ASTER Association of Software Test Engineering

TEST DESIGN COMPETITION

テスト設計チュートリアル

ちびこん編 2024

坂 静香（ASTER/テスト設計コンテスト審査委員会）

このチュートリアルについて

想定している聴講者

- テストケースは作成したことはあるけれども、テスト設計でなにをしたらよいか、よくわかっていない
- テスト設計の勉強をしたいと思っている
- 過去のテスト設計チュートリアルを見ても難しくてよくわからなかった・・・
- テスト設計に対して特に悩みはないけれども、なんか参考になるかも？と思っている
- テスコンU-30に挑戦しようかなぁ・・・と思っている

このチュートリアルについて

話すこと

- テスト設計の各プロセスの概要
- いまよりも納得がいくテスト設計にするためのヒント

話さないこと

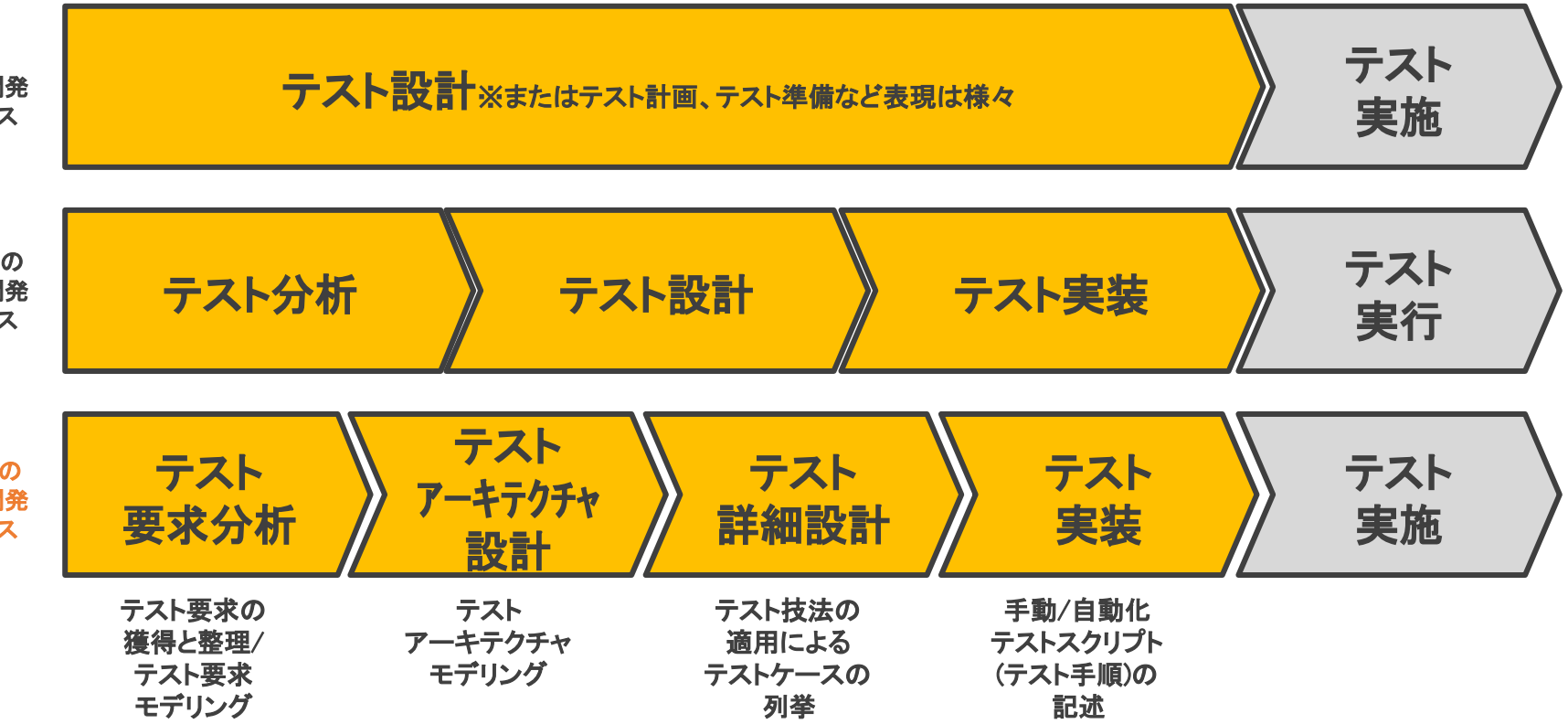
- ソフトウェアテストとはなにか？などの、基礎知識
- テスト技法の種類や、技法の使いかた
- テストケースなど、成果物のつくりかた

わからないところは、お気軽にご質問ください。
後日公開される動画を、先輩や同僚と一緒に
わいわい語りながら観てもよいかもしれません。

おしながき

- 準備：情報を集めよう
- テスト要求分析：テストのスコープとテスト対象を理解しよう
- テスト要求分析：テスト観点を出してみよう
- テストアーキテクチャ設計：観点をもとに全体像を示してみよう
- テストアーキテクチャ設計：テストケースの骨組みを構成してみよう
- テスト詳細設計：これまで作成してきた成果物を利用してテストケースを作成しよう
- テスト実装：テスト実行効率と保守性を考慮してテストスクリプトを作成しよう
- テストアプローチ：限られた時間の中で狙いを定めたテストを考えよう

前提：テスト開発プロセス



引用元：テスト設計チュートリアル ちびこん編 ‘21 資料　67ページ

前提：用語

テスト観点	テストで確認すべきこと （例：テスト条件・テストパラメータや、それを抽象化したもの）
テストアーキテクチャ	テストの全体像を、テストの構成要素とその関係性、連携の段取りで表現したもの
テストコンテナ	テスト観点やテストフレームをまとめたもの
テストフレーム	テストケースの構造。テストケースを構成する複数のテスト観点
テストパラメータ	テスト観点の最も具体的なもの
テスト値	テストパラメータが取る具体的な値 （数値や経路などテスト値の集合でテストパラメータを網羅する）
テストデータ	テスト値をテスト手順として実装する際に実際のデータとして定義されるインスタンス

その他は基本的にJSTQB/ISTQBの用語定義に従います
https://glossary.istqb.org/ja_JP/search

前提：紹介している例

本スライドに掲載している成果物例は「こうすれば必ずうまくいく」という例ではなく「こうしたらよいかもしれない」という例に過ぎません。

完全無欠な正解例でもありません。

大事なことは、みなさんが「自分たちが納得できる成果物を目指すためにどうしたらよいだろう？」と考え、試して、自分たちに適したやりかたで実践して、自分たちなりの成果物を作り上げることです。

本講演での紹介例が、テスト設計に挑戦するための材料になりましたら幸いです。

「正解」よりも「納得」を目指そう！

プチ演習：テストケースを考えてみよう

次の文章を読んで、必要なテストケースをあげてみよう

- このプログラムは、入力ダイアログから３つの整数を読む
- この３つの値は、それぞれ三角形の三辺の長さを表すものとする
- プログラムは、三角形が不等辺三角形、二等辺三角形、正三角形のうちのどれであるかを示すメッセージを表示する

Glenford J. Myers著「ソフトウェア・テストの技法第2版」P-2より引用

回答例

1. 有効な不等辺三角形
2. 有効な正三角形
3. 有効な二等辺三角形
4. 有効な二等辺三角形の際の三種類の辺の組み合わせ
5. 一つの辺がゼロ
6. 一つの辺が負の値
7. 二辺の和がもう一辺と等しい
8. 二辺の和がもう一辺と等しい際の三種類の辺の組み合わせ
9. 二辺の和がもう一辺より小さい
10. 二辺の和がもう一辺より小さい際の三種類の辺の組み合わせ
11. 三辺がゼロ
12. 整数でない辺
13. 辺の数が三つ以外
14. 各ケースごとに、入力値に対する出力値を示している

Glenford J. Myers著「ソフトウェア・テストの技法第2版」P-3を基に要約

情報を追加してみる

このUIを追加提示されたうえで、テストケースを考えると、
なにか違うケースを考えるだろうか？

三角形判定アプリ

3つの数字を入力してください：

4

4

4

判定結果を確認する

正三角形です。

情報を追加してみる

次の条件が付加されると、
テストケースはどう変わるだろうか？

- 「三角形判定アプリ」は、
  スマホアプリである
- Android・iOSに対応する
- ソフトウェアテストを学ぶ人に
  いろいろ触ってもらうことを
  目的として無料公開する
- テスト実行時間は1分とする

準備：テストケースを考える前に情報を集めよう

「テストで確認すること」は、得た情報に影響される

- UIが示されることで、UIならではの確認事項を意識しやすくなる
- 動作環境が示されることで、互換性や性能などの確認事項を意識しやすくなる
- 利用目的が示されることで、確認事項の必要性が変わってくる

