品質・仲間・技術と向き合って テスト設計技法の力を引き出す

～テスト設計技法を現場活用する～

井芹　洋輝

JaSST'18 Kyushu

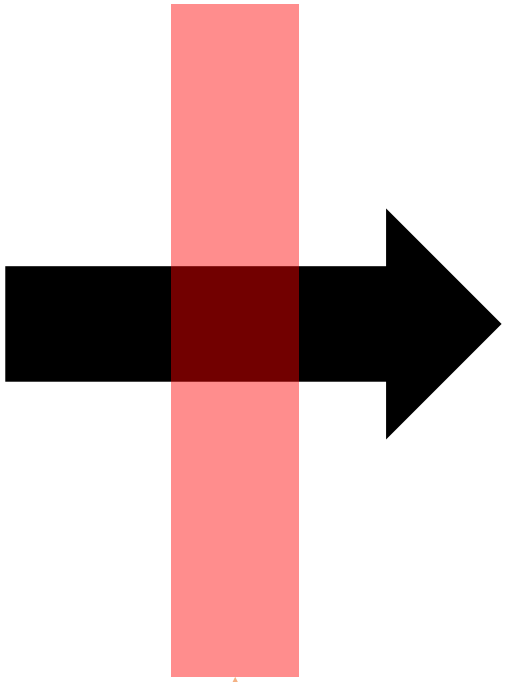
2018/11/22

プロフィール

- 福岡県出身
- 組込み製品の開発・テスト・コンサルタント
  現在：テスト自動化やインフラ整備を率いる
- 社外活動
  - JSTQB技術委員（テストアナリストリーダ）
  - U30テスト設計コンテスト審査員（審査委員長）
- テスト技法について講演・著作複数
  - 「システムテスト自動化標準ガイド」
    「Androidアプリテスト技法」など

講演の目的

テスト
設計技法
の勉強



テスト
設計技法
の現場実践

コンサルティングや教育で壁を実感
この壁を超える手助けをしたい

講演で伝えたいこと

- テスト設計技法を効果的に活用するために、
  - 仲間と力を合わせる
  - 技術力を高めできることを増やす
  - 品質への理解を追求する

→チーム成功のために必要な、テストの役割の本当の姿が見えてくる

→選択すべきテスト設計技法が分かる

アウトライン

1. テスト設計技法の概要
2. テスト設計技法の実践
   - テスト設計の全体の流れ/技法選択のインプットと判断
3. テスト設計技法の選択の難しさ
   - 一部しか網羅できない/テストベース不足/
     本質的な妥当性はわからない/テストリソース不足
4. テスト設計技法の活用所を見つける
   - 仲間と力を合わせる/技術力を高めできることを増やす/
     品質への理解を追求する
5. 具体例を用いた解説

テスト設計技法とは

テスト設計技法とは

テスト設計技法（test design technique）：
「テストケースを作成したり選択したりするための技法」

- JSTQBソフトウェアテスト標準用語集 ver.3.J01

テスト設計技法の主要な構成要素（全てまたは一部で構成）

1. モデル化
   - 対象を技法適用が可能なモデルに変換
     - 例）デシジョンテーブル、状態遷移図、パラメータと値のリスト
2. 網羅基準の設定
   - モデルに対する網羅基準を設定
     - 例）状態遷移図に対するnスイッチカバレッジ
3. テストケースの作成
   - モデルと網羅基準からテストケースを導出

テスト設計技法の恩恵

1. テスト目的に沿ったテストを作成できる
   - 例）特定の組み合わせを網羅
2. テストベースの理解を助ける
3. テストベースの誤り・不整合を見つける
4. テスト設計のコミュニケーションを支える
   - 例）議論、レビュー、教育

多種多様な技法を使いこなすことで
様々な場面で技法の恩恵を発揮できる

テスト設計技法の例（JSTQBテストアナリスト）

- 仕様ベースの技法
  - 同値分割法
  - 境界値分析
  - デシジョンテーブル
  - 原因結果グラフ法
  - 状態遷移テスト
  - 組み合わせテスト技法
  - ユースケーステスト
  - ユーザストーリーテスト
  - ドメイン分析
- 欠陥ベースの技法
  - 欠陥分類法
- 経験ベースの技法
  - エラー推測
  - チェックリストベースドテスト
  - 探索的テスト

JSTQB Advanced Level シラバス日本語版テストアナリスト Ver. 2012.J01

テスト設計技法の詳細

- 入門図書
  - 「ソフトウェアテスト技法ドリル」
    秋山浩一（日科技連出版社）
  - 「はじめて学ぶソフトウェアのテスト技法」
    リー・コープランド（日経BP社）
  - 「ソフトウェアテスト技法」
    ボーリス・バイザー（日経BP社）
  - クラシフィケーションツリー法入門
    https://www.slideshare.net/goyoki/ss-42412647

