JaSST'20 Niigata 事例紹介

WingArc 1st
The Data Empowerment Company

APIテストを自動化してリグレッションテストにしたら、安心で安全な開発ができて気持ちが楽になった

2020/09/28

ウイングアーク1st株式会社

伊藤　潤平

自己紹介

- 伊藤 潤平(@jp_110)
- ウイングアーク１ｓｔ株式会社
  - 帳票関連ソフトウェアの品質保証/プロセス改善
- 社外活動
  - SQiPシンポジウム2017、2019 論文発表
  - 品質管理学会員
  - JaSST Niigata 実行委員
  - Agile Testing Fellowship会員
  - Scrum Fest Osaka 新潟トラックオーナー
  - 翻訳活動
    - 書籍『Agile Testing Condensed』
    - YouTube『Appendix A: What We‘ve Learned Since Agile Testing -- Janet Gregory and Lisa Crispin.』
- ブログ
  - https://medium.com/@sadonosake
- 趣味
  - 日本酒（佐渡限定）
  - 筋トレ（健康志向）

ウイングアーク１ｓｔ株式会社

会社情報

- 商号　　：ウイングアーク１ｓｔ株式会社（WingArc1st Inc.）
- 所在地　：〒106-6235東京都港区六本木三丁目2番1号　六本木グランドタワー
- 創業　　：2004年3月
- 資本金　：2億円
- 事業内容：ツール・ミドルウエア製品の開発・販売、導入支援・保守サポートサービスの提供
- 従業員数：連結651人／単体561人（2020年2月末現在）

JAPAN

8拠点

GLOBAL

4拠点

帳票・文書管理ソリューション事業

業務に欠かせない帳票の効率的な運用を支援

ビジネスの現場から日々の暮らしのシーンまで、さまざまな場所で利用されている帳票。その設計・出力、運用を支えるソリューションおよびサービスを提供しています。業務処理におけるすべての帳票を対象としており、大手企業や官公庁を中心に多くの企業・団体に導入されています。

帳票の利用シーン

あらゆるシーンで利用される帳票の、安定的かつ効率的な出力・運用を支援しています。

ソフトウェアサービス

帳票基盤ソリューション

SVF

帳票クラウドサービス

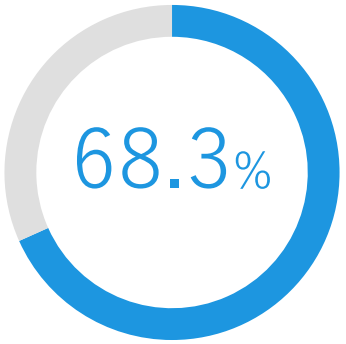
SVF Cloud

文書データ活用ソリューション

SPA

帳票クラウドサービス「SVF Cloud」の提供で、より幅広い企業に導入いただいています。

市場シェア (SVF)



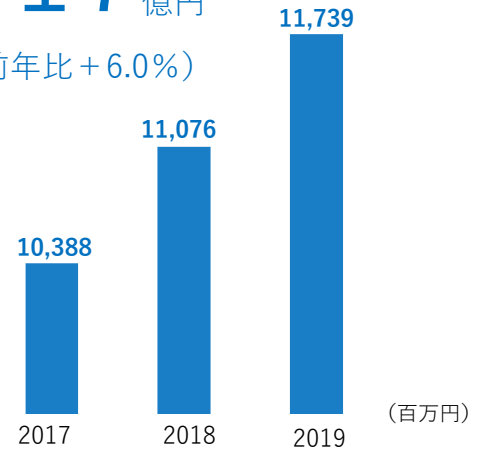
国内シェア第1位

※株式会社ミック経済研究所「帳票設計・運用製品の競合調査 2019年度版」（帳票運用製品ベンダー別出荷金額推移）をもとにウイングアーク１ｓｔが作成。日立製作所と帳票分野で提携し、「EUR」の資産を取得（2018年4月）したことに伴う数値。

累計導入社数は2万6000社にのぼり、10年以上にわたってトップシェアを維持しています。

販売実績

117億円
（前年比＋6.0%）



SVFの特長のひとつである「保守継続率の高さ」が、ソフトウェア基盤事業の安定的な成長を下支えしています。

データエンパワーメントソリューション事業

成長のための業務改革をデータ活用からサポート

限られた人員でも、最大限の成果をあげたい——多くの企業が抱えるそんな想いを実現するために必要なのは、データに基づく意思決定・行動です。適切なデータを、適切なタイミングに、適切な形で見られるよう環境を整備するための製品・ソリューション・サービスを提供しています。

実現できること

製造IoT・
生産性向上

働き方改革

営業改革

集計分析・可視化

ビジネスの現場が抱えるさまざまな課題。データ活用によって解決できるものは少なくありません。

ソフトウェアサービス

集計・分析プラットフォーム

Dr. Sum

BIダッシュボード

MOTIONBOARD

新ソリューション
自動化できるコミュニケーションツール

DEJIREN

企業内外の情報をデータ化・管理し、必要なときに集計・分析・可視化する製品・サービスを提供しています。

顧客満足度(Dr.Sum、MotionBoard)

日経コンピュータ 顧客満足度調査 2020-2021
データ分析・利活用支援ソフト／サービス 部門1位

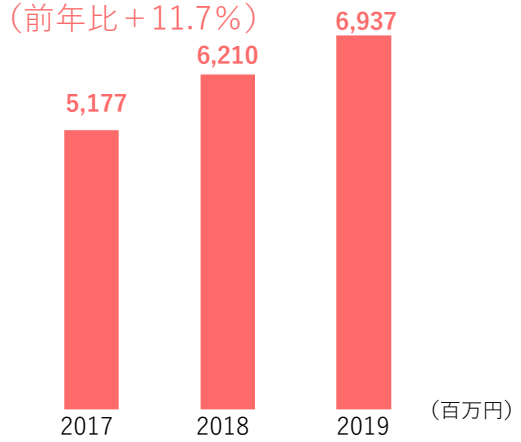
出所：富士キメラ総研「ソフトウェアビジネス新市場 2019年版」に基づき当社推定

BI市場の成長を牽引するリーディングカンパニーとなっています。

市場シェアは富士キメラ総研「ソフトウェアビジネス新市場 2019年版」BIツール・2018年度実績を基に、エンドユーザーが有用性及び生産性向上のために社内外のデータを集計・分析できるBIツールをオペレーショナルBIツールと定義し、2019年3月31日時点における当該企業群のパッケージ及びSaaSの販売金額の合計から当社オペレーショナルBIツール市場における市場シェアを約25%と独自に算出。市場シェアの算出から除外した企業はSAPジャパン、日本オラクル及び日本IBM