仕様書に記載されていることがすべてではない

- 仕様書に記載しきれない情報はたくさんある

テストで確認したいことを考えるために、テスト対象（プロダクト）を知る必要がある

- プロダクトが提供する価値（なぜ開発する必要があるか）
- プロダクトに使われる技術（実現手段）
- プロダクトの構成

テストでなにをどこまで確認するかを決めるために、プロジェクトの背景を知る必要がある

- このプロジェクトの制約はなにか
- 自分たちの責務の範囲はどこまでか

準備：情報を幅広く集めよう

- まずはドキュメントを集める
  - 仕様書・設計書
  - 要件定義書や顧客要求資料
  - テスト計画書やプロジェクト計画書
- ドキュメント以外の情報も集める
  - 会議議事録
  - メールのやりとり
  - チケットやwiki
  - プロダクトが関わる業界の知識
  - プロダクトそのものを熟知している人の経験談

「この情報は自分に関係ないだろう」と思い込まずに、集めてみる

この先テスト設計を考えるときに使う「引き出し」を用意する

準備：（例）テスコンU-30の場合

テストベースとして提供されているもの

- 要求仕様書
  「割り勘支援アプリ Warikan仕様書」
  https://www.aster.or.jp/testcontest/doc/warikan_specification_2024.pdf
- テストプロジェクト要求補足書
  プロジェクトの背景やテストチームの責務などを記載したもの
  https://www.aster.or.jp/testcontest/doc/aster_u30_tdc_supplement_2024.pdf

この他の情報は推測するしかない

- 推測で進めることにより、無駄なテストを実施するなどのリスクが発生する
- 実際の開発現場では、推測を抑えられるように情報を求めることが望まれる

足りない情報を導き出すために、推測することも必要である

- いきなり情報提供依頼しても、機密などの理由で簡単には情報提供されないこともある→検証が必要なことを伝えられるように仮説を示そう
- テスコンで推測することを体験してみよう

テスト要求分析

テスト要求分析：テストのスコープを理解しよう

テストレベル

具体的にインスタンス化したテストプロセス。

（ISTQB用語集より引用）

テストタイプ

コンポーネントやシステムのある特性に対応したテストの目的を基にテスト活動をまとめたもの。

（ISTQB用語集より引用）

テストレベル・テストタイプについて、自組織・自プロジェクトでの定義を明確にする

- マスターテスト計画書（プロジェクト全体における複数のテストレベルやテストタイプとそれらの関係性を示した計画書）を確認しよう
- なければ、マスターテスト計画書を作って、テストレベルやテストタイプを定義しよう

テスト要求分析：テストのスコープを理解しよう

自分たちの責務範囲を明確にする

- どのテストレベルを担当するのか
- 担当するテストレベルの中でどのテストタイプを担当するのか

他のテストレベル・テストタイプの担当者と、スコープの調整をする

- どこまでを誰が担当するのか
- 自分たちはどこに着目する必要があるのか
- 調整をするために、自分たちの責務以外のテストにも関心を持とう

テスト要求分析：テスト対象を理解しよう

仕様書を見るだけでテスト対象を理解できるだろうか？

- ドキュメントを黙読する
- 声に出して読む・書き写す
- ドキュメントに線を引く、メモを書く
- ドキュメントの記述を要約する
- ドキュメントの内容を図や絵にする
- 自分の言葉で書き直す・説明する

↓にいくほど、
仕様の理解度は増す

より理解度が増す行動をとろう

3色ボールペン法などを活用する

ここらへんに挑戦してみる
テスト技法を活用してみる
その他自分に合う手段で表現してみる

準備段階で集めた情報も活用する

テスト要求分析：テスト対象を理解しよう

ドキュメントを読むだけで理解することは難しい

申請作成者が申請作成を開始すると、作成中という状態になる。申請作成者は作成を終えたら、申請ボタンを押す。申請ボタンの代わりに一時保存ボタンを押すかEscキーを押すと一時保存処理が実行されるが、状態は変わらず作成中のままである。

申請ボタンを押した時点での予算残高に比べて申請額が高額の場合は所属部門部長承認待ちとなる。予算残高に比べて申請額が同額か少額の場合は所属部門の課長承認待ちとなる。所属部門の課長または所属部門の部長が所属部門承認ボタンを押すと、経理部課長であるAさんまたはBさんによる承認待ちとなる。

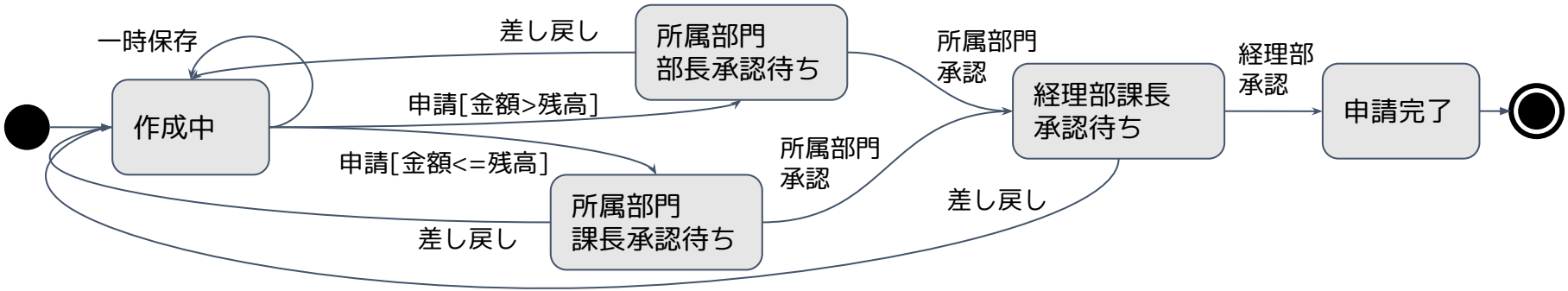
AさんまたはBさんが経理部承認ボタンを押すと申請が完了状態となる。

各承認者は、申請に不備を見つけた場合、差し戻しボタンを押す。差し戻した場合、通常は1つ前の状態に戻る。ただし、Aさんが差し戻したとき、または、Bさんが差し戻したときは常に申請者宛となるため、作成中となる。

引用元：JaSST'23Tokyo テスト設計コンテストU-30クラス セッション 資料 57ページ

テスト要求分析：テスト対象を理解しよう

状態遷移図・状態遷移表を使って理解度を上げる（例）

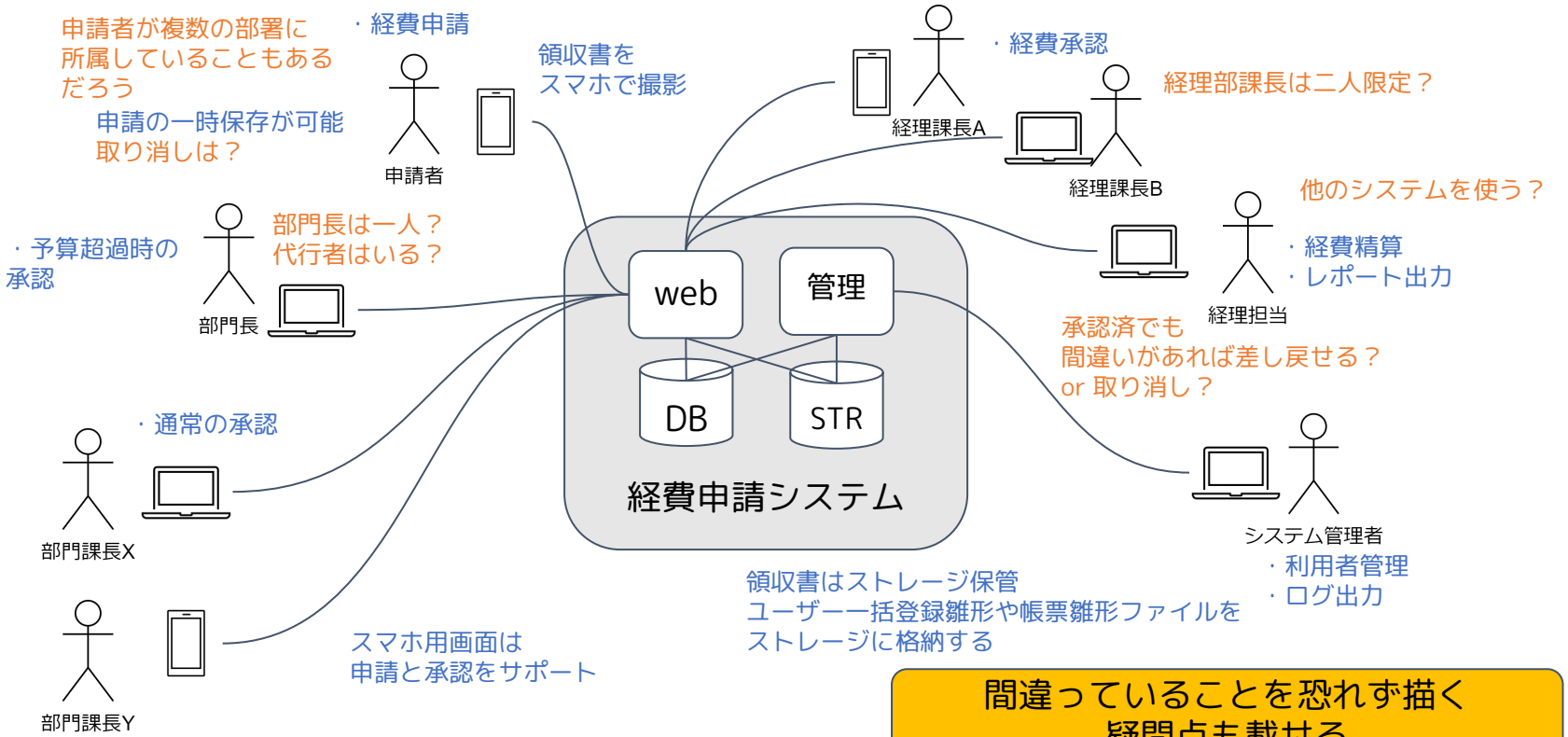


状態 / イベント	作成中	所属先部門課長承認待ち	所属部門部長承認待ち	経理部課長承認待ち	申請完了
申請[金額<=残高]	所属部門課長承認待ち	N/A	N/A	N/A	N/A
申請[金額>残高]	所属部門部長承認待ち	N/A	N/A	N/A	N/A
一時保存	作成中	N/A	N/A	N/A	N/A
所属部門承認	N/A	経理部課長承認待ち	経理部課長承認待ち	N/A	N/A
経理部承認	N/A	N/A	N/A	申請完了	N/A
差し戻し	N/A	作成中	作成中	作成中	N/A

