ソフトウェアテストの本質を振り返る

Agenda

- ソフトウェアテストとは
- ソフトウェアのテスト技法とは
- 技法の振り返り
  - 同値分割法
  - 境界値分析
  - デシジョンテーブルテスト
  - CFD法
- まとめ

ソフトウェアテストとは？

1. 欠陥を検出する
   - 検出した欠陥を修正すれば、ソフトウェアの品質が確保できる
2. 対象となるソフトウェアの品質レベルが十分であることを確認し、その情報を示す
   - 適切に設計したテストを実施して欠陥が検出されなければ、そのソフトウェアの品質レベルは高いと言える
   - テストを実施することで、品質を計測するための客観的な指標を得られる。（欠陥の検出率等）
3. 欠陥の作り込みを防ぐ
   - テスト要件を元に開発要件に対して揺さぶりをかけることで、未然に欠陥の混入を防いだり、テスト結果を分析して開発プロセスの改善へつなげることもできる

すなわち、ソフトウェアの品質を確保するための一連の作業といえる

45

良いソフトウェアテストとは？

1. 多くの欠陥を見つけることができる
   - 効果的にテストを実施する
2. 少ない時間で欠陥を見つけられる
   - 効率的にテストを実施する
3. テスト範囲を漏れなくテストする
   - 網羅的にテストを実施する

46

ソフトウェアのテスト技法とは？

- 先人の知恵（経験）を形式知にしたもの
  - 今まで行ってきたテストを技法として昇華
- テストを効果的、効率的、網羅的に行うための手段・道具

47

本セッションで振り返るテスト技法

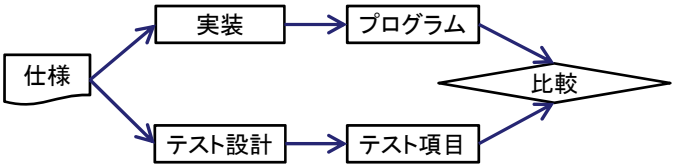
- 対象とするテスト技法
  - 同値分割法　・・・効率的
  - 境界値分析　・・・効果的、効率的
  - デシジョンテーブル　・・・網羅的
  - CFD（Cause Flow Diagram）　・・・網羅的、効率的

48

本セッションで振り返るテスト技法

- これらのテスト技法はブラックボックスのテスト技法と呼ばれる
  - ブラックボックステスト技法とは
    - 入力データに対する出力結果に着目し、機能が仕様通りであることを確認する（内部構造は意識しない）
    - 機能の振る舞いを確認するので仕様ベースのテスト技法とも言われる

IPOモデルの再設計そのもの！！



注：CFDはグレーボックステストと呼ばれるが、機能が仕様通りであることを確認するという意味で、ブラックボックステストと同様とみなすことができます。

49

技法の振り返り（同値分割法）

- 同値分割法
  - ブラックボックステスト技法の1つ。同値分割した領域から代表値を実行するテストケースを設計する。最低1回各同値領域を実行するように設計するのが原則。（JSTQB）
- 同値分割
  - 仕様に基づき、コンポーネントやシステムの振る舞いが同じとみなせる入力ドメインや出力ドメインの部分。（JSTQB）

50

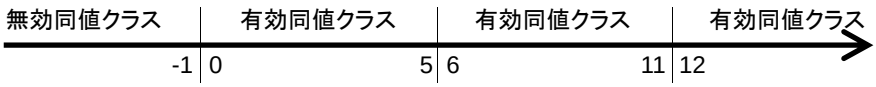
技法の振り返り（同値分割法）

- テストに使う入力値が、同様の結果をもたらす場合、その入力値を「同値」と呼び、同値の取りうる範囲を「同値クラス」と呼ぶ。同値クラスは“有効同値クラス”“無効同値クラス”に分割することが出来る
  - 有効同値クラス・・・入力値に対して正常系の処理を行うクラス
  - 無効同値クラス・・・入力値に対して異常系の処理を行うクラス
- 同値クラスに分ける場合、処理や出力結果に着目する場合が多い

