NTT DATA
Trusted Global Innovator

ソフトウェア品質シンポジウム SQiP2020

コード行数を用いない品質分析技術と開発速度を落とさない品質管理手法の提案

2020年9月10日

株式会社NTTデータ　技術革新統括本部 システム技術本部
生産技術部 ソフトウェア技術センタ
熊谷 尚俊

目次

1. 定量的品質管理における昨今の課題
   - プログラムコード行数を用いた品質管理指標の課題
   - クオリティーゲート型品質管理の課題
2. 解決手法の提案
   - 提案手法の方針
   - コード行数を用いない品質分析技術の提案
   - 開発速度を落とさない品質管理手法の提案
3. 実プロジェクトでの適用結果
4. まとめと今後の展望

1.定量的品質管理における昨今の課題

NTT DATA

プログラムコード行数を用いた品質管理指標の課題（1/2）

テストフェーズの品質管理に「バグ摘出密度・テスト密度」を指標として利用されていることが多いのではないでしょうか

昨今は業務パッケージソフトウェアやSaaSといったプログラムコードを直接書かないプロジェクトが増えています

プログラムコード行数の計測が難しい

分母としてプログラムコード行数を利用するのは適切？

定量的な品質管理は大事。しかし

プログラムコード行数に依存しない指標が欲しい！

プログラムコード行数を用いた品質管理指標の課題（2/2）

既存の品質管理

品質が悪い可能性がある

製造業の生産プロセスが元

5Mのバラツキを抑えることが大事

5M
- 材料・部品（Material）
- 設備・機械（Machine）
- 作業者（Men）
- 作業方法（Method）
- 検査・測定（Measurement）