販売実績

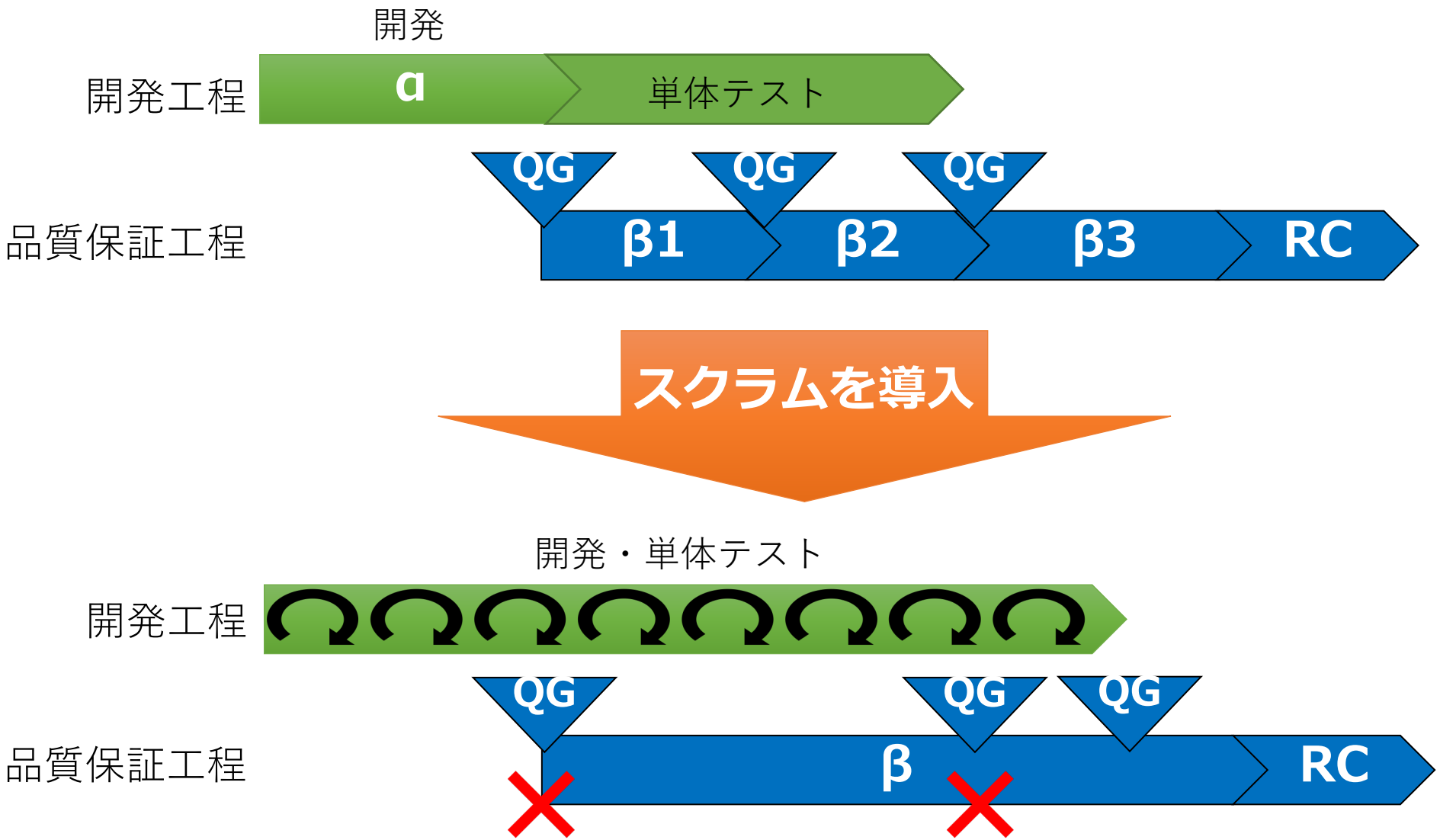
69億円
（前年比＋11.7%）



大規模での導入やソリューション化（複数の製品の組み合わせ）の推進により、成長が加速しています。

APIテストを自動化して
リグレッションテストにしたら
安心で安全な開発ができて
気持ちが楽になった

ウォーターフォール開発からアジャイル開発へ



問題点

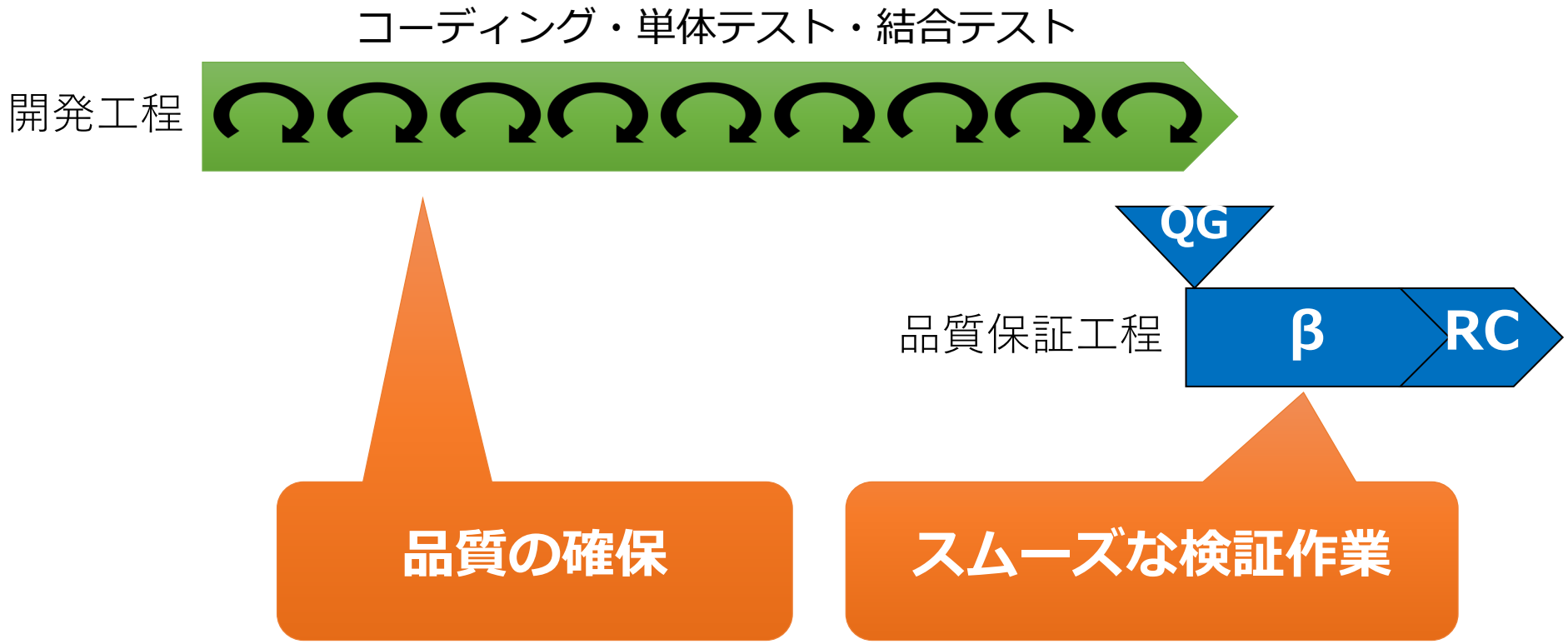
- 開発リリース物の品質が悪い。
- クオリティゲートを通過できない。
- 品質保証工程を開始できない。
- スケジュール遅延のプロダクトリスク発生。

開発チームと振り返り

- 開発者によってはコンポーネントテストを実行していない。
- 自分の実装環境のみのテストで、バックエンドとフロントエンドの結合テストレベルが不十分。
- 非機能は品質保証部門のメンバーがテストしてくれるだろうマインド。
- テストする時間がない。
- 品質に関する定義やルールがない。

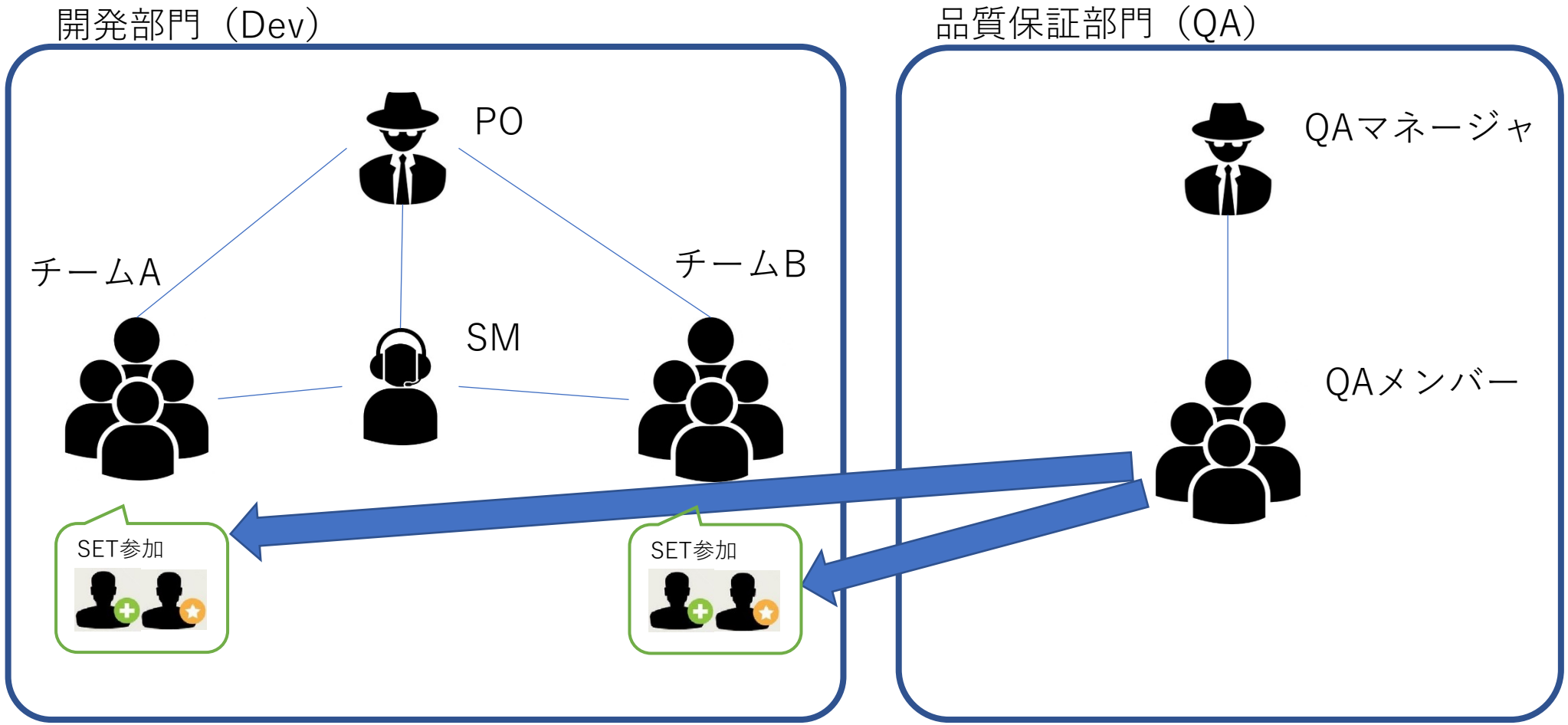
気付き

開発工程内で品質を高めて、スムーズな品質保証工程作業を実現しよう！



品質保証部門から開発部門にメンバーをアサイン！

品質保証部門からSoftware Engineer in Test（SET）としてメンバーを開発部門にアサイン

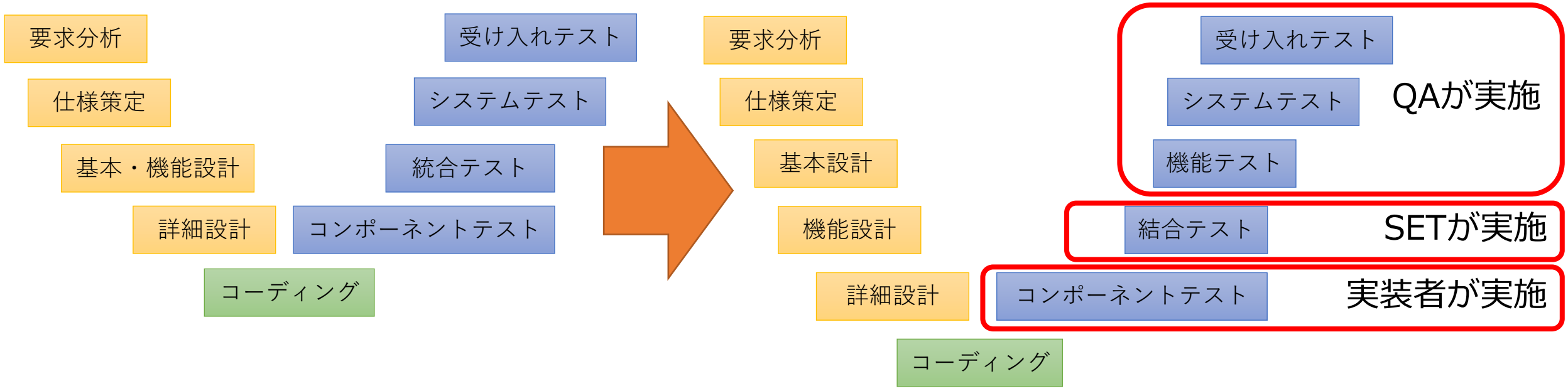


改善　その①

スクラムにテストのルールを作る！

改善　その①　テストのルールを作る

- テストレベルの定義付け



改善　その①　テストのルールを作る

- 「完成」の定義

Definition of Done

Done		unDone	
設計	基本設計が作成されていること	設計	QAからの要望があった場合、詳細設計が作成されていること
	機能仕様書が作成されていること	実装	アイコンやCSSが調整されていること
	設計レビューがされていること		メッセージが調整されていること
実装	スクランブル化されていること	テスト	サポート対象ブラウザでのテストされていること
	ソースコードレビューされていること		対象のサーバーOSでテストされていること
テスト	テスト設計されていること		性能テストされていること
	単体テストされていること		リグレッションテストされていること
	結合テストされていること	その他	リソースが翻訳されていること
	バグ0件であること		
	テストのエビデンスが残っていること		

