラメータを要因組合せとすることで、テストケース数をそれほど増加させることなく、要因組み合わせを混在させた要因列挙のテストケースを生成することができます。
この例では要因列挙テストとするパラメータが２つでしたが、３つ以上とした場合でも各パラメータの組み合わせはランダムとなり、組み合わせが１対１となるようなことはありません。

要因組み合わせと要因列挙を混在したテストケースはそれほど多くはないと思いますが、要因列挙のみのテストケースはよく使われることになると思います。要因列挙であればツールを使わなくてもいいではないかと考える方がいるかもしれませんが、制約がある場合はツールを使ったほうが間違いのないものになりますし、組み合わせがランダムとなるのでよいでしょう。

要因列挙テストは、要因の組み合わせによるソフトウェアの欠陥はないだろうということを前提としています。既存機能の流用であれば、要因を組み合わせてまでの徹底したテストは不要だとすることでテスト工数を大きく削減してよいと考えます。

７．デシジョンテーブルテストへの適用

デシジョンテーブルは、テスト対象が複雑な論理を含む場合、優先度が関係する場合などに最適なテスト技法です。PictMaster はデシジョンテーブルのテストケースを生成することができます。

それではデシジョンテーブルの問題として一般に公開されているJaSST'07 Tokyo での「三賢者、テストを語る　(DTvsCEGvsCFD)」 の「入場料問題」（28 ページ目）を使うことにします。この問題を以下に示します。


テーマ2　入場料問題

- 問題
  - ある遊園地の入場料に関する仕様です。

	個人	団体
一般	1200円	1000円
小学生	600円	500円
ただし六歳未満と六十五歳以上、 及び県内在住の小学生は無料		

図７－１　デシジョンテーブルを作成する入場料問題

デシジョンテーブルに PictMaster を適用する際は、デシジョンテーブルの各条件がパラメータとなり、条件のとりうる値（Yes か No かなど）がそのままパラメータの値となります。そして**組み合わせるパラメータ数を実際のパラメータ数と同じに設定します**。これはデシジョンテーブルがパラメータ数のレベルを持つ木構造となっており、そのすべての組み合わせを生成する必要があるためです。その意味では全数組み合わせテストと同じです。

７．１　デシジョンテーブルのモデルの作成

図７－１の問題では、個人と団体は排他の関係にあるため、パラメータは個人のみとし、その値の Yes、No で個人と団体を表します。その他に必要なパラメータは、一般、小学生、６歳未満、６５歳以上、県内在住、で全体では６個の パラメータとなります。いずれも値は Yes と No です。この問題で６歳未満、小学生、一般、６５歳以上は、それぞれ排他の関係にあるパラメータであり、同時には Yes とはなりません。

この問題を解くモデル（パラメータと値の並び、制約表および結果表）を**図７－２**に示します。

パラメータ	値の並び
６歳未満	Yes，No
小学生	Yes，No
一般	Yes，No
６５歳以上	Yes，No
県内在住	Yes，No
個人	Yes，No

制約表

パラメータ	制約1	制約2	制約3	制約4	制約5	制約6
６歳未満	Yes	No	No	No	Yes	No
小学生	No	Yes	No	No	No	No
一般	No	No	Yes	No	No	No
６５歳以上	No	No	No	Yes	No	Yes
県内在住						
個人						

結果表

結果内容	6歳未満	小学生	一般	65歳以上	県内在住	個人
１２００円			Yes			Yes
１０００円			Yes			No
６００円		Yes			No	Yes
５００円		Yes			No	No
無料				Yes		
無料	Yes					
無料		Yes			Yes	

図７－２　入場料問題のモデル

パラメータの県内在住と個人（団体）は、組み合わせに制約を持ちません。それ以外の６歳未満、小学生、一般、６５歳以上は、それぞれ排他の関係となるよう、**制約１～４**で指定しています。**制約５と６**で、小学生でもなく一般でもない場合は、６歳未満か６５歳以上のいずれかであることを指定しています（制約対象が OR 指定となります)。

結果表を用いてパラメータの値の組み合わせに応じた入場料を自動的に設定するようにしています。デシジョンテーブルでは必ず結果表を使用することになります。

こうして生成した組み合わせを**表７－１**に示します。

表７－１　生成された組み合わせ

No.	6歳未満	小学生	一般	65歳以上	県内在住	個人	結果内容
1	No	No	No	Yes	Yes	Yes	無料
2	No	No	No	Yes	Yes	No	無料
3	No	No	No	Yes	No	Yes	無料
4	No	No	No	Yes	No	No	無料
5	No	No	Yes	No	Yes	No	1000円
6	No	No	Yes	No	Yes	Yes	1200円
7	No	No	Yes	No	No	No	1000円
8	No	No	Yes	No	No	Yes	1200円
9	No	Yes	No	No	No	Yes	600円
10	No	Yes	No	No	Yes	No	無料
11	No	Yes	No	No	Yes	Yes	無料
12	No	Yes	No	No	No	No	500円
13	Yes	No	No	No	No	No	無料
14	Yes	No	No	No	No	Yes	無料
15	Yes	No	No	No	Yes	Yes	無料
16	Yes	No	No	No	Yes	No	無料

生成された組み合わせは**１６通り**となりました。**アクション（入場料）を結果表で決定しています**。徹底したテストが必要な場合は、この組み合わせでテストを行なうことになります。それほど徹底したテストが必要でない場合は、１６通りの組み合わせの中からアクション（入場料）に影響を与えない組み合わせを削除し、テストケース数を削減することになります。

７．２　テーブルを圧縮する

「**整形**」ボタンで生成結果の「**結果内容**」を第一優先のパラメータとして並べ替えを行います。こうすることで組み合わせが入場料ごとに並び、入場料に影響を与えない組み合わせを見つけやすくなります。

入場料に影響を与えない組み合わせ（入場料が同じで組み合わせに共通点がある組み合わせ）をそれぞれ異なる色で塗りつぶした組み合わせを**表７－２**に示します。

表７－２アクション（入場料）に影響を与えない組み合わせ

No.	6歳未満	小学生	一般	65歳以上	県内在住	個人	結果内容
5	No	No	Yes	No	Yes	No	1000円
7	No	No	Yes	No	No	No	1000円
6	No	No	Yes	No	Yes	Yes	1200円
8	No	No	Yes	No	No	Yes	1200円
12	No	Yes	No	No	No	No	500円
9	No	Yes	No	No	No	Yes	600円
1	No	No	No	Yes	Yes	Yes	無料
2	No	No	No	Yes	Yes	No	無料
3	No	No	No	Yes	No	Yes	無料
4	No	No	No	Yes	No	No	無料
10	No	Yes	No	No	Yes	No	無料
11	No	Yes	No	No	Yes	Yes	無料
13	Yes	No	No	No	No	No	無料
14	Yes	No	No	No	No	Yes	無料
15	Yes	No	No	No	Yes	Yes	無料
16	Yes	No	No	No	Yes	No	無料

