第1回
「ソフトウェアテスト技法ドリル」
勉強会

著者　秋山浩一さんと一緒に学ぼう！

2010/11/02

並木

はじめに

- ありがとうございます。
  - 会場提供と幹事の加瀬さん。
  - 幹事の鈴木さん。
  - オブザーバーの秋山さん。

さくっと自己紹介

- 川崎市のSI会社に勤務しています。
- 出没
  - WACATEには、ほぼ毎回参加。
  - もちろんWACATE2010冬に申し込み済み。
- TEF Web部会のお世話係です。
  - 最近盛り上がりイマイチ。残念。
  - ネタ投稿してくれるとうれしいです。

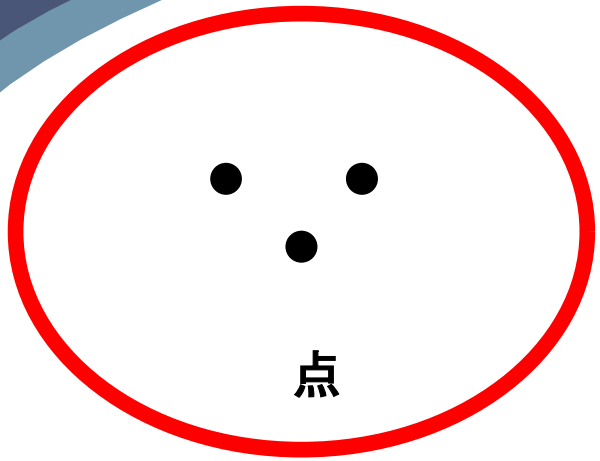
目次

第1章のまとめ

- ピンポイントテストの話
  - 怪しいところを探すこと、三色ボールペン
  - データバリエーションの考え方、間、対象、類推、外側
  - 意地悪テスト
- 過去の経験
  - 開発の現状
  - 未然防止
  - バグパターンの蓄積
  - 経験ベース

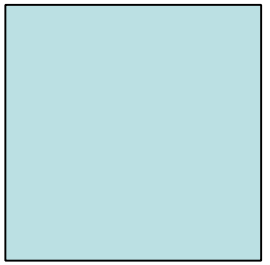
第1章 点に注意を向ける

- P.3
- 「網羅性」と「ピンポイント」
  - テストが上手か下手かとは、
    - 網羅性をもったテストを作成し実施できるか
    - うまくピンポイントで狙っていけるか
- 本書では、以下の順番で説明する
  - **点・線・面・立体・四次元・多次元**
  - 第1章は、点。