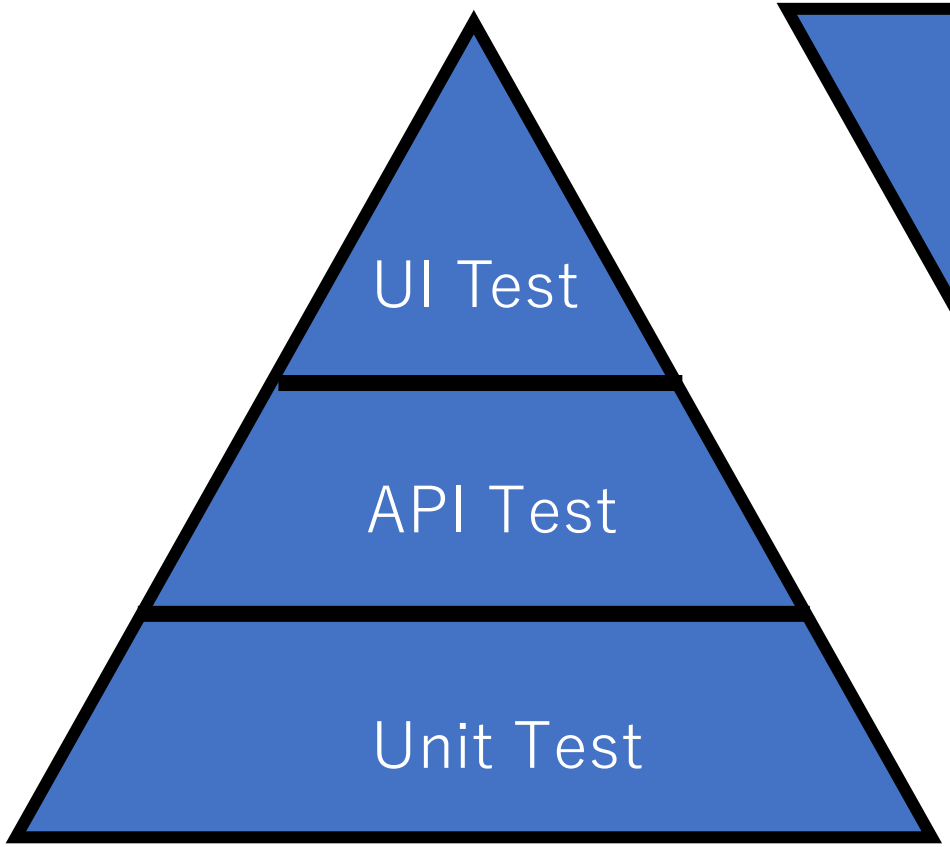
改善　その②

WebAPIテストの自動化

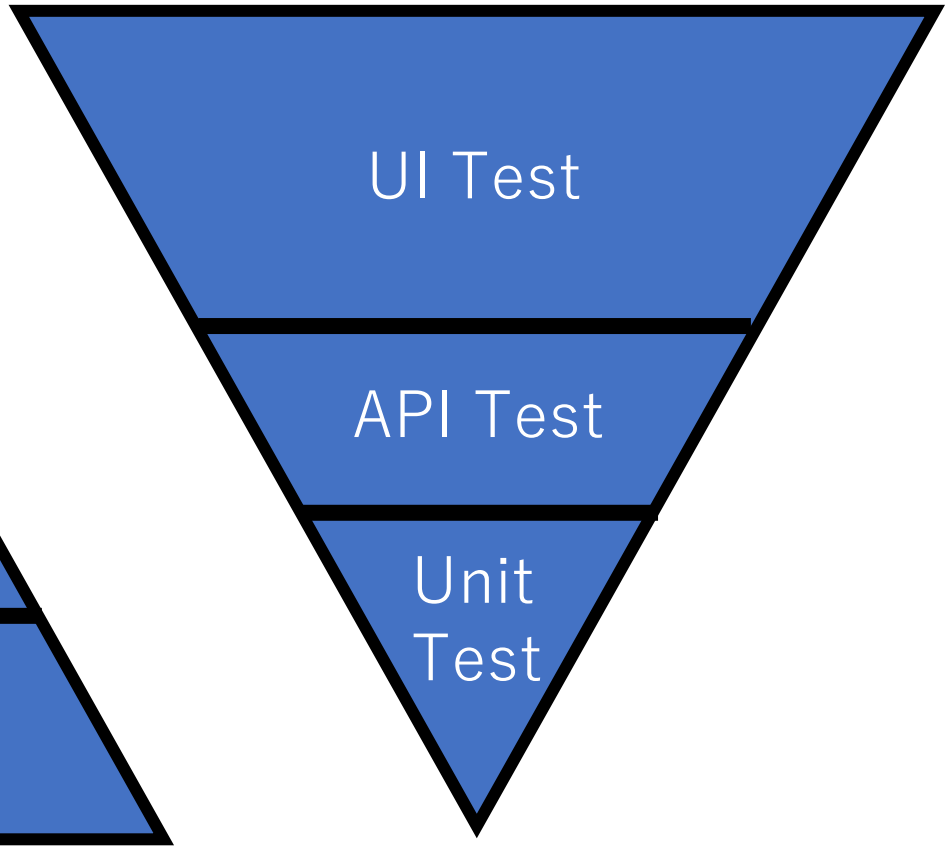
テストの理想と現実

WingArc 1ST
The Data Empowerment Company

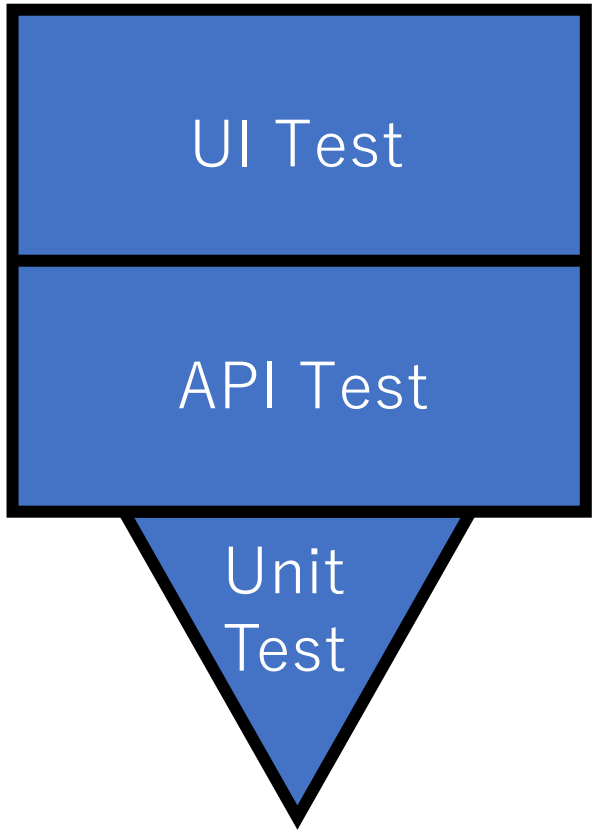
理想



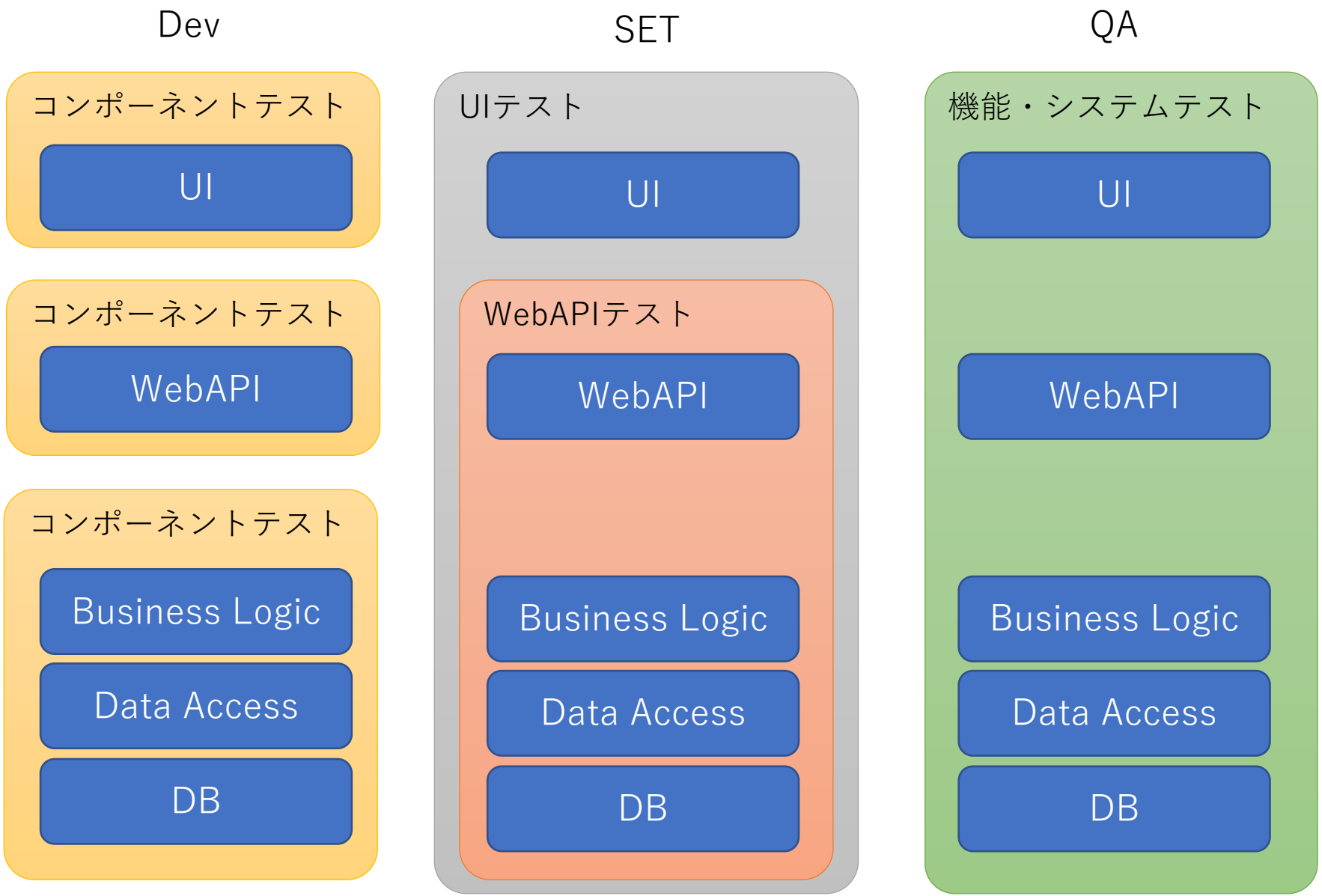
現実



目指すべき



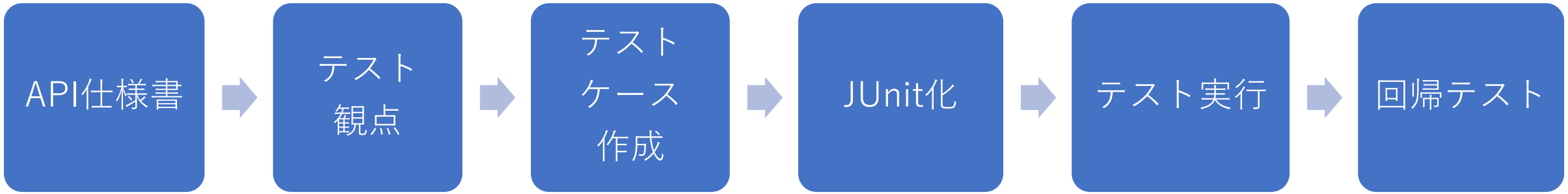
検証範囲



WebAPIテスト　テスト概要

WingArc 1ST
The Data Empowerment Company

- WebAPIを入り口として、WebAPIのバリデーション処理や、バックエンドの単機能を検証するテストセットとする。
（バックエンドのコア部分はDevメンバーのコンポーネントテストでカバーしていることを前提とする）
- JUnitを使用して自動化することにより、回帰的にテスト実行するリグレッションテストを目的とする。



WebAPIテスト　テスト設計方法

WingArc 1ST
The Data Empowerment Company

- 例えばAuth LoginのAPI
  - パラメーター

キー	値	省略時の値
user	ログインユーザーID	
domain	ドメイン名	Local
password	ログインパスワード	
tenant	テナントID	0

  - 戻り値

HTTPステータス	エラーコード	説明
200	0	正常終了（ログイン終了）
401	-1	ユーザー認証に失敗した場合
403	-20012	すでに別のユーザーIDでログイン中の場合
403	-9800	ライセンスエラーが発生した場合
403	-9801	有効期限切れの場合

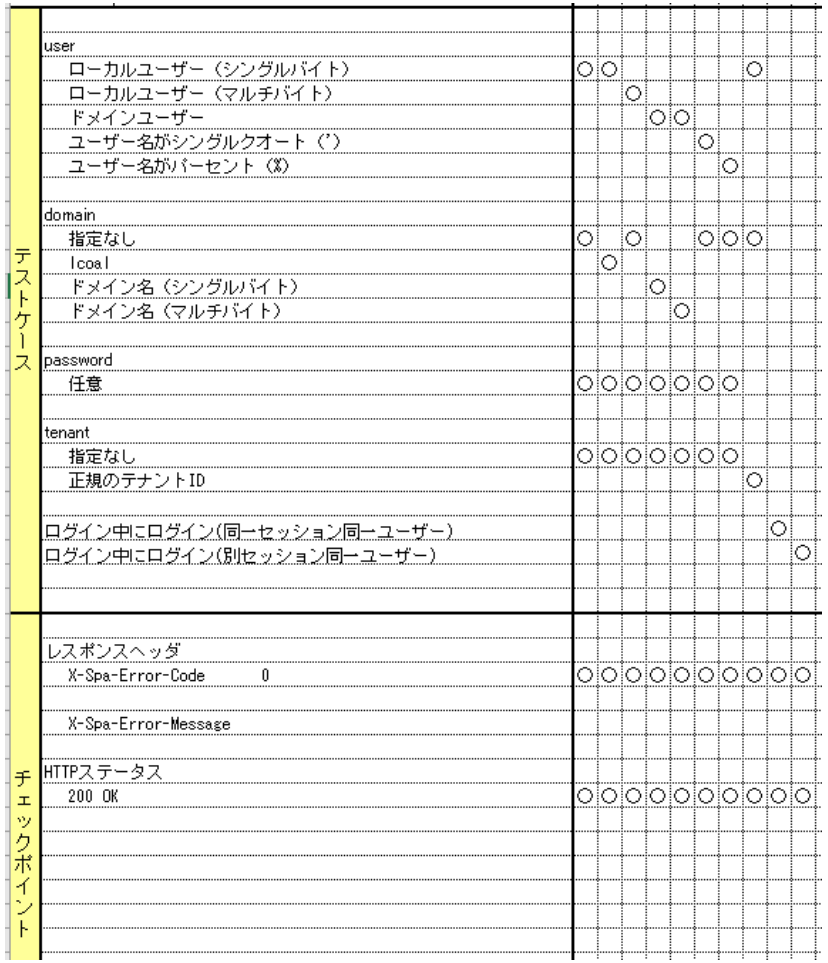
テスト設計方法

テスト観点を作成

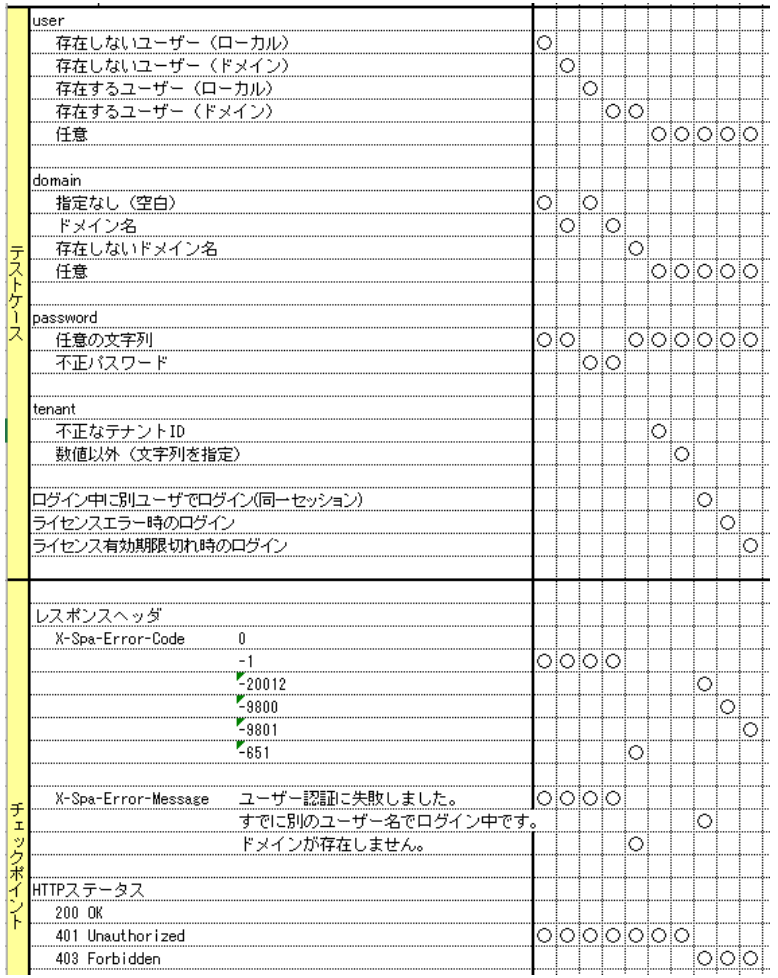
No.	テスト観点	テストケース
2.1	Auth Login	
AA.2.1.1	基本ログイン動作	
		user
		ローカルユーザー(シングル、マルチバイト)
		ドメインユーザー
		ユーザー名がシングルクオート、%
		domain
		指定なし、local
		ドメイン名(シングル、マルチバイト)
		password
		任意
		tenant
		指定なし
		正規のテナントID
		ログイン中にログイン(同一セッション同一ユーザー)
		ログイン中にログイン(別セッション同一ユーザー)
AA.2.1.2	ログイン（準正常）	
		user
		存在しないユーザー(ローカル、ドメイン)
		password
		パスワード不一致(ローカル、ドメイン)
		domain
		存在しないドメイン名
		tenant
		不正なテナントID
		数値以外(文字列)
		ログイン中に別ユーザーでログイン
		ライセンスエラー時のログイン
		ライセンス有効期限切れ時のログイン

テスト設計方法

