HITACHI
Inspire the Next

設計レビューで問題を叩き出せ！
～QAの役割り～

2018/12/14

株式会社 日立製作所 情報・通信システム社
IoT・クラウドサービス事業部　品質保証本部
IT基盤ソリューション品質保証部
白水俊治

HITACHI
Inspire the Next

Contents

HITACHI
Inspire the Next

はじめに

（いきなりですが）日々悩んでいます

HITACHI
Inspire the Next

汎用ソフトウェア製品開発を取り巻く環境変化と品質課題

（引用文献：梯雅人・居駒幹夫
「ソフトウェア品質保証の基本」より）

ソフトウェアを利用する顧客の分野・業種が拡大！

金融から分析サービスまで、要望の幅が増えたのに重い高信頼性確保作業は必須。

ハードウェア性能の飛躍的向上と、ネットワークでシステム構成が複雑化

組合せるべき状態やタイミングが増え、詳細設計が複雑化！テスト漏れも！

ビジネスにおけるソフトウェアの位置付けが変化！

顧客ビジネスに直結する為、スピード開発、短納期開発が増加！

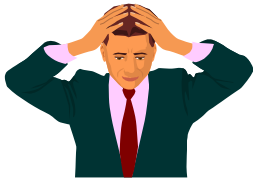
ソフトウェア開発方法が多様化

アジャイル開発は、不慣で作業の質の維持が難しそう・・・

ソフトウェア長命化によるライフサイクルの変化

昔々のバグが発生しますがソースしか残ってません。対策するのが怖いです・・・

どうすれば。。。



**重い開発ルール、厳しい短期開発、急な開発手法変更、
開発ノウハウの俗人化や偏り、スキル・人員不足、
品質問題に対する世間の強いバッシング、、、**

今日お話しすること

HITACHI
Inspire the Next

【発表概要】

今回の発表は、
汎用ソフトウェア製品をウォーターフォール型で開発するケースにおいて、
弊社の品質保証部門(*)が、「レビュー」という手段を、どう活用しているか。を
私の経験も含めての紹介と、課題や悩みについての話をさせて頂きます。
良い意味でも、悪い意味でも、参考になれば幸いです。

＊：以降、「QA」と表記、呼称します。

◆発表者◆

白水俊治（しろうず　としはる） 汎用ソフトウェア開発品質保証部門　課長
品質保証部門だけに　ウン十年所属。
主な担当製品は、メインフレームとオープン、それぞれの汎用DBソフトウェア製品。
担当製品は、金融や公共系等のミッションクリティカルシステムで使用される事が多い。
ビックデータ分析やサービス系製品、アジャイル型開発の経験もあり

前説

HITACHI
Inspire the Next

- レビューの考え方
- ウォーターフォール型開発の注意
- 弊社の基本的な汎用ソフト開発の流れ紹介

レビューとテストの違いって？

HITACHI
Inspire the Next

レビューは机上テストともいえますが、、、

『テストは、書いて**ある**通りに動くことを確認』

『レビューは、書いてい**ない**事を見付ける』

「ない」事が正しいかの証明は非常に難しいです。

また、言い換えるなら、

『テストは、**書かれていないことは確認しない**』と言えます。

ならば、品質確保の為には、

自分が気付いて『**書かせる**』のと、

自分が気付いて『**書いていない事もテストする**』

しかない。と考えています。

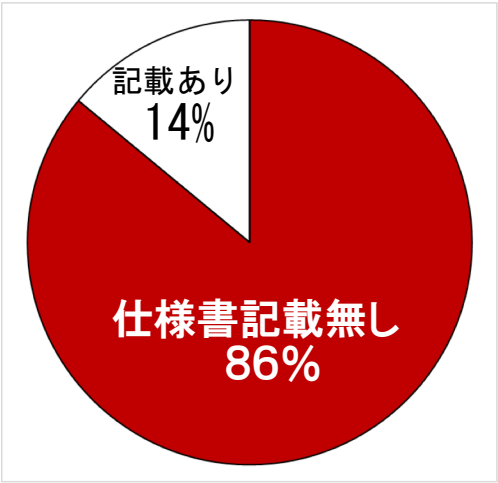
不具合と仕様書記載有無の関係分析

HITACHI
Inspire the Next

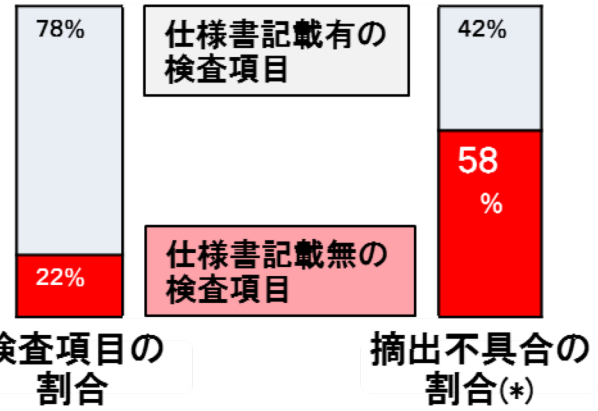
書かれていないところに、問題は潜む

引用：ソフトウェア品質シンポジウム2015
「製品検査における仕様書記載外の問題点」
星名卓郎、他

社外発生不具合の割合



検査摘出不具合の割合



ある製品の分析では

社外発生不具合の
86%が記載無し
社外発生不具合の86%が仕様書に記載が無い動作から発生

検査摘出不具合の
58%が記載無し
検査摘出不具合の58%が仕様書に記載が無い動作から摘出

記載が無いから、不良を作り込み？
記載が無いから、テストで摘出できず？

でも、開発者は気付かないから書いてない…
レビューアも気付かないから書かせてない・・・

気付く・書かせる「取り組み」が重要

＊：偶発的に摘出した不具合含む（約半数）

（いまさらですが）ソフトウェアのレビューとは

HITACHI
Inspire the Next

レビューの良し悪しは「人」から「暗黙知」を引き出せるかどうか

ソフトウェアのレビューの定義

ソフトウェア開発において、
各工程の成果物に対し、
開発の**関係者・責任者**が
主に**会合の形式**を用いて、
内容を**検証**、
または合意・承認すること。

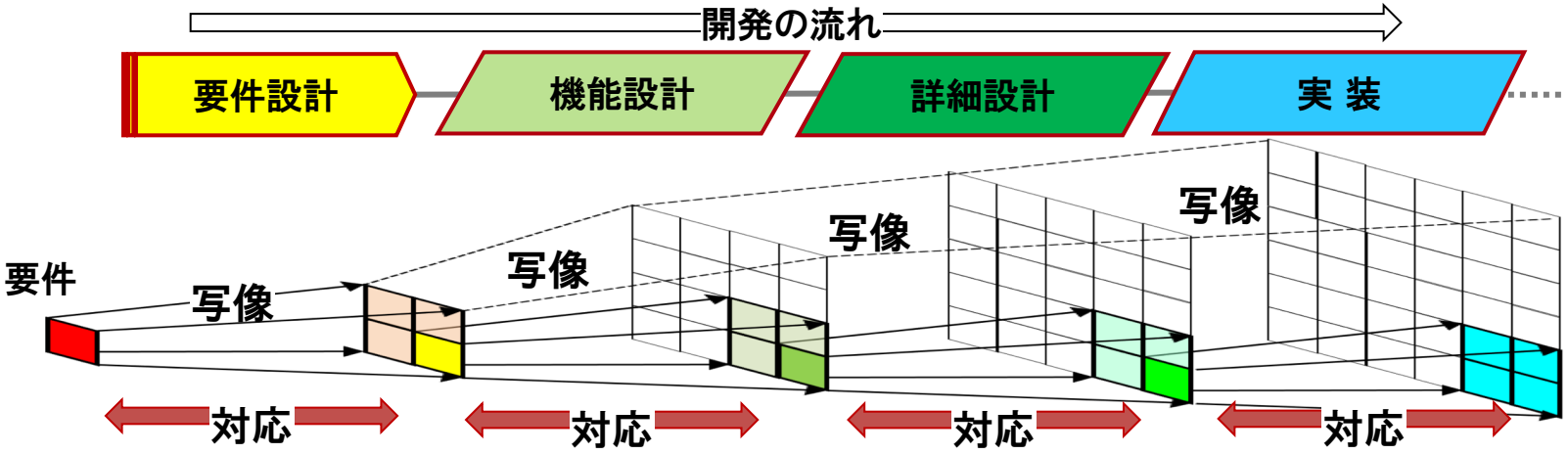
レビューの5つの目的

①	審査
②	品質検証
③	整合性確認
④	連携
⑤	教育

（おさらい）ウォーターフォール型開発

HITACHI
Inspire the Next

ソフトウェアの設計とは、
要件や機能を写像（展開）させる事によって設計（詳細化）が行われる。



※：1マスが設計（レビュー）単位

ウォーターフォール開発では、
要件→外部仕様→機能仕様→内部仕様→実装→テスト までを対応させ、
それぞれをドキュメントに書き起こし、機能単位ごとに、レビュー等で検証する。

