

JaSST'21 Tokyo

事例からわかる！ テスト自動化導入パターン

浅黄友隆 山口真央

Human Crest

自己紹介

- ❖ 名前 : 浅黄 友隆
- ❖ 主な業務 : テスト自動化、サービス開発
- ❖ 経歴 : 開発エンジニア 10年
テストエンジニア 10年

- ❖ 名前 : 山口 真央
- ❖ 主な業務 : テスト自動化、サービス開発、マーケティング
- ❖ 経歴 : テストエンジニア 7年



株式会社ヒューマンクレスト

設立: 2002/12/16

本社: ランドマークタワー

社員数: 156名

認証資格: ISO27001

ISTQBパートナーシップ: [Platinum](#)



株式会社ヒューマンネクスト (関内)



JapanQuality Co.,Ltd (ベトナム ダナン)

テスト自動化ソリューション

お客様の実態に即したテスト自動化と継続運用を実現させるテスト自動化ソリューション群です。
サブスクリプションモデルから納品型モデルまでご用意しております。



品質改善ソリューション



QAコンサルティングサービス・
マネジメントサービス

▶ MORE DETAILS



品質エンジニアリング・
ソフトウェアテスト

▶ MORE DETAILS



HCテストセンター
(横浜／ベトナム・ダナン)