テストケース（CL）を作成



テストケース	user										
	ローカルユーザー（シングルバイト）	○	○						○		
	ローカルユーザー（マルチバイト）			○							
	ドメインユーザー				○	○					
	ユーザー名がシングルクオート（'）						○				
	ユーザー名がパーセント（%）							○			
	domain										
	指定なし	○		○			○	○	○		
	Icoal		○								
	ドメイン名（シングルバイト）				○						
	ドメイン名（マルチバイト）					○					
	password										
	任意	○	○	○	○	○	○	○			
	tenant										
	指定なし	○	○	○	○	○	○	○			
	正規のテナントID								○		
	ログイン中にログイン(同一セッション同一ユーザー)									○	
	ログイン中にログイン(別セッション同一ユーザー)										○
チェックポイント	レスポンスヘッダ										
	X-Spa-Error-Code 0	○	○	○	○	○	○	○	○	○	○
	X-Spa-Error-Message										
	HTTPステータス										
	200 OK	○	○	○	○	○	○	○	○	○	○



テストケース	user									
	存在しないユーザー（ローカル）	○								
	存在しないユーザー（ドメイン）		○							
	存在するユーザー（ローカル）			○						
	存在するユーザー（ドメイン）				○	○				
	任意						○	○	○	○
	domain									
	指定なし（空白）	○		○						
	ドメイン名		○		○					
	存在しないドメイン名					○				
	任意						○	○	○	○
	password									
	任意の文字列	○	○			○	○	○	○	○
	不正パスワード			○	○					
	tenant									
	不正なテナントID						○			
	数値以外（文字列を指定）							○		
	ログイン中に別ユーザでログイン(同一セッション)								○	
	ライセンスエラー時のログイン									○
	ライセンス有効期限切れ時のログイン									
チェックポイント	レスポンスヘッダ									
	X-Spa-Error-Code 0									
	-1	○	○	○	○					
	-20012							○		
	-9800								○	
	-9801									○
	-651					○				
	X-Spa-Error-Message ユーザー認証に失敗しました。	○	○	○	○					
	すでに別のユーザー名でログイン中です。							○		
	ドメインが存在しません。					○				
	HTTPステータス									
	200 OK									
	401 Unauthorized	○	○	○	○	○	○	○		
	403 Forbidden								○	○	○