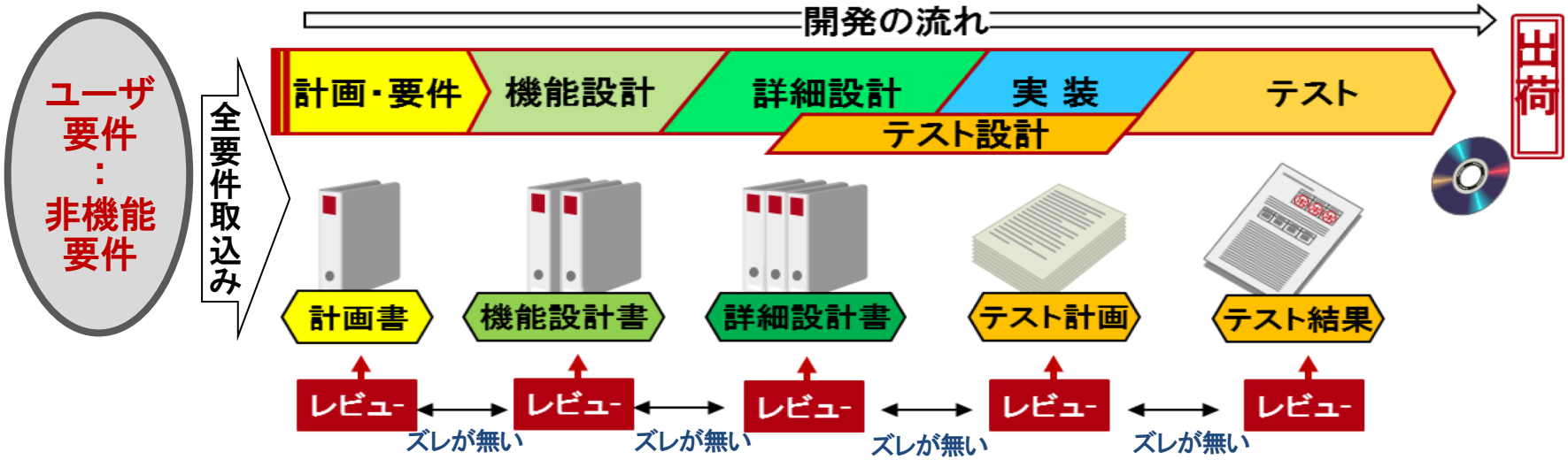
【利点】段階的に詳細化する為「前工程の成果物とのずれ」有無に気付き易く、
小さい単位で設計・レビューする為、内容を検証・確認しやすい。

【注意点】
「記載漏れ」に気付かないと、そのまま開発が進み、問題に気付くのが遅くなる。
➔「書き漏れ」が無いか？の視点は全レビューで意識し続ける必要がある。

（おさらい）ウォーターフォール型開発

HITACHI
Inspire the Next

ウォーターフォール型開発　≒　「最初に決めて守り続ける！」



・・・最初に全体計画を決める。前工程の成果物を詳細化する形で設計。
　　成果物にズレや漏れが無い事をレビュー・テストで確認し、次工程へ進む

- 長い歴史（経験）があるので、（なんとなく）安心。
- ただ、プロセス改善をちゃんとやると、ルールやチェックリスト類が多くなる。 **重くなる**
- 問題は後半で表面化する。リカバリ/手戻り作業の量が致命的となる。
- だから、設計書はしっかり作成し、レビューを繰り返し確認する。 **重くなる**

工程が遅れると、残業休出。テスト工程短縮（項目削減？）。 納品・出荷遅延・・
ウォーターフォール開発は、上流の設計作業が全体作業の約5割を占めているな・・・
短期開発やるなら、ここを減らすしかないな。。。

弊社の基本的な汎用ソフト開発の流れ

HITACHI
Inspire the Next

【設計書作成の流れ】
開発工程毎の規格や設計ガイド、チェックリストに沿って作成され、
開発部門の内部レビュー等でのチェックを経て、
QAや関連部署の責任者・有識者での公式の検証レビューを行う。

製品・機能毎開発
ノウハウ類
・教育資料
・チェックリスト類

規格、公式ガイド
チェックリスト

機能仕様書

総括編
処理方式編
入出力データ編
ライブラリ編
接続編
運用編

詳細設計書

総括編
構造設計編
ﾃｰﾌﾞﾙ設計編
処理方式編
性能設計編
：

内部レビュー

公式レビュー

（基準）公式レビュー出席者

対象ﾄﾞｷｭﾒﾝﾄ	開発部署		ﾏﾆｭｱﾙ部署		QA部署	
開発方針書	課長	主任	主任	担当	主任	担当
機能仕様書	◎	◎	△	◎	△	◎
外部仕様書	◎	◎	△	◎	△	◎
取扱い説明書	△	◎			◎	◎
詳細設計書	◎	◎			△	◎
テスト計画書	◎	◎			◎	◎
ﾃｽﾄ結果報告書	◎	◎			◎	◎
テスト項目	△	◎				△
：						

弊社の基本的な汎用ソフト開発の流れ

HITACHI
Inspire the Next

設計ガイド・チェックリスト類

メインフレーム時代から蓄積・継承された
開発や事故事例等から得た開発ノウハウの集合体

計画・要件設計

品質機能展開：QFD適用ガイド
機能仕様定義のあり方（教育）
ユーザ中心設計ガイド
：

機能設計

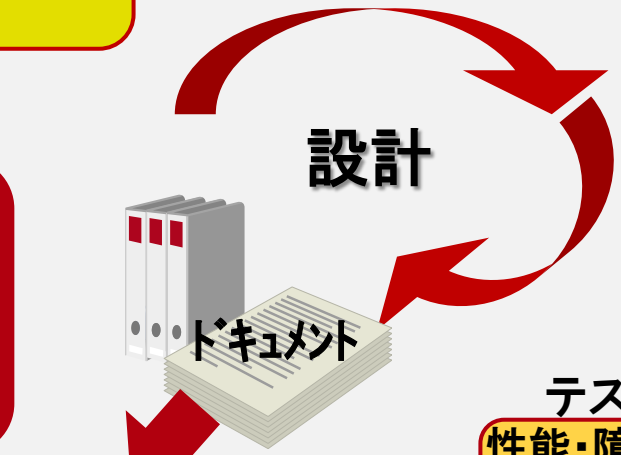
機能仕様書作成教育
障害回復設計ガイドライン
トラブルシュート機能設計方針
セキュリティ設計ガイドライン
：