ソフトウェア開発に置き換えると・・・

- ソフトウェア部品・フレームワーク
- 開発ツール・システム基盤
- 開発者
- 開発手法・実施要領
- テスト・計測

昨今は技術が多様化しておりバラバラ

類似プロジェクトが見つからない

水準値に依存せず品質分析がしたい！

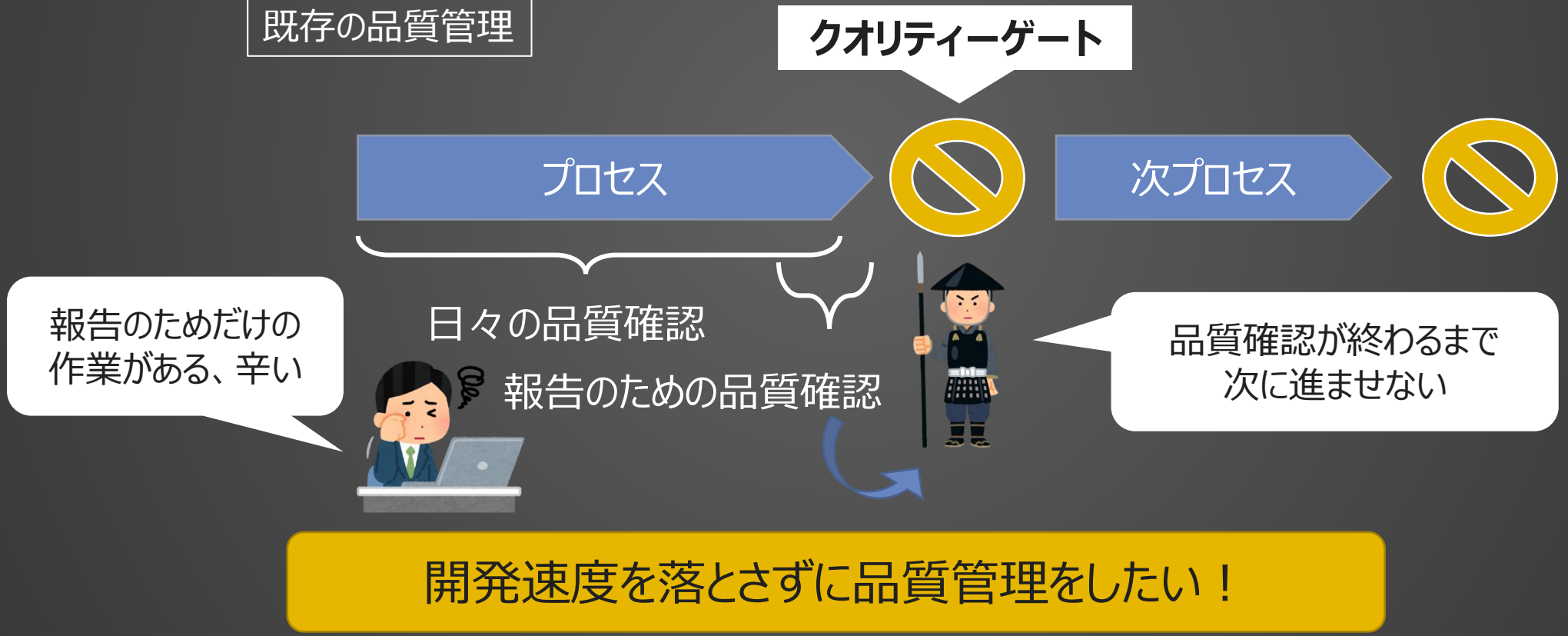
NTT DATA

クオリティーゲート型品質管理の課題

昨今のビジネスにとってソフトウェアの重要性は増しており、その開発を高速に行うことが市場に求められています

品質は保ちつつ、品質管理にかかる時間を短くする必要が生じています

既存の品質管理

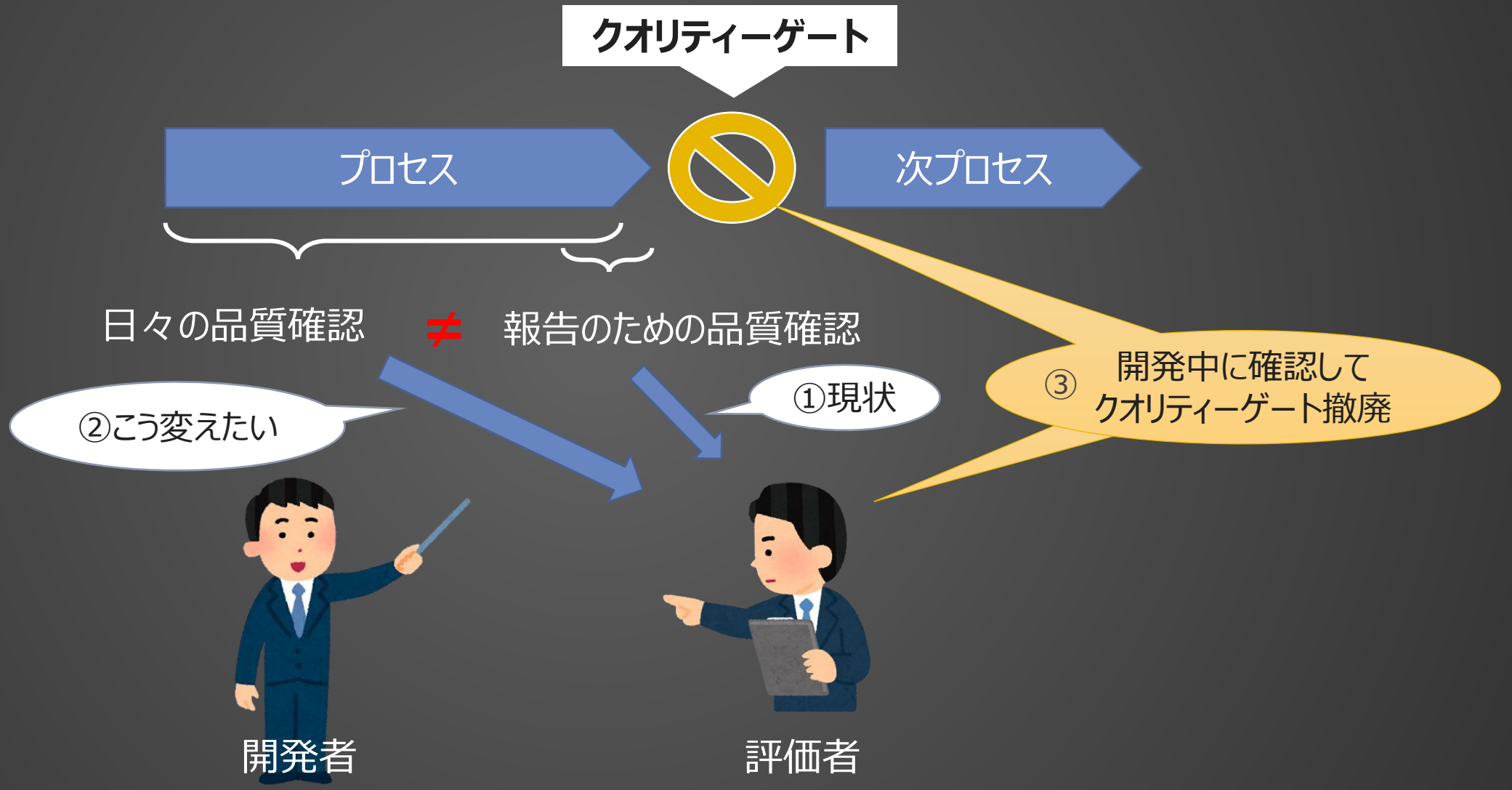


開発速度を落とさずに品質管理をしたい！

2.解決手法の提案

1. 定量的品質管理における昨今の課題
   - プログラムコード行数を用いた品質管理指標の課題
   - クオリティーゲート型品質管理の課題
2. 解決手法の提案
   - 提案手法の方針
   - コード行数を用いない品質分析技術の提案
   - 開発速度を落とさない品質管理手法の提案
3. 実プロジェクトでの適用結果
4. まとめと今後の展望