テスト設計技法の実践

テスト設計技法の実践を支えるもの

- テスト設計技法の習得

- テスト設計技法の使いどころの識別
  - 技法内の選択含む

テスト設計技法の実践を支えるもの

- テスト設計技法の習得

  基礎教養。教材多数。習得は学習意欲依存

- テスト設計技法の使いどころの識別
  - 技法内の選択含む

  適切な分析が求められる。
  技法実践での定番の課題

テスト設計の全体の流れ（JSTQB）

テスト分析

- テストベースを分析
- テスト条件を識別

テスト設計

- テスト網羅基準、テスト設計技法を選択
- テストケースを作成

テスト実装

- 手順を作成
- データや環境を準備

テスト実行

テスト設計の全体の流れと テスト設計技法の選択

テスト分析

- テストベースを分析
- テスト条件を識別

▲技法選択の準備をする
・分析に用いる技法あり（同値分割。技法のモデルを活用）
・技法選択の判断情報を集める

テスト設計

- テスト網羅基準、テスト設計技法を選択
- テストケースを作成

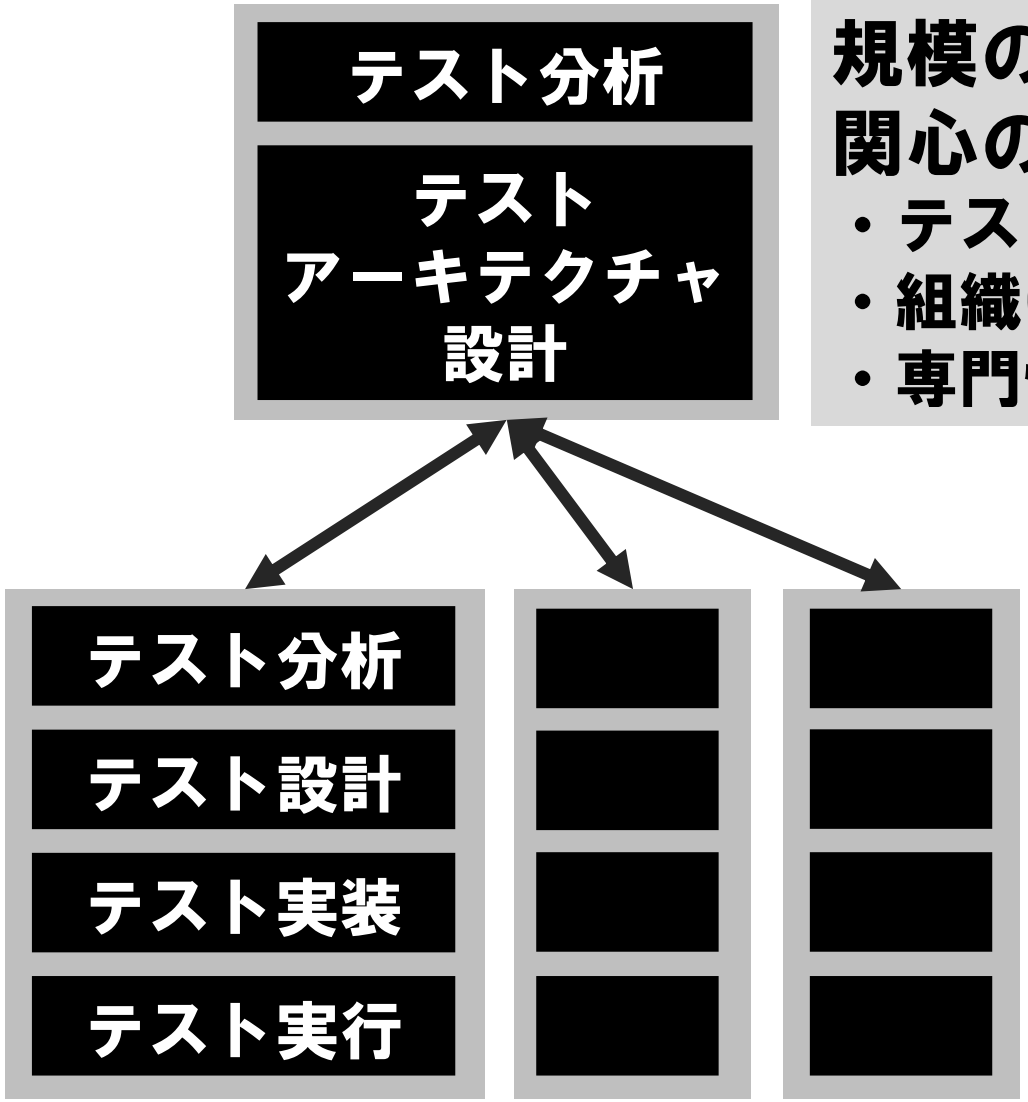
★技法を選択する

テスト実装

- 手順を作成
- データや環境を準備

テスト実行

閑話：テスト設計の全体の流れ
規模の大きさ・複雑さへの対応



規模の大きさ・複雑さに
関心の分離で対応

- テストレベルによる分割
- 組織による分割
- 専門性が求められるテストの分離

個々の詳細なテスト設計

テスト設計技法の選択判断

テスト設計技法の特徴

- 特定のモデルを扱う
- 特定の品質リスクを扱う
- 課題に応じた強み/弱みがある

テスト設計技法の選択判断

テスト設計技法の特徴

- 特定のモデルを扱う
- 特定の品質リスクを扱う
- 状況に応じた強み/弱みがある

本質的な対象のモデルに合わせて技法を選択

目的の品質リスクに合わせて技法を選択

強みを活かせる技法を選択

3つの基準で技法選択

テスト設計技法は適材適所に選択する必要がある
誤った技法はテストの生産性を低下させる

テスト設計技法の選択判断の例（クラシフィケーションツリー法の場合）

プロジェクトの状況

- 環境構成のテストをしたい
- それぞれの環境構成の単機能・組み合わせをテストしたい
- 環境条件は複雑
- テストベースの仕様書の記述は曖昧で不十分。全容がわからない

クラシフィケーションツリー法の特徴

【対応モデル】値（同値クラス）のリスト、組み合わせ

【対応品質リスク】組み合わせバグ、単機能バグを網羅検出