51

技法の振り返り（同値分割法）

＜例＞年齢で料金を判定するシステムを考える（年齢は整数）。
「6歳未満は無料」
「6歳以上12歳未満は小児料金」
「12歳以上は通常料金」
とした場合



無効同値クラスは、
年齢＜0歳
有効同値クラスは、
0歳≦年齢＜6歳　　6歳≦年齢＜12歳　　年齢≧12歳
に分けられる。

よって、4つの同値クラスにそれぞれ代表値を1つずつ選ぶことになります。
例として入力データは、－5、3、9、30を挙げることができます。

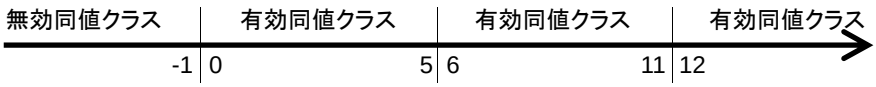
52

技法の振り返り（同値分割法）

- なぜ同値分割を行う必要があるのか？
  - 効率的なテストをしたいから！！
    - 仕様からデータを“意味のあるグループ”（同値クラス）に分類することで無駄なテストケースを削減することができる
  - テストの偏りをなくしたいから！！
    - 同じような意味を持つデータばかりの偏ったテストケースになることを避けることができる

技法の振り返り（同値分割法）

＜例＞年齢で料金を判定するシステムを考える（年齢は整数）。
「6歳未満は無料」
「6歳以上12歳未満は小児料金」
「12歳以上は通常料金」
とした場合



正常値のテストをする場合のテストデータ数は、
年齢は0歳から120歳としても、121通りのデータが必要。

同値分割法を使うことで、正常値のテストデータは
3、9、30
の3つにまで削減することができます。
※実際は無効同値クラスのテストが必要な場合もあります。
その場合は、“-5“などを加えて4つのテストデータが必要になります。

同値分割の適用範囲

- 同値分割の考え方はテストだけには収まらない
  - システム設計時
    - システムが解決する領域を明らかにする場合、すなわち、システムが解決する領域を有効同値クラス、それ以外を無効同値クラスと考えることができる
  - プログラム外部設計時（モジュール分割時）
    - プログラムへの入力値によってプログラムの振舞いが異なる場合、入力を有効同値クラス分けすることで、適切にモジュール機能分割できる
  - プログラム詳細設計時
    - プログラムへの入力値によって処理が分岐する場合、有効範囲内外が同値クラスとして分析される。その同値クラスの境界値を条件判断・分岐（IF-ELSE）のパラメータとして用いることができる
  - テスト設計時
    - ホワイトボックステストのコードカバレッジを高めるために、テストデータはプログラムの条件判断・分岐（IF-ELSE）に用いる値、すなわち有効同値クラス、無効同値クラスを適用することができる

同値分割の意識を常に持とう！！

55

技法の振り返り（境界値分析）

- 境界値分析
  - ブラックボックステスト技法の1つ。境界値に基づいてテストケースを設計する。Boundary valueも参照のこと。（JSTQB）
- 境界値
  - 同値分割した領域の端、あるいは端のどちらか側で最小の増加的距離にある入力値または出力値。たとえばある範囲の最小値、最大値。（JSTQB）

56

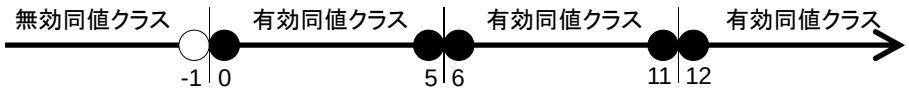
技法の振り返り（境界値分析）

- 「境界値分析」は「限界値分析」とも呼ばれる
- 通常は、同値分割とセットで実施する
  - 同値クラスの端を狙うので、同値分割を実施してからでないと境界値分析はできない
- 同値分割では同値クラスから任意の代表値を1つ選ぶだけでよいが、境界値分析では同値クラスの上限と下限から2つの値を選び出す場合があるので、テストに必要な代表値は増える

57

技法の振り返り（境界値分析）

＜例＞年齢で料金を判定するシステムを考える（年齢は整数）。
「6歳未満は無料」
「6歳以上12歳未満は小児料金」
「12歳以上は通常料金」
とした場合



●：有効境界　○：もっとも近傍の無効境界

対象となるテストデータは、−1、0、5、6、11、12となる

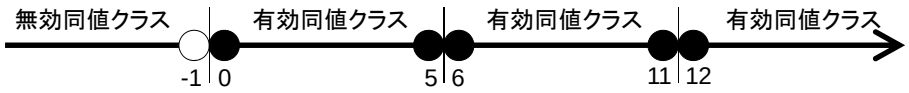
【参考】
同値分割法のテストデータは、−5、3、9、30の4つ。

58

技法の振り返り(境界値分析)