提案手法の方針（1/3）



提案手法の方針（2/3）

社内のプロジェクトへ、テストプロセス時に日々どう品質確認しているかヒアリング

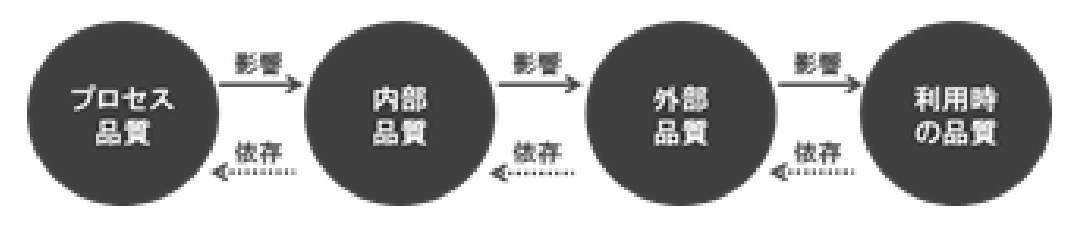
- テスト観点に対してテストを網羅的に実施できているかを確認している(a)
- すり抜けバグの発生状況を確認している(b)

この２つを定量的な品質管理指標に落とし込もう！

提案手法の方針（3/3）

既存の品質管理の目的を振り返ります

- ソフトウェアは作成過程で最終的な価値を評価するのが困難
- 既存の品質管理手法はプロセス上の問題点に着目



品質モデル(JISX0129-1より)

