機能性テストで設計の違和感を可視化

2023/12/15

JaSST 東海 2nd

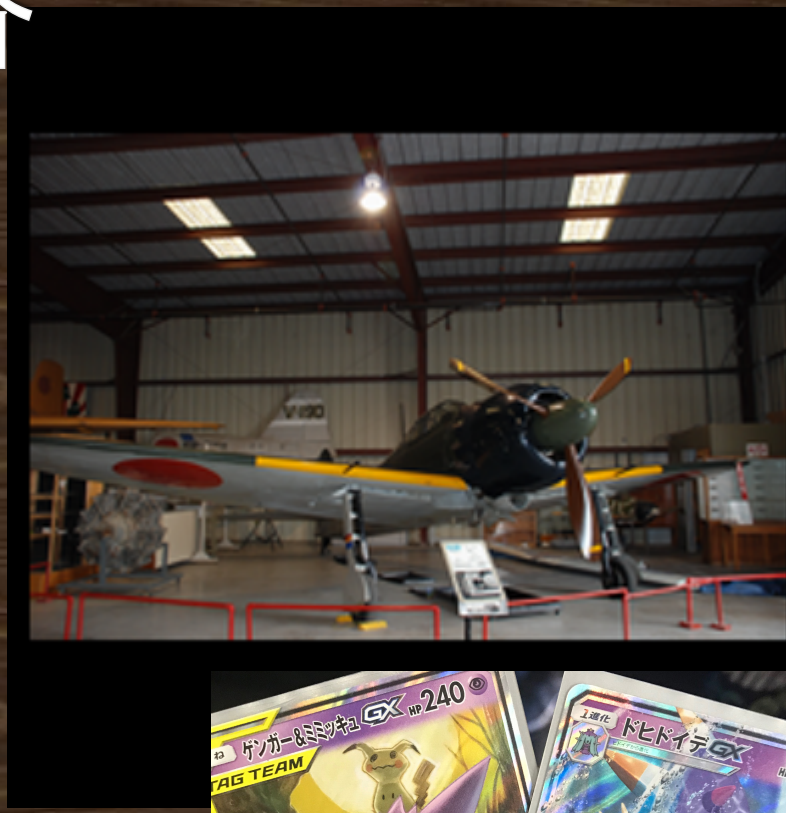
井関 武史

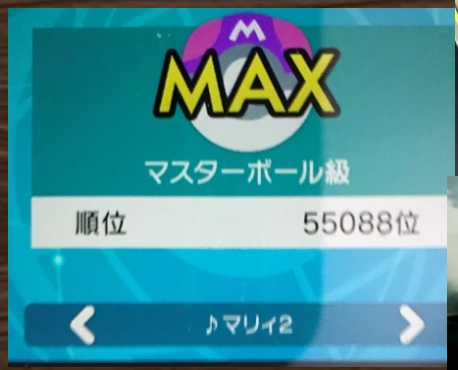
アジェンダ

- 目次
- 自己紹介
- はじめに
- テスト・QA エンジニアの役割
- 機能性のテスト分析・設計
- 違和感しかない設計
- まとめ

自己紹介

- 名前: 井関 武史（いせき たけふみ）
- 所属: テストの街「葛飾」
- 職業: 某 ID 管理製品のQA・テストエンジニア
- 趣味
  - 歴史
  - チンタラン（マラソン）
  - ゲーム
    - ポケモン（カード・Switch）、Acecombat、その他
- Twitter: @katsushika_take
- 生息地: 葛飾、神田小川町





テストの街 葛飾

はじめに

- プロダクト (サービス) の中核をなす機能性のテストで、どのようなテスト分析・設計をしていますか？
- テスト実施の時点でアレコレ機能が足りないとか、動作がおかしいとか、依存関係が間違えているとか、そもそも機能が自体がおかしい (要件を満たしていない) とか
- そんな経験はありませんか？

はじめに

- そんな経験をたくさんしてきました (私はw)
- プロダクト (サービス) の中核をなす機能性で、実環境で動作させているとき、バグ・仕様変更などあったら、はてしなくしんどいことになる。(精神的にも、肉体的にも)
- 最終的に根性論とかになる
  - そして、疲弊して品質はさがり、無駄にリリースが延びていいこと一つもない
- そんなツラタンなことを防ぎたい。
  - 三方ヨシっ！な世界になるプロダクト・サービスを作りたい

テスト・QAエンジニアの役割

- 作成された設計書、プロダクト(サービス) が期待する (設計書どおり) に動作することを確認することでしょうか？
  - それも、大事なことかもしれません。(これが基本だと思います)