引用元：JaSST'23Tokyo テスト設計コンテストU-30クラス セッション 資料 58ページ

テスト要求分析：テスト対象を理解しよう

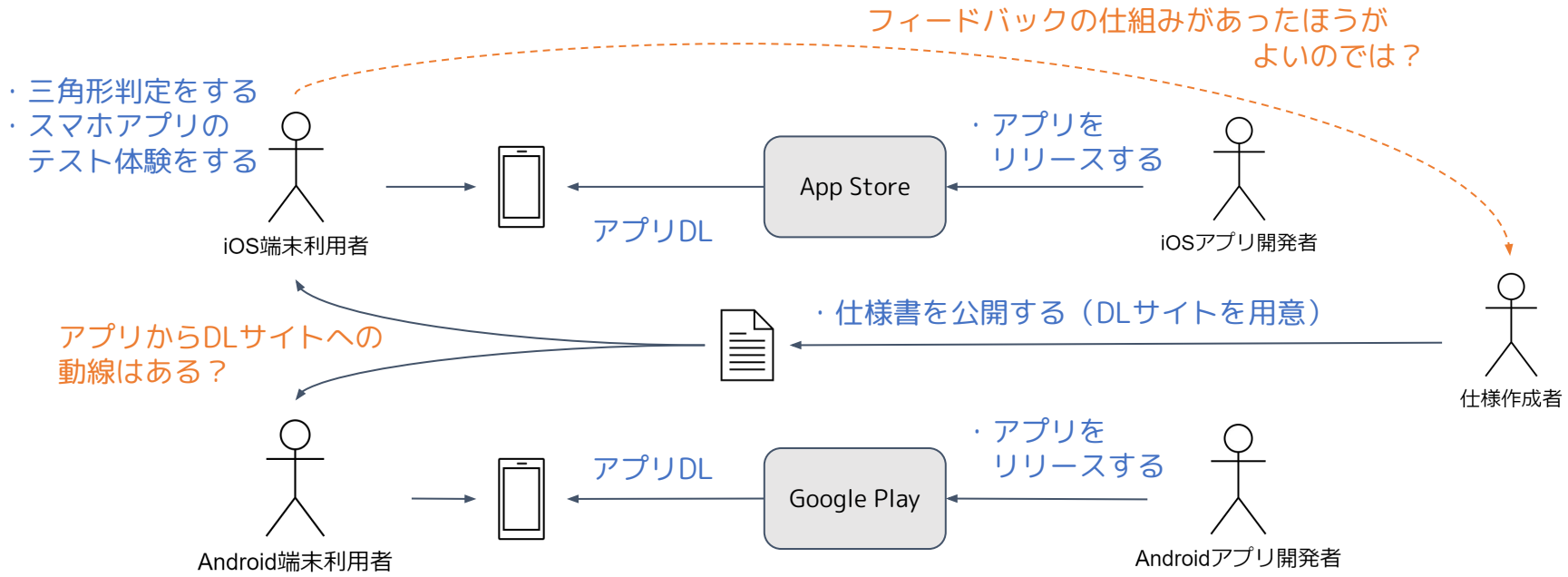
テスト対象と取り巻く環境を描いてみる（例：経費申請）



間違っていることを恐れず描く
疑問点も載せる

テスト要求分析：テスト対象を理解しよう

テスト対象と取り巻く環境を描いてみる（例：三角形アプリ）



対象OSバージョンのほかに、
端末も複数機種・複数メーカー必要そう

気になることを遠慮せず書き込む

テスト要求分析：テスト対象を理解しよう

自分なりに表現してみたら、誰かに見せよう

- 自分だけでは気づかないことが見つかる
- 疑問点が解消される
- 自分の解釈に自信を持てる

見せる段階で正しくないほうが、認識を合わせやすくなる

- きっかけがあるほうが相手の思考が働きやすくなる
- 指摘内容を説明することで別の課題に気づきやすくなる

情報量に気を付けよう

- 相手が短い時間で受け取れる量には限界がある
- 情報量に圧倒されると見る気が失せる