詳細設計

高信頼性設計基準・ガイド
重要障害防止チェックリスト
デグレード防止 チェックリスト
トラブル事例・設計ノウハウ集
：

品質保証

品質管理
プロジェクト診断
設計書チェックリスト
デザインレビュー
検査計画
探針・製品検査



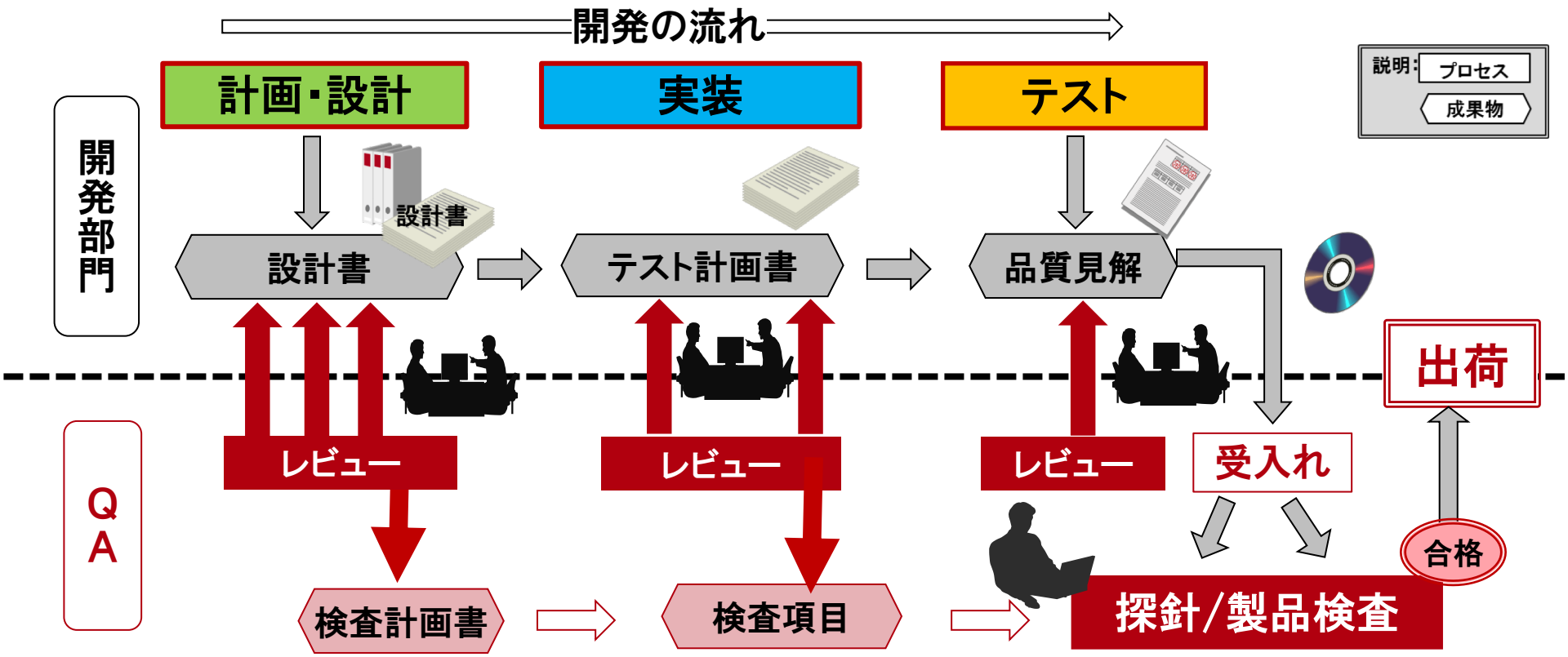
実装

各種言語別
コーディング規則集
信頼性コーディング技法
：

テスト

性能・障害テスト実施方法
統合テストガイドライン
：

弊社の基本的な汎用ソフト開発の流れ



各工程の**成果物(設計書・結果)**に対し、**関係者**がレビューで、**複数回**検証。
更に、最終的にQAが**製品を動作確認**。

ちなみに、検査で発見不具合や、顧客で発生した不具合は、原因分析し、
製品見直しと、開発プロセスへのフィードバックまで行っています。

結構、ちゃんとやっているつもりなんですが・・・

2. 弊社製品開発におけるQAの役割り紹介

HITACHI
Inspire the Next

「気付く」「書いてもらう」ために、

どうしたらいいでしょう？

私達はQAの行動が重要だと考えています。

まずは、

その弊社QAの特徴から説明します。

弊社のQAの特徴

HITACHI
Inspire the Next

弊社QAの特徴・・・「少し変わってる」と言われることがあるのですが

■ 現物確認至上主義！

- 開発内容は自分の目で確認します（設計書、証拠物レビュー）
- 製品の品質は、**実際に動かして確認します**（製品検査）
- 外部仕様だけでなく、処理方式まで踏み込んだ確認もします

■ 顧客対応スペシャリスト!?

- サポート部門が対応できない難しい案件は、自ら解決させます
- **現地でのトラブル復旧支援、解決対応**も行います
- 広い知識で顧客のシステム設計支援も行います