- 『機能性』としては、それだけではなく、テスト・QA エンジニアの役割として以下のようなことも大事なことの一つだと考えています。(特にテストエンジニア)
  - 保証すべき (機能の) 品質を定義・実証
  - 暗黙知となっている仕様の可視化
  - 予期される例外の (パターンを) 定義
  - リスクの把握・可視化

テスト･QAエンジニアの役割

- テスト･QA エンジニアは、早期警戒機･(対潜) 哨戒機の役割も担っているものだと思っています。
  - ※) 詳しい話はテストの街「葛飾」でします

テスト分析･設計の時点で、発見しだいバグを掃討していくので対潜哨戒機に近いかも。

テスト・QAエンジニアの役割

- 品質を実証する後工程ではなく、分析・設計時点でバグ・仕様漏れを見つけるので、「テスト実施」する前からすでに「バグ」はつぶされていきます。
- つまり。。。。
- (テスト分析・設計中に)「バグだ！」と心の中で思ったならッ、
  その時スデに行動 (テスト) は終わっているんだッ!
- というわけです。
- 機能性のバグ・仕様漏れの発見はテスト設計でほぼ見つけます。(私の場合ですが)

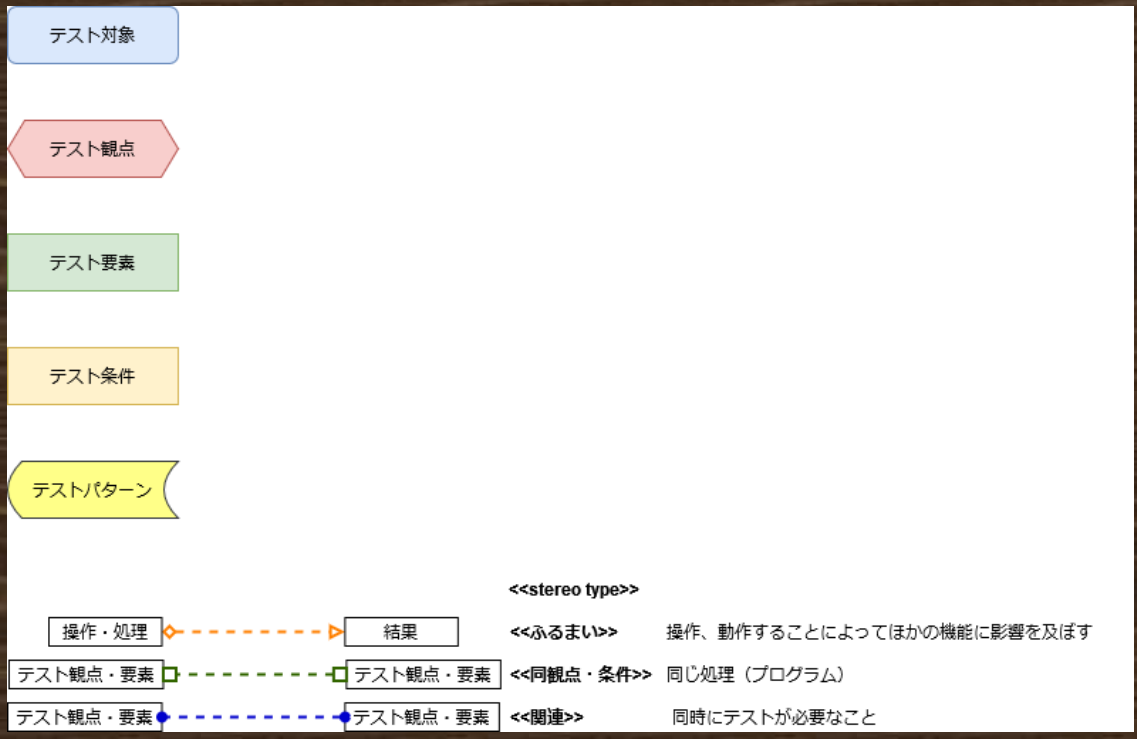
機能性のテスト設計

- テスト設計をするうえで VSTeP というテストフレームワークを利用しています。
- VSTeP のすべてを話すのは時間が足りないため、「テストアーキテクチャ設計」についてさわりだけ話します

- プロダクト (サービス)の機能を分析し可視化するのに使うことができます。
  - ※) 機能性だけではなく様々な用途に利用できます。

機能性のテスト設計

- テストアーキテクチャ設計をする上で、NGT 表記を利用し MECE になるように機能性を分析しながら設計をしています。
  - 分類は以下の5つとしています。
    - テスト対象 (機能)
    - テスト観点 (品質特性)
    - テスト要素
    - テスト条件
    - テストパターン