【強み】複雑・曖昧な組み合わせを整理しながら分析できる
ツリー構造で全体を俯瞰しながら分析できる
さまざまな網羅基準に対応

対象のモデル、品質リスクに対応。強みを活かせる→技法として採用を判断できる

テスト設計上の情報で技法選択に用いるもの：テストの要求と制約

テスト対象の情報

- 例）組み合わせor状態遷移

テストの十分性要求：どこまでテストすべきか

- 例）品質リスクをどこまで抑えたいか、テストベースをどこまで網羅したいか

テストの実現性制約：どこまでテストできるか

- 例）チームのスキルレベル、使える時間・コスト

テスト設計上の情報で技法選択に用いるもの

【テストのインプット】●テストベース　●知識や経験
●上位の標準、計画、方針（例：プロジェクト計画）　●プロジェクト状況
●テストニーズ・シーズ　●使用可能なリソース（時間、環境、人材など）
●過去のフィードバック　●テスタビリティ

テスト分析と計画づくり

【分析成果物】●テスト計画・戦略　●テストの制約
●テスト要求の分析結果（仕様化されたニーズ・シーズ）
●テストベースの分析結果（本質的なテストベース）
●テスト条件　●品質リスク

テスト対象情報　テストの十分性　テストの実現性

3つの判断基準を適用

テスト設計技法の選択

テスト設計上の情報で技法選択に用いるもの

【テストのインプット】

テスト分析

- テスト要求分析
- テスト条件の識別

品質
リスク管理

見積もりと計画づくり

- プロジェクトリスク管理
- リソース・スケジュール
  段取りなどの計画づくり

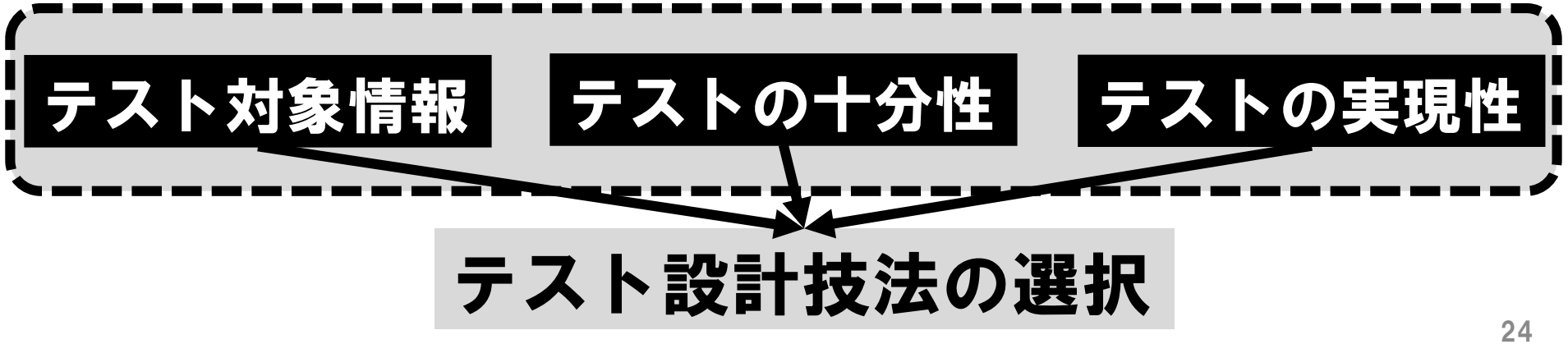
テスト対象情報　テストの十分性　テストの実現性

3つの判断基準を適用

テスト設計技法の選択

テスト設計上の情報で技法選択に用いるもの

すり合わせて妥当なテストを目指す
・例）テストの十分性要求の達成が困難
　→リソース増強やテスト技術向上を行う



適材適所の
テスト設計技法選択の難しさ

テスト設計技法の選択は難しい

- 大まかな方針は分析や、原則・グッドプラクティスの適用で見えてくる
- それ以降で、4つの大きな制約から、正解が見えないまま試行錯誤が求められる

制約1：
テストは対象のごく一部しか網羅できない

1. ソフトウェアが取りうる条件は膨大
   - 入出力の取りえるパターン
   - 入出力の組み合わせ
   - タイミングのパターン（並行処理では無数にある）
   - 状態のパターン（OSの状態など膨大にある）
2. 一般的に本番環境は一部しか再現できない