- なぜ、境界値を分析する必要があるのか？
  - 境界には、人が犯した誤りが紛れ込みやすい！！
    - すなわち、人が犯した誤りを効果的に見つける必要がある
- 人が犯す誤り(ヒューマンエラー)とは？
  - ミステイク・・・計画時の誤り。正しく実行はできたが、前提条件や認識が誤っている
    - “以下”を不等号の“＜“と認識誤りをしていたためにコーディング時に”＜”と書くことでエラーが発覚
  - スリップ・・・実行時の誤り。計画自体は正しいが実行の段階で誤りを犯す。“うっかりミス”はこのことを指す。
    - “以下”を不等号の“＜＝“と正しく認識できているが、仕様書作成時やコーディング時に誤って“＜”や“＞”と書いてしまう

境界値分析(Question！！)



- 数直線を見て、何か言いたい人いませんか？
  - 私はこれで大丈夫と思っているんですが・・・

技法の振り返り(境界値分析)

- 仕様を元に分けた同値クラス以外にも境界がある
  - “年齢で料金を判定するシステム”で、年齢をbyte型で指定している場合は、byte型の範囲は-128 ～ 127なので128歳は入力エラーとなる
- 仕様に書かれていない境界も考慮する必要がある
  - “年齢で料金を判定するシステム”の条件「12歳以上は通常料金」
    - 年齢が150歳はどうしますか？5000歳は？
      - このように上限(もしくは下限)の再検討が必要な場合、設計者に確認をとり、同値クラスを再検討する必要があります。

61

技法の振り返り(デシジョンテーブルテスト)

- デシジョンテーブルテスト
  - ブラックブラックボックステスト設計技法の1つ。デシジョンテーブルにある入力と刺激(原因)の組み合わせを実行するテストケースを設計する。 decision tableも参照のこと。(JSTQB)
- デシジョンテーブル
  - 入力と刺激(原因)、及び、対応する出力と処理(結果)の組み合わせを示す表。テストケースの設計に利用できる。(JSTQB)

62

技法の振り返り（デシジョンテーブルテスト）

- デシジョンテーブルそのものはテストのために考案されたものではなく、システム分析や設計情報を整理して記述するために開発された手法
- 日本語で決定表とも呼ぶ（JISX0125　決定表）
- 複雑な仕様の整理、確認に有効
- デシジョンテーブル作成後のテストケースは同値分割法や境界値分析を併用して抽出

技法の振り返り（デシジョンテーブルテスト）

- プログラムの動作を、入力条件と出力動作の組合せに対応付けた表
- 条件部と動作部に分けて、入力の組合せと結果を整理
  - 条件部：“Y”=有効、”N”=無効、”－”=どちらでもよい（結果に影響しない）
  - 動作部：“X”=動作実行、”－”=実行しない

※拡張表記として、条件部、動作部共に「語句、値またはコード」も記述可能

ルール	1	2	3	4	5
条件					
電源スイッチを押す	N	Y	Y	Y	Y
ボタンAを押す	－	N	Y	N	Y
ボタンBを押す	－	N	N	Y	Y
動作					
赤ランプ	消	消	点	消	点
青ランプ	消	消	消	点	点
ビープ音	鳴らない	鳴らない	鳴らない	鳴らない	鳴る

技法の振り返り（デシジョンテーブルテスト）

- デシジョンテーブルの条件部分は、（if文やswitch文になる）仕様の分岐条件からも導き出されるので、同値分割によって分けられる同値クラスがその入力条件になることが多い
- 条件部と動作部の関連は、
  IF “条件” THEN “動作” となる

“年齢で料金を判定するシステム”のデシジョンテーブル

	ルール	1	2	3
条件	年齢<6歳	Y	–	–
	6歳≦年齢<12歳	–	Y	–
	12歳≦年齢	–	–	Y
動作	無料	X	–	–
	小児料金	–	X	–
	通常料金	–	–	X

65

技法の振り返り（デシジョンテーブルテスト）

- なぜデシジョンテーブルを作成するのか？
  - 入出力の組み合わせをテスト時に再設計することで仕様に誤りがないことを確認したいから！！
  - 条件の組み合わせを網羅したいから！！
  - 複雑に絡み合う組み合わせを明確にしたいから！！
- デシジョンテーブル作成時は以下の点に注意
  - 入力条件、動作結果の漏れや誤り
  - ルール間の矛盾
    - うるう年の計算で、400では割り切れるが、100で割り切れない
  - 条件数が増えるとルール数が爆発する
    - 条件となる項目が5種類あり、それぞれがY/Nの2種類の条件だとしても、2の5乗となり32通りのルールが適用される