  - ※) 用語は組織・チームで異なる場合があるため、定義していることが重要です。

違和感しかない設計

- 私が出会った違和感しかない設計例の一部を見せます。

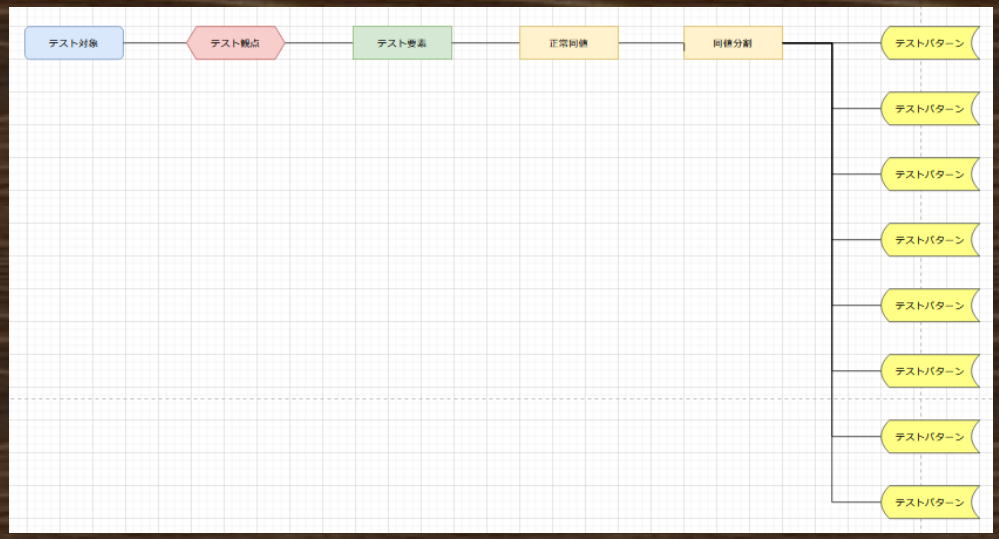
違和感しかない設計

- まず最初に、私が『独り言』で、よく使う用語の説明をします。
  - アンノーン (なんか違和感があるなぁ。。。コレ)
  - ボギー (マヂで違和感しかない、なんじゃこれ？？)
  - バンディット (これバグ・仕様漏れだろw)

- 「ボギー出現、分析を開始」とか
  「バンデットと判定」とかいうと、
  開発の人が振り向くw

違和感しかない設計

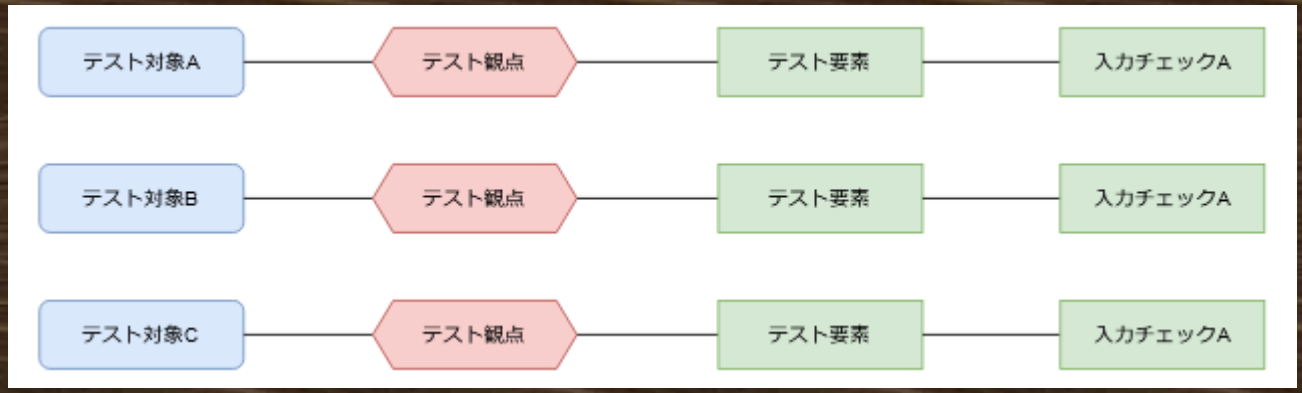
- テストパターンがやたら多い



- アンノーンを補足
  - テスト要素 (やっていること) が大きすぎる可能性
    - 要素を分離しないとテスト漏れが発生
    - 1つのクラス (関数) で複数の仕事させんな (SOLID の単一責任の原則に違反の可能性)