■ 開発部門と独立組織！

- 品質活動に特化した全社組織です
  ➔製品出荷の**『最終障壁』**と言われてます

弊社のQAの活動

HITACHI
Inspire the Next

「製品開発」と「顧客対応」で、製品品質向上と稼働品質維持を図る

実機を使用した検査作業で
関連製品を含め広い知識を得る

開発部門と独立した組織
➔製品出荷の**最終障壁**

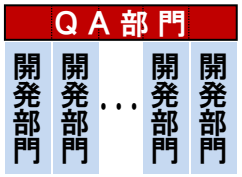
「システム視点」

出荷

製品開発

顧客対応

全社横断
品質保証組織



反映

実機検査で得た知識を
顧客対応に活かす

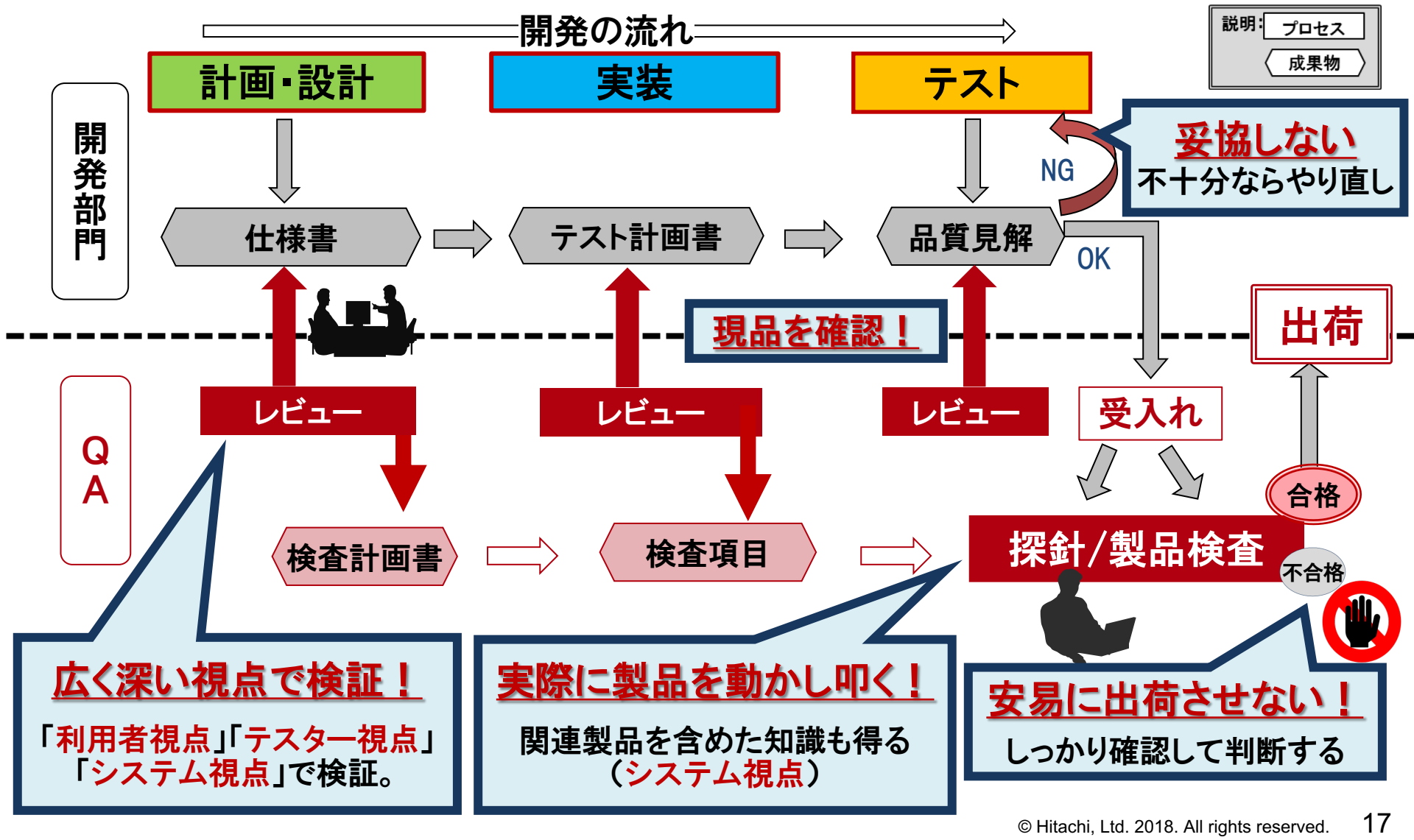
顧客対応で得た知識を
製品開発に活かす

「利用者視点」
顧客の要望
実際の使用条件・環境

弊社のQAの活動［製品開発］

HITACHI
Inspire the Next

設計プロセスだけでなく、現品確認重視の活動➔現品レビュー・実機確認



弊社のQAの活動［顧客対応］

HITACHI
Inspire the Next

確認で得た知識を最大限に活かして顧客対応を行う

稼働品質維持活動

顧客
・SE

？
製品仕様や
使用方法

サポートサービス部門

QA/製品開発部門

サポート部隊で解決
しきれないものを対応

障害解決支援活動

トラブル発生

重大トラブル発生時は
駆け付け支援

障害調査を仕切る
調査観点出しも

自ら報告書も作成
顧客説明もします

システム品質確保支援活動

顧客システム設計に
製品有識者として参画

- 設計書レビュー
- テスト支援
- トラブル解決支援

弊社のQAの活動［意識と効果］

HITACHI
Inspire the Next

自分で確認・検査する

見逃せば社外事故

意識が変わる

責任感が高まる

危機感を持つ

障害時の惨状を知る

社外トラブル発生

自分で顧客対応する

早期の復旧、解決

➔顧客の信頼を得る

顧客意見や考え方、
想定していなかった使い方を知る

利用者視点　視野が広く変わる

開発レビューで
広く深い指摘が出来る

ここで漏らせば事故・トラブルに・・・

検査作業で
広く深い確認が出来る

問題
摘出

組織の品質意識の醸成

製品品質
上がる

システム設計
レビューに活かす

顧客意見

稼働品質
上がる

弊社のQAの活動［意識と効果］

HITACHI
Inspire the Next

自分で確認・検査する

見逃せば社外事故

意識が変わる

責任感が高まる

危機感を持つ

障害時の惨状を知る

社外トラブル発生

自分で顧客対応する

早期の復旧、解決

➔顧客の信頼を得る

顧客意見や考え方、
想定していなかった使い方を知る

利用者視点　視野が広く変わる

知識を活用

自製品の処理方式と、
OSやハード、関連製品の知識を得る

視点が深く変わる　テスター視点 システム視点

開発レビューで
広く深い指摘が出来る

ここで漏らせば事故・トラブルに・・・

検査作業で
広く深い確認が出来る

システム設計
レビューに活かす

問題摘出

製品品質上がる

稼働品質上がる

弊社のQAの活動［意識と効果］

HITACHI
Inspire the Next

自分で確認・検査する

見逃せば社外事故

意識が変わる

責任感が高まる

危機感を持つ

障害時の惨状を知る

社外トラブル発生

自分で顧客対応する

早期の復旧、解決

➔顧客の信頼を得る

顧客意見や考え方、
想定していなかった使い方を知る

利用者視点　視野が広く変わる

知識を活用

自製品の処理方式と、
OSやハード、関連製品の知識を得る

視点が深く変わる　テスター視点 システム視点

開発レビューで
広く深い指摘が出来る

ここで漏らせば事故・トラブルに・・・

検査作業で
広く深い確認が出来る

問題摘出

組織の品質意識の醸成

製品品質上がる

システム設計
レビューに活かす

顧客意見

稼働品質上がる

弊社のQAの活動［役割り纏め］

HITACHI
Inspire the Next

「製品開発」と「顧客対応」で、製品品質向上と稼働品質維持を図る

設計プロセスだけでなく、
現品確認主体の活動
➔現品レビュー・実機確認

他人に気付かせる

現品確認・実機確認で
得た知識を最大限に活かして
顧客対応を行う

自分で気付く

品質活動の原動力

- 自分で確認することによる「品質への責任感」
- 顧客対応することによる「品質への危機感」

加えて

- トラブル対応を通して **お客様、SEとの信頼関係強化**
- 製品サポートで得た **お客様の声を製品開発に反映**
- お客様意見の代弁者として **開発部門全体の品質意識を醸成**

