組織的にシステムテスト自動化を推進する体制の構築

株式会社エビデント

EVIDENT

要件・ソフトウェア開発

〇内山 守　　石野 武　　江良　徹

E-mail：mamoru.uchiyama@olympus.com

アジェンダ

01　はじめに

02　自動化における問題・解決策

03　自動化の活動内容

04　結果

05　今後の課題

エビデント設立

Scientific Solutions Division

EVIDENT

開発製品

Life Science Solutions

Life Science Research Solutions

Cell Culture Solutions

Clinical Research Solutions

Education Solutions

Industrial Solutions

Industrial Microscope Solutions

Videoscope and Borescope Solutions

NDT Solutions

XRF Solutions

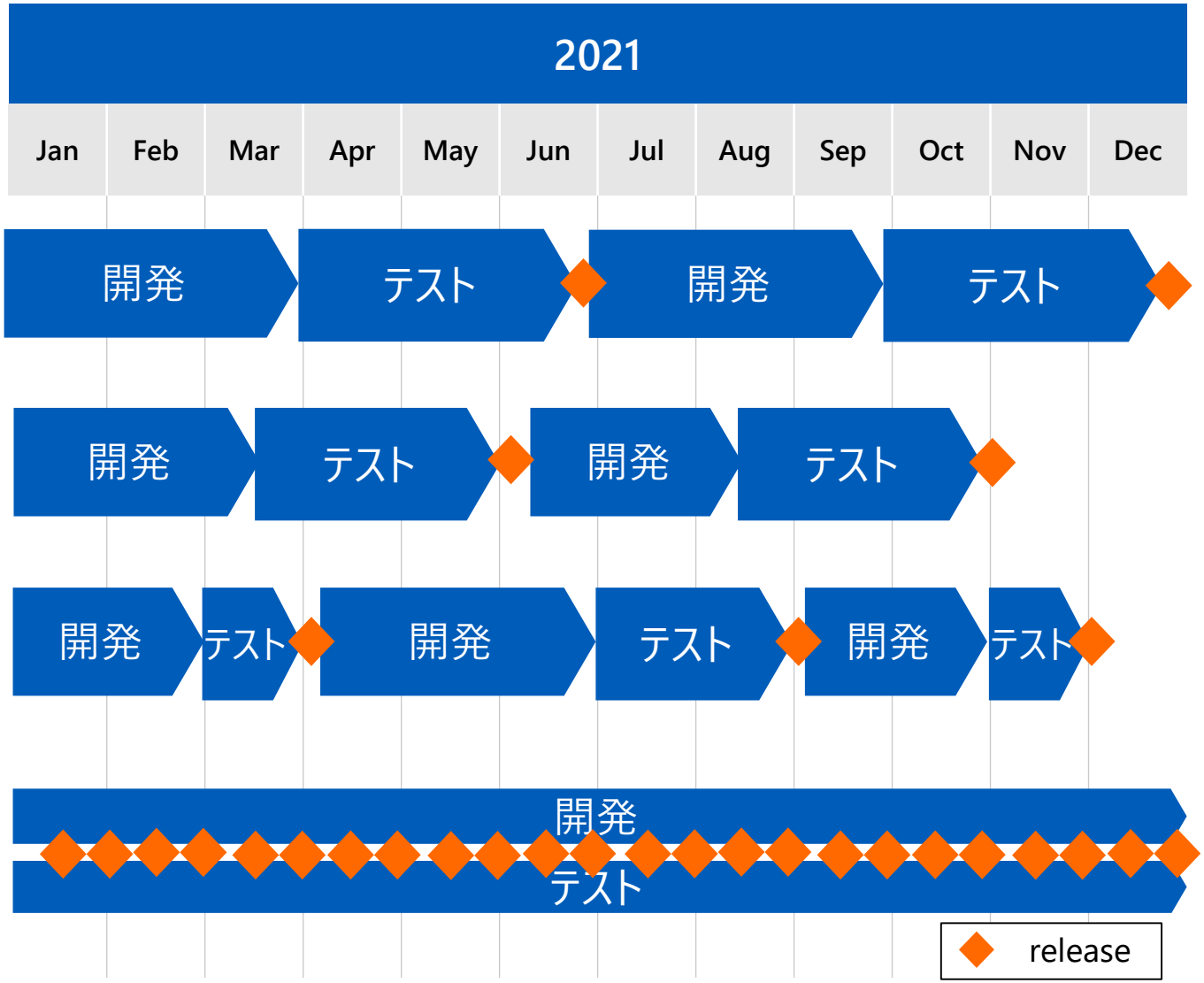
開発体制

プロジェクトA
開発体制：18人
(テスト：9人)

プロジェクトB
開発体制：15人
(テスト：4人)

プロジェクトC
開発体制：6人
(テスト：3人)

プロジェクトD
開発体制：10人
(テスト：2人)