テスト技法で網羅できるのは、ほんの一部。
サンプリングして確認する程度しかできない

制約2：
テストベースは最後まで不足している

- 仕様書・設計書は十分に明文化されない
  - リソースの限界のため
  - 設計・実装（収束）に必要はドキュメントは、テスト（発散）に必要なものより小さいため
  - テストベースのないブラックボックス（SOUP）がありふれている
    - OTS：フレームワーク、ライブラリ、OSなど
    - ドキュメントの欠落したレガシーコード

技法選択にとっての情報は不十分

制約3：ソフトウェアの本質的な妥当性や品質リスクはわからない

- プロジェクトにとっての妥当性は、製品を市場に出して結果を見るまで不透明
  - ビジネスの成否、社会的責務の達成は手探り
    市場での品質リスクも推論するしかない
    - 例）努力して品質を保証しても製品が売れない

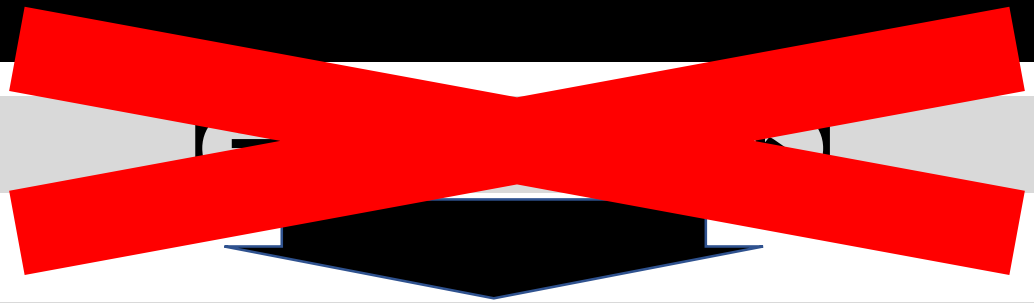
テストの十分性の源泉である、製品の妥当性や品質リスクは事後までわからない

制約4：
テストリソースの不足は解消しない

- テスト設計の良否は見えにくい
  - 必要なテストリソースがみえない・理解されない
  - 利益確保のためテストリースが最小化される
- テスト実施タイミングは後ろ
  - 他工程の遅延・リソース不足の影響を受け、リソースをさらに減らされる