異なる色で塗りつぶした組み合わせについて、各色に１つを残してあとは削除します。この際、入場料に影響を与えないパラメータについてはそうであることを示す意味で「－」を記入します。このパラメータの値は Yes でも No でも結果に影響を与えません。

最終的に決定したデシジョンテーブルを以下に示します。

表７－３　テストケースを削減したデシジョンテーブル

No.	6歳未満	小学生	一般	65歳以上	県内在住	個人	結果内容
1	No	No	Yes	No	－	No	1000円
2	No	No	Yes	No	－	Yes	1200円
3	No	Yes	No	No	No	No	500円
4	No	Yes	No	No	No	Yes	600円
5	No	No	No	Yes	－	－	無料
6	No	Yes	No	No	Yes	－	無料
7	Yes	No	No	No	－	－	無料

この組み合わせは JaSST'07 Tokyo での「三賢者、テストを語る　(DTvsCEGvsCFD)」の 34 ページ目の組み合わせと**実質的に同じ組み合わせ**です。

従来の方法では、テスト対象からいきなりデシジョンテーブル作成となり、デシジョンテーブル作成の過程が明らかではありませんが、PictMaster を使用することで**「モデル」の形でデシジョンテーブル作成の根拠を明らかな形で残す**ことができます。

８．状態遷移テストへの適用

テスト対象が状態を持ち、イベントにより状態の遷移が起こる場合、状態遷移テストが有効なテスト技法となります。PictMaster は状態遷移テストのテストケースを生成することができます。状態遷移テストでは１回の状態遷移では正常に動作するが、状態遷移を繰り繰り返すと発生するバグが少なからずあります。状態遷移のくり返しを含むテストケースを生成する方法を説明します。

８．１　状態遷移のくり返しを含むテストケースの生成

パラメータには各状態の名称を採用し、パラメータの値には、その状態で有効なイベントの名称を採用します。なお、状態遷移の過程で通らない状態が発生するので、そのことを表すダミーの値「－」も定義することにします。

例として**図８－１**の状態遷移図を用いることにします。この図で円形の s0 から s4 は状態を表し、状態から他の状態に接続する矢印は状態遷移の方向を表し、矢印に併記されている e1～e9 はその状態で有効なイベントを表します。

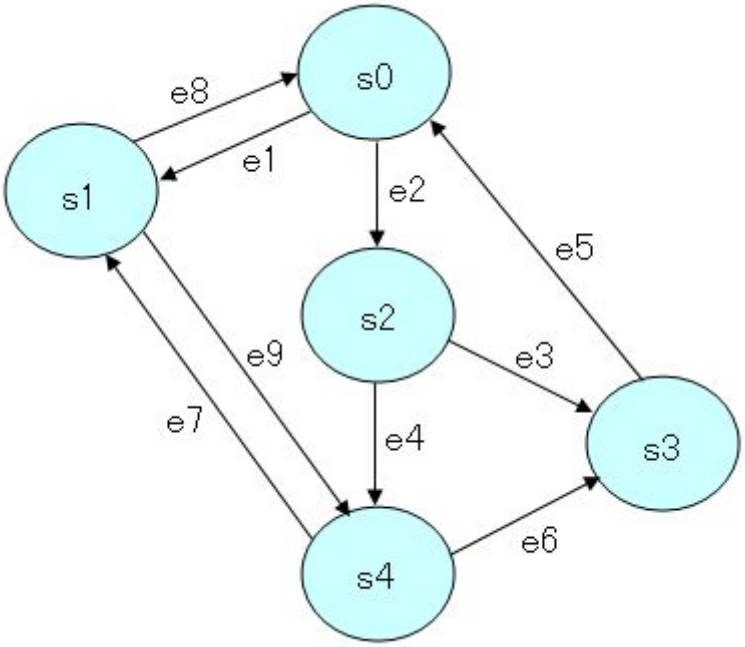

図８－１　状態遷移図の例

状態遷移は s0 から始まるものとします。この状態遷移図ですべての状態の遷移を網羅し、かつ状態遷移のくり返しも網羅するために、パラメータとして状態 s0～s4 と s0’～s4’そして s0''を定義します。こうすることですべての状態を通る状態遷移のルートのセットと同じ状態を複数回通るルートのセットが得られます。

Pairwise 法（All-Pairs 法）によるテストケース生成では、２つの状態の組み合わせを作成するため、パラメータとして最低でも同じ状態が２回現れるようにパラメータを指定する必要があります。同じ状態が１回しか現れない場合、生成されるテストケースは不完全なものとなります。

８．２　状態遷移テストのモデルの作成

図８－１の状態遷移図をもとにした状態遷移テストのモデルを**図８－２**に示します。

パラメータ	値の並び
s0	e1, e2
s1	e8, e9, -
s2	e3, e4, -
s3	e5, -
s4	e6, e7, -
s0'	e1, e2, -
s1'	e8, e9, -
s2'	e3, e4, -
s3'	e5, -
s4'	end, -
s0''	end, -

制約表

パラメータ	制約1	制約2	制約3	制約4	制約5	制約6
s0	e1	e2				
s1	e8, e9	-	e8	e9		
s2		e3, e4	-	-	e3	e4
s3			-	-	e5	-
s4			-	e6, e7		e6, e7
s0'			e1, e2			
s1'						
s2'						
s3'						
s4'						
s0''						

図８－２　状態遷移テストのモデル（１／２）

制約表への記入は、ある状態で有効なイベントを制約条件として記入し、イベントごとに遷移先の状態の欄に、その状態で有効なイベントを制約対象として記入します。**遷移先の状態が他の状態の行をまたぐ場合は、その状態との組み合わせがないことを表すダミー値の「－」をその状態の欄に記入**します。

制約対象として記入した１つ以上のイベントは、新たな制約欄に制約条件として１つずつ記入します。後はこれのくり返しです。

制約表

パラメータ	制約7	制約8	制約9	制約10	制約11	制約12
s0						
s1						
s2						
s3	e5					
s4	-	e6	e7			
s0'	e1, e2	-	-	e1	e2	
s1'		-	e8, e9	e8, e9	-	e8
s2'		-			e3, e4	-
s3'		e5				-
s4'						-
s0''						end