提案手法でもプロセス品質を確認することは変えません

- プロセス品質をあげるためには改善活動が重要
- PDCAサイクルを軸とする事例が多数
- しかし昨今は不確実性が増し計画通りの遂行が困難

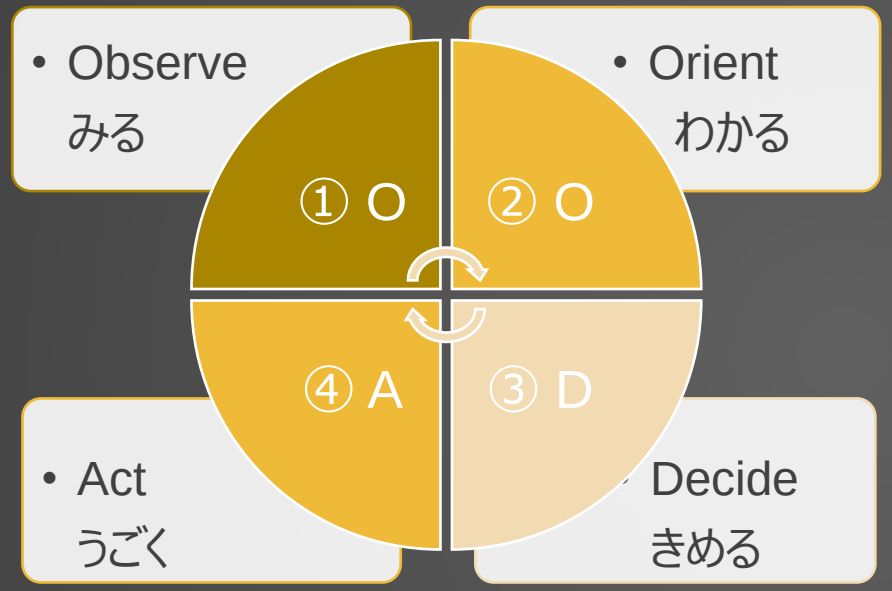
計画からズレるほど
Checkに時間がかかる

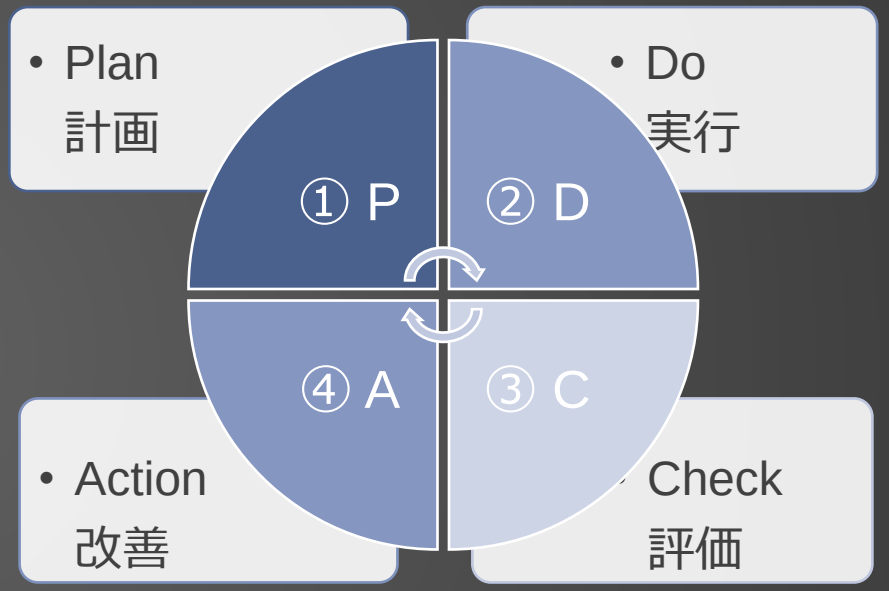
OODAループ※1を取り入れてPDCAを軽量化します

※１：OODAループの説明は次ページ

【参考】OODAループとは

PDCAが計画を立てることから始めるのに対して、OODAは現状を観測することから始めます
変化の激しい場合に臨機応変な対応をするための思想という特徴があります





利点	想定外のことにすぐ対応できる	計画を立てやすい
難点	方針がブレやすい	計画外のことへの対応が苦手
活用先	状況が多様に変化する場合	概ね計画通り進む場合

一長一短あり、本手法では組み合わせて利用する

1. 定量的品質管理における昨今の課題
   - プログラムコード行数を用いた品質管理指標の課題
   - クオリティーゲート型品質管理の課題
2. 解決手法の提案
   - 提案手法の方針
   - コード行数を用いない品質分析技術の提案
   - 開発速度を落とさない品質管理手法の提案
3. 実プロジェクトでの適用結果
4. まとめと今後の展望

【再掲】

社内のプロジェクトへ、テストプロセス時に日々どう品質確認しているかヒアリング

- テスト観点に対してテストを網羅的に実施できているかを確認している(a)
- すり抜けバグの発生状況を確認している(b)

この２つを定量的な品質管理指標に落とし込もう！

コード行数を用いない品質分析技術の提案①　PDCAの品質分析技術

既存は重厚なPDCAサイクルを行っていましたが、提案手法では軽量化します

『テスト観点に対してテストを網羅的に実施できているか』が日常的に確認されていた(a)

テスト観点に対するテスト項目のカバレッジをチェック
⇒ 『観点カバレッジ』と呼称

テスト項目の網羅性view

テスト対象＼テスト観点	処理フローの網羅性	レイアウトの正当性	入力チェック結果の正当	・・・
処理a	10項目	8項目	1項目	
処理b	5項目	0項目	3項目	
・・・				

プロセス品質が悪い（可能性がある）

テスト項目がない

バグ摘出状況の網羅性view

テスト対象＼テスト観点	処理フローの網羅性	レイアウトの正当性	入力チェック結果の正当	・・・	紐づく観点なし
処理a	2件	5件	0件	・・・	1件
処理b	1件	7件	6件	・・・	0件
・・・					

バグを発見できるテスト観点がなかった

【再掲】

社内のプロジェクトへ、テストプロセス時に日々どう品質確認しているかヒアリング

- テスト観点に対してテストを網羅的に実施できているかを確認している(a)
- すり抜けバグの発生状況を確認している(b)

この２つを定量的な品質管理指標に落とし込もう！

コード行数を用いない品質分析技術の提案②　OODAの品質分析技術

軽量化したPDCAの弱点を、OODAループで補足します

『すり抜けバグの発生状況』が日常的に確認されていた(b)