違和感しかない設計

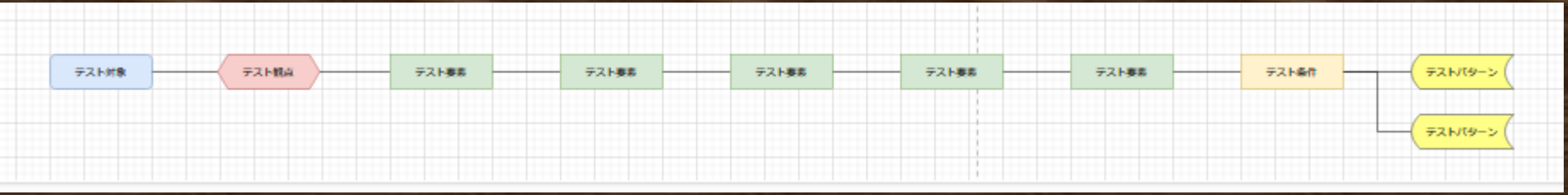
- 同じ(ような) テスト要素が乱立している



- ボギーを補足
  - コピペして増やしている可能性大 (修正漏れがいずれかにある)
    - 電探には正常と欺瞞する可能性
    - というか、ドッペルゲンガー作るなw (DRY: Don't Repeat YourSelf に反している)
    - どれをテストしたかわらんくなるやろ

違和感しかない設計

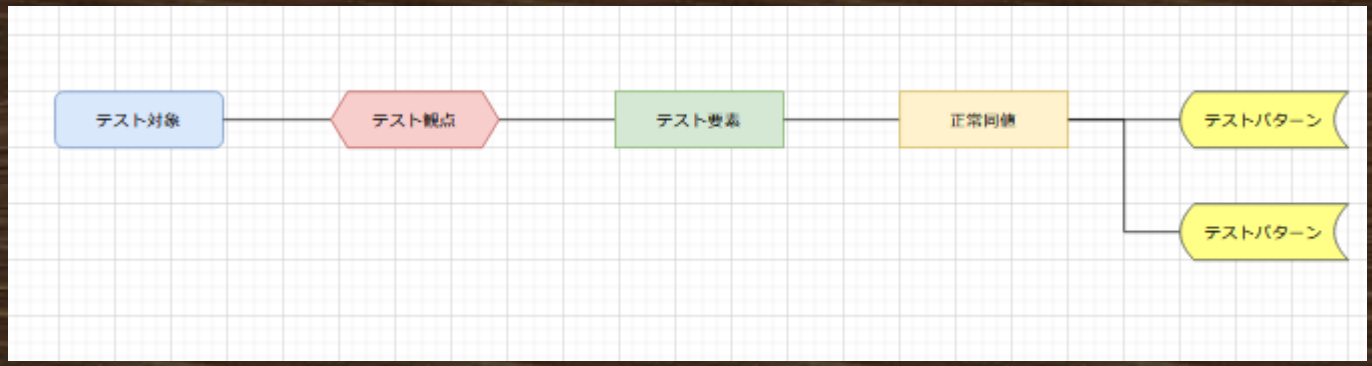
- 要素がやたらとネストしている (依存性が高すぎ)



- アンノーンを確認
  - 要素の依存が多く伝言ゲームになり、テストがしにくい(パターンが多すぎになる問題)
- ボギーと判定
  - 要素の依存が多いのに、なぜかテストパターンが少なすぎ

違和感しかない設計

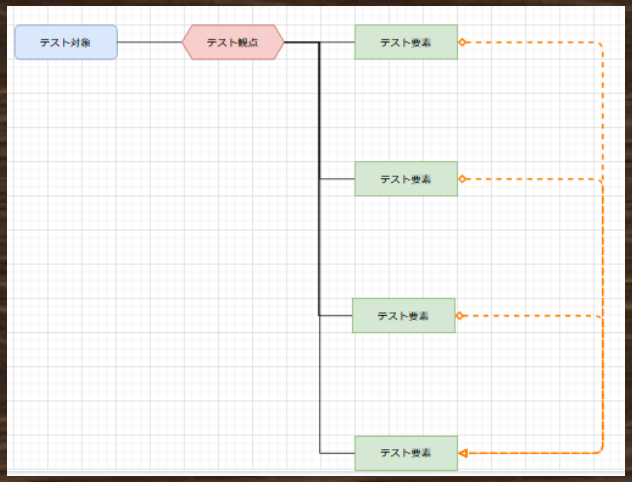
- 正常同値しかない



- バンデット！
  - 異常同値が考慮されていない
  - 考慮されているかもしれないが全てcatch されて捨てられている可能性が大 (すべてヌルポで捨てるとかヤメレ)