制約表

パラメータ	制約13	制約14	制約15	制約16	制約17	制約18
s0						
s1						
s2						
s3						
s4						
s0'						
s1'	e9					
s2'	-	e3	e4			
s3'	-	e5	-	e5		
s4'	end		end	-		
s0''	-		-	end		

図８－２　状態遷移テストのモデル（２／２）

このモデルでは制約が１６個となりました。制約の数は状態の数とイベントの数で大きく異なってきます。状態遷移図の制約表では、ある状態であるイベントがどの状態へ遷移できるのかをすべて記述します。このことは Pairwaise 法（All-Pairs 法）で最少テストケース生成を行なっても、**かならず一定のテストケ**

ース数となることを意味しています。

８．３　状態遷移テストの組み合わせ結果

図８－２のモデルで生成した組み合わせ結果を**表８**に示します。

表８．状態遷移テストの組み合わせ結果

No.	s0	s1	s2	s3	s4	s0'	s1'	s2'	s3'	s4'	s0''
1	e1	e8	−	−	−	e1	e9	−	−	end	−
2	e1	e8	−	−	−	e1	e8	−	−	−	end
3	e1	e8	−	−	−	e2	−	e3	e5	−	end
4	e1	e8	−	−	−	e2	−	e4	−	end	−
5	e1	e9	−	−	e7	−	e8	−	−	−	end
6	e1	e9	−	−	e6	−	−	−	e5	−	end
7	e1	e9	−	−	e7	−	e9	−	−	end	−
8	e2	−	e3	e5	−	e1	e8	−	−	−	end
9	e2	−	e3	e5	−	e1	e9	−	−	end	−
10	e2	−	e3	e5	−	e2	−	e4	−	end	−
11	e2	−	e3	e5	−	e2	−	e3	e5	−	end
12	e2	−	e4	−	e7	−	e8	−	−	−	end
13	e2	−	e4	−	e7	−	e9	−	−	end	−
14	e2	−	e4	−	e6	−	−	−	e5	−	end

この例では組み合わせ数は１４個となりました。PictMaster の最少テストケース生成を行なうと、最大数と最少数はともに１４個となります。このことは、組み合わせ条件が制約表の記述によって完全に決定されていることを意味しています。逆にいえば、最少テストケース生成で最大数と最少数が異なる場合は、制約表の記述に不足した点があることになります。特に状態遷移の最後の状態で、後に続く状態がある場合は、その状態の欄にダミー値の「－」を記入することを忘れないようにしましょう。**図８－２**でいえば制約１３と制約１５がそれにあたります。

表８の組み合わせ結果では、最少で３つの状態、最大で５つの状態を経由する最小限の組み合わせとなっています。ダッシュ（'）のついた状態は、ついていない状態と同じ状態とみなします。そうすると同じ状態を２回通る組み合わせが数多く網羅されていることが分かります。

PictMaster による状態遷移テストでは数多くの制約を指定することになります。組み合わせ生成エンジン PICT の処理能力により、あまり多くの状態を含む組み合わせの生成には非常に長い時間がかかります。**実用的にはパラメータ数が２０未満に収まるように**したほうが良いでしょう。時間がかかる場合は、最少テストケース生成ではなく、デフォルトの条件による１回だけの生成にしましょう。

対象とする状態遷移図がこの制限を超えて大きい場合は、可能であれば大きな状態遷移図をいくつかのサブの状態遷移図に分けて扱うほうが良いかもしれません。

生成した組み合わせ結果には状態遷移のルートしか記述されていません。実際のテストケース仕様書では、**表８**の各列の間に、その状態遷移の結果として起きることの確認内容を記述した列を挿入して完成となります。

９．制御パステストへの適用

制御パステストはホワイトボックステストのテスト技法で、単体テストレベルで使用します。プログラムから一連のテストパスを選んでそのパスを実行し、処理結果を確認するテスト技法です。

９．１　従来の制御パステストとの違い

従来の制御パステストの方法は、まずプログラムの条件文以外を省略して制御構造のみを表した制御フローグラフを作成します。次にプログラムのソースリストを何部かコピーしたものを用意し、ソースリスト上に１つずつ制御パスを記入していきます。その記入内容を制御フローグラフにプロットします。この作業を繰り返し、制御フローグラフ上のすべての条件分岐を網羅するまで続けます。すべてを網羅した時点で、何部かのソースリスト上に書き込まれた個々の制御パスの集まりがテストケースとなります。こうした作業はかなり手間のかかる作業です。**PictMaster は制御フローグラフを作成することなく、直接ソースリストからモデルを作成し、制御パステストのテストケースを生成することができます。**

パスの網羅基準には大きく分けて、命令網羅、条件網羅、経路網羅の３つがあります。PictMaster は条件網羅と経路網羅に対応しています。通常の条件網羅では、すべての条件文の真と偽の両方を通るパスを網羅します。これに対して Pairwise 法（All-Pairs 法）では、すべての２つの条件文との組み合わせにおいて真と偽の両方を通るパスを網羅します。PictMaster の組み合わせ生成エンジン PICT は K-way テスト（組み合わせるパラメータ数が３個以上）に対応しています。それゆえ組み合わせるパラメータ数をすべての条件文の合計数とすることで経路網羅のテストケースも生成することができます。

PictMaster を使用することで制御パステストの網羅基準は以下のようになります。ここで括弧内の Pn などは組み合わせる条件文の数を表しています。Pmax はすべての条件文の合計数を意味します。

１．命令網羅（P0）
２．通常の条件網羅（P1）
３．Pairwise 法による条件網羅（P2）
４．ｎ個の条件文の組み合わせによる条件網羅（Pn）
５．経路網羅（Pmax）

以上の網羅基準によれば、PictMaster を使用することで P1～Pmax までの制御パステストを実施することができます。

複数の条件文の組み合わせを網羅することで通常の条件網羅では発見できないバグを発見することができます。例えば**リスト９－１**のプログラムがあるとします。左端の番号は行番号です。

リスト９－１　誤りのあるプログラム

```
 1    if x1 = 3 then
 2            y = 1
 3    else
 4            y = 0  …  間違った代入文
 5    end if
 6    if x2 = 1 then
 7            z = y
 8    else
 9            z = x3/y  …  ４行目を通ってくると０除算となる代入文
10    end if
```

パスが４行目と９行目を通る場合のみ、０除算のバグが発見できます。

このプログラムの通常の条件網羅のテストケースは**表９－１**となる可能性があります。

表９－１　通常の条件網羅のテストケースの例

No.	1if	2s1	4s2	6if	7s3	9s4
1	False	−	y=0	True	z=y	−
2	True	y=1	−	False	−	z=x/y

通常の条件網羅ではテストケース数が２となります。パラメータの名称には分かりやすいように行番号が追加されています。この例では０除算のバグが発生する４行目と９行目を通るパス（２つの条件文がともに偽の場合）が網羅されていません。