DDP※1をモニタリング
（※１：Defect Detection Percentage;欠陥摘出率）

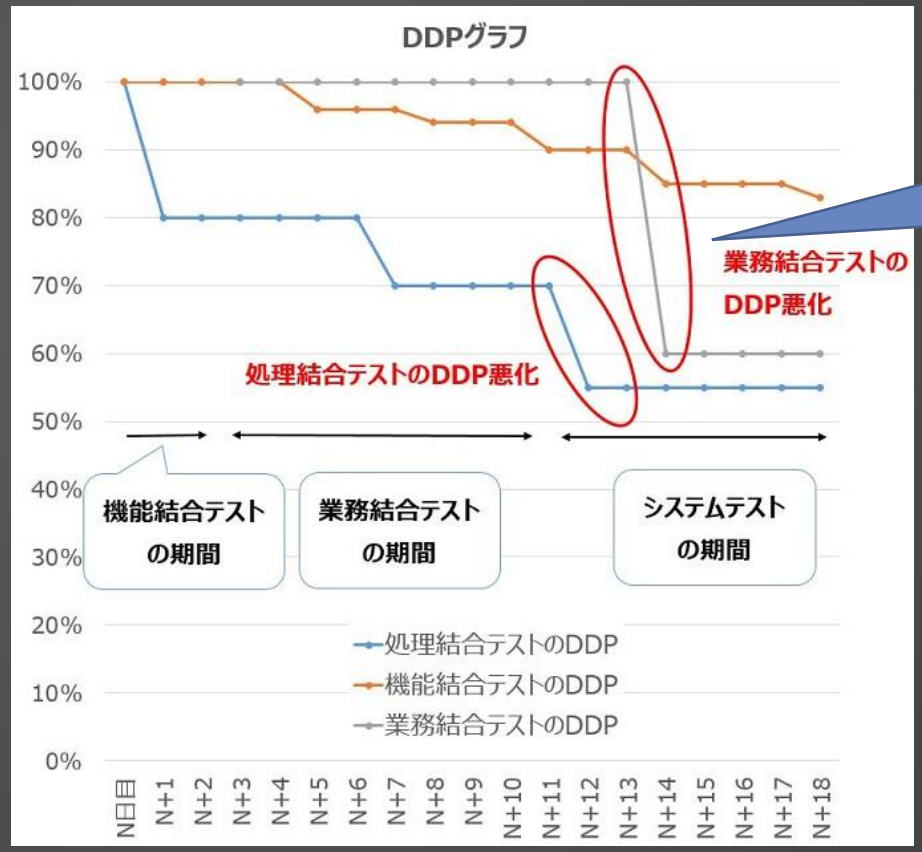
DDPの算出方法

$$DDP = \frac{n}{n+x}$$

n：計測対象のプロセスで摘出したバグの数

x：計測対象のプロセスをすり抜けたバグの数

- すり抜けバグ発生でDDPは下がる
- 計測対象プロセス終了後からモニタリング開始
- 悪化傾向が見られたら計測対象プロセスに対して是正対策



計測対象のプロセスの
品質が悪かった

悪化傾向が見えたら対策

【参考】DDP閾値の参考情報

DDPの閾値	評価	
x > 95%	改善検討 (後工程)	すり抜けバグは一定量出るほうが自然である. 長期間DDPが95%を超えている場合，後続のテストプロセスが狙い通りの効果を出せていない可能性がある.
95%≧ x ≧85%	順調	テストは狙い通りの効果を出している.
85%> x ≧60%	改善検討	今後，DDPが60%を下回るか注意深く観察する必要がある. 重要な機能のテストについては早期にプロセス改善を検討してもよい.
60% > x	改善必須	テストは狙い通りの効果を出せていない．実施中のテストを停止し， すぐさまプロセス改善に取り組まなければいけない.