本来必要なテストの十分性＞テストの実現性

制約の影響



テスト分析　　見積もりと計画づくり

テスト対象情報　テストの十分性　テストの実現性

3つの判断基準を適用

テスト設計技法の選択

テスト設計技法の選択の制約まとめ

- リソースが不足していて、妥当性も全容もわからないまま、サンプリング程度の網羅で、テスト設計技法の選択が求められる

  対象の情報：足りない。妥当か分からない

  十分性：全体が不明。一部しかできない

  実現性：リソースが足りない

- 留意点：他の品質確保の手段も同じ
  - 例）レビュー、モデル検査、静的解析、機能安全など

テスト設計技法の活用所を見つけ技法の力を引き出す

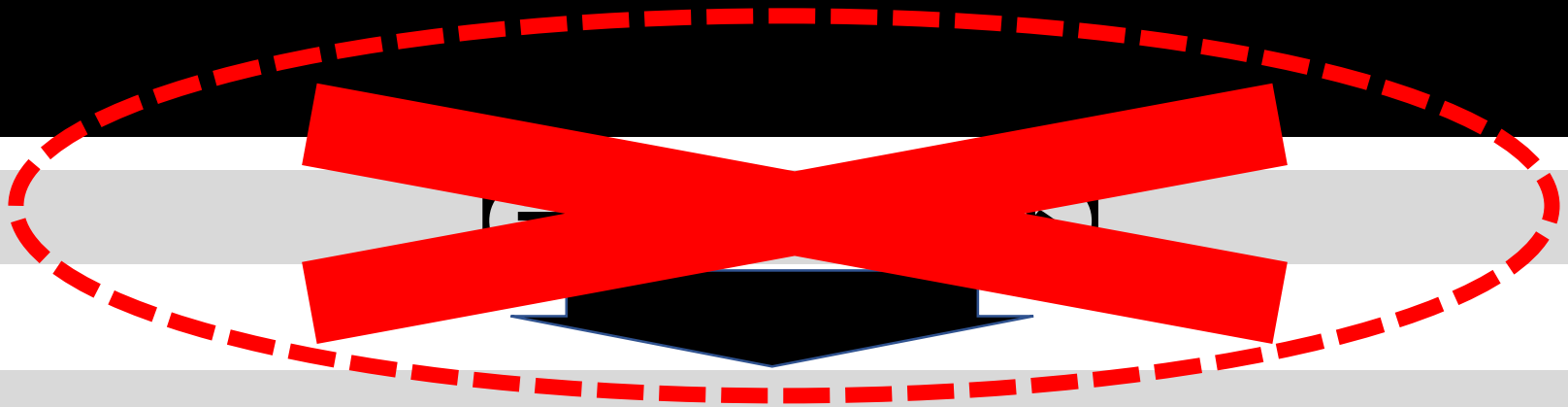
適切なテスト設計技法を選ぶための方針

- 非力な手段で困難な問題に対応する方法：
  - 仲間と力を合わせる
  - 技術力を高めできることを増やす
  - 品質への理解を深めその成果を活かす

→最初から戦略的に継続すると、チーム成功のための**本当のテストの役割**が見える

→選択すべきテスト設計技法が分かる

制約に対応しテスト設計の前提を立て直す



テスト分析　　見積もりと計画づくり

テスト対象情報　　テストの十分性　　テストの実現性

3つの判断基準を適用

テスト設計技法の選択

1.仲間と力を合わせる

- 最初から仲間と協力しながらテストの責務を見出す
  - 包括的な品質保証の戦略立て、重要なリスクコントロール

パターン	目的	例
分業	作業を分割して、扱える規模にする 強みを活かせる作業を選ぶ	静的構造を静的解析で確認、動的なふるまいをテストで確認
重ね合わせ	複数の手段を重複適用して、網羅性を高める 改善サイクルを回す	開発者のテスト、テストチームのテストを二重で実施し、リスクを下げる
横断的最適化	強みを組み合わせて 大きな改善を実現する	テスタビリティを確保し、テストを自動化する

2. 技術力を高めできることを増やす

- チームの技術を高めテストの実現性を拡大。選択可能なテスト設計技法を増やす
  - 既存の作業の効率化
    - テスト作業の自動化技術
    - テスタビリティの開発技術
  - 新たなテストの実現性の拡大
    - 需要のあるテクニカルテスト技術の獲得
    - さらなる複雑さ・大規模さに対応するための整備

2. 技術力を高めできることを増やす：
日頃から・反復的に・進化的に

- 課題：テスト技術は初期導入が重要な技術が多い
  - 例）アーキテクチャレベルのテスタビリティ

  →最初で誤らないよう日頃から技術蓄積が重要
- 課題：テスト技術は事後では対応困難な制約が多い
  - 例）テスト自動化の効果見積もり

  →反復で改善サイクルをまわす
- 課題：テスト技術にはブレークスルーがある
  - 例）テスト自動化の成功→テスタビリティの注入

  →進化的に改善する：
  技術向上に合わせてテスト設計を最適化
  - 例）費用対効果の良いテスト自動化を達成
    →テスト設計でその自動テストの比重を増やす

3.
品質への理解を深めその成果を活かす

- 本質的な品質リスク・妥当性が、制約によって得られないとしても、得ようとする努力の継続は必要
  - 本質的なテストの十分性の見極めに必要
  - 品質保証の穴は品質の精通者によるフォローで埋められている
- 継続的に理解向上の機会を設ける
  - 学習機会の確保（反復開発、早期のユーザテスト）
  - ロールを横断（SETの確保）
  - コミュニケーションの促進

適切なテスト設計技法を選ぶための前提

	対策		
技法選択の制約	仲間と協業	技術向上	品質理解
少ししか網羅できない	●	●	
本質的な妥当性はわからない			●
テストのリソースは足りない	●	●	
テストベースは不十分	▲		●

制約に対応しテスト設計の前提を立て直す

3アプローチの継続推進で立て直す

テスト分析　　見積もりと計画づくり

テスト対象情報　テストの十分性　テストの実現性

3つの判断基準を適用

テスト設計技法の選択

本当のテストの責務と その中での技法の役割を知ろう

- テスト設計技法は適材適所の妥当な用法によって、力を発揮する
- そのために
  - 仲間との協業
  - 技術向上
  - 品質への精通

が土台として必要。テストの戦略に組み込み、継続的に推進する必要がある

閑話：仲間との協業、品質理解はテスト設計技法より重要

- 協業・品質理解のコミュニケーションはテスト設計の重要な基礎。できてないまま技法を勉強しても改善は望めない
  - 基礎ができていないのを、技法による権威付けで紛らわさないようにしよう
  - 開発者やステークホルダと対話しながら、探索的にテストする方が成果を出せる場面は多い