我々を取り巻く状況

ソフトウェア開発の現状

①開発サイクルの短期化

②リリースサイクルの短期化

③ソフトウェアテストの増大化

テストへの要望

①スピード

②正確性

③費用対効果

システムテストの現状

①ほとんど手動テスト

②一部自動テストを実施してきたが、続かない

EVIDENT

我々の想い

本来であれば、充分なリグレッションテストにより
デグレードが無いことを確認したい！　だがやり切れない！

テスト工数の増加

テスト時間の制約

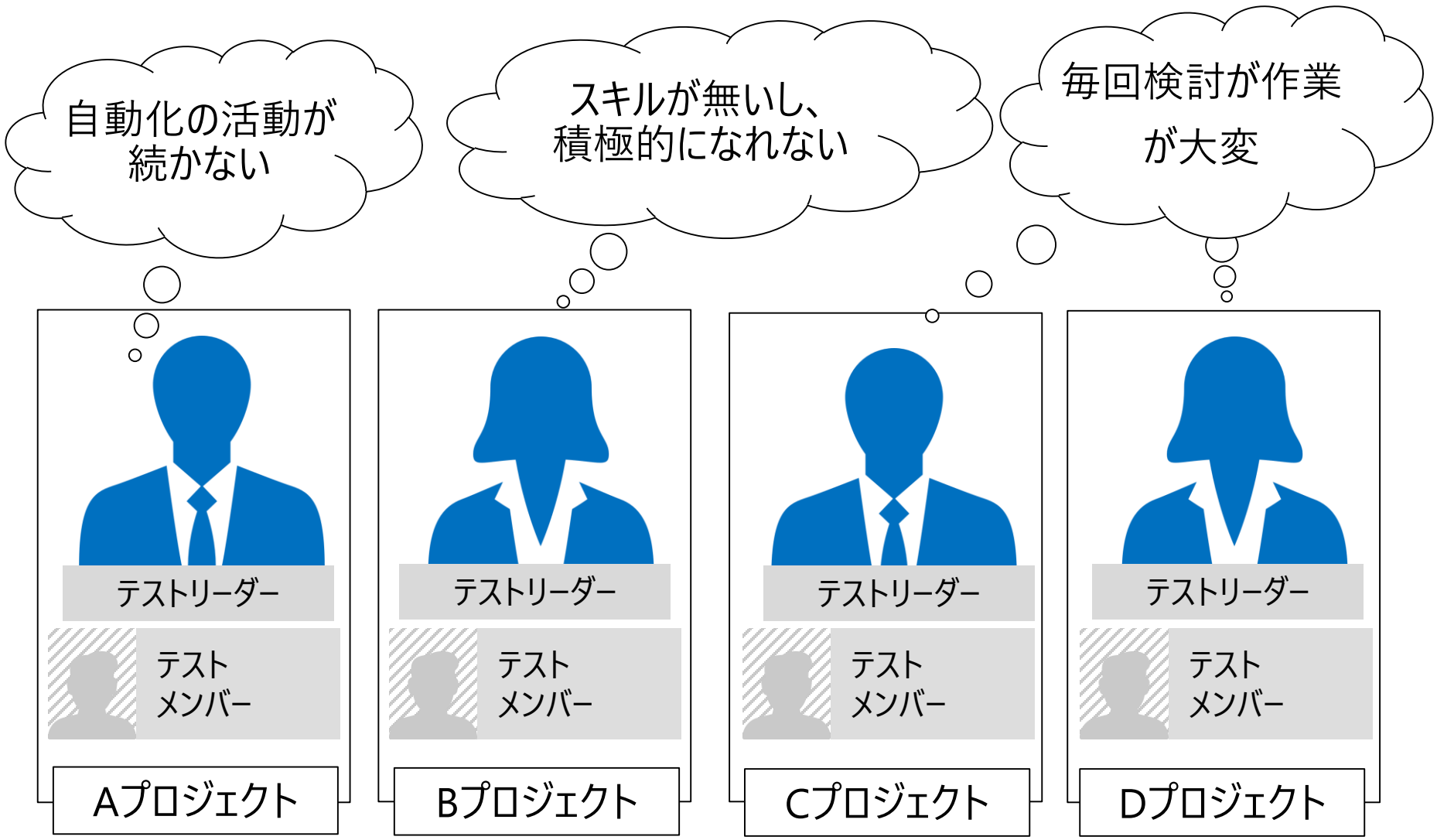
デグレードリスクの増加

テストへの要望、我々の実現したいことを達成するため、
テスト自動化を実現する!