▶ MORE DETAILS



アジャイルテスト

▶ MORE DETAILS



セキュリティ脆弱性診断

▶ MORE DETAILS



負荷テスト・性能テスト

▶ MORE DETAILS

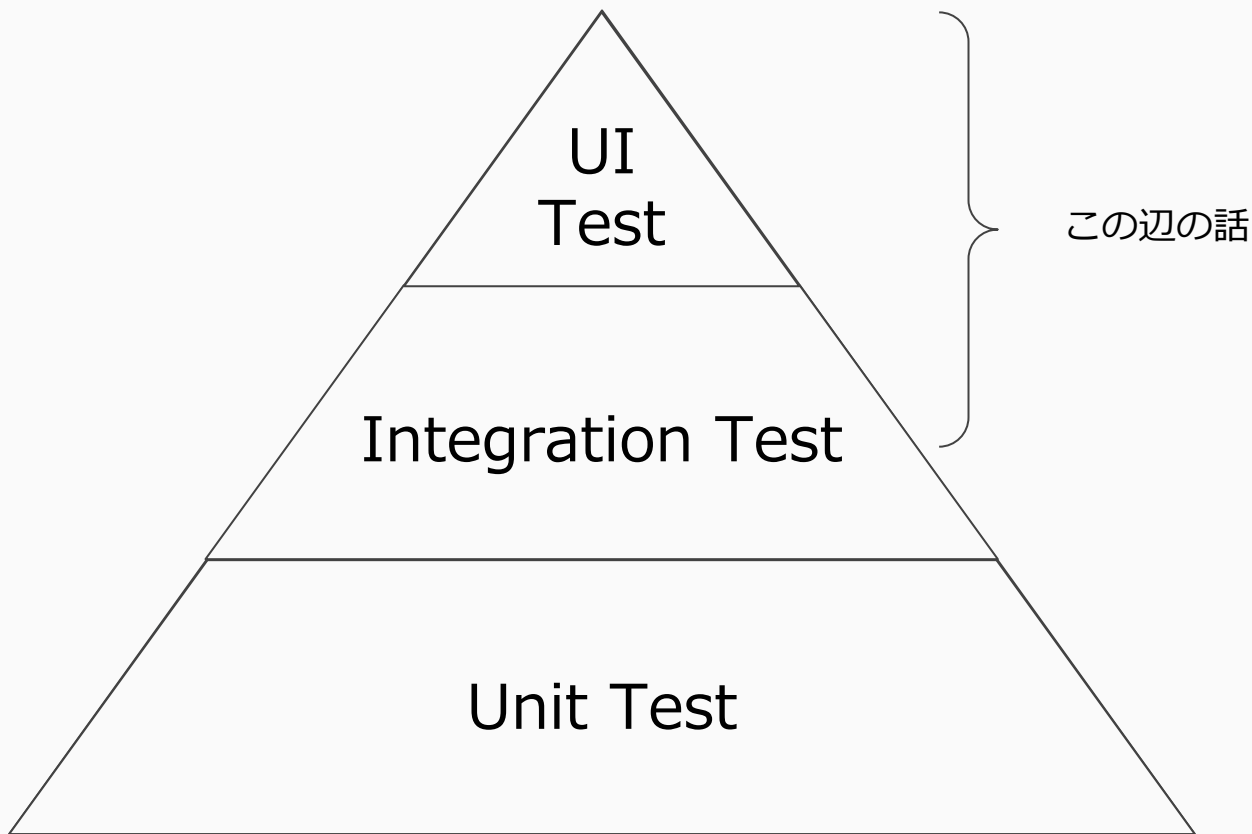


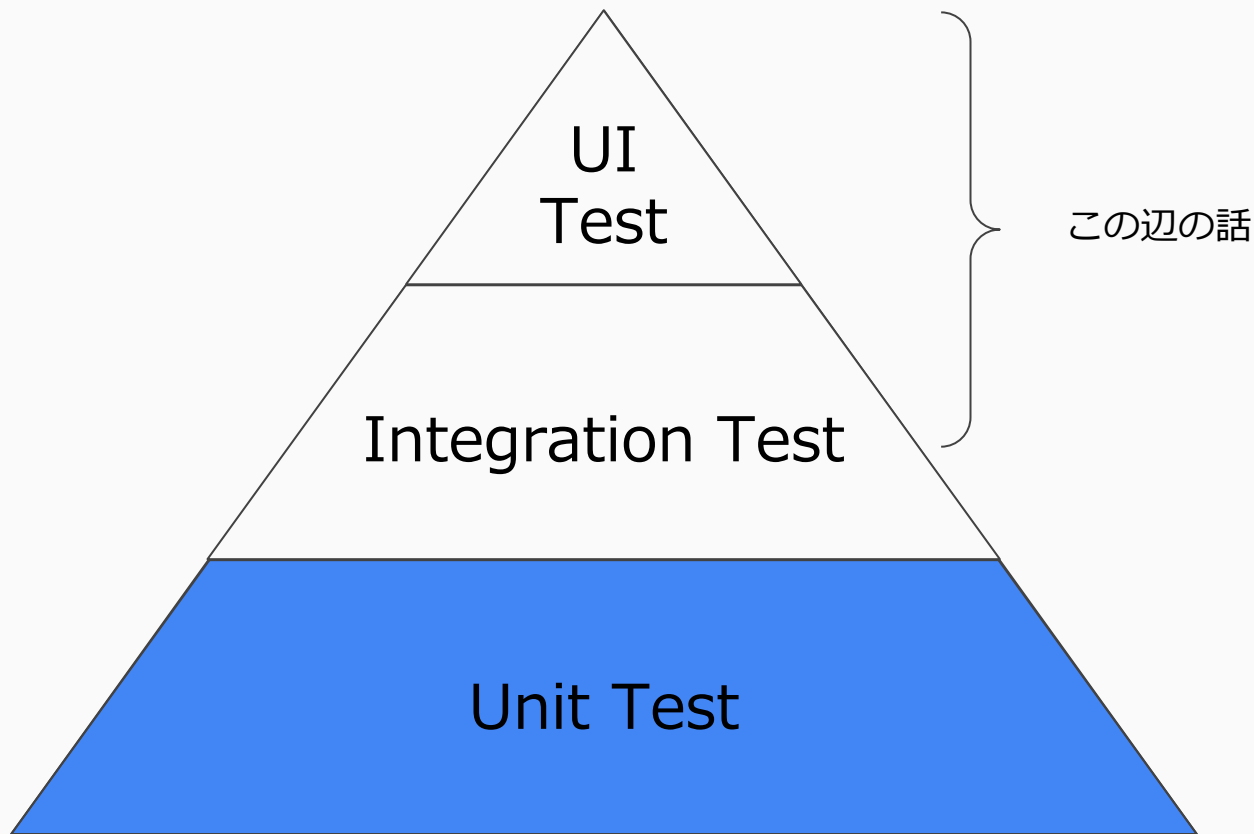
AWSマイグレーションテスト

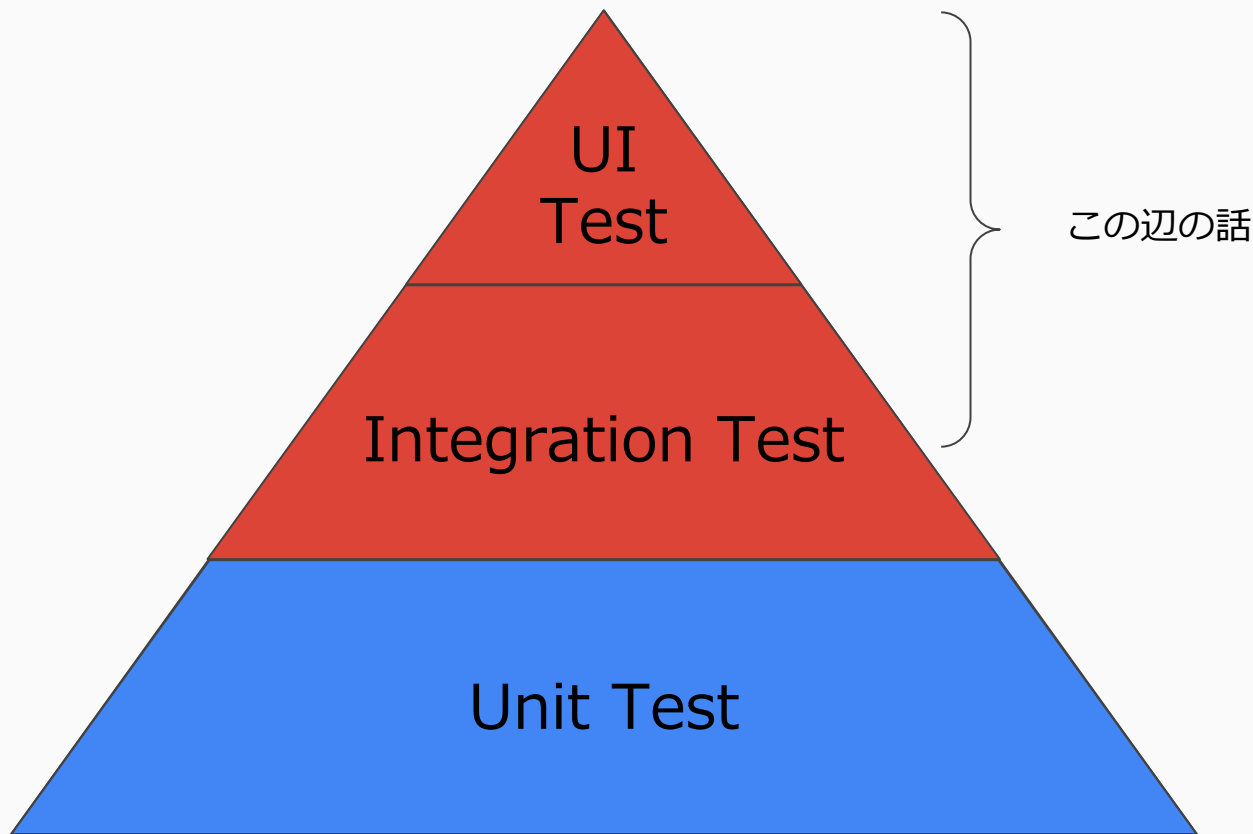
▶ MORE DETAILS

テスト自動化は普通になった

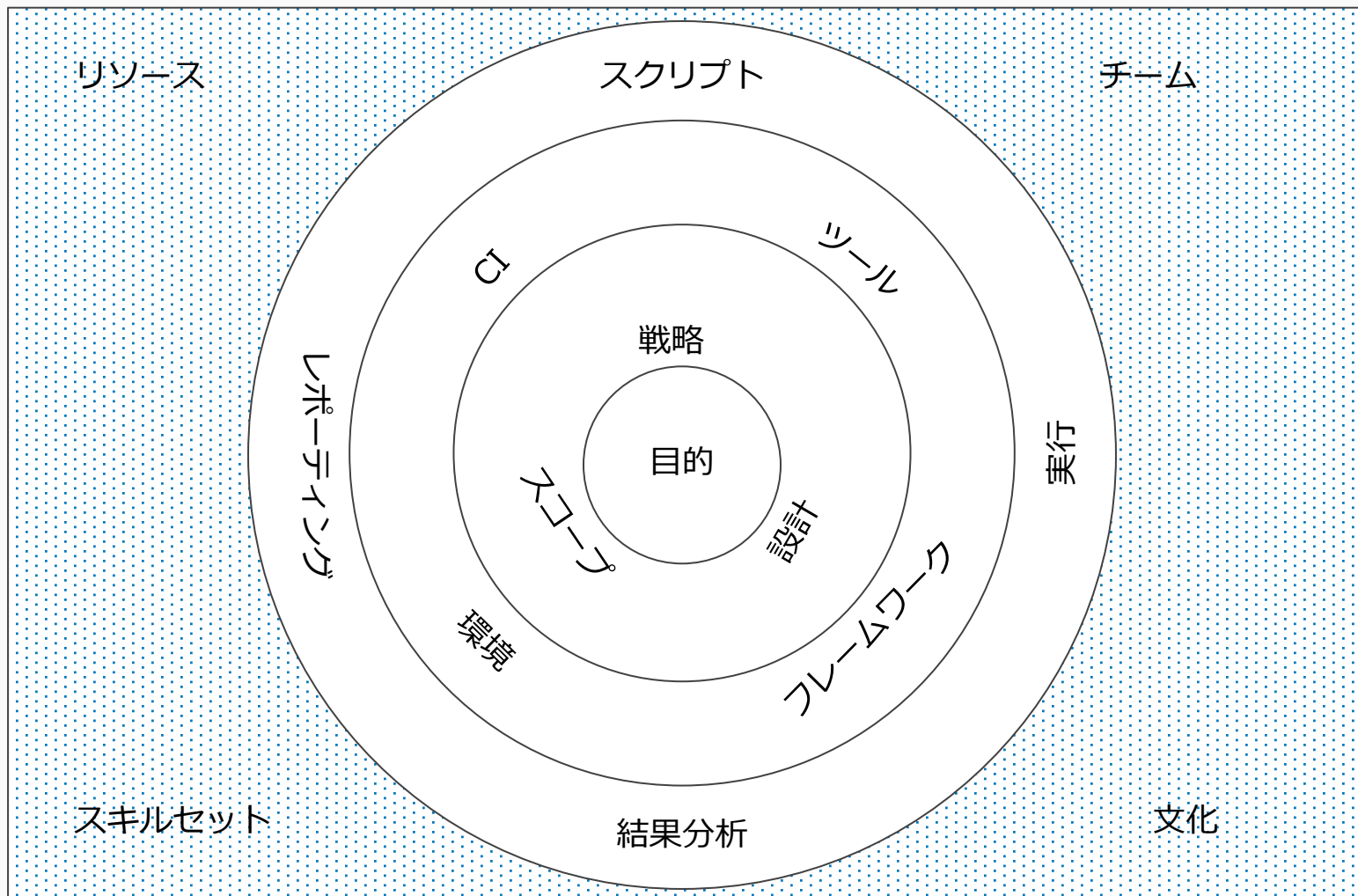
導入できない
維持できない

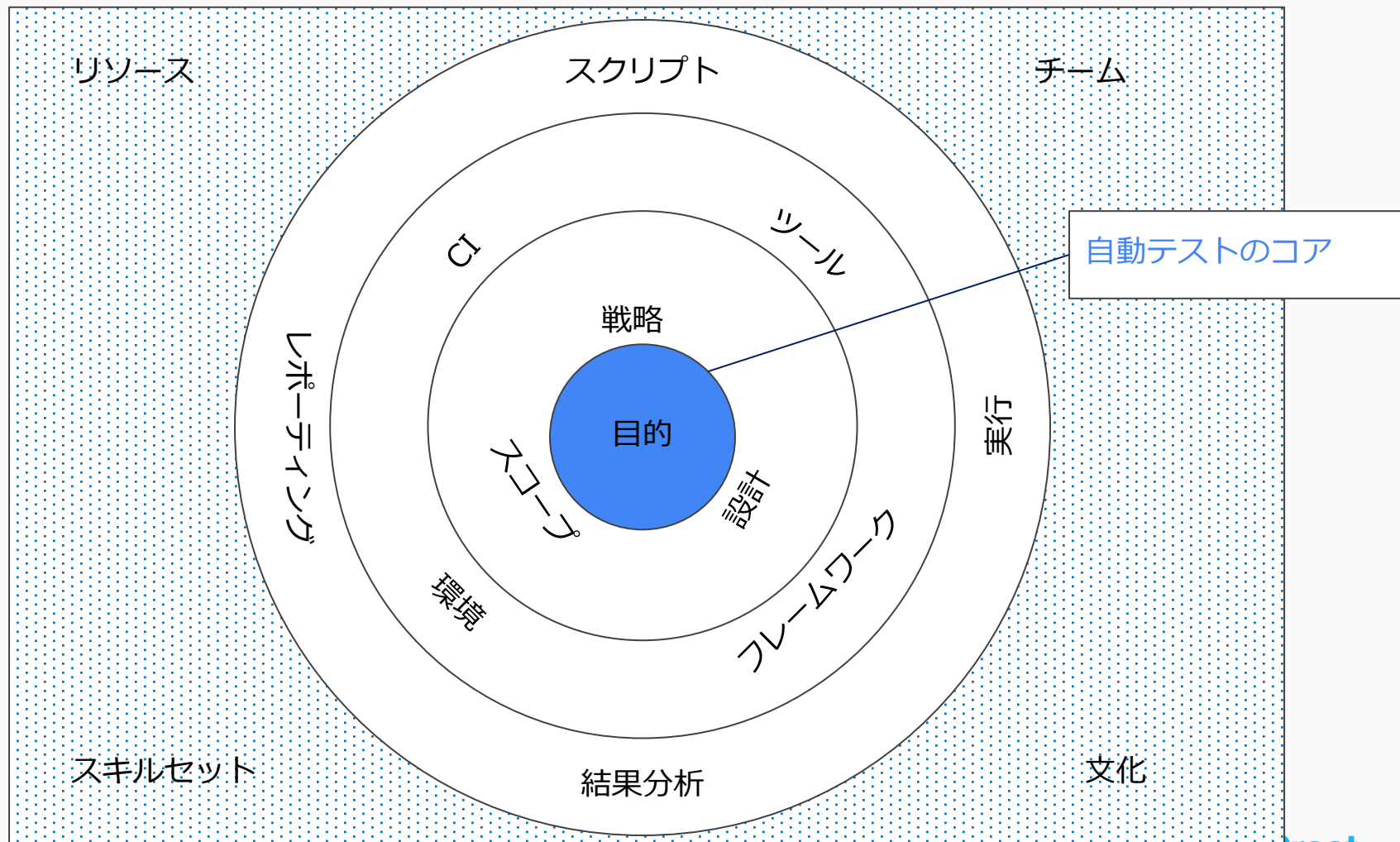


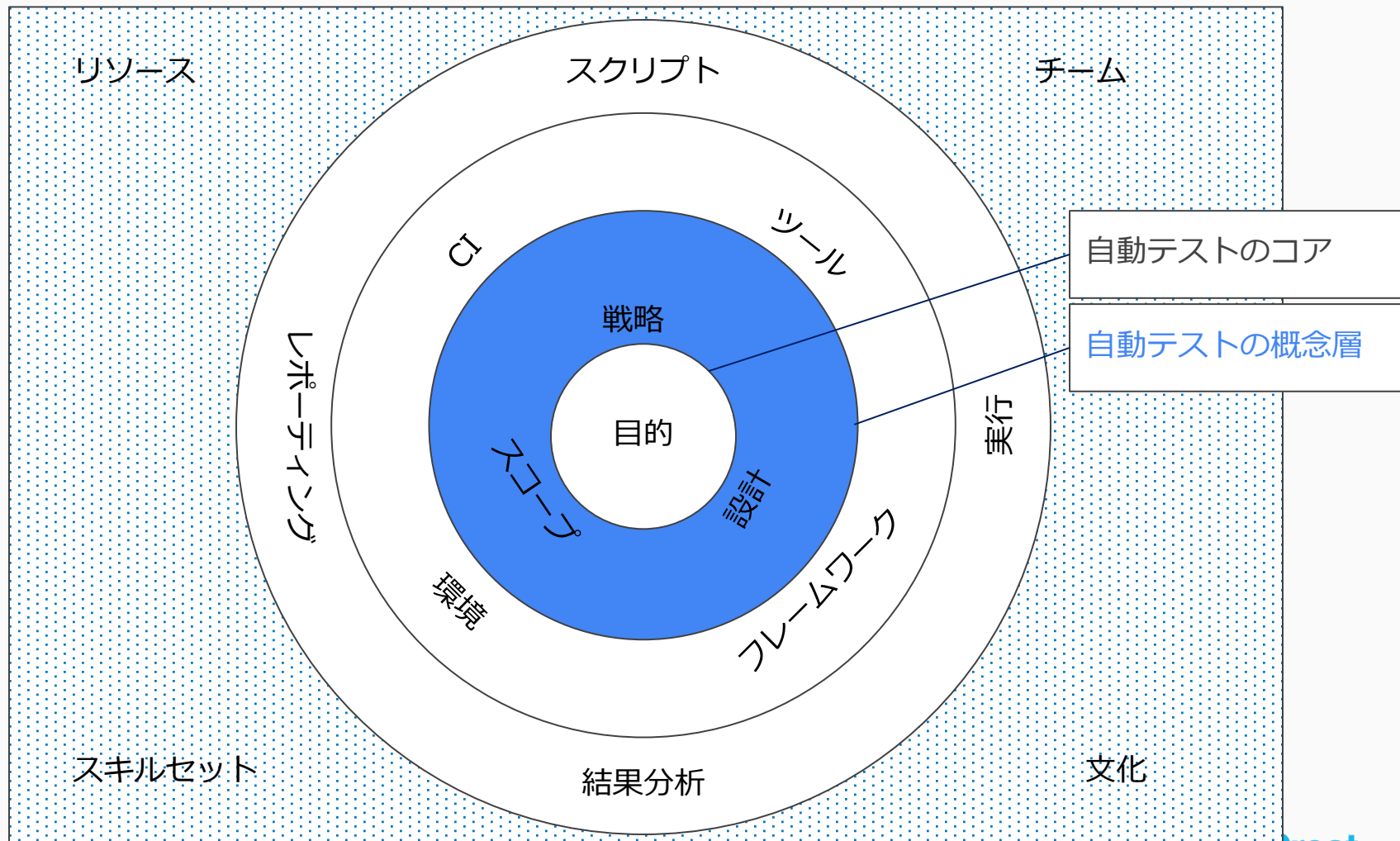


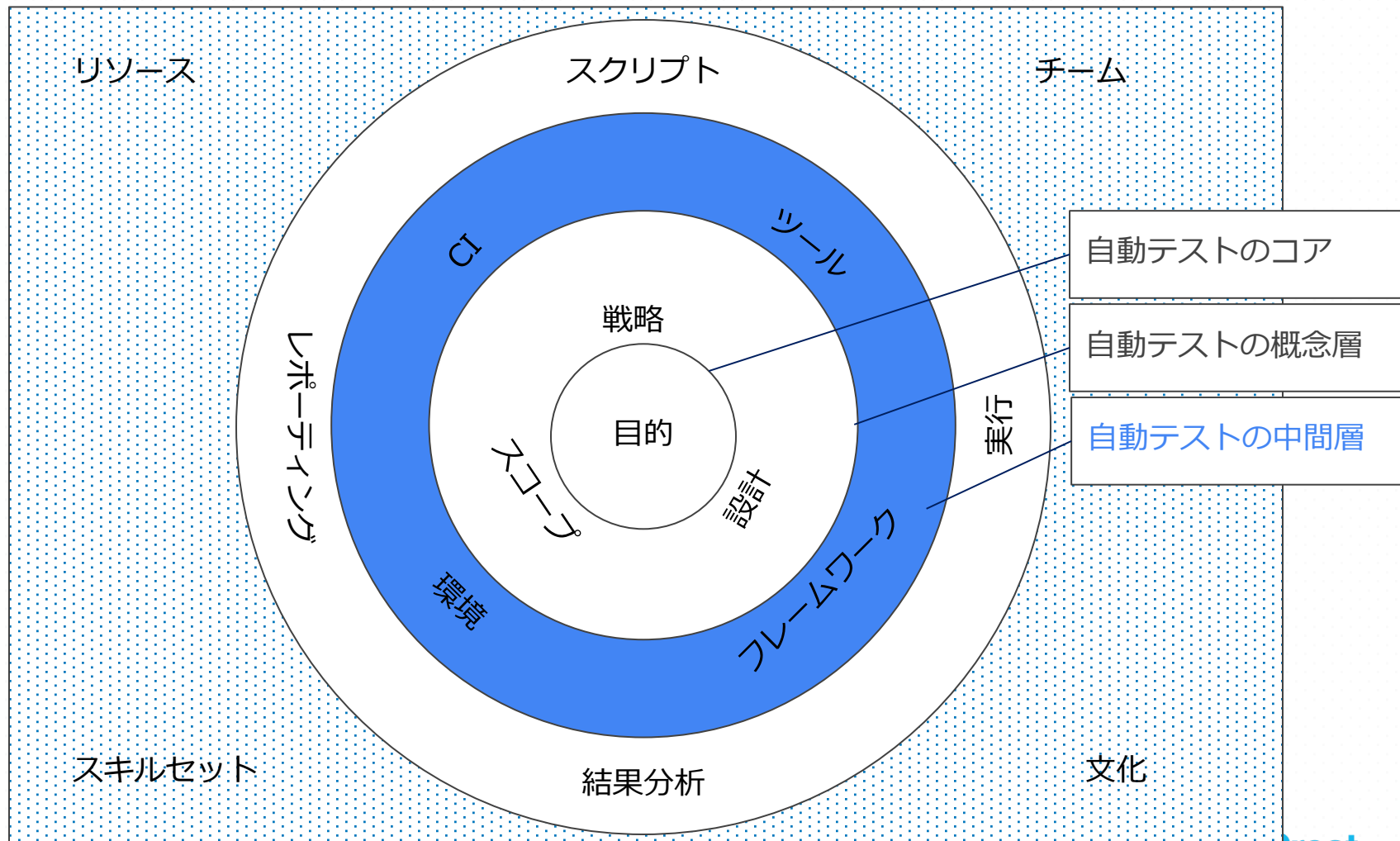


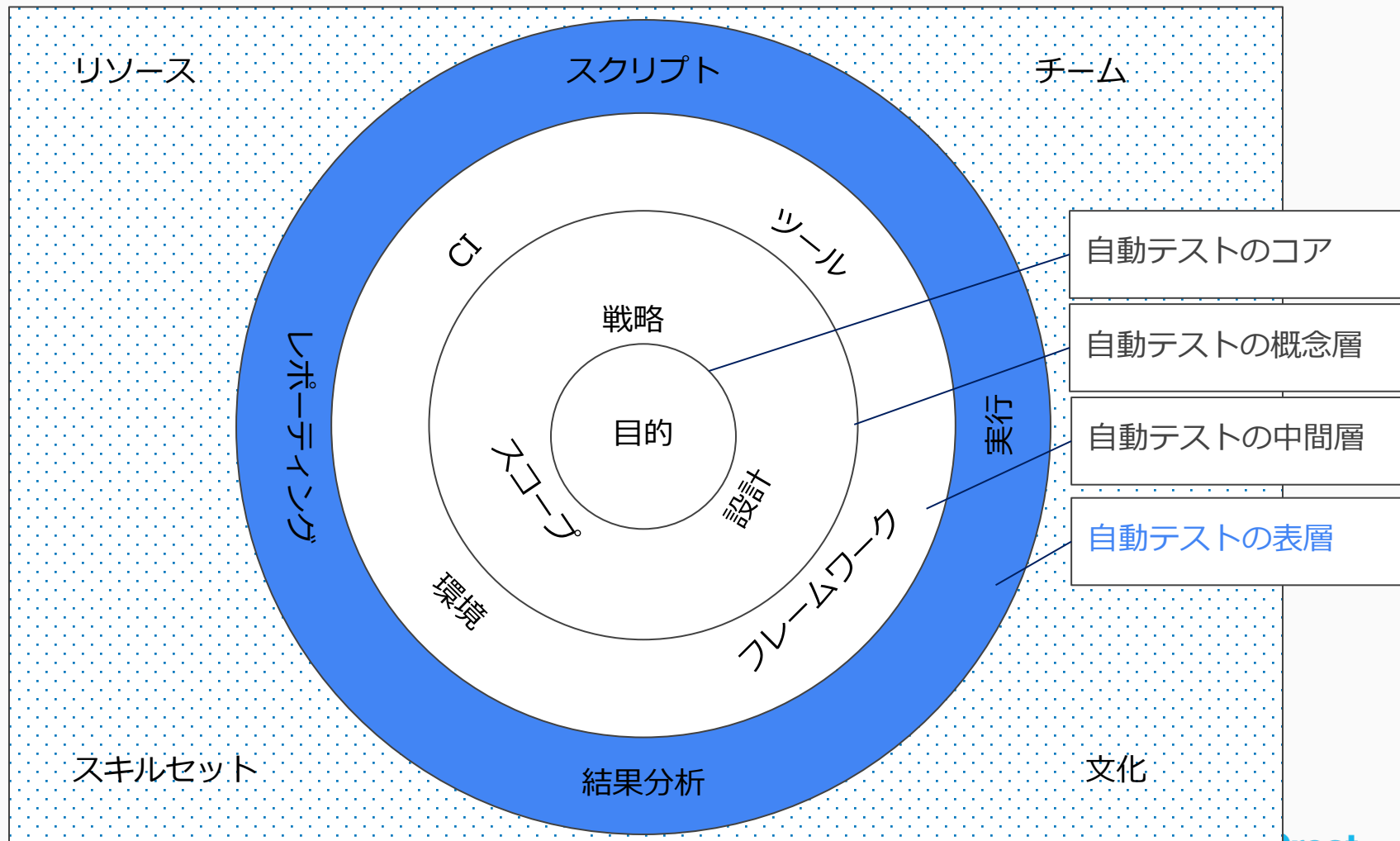
テスト自動化で重要なことは？

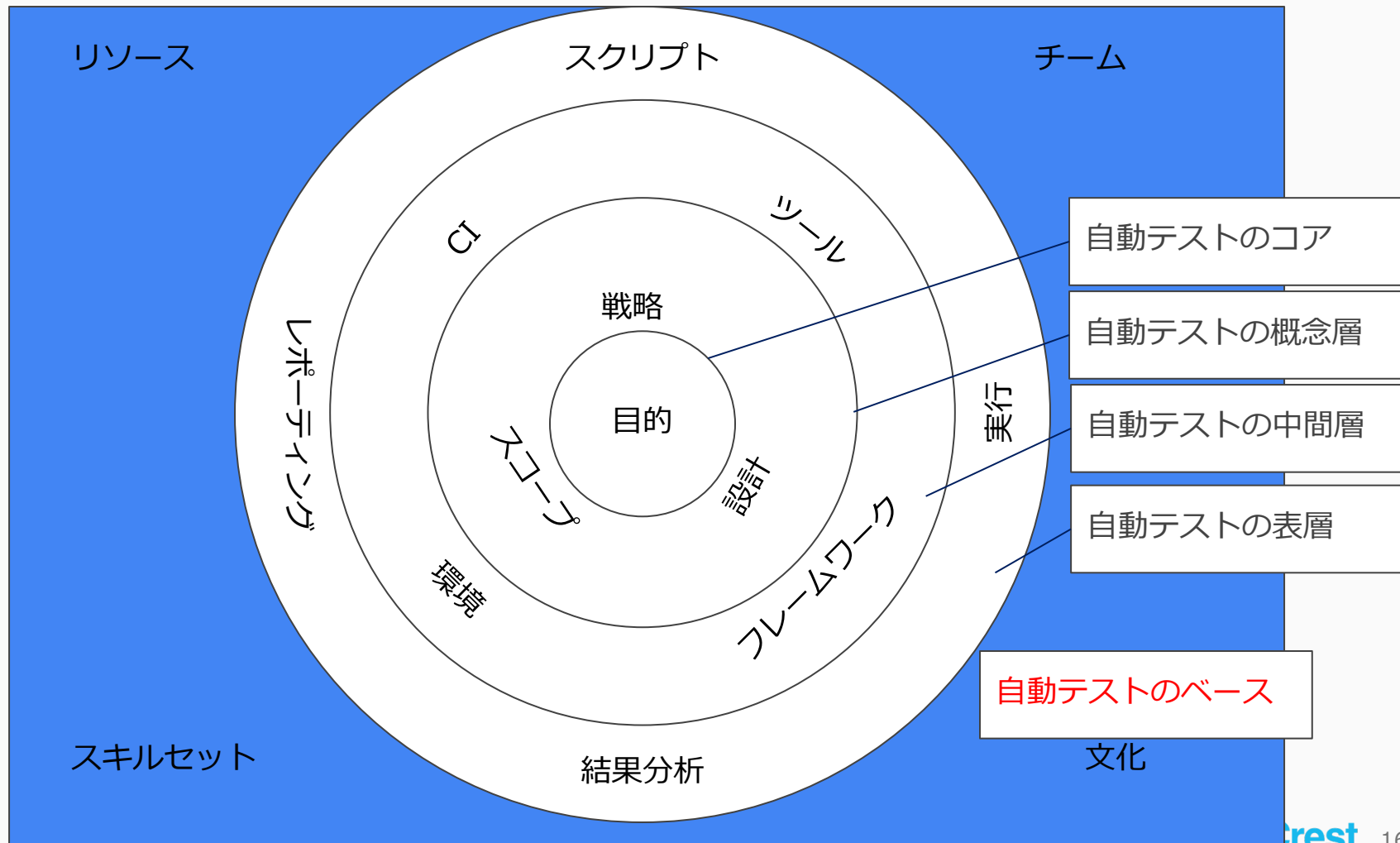








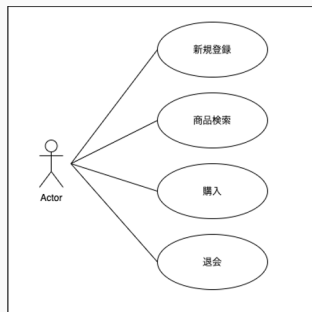