テスト要求分析：テスト対象を理解しよう

プロダクト開発段階でも図は作成しているであろう

- システム構成図
- アプリ構成図
- 処理フローやシーケンス図
- データモデル
- ホワイトボードに描いた絵や図

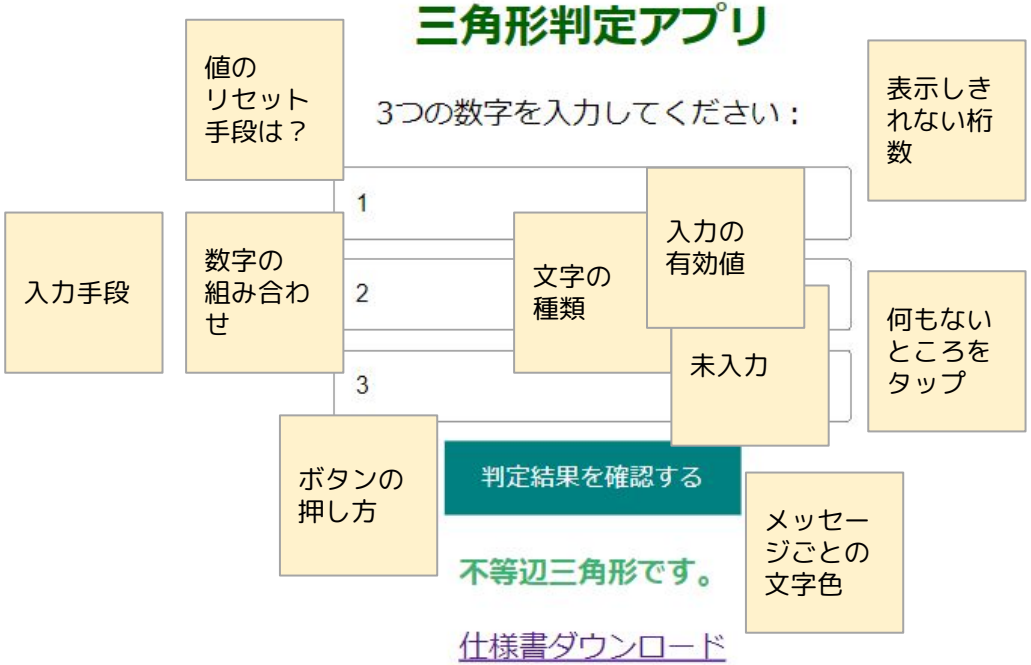
開発者など関係者と図を共有して、理解を促進しよう

すでに存在する成果物を活用しよう

テスト要求分析：テスト観点を出してみよう

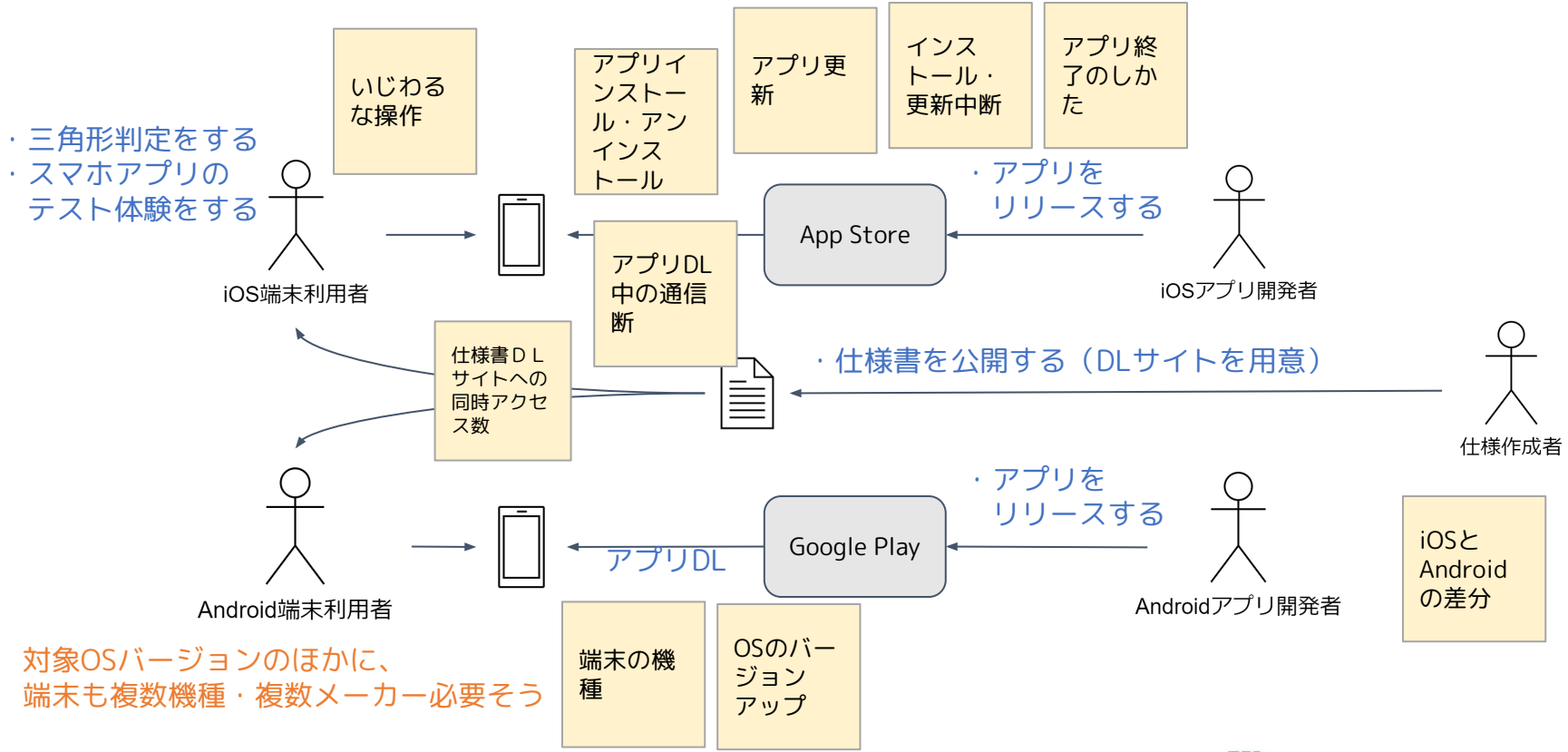
テストで確認したいことや、気になることを出してみよう

UIのイメージから出してみる（例）



テスト要求分析：テスト観点を出してみよう

理解のために作成した成果物を利用する（例）



テスト要求分析：テスト観点を出してみよう

出したものを整理しよう

整理する手段は複数ある

自分たちにはまる手段を選ぼう

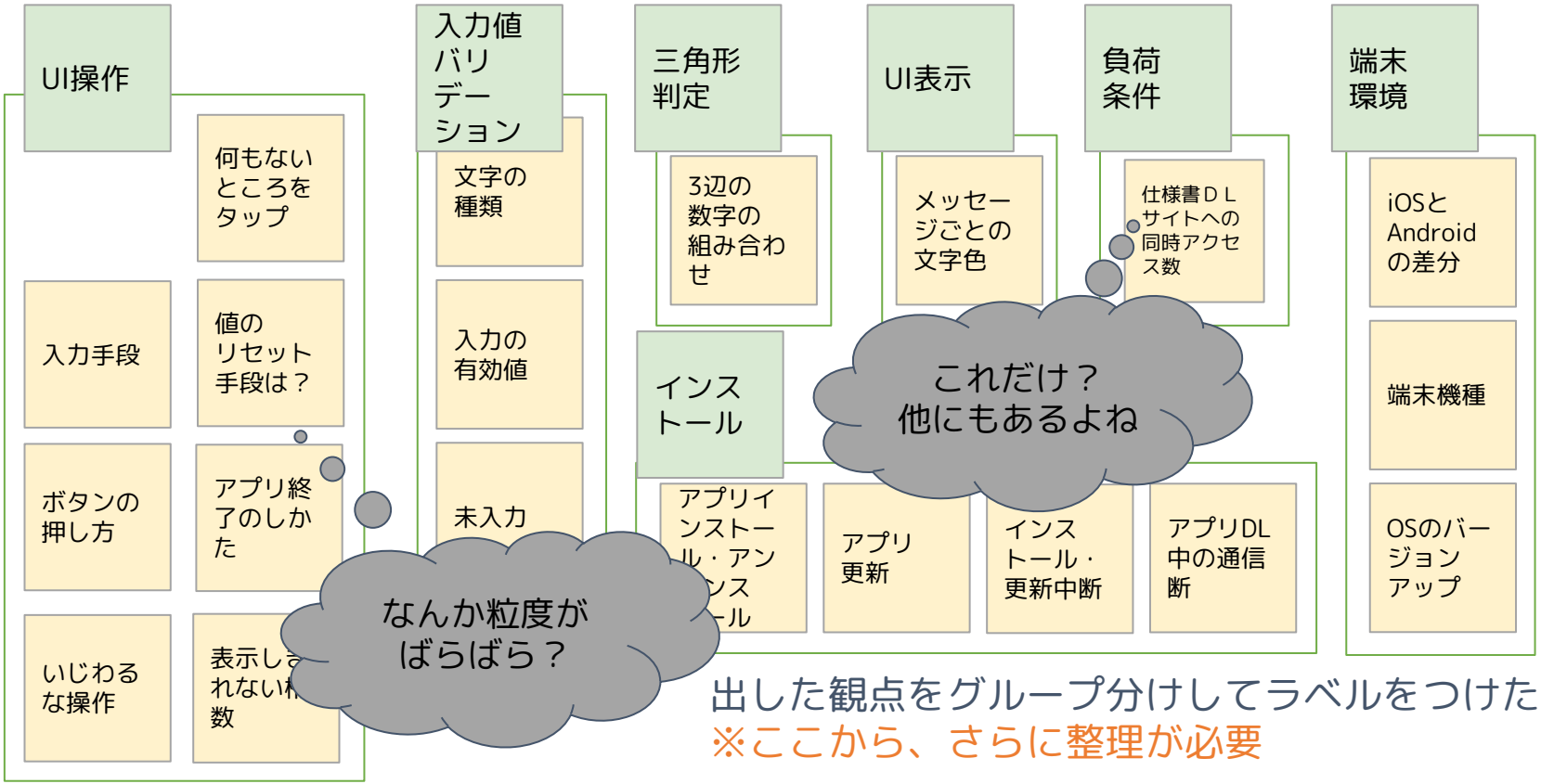
- グループ分けしてラベルをつける
- マインドマップで整理する
- フレームを用意して当てはめる