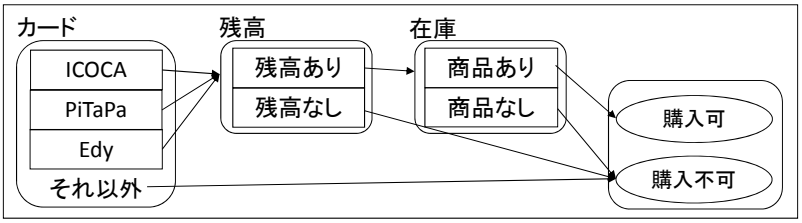
66

技法の振り返り（CFD法）

- CFD法
  - 複雑な論理関係の仕様から重要なテスト条件を漏らさない技法
  - CFD法はCFDという図を作成するだけではなく、「同値分割」→「CFD作成」→「デシジョンテーブル作成」の一連のテスト技法が組み合わさったもの
  - 原因結果グラフ（CEG）を超えることを目指して日本で開発された技法
    - 現在も発展途上中
  - CFD法は処理の流れが重要となるため、実装情報（プログラム構造）も必要となる
    - 処理の順序を意識することで、無駄なテストケースが削減できる

技法の振り返り（CFD法）

- CFD法
  - ソフトウェアに与えられる入力（原因）と機能（結果）を、それぞれ集合で用いるベン図と同様の表記法で図式化する
    - ベン図のことを“同値分割図“と呼ぶ
  - 仕様に書かれていない入力の組み合わせや出力結果を検討するので、設計漏れやテストケース漏れを防ぐことができる
  - 「原因の集合」と「原因同士のつながり」に着目し、“流れ線”で「結果」へつなぐことで仕様を図式化する



技法の振り返り（CFD法）

- CFD法によるテスト設計手順
  1. 機能（結果）の同値分割を行う（結果の同値分割図）
  2. 入力（原因）の同値分割を行う（原因の同値分割図）
  3. 実装情報による構造配置を行う（原因と結果が配置されたCFD）
  4. 流れの推測を行う（流れ線が記述されたCFD）
  5. デシジョンテーブルを作成する（デシジョンテーブル）

＜例＞年齢で料金を判定するシステムを考える（年齢は整数）。
「6歳未満は無料」
「6歳以上12歳未満は小児料金」
「12歳以上は通常料金」

69

技法の振り返り（CFD法）

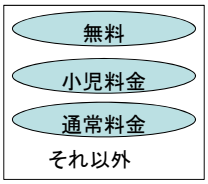
- CFD法によるテスト設計手順
  1. 機能（結果）の同値分割を行う（結果の同値分割図作成）
     1. 仕様に書かれている機能や処理の同値クラスを探す
     2. 仕様に書かれていない機能や処理の同値クラスを探す
        - 1.で抽出された同値（部分集合）の補集合を探す
     3. 結果の同値分割図を作成する
        - 補集合も含めたすべての同値が洗い出された同値分割を「完全同値分割」と呼ぶ

70

技法の振り返り(CFD法)

- CFD法によるテスト設計手順
  1. 機能(結果)の同値分割を行う(結果の同値分割図作成)
     1. 仕様に書かれている機能や処理の同値クラスを探す
        “無料”、“小児料金”、“通常料金” の3つの同値クラス
     2. 仕様に書かれていない機能や処理の同値クラスを探す
        どんな同値クラスがあるかはっきりしない場合は“それ以外“として仮置きしておく
     3. 結果の同値分割図を作成する

結果の同値分割図

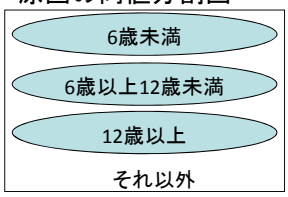


71

技法の振り返り(CFD法)

- CFD法によるテスト設計手順
  2. 入力(原因)の同値分割を行う(原因の同値分割図作成)
     1. 仕様に書かれている原因の同値クラスを探す
        “6歳未満”、“6歳以上12歳未満”、“12歳以上”
     2. 仕様に書かれていない原因の同値クラスを探す
        どんな同値クラスがあるかはっきりしない場合は“それ以外“として仮置きしておく
     3. 原因の同値分割図を作成する