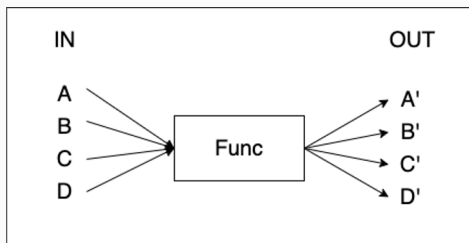
テスト自動化導入パターンは

：

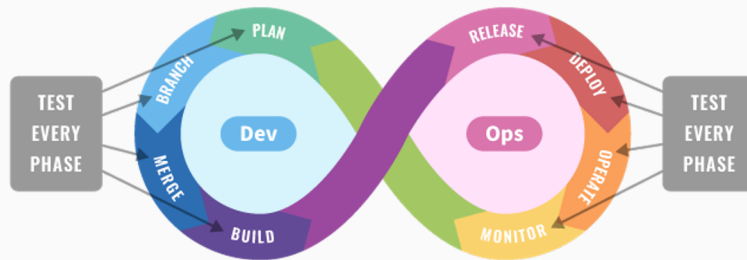
ユーザの価値を重視



繰り返し重視



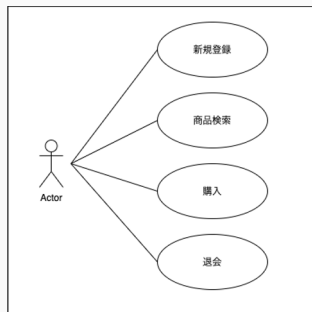
テストスピード重視



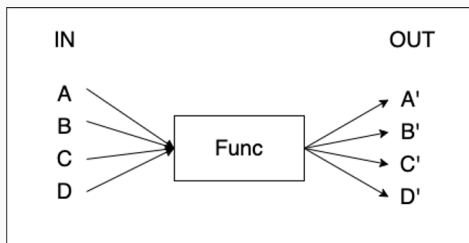
インターフェース重視



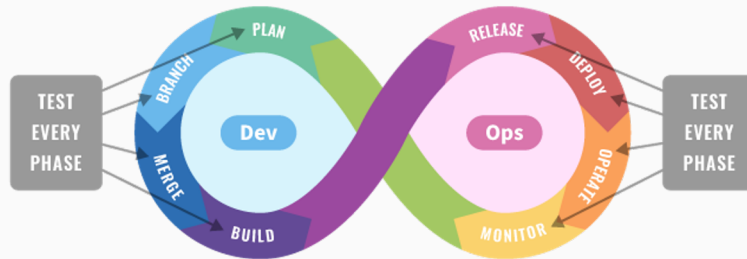
ユーザの価値を重視



繰り返し重視



テストスピード重視



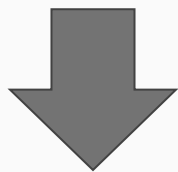
インターフェース重視



目的

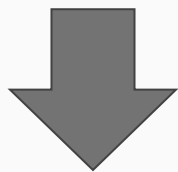
主要な機能/要件が
正常に動作することを確認する

手動テストを自動化

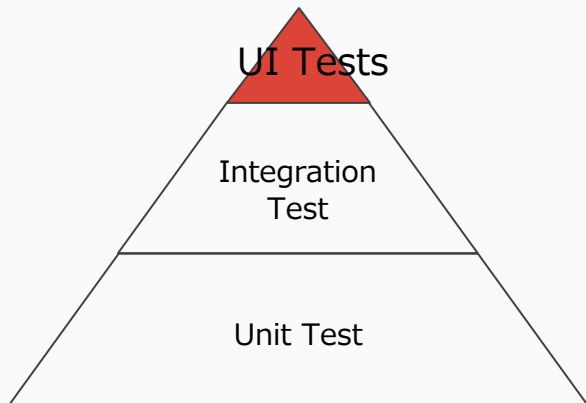


ユーザに提供する価値を
テスト

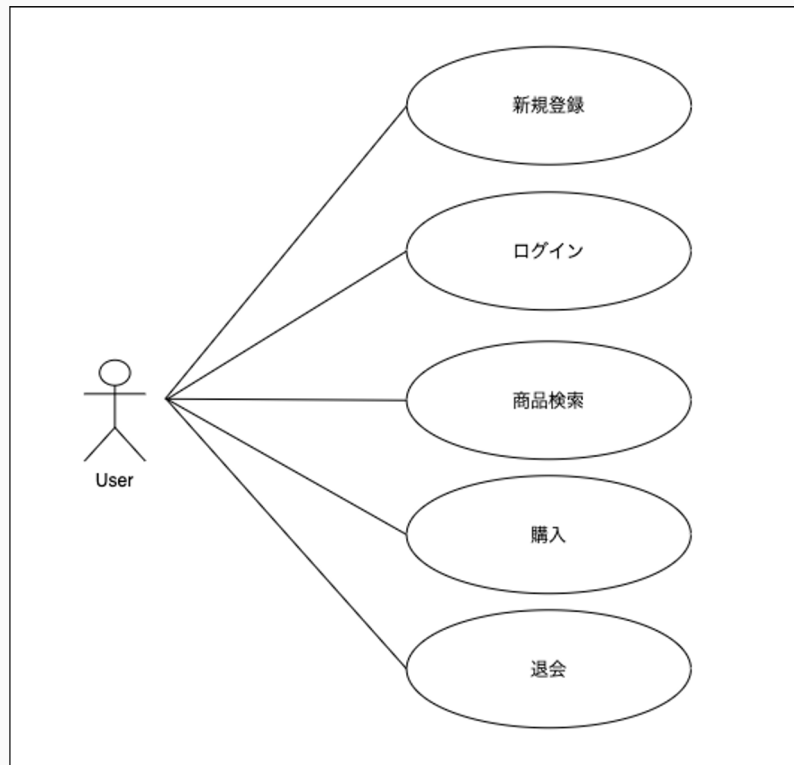
ピラミッドの頂上



少ない



ECサイトの場合