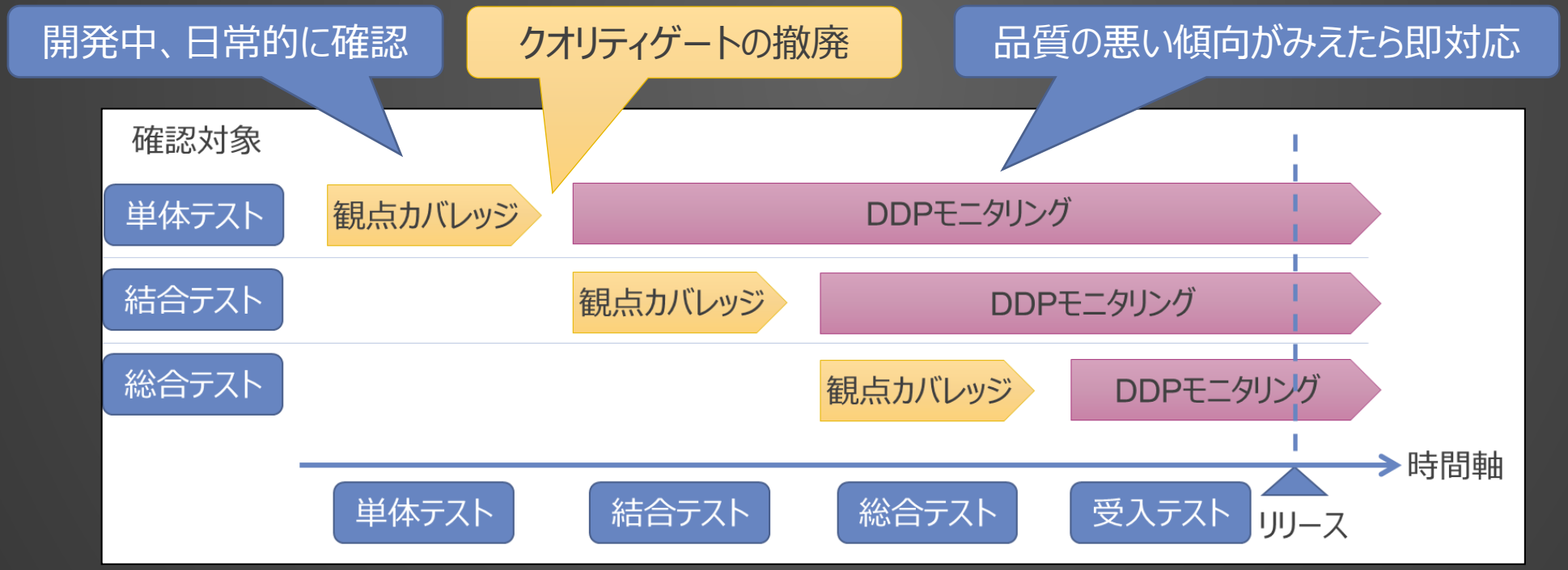
［出典］Dorothy Graham, Challenges in Software Testing, JaSST'13 Tokyo, 2014

1. 定量的品質管理における昨今の課題
   - プログラムコード行数を用いた品質管理指標の課題
   - クオリティーゲート型品質管理の課題
2. 解決手法の提案
   - 提案手法の方針
   - コード行数を用いない品質分析技術の提案
   - 開発速度を落とさない品質管理手法の提案
3. 実プロジェクトでの適用結果
4. まとめと今後の展望