違和感しかない設計

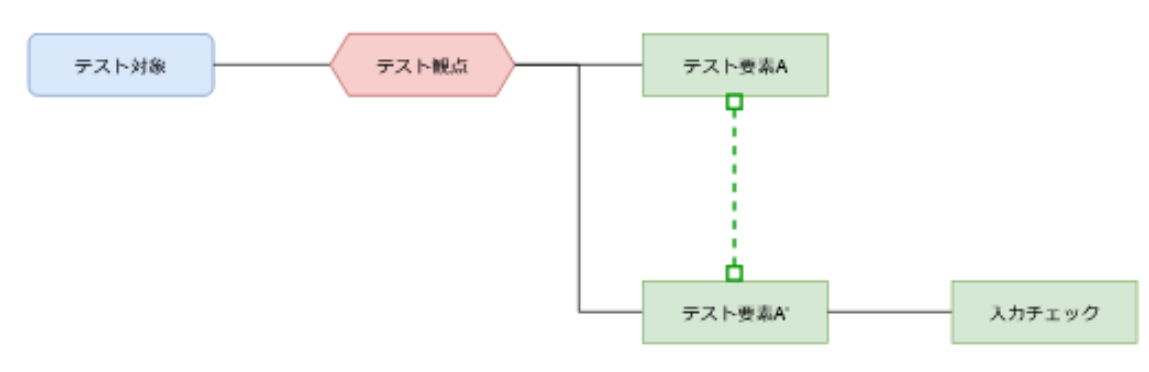
- 複数の要素での結果を1つの要素に入れているのに入力チェックがない



- バンディットと認識し設計を確認せよ
  - 複数からの入力同期などの確認が必要（NULL チェックとか)
  - もし、要素がクラスで親子継承していればバンディット確定
    - 派生開発でよくあるのが、知らぬうちに得体のしれない親を継承している

違和感しかない設計

- 関連している(継承されている) 要素でデータを制限としている



- アンノーンだけど、将来ボギーに変わる可能性大
  - 事前条件を末端で制限している可能性 (リスコフの置換原則に違反)
    - 『事前条件を派生型で強めることはできない。派生型では同じか弱められる。』
  - 仕様が変わった瞬間、修正漏れでバグが埋まる
  - ※) 仕様はコロコロかわるものを認識せよ

まとめ

- テスト・QA エンジニアが「設計」のバグ・仕様漏れなどを (出来る限り) 早期&正確に把握し警報を出して未然に防ぐことも重要なことです。

まとめ

- 機能性のバグ・仕様漏れを発見するためには、テスト・QA 担当は開発担当者と協力して、機能を構造化する必要があります。

- そのために。。。
  - 幅広いコンピュータシステム (OS、ネットワークなど)、情報処理技術、ミドルウェアの基礎知識は必要です。
  - 最新のテクノロジーの知識もある程度必要です。
  - 開発の設計・プログラムの手法(技法) も知っている必要もあります。
    - 歴戦錬磨の職人プログラマーのように美しいプログラムを創造する必要はありませんが作ったものを理解できる (読める) 必要があります。
  - そして、ドメイン (業務プロセス) 知識 は絶対に必要です。
  - もちろん、テスト技法を使いこなせるスキルは必要です。
    - 構造化・可視化して開発担当者に論理的に説明できる必要があります。

おまけ

- ライトニングトークのため表面しか話していませんので、完全版はどこかで話せるかもしれません
- 要求・要件を満たせているかのテスト設計もありますが、別でお話させていただきます。

詳しくは、テストの街「葛飾」で！

テストの街「葛飾」？

- グループ：https://ost-zatu.connpass.com/
- 通称　　：葛飾テストの会
- イベント
  - 毎週火曜日 21:30 ～ 23:00
  - テスト関係の話を中心に雑談（なんでもあり）

- 勉強会やセミナーが渋谷や新宿が多くて東側少ないよね、ないなら自分たちで作っちゃえで始まった会

- キャベツは「中野甘藍」（葛飾の野菜生産量　第二位）
  - 第一位は「こまつな」だが、葛飾区の南にある某区によってゆるキャラ出ているから却下
- 虫はバグ、早急に発見・対処しないと大変なことに。。。

◈ ご清聴ありがとうございました

テストの街

葛飾