02.自動化における問題・解決策 – 問題抽出

EVIDENT

現場の声



02.自動化における問題・解決策 – 問題分析

EVIDENT

問題

- 自動化の活動が継続しない
- 自動化の推進に消極的
- 重複した検討作業を行い非効率

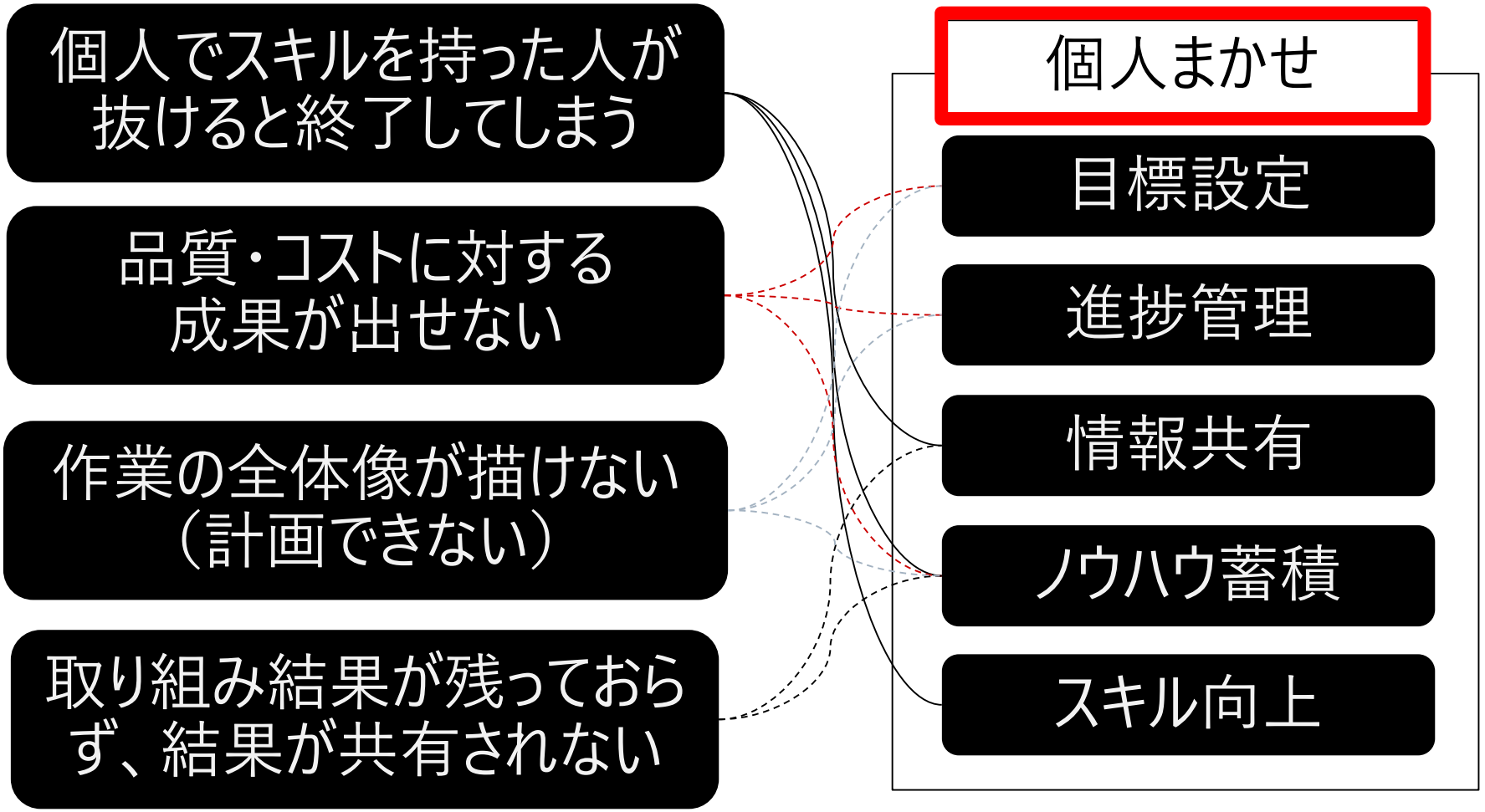
原因

- 個人でスキルを持った人が抜けると終了してしまう
- 品質・コストに対する成果が出せない
- 作業の全体像が描けない（計画できない）
- 取り組み結果が残っておらず、結果が共有されない

02.自動化における問題・解決策 – 原因分析

EVIDENT

原因



02.自動化における問題・解決策 – 対策

EVIDENT

個人まかせ → 組織化で対応

- 目標設定
- 進捗管理
- 情報共有
- ノウハウ蓄積
- スキル向上

02.自動化における問題・解決策 – 対策詳細

EVIDENT

組織化で対応

- 目標設定
- 進捗管理
- 情報共有
- ノウハウ蓄積
- スキル向上