次に Pairwise 法による条件網羅のテストケースの例を**表９－２**に示します。

表９－２　Pairwise 法による条件網羅のテストケースの例

No.	1if	2s1	4s2	6if	7s3	9s4
1	False	−	y=0	True	z=y	−
2	False	−	y=0	False	−	z=x/y
3	True	y=1	−	False	−	z=x/y
4	True	y=1	−	True	z=y	−

Pairwise 法による条件網羅ではテストケース数が４となりました。この例では No.1 で０除算のバグが発生する４行目と９行目を通るパスが網羅されています。このように**２つの条件文の組み合わせで発生するバグを Pairwise 法による条件網羅で確実に発見することができます**。

この例では条件文だけでなく代入文もパラメータに採用していますが、基本的には条件文だけをパラメータにすれば十分です。テストケース数が通常の条件網羅の２に対して４と倍増していますが、実際のプログラム全体を対象としたテストケース数では、Pairwise 法による条件網羅は通常の条件網羅に対して約 50%前後の増加にとどまります。

９．２　パラメータと値を決めるには

パラメータはパスの流れを変えるすべての文が対象となります。具体的にはＣ言語でいえば、
if else switch case for do while break continue プログラム途中での **return** などです。言語によっては、**end if** などのようにパスの合流を意味する文がありますが、こうした文は基本的にはパラメータにする必要がありません。なぜなら次に続く条件文でパスの合流が表現されるからです。

パラメータの値は、条件文を含む文の場合は基本的には **TRUE と FALSE** と、その条件文を通らないパスを表現するためのダミーの値「-」を定義すれば十分です。条件文を含まない場合は、任意の名称の値とダミーの値を定義します。

例として**リスト９－２**のプログラムで説明します。このプログラムのモデルを**図９－１**に示します。このプログラムでは制御パスを変える文が２つあります。

リスト９－２　条件文が２つ続くプログラム

```
1   if (a == 1) {  …  パスを変える文
2           X = 1;
3   }
4   if (b == 2) {  …  パスを変える文
5           y = 2;
6   } else {
7           y = 3;
8   }
```

パラメータには制御パスを変える文である１行目と４行目を採用します。このモデルでは分かりやすさのために６行目の else もパラメータに含めています。パラメータが条件文の場合は、真と偽が値になります。条件文以外のパラメータの場合、値の名称は分かりやすい任意の名称とします。

パラメータ	値の並び
1if	TRUE, FALSE
4if	TRUE, FALSE
5else	else, -

制約表

パラメータ	制約1	制約2	制約3	制約4
1if	TRUE	FALSE		
4if	TRUE, FALSE	TRUE, FALSE	TRUE	FALSE
5else			-	else

図９－１　プログラムのモデルの例

このモデルでは、分かりやすいように else 節をパラメータに含めています。**リスト９－２**のように else 節の中にパスを変える文がなければ、else 節は省略してもかまいません。

制約表への記入は、制約対象（TRUE と FALSE）を次の制約欄でそれぞれ独立した制約条件として記入します。

このモデルで生成された Pairwaise 法（All-Pairs 法）によるテストケースを**表９－３**に示します。

表９－３　２つの if 文によるテストケース

No.	1if	4if	5else
1	False	False	else
2	False	True	-
3	True	False	else
4	True	True	-

実際のテストでは、この４つのパスを通る４つのテストデータのセットを用意します。この場合のテストデータのセットは以下の通りとなります。これは１つの例です。

(1) a = 2, b = 1
(2) a = 2, b = 2
(3) a = 1, b = 1
(4) a = 1, b = 2

９．３　前の条件文によって次の条件文のパスが影響を受ける場合の値の決め方

条件文が２つ以上連結しているプログラムでは、前の条件文によって次の条件文のパスが影響を受ける場合があります。これは前の条件文で分かれたパスで同じ変数に異なる値を代入しており、次の条件文でその変数を判定対象としている場合などが該当します。こうした条件文を**相互作用のある条件文**と言います。

相互作用のある条件文の例を**リスト９－３**に示します。

リスト９－３　相互作用のある条件文のプログラム

```
 1  if (a == 1) {
 2          a = a + 3;
 3  } else {
 4          b = b + 1;
 5  }
 6  if (b == 2) {
 7          b = b + 2;
 8  } else {
 9          a = a + 1;
10  }
11  if (a > 3) {
12          x = 1;
13  } else {
14          x = 0;
15  }
```

このプログラムのモデル（**図９－２**）と生成されたテストケース（**表９－４**）を示します。ここでは分かりやすいように代入文もパラメータに含めています。

パラメータ	値の並び
1if	a == 1, a <> 1
2s1	a = a + 3, -
3s2	b = b + 1, -
6if	b == 2, b <> 2
7s3	b = b + 2, -
9s4	a = a + 1, -
11if	a > 3, a <= 3
12s5	x = 1, -
14s6	x = 0, -

制約表

パラメータ	制約1	制約2	制約3	制約4	制約5	制約6
1if	a == 1	a <> 1				
2s1	a = a + 3	-				
3s2	-	b = b + 1				
6if	b == 2, b <> 2	b == 2, b <> 2	b == 2	b <> 2		
7s3			b = b + 2	-		
9s4			-	a = a + 1		
11if			a > 3, a <= 3	a > 3, a <= 3	a > 3	a <= 3
12s5					x = 1	-
14s6					-	x = 0

図９－２　相互作用のある条件文のモデル

表９－４　相互作用のある条件文のテストケース

No.	1if	2si	3s2	6if	7s3	9s4	11if	12s5	14s6	a, b
1	a <> 1	−	b = b + 1	b <> 2	−	a = a + 1	a > 3	x = 1	−	a=3, b=0
2	a <> 1	−	b = b + 1	b == 2	b = b + 2	−	a <= 3	−	x = 0	a=0, b=1
3	a == 1	a = a + 3	−	b <> 2	−	a = a + 1	a <= 3	−	x = 0	a=?, b=1
4	a == 1	a = a + 3	−	b == 2	b = b + 2	−	a > 3	x = 1	−	a=1, b=2

このテストケースの右端の列は、対応するテストケースの制御パスを通すための変数 a と b の値のセットです。このテストケースでは No.3 で変数 a の値が１１行目の if 文と矛盾しており決定できません。こうした場合は新たに制約を追加して、すべてのテストケースの制御パスが通れる変数 a と b の値が決定できるようにします。新しく追加した制約（**図９－３**）と生成されたテストケース（**表９－５**）を示します。