事例：ユーザの価値を重視

【ECサイトの場合】2015～

- ・ユーザの価値：新規会員登録、ログイン、検索、購入、退会
- ・テストタイプ：シナリオテスト

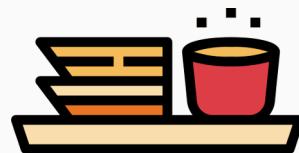


【グループウェアの場合】2014～

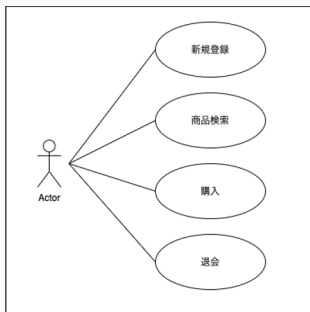
- ・ユーザの価値：予定作成、メール送信
- ・テストタイプ：ユースケーステスト

【グルメサイトの場合】2017～

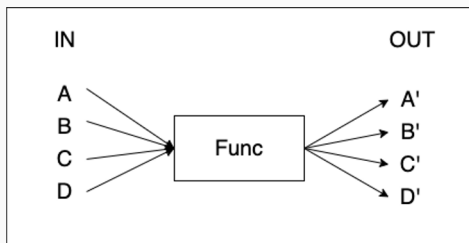
- ・ユーザの価値：店舗検索、予約
- ・テストタイプ：シナリオテスト



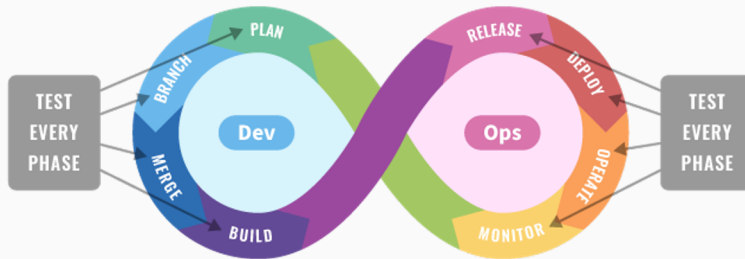
ユーザの価値を重視



繰り返し重視



テストスピード重視



インターフェース重視

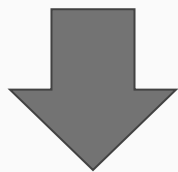


目的

繰り返しテスト
(環境やデータ)
を自動化によって効率化する

繰り返しを自動化

テストケースを繰り返す



ブラウザ
OS
デバイス

事例：テストケースを繰り返す

【ECサイトの場合】2015～

- ・ユーザの価値：新規会員登録、ログイン、検索、購入、退会