ステップ①組織の行動指針

テスト自動化の目指すべき姿を定義

テスト自動化戦略

ステップ②体制構築

自動化を推進する司令塔

テスト自動化WG

02.自動化における問題・解決策 自動化戦略

EVIDENT

ステップ①組織の行動指針

戦略を作成することで
組織としてテスト自動化の目指すべき姿を定義した

戦略例：

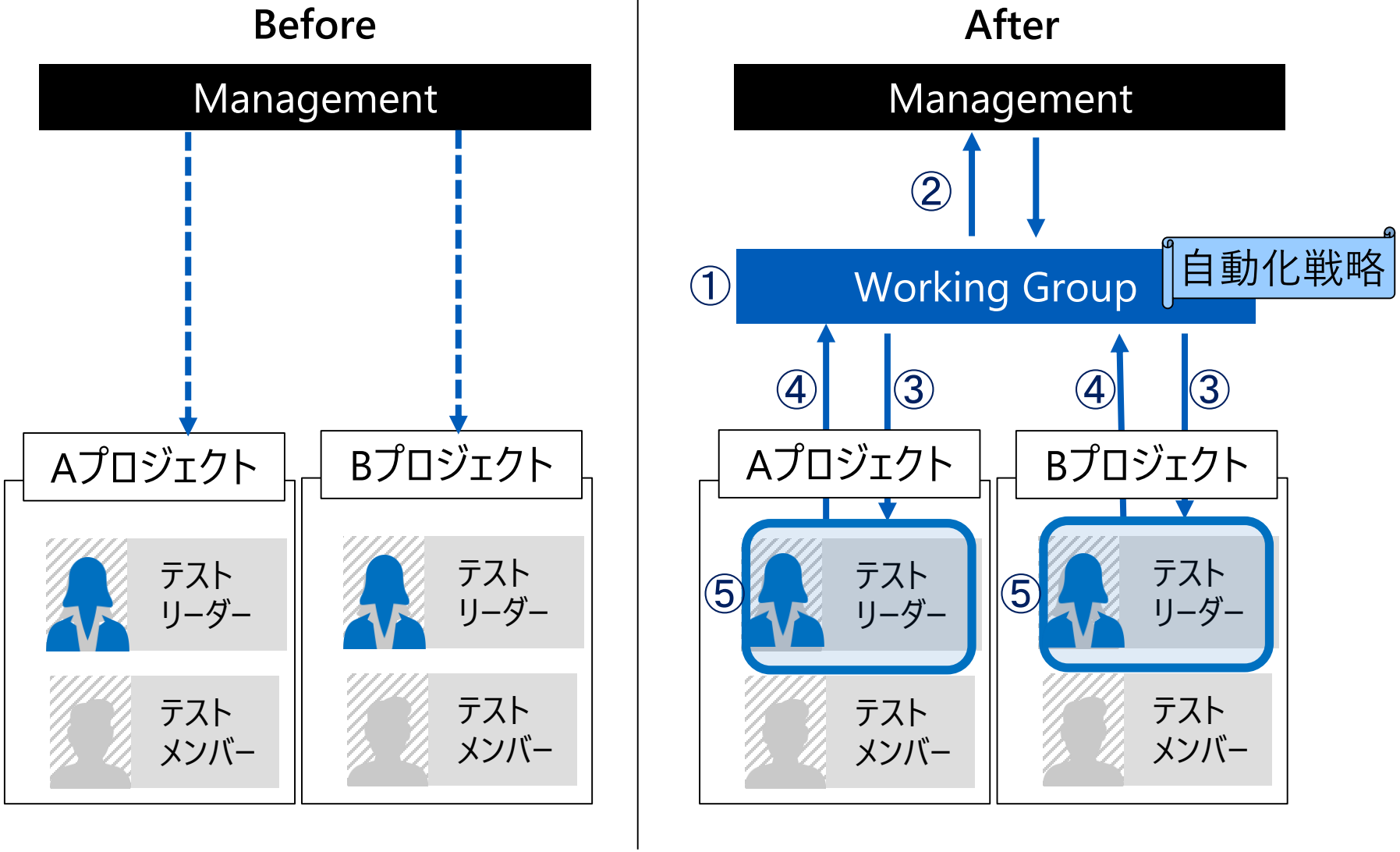
戦略名	テスト自動化の推進
現状分析	・テスト業務において、非効率な部分がある ・製品開発終了と共に自動化活動も終了（非継続） ・製品毎に自動化を検討し、業務が重複している ・自動化に消極的であり、手動で行っている
目指す姿	・テストの効率を上げることで、コスト(C)を抑えつつ品質(Q)日程(D)を確保している ・テストの効果的なノウハウが製品間で共有・活用できている
活動指針	テストチームでWGを結成し、WGを主体に活動を推進する ① テストの自動化を進め、開発業務におけるテスト効率の改善 ② 製品間のテスト業務を統括し、検討結果やノウハウを共有する