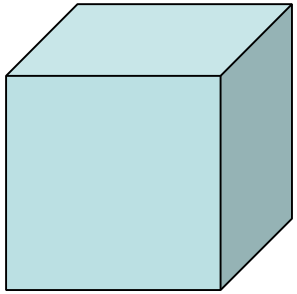


点

線

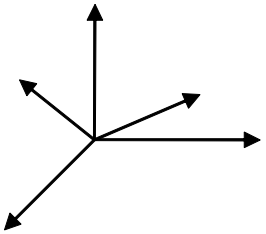


面



立体

時間



多次元

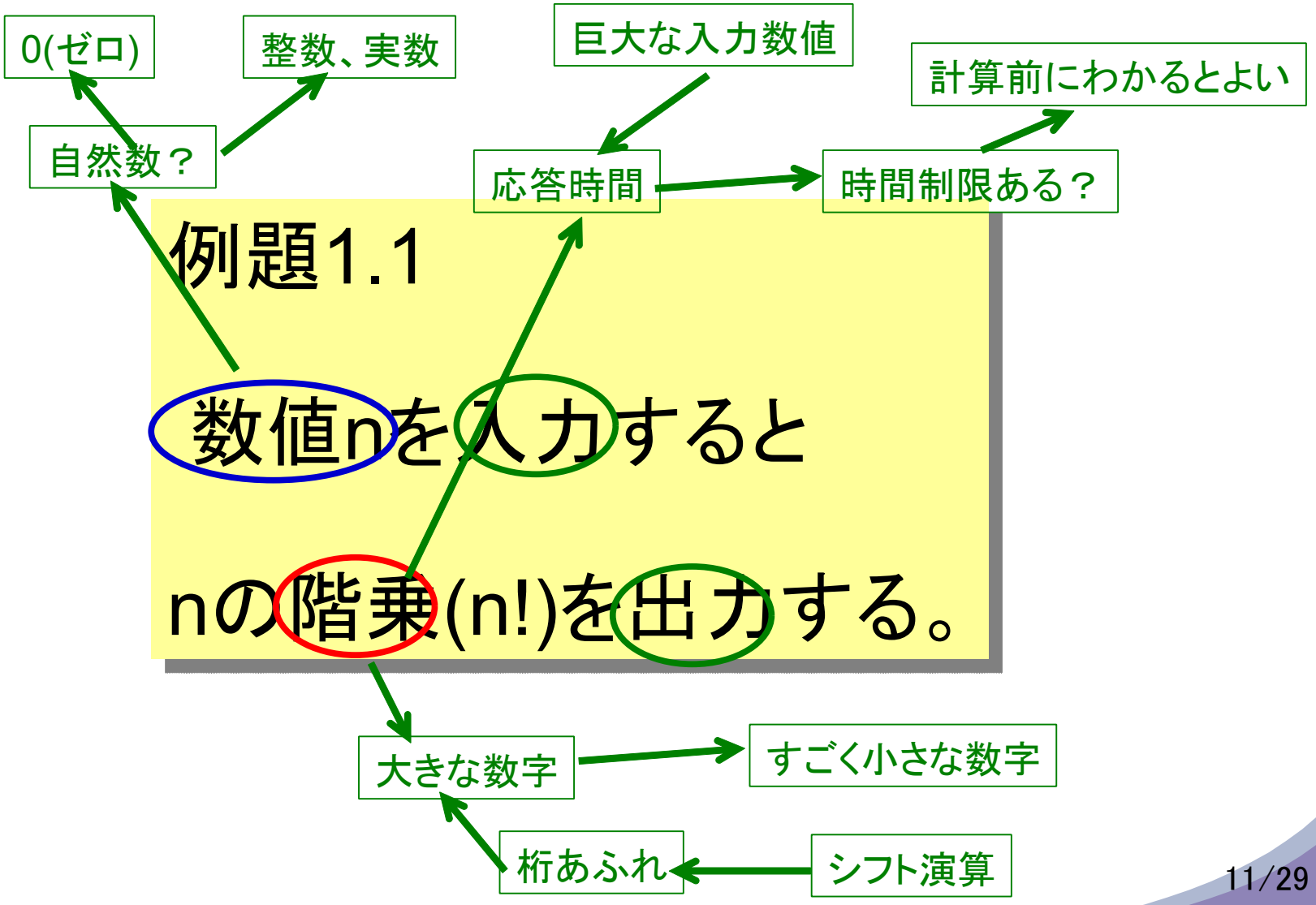
1.1 ピンポイントテスト

- ピンポイントテスト→狙い撃ちテスト
- バグを狙う方法を3つあげている
  - (1) 怪しい箇所の認識
    - 三色ボールペンを使う方法
  - (2) 間、対象、類推、外側を考える
    - データバリエーションの考え方
  - (3) 意地悪条件を考える
    - 思いっきり意地悪になる

三色ボールペンを使ってみる

- P.5「(1)怪しい箇所の認識」の内容をそのとおりに書いてみた。
  - 赤色・・・客観的に見て、最も重要な箇所
  - 青色・・・客観的に見て、まあ重要な箇所
  - 緑色・・・主観的に見て、自分が怪しいと感じた箇所
- 次のページに
  - 手書きの字を写真にと考えたのですが、汚すぎて断念。

三色ボールペン


0(ゼロ)
整数、実数
巨大な入力数値
計算前にわかるとよい
自然数？
応答時間
時間制限ある？
例題1.1
数値nを入力すると
nの階乗(n!)を出力する。
大きな数字
すごく小さな数字
桁あふれ
シフト演算