閑話：
テスト設計技法のアンチパターン

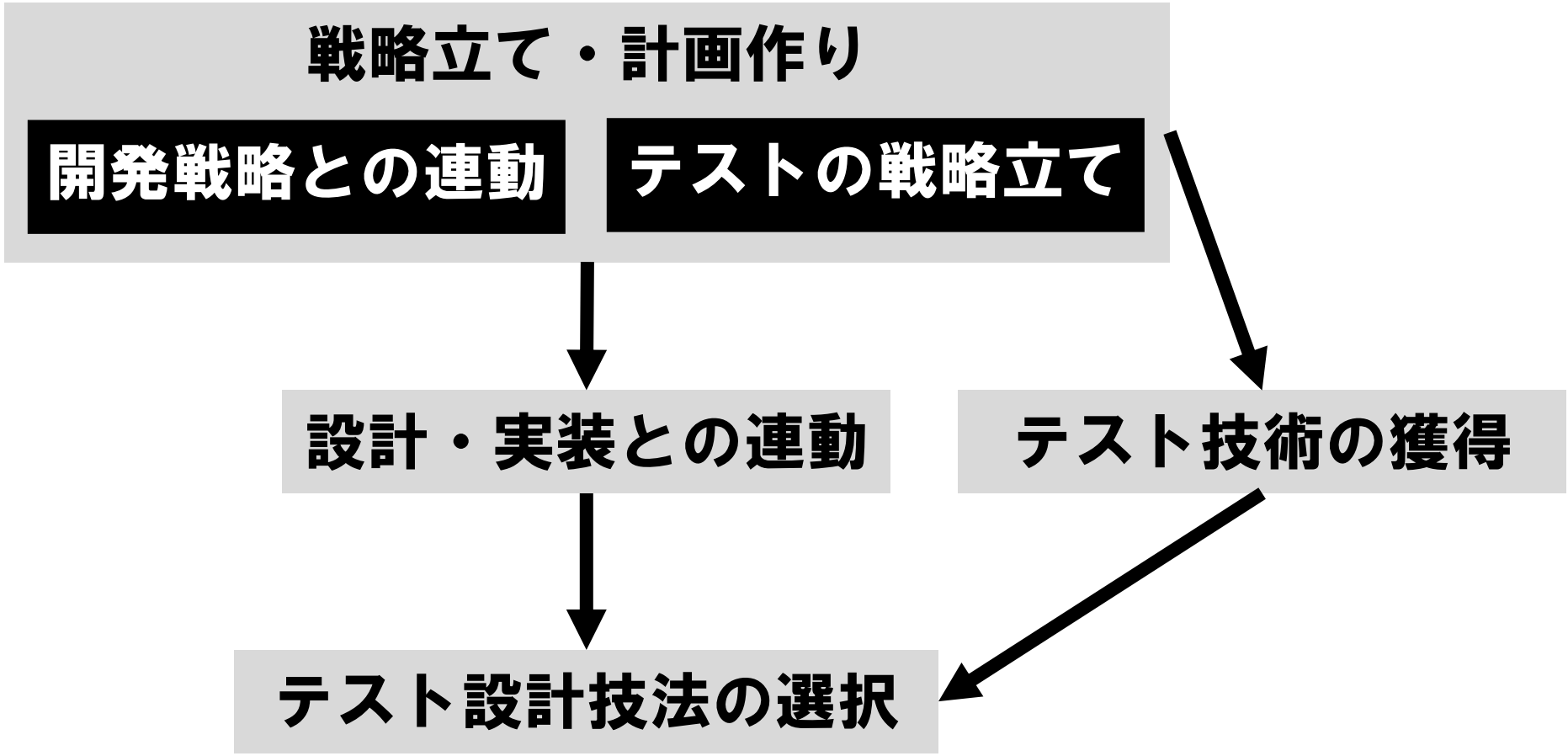
- **手段と目的の取り違え**
  技法遂行をテストの目的に
  「目標コードカバレッジ100%」
- **防衛機制としての技法への昇華・補償**
  「コミュニケーションが苦手でやりたくないから技法で武装」
- **技法権威主義**
  技法をマウンティングや権威付けに用いる
  「ISO29119を使っている！」
  「私は●先生の教えを受けた」
- **技法カルト**
  技法の実践だけでテストを成功させるオカルトパワーが得られる
  「原因結果グラフを使っているから大丈夫」
- **技法提唱者カルト**
  技法提唱者のやり方を完璧に再現するとオカルトパワーが得られる
  「●の著作・教えからは外れてはならない」

具体例で学ぶ
テスト設計技法の選択

事例のサンプル

- プロジェクト
  - ロボット制御ソフトウェア開発
    テストチームを担当
- 状況
  - 高度なデバイス制御が求められる
  - 本番ハードウェアは終盤にしか使えない。
    本番ハードウェアはUIが使いにくい
- 解説中のゴール
  - 適切なテスト設計技法の選択

全体の流れ


戦略立て・計画作り
開発戦略との連動
テストの戦略立て
設計・実装との連動
テスト技術の獲得
テスト設計技法の選択

戦略立て・計画づくり

- 最初からテスト・開発が協調して行う
  - 受け身でなく相互に貢献しあって全体の生産性を高めていく
    - 技法やカバレッジを固定しない（その時の目標として表現してもいいが、状況に応じて変える）
    - 心理的安全性が必要
  - アーキテクチャもテストシステムアーキテクチャを含めて総合的に設計する
  - リスクベースの戦略立ては、比較的テストの必要性を皆に認知させやすい