制約表

パラメータ	制約7
1if	a == 1
2s1	
3s2	
6if	
7s3	
9s4	
11if	a > 3
12s5	
14s6	

図９－３　通れるパスを生成するために追加した制約

表９－５　すべてのテストケースが通れるように修正されたテストケース

No.	1if	2si	3s2	6if	7s3	9s4	11if	12s5	14s6	a, b
1	a <> 1	−	b = b + 1	b <> 2	−	a = a + 1	a > 3	x = 1	−	a=3, b=0
2	a <> 1	−	b = b + 1	b <> 2	−	a = a + 1	a <= 3	−	x = 0	a=0, b=0
3	a <> 1	−	b = b + 1	b == 2	b = b + 2	−	a <= 3	−	x = 0	a=2, b=1
4	a == 1	a = a + 3	−	b == 2	b = b + 2	−	a > 3	x = 1	−	a=1, b=2
5	a == 1	a = a + 3	−	b <> 2	−	a = a + 1	a > 3	x = 1	−	a=1, b=0

追加した制約で、変数 a が１のときは１１行目の if 文で、a > 3 とのみ組み合わせ可能としています。

相互作用のある if 文で通れない矛盾したパスが生成された場合は、このような方法で新しい制約をすることによって矛盾のないテストケースを生成することができます。

９．４　１つの条件文が複数の条件式を含む場合の対処法

１つの条件文の中で２つ以上の条件式を含む場合があります。例えば**リスト９－４**の場合です。

リスト９－４　複数の比較判定を行なう条件文

```
1    if (a > 0 && c == 1) {
2            x = x + 1;
3    }
4    if (b == 3 || d < 0) {
5            y = 0;
6    }
```

この例では、１行目と４行目の if 文でそれぞれ２つの条件式を含んでいます。１行目と４行目の個々の２つの条件式をまとめて１つずつの条件式として扱うこともできますが、こうした場合は個々の条件式がすべて評価される訳ではないため、どちらかの条件式にバグがあっても発見できない可能性があります。それはコンパイラが条件式をどのような順番で評価するかに依存する事項です。これに対して、個々の条件式をすべてパラメータの値として扱うことでバグの見逃しを防ぐことができます。こうした１つの条件文が複数の条件式から成り立っている場合に、個々の条件式まで網羅することを、**複合コンディションカバレッジ**を満たしていると言います。

リスト９－４のプログラムの、条件文の２つの条件式を独立の if 文としたフローチャートを**図９－４**に示します。

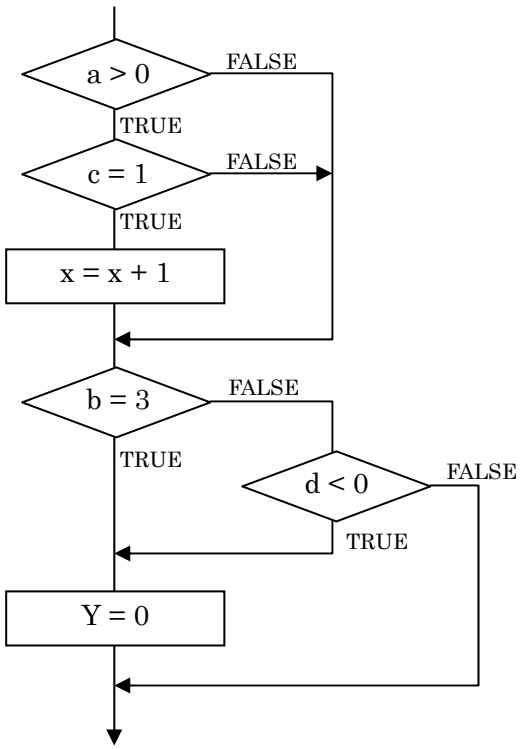

図９－４　複数の条件式を含む条件文を詳細化したフローチャート

リスト９－４の複合コンディションカバレッジを満たすモデルを図９－５に、そのテストケースを表９－６に示します。このモデルでは分かりやすいように代入文も含めています。１つの if 文が２つのパラメータで構成されていることに注意してください。この例の場合は、コンパイラが if 文の左側から条件式を評価するものとしています。

パラメータ	値の並び
1if	a > 0, a <= 0
1if'	c == 1, c <> 1, -
2x	x = x + 1, -
4if	b == 3, b <> 3
4if'	d < 0, d >= 0, -
5y	y = 0, -

制約表

パラメータ	制約1	制約2	制約3	制約4
1if	a > 0	a <= 0		
1if'	c == 1, c <> 1	-	c == 1	c <> 1
2x		-	x = x + 1	-
4if		b == 3, b <> 3	b == 3, b <> 3	b == 3, b <> 3
4if'				
5y				

制約表

パラメータ	制約5	制約6	制約7	制約8
1if				
1if'				
2x				
4if	b == 3	b <> 3		
4if'	-	d < 0, d >= 0	d < 0	d >= 0
5y	y = 0		y = 0	-

図９－５　複合コンディションカバレッジを満たすモデルの例

表９－６　複合コンディションカバレッジのテストケースの例（**Pairwaise 法による条件網羅**）

No.	1if	1if'	2x	4if	4if'	5y
1	a <= 0	−	−	b <> 3	d >= 0	−
2	a <= 0	−	−	b == 3	−	y = 0
3	a <= 0	−	−	b <> 3	d < 0	y = 0
4	a > 0	c <> 1	−	b == 3	−	y = 0
5	a > 0	c <> 1	−	b <> 3	d < 0	y = 0
6	a > 0	c <> 1	−	b <> 3	d >= 0	−
7	a > 0	c == 1	x = x + 1	b <> 3	d >= 0	−
8	a > 0	c == 1	x = x + 1	b == 3	−	y = 0
9	a > 0	c == 1	x = x + 1	b <> 3	d < 0	y = 0

テストケースは９個となりました。これは**図９－２**のプログラムを対象としていることによります。これに対して、Pairwise 法によらない通常の条件網羅のテストケースは**表９－７**となります。

表９－７　複合コンディションカバレッジのテストケースの例（**通常の条件網羅**）

No.	1if	1if'	2x	4if	4if'	5y
1	a <= 0	−	−	b == 3	−	y = 0
2	a > 0	c <> 1	−	b <> 3	d < 0	y = 0
3	a > 0	c == 1	x = x + 1	b <> 3	d >= 0	−