テスト設計方法

- テストケースをJUnit化

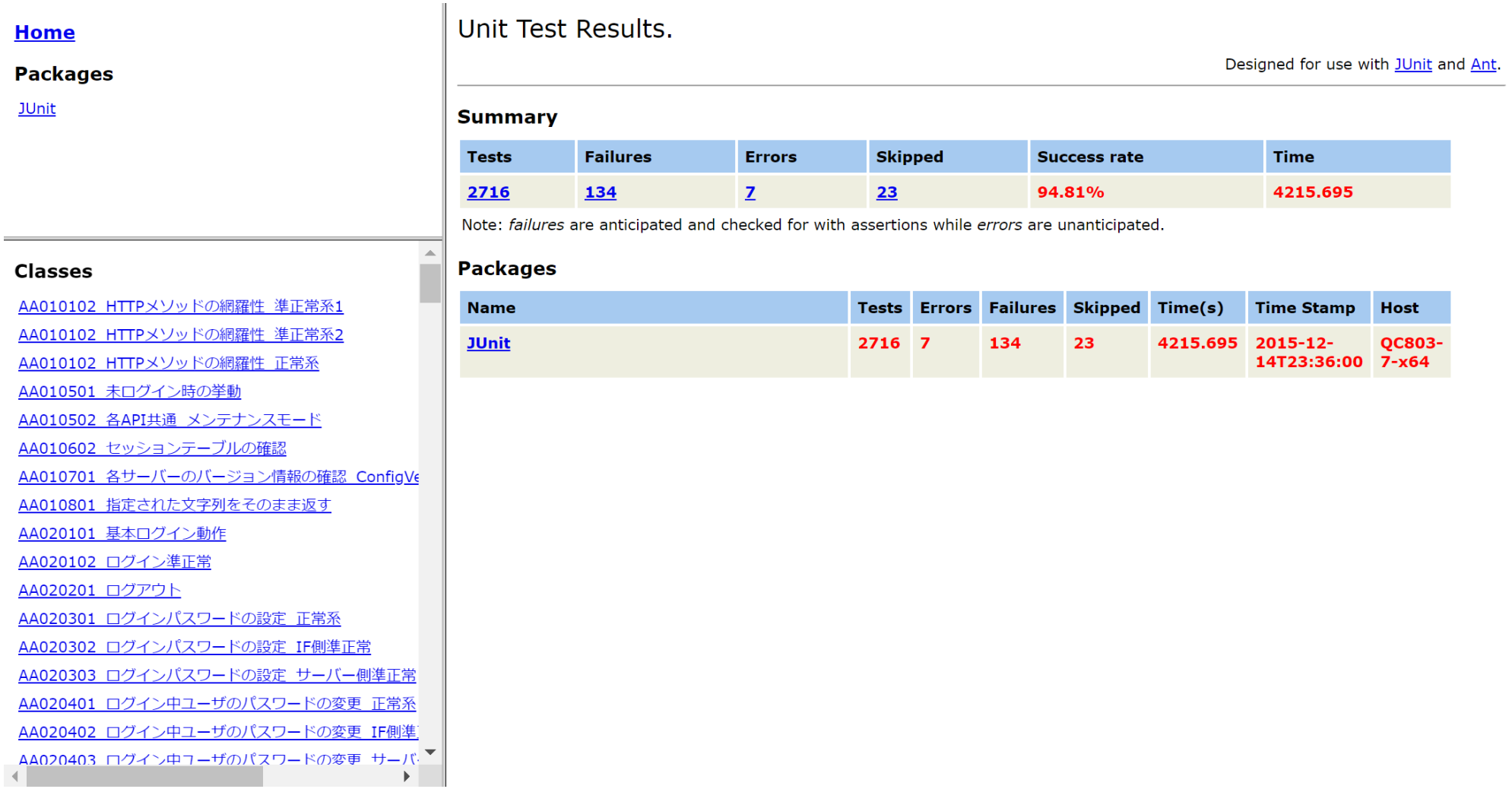
```java
/**
 * AA.2.1.1 基本ログイン動作
 */

@Test
public void AA2_1_10001() {
    System.out.println(System.getProperty("file.encoding"));
    System.out.println("----------AA2_1_10001----------");
    AuthLogin api = new AuthLogin();
    api.setSessionID(null);
    api.setMethod(Constants.Method_POST);
    api.setUrl(URL);
    api.setContentType(Constants.ContentType_urlencode);

    api.login("", "admin", "admin", "", Constants.PlainPasswd);
    sessionID = api.getSessionID();

    AssertUtil.okAssert(api);
    assertThat(api.getErrorUser(), is("[admin]"));
}


/**
 * OKの実行結果を確認
 *
 * @param com.wingarc.spa.regression.api apiBaseを継承したAPIオブジェクト
 */
public static void okAssert(apiBase api) {
    assertThat(api.getErrorCode(), is("[0]"));
    assertThat(api.getErrorMessage(), nullValue());
    assertThat(api.getResponseCode(), is(200));
}
```

```java
/**
 * AA.2.1.2 ログイン（準正常）エラーメッセージの内容をアサーションに反映させる
 */

@Test
public void AA2_1_20001() {
    System.out.println("----------AA2_1_20001----------");
    AuthLogin api = new AuthLogin();
    api.setSessionID(null);
    api.setMethod(Constants.Method_POST);
    api.setUrl(URL);
    api.setContentType(Constants.ContentType_urlencode);

    api.login("aaa", "aaa", "", Constants.PlainPasswd);

    AssertUtil.ngAssert1(api);
}
/**
 * NGの実行結果を確認
 * -1 ユーザー認証に失敗
 *
 * @param com.wingarc.spa.regression.api apiBaseを継承したAPIオブジェクト
 */
public static void ngAssert1(apiBase api) {
    ngAssert1_Core(api);
    assertThat(api.getResponseCode(), is(401));
    // assertThat(com.wingarc.spa.regression.api.getResponseMessage(), is("Unauthorized"));
}

/**
 * NGの実行結果を確認
 * -1 ユーザー認証に失敗
 *
 * @param com.wingarc.spa.regression.api apiBaseを継承したAPIオブジェクト
 */
private static void ngAssert1_Core(apiBase api) {
    assertThat(api.getErrorCode(), is("[-1]"));
    String spaLocale = api.getLocale() != null ? api.getLocale() : "";
    switch (spaLocale) {
        case en:
            assertThat(api.getErrorMessage(), is("[User authentication failed.]"));
            break;
        case zh:
            assertThat(api.getErrorMessage(), is("[验证用户失败。]"));
            break;
        default:
            assertThat(api.getErrorMessage(), is("[ユーザー認証に失敗しました。]"));
    }
    assertThat(api.getErrorUser(), nullValue());
}
```

テスト実施

WingArc 1ST
The Data Empowerment Company

- Antを利用したJunitのテスト結果をレポーティング



Home

Packages

JUnit

Classes

AA010102_HTTPメソッドの網羅性_準正常系1
AA010102_HTTPメソッドの網羅性_準正常系2
AA010102_HTTPメソッドの網羅性_正常系
AA010501_未ログイン時の挙動
AA010502_各API共通_メンテナンスモード
AA010602_セッションテーブルの確認
AA010701_各サーバーのバージョン情報の確認_ConfigVe
AA010801_指定された文字列をそのまま返す
AA020101_基本ログイン動作
AA020102_ログイン準正常
AA020201_ログアウト
AA020301_ログインパスワードの設定_正常系
AA020302_ログインパスワードの設定_IF側準正常
AA020303_ログインパスワードの設定_サーバー側準正常
AA020401_ログイン中ユーザのパスワードの変更_正常系
AA020402_ログイン中ユーザのパスワードの変更_IF側準
AA020403_ログイン中ユーザのパスワードの変更_サーバ

Unit Test Results.

Designed for use with JUnit and Ant.

Summary

Tests	Failures	Errors	Skipped	Success rate	Time
2716	134	7	23	94.81%	4215.695

Note: *failures* are anticipated and checked for with assertions while *errors* are unanticipated.

Packages

Name	Tests	Errors	Failures	Skipped	Time(s)	Time Stamp	Host
JUnit	2716	7	134	23	4215.695	2015-12-14T23:36:00	QC803-7-x64

テスト実施結果

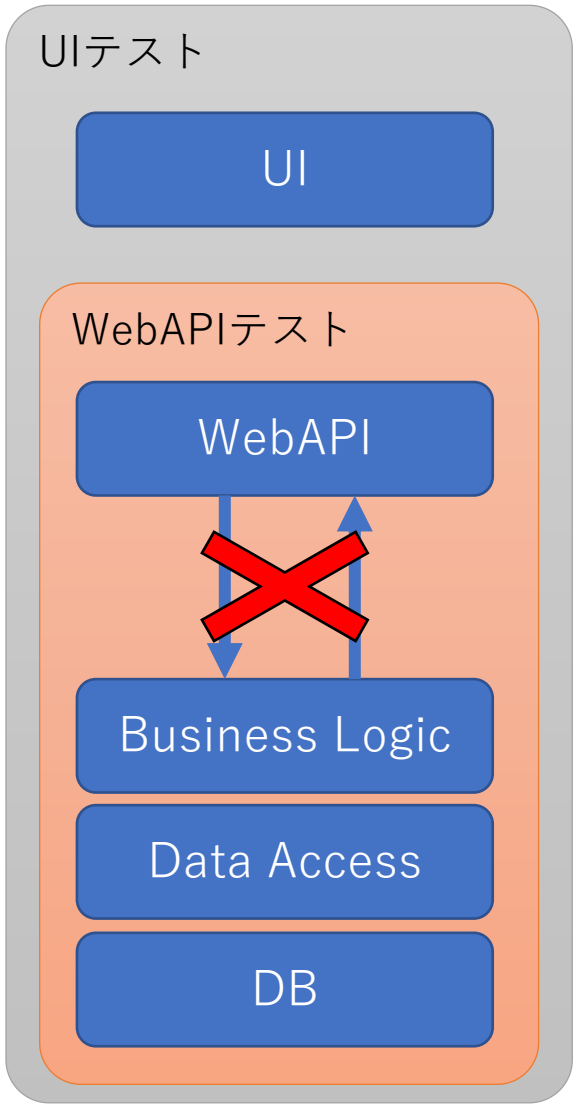
WingArc 1ST
The Data Empowerment Company