- ・テストタイプ：シナリオテスト

・ブラウザ：



【グルメサイトの場合】2017～

- ・ユーザの価値：店舗検索、予約
- ・テストタイプ：シナリオテスト

・ブラウザ：



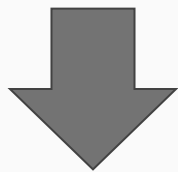
【グループウェアの場合】2014～

- ・ユーザの価値：予定作成、メール送信
- ・テストタイプ：ユースケーステスト

・ブラウザ：



データを繰り返す

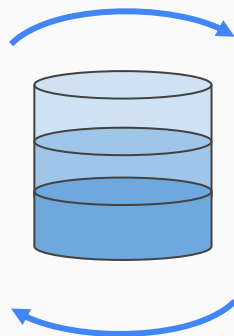


IN/OUTデータの
組み合わせ
パターン網羅

事例：データを繰り返す

【事例A】 2020

- ・ 目的：数値計算の正確性確認（保険）
- ・ 制約：入力→出力はブラウザで確認が必須
- ・ 方法：Seleniumを利用したデータ駆動テスト。ケース数は数万件。
入力パターンによって画面遷移も変わる為、キーワード駆動の考えを取り入れ、パターン毎に画面遷移フローを自動生成するところから開発。



【事例B】 2017

- ・ 目的：超高速開発ツールで作成したシステムの設定ミスを検知
- ・ 制約：ツールの性質上、単体テストは書けない
- ・ 方法：Seleniumを利用したデータ駆動テスト
画面からの入力値に対するエラーメッセージのパターンを確認