通常の条件網羅では複数の条件の組み合わせは網羅しないため、テストケース数は３個となります。

９．５　プログラムがループを含む場合の記入方法

プログラムがループを含んでいる場合でもモデルを記述することができます。例として文字列 aStrings をもう１つの文字列 aText でサーチし、一致した最初の文字の桁位置を返す関数を取り上げてみます。一致しなかった場合は -1 が返されます。ソースリストは以下の通りです。VBA で書かれています。

この関数で　Len(文字列) は文字列の長さを意味し、Mid(文字列, 桁数１, 桁数２) は文字列の先頭から桁数１の位置から桁数２の長さの文字を意味します。

リスト９－５　ループを含むプログラム（文字列のサーチを行なう関数）

```
 1 Function StringSearch(aStrings As String, aText As String) As Integer
 2
 3     Dim i As Integer, j As Integer
 4
 5     If Len(aStrings) < Len(aText) Then
 6         StringSearch = -1
 7         Exit Function
 8     End If
 9     For i = 0 To Len(aStrings) - Len(aText)
10         j = 1
11         Do
12             If Mid(aStrings, i + j, 1) <> Mid(aText, j, 1) Then
13                 Exit Do
14             End If
15             j = j + 1
16             If j = Len(aText) + 1 Then
17                 StringSearch = i + 1
18                 Exit Function
19             End If
20         Loop
21     Next i
22     StringSearch = -1
23
24 End Function
```

リスト９－５を見ると、この関数は２つのループが入れ子になった構造となっています。
この関数の制御パステストのテストケースを生成するモデルを図９－６に示します。

パラメータ	値の並び
5if	TRUE, FALSE
7Exit	Exit F, -
9For	TRUE, FALSE, -
11Do	Do, -
12if	TRUE, FALSE, -
13Exit	Exit Do, -
16if	TRUE, FALSE, -
18Exit	Exit F, -
20Loop	Loop, -
21Next	Next, -
24End	End, -

制約表

パラメータ	制約1	制約2	制約3	制約4
5if	TRUE	FALSE		
7Exit	Exit F	-		
9For	-	TRUE, FALSE	TRUE	FALSE
11Do	-		Do	-
12if	-		TRUE, FALSE	-
13Exit	-			-
16if	-			-
18Exit	-			-
20Loop	-			-
21Next	-			-
24End	-			End

制約表

パラメータ	制約5	制約6	制約7	制約8
5if				
7Exit				
9For				
11Do				
12if	TRUE	FALSE		
13Exit	Exit Do	-		
16if	-	TRUE, FALSE	TRUE	FALSE
18Exit	-		Exit F	-
20Loop	-		-	Loop
21Next	Next		-	-
24End	-		-	-

図９－６　ループを含むプログラムのモデル（その１）

このモデルでは代入文を含んでおらず、制御パスに影響を与える条件文などのみから成り立っています。
生成したテストケースを表９－８に示します。

表９－８　ループを含むプログラムのテストケース

No.	5if	7Exit	9For	11Do	12if	13Exit	16if	18Exit	20Loop	21Next	24End	Data #
1	FALSE	-	FALSE	-	-	-	-	-	-	-	End	*2
2	FALSE	-	TRUE	Do	TRUE	Exit Do	-	-	-	Next	-	*2, *3, *5
3	FALSE	-	TRUE	Do	FALSE	-	TRUE	Exit F	-	-	-	*3, *4, *5
4	FALSE	-	TRUE	Do	FALSE	-	FALSE	-	Loop	-	-	*2, *3
5	TRUE	Exit F	-	-	-	-	-	-	-	-	-	*1

右端の Data# は、関数の入力パラメータ（テストデータ）の識別番号であり、その内容は以下の通りです。

表９－９　テストデータと関数の戻り値（Value）

Data #	aStrings	aText	Value
*1	a	ab	-1
*2	abc	bcd	-1
*3	ababcab	abc	3
*4	a	a	1
*5	abc	c	3

表９－８から、すべての制御パスを網羅していることが分かります。ここでわかることは、**１つのテストデータで２つ以上の制御パスを通る場合がある**ということです。プログラムがループを含む場合、**生成されたテストケースは、ループの途中までの制御パスを含みます**。そのため、１つのテストデータで２つ以上の制御パスを通ることになります。もう１つ注意が必要なのは、Data # の ***4** です。このテストデータは他のテストデータも通る制御パスを通っていますが、No.3 の制御パスしか通りません。こうしたテストデータをもれなくテストする必要があります。

今回のプログラムでは、組み合わせるパラメータ数が１、２、すべて、のいずれの場合でも制御パスは５で同じ数です。条件網羅と経路網羅で制御パスの数が違ってくるのは、**条件文が直列に２つ以上接続されている場合**です。それも条件文の真と偽の双方で１つのルートに合流する場合です。どちらかがパスから外れる場合は除かれます。その意味で今回のプログラムには２つ以上の条件文が完全に直列になっている箇所がないため、条件網羅と経路網羅で同じ制御パスとなります。

このプログラムの性質上、ループについては、ループする、ループから抜ける、の２つしかありませんが、プログラムによっては境界値分析・同値分割の考え方をループに適用し、１回もループしない、１回だけループする、最大回数までループする、の３つのケースをカバーする方がよい場合もあります。

もう１つ、ループを含むプログラム例を示します。このプログラムは、Excel のワークシート上にある数値を比較します。プログラムの仕様は以下のとおりです。

１．Ａ列のセルにあるカンマで区切られた数値の並びをチェックし、最小値をＢ列に、最大値をＣ列に設定する。
２．数値の並びのチェックは、空きの行が見つかるまで繰り返す。
３．最後の行までチェックしたら、Ｂ列とＣ列の中でＢ列の最小値をＤ列に、Ｃ列の最大値をＥ列に設定する。
４．数値の最大値は９９とし、それ以上の値は９９とみなす。最小値は０とし、マイナスの数値はワークシート上にないものとする。
５．Ａ列１行目が空きのときは、“データがありません”と表示して処理を中止する。