3. 弊社製品開発でのレビューとQAの取り組み

HITACHI
Inspire the Next

ここからは、

開発者が「気付いてない」「書いてない」ことを

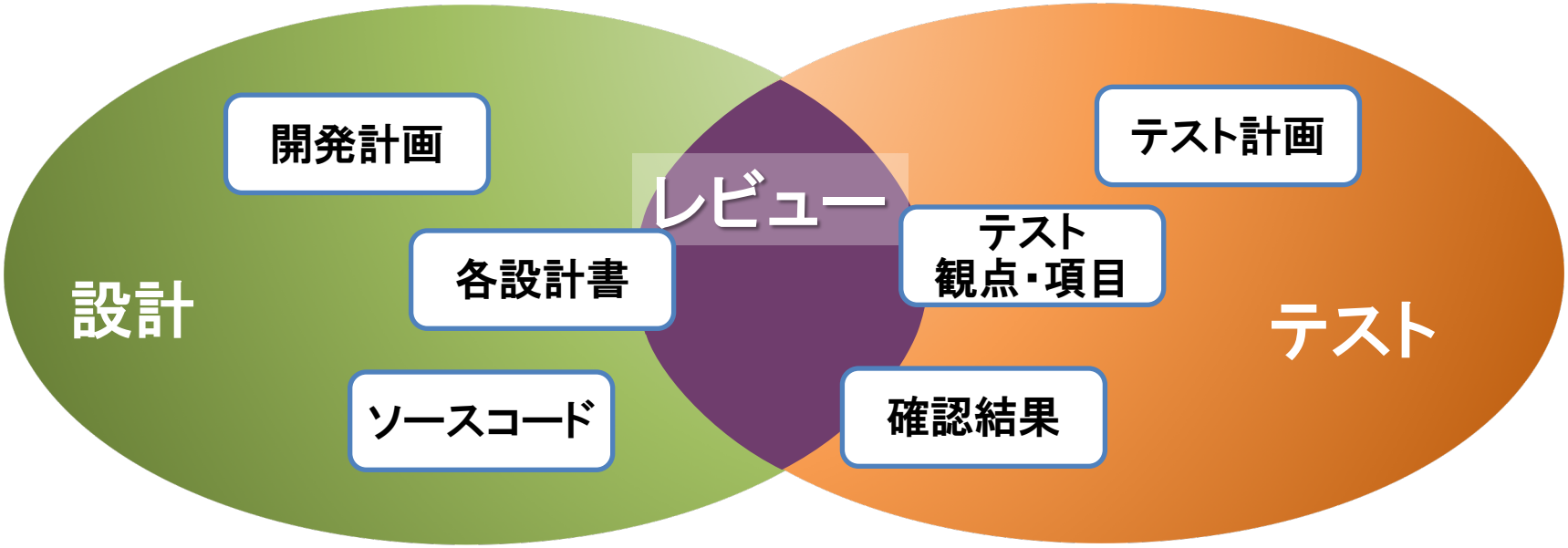
QAは、どのような観点で探し、指摘しているか、

について、少し具体的に説明します

製品開発での各種レビューとQA

HITACHI
Inspire the Next

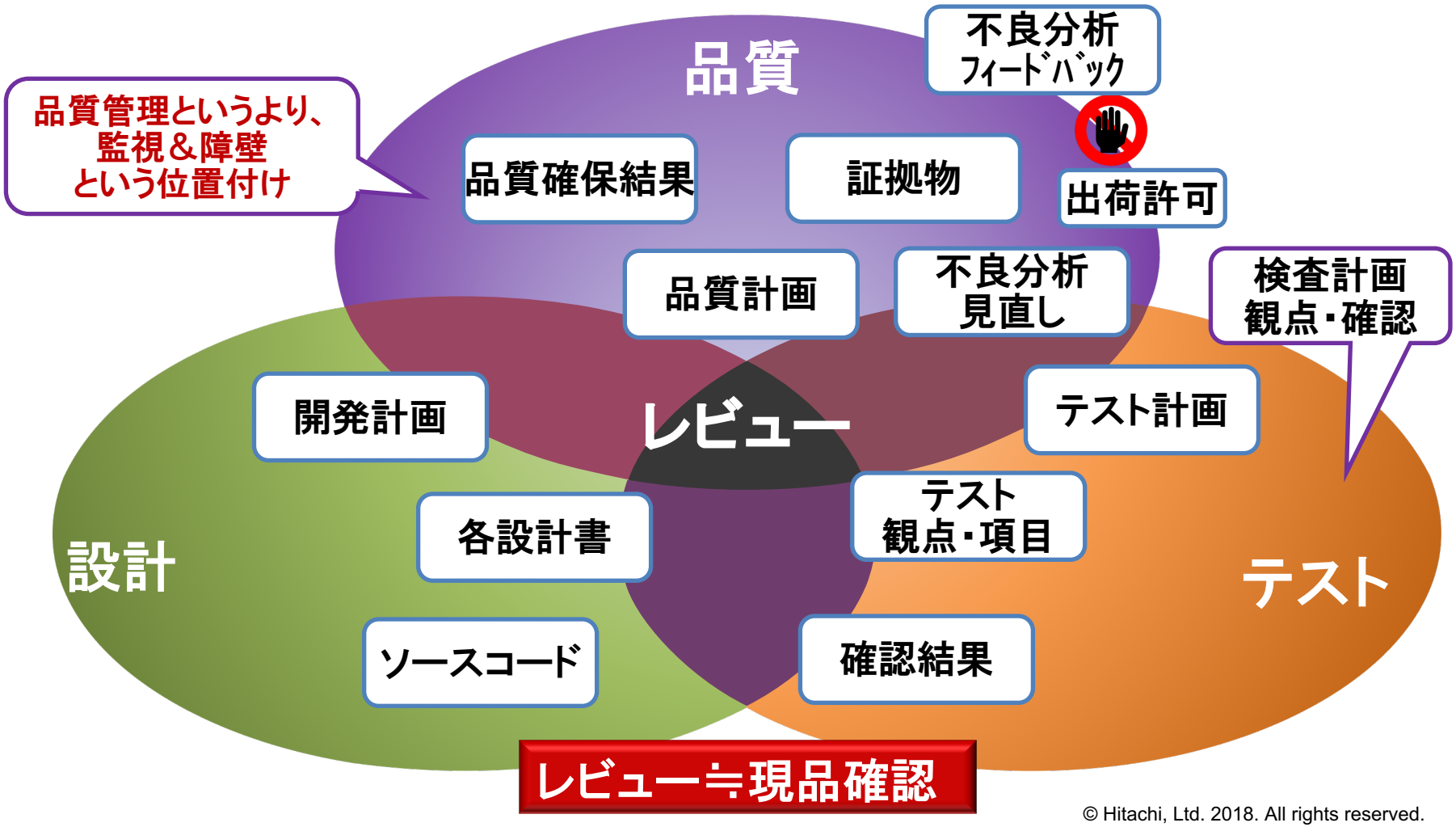
一般にレビューというと、
設計レビューを差す事が多いのでないでしょうか？



製品開発での各種レビューとQA

HITACHI
Inspire the Next

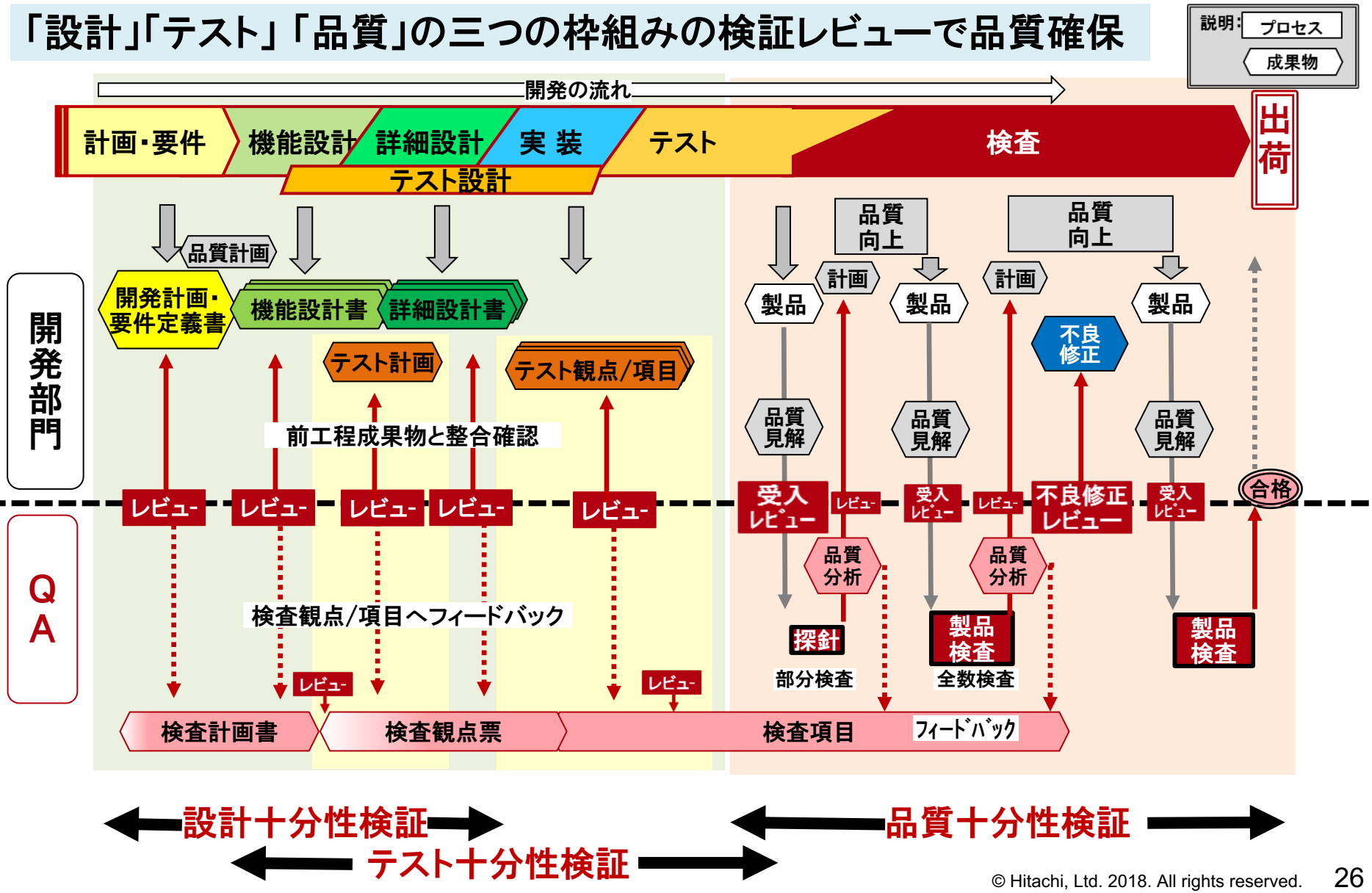
製品品質を確保するためには
「設計」「テスト」＋「品質」の三つの枠組みのレビュー対応