開発速度を落とさない品質管理手法の提案

観点カバレッジとDDPモニタリングの組み合わせにより、既存手法より高速な品質管理の実現を目指します

思想	品質分析技術	確認事項
PDCAサイクル	①観点カバレッジ	テスト観点の網羅性の確認
OODAループ	②DDPモニタリング	すり抜けバグ状況の確認



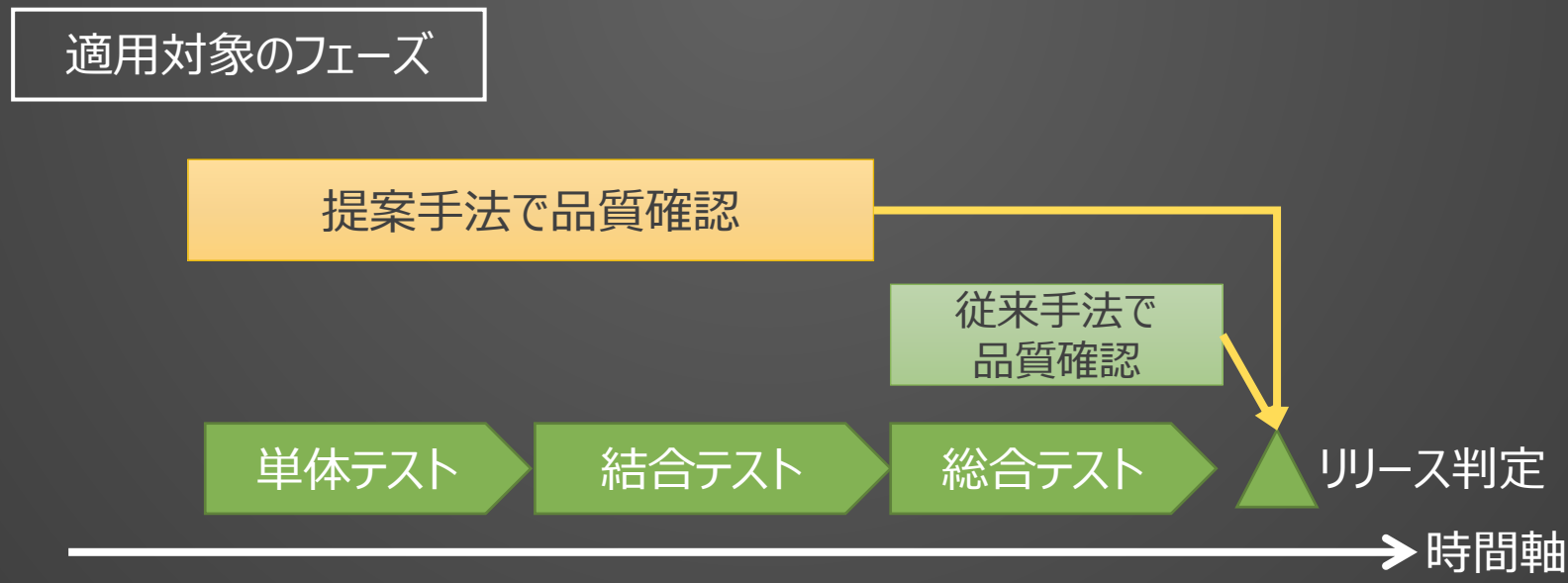
NTT DATA

3.実プロジェクトでの適用結果

NTT DATA

適用プロジェクトのプロファイル

顧客	某公共団体	**開発区分**	機能追加・改修（マイグレ）
アーキテクチャ	Java/Web	**開発プロセス**	ウォーターフォール
開発規模	4kStep（適用箇所のみ）		
その他	開発量が少なく、バグも少ない見込みであった。 リリース対象はこの機能以外もあり、合わせて開発していた。		



適用結果①　観点カバレッジ

観点カバレッジ表を確認し、問題なかったため次フェーズへ通過

■テスト項目の網羅性ビュー（一部抜粋）

	テスト観点			
モジュールA	入力バリエーションの網羅性	繰り返し処理の網羅性	帳票様式の正当性	…
テスト項目数	10件	1件	0件	…
0件で問題ない根拠	-	-	当該処理で帳票出力はないため	…

各開発対象について
テスト項目0件の理由を
確認し問題なかった。

■バグ摘出状況の網羅性ビュー（一部抜粋）

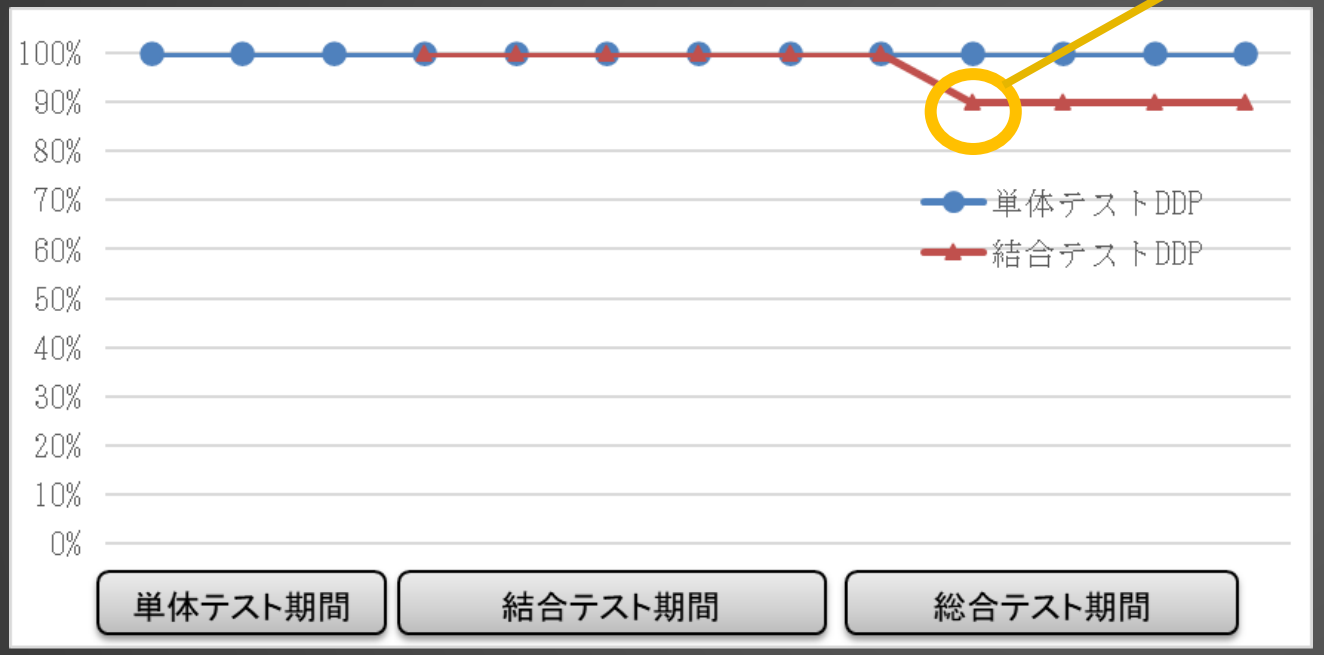
	テスト観点		
テスト対象	境界値の網羅性	入力バリエーションの網羅性	その他の観点
モジュールX	1件	0件	0件
モジュールY	0件	1件	0件
その他モジュール	0件	0件	0件