など

一度で完全なものができるわけではない

テスト要求分析の過程でも、その先のテスト詳細設計に至るまでの過程でも、必要に応じて育てていく

テスト要求分析：テスト観点を出してみよう

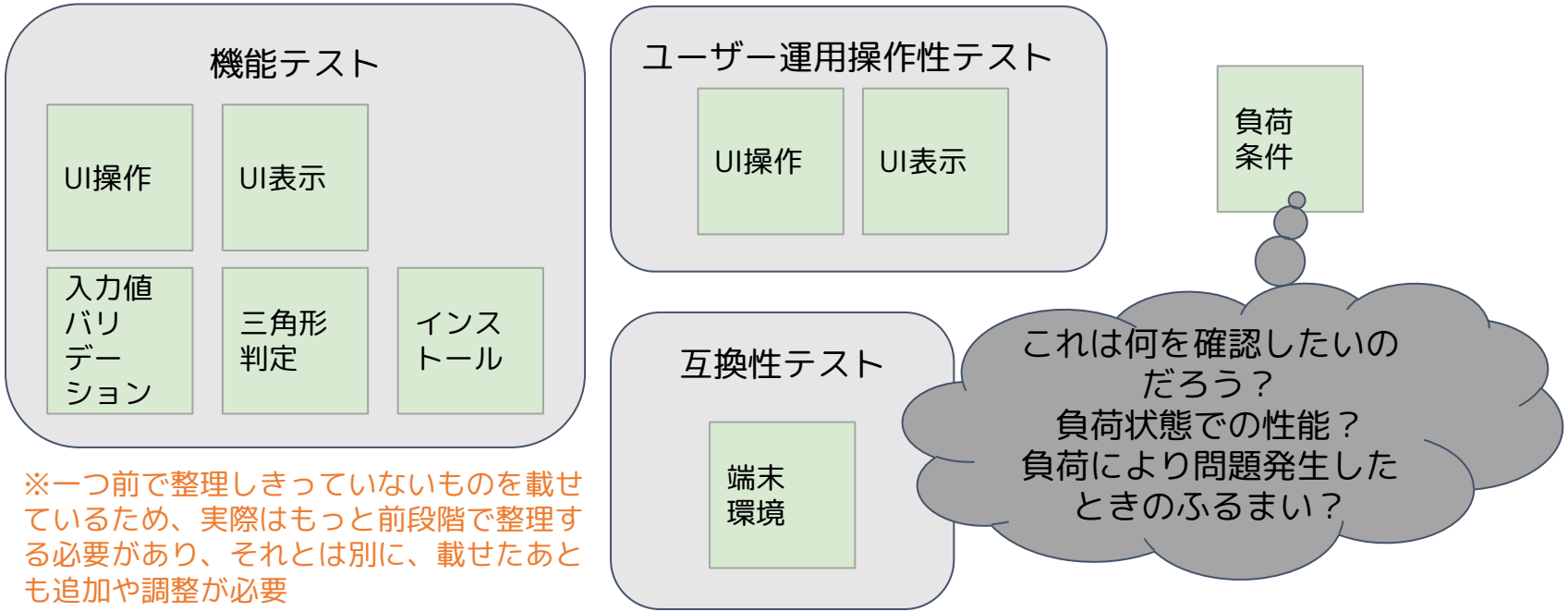


出した観点をグループ分けしてラベルをつけた例

※ここから、さらに整理が必要

テスト要求分析：テスト観点を出してみよう

テストタイプに当てはめてみよう
（観点のグループをテストタイプに割り当てた例）



※一つ前で整理しきっていないものを載せているため、実際はもっと前段階で整理する必要があり、それとは別に、載せたあとも追加や調整が必要

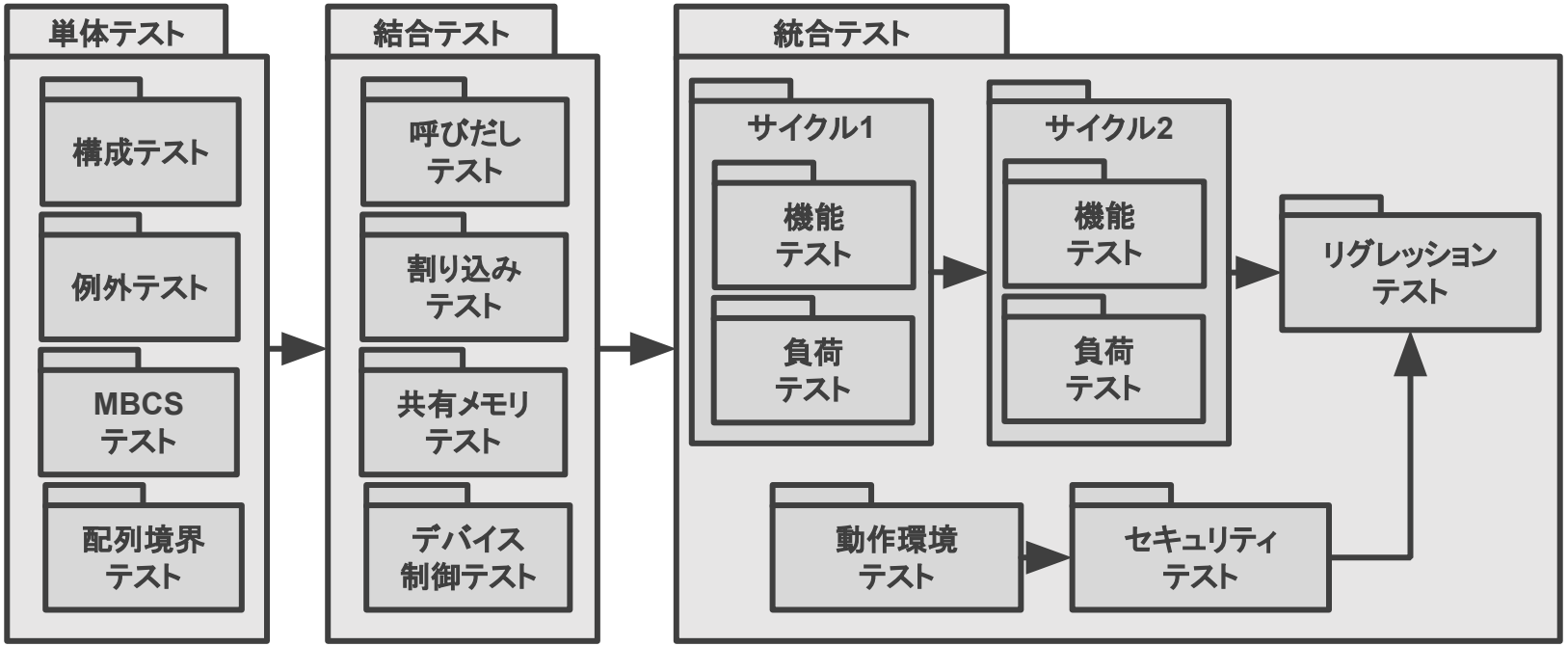
テストアーキテクチャ設計

テストアーキテクチャ設計：全体像を示してみよう

テストのスコープに対して、どんなテストをするのかを、
少しだけ具体的に見せてみよう（イメージ）

テストアーキテクチャ設計：全体像を示してみよう

テストレベルごとにテストタイプとサイクルを表現した例



※図はあくまで表現方法の一例であることに注意

引用元：テスト設計チュートリアル ちびこん編 ‘21 資料　44ページ

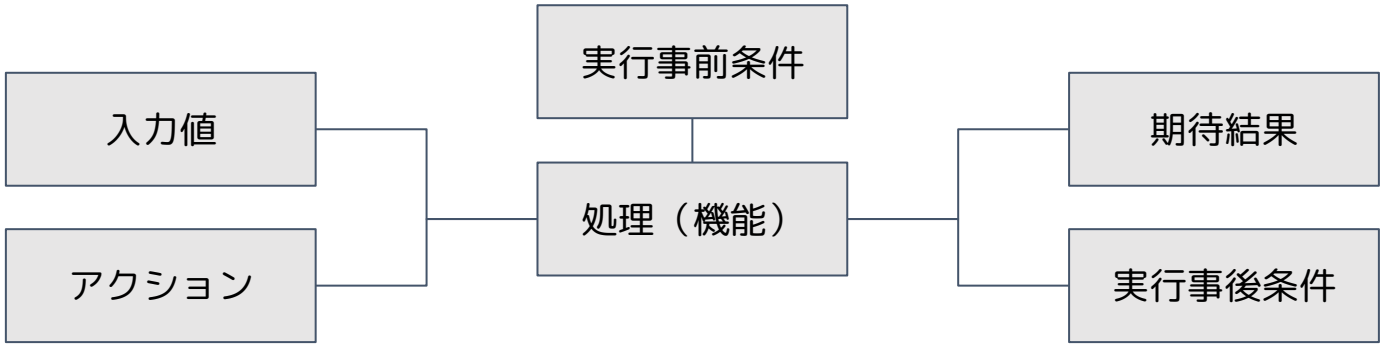
テストアーキテクチャ設計：テストケースの骨組みを作ろう

テストケース

実行事前条件、入力値、アクション（適用可能な場合）、期待結果、および実行事後条件のセットであり、テスト条件に基づいて開発されたもの。

（ISTQB用語集より引用）