弊社製品開発での各種レビューとQA

HITACHI
Inspire the Next

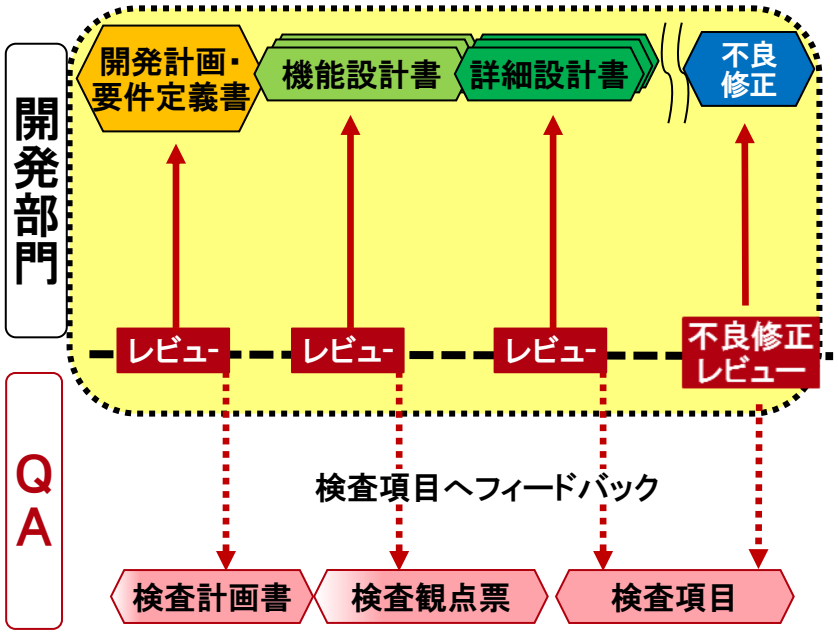
「設計」「テスト」「品質」の三つの枠組みの検証レビューで品質確保



設計十分性検証フェーズ

HITACHI
Inspire the Next

設計内容の検証



検査観点/項目へのフィードバック
各機能のテスト観点や、レビュー結果を基に、検査観点/項目の検討を行う。

◆ レビューのポイント

開発計画 要件設計	・開発や検査日程、体制の妥当性 ・前Verからの反省のﾌｨｰﾄﾞﾊﾞｯｸが十分か ・要件、非機能要件の充足性
機能設計	・各種ガイドやチェックリストに沿っている ・設計内容やインターフェースは妥当か ・表やマトリクス、状態遷移図で設計されているか（検討・テスト漏れ防止） ・曖昧、不適切な文章になっていないか
詳細設計	・顧客運用を考慮した設計になっているか。 ・障害や負荷等の外部条件は、最悪の可能性を考慮しているか。 ・障害復旧機能、原因調査資料は十分か。
不良修正	・修正内容は妥当か。テスト項目は十分か。 ・修正の前提となる関連処理の確認は十分か（サーチ結果等の証拠物での確認）

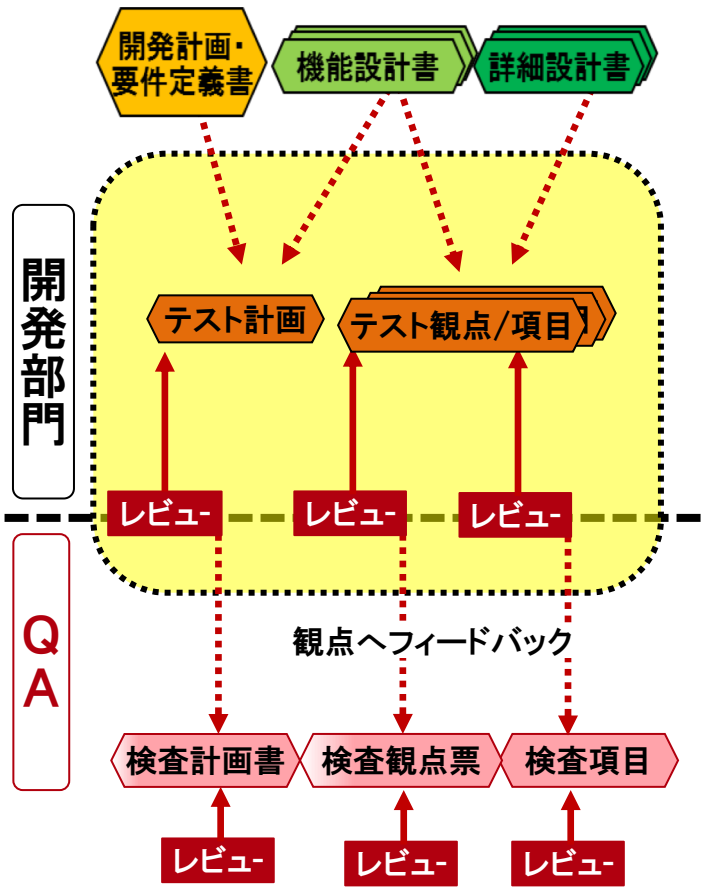
【QAレビューポイント】
QAは、開発部門とは異なる視点で、より広く踏み込んだ検証を行う。
- ◆ 実際に検査する「テスター視点」
- ◆ 関連製品動作含めた「システム視点」
- ◆ 顧客対応で得た「利用者視点」

開発者が気付いていない事、
設計書に書いていない事を挙げる

テスト十分性検証フェーズ

HITACHI
Inspire the Next

テスト内容の検証



設計書と同様に、テスト内容から
検査観点/項目へのフィードバック

◆ レビューのポイント

テスト計画	全体計画。主に時期や環境、体制が十分かを確認。
テスト観点/項目	【網羅】全機能、処理に漏れなく対応していること
	【要件】要件を考慮した確認が含まれている事
	【実装】処理方式を考慮した確認がされること
	【量質】項目数や分類(正常、異常、境界等)は妥当か
	【確認】確認対象の資料に漏れや不足が無いこと

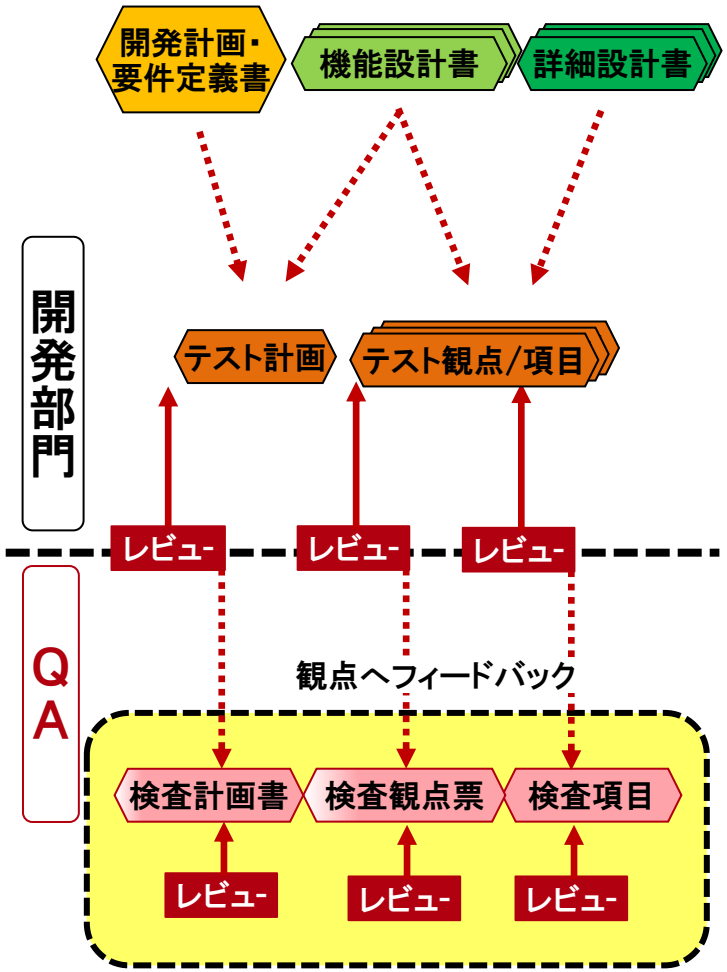
【QAレビューポイント】
テスト計画・・・重要機能テストや性能測定作業がリスクも考慮した日程であること等。
テスト項目・・・機能網羅、要件充足、実装考慮、量と質、確認方法等を検証する。

できるだけ、QA検査の観点を取り込ませ、
逆に、テストでは確認できてない条件を探して、
検査観点に取り込みます。

テスト十分性検証フェーズ

HITACHI
Inspire the Next

検査内容の検証



◆ レビューのポイント

網羅する	機能の**外部仕様を網羅**した確認が十分か（設計書に記載された動作を確認）
より広く	顧客対応から得た**利用者視点**の確認
	関連製品やOSやハードの動作や知識からの**システム視点**の確認
より深く	不具合事例や非機能要件からの確認
	自製品の**内部処理**から導出した**テスター視点**の確認 （内部変曲点、資源確保/解放，競合、等）

【QAレビューポイント】

テストとは異なる視点、テストでは確認が難しい条件が含まれていること

- 過酷条件（高負荷、競合、大規模、延々と繰り返す、等）
- 障害観点（多重障害、障害復旧・回復、等）
- 実際のユーザ運用（日々運用、無停止、保守運用、等）
- 製品組合せ（関連製品と組み合わせた動作確認）