テスト自動化の**変化**

様々な開発・環境の変化

インフラ デバイス アーキテクチャ プロセス

開発技術とテストの変化

インフラ環境の変化

オンプレ → クラウド

デバイスの変化

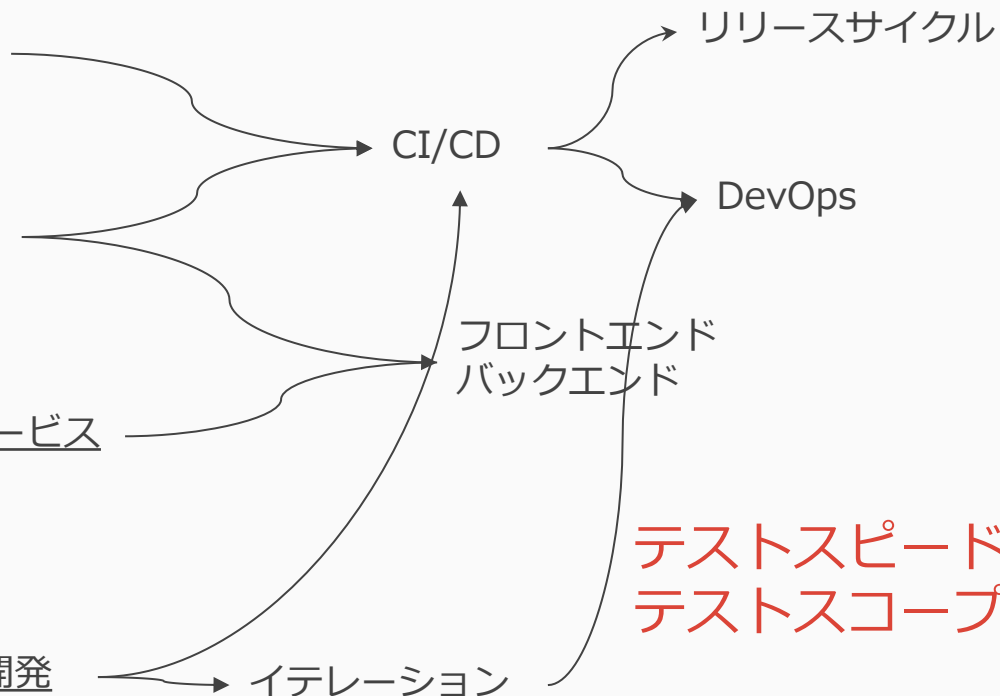
Web → モバイル

アーキテクチャの変化

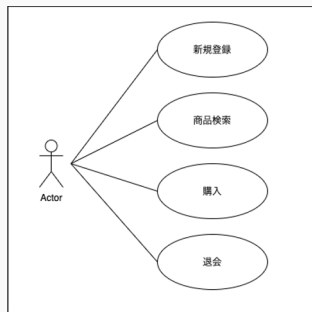
モノリシック → マイクロサービス

開発プロセスの変化

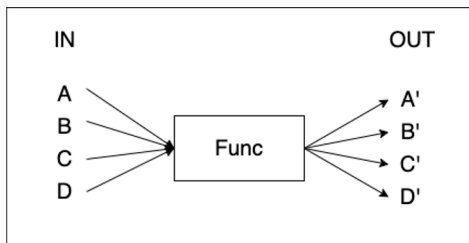
WF開発 → アジャイル開発



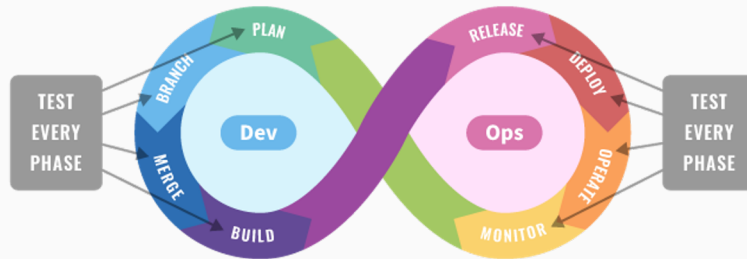
ユーザの価値を重視



繰り返し重視



テストスピード重視

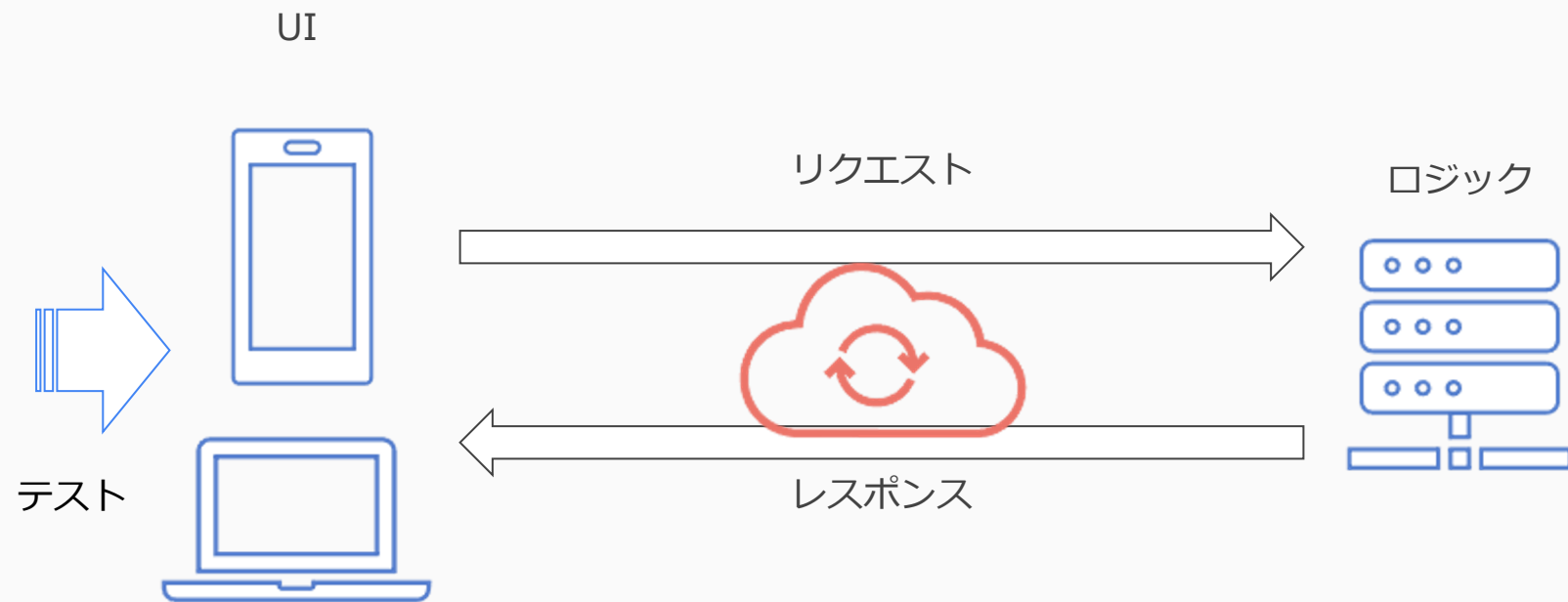


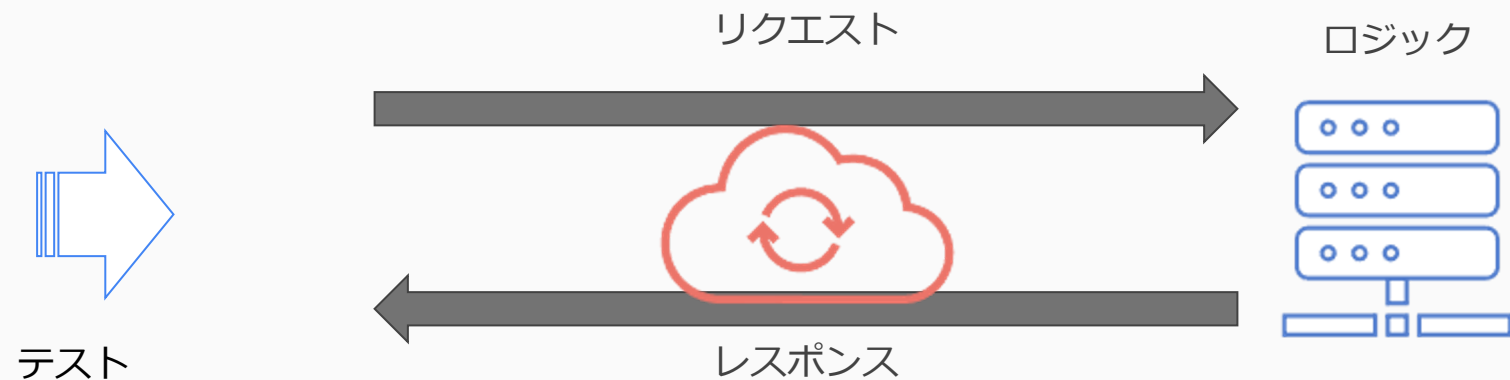
インターフェース重視



テスト自動化のスコープ

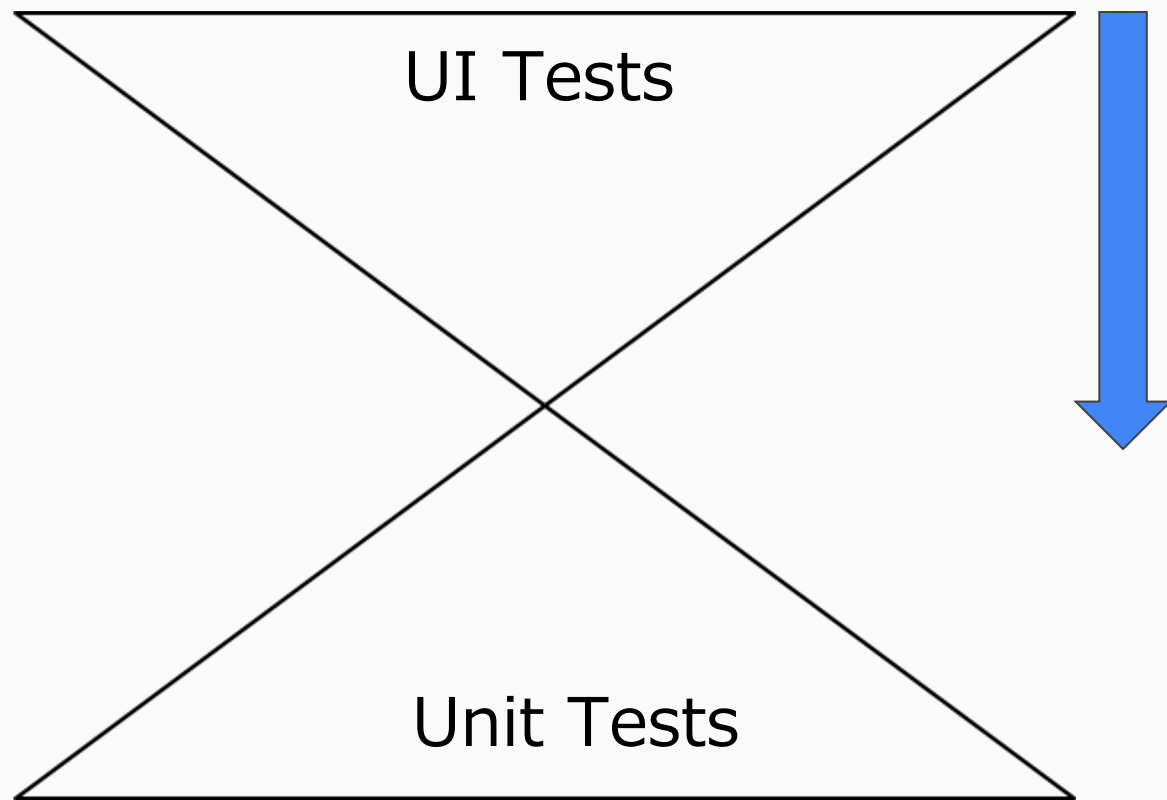
テストスコープ



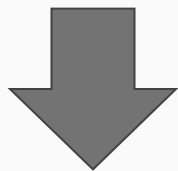


目的

バックエンドの動作を確認
API（サービス）の動作を確認



UIのテスト

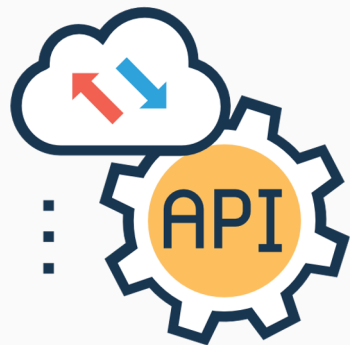


インターフェースのテスト

事例：パターン④インターフェース重視

【事例A】

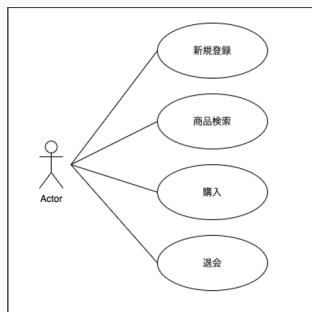
- ・ 目的：APIのリクエスト・レスポンス等が正しいことを確認
- ・ 制約：ステータスコードが特殊
- ・ 方針：API毎のパラメータパターンをテストケースとする
ステータスコードではなく、レスポンスの内容でテスト
APIの順序性、アクセストークン等の受け渡しも考慮する