怪しいところ一覧

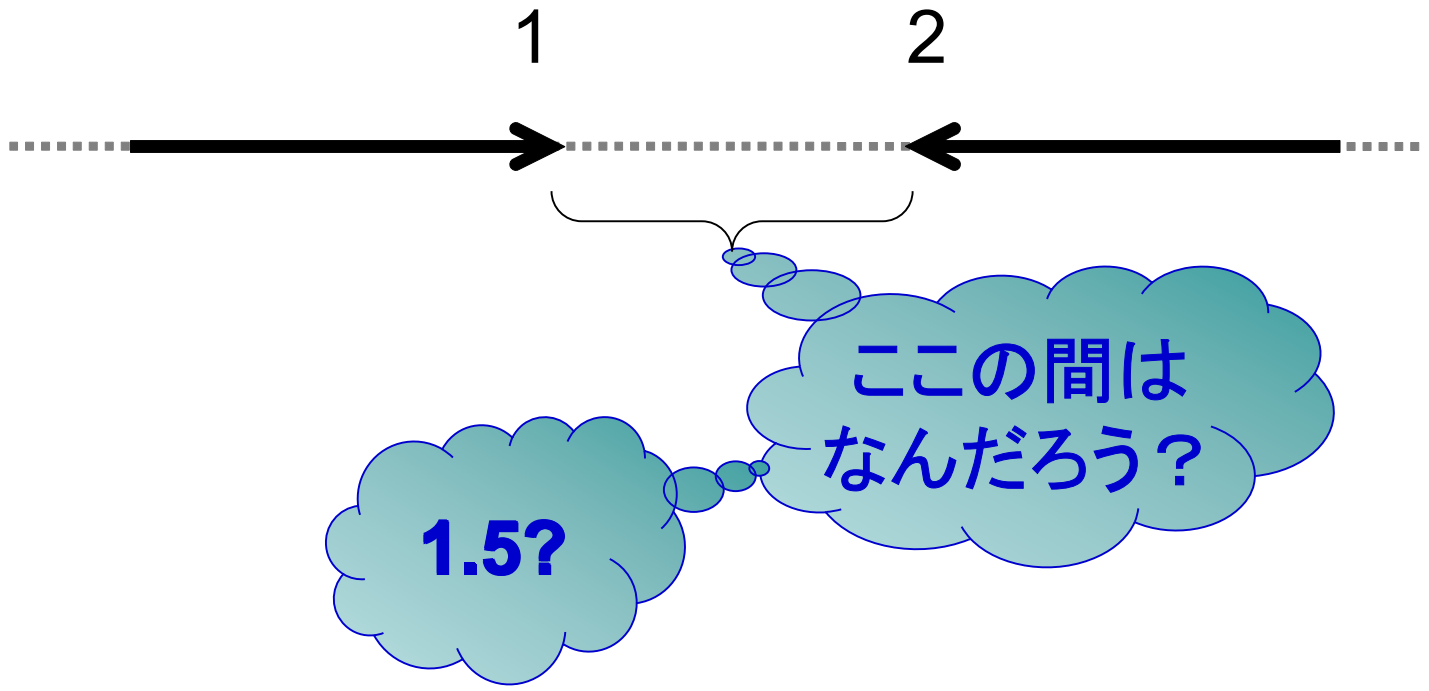
- たった1行の例題でも、多数の怪しいところが
  - 大きな数字
    - すごく小さな数字
    - 桁あふれ
    - シフト演算
  - 応答時間
    - 時間制限ある？
    - 計算前にわかるとよい
    - 巨大な入力数値
  - 自然数？
    - 0(ゼロ)
    - 整数、実数

何が怪しいのだろうか？

- 緑色にマークした部分。
  - 怪しい気がする
  - しかし何が怪しいかわからない
- 例示する。その後、他のバリエーションを考える。
  - データとデータの間を考える
  - データの対称を考える
  - 類推つまり類似しているものを想像する
  - 例示したものの外側を想像する