02.自動化における問題・解決策 – 体制構築

EVIDENT

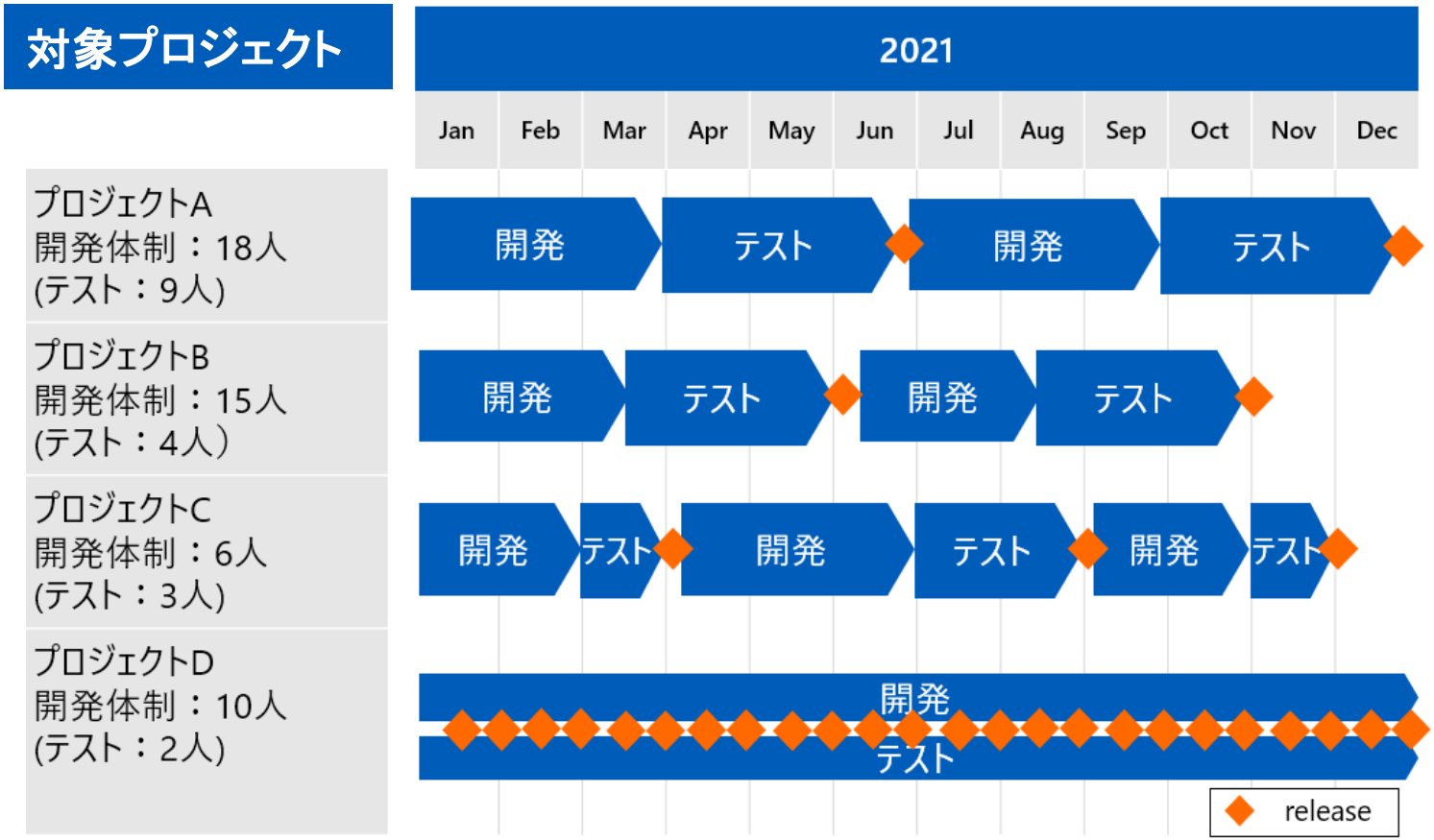
新体制での運用

ステップ②体制構築



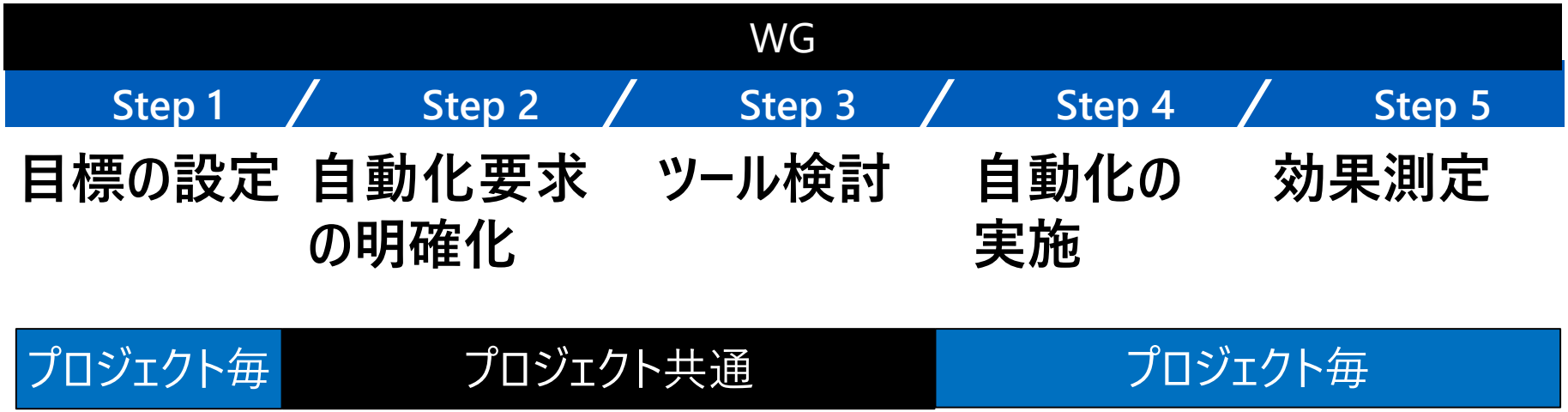
活動紹介

ここからは、新体制を構築し、実際に自動化を行った事例を紹介する



03.自動化の活動内容

自動化実現のためのステップ

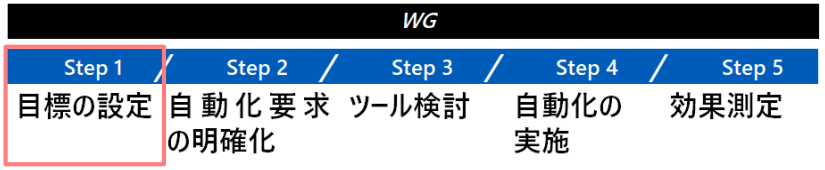


03.自動化の活動内容 – 目標の設定

EVIDENT

プロジェクト目標の設定

プロジェクト毎



プロジェクト	Q：カバレッジ目標	C：コスト改善目標
プロジェクトA	システムテスト項目の30%以上	20%効率化
プロジェクトB	システムテスト項目の5%以上	5%効率化の目途を立てる
プロジェクトC	システムテスト項目の5%以上	5%効率化の目途を立てる
プロジェクトD	システムテスト項目の10%以上	5%効率化の目途を立てる