（補足）テストと検査の違い

HITACHI
Inspire the Next

テストは、**処理の十分性検証**が中心（verification）
検査は、**要求に対する妥当性確認**を重視（validation）

お客様の視点からの品質保証部門の検査

・お客様の使い方や
　環境を模して検証
・外部仕様に着目、
　システム整合性を検査
・ブラックボックステスト

＋

品質確保

ものづくりの視点からの設計部門のテスト

・設計仕様通りの動作を確認
・内部構造に着目したテスト
・ホワイトボックステスト

QA独自視点での確認

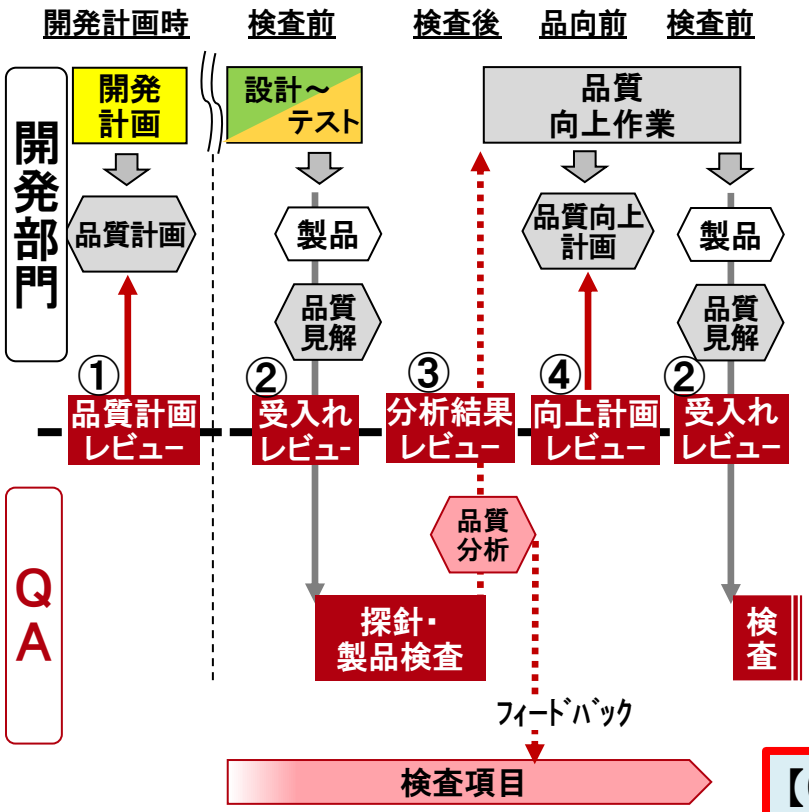
・開発部門のテストでは出来ない条件　（負荷・障害・長時間・・・）
・内部実装を考慮した確認（不具合がありそうな）
・障害時の復旧機能、原因調査資料の十分性→トラブル早期解決に
（製品不具合でも、早く復旧でき、早く原因究明出来れば、皆幸せ）

私は、特にココを重要視してます！

品質十分性検証フェーズ

HITACHI
Inspire the Next

品質監視(計画・結果の検証)



不良の傾向や品質分析結果を
フィードバックし検査項目を拡充する

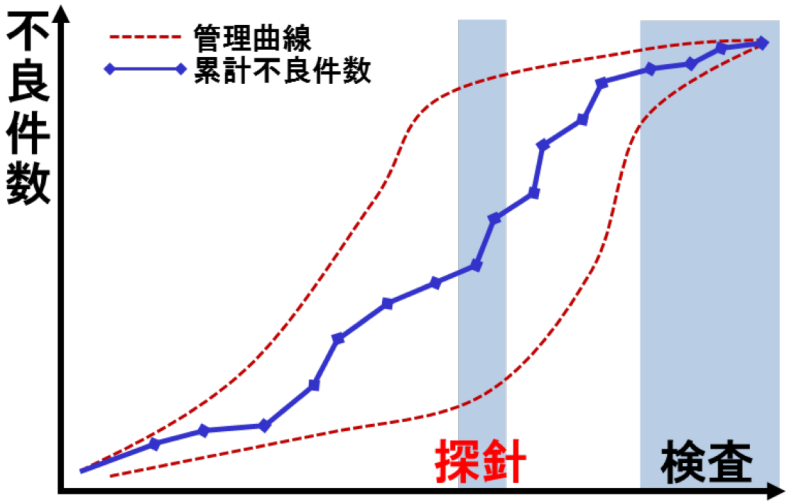
＊探針摘出不良においては「区間推定」から算出。
製品検査不良においては過去Vr実績等から算出。

◆ レビューのポイント

①品質計画レビュー【開発計画時】	品質確保施策や不良目標値等が妥当か、前Ver反省や他製品横展開等の施策取込みされているか確認する。
②検査受入れレビュー（品質見解）	開発工程での品質状況と施策の結果、又は、検査後の品質向上の結果について、検査開始に十分な品質であることを、書面と証拠物等で確認する。
③検査品質分析結果レビュー	QAが、探針、製品検査の結果を分析し、「見直し観点」や「品質向上策」、及び「不良目標数」を提案・要求する。
④品質向上計画レビュー	開発部門は、QAの検査品質分析の内容を基に、不良摘出や、再設計やテストやり直し等の、具体的な品質向上作業を立案。レビューで合意を得て、作業を実施する。

【QAレビューポイント】
・不良の数だけでなく不良の質や傾向の分析結果から、導き出された作業計画であること。
・検査受け入れ時は、計画通りの作業が行われている事を証拠物も含めて確認する

（参考）探針（QP：Quality Probe）について



探針（QP：Quality Probe）とは，
開発工程のテスト終盤で、
QAが、検査項目の「一部」を先行評価し
その品質状況分析結果から
以降の品質向上計画を決める為に行う。

【ポイント】主な目的は、品質分析からの「見直し/品質向上策案」を挙げる事。

【探針で摘出した不良の扱いについて】

- QAは、作込み、見逃し原因の二面に対し、動機的原因まで踏み込んで分析し、同様の問題が内在する危険がある「被疑範囲」の検討や、有効と考えられる「見直し観点」、「品質向上案」を挙げる。
- 上流工程に戻って作業やり直しや、内容によっては開発プロセス改善を要求する。
- 探針時は、併せて「区間推定(*)」を利用し「推定残存不良数」を算出し提示する

＊「区間推定」・・・ 統計学の分析手法。少ないサンプリング数から、想定される実際の数値（幅や悲観値）を算出するもの。

4. レビュー力強化について

HITACHI
Inspire the Next

ここからは、

「気付く」「気付かせる」為のレビュー力強化や

私が「設計レビューで気を付けていること」、

変わった「開発事例」などを紹介します

レビュー力強化について

HITACHI
Inspire the Next

設計レビューのポイント：　人事と人智を尽くす

良い設計レビュー

良いレビューとは「できる事をしっかりやる」こと
- レビューの本質は「他人の知恵を引き出すこと」にある
- 「効果的」に「適切」な知恵を引き出すには、入念な準備・始末は当然で、良い設計書作成、最適なメンバ召集、適切な手法でのレビューが必要

●レビュー準備のポイント
- レビューより前にデザイン
  （書かれていないところに問題は潜む）
- 曖昧表現、不確定表現の徹底排除
  （校閲支援系ツール等も活用）
- チェックリストを過信しない　説明します

● レビュー実施時のポイント
- 設計・レビューで重要な三大思考　説明します
  俯瞰力、先見力、想像力
- レビューアへの注意：日和らない　説明します
- 回覧/メールレビューは要注意

●レビュー実施後のポイント
- 指摘の結論と責任者＋期限の明確化
- レビュープロセスの改善

○レビュー支援系ツール活用　説明します
- いつでも、どこでも(レビュー時間も自動計測)
- Web形式で簡単指摘(回答も入力可)
- 管理も自動化(議事録、各種レポート自動生成)