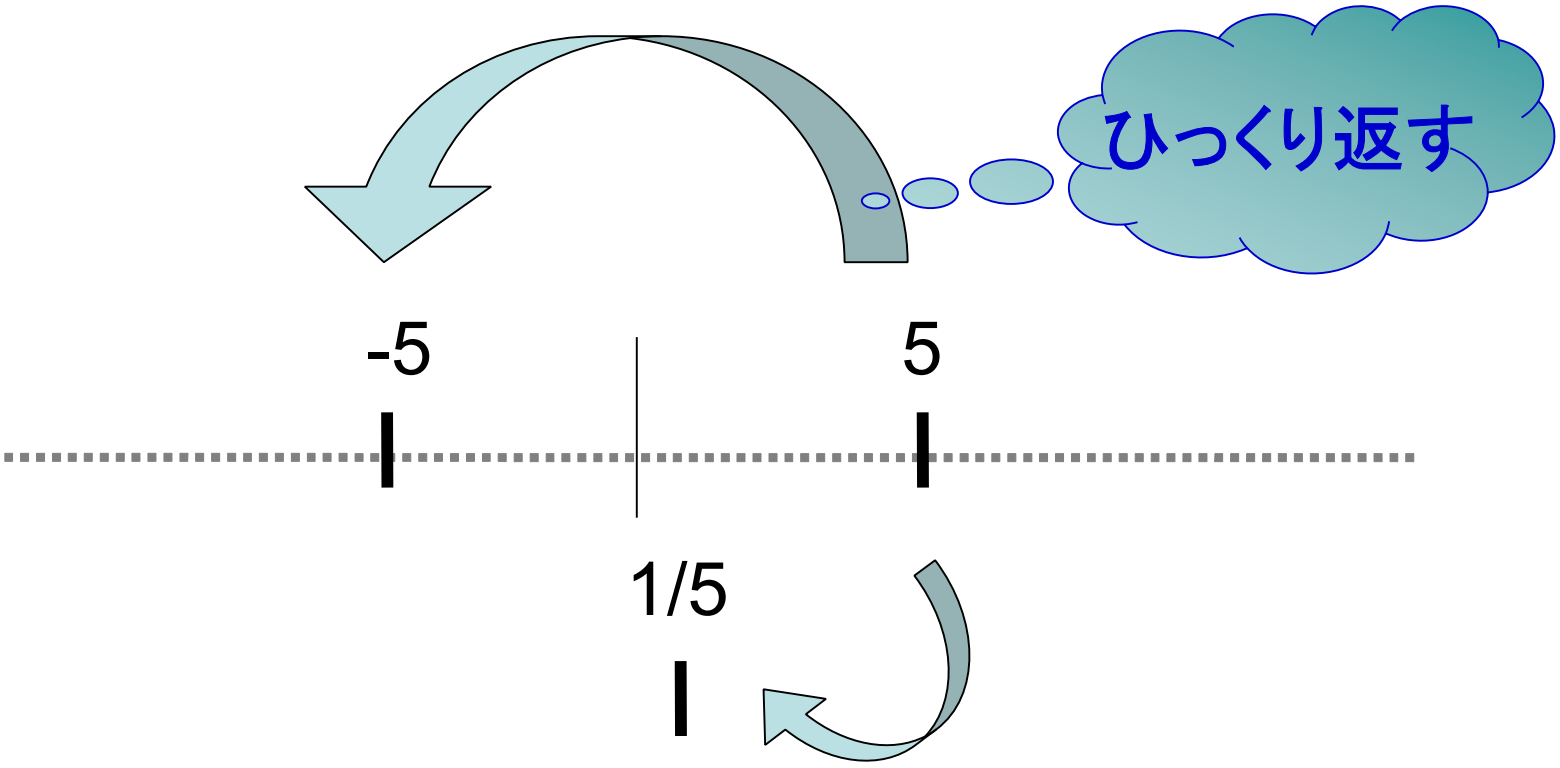
間

- データとデータの間を考える



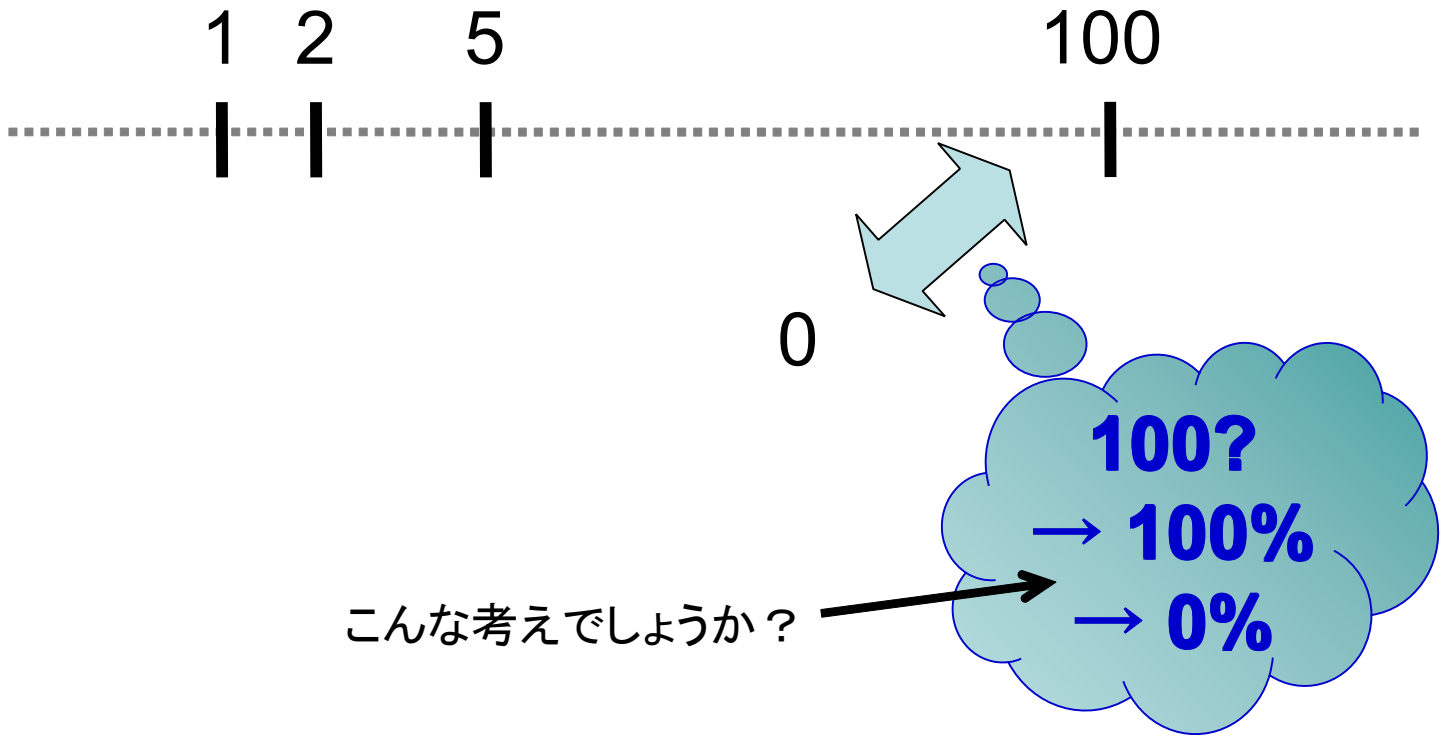
対称

- データの対称を考える



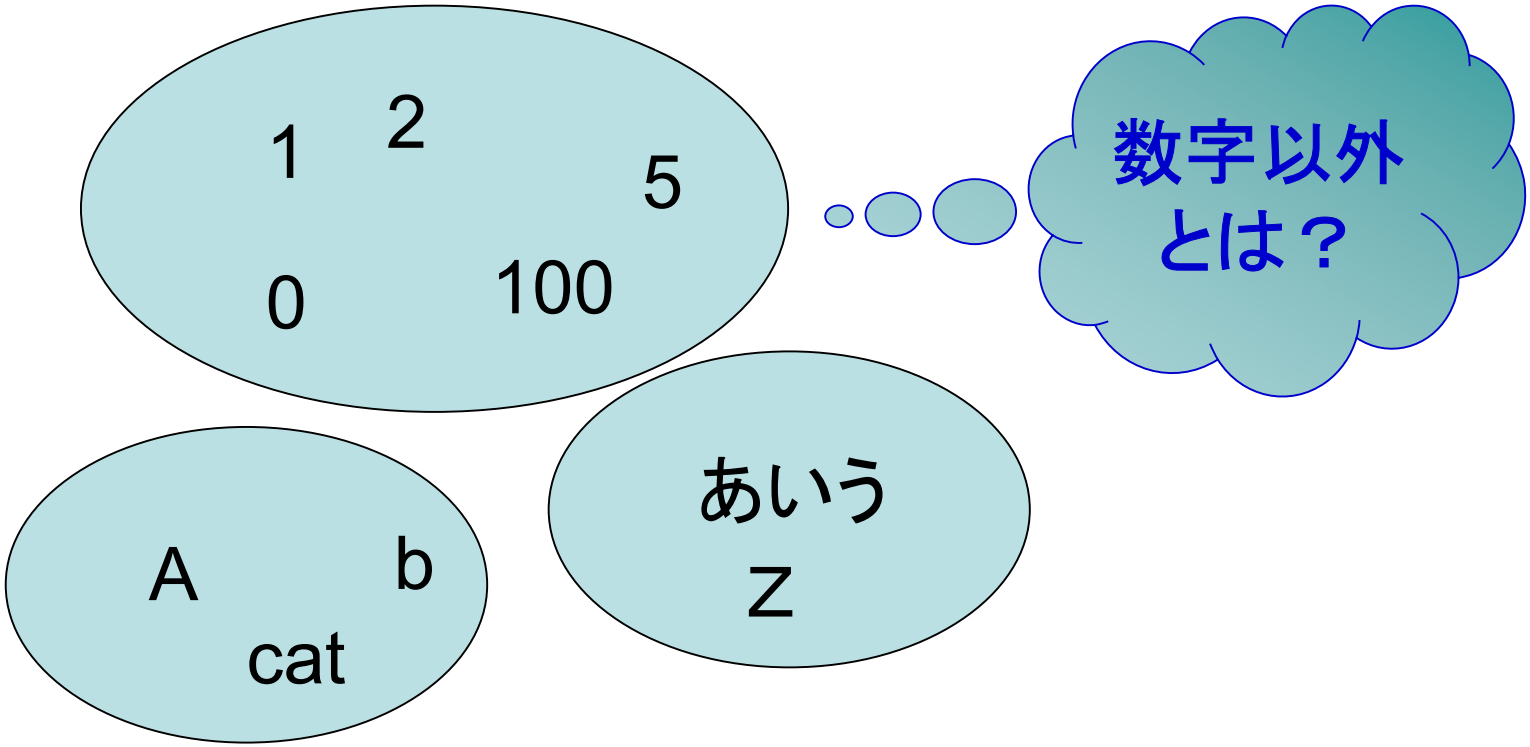
類推

- 類推つまり類似しているものを想像する



外側

- 例示したものの外側を想像する



(3) 意地悪条件を考える

思いっきり意地悪になる

意地悪になる

- 階乗計算の例で意地悪する
  - 与えるデータを大きなものにする
  - メモリを少なくするなどして動作環境を厳しくする
  - 手順をめちゃくちゃにしてみる
  - 小学生に操作させて何をするかわからないようなテストをする

1.2 過去の経験を活かす

- (1) ソフトウェア開発の現状
- (2) 不具合モードによる未然防止
- (3) バグパターンの蓄積トレーニング
- (4) 経験ベースのテスト

- 過去の経験を活かすこと
  - テストを点で捉えるもう一つの方法
- JUAS
  - システム開発の58%は保守運用
    - http://www.juas.or.jp/servey/it10/press-pp100409.pdf