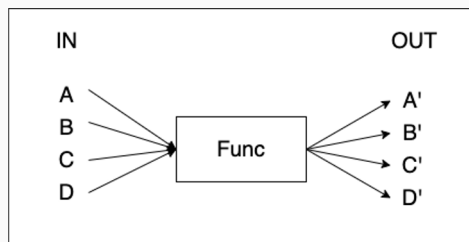
【バックエンドシステム】

- ・ 目的：サービス仕様書の記述を元にインプットデータの
パラメータパターンで機能を有している事を確認
- ・ 方針：APIのリクエスト/レスポンスだけでなく、
DBやログまで確認を実施する

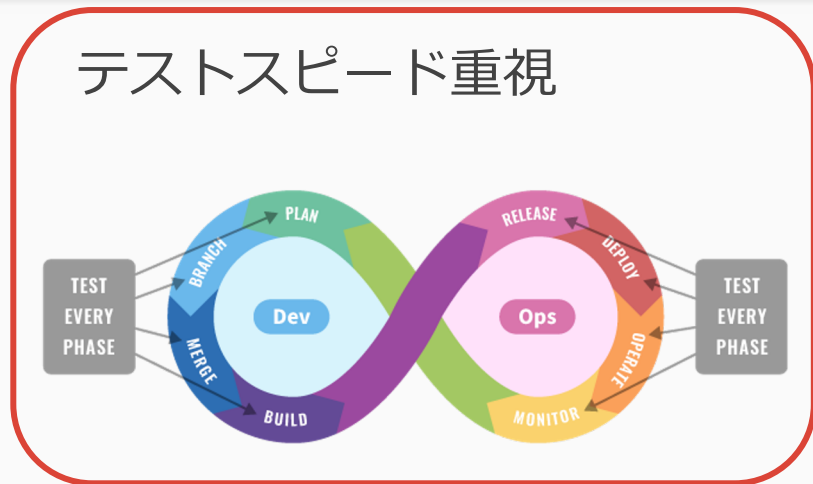
ユーザの価値を重視



繰り返し重視



テストスピード重視



インターフェース重視

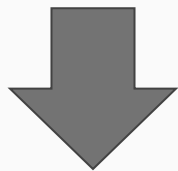


テストスピードを改善

目的

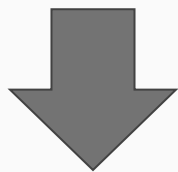
イテレーション
リリースサイクル
の中でテストする

開発スピードの変化



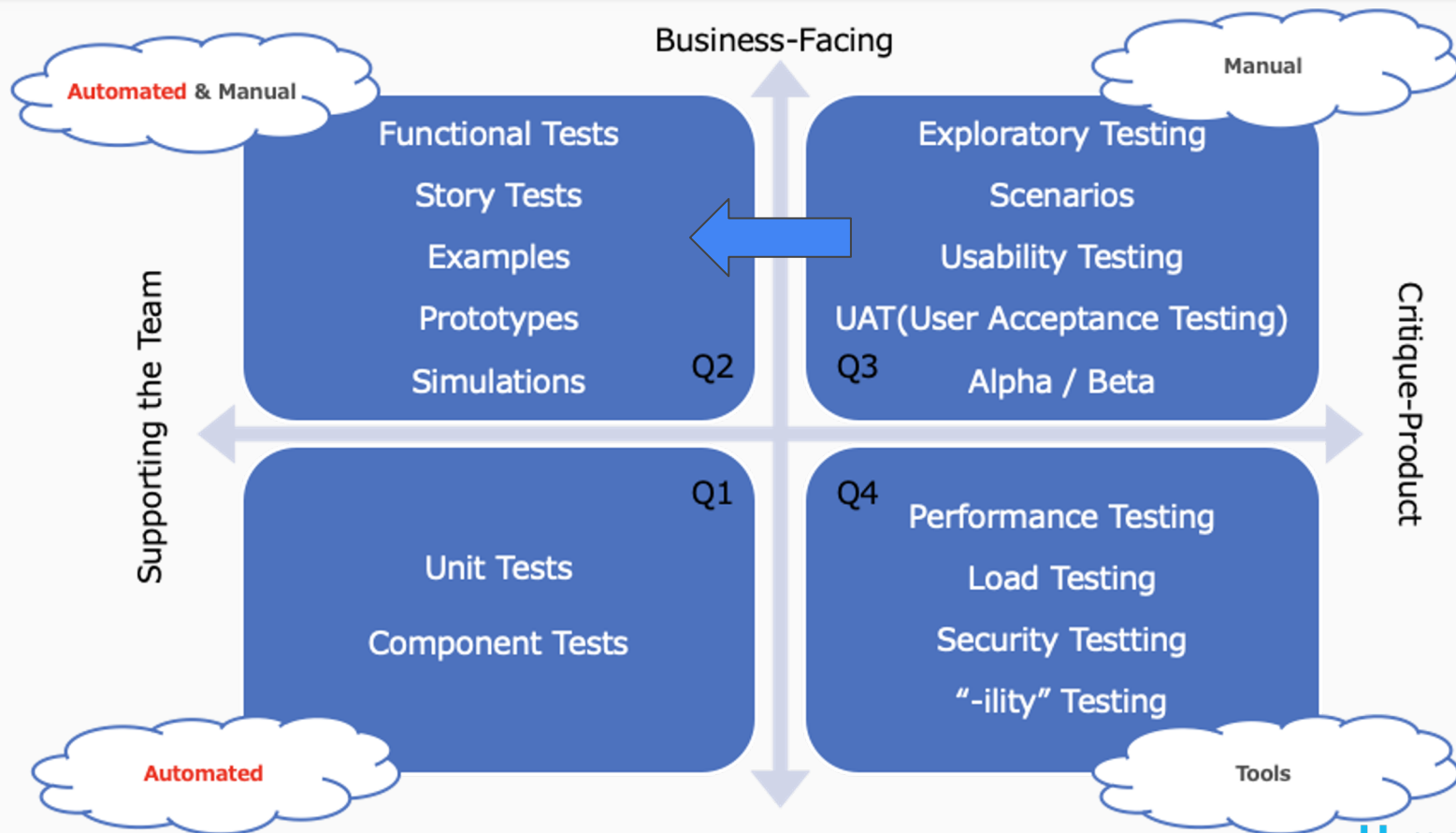
テスト実施後の フィードバックが重要

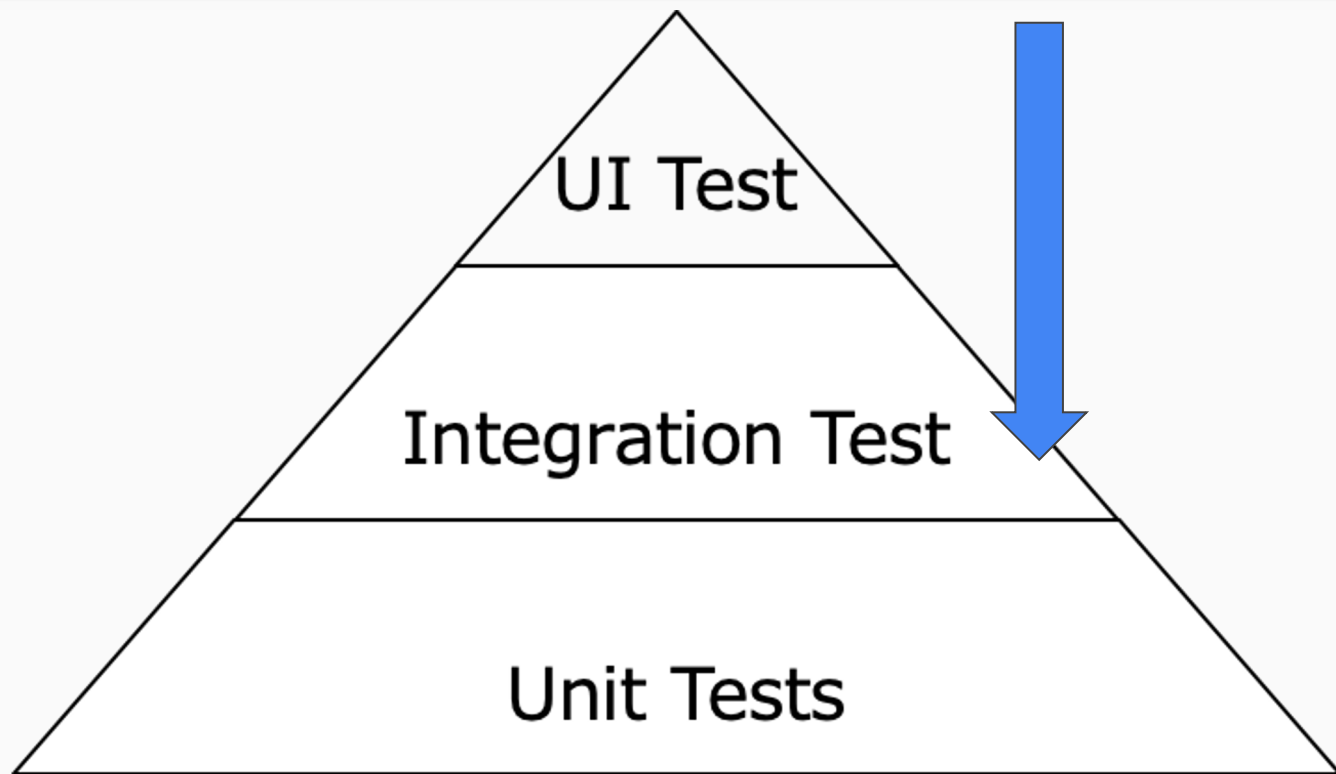
開発環境の変化



いつでもテスト実行可能

アジャイルテストの4 象限





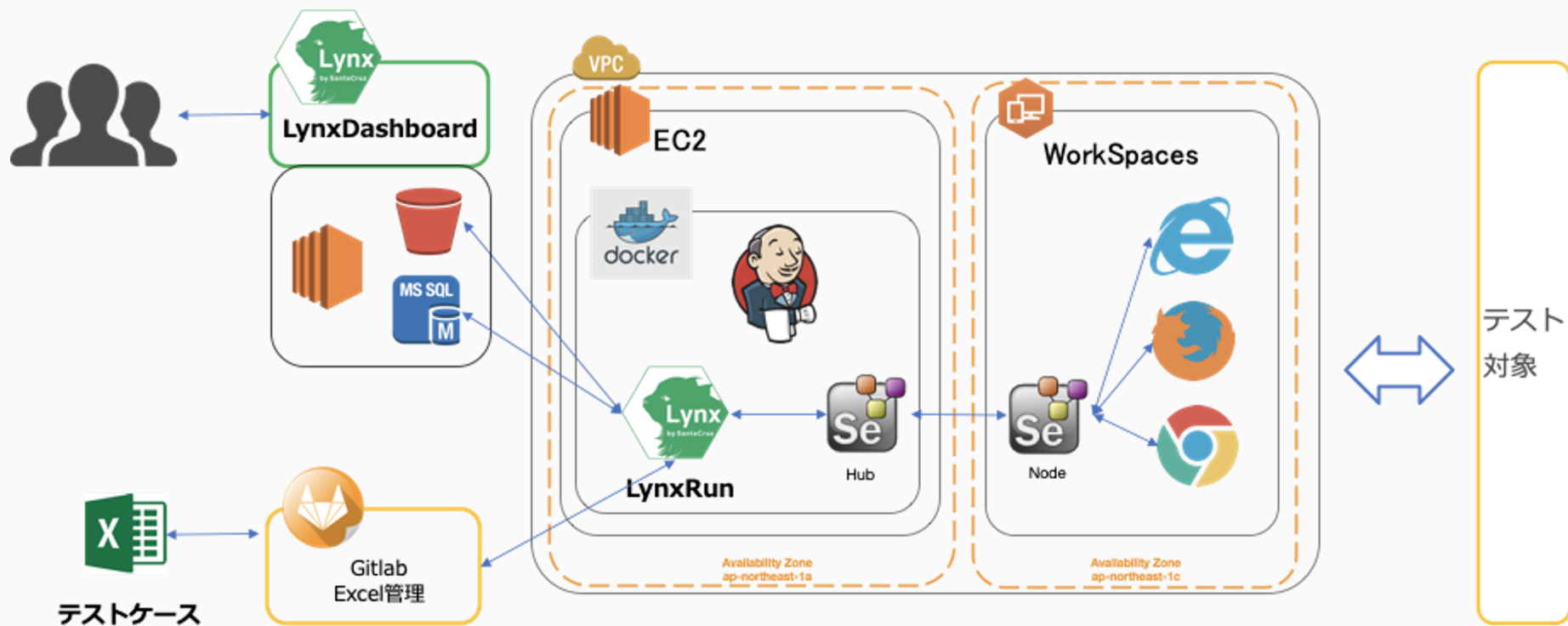
事例：テストスピードを改善

【ロジスティクスの場合】

- ・リリースサイクル：週1回
- ・テスト対象：本番環境、ステージング環境
- ・テスト機能：受付、人員管理等の主要機能
- ・自動テストの使われ方：ステージングリリース後、
自動テストが全て通るか確認
問題がなければ、その後本番環境へリリース



事例：簡易アーキテクチャ図



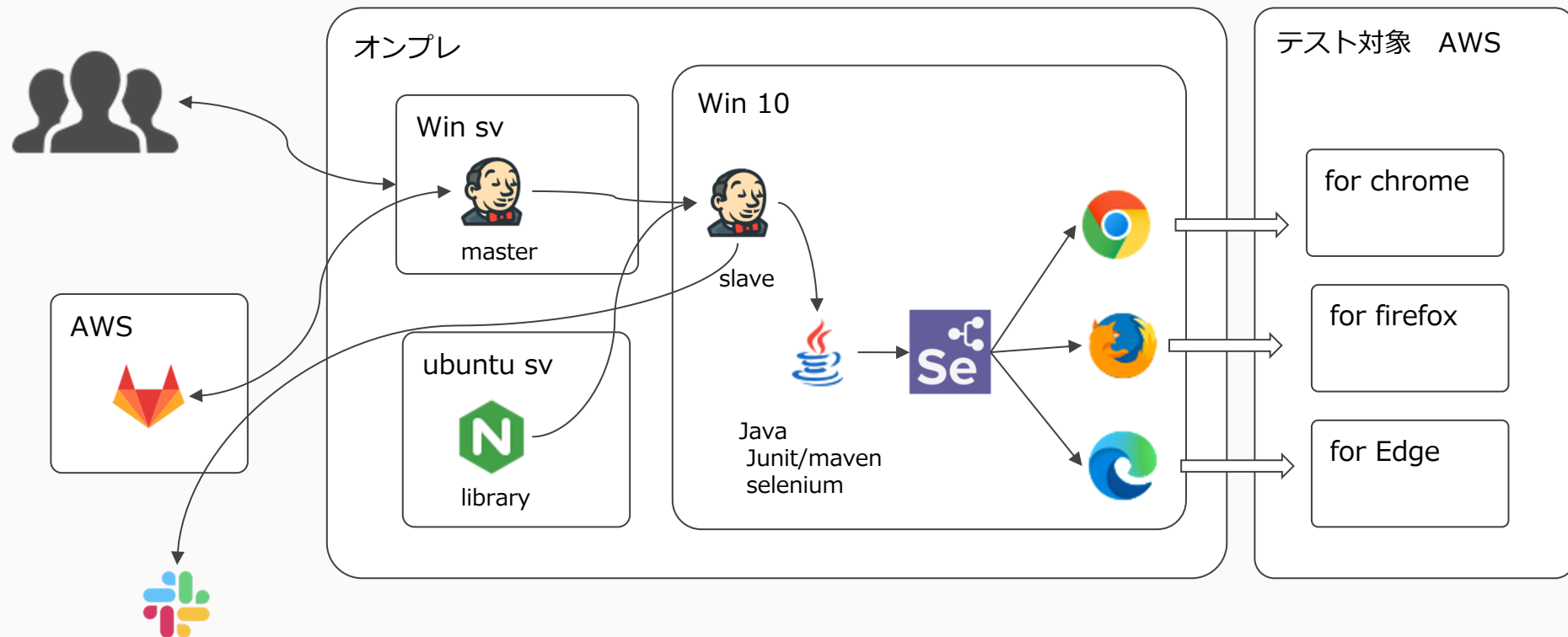
事例：テストスピードを改善

【CMSの場合】

- ・リリースサイクル：月 1 回
- ・スプリント：1 週間
- ・アジャイルチーム：朝会、振り返り会に参加
- ・テスト機能：すべての画面に遷移可能であること
主要機能、外部結合に問題がないこと
- ・テスト対象環境：自動テスト専用環境が 3 環境
- ・自動テストの使われ方：日次ビルドでテストが実行される



事例：簡易アーキテクチャ図



各パターンの組み合わせ

目的によって導入の仕方が変わり、パターンを組み合わせることもある！

例

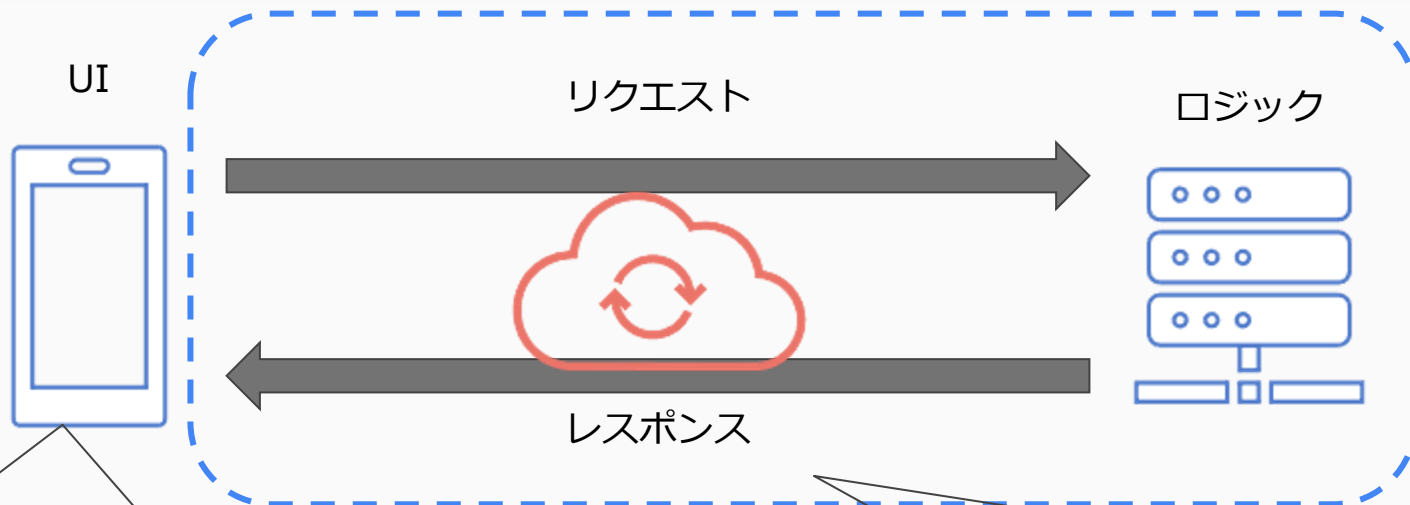
- ❖ テストスピード重視 × 繰り返し重視
- ❖ テストスピード重視 × インターフェース重視
- ❖ 繰り返し重視 × インターフェース重視
- ❖ ユーザの価値を重視 × テストスピード重視 × 手動テスト
- ❖ ユーザの価値を重視 × 繰り返し重視 × インターフェース重視

事例：ユーザの価値を重視 × テストスピード重視 × 手動テスト

アジャイルテストの台頭

イテレーション中	手動テスト	新機能テスト ＋ セッションベースドテスト
ビルド時	リグレッションテスト	各機能のテスト スピード重視
リリース時	シナリオテスト	要件網羅のテスト ユーザの価値を重視