このプログラムのソースリストを**リスト９－６**に示します。

リスト９－６　ループを含むプログラム（ワークシート上の最小値、最大値を求めるサブルーチン）

```
Sub CompCell()

    Dim i As Long, j As Long, k As Long
    Dim arry As Variant, n As Long
    Dim wkMax As Long, wkMin As Long, wk As Long

    If Cells(1, 1) = "" Then
        MsgBox "データがありません。"
        End
    End If
    i = 1
    Do
        arry = Split(Cells(i, 1), ",")
        n = UBound(arry)
        wkMin = 100
        wkMax = 0
        For j = 0 To n
            wk = arry(j)
            If wk > 99 Then
                wk = 99
            End If
            If wk < wkMin Then
                wkMin = wk
            End If
            If wk > wkMax Then
                wkMax = wk
            End If
        Next j
        Cells(i, 2) = wkMin
        Cells(i, 3) = wkMax
        i = i + 1
    Loop While Cells(i, 1) <> ""
    i = 1
    wkMin = 100
    wkMax = 0
    Do While Cells(i, 2) <> ""
        If wkMin > Cells(i, 2) Then
            wkMin = Cells(i, 2)
            j = i
        End If
        If wkMax < Cells(i, 3) Then
            wkMax = Cells(i, 3)
            k = i
        End If
        Cells(i, 4) = ""
        Cells(i, 5) = ""
        i = i + 1
    Loop
    Cells(j, 4) = wkMin
    Cells(k, 5) = wkMax

End Sub
```

このプログラムでは、Do - Loop While 文が含まれています。そのループ内に For 文が入れ子になっています。その For 文のループ内には if 文が３つ連続しています。このプログラムの制御パステストのテストケースを生成するモデルを**図９－７**に示します。

パラメータ	値の並び
7if	true, false
12do	do, -
17for	true, false, -
19if	true, false, -
22if	true, false, -
25if	true, false, -
28next	next, -
32loop while	true, false, -
36do while	true, false, -
37if	true, false, -
41if	true, false, -
48loop	loop, -

制約表

パラメータ	制約1	制約2	制約3	制約4
7if	true	false		
12do	-	do		
17for	-	true, false	true	false
19if	-		true, false	-
22if	-		true, false	-
25if	-		true, false	-
28next	-		next	-
32loop while	-		-	true, false
36do while	-		-	
37if	-		-	
41if	-		-	
48loop	-		-	

制約表

パラメータ	制約5	制約6	制約7	制約8
7if				
12do				
17for				
19if				
22if				
25if				
28next				
32loop while	true	false		
36do while	-	true, false	true	false
37if	-		true, false	-
41if	-		true, false	-
48loop	-		loop	-

図９－７　ループを含むプログラムのモデル（その２）

このモデルで生成したテストケースを次に示します。

表９－１０　ループを含むプログラムのテストケース（その２）

No.	7if	12do	17for	19if	22if	25if	28next	32loop while	36do while	37if	41if	48loop
1	FALSE	do	FALSE	-	-	-	-	FALSE	TRUE	TRUE	FALSE	loop
2	FALSE	do	FALSE	-	-	-	-	TRUE	-	-	-	-
3	FALSE	do	FALSE	-	-	-	-	FALSE	TRUE	FALSE	TRUE	loop
4	FALSE	do	FALSE	-	-	-	-	FALSE	TRUE	FALSE	FALSE	loop
5	FALSE	do	FALSE	-	-	-	-	FALSE	FALSE	-	-	-
6	FALSE	do	FALSE	-	-	-	-	FALSE	TRUE	TRUE	TRUE	loop
7	FALSE	do	TRUE	TRUE	TRUE	FALSE	next	-	-	-	-	-
8	FALSE	do	TRUE	FALSE	TRUE	TRUE	next	-	-	-	-	-
9	FALSE	do	TRUE	FALSE	FALSE	FALSE	next	-	-	-	-	-
10	FALSE	do	TRUE	TRUE	FALSE	TRUE	next	-	-	-	-	-
11	TRUE	-	-	-	-	-	-	-	-	-	-	-

このテストケースの制御パスをすべて網羅するようにテストデータのセット（ワークシートＡ列の内容）を決めます。テストデータの例を**表９－１１**に示します。この例では、Ａ列の各テストデータのセットがテストケースの No.のどれを通るかを表しています。

表９－１１　テストデータのセットと対応するテストケース番号

行番号	Data Set	Test Case #
1	1	8, 2
2	1, **0**, 1	8, 9, 2
3	99, **100**	8
4	（空白行）	6, 1, 4, 3, 5

表９－１１では、太字で示した値が通るパスが**表９－１０**のテストケースにありません。これは Pairwise 法によるテストケースのため、経路網羅と異なりすべてのパスを網羅していないことが理由です。また、テストケースの No.7 と No.11 が通らないパスとして残っていますが、No.11 については、Ａ列１行目が空白のテストデータで網羅されます。しかし、No.7 のテストケースについては、プログラム制御構造上、どのようなテストデータでも通れないパスとなっています。具体的には、19if が TRUE、22if が TRUE の場合、25if が FALSE というパスは通れないパスです。そこで通れないパスがテストケースに含まれないように**図９－８**の制約を追加して新しく生成したテストケースを**表９－１２**に示します。

制約表

パラメータ	制約9
7if	
12do	
17for	
19if	true
22if	true
25if	true
28next	
32loop while	
36do while	
37if	
41if	
48loop	

図９－８　追加した制約

表９－１２　修正したテストケース

No.	7if	12do	17for	19if	22if	25if	28next	32loop while	36do while	37if	41if	48loop
1	FALSE	do	FALSE	-	-	-	-	TRUE	-	-	-	-
2	FALSE	do	FALSE	-	-	-	-	FALSE	TRUE	FALSE	TRUE	loop
3	FALSE	do	FALSE	-	-	-	-	FALSE	TRUE	TRUE	TRUE	loop
4	FALSE	do	FALSE	-	-	-	-	FALSE	FALSE	-	-	-
5	FALSE	do	FALSE	-	-	-	-	FALSE	TRUE	TRUE	FALSE	loop
6	FALSE	do	FALSE	-	-	-	-	FALSE	TRUE	FALSE	FALSE	loop
7	FALSE	do	TRUE	TRUE	FALSE	FALSE	next	-	-	-	-	-
8	FALSE	do	TRUE	FALSE	TRUE	FALSE	next	-	-	-	-	-
9	FALSE	do	TRUE	FALSE	FALSE	TRUE	next	-	-	-	-	-
10	FALSE	do	TRUE	TRUE	TRUE	TRUE	next	-	-	-	-	-
11	TRUE	-	-	-	-	-	-	-	-	-	-	-

このテストケースの制御パスをすべて網羅するようにしたテストデータのセットの例を**表９－１３**に示します。

表９－１３　修正したテストデータのセットと対応するテストケース番号

行番号	Data Set	Test Case #
1	**10**, 5	10, 5
2	**3**, 2, 4	10, 7, 5
3	100, 100	9, 8, 5
4	100, 0	9, 10, 5
5	**2**	5
6	（空白行）	3, 2, 6, 2, 4, 1