戦略立て・計画作り

●プロジェクトリスク（リスクレベルが高く皆の協調が必要なもの）

リスク	コントロール方針	リスクコントロール役割
実行環境が終盤まで手に入らない。ハードウェア依存のバグが終盤まで見つからない	エミュレーション、試作基板で、テスト環境を早期に確保	【開発】マルチターゲット対応。 【テスト】エミュレータテスト実施。 エミュレータ技術蓄積 【マネージャ】環境の手配
本番環境の操作制約により、システムレベルのテスト実行に制約がある	品質保証の主体をコンポーネントレベルで実施する	【開発】モジュール性の改善（結合性削減、契約による設計） 【テスト】システムテスト縮小に対するリスクベースドテスト
エミュレーションテストの経験がなく、フィージビリティが不明	プロトタイピングでフィージビリティスタディを行う	【マネージャ】計画確定前のプロトタイピングフェーズの確保 【開発】CIの自動テストとしてエミュレーションテストを導入する

仲間との協業　**技術向上**

戦略立て・計画作り

●プロジェクトリスク（開発中に追加）

リスク	コントロール方針	リスクコントロール役割
制御アルゴリズムの要件が不足。明文化されていない挙動で、顧客要望を満たせない可能性がある	モデル駆動開発でプロトタイピングを行い、シミュレーションを通して顧客とアルゴリズムの同意を得る	【開発】制御アルゴリズムのカプセル化。クラスタ単位で一通りのシミュレーションを実行可能にする 【テスト】モデル駆動テストの導入

仲間との協業

品質理解

●仕様項目ごとのリスクマネジメント

仕様項目	リスクレベル	リスコントロール
姿勢制御	1	【開発】・・・ 【テスト】・・・
・・・		

要件定義・設計でのリスクコントロールの検討を行うテストの十分性に展開する

開発戦略との連携

- テスト設計技法の選択は、テスト対象の設計・実装に依存する
  - テストベースは不十分＆テストベースは一部しか網羅できない

  →構造に対するリスクベーステストで補っていく

開発戦略との連携

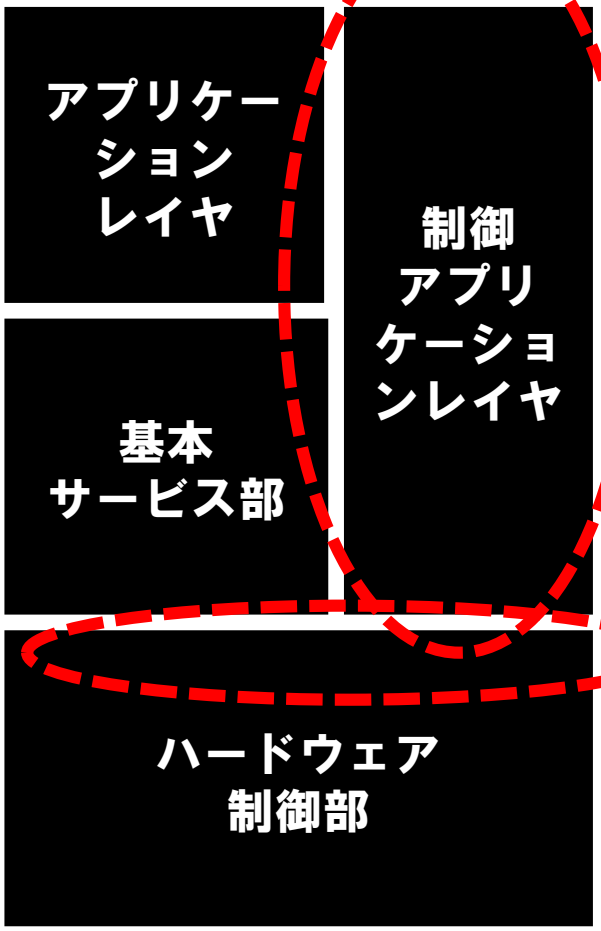
- 設計方針とアーキテクチャ

【アーキテクチャ設計方針】
・契約による設計
・マルチターゲット対応のためのアダプティブなインターフェース
・レイヤドアーキテクチャ
・制御アルゴリズムの責務の分離

開発戦略との連携

- 設計方針とアーキテクチャ

テストレベル・テストタイプの計画の基準にする



【アーキテクチャ設計方針】
- 契約による設計
- レイヤドアーキテクチャ
- 制御アルゴリズムの責務の分離
- マルチターゲット対応のためのアダプティブなインターフェース