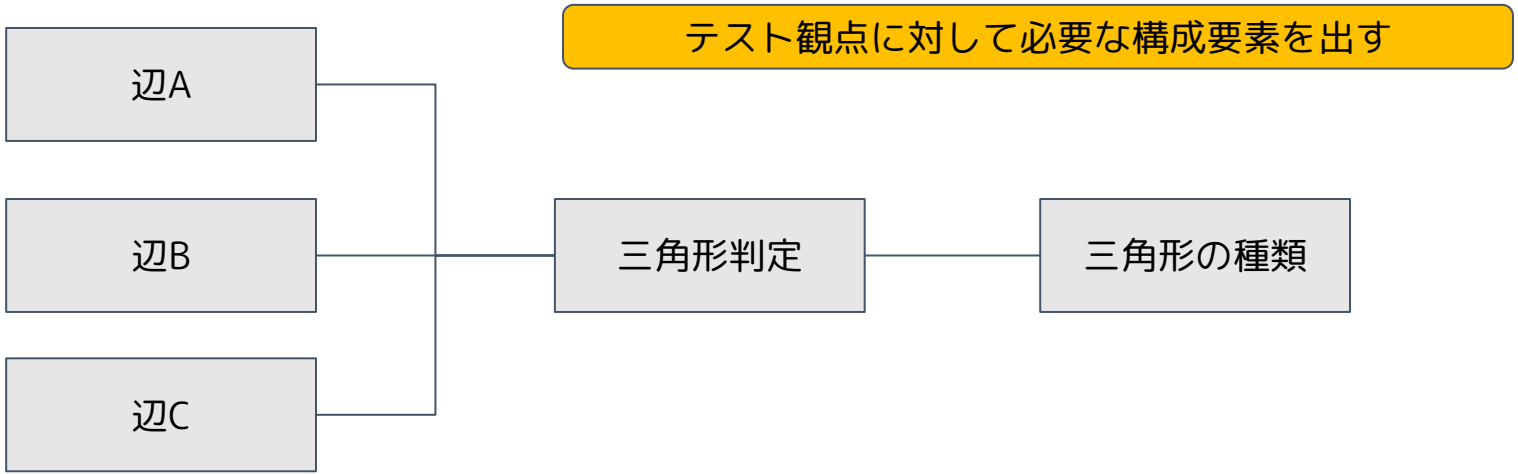
テストケースの構成を示してみる（一例）



テストアーキテクチャ設計：テストケースの骨組みを作ろう

例：三角形判定アプリ

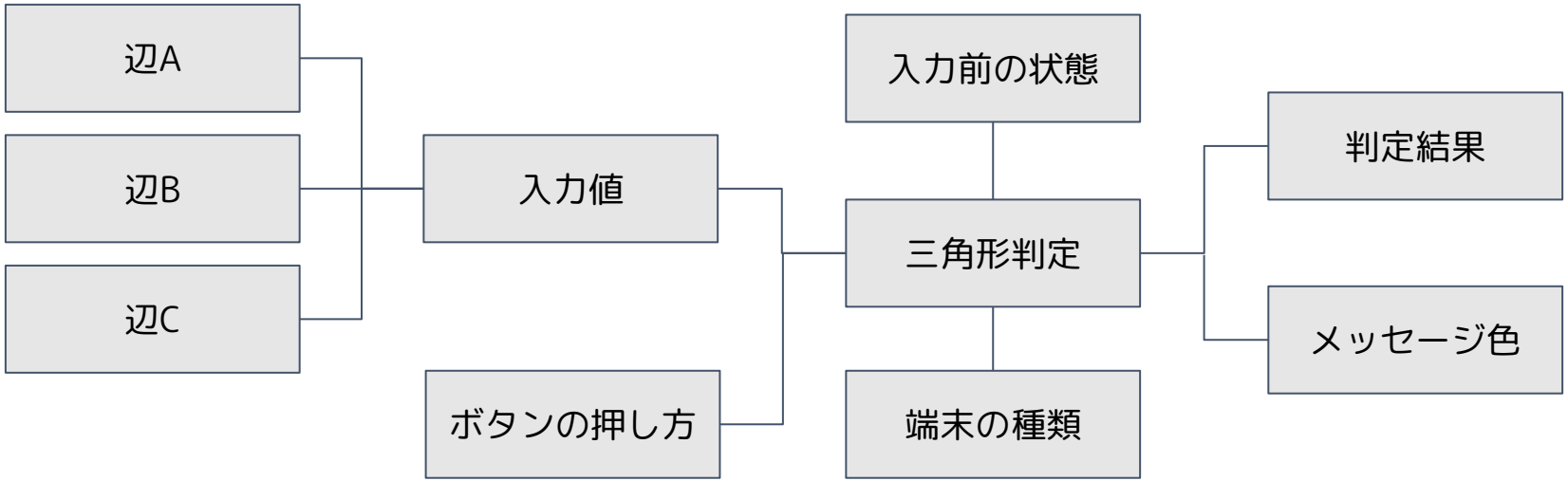
テストで確認したいこと 「三角形判定」	三つの整数を入力することで三角形の種類を適切に判定すること



テストアーキテクチャ設計：テストケースの骨組みを作ろう

複数の観点を含めると骨組みは複雑になる
（ケース導出難易度が上がる）

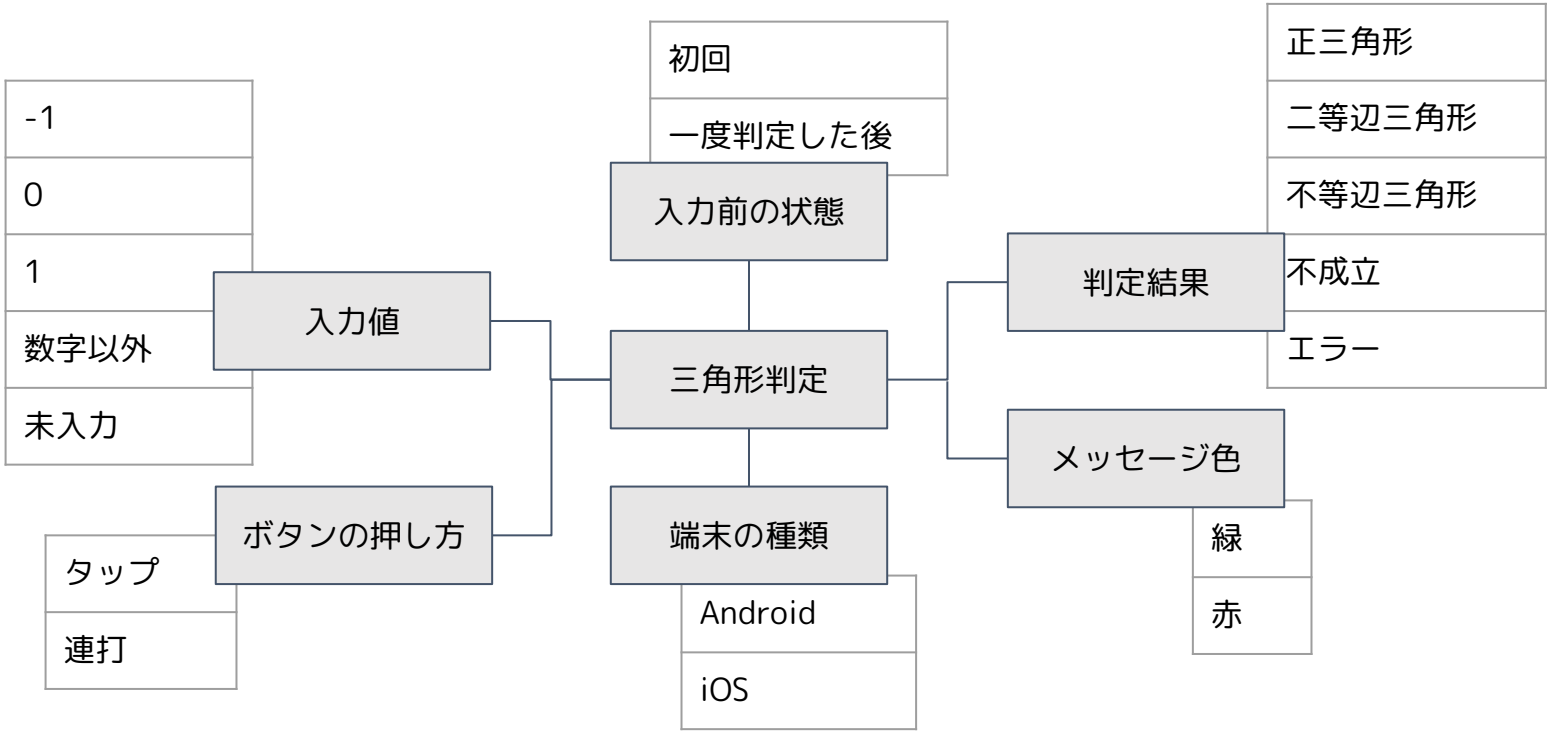
初めはシンプルな骨組みを目指そう



テスト詳細設計

テスト詳細設計：成果物を生かしてテストケースを書こう

テストケースの構成要素に対して「とりうる値」を考える（例）



テスト詳細設計：成果物を生かしてテストケースを書こう

同値分割法（Equivalance Partition）

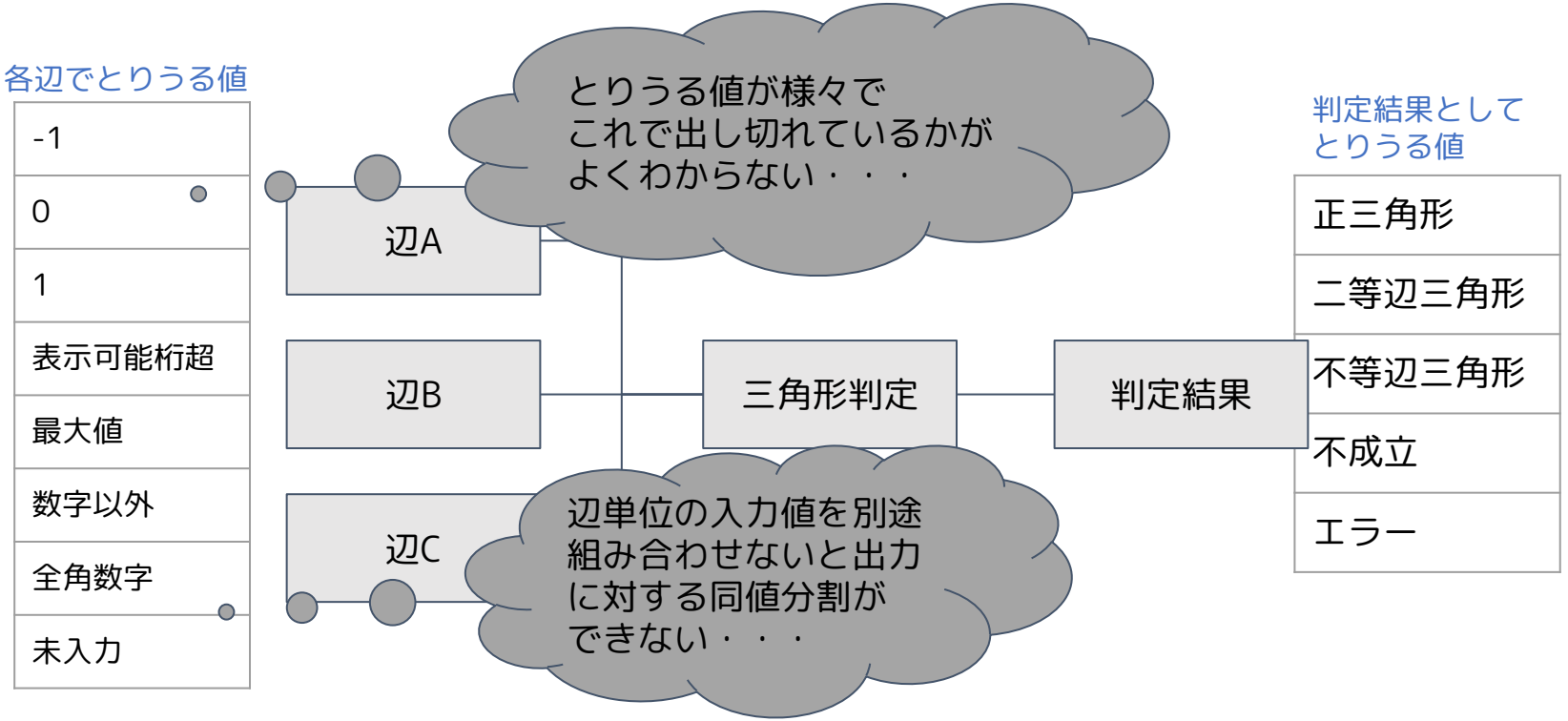
同値分割法(EP)は、**ある特定のパーティションのすべての要素がテスト対象によって同等に処理されることを想定して、データをパーティション（これを同値パーティションと呼ぶ）に分割する。**この技法の背景にある理論は、同値パーティションから1つの値をテストするテストケースが欠陥を検出した場合、この欠陥は同じパーティションから他の値をテストするテストケースでも検出されるはずだというものである。したがって、各パーティションに対して1つのテストがあれば十分である。