修正したテストデータでは、テストケースのすべてを網羅しています。この表で、太字で示した Data の通るパスの一部がテストケースにありませんが、テストケース全体から見れば問題となるものではありません。

このプログラムの通常の条件網羅のテストケースを次に示します。

表９－１４　通常の条件網羅のテストケース

No.	7if	12do	17for	19if	22if	25if	28next	32loop while	36do while	37if	41if	48loop
1	FALSE	do	FALSE	-	-	-	-	FALSE	TRUE	TRUE	TRUE	loop
2	FALSE	do	FALSE	-	-	-	-	TRUE	-	-	-	-
3	FALSE	do	FALSE	-	-	-	-	FALSE	TRUE	FALSE	FALSE	loop
4	FALSE	do	FALSE	-	-	-	-	FALSE	FALSE	-	-	-
5	FALSE	do	TRUE	TRUE	TRUE	TRUE	next	-	-	-	-	-
6	FALSE	do	TRUE	FALSE	FALSE	FALSE	next	-	-	-	-	-
7	TRUE	-	-	-	-	-	-	-	-	-	-	-

この例では７個のテストケースとなりました。このプログラムでは連続した if 文が多くあるため、Pairwise 法による条件網羅のテストケースより５個少なくなっています。

９．６　制御パステストのまとめ

PictMaster を制御パステストのテストケース生成に適用する場合のポイントをまとめます。

(1) プログラムの重要性などに応じて通常の条件網羅とするか、Pairwise 法による条件網羅とするかを決める。

(2) 原則として制御パスを変える if 文などをパラメータに採用する。ただし、分かりやすさのために代入文も含めてよい。

(3) パラメータの値は制御パスを変更する TRUE、FALSE などを採用する。ただし、分かりやすさのために実際の値を採用してもよい。この場合は境界値分析・同値分割の考え方を適用する。ダミーの値も定義する。

(4) 生成したテストケースをすべて網羅するようにテストデータのセットを決める。

Pairwise 法（All-Pair 法）による制御パステストのテストケースを生成するやり方には、こうしなければならないという重要な決まりごとはほとんどありません。それだけ記述の自由度が高い訳ですが、どのように記述するかのよって制御パステストの質が決まると言えます。そしてこれは生成エンジンの処理能力の問題ですが、パラメータの数はむやみに多くすることはできないことに注意してください。

１０．困ったときは

（1）「'PictMaster.xls' が見つかりません。」というエラーメッセージが表示される

ローカルのPC 上で作成したBook を別の環境（他のPC など）で開き、生成、整形、環境設定のボタンをクリックすると、「'PictMaster.xls' が見つかりません。ファイル名およびファイルの保存場所が正しいかどうか確認してください」 といったメッセージが表示されて、ボタンが働かないときは、ボタンとVBA（マクロ）のつながりが切れている場合です。このメッセージが表示されたときは、当該ボタンを右クリックし、「マクロの登録…」を選び、表示されるマクロのリストから、ボタンの名称と同じマクロを選択し、OK をクリックすれば問題は解決します。

（2）ウインドウ分割が正しく行なわれない

バージョンの新しいPictMaster とバージョンの古いPictMaster を同時に開いた状態で、ウインドウ分割を行なうショートカットを指定すると、ウインドウ分割が正しく行なわれない場合があります。これはバージョンの違いにより、ウインドウ分割の処理方法に互換性がないために起きる現象です。この場合は開いているBookを１つにしてください。

（3）既存のBookをPictMasterのワークシートのコピー先に指定するとエラーとなる

既存のBook（PictMasterではない）をPictMasterのワークシートのコピー先に指定することはできません。PictMasterのBookに新規ワークシートを設けて、そこに既存のBookのワークシートの内容をコピー＆ペーストしてください。

（4）「実行時エラー ‘70’： 書き込みできません」というエラーメッセージが表示される

モデルファイル a.txt を開いた状態で生成を行なうとこのエラーメッセージが表示されます。
a.txt を閉じてから生成を行なってください。

（5）「実行時エラー‘62’： ファイルにこれ以上データがありません」というエラーメッセージが表示される

文字コード変換プログラム **nkf.exe** がPICTのあるフォルダ内になく、PictMasterのあるフォルダ内にもないとこのエラーメッセージが表示されます。これは最少テストケース生成のときで、１回だけの生成時は**空白のシートが表示**されます。
PICTのあるフォルダ内に nkf.exe を置いてください。

（6）数値のセルにエラーマークが表示される

PictMaster 4.0以降では、生成結果はテキスト形式のセルに格納されるようになりました。そのため、Excelのデフォルトの設定では、セルのテキスト形式と内容が数値形式で一致しないとして、セルにエラーマークが表示されます。
Excelで生成結果が数値のセルにエラーマークが表示されないようにするには、以下の設定を行なってください。

Excel2007の場合

Officeボタン → Excelのオプション → 「数式」 → エラー チェック ルール のカテゴリの 「文字列形式の数値、またはアポストロフィで始まる数値(H)」 のチェックを外す。

Excel2003の場合

ツール → オプション → エラーチェック のタブ → 「文字列として保存されている数値」のチェックを外す。

附録Ａ　仕様

No.	項　目	値
1	パラメータの最大個数	30
2	パラメータあたり値の最大個数	30
3	制約表の最大制約数	50
4	結果表の最大条件数	30
5	結果表の1つの行に記入可能な値展開後の値の最大個数（*1）	300
6	結果表で処理可能な生成結果の最大行数	10000
7	パラメータの組み合わせ数範囲	1～30
8	最少テストケース生成試行回数範囲	2～9999
9	デフォルトの試行回数	30
10	最少テストケース生成シード値の範囲	0～65535
11	デフォルトのシード値	0
12	整形可能な最大行数	10000
13	原型シートの最大行数	10000
14	指定可能なサブモデルの最大個数	PICT に依存
15	値の重み付けの指定範囲	2 倍～10 倍
16	重複した組み合わせを削除可能な最大行数	10000

*1: 1 つの行のすべてのパラメータの列に記入された値の個数の合計。値がエイリアスを含む場合は、エイリアスで指定した値の数をその値の個数に加算してカウントする。

付録Ｂ　制限事項

【重要な注意点】

多くのパラメータ、値、または制約を指定した場合など、PICT の処理能力の限界により、**テストケースを生成するまできわめて長時間かかる場合があります**、こうした場合に処理を途中で止めたい場合は、タスクマネージャを起動し、プロセスから pict.exe を選択し、プロセスの終了を行なってください。

あらかじめ処理時間がかかりそうに思われる場合は、最少テストケース生成は行なわず、1 回だけの生成を行なって処理時間を確認することをお勧めします。