ハードウェア制御部を置換した自動テスト導入を検討する

品質リスクを分析しテストを提供する
- 切り替えのメカニズムによる処理時間の低下
- 制御アプリレイヤが、通常レイヤから分離・横断していることによる不具合

仲間との協業

テスト戦略

- 制御アルゴリズムのモデル駆動テスト
  - 制御アルゴリズムの要求分析や妥当性確認に用いる
    顧客とのコミュニケーションに活用
- マルチターゲットテスト
  - ホスト環境、エミュレータ、本番環境それぞれでテスト
- コンポーネントテスト主体の品質保証
  - レイヤレベルのテストを充実させ、システムテストの責務を縮小する
- 反復的なテスト
  - プロトタイピングフェーズを設け、各種技術のフィージビリティスタディを行う

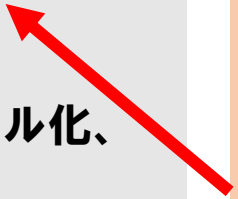
仲間との協業 **技術向上** **品質理解**

設計・実装との連動
コンポーネントテスト主体戦略の場合

- 設計・実装の品質に応じてテスト戦略を変更

契約による設計・レイヤドアーキテクチャの品質確保手段

1. 契約違反の実現不能化
   - ハードウェアによる制限、カプセル化、ロックなど並行処理保護
2. 契約違反の可視化・デバッグ
   - Assertionによる契約遵守チェック
   - サニティテストの搭載
3. 契約違反を検出するプロセス
   - 動的テスト・静的テストを拡充
4. 観察困難な設計上の副作用対策
   - メモリ破壊対策や共有データの保護
   - エラー・例外処理設計（例外保護等）
   - 並行処理設計（マルチタスク、マルチコア、非同期通信等）



リスクを分析
【適切にできている】
コンポーネントテスト主体の品質保証を推進
【適切でない】
設計を是正させるかシステムテスト主体に変更

仲間との協業

品質理解

テスト技術の獲得
マルチターゲットテストの場合

- 環境切り替えのテスタビリティ実装技術
  - 観測点（ログ設計、Assertion）、制御点（API設計）、可変点（Link Seam）の実装
- 必要なテストツールの技術
  - エミュレータ、ホスト環境のテストツール
- インテグレーション
  - ビルドシステム
- 自動化インフラ技術
  - CIツール

・プロトタイピングでフィージビリティを明らかに
・SETを設けて開発者をテストに引き込む

技術向上

技術習得も戦略に織り込む
早期からフィージビリティを明らかにして戦略に反映する

テスト技術の獲得

- 技術獲得は戦略とマネジメントのサポートが必要
- テスト専業の担当では難しい。開発を巻き込んで技術を獲得する
- 改善サイクルの確保のため、反復開発を志向
  - 上流でテストの方針を確定する進め方では、技術の力を矮小化してしまうか、失敗リスクを抱えるかになってしまう

テスト要求からテスト設計の導出

テスト戦略
レベルテスト計画

【テストタイプ】
・ユニットテスト
・結合テスト
　−モデル駆動テスト
　　プロトタイピングテスト
　　制御アルゴリズムテスト
　−エミュレーションテスト
　−ホスト環境テスト
・システムテスト

【テスト要求】

【テスト観点】
・姿勢制御
　オーバーシュート
　発散
・・・

【エミュレーションテスト
上位テストケース】
・ブートテスト
・センサ値入力テスト
・リアルタイム設定更新テスト
・・・・

テスト設計技法の選択

エミュレーションテスト：
上テストケース
・ブートテスト
・センサ値入力テスト
・・・・

テスト設計技法	網羅基準
状態遷移	1スイッチカバレッジ100% 遷移条件MC/DC100%
同値分割 デシジョンテーブル	0ワイズカバレッジ100% 順序を加味して単純化したDT網羅100%

テスト設計技法	網羅基準
境界値分析	3値の境界網羅
エラー推測	センサー入力観点の網羅

まとめ

講演で伝えたいこと

- テスト設計技法を効果的に活用するために、
  - 仲間と力を合わせる
  - 技術力を高めできることを増やす
  - 品質への理解を深めその成果を活かす

→チーム成功のために必要な、テストの役割の本当の姿が見えてくる

→選択すべきテスト設計技法が分かる

制約に対応しテスト設計の前提を立て直す

3アプローチの継続推進で立て直す

テスト分析

見積もりと計画づくり

テスト対象情報

テストの十分性

テストの実現性

3つの判断基準を適用

テスト設計技法の選択

本テーマの推薦図書

- リスクベースでの戦略立て
  - ソフトウェアテスト12の必勝プロセス
  - ソフトウェアテスト実践ワークブック（いずれもRex Black、日経BP社）
- チームやステークホルダとの協業
  - ソフトウェアテスト293の鉄則（Cem Kaner、日経BP社）
  - 実践アジャイルテスト（Janet Gregory他、翔泳社）
  - テストから見えてくるグーグルのソフトウェア開発（James A. Whittaker他、日経BP社）
- テスト技法の使い分け
  - ソフトウェアテスト技法（Boris Beizer、日経BP社）