同値パーティションは、入力、出力、構成アイテム、内部値、時間関連の値、インターフェースパラメーターなど、テスト対象に関連するあらゆるデータ要素について識別できる。パーティションは、連続または離散、順序性ありまたは順序性なし、有限または無限のいずれでもよい。パーティションは、重複してはならず、空でない集合でなければならない。

JSTQB テスト技術者資格制度 Foundation Levelシラバス Version 2023V4.0.J01　4.2.1　より引用

テスト詳細設計：成果物を生かしてテストケースを書こう

テストケースの構成要素に対して「とりうる値」を考える（例）



テスト詳細設計：成果物を生かしてテストケースを書こう

分割方針を明確にして、分ける（例）

整数の境界値を
意識して分けた

-1
0
1
表示可能桁超
最大値
最大値＋１

文字種類を
意識して分けた

半角数字整数
全角数字
半角16進数字 （英字＋数字）
半角数字小数
半角整数 カンマ入り
半角英記号

入力ボックスの状態を
意識して分けた

未入力
入力
入力を削除

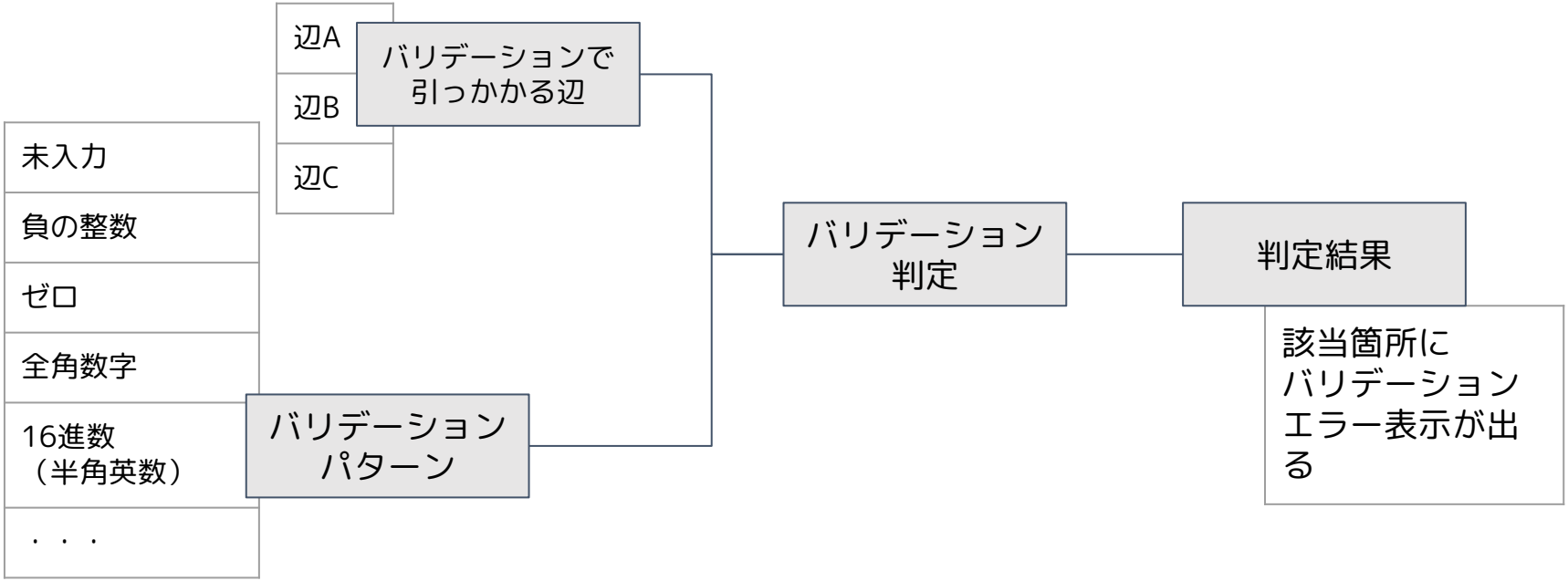
他辺との関係を
意識して分けた

他の2辺と同じ
この辺だけ異なる
他の1辺と同じ
他の辺と異なる
他の2辺の和と等しい
他の2辺の和より大きい

テスト詳細設計：成果物を生かしてテストケースを書こう

観点を意識して、ケース展開しやすい構成要素にする（例）

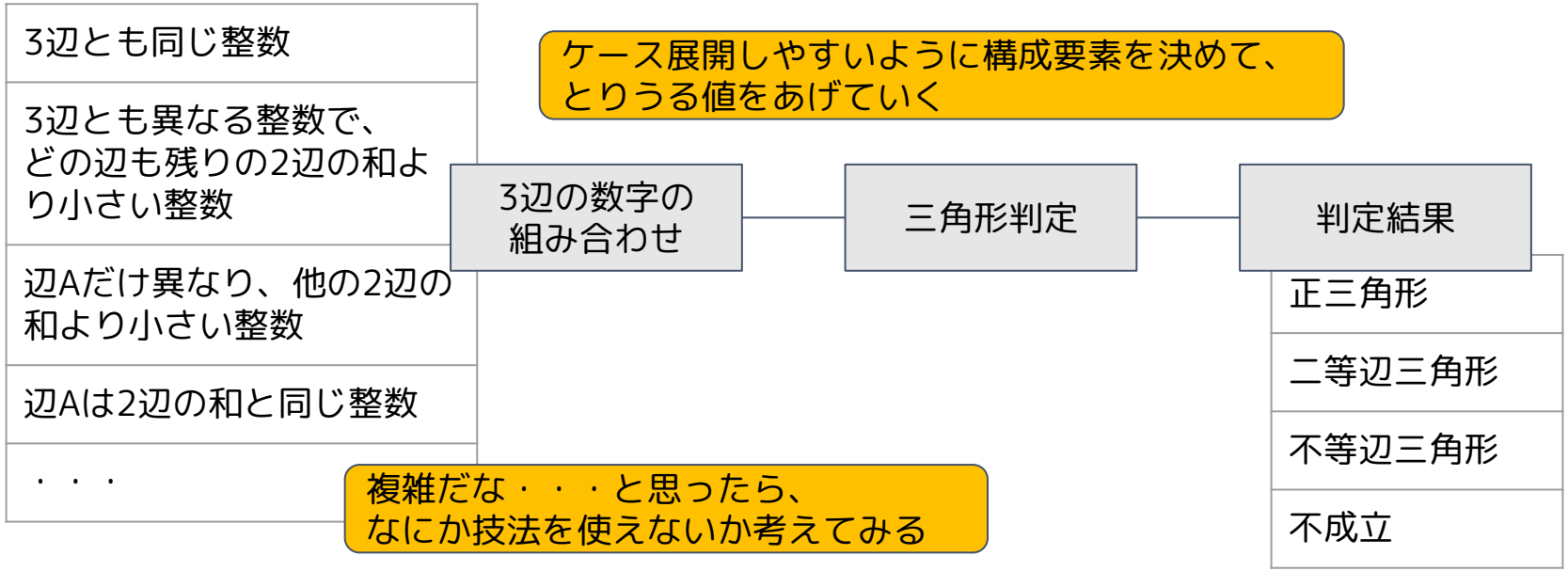
テストで確認したいこと 「入力バリデーション」	三辺のどれか一つでも無効値が入力されたときに バリデーションエラーになること