テスト観点に紐づかないバグは
摘出されなかった
⇒テスト観点の漏れなし

バグの偏りもなく問題なし

適用結果②　DDPモニタリング

開発中DDPモニタリングを継続し、本開発では問題検出なし
リリース後も問題は発生しておらず品質達成



【Observe】 DDPが10%降下

【Orient】
- すり抜けたのは1件のみ
- バグの中身に問題はなし
- 横展開で新たなバグもなし

【 Decide】 総合テストは中断せず継続

【 Act】 （本案件では無し）

リリース判定時に、従来の品質管理手法でも確認して問題なし

	テストフェーズ中のバグ	開発中のすり抜けバグ	リリース後すり抜けバグ
単体テスト	2件	0件	0件
結合テスト	9件	1件	0件

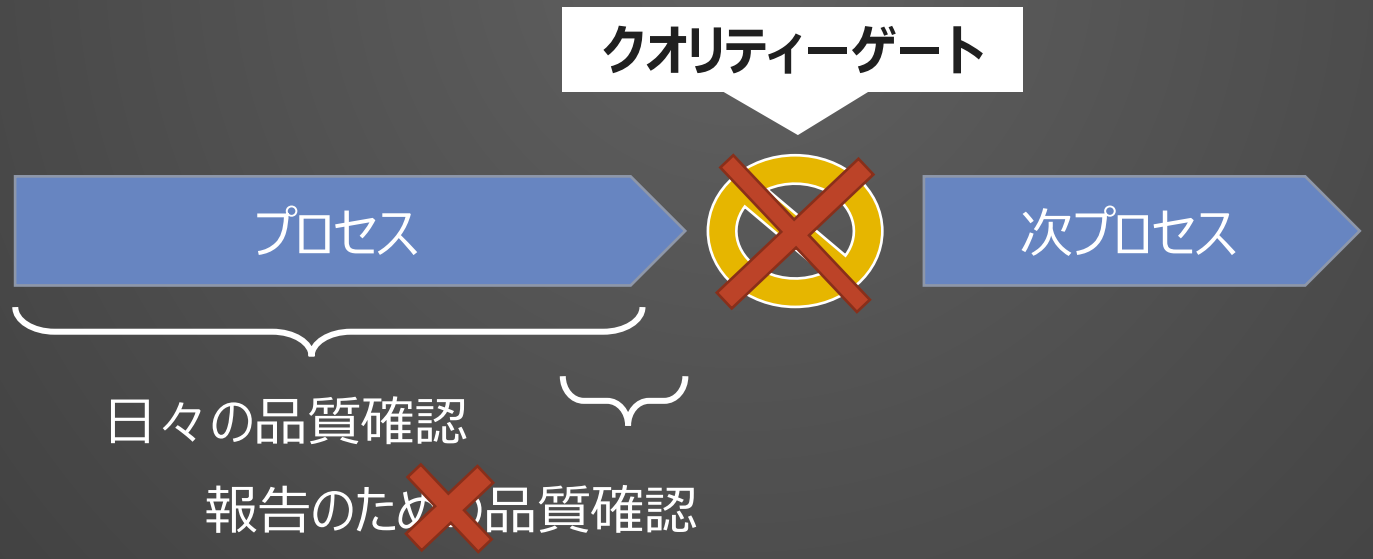
リリース後も問題なし

NTT DATA

効率化の考察

提案手法の適用により下記の効率化が図れました。

- クオリティーゲートの撤廃により、開発停滞が減少
- クオリティーゲートのためだけの品質管理作業の削減
- 観点カバレッジは作成者がテストケースを作り終わった時点で品質問題に気付けるため、レビュアーや評価者に品質指摘を受けていたときと比べ手戻り削減



4.まとめと今後の展望

NTT DATA

まとめと今後の展望

まとめ

- プログラムコード行数を用いた品質分析技術に変わり、“観点カバレッジ”と“DDPモニタリング”を提案
- 実プロジェクトにおいて問題なく品質管理を行えたことを紹介
- 従来より高速に定量的な品質管理を行える見込を示唆

今後の展望

- 適用結果をさらに蓄積（適用完了のプロジェクトが他に2件と、適用中が複数あり）
- 既存手法と提案手法の併用ではなく、提案手法のみ適用したプロジェクトの早期開始
- 今回はテストプロセスのみを対象にしたご紹介だったが、上流プロセスに対しても検討中であり成果が得られたら追ってご紹介

NTT DATA

Trusted Global Innovator