03.自動化の活動内容 – 自動化要求の明確化

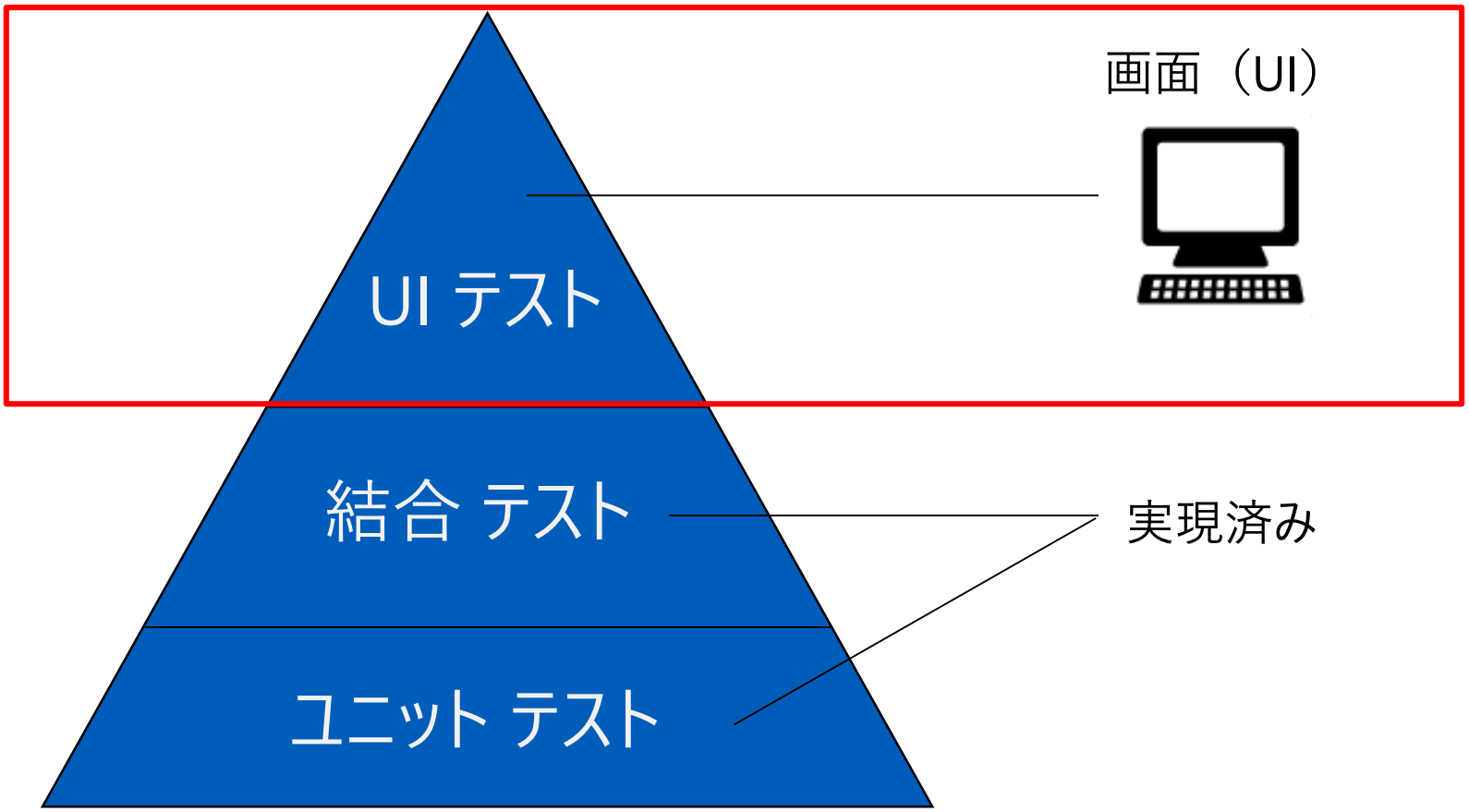
EVIDENT

テスト対象

共通

WG

Step 1	Step 2	Step 3	Step 4	Step 5
目標の設定	自動化要求の明確化	ツール検討	自動化の実施	効果測定

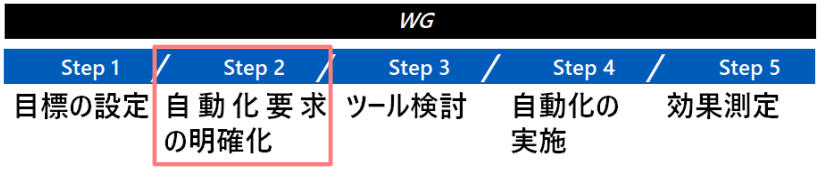


03.自動化の活動内容 – 自動化要求の明確化

EVIDENT

テストツールに対する要求

共通

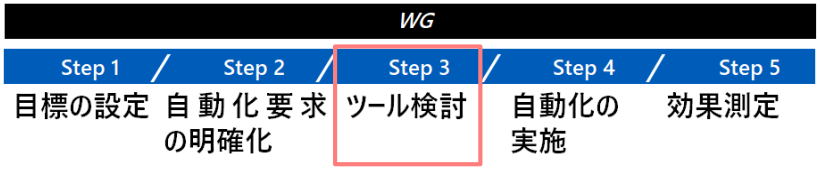


テストツールへの要求事項	
①テスト実現度	UI認識、結果判定
②メンテナンス性	UI変更対応の容易さ
③購入・維持コスト	購入コスト、ランニングコスト

03.自動化の活動内容 – ツール検討

検討及びツールの選定

共通



一般的なテスト自動化ツールを使い、製品アプリケーションで要求を満たしているかどうか検討を行った

ツール名	①テスト実現度		②メンテナンス性	③購入・維持コスト
	IDを識別	画像を識別	変更への対応	
ツールA	○	○	○	◎
ツールB	○	○	○	○
ツールC	○	◎	○	△
ツールD	✖	○	✖	◎