- WebAPIテストのテストケース数と発生したインシデント数

	結果
WebAPI数	約50
テストケース数	4,682TCs（観点数：437）
インシデント数	185件
ソースkステップ数	257k（参考：ソフトウェア本体518k）

効果

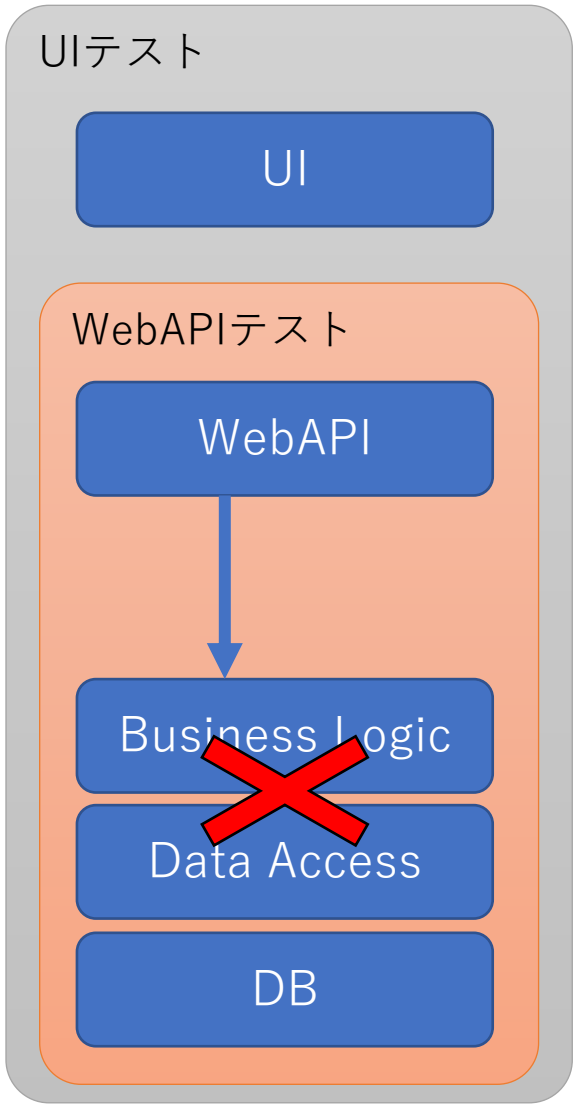
効果



- 重大な不具合を早期に発見することができた

WebAPIのパラメーター仕様と、バックエンドのパラメーター仕様に、開発者同士の認識の違いがあることが分かった。

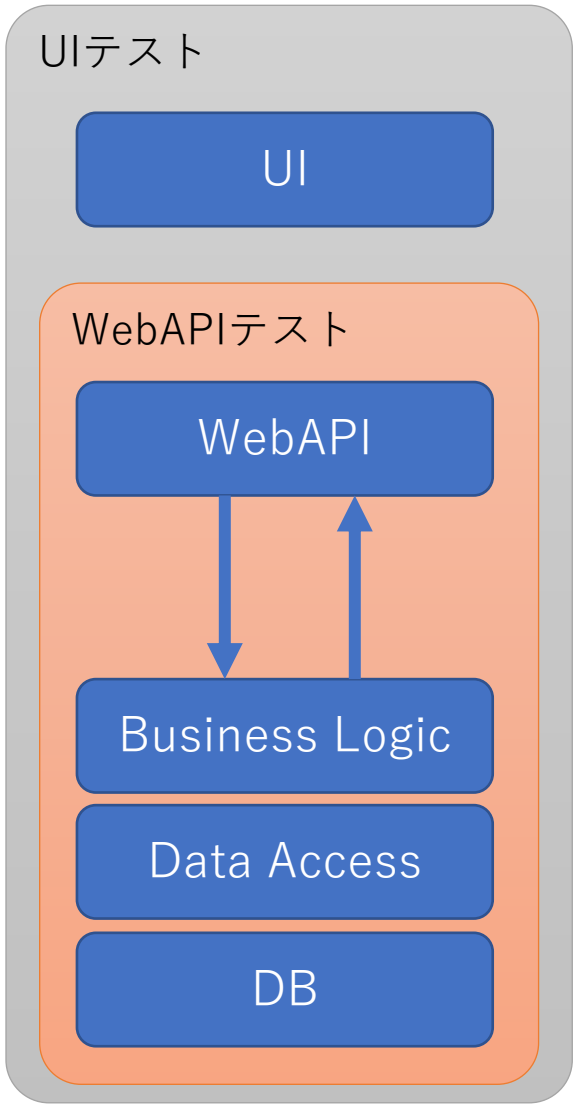
効果



- ユーザー利用を想定した準正常テスト

WebAPIから準正常系のリクエストを送ると、バックエンドで想定外の動作をする等の新たな気付きを得られた。（QA視点でのテストのよる安心感）

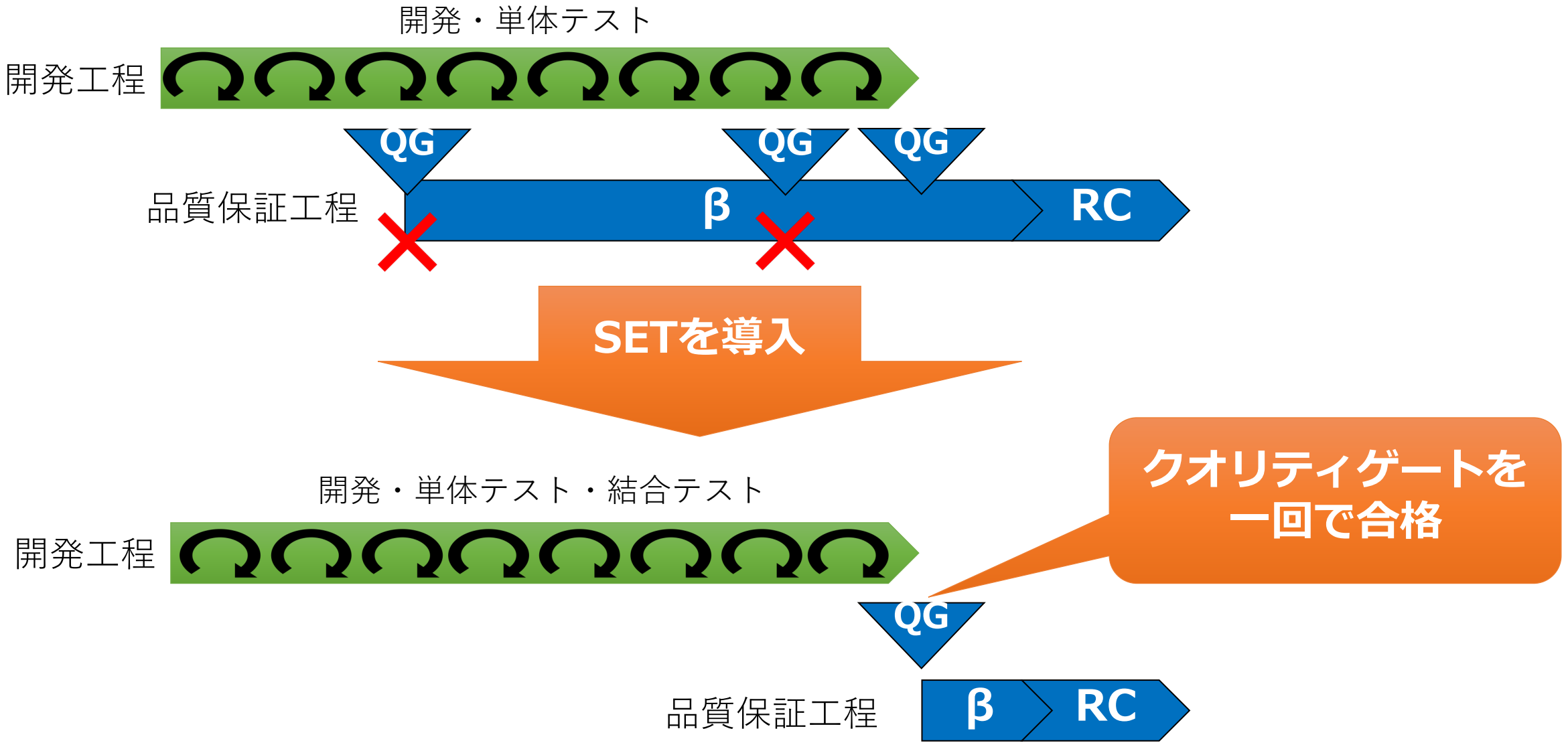
効果



- リグレッションテストとして確立

WebAPIテストをリグレッションテストとしてスプリント毎に実行すると、仕様変更やリファクタリングの影響箇所で不具合が発見できた。（デグレードを発見できる安全性）