原因の同値分割図



72

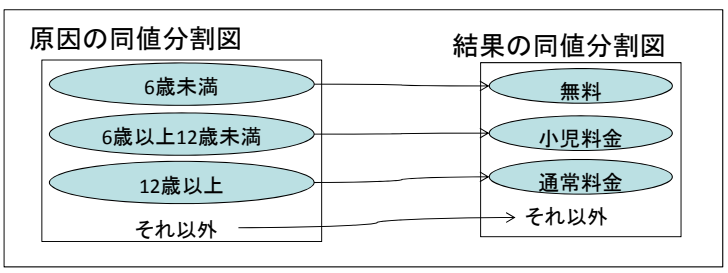
技法の振り返り（CFD法）

- 仕様に書かれていない同値クラスの見つけ方
  - プログラミングを考えた時のif-else文の最後のelseに相当する同値クラスを洗い出す
- 同値クラス（“それ以外“）の明確化のタイミング
  - CFD作成時は“それ以外”の詳細については、考えない。テストデータを作成するときに考える。結果に影響を与える影響から、同じような特性を持った入力要素の集合を同値として束ねる

技法の振り返り（CFD法）

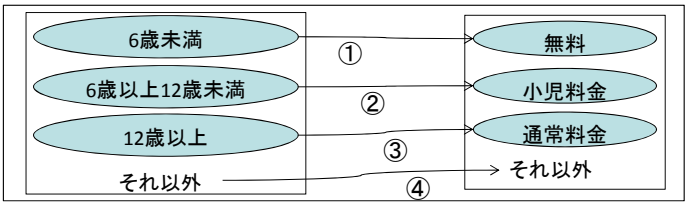
- CFD法によるテスト設計手順
  3. 実装情報による構造配置を行う（原因と結果が配置されたCFD）
  4. 流れの推測を行う（流れ線が記述されたCFD）
     有効系と無効系の流れ線を引く

CFD



技法の振り返り（CFD法）

- CFD法によるテスト設計手順
  5. デシジョンテーブルを作成する（デシジョンテーブル）



流れ線をテストケース番号の縦に転記する

テストケース番号		1	2	3	4
原因	6歳未満	Y			
	6歳以上12歳未満		Y		
	12歳以上			Y	
	それ以外				Y
結果	無料	X			
	小児料金		X		
	通常料金			X	
	それ以外				X

75

技法の振り返り（CFD法）

- なぜCFD法を使うのか？
  - テストの漏れをなくしたい！！
    - “動作しない”などの、仕様に書かれていない機能や処理に対するテストケースも抽出することができる
  - 無駄なテストを省きたい！！
    - 実装情報を考慮することで、無駄なテストケースの省略が可能
      - ユーザIDとパスワード入力がある画面で処理順がユーザID→パスワードの場合、ユーザIDが誤っていて、パスワードが正しいという組み合わせのテストは省くことが可能
- CFDを見せることで、レビュー実施時にテストケース作成根拠が理解しやすくなる
  - デシジョンテーブルの作成根拠が明確になる

76

まとめ

- テスト技法を使用する際は、技法を使う目的を考えて適用する必要がある

テスト技法	目的	注意点
同値分割法	• 効率的にテストをする • テストの偏りをなくす	• 同値クラスの洗出しに失敗すると，該当するクラスのテストケースが丸ごと漏れる
境界値分析	• 人が犯した誤りが紛れ込みやすい境界付近をテストする	• 仕様を元に分けた同値クラス以外にも境界がある • 仕様に書かれていない同値クラスの境界がある
デシジョンテーブルテスト	• 入力条件の組み合わせによる仕様の漏れや誤りをテストする	• 入力条件、動作の漏れや誤り • ルール間の矛盾 • 条件数が増えるとテスト数が爆発する
CFD法	• 仕様に書かれていない機能をテストする • 入力条件の組み合わせから無駄なテストを省く	• 論理構造が明らかでない場合、テストケースが漏れる可能性がある

77

グループワーク3

78

Agenda

- グループワーク3解説
  - お題
- グループワーク
  - CFD法を用いる
  - テストケースを作成する
- 発表タイム

お題

- 対象システム”電気やかん”の「沸騰ボタン」を押した時（要求仕様書　番号 300，500）のテストケースを作成しましょう

※各テストケースに対して、なぜその項目が必要なのか（妥当性の説明）も合わせて考えてみてください

グループワーク3

- CFD法を用いる
  - 入力（原因）と処理（結果）に着目
    1. 同値分割図を作成する
    2. CFDを作成する
    3. デシジョンテーブルを作成する
- テストケースを作成する
  - 同値分割法や境界値分析よりテストのデータを作成

発表タイム