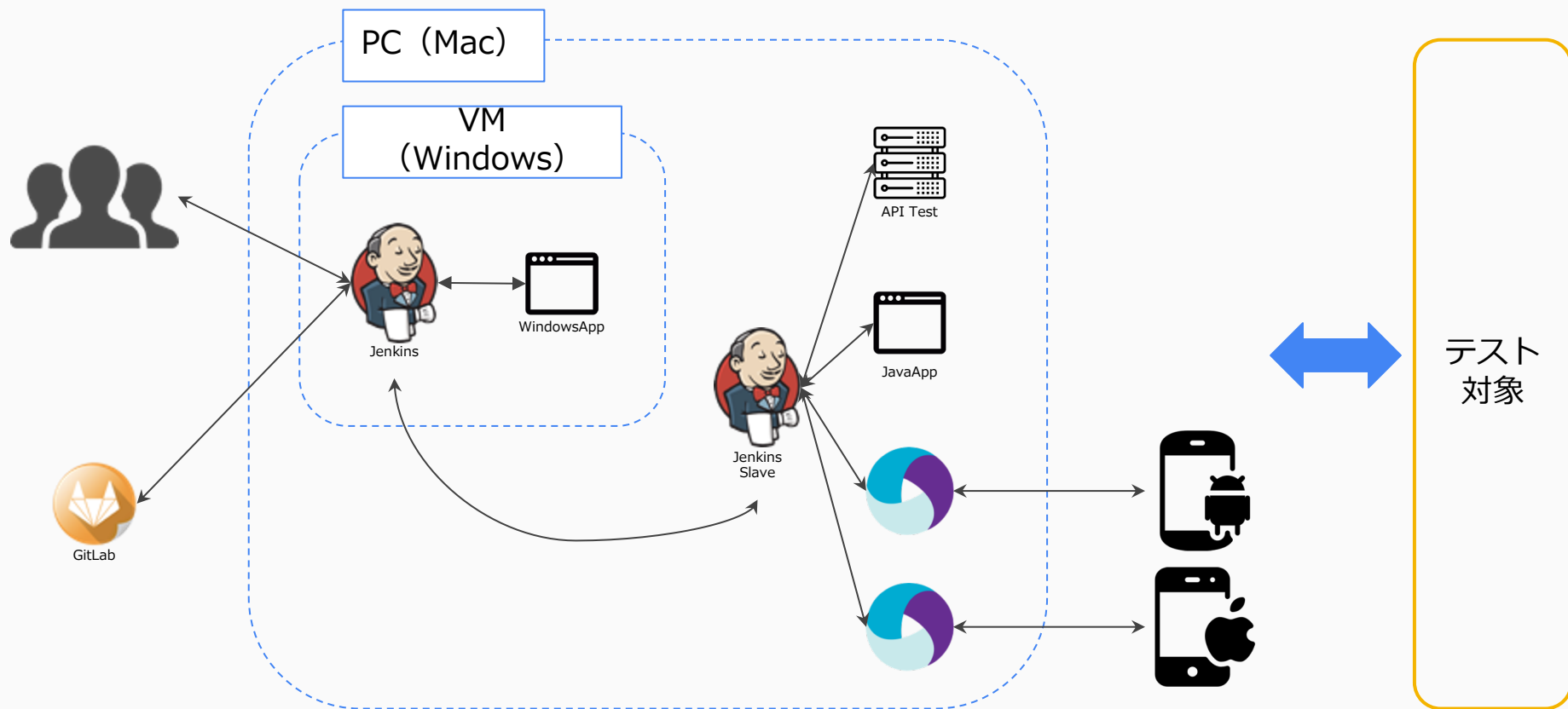
事例：ユーザの価値を重視 × 繰り返し重視 × インターフェース重視



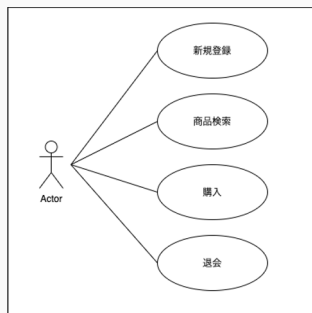
- ・テスト内容：**ユーザの基本操作**を自動化
対象は正常系のみ
繰り返し行うテストを自動化
- ・対象：SDKが組み込まれたサンプルアプリ
- ・デバイス：iOS、Android、Java、Windows
- ・テスト使用タイミング：リリース前、受け入れ時

- ・テスト内容：サーバーの設定確認
入力/出力のパターン網羅
- ・対象：**REST API**
- ・テスト使用タイミング：開発中、リリース前

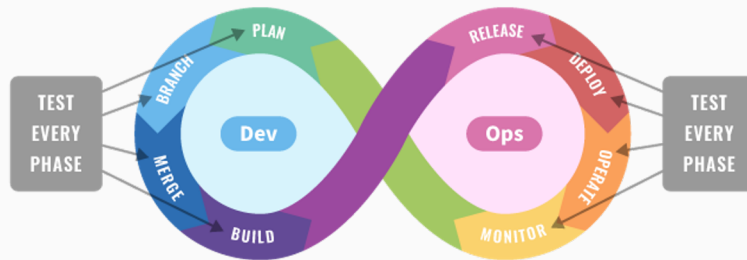
事例：簡易アーキテクチャ



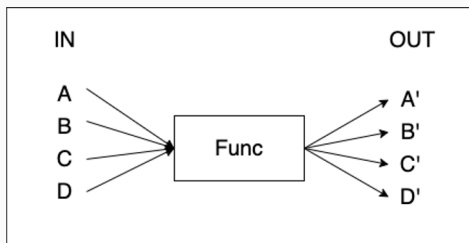
① ユーザの価値を重視



② テストスピード重視



③ 繰り返し重視

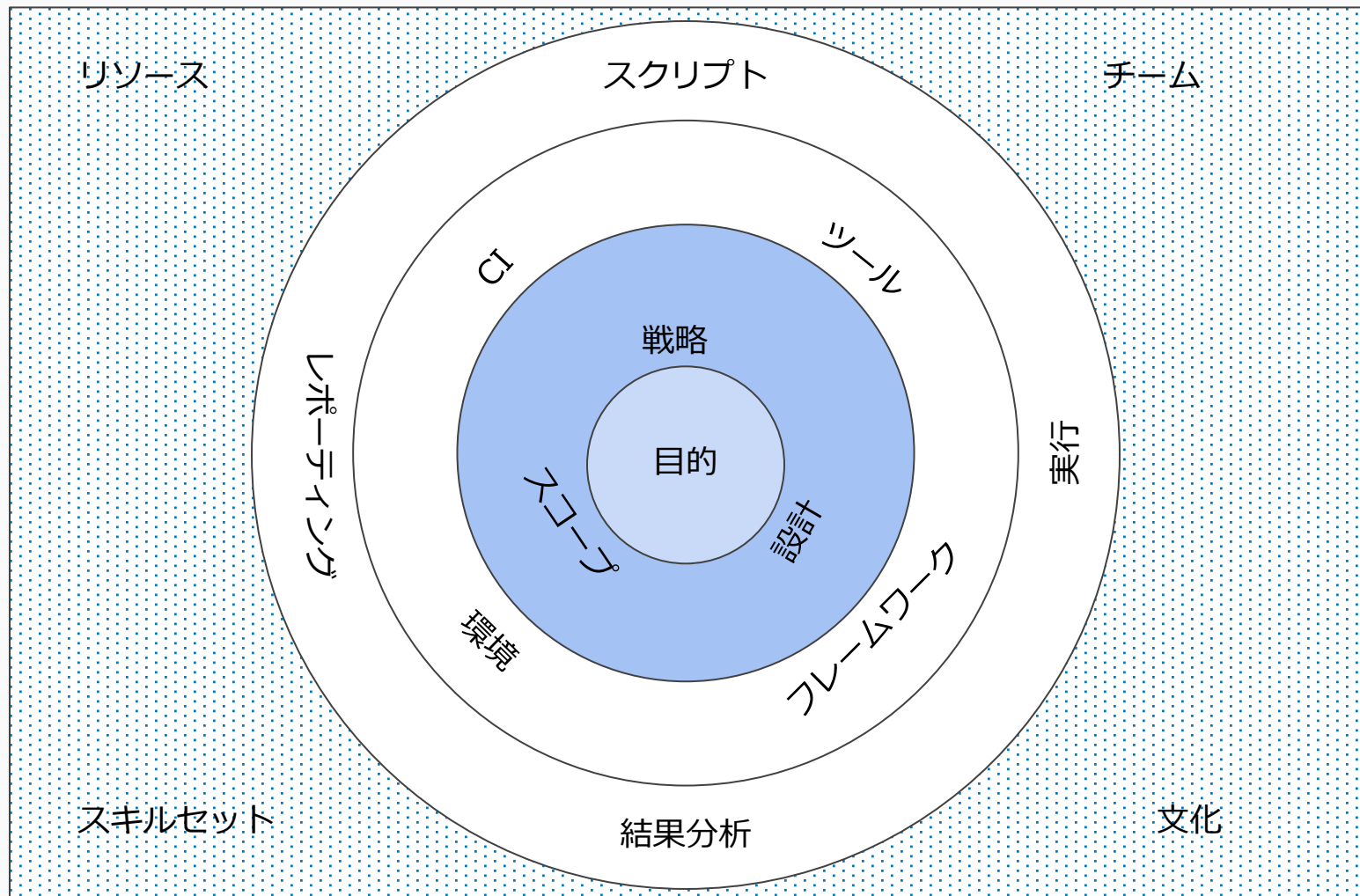


④ インターフェース重視

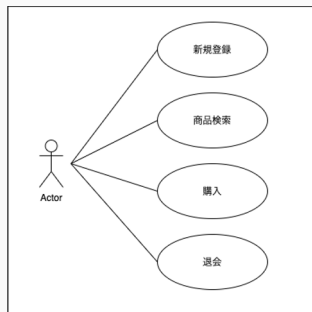


	①ユーザの価値重視	②スピード重視	③繰り返し重視	④インターフェース重視
目的	ユーザが使用する主要機能/要件が正常に動作することを確認する	イテレーション/リリースサイクルの中でテストする	繰り返しテスト（データ、マルチ環境）を自動化し、効率化する	インターフェースレベルでの入力/出力パターンが仕様書通りであり、機能が動作していることを確認する
特徴	・UIテスト（ハッピーパスから） ・本番環境に対してのテスト ・外部のテストエンジニアでも対応可能	・基本機能のテスト ・テスト専用環境がある ・パラレル実行する	・同一操作を何度も実施する必要があるテスト	・APIテスト ・正常系だけでなく異常系のパターンまで確認することが多い
メリット	・自動テストの動きが想像しやすく、作りやすい	・リリースの判断基準にできる ・1つのテストが短いため、開発者へのフィードバックが早い ・開発者に効果を実感してもらいやすい ・CI/CDに組み込みやすい	・1個スクリプトで様々なパターンのテストができる ・人が実施するよりもミスが防げる ・ROIが高い	・UIを通さずにテストできるため、自動テストがはやい
デメリット	・1つのテストケースが長くなるため、自動テストが不安定となりやすい	・自動テストの変更頻度が高い	・[データ]正確な入力値と期待値の組み合わせがないとテストできない ・[マルチ環境]複数環境の準備、メンテナンスが大変	-
注意点	・全てを自動化しない ・変更箇所が少ないところからテストを作成する ・主要機能を自動化するため、バグを発見しにくい	・開発者やプロジェクトメンバーとの協力が重要	-	・開発者やプロジェクトメンバーとの協力が重要

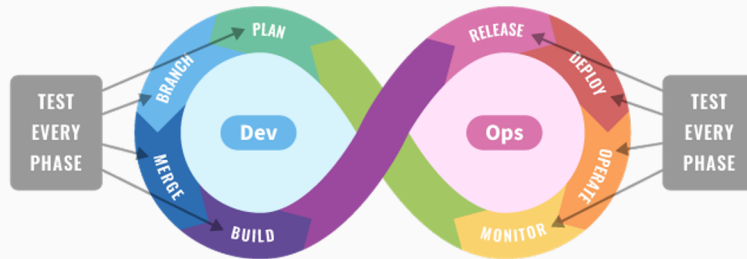
1. リソース
2. 優先順位
3. ツールと環境
4. テストに対する文化
5. スキルセット
6. プロセス



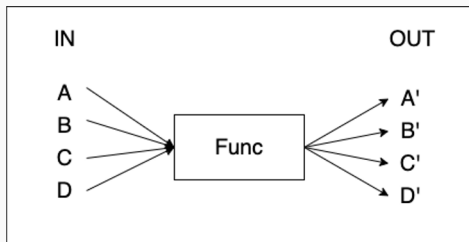
①ユーザの価値を重視



②テストスピード重視



③繰り返し重視



④インターフェース重視



① ユーザの価値を重視



② テストスピード重視



③ 繰り返し重視



④ インターフェース重視



Human Crest

仲間 募集してます

- ・ 企
- ・ IS
- ・ (

変革と研鑽

Human Crest

ご清聴ありがとうございました！

ご質問・ご意見 等は
バーチャルブースにて！