テスト詳細設計：成果物を生かしてテストケースを書こう

観点を意識して、ケース展開しやすい構成要素にする（例）

テストで確認したいこと「三角形判定」	三つの辺に入力した有効値の組み合わせにより三角形の種類を適切に判定すること



テスト詳細設計：成果物を生かしてテストケースを書こう

テストの骨組みをもとにテストケースに展開する

- 適切なテスト技法で、カバレッジを定めて、ケース展開していく

テスト要求分析で作成したモデルからテストケースを導出する

（例）

- 状態遷移図をもとに、状態遷移表を作成する
- ユースケース図・ユースケース記述から、シナリオ形式のテストケースを作成する

テスト実装

テスト実装：実装効率と保守性を考慮しよう

特に手動テストの場合、テストケースとテスト実装成果物に差がなくなりやすい

よりよいテスト実装にするためにどんな工夫が必要だろうか

テスト実装で考慮すること（例）

- テスト実行のしやすさ
- テスト実行効率
- テスト実装成果物の保守性

テスト実装：テスト実行効率を考慮してみよう

テストの実行順を考慮すべきものがあるだろうか

- 例えば承認回答のテストは承認申請のテストの前に行うのは非効率であろう
- 一覧表示のテストは、一覧に表示されるデータを登録する機能のテストを先にすることで、わざわざデータを投入しなくてすむかもしれない

まとめてテスト実行できるものがあるだろうか

- UIの操作確認と操作による画面遷移と操作によるふるまいの確認は一連でまとめるほうがよいかもしれない（逆に分けたほうがよいかもしれない）

テスト実装：保守性を考慮しよう

手順をコピーペーストで量産すると修正対応が難しくなる

- 修正漏れが起こりやすい
- 手順をケース単位で記載することでケースの表示幅が大きくなり、修正時だけでなく、実行時でも表示に支障が出やすくなる

まとめられるものはまとめよう

- 同じ手順ならシートの上部に手順を載せてケースと分けてもよいかもしれない
- 自動実行であればページオブジェクト作成やデータ駆動・キーワード駆動など、技術を活用しよう
- 実行効率が下がらないように調整しよう

テストスクリプトを複数のテストで活用する

- スモークテストのスクリプトを複数仮想端末から実行して負荷テストに利用できるかもしれない

テストアプローチ

テストアプローチ：狙いを定めたテストを考えよう

いままでよりも、テスト観点を多くだせるようになった！
しかし、この量のテストを設計して実行する時間は、ない！！

さてどうしよう・・・

- プロダクトリスク（プロダクトの品質に影響を与えるリスク）を考慮して、テストの「厚み」を調整する
- 集めた情報から、テスト対象の弱点を推定して狙いを定める

テストアプローチ：狙いを定めたテストを考えよう

テストの厚み調整例

- よく使われる機能であれば、状態遷移や組み合わせテストのカバレッジを上げる
- おまけ機能であれば、最低限の動作確認をテストケースで用意して、テスト実行時間が確保できればハイレベルのテストケースを使って探索的にテスト実行する
- 管理者機能であれば、顧客利用機能よりもUIに対する重要度は下がるため、UIのテストを軽めにする

テスト対象の弱点を狙う例

- 複雑な仕様の箇所を中心にチャーターをつくり、テストケース実行後にさらにセッションベースでモブテストを行う

いずれも、準備段階で集めた情報が活かされる

まとめ

- テスト設計を始める前に情報を集めよう
- テスト対象の理解を深めよう
  そのためにモデルを活用しよう
- 作成した成果物をたたき台にして関係者の認識合わせを積極的に行おう
- テストレベル・テストタイプを把握して全体像を描き、自分たちの責務範囲を明確にしよう
- テストケースの骨組みと構成要素の同値パーティションを明確にしてからテストケースを作成しよう
- テスト要求分析で作成した成果物をテスト詳細設計で活かそう
- テストの実行順番や保守性を意識してテスト実装しよう
- プロダクトリスクやプロジェクトの制約に応じてテストの厚みを調整しよう

参考文献

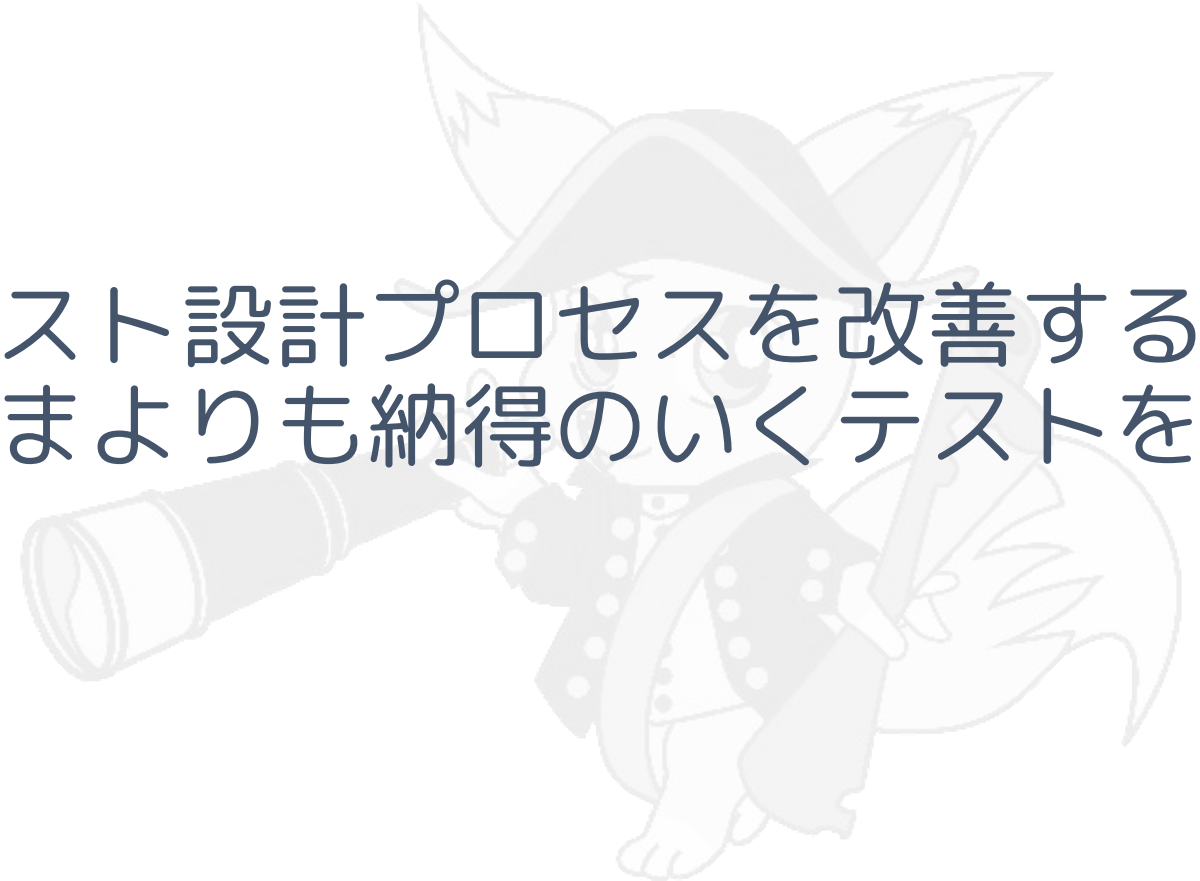
- テスト設計チュートリアル ちびこん編 '21 資料
  - http://www.aster.or.jp/business/contest/doc/2021_chibicon_V1.0.0.pdf
- テスト設計チュートリアル テスコン編 '21 資料
  - http://www.aster.or.jp/business/contest/doc/2021_tescon_V1.0.0.pdf
- ISTQBテスト技術者資格制度　Foundation Level シラバス 日本語版 Version 2023V4.0.J01
  - https://jstqb.jp/dl/JSTQB-SyllabusFoundation_VersionV40.J01.pdf
- ISTQB用語集
  - https://glossary.istqb.org/
- JaSST'23Tokyo テスト設計コンテストU-30クラス セッション 資料
  - https://www.jasst.jp/symposium/jasst23tokyo/pdf/C6.pdf
- テスト設計チュートリアル ちびこん編 '23 資料
  - https://www.aster.or.jp/testcontest/doc/2023_chibicon_V1.0.0.pdf
- Glenford J. Myers著「ソフトウェア・テストの技法第2版」（近代科学社,2006年）

参考動画

テスト設計コンテストチャンネル

https://www.youtube.com/@user-uz8mw1ux4p

- 過去のテスト設計チュートリアル
- JaSST東京のテスト設計コンテストU-30クラスセッション



テスト設計プロセスを改善することでいまよりも納得のいくテストをしよう

テスト設計コンテスト U-30クラス
昨年度の傾向

昨年度（2023）の傾向

- 成果物を作成するにあたり、事前学習をしているチームが目立った
- テスト要求分析でモデル作成や様々な手法を取り入れるチームが目立った
- ↑そのモデルを詳細設計に活かせたら、もっとよくなる
- 仕様に対するフィードバックがしっかりできているチームが多かった
- テスト観点の時点でテストケースと変わらなくなってしまう傾向がある
- テスト要求分析をがんばっているのに、なぜかテストケースの時点で仕様をなぞったテストができあがる傾向がある

予選敗退したチームも含めて、どのチームも挑戦した様が伝わりました！
今年度も皆様の成果物を拝見できることを、楽しみにしています！