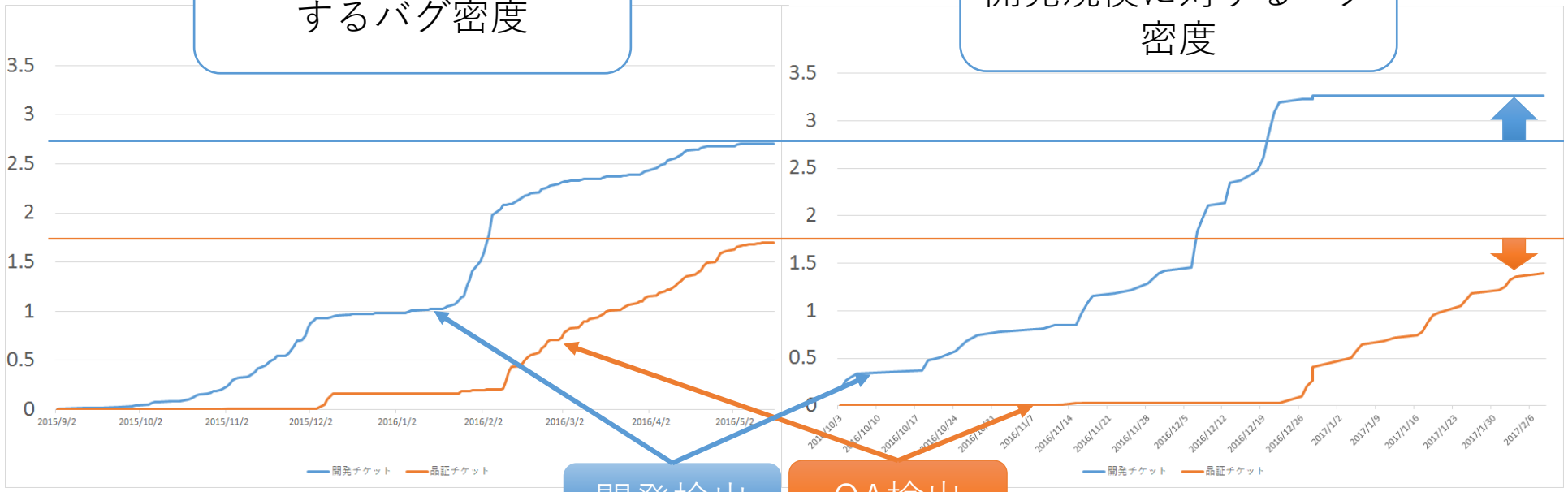
結果



品質向上結果

初期版の開発規模に対するバグ密度

バージョンアップ版の開発規模に対するバグ密度



開発検出

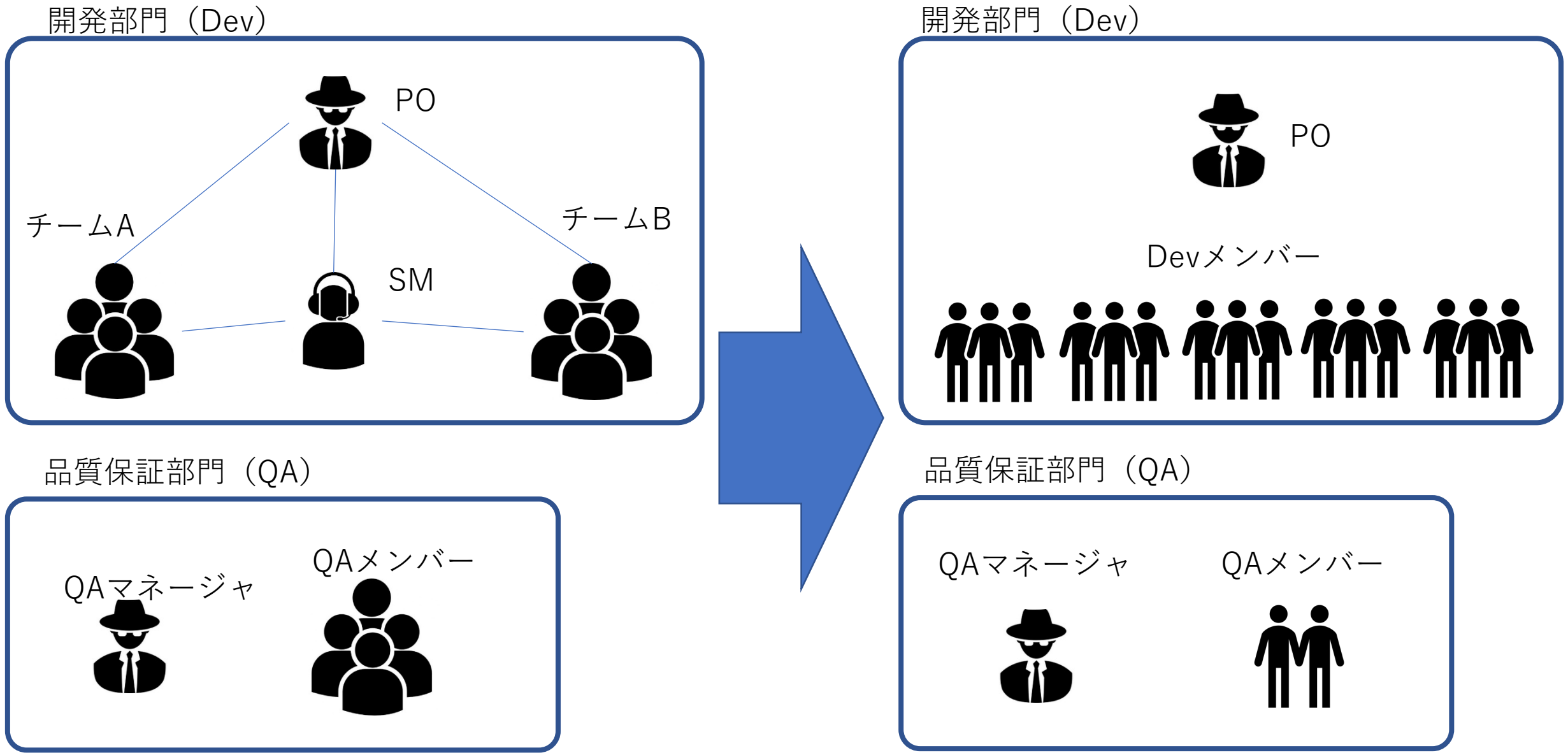
QA検出

開発工程で検出された不具合が増え、
品質保証工程での検出は減った。
つまり品質は開発段階から作りこまれている。

現在のWebAPIテストの状況

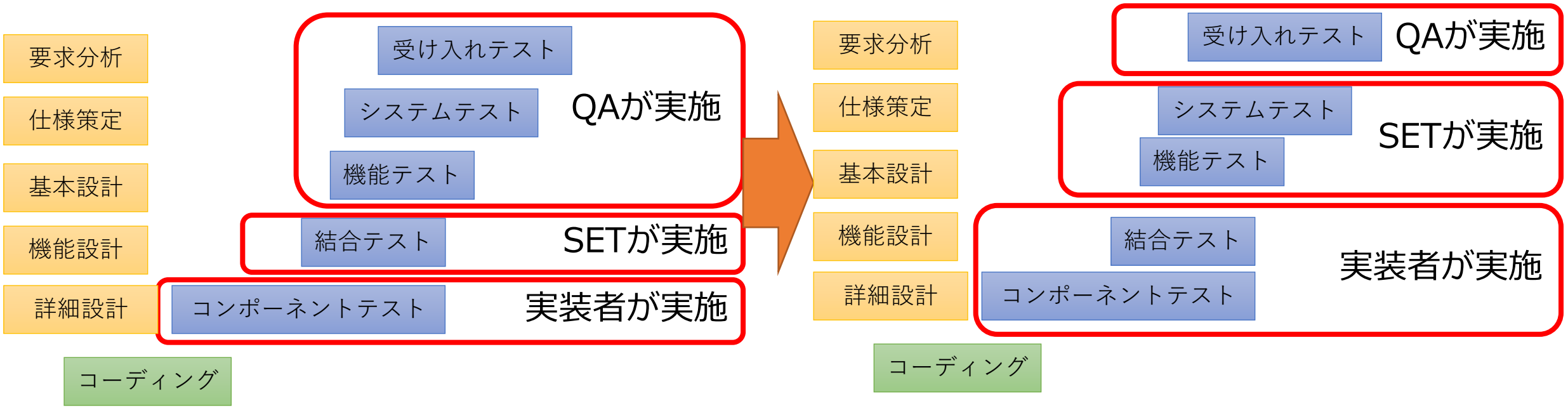
スクラムから独自のアジャイル開発へ

体制の変更

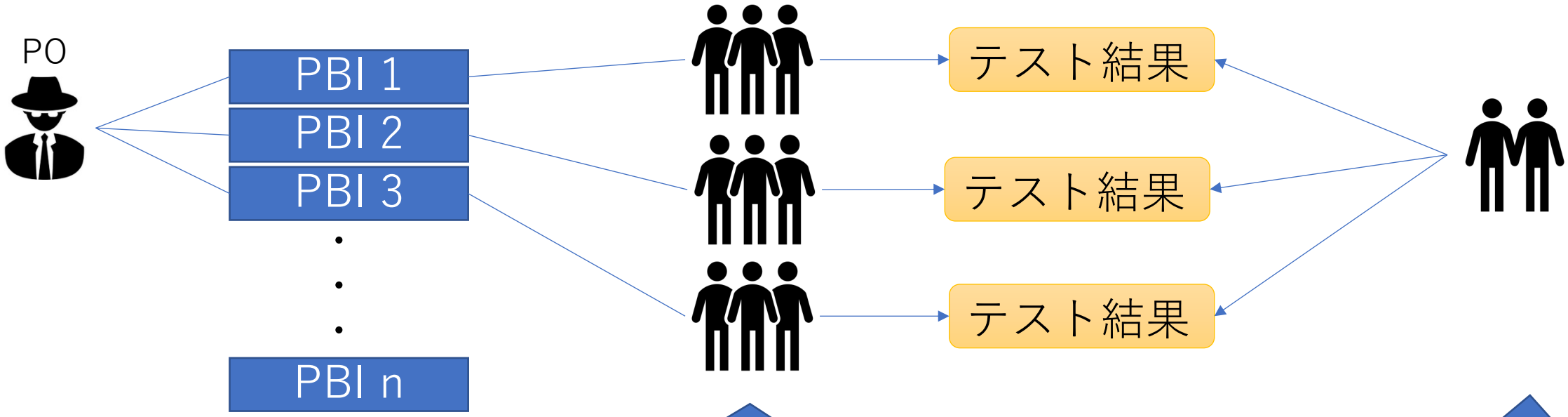


開発プロセスの変更

- 開発部門（Dev）
  - 実装からテストまですべてを実施する
- 品質保証部門（QA）
  - テストが計画通り行われ、品質が確保された状態であることを監査する