【結論】
・要求を十分満たしたツールAを採用とした
・画像による識別が必要な製品は、コストが高いが、ツールCを 採用とした

EVIDENT

自動化事例紹介①

プロジェクト毎

WG

Step 1	Step 2	Step 3	Step 4	Step 5
目標の設定	自動化要求の明確化	ツール検討	自動化の実施	効果測定

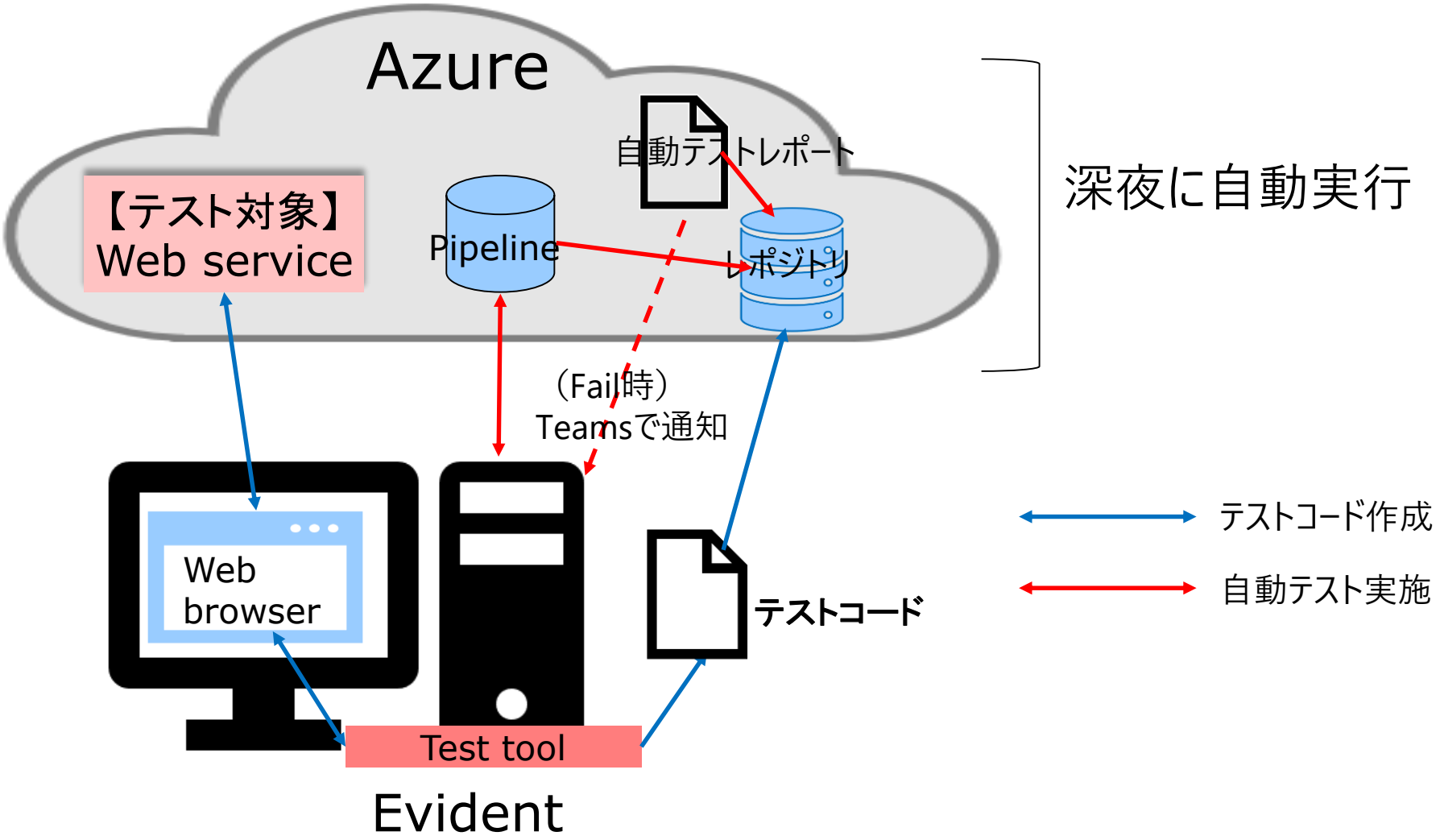
EVIDENT

自動化事例紹介②

プロジェクト毎

WG

Step 1	Step 2	Step 3	Step 4	Step 5
目標の設定	自動化要求の明確化	ツール検討	自動化の実施	効果測定

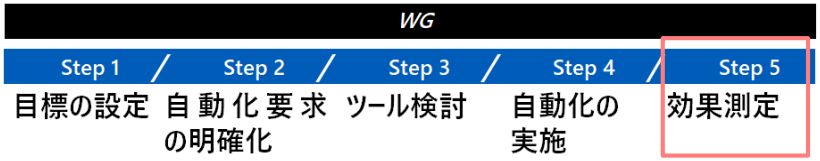


04.結果 – 効果測定

EVIDENT

活動結果

プロジェクト毎



	Q：カバレッジ		C：コスト改善	
プロジェクト	目標	結果	目標	結果
プロジェクトA	テスト項目の30%以上	テスト項目の30%を自動化	20%効率化	約30%の効率化を達成
プロジェクトB	テスト項目の5%以上	テスト項目の5%を自動化	5%効率化の目途を立てる	5%効率化の目途が立った
プロジェクトC	テスト項目の5%以上	テスト項目の5%を自動化	5%効率化の目途を立てる	5%効率化の目途が立った
プロジェクトD	テスト項目の10%以上	テスト項目の15%を自動化	5%効率化の目途を立てる	5%効率化の目途が立った