    - 75ページ。
- バグ再発リスト
  - リリースするごとに増加してしまいテスト効率が下がることも
  - 根拠のない意見で、リストからばっさりと捨てられる
  - 似たようなバグが発生してしまう悲劇

(2) 不具合モードによる未然防止

- 不具合モード
  - 故障モードをSW向けに拡張
  - バグ発生のメカニズムを見極めてそれを開発中のソフトウェアに当てはめてテストで再発防止を図る
  - 西さん、河野さんが提案している
- SWのバグ発生メカニズム
  - **人間の習性**によるものが大半
    - 開発者のうっかりミス
    - 使用の読み間違えの起こしやすさ
    - 長時間残業による疲労
    - 担当の引継ぎの悪さ
- 日本人が間違いやすい問題でも、ブラジル人が間違えやすいとは限らない

(3) バグパターンの蓄積トレーニング

- 対策
  - バグ票を読むこと
    - シャワーを浴びるように
    -
  - 想像する
    - バグの原因から「どうしてそのようなバグを作りこんでしまったのだろう？」想像する
- 効果
  - デグレードも少なくなることが期待できる

(4) 経験ベースのテスト

- 経験ベースのテスト
  - エラー推測
    - 開発者が間違えを犯しやすいところが見えてくる
    - そしてそこを狙う
  - 探索テスト
    - エラー推測に似たテスト
    - 事前にテストケースを作ることはしない
- 「知っている」と思ったものこそ危険が潜んでいる
  - 性別の例題 性別の選択肢は4通りで
  - ISO 5218
  - 単独動作させているうちは、問題ないであろう
  - 別システムとつなげる時に問題となる

1.3 本章のまとめ(1)

- 怪しい点に注意を向ける
  - ピンポイントテストでは、三色ボールペンを使って仕様書を読むことを推奨
- 仕様書のあいまいなところ
  - 具体的なデータを示すこと
  - 間、対象、類推、外側の順番で考える
  - 思いっきり意地悪になること

1.3 本章のまとめ(2)

- 過去の経験を活かすこと
  - 「不具合モード」→バグを作り込んでしまうメカニズムを考えることの重要性

1.3 本章のまとめ(3)

- 2章以降のテスト技法を知った
- でも、どうすればいいかわからない、うまくいかないとき
  - 怪しいところをテストする方法が不足しているのでは？
  - 本章を読み直してテストのヒントを見つけて欲しい

終わり

- 以上です。