開発プロセスの変更



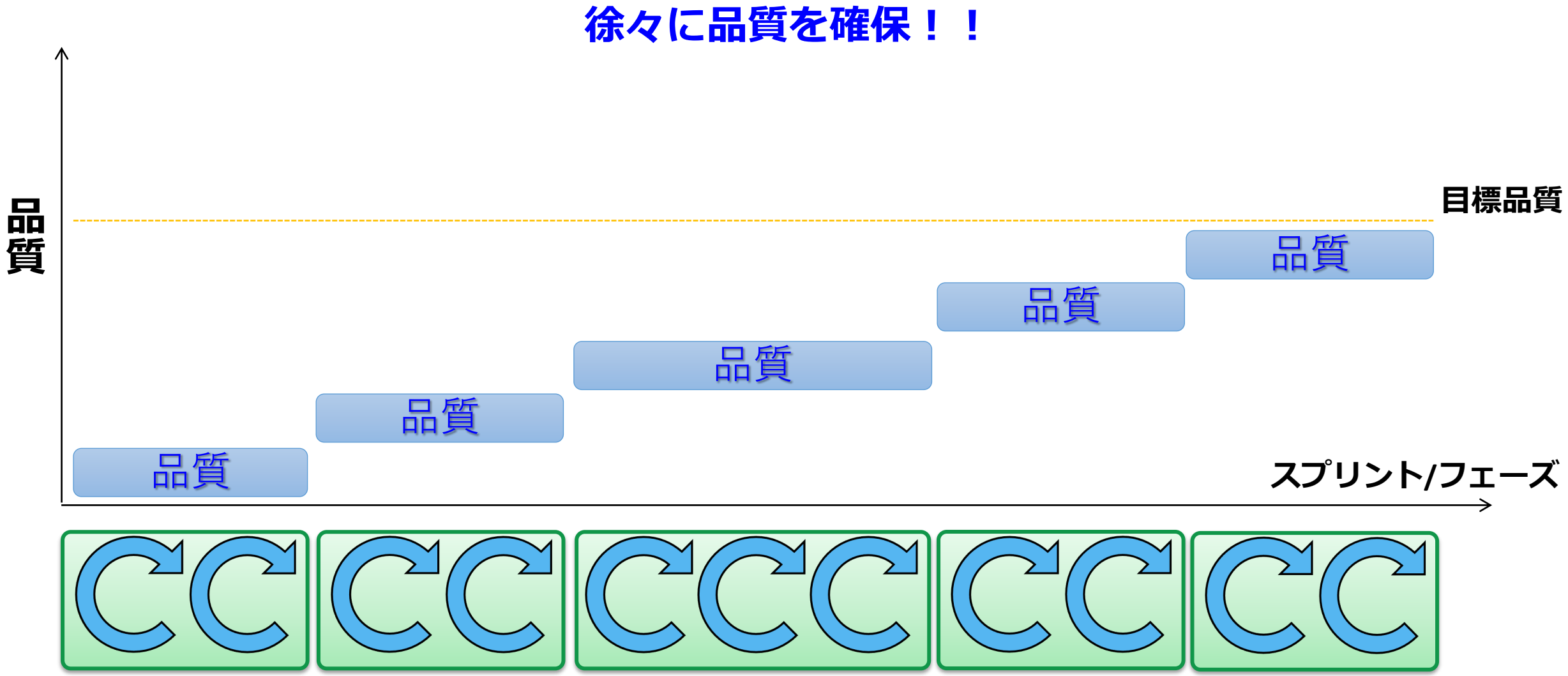
POから要求をPBIとしてバックログで管理

PBI毎にDev+SETがチームとなり、実装、コンポーネントテスト、APIテスト、機能テスト、システムテストまでを実施する

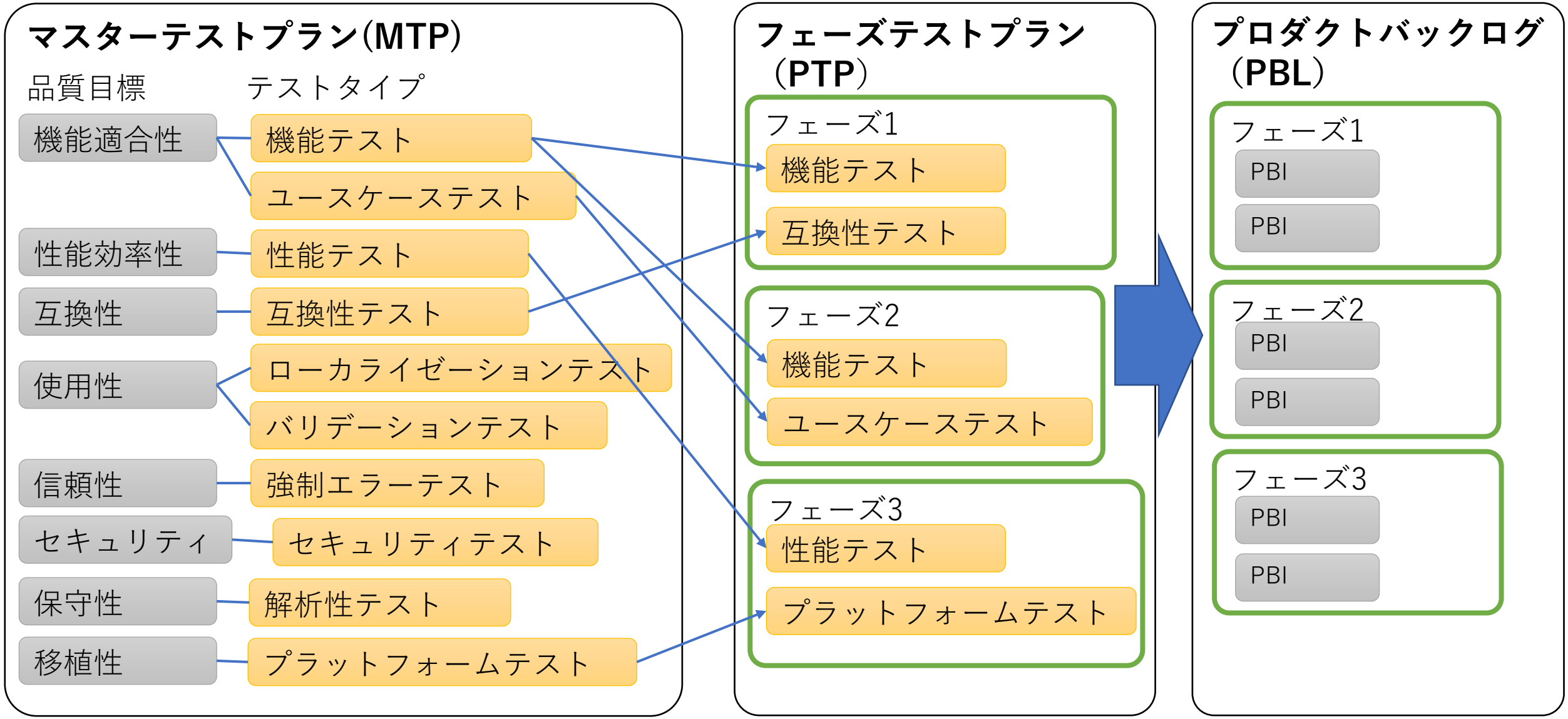
Dev+SETがチームが実施したテスト結果に対して、QAメンバーが監査する

監査型のQA

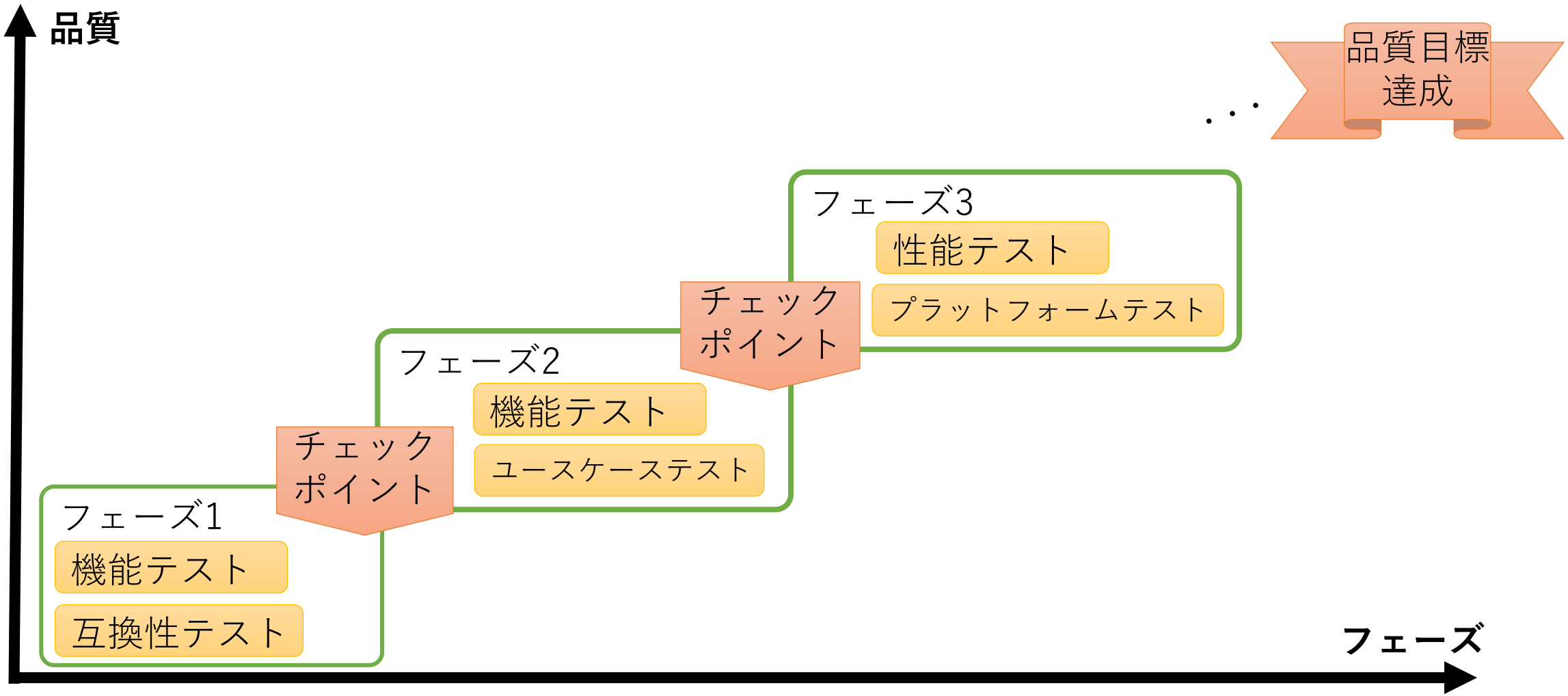
段階的な品質の積み上げ



戦略的テスト計画の作成



フェーズと品質ゲートの設置による段階的品質の積み上げ



WebAPIテスト課題

- テストケース数が膨大になりテスト実行に時間がかかる
  - 1プラットフォームの実行時間：30時間
  - テストケース数：4,682TCs　→　17,114TCs
  - ソースkステップ数：257k　→　853k
- 実装者任せになっている
  - 管理が大変
  - 属人化が進んでいる
- フレイキーが増えている
  - テスト結果分析という新たなタスクが生まれる
  - テストコード自体のリファクタリングが必要

The Data Empowerment Company

データに価値を、企業にイノベーションを。