04.結果 – WGでの活動

EVIDENT

WGにおける組織的活動の詳細

①週次で定例を開催し、自動化活動を支援

・各製品ごとの進捗確認

＞ツールの検討

＞自動化の実現状況

※状況を共有することで、他参加者の自動化への理解が深まる

・自動化に課題があれば、課題の解決

＞技術的に解決する必要があるもの

②汎用的なノウハウのまとめ

・定期的なノウハウ資料の及び共有

③自動化ツールの集中管理

・ツール購入やインフラ整備を集中管理する

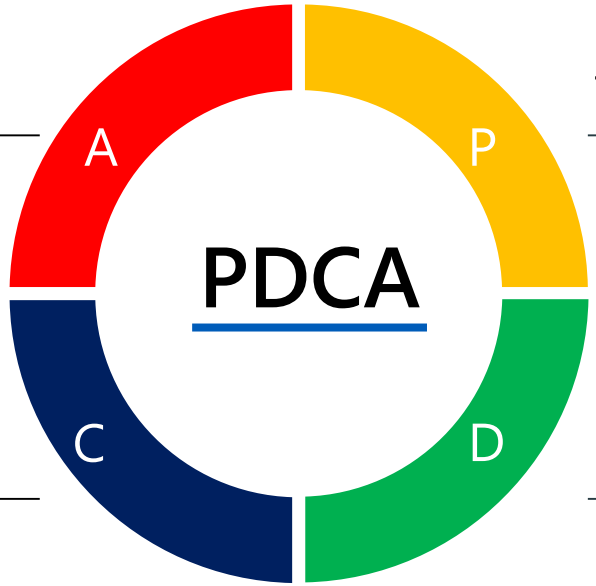
04.結果 – 継続的な活動

EVIDENT

目標の見直し

1年ごとに戦略を更新し、製品毎の計画をアップデートする

4 .Action
改善点をまとめ、WGから、
次年度のインプットとする



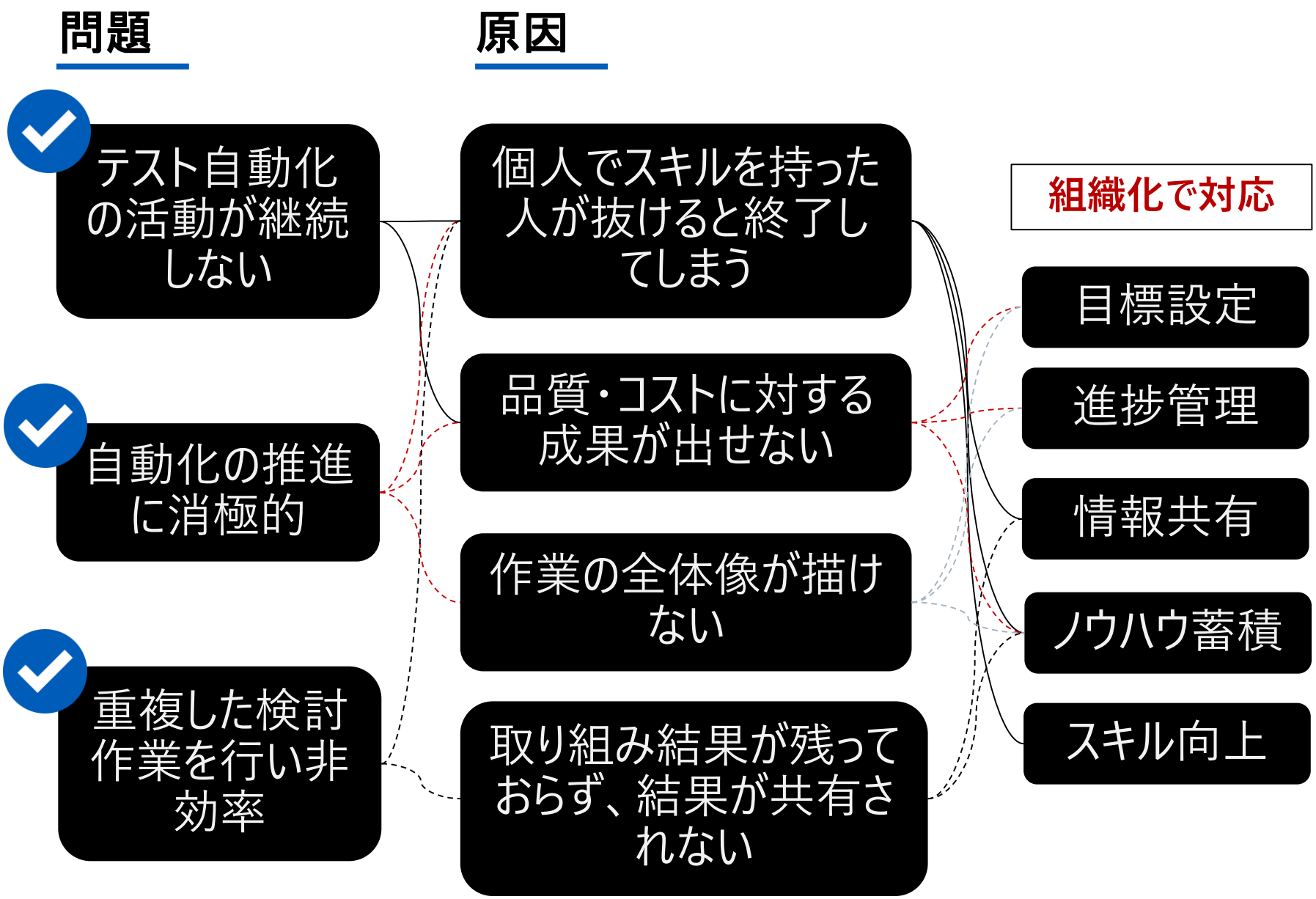
1 .Plan
前年の結果を元に目標を設定

3 .Check
結果の振り返り
成功・失敗要因の深堀

2.Do
WGで定期的に作業の
進捗・ノウハウの共有

04.結果 – 問題の解決状況

EVIDENT



課題

①さらなる自動化スコープの拡大

②製品によってコスト削減効果が小さい場合の見極め

③自動化要員の育成・調達


EVIDENT™
SEEING IS SOLVING

目標値について

EVIDENT

カバレッジの考え方

・システムテストの自動化できる範囲に対して、どれだけ自動化したか

$$\text{カバレッジ（%）} = \frac{\text{自動化した機能}}{\text{全機能}} \times 100$$

自動化できる範囲：
テストツールを使用し、ソフトウェアの操作及び確認が自動化できるもの

自動化できない例：

・H/W起点の操作、もしくは、H/Wの状態を見ないと結果を確認できないもの
※自動化向けにH/Wと連携するよう機能実装すれば、対応は可能だが
複雑になり、対応コストがかさむため
・意図せぬタイミングによって挙動が変わる（たまに別のダイアログが割り込む、など）
※厳密には条件分岐などで対応は可能だが、複雑になり、対応コストがかさむため

目標値について

コスト改善の考え方

・自動テストを実施することで、手動テストと比べどれくらいテスト実施工数が削減できたか

$$\text{コスト改善}(\%) = \frac{\text{手動テスト工数} - \text{自動テスト工数}}{\text{手動テスト工数}} \times 100$$

※自動テストを実施した箇所を対象とする
※手動テスト工数：手動実施した場合にかかるテスト工数
　繰り返しテストした場合は、ここが増える
※自動テスト工数：スクリプト作成工数＋エラー発生時のデバッグ工数
※自動テストは、複数回実施することを前提とする
　複数回実施しないと、テストコード作成コストに見合わないため
プロジェクトAは複数年の実績があるため、実績値から計算
プロジェクトB～Dは、複数回テストを繰り返すことを前提にシミュレートした

ノウハウ例：自動化対象の選定

EVIDENT

・製品ごとに、テスト自動化と相性がある
相性が良い製品：
　・ソフトウェアに閉じた製品（クラウド製品、アプリケーション）
　・定期的にバージョンアップされる製品
効果が限定的な製品
　・H/Wと密に連携しており、挙動が一意になりづらい
　・バージョンアップをしない単発製品

・初期段階は、自動化に対するノウハウや担当者のスキルが十分でないため、自動化範囲を基本的な部分に限定し、その後段階的に範囲を広げていくと実現しやすい

・自動化する機能は、複数の観点から評価し、優先度の高いものから実施する
　・ユーザーの利用頻度が高い
　・障害発生が多い
　・スクリプト化の容易さ
　・物理的な作業が不要
　・ハードウェアの目視確認が不要

ノウハウ例：テストスクリプト、コスト

EVIDENT

テストスクリプト

・テストスクリプト毎の共通処理はコンポーネント化することで、メンテナンス工数を下げられる。

・テストスクリプトを適度な長さに分割することで、途中で止まった際の影響を少なくする。調査しやすくする

・テストスクリプトは、定期的にリファクタリングする機会を設け、効果的な自動化ができるよう維持する

コスト

・自動化は品質向上に寄与する活動と理解し、短期的な工数削減に執着しない