設計レビューで気を付けていること

HITACHI
Inspire the Next

レビューアは自ら気付くべし！

だから
記述漏れる

設計書は、作成者の狭い知見の世界で記載されること多い。
レビューアは、作成者とは違う視点で、思考を巡らすことが必要。

◆設計・レビューで重要な三大思考力◆

俯瞰する	・連携する全ての機能を考慮しているか ・連携機能とのインタフェースは妥当か？
先見する	・使用したリソースは、いつ、どうやって解放されるのか？ ・連続して使い続けた場合、大丈夫か？ ・障害復旧後はスムーズに再開できるのか？
想像する	・もし、想定してない障害が起きたら、どう動くのだろう？ ・どんなユーザも使い易い仕様だろうか？ ・xxxと書いてあるが逆、別のケースがあるのでは？

思考を変えるだけで
知識不足を補う事が
可能！

文章表現が
よくない箇所は
要注意です

【私がレビューで危険視している文章パターン】
長い文章　・・・改行が無い。句読点が少ない。は、分解不足のケース多い。
曖昧語使用　・・・一部の条件を示す「場合」。主語が不明確な「される」等。
例で表現　・・・例を書く事は良いですが、記述をサボっているケース多い。

設計レビューで気を付けていること

HITACHI
Inspire the Next

レビューアは日寄らない。発言し気付かせるべし！

×勝手な解釈で「指摘しない」ことが無い様に

書いていない事に気付かないのは論外です

書かれていない事に気付いたけれど、
勝手な自己解釈で、発言しないケースありますよね。。。

【まずい自己解釈の例】

リソース不足時の動きが書いてないな。。。でも、シビアな環境では使わないだろう。

いろいろ書いてないな。。。でも、チェックリストには書いてあるから、大丈夫だろう。

～ けど、誰も気にしてないみたいだし、指摘は控えよう。

～ けど、たぶん似たような機能と同じ動きという事だろう。

～ けど、この担当者はベテランだから大丈夫だろう。 ありがち

指摘でなくても、質問すれば、関係者間で認識を合わせておく事ができますね。

レビュー指摘を阻害する人間心理

作成者「記載は、なるべく少なくしたい。。。」

みんな「レビューは、早く終わらせたい。。。」

嫌われても、
心を鬼にして、指摘が出来る
強い人間に私はなりたい

設計レビューで気を付けていること（チェックリストの功罪）

HITACHI
Inspire the Next

チェックリストを過信しない！

「チェックリストは英知の結晶」

○チェックリストの効果

○「暗黙知」である注意点や失敗事例を「形式知」化したもの。
○レビュー前に設計書類の品質底上げ。効率化・・・etcに効果が高い。

×チェックリストの問題点

×同じチェックリストでも、文章解釈は十人十色。使う人間の経験・能力次第。
×「チェックリストで確認すれば大丈夫」という過信や依存。
×古いチェックリストの形骸化、多すぎるチェックリストによる疲弊。

思考停止する

チェックリストのウィークポイントを認識した運用が必要

➔基本は安易に増やさない。ツールや自動確認の仕組みで。
➔定期的にチェックリストを見直す。減らす。補足追記。教育化。

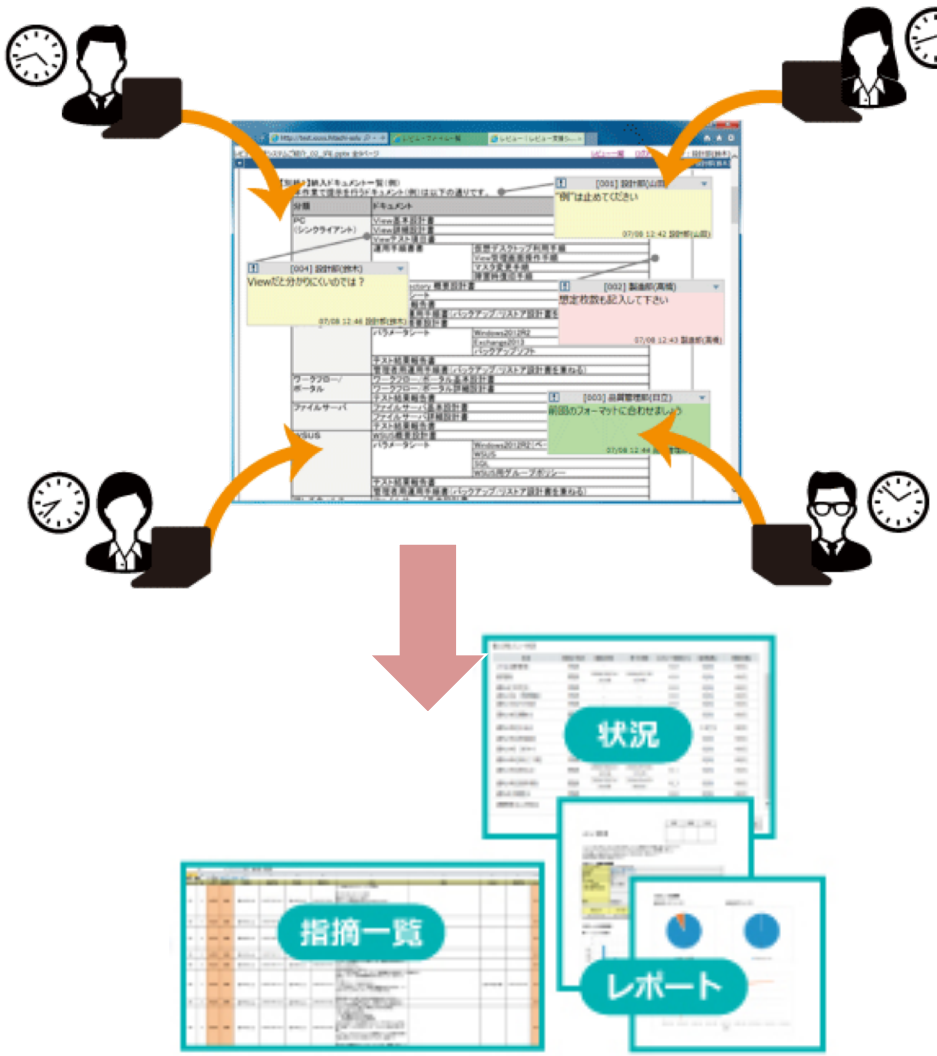
【私の悩み】

増え過ぎたチェックリストの削減は難しい
「将来、同じ失敗を絶対にしない」とは言い切れないから

設計レビューで気を付けていること

HITACHI
Inspire the Next

レビュー支援系ツールを活用しよう！



●知恵を引き出しやすくなる環境に!
- 対面レビュー時間が大幅削減
- レビュー付帯作業も軽減

一般的なツールの機能

各自のペースで集中しレビュー

◆ ドキュメントをWeb共有
➔時間や場所を問わずレビュー

◆ 他レビューアの指摘も同時共有
➔指摘の重複を防止

レビューの質も向上

◆ 全レビューアのレビュー時間など把握可
➔レビュー状況管理、フォローも可能

レビュー準備・始末作業も軽減

◆ 議事録の自動作成機能で省力化
➔作成者も指摘対策に集中

（参考）レビュー以外での取り組み紹介

HITACHI
Inspire the Next

開発スタイルに合わせたレビュー・テストの連携が重要と思ってます

ウォーターフォール型開発での課題

設計書類の数・量が多くなる

↓

作成やレビュー工数が増加

↓

工程遅延や設計者の疲弊

➔予め設計書作成量やレベル等の作業軽減策を計画する事、支援系ツールの積極活用が有効

アジャイル型開発での課題

設計書の作成・レビューを絞る

↓

ドキュメントに書き起こす事で防止できるレベルの問題が発生

➔更なるテスト等での早期問題把握と対処、必要であれば、ﾄﾞｷｭﾒﾝﾄ作成+再設計。

ウォーターフォール型開発にアジャイル的思考を加えた開発モデル

弊社ＤＢソフト開発で適用

基本はウォ-タ-フォ-ル型開発。
重要機能は機能単位に開発・テスト。
早期段階でQAが部分検査(FDD？)。
自動ビルド・テスト環境。

- 重要機能の品質や仕様問題を早い段階で発見。早い段階からの製品対策が可能に。
- 検査確認ジョブは部分検査時に自動化し、最終製品検査は短い期間で実施。
- 最初に機能毎の設計ﾄﾞｷｭﾒﾝﾄ作成・ﾚﾋﾞｭｰ要否を設計-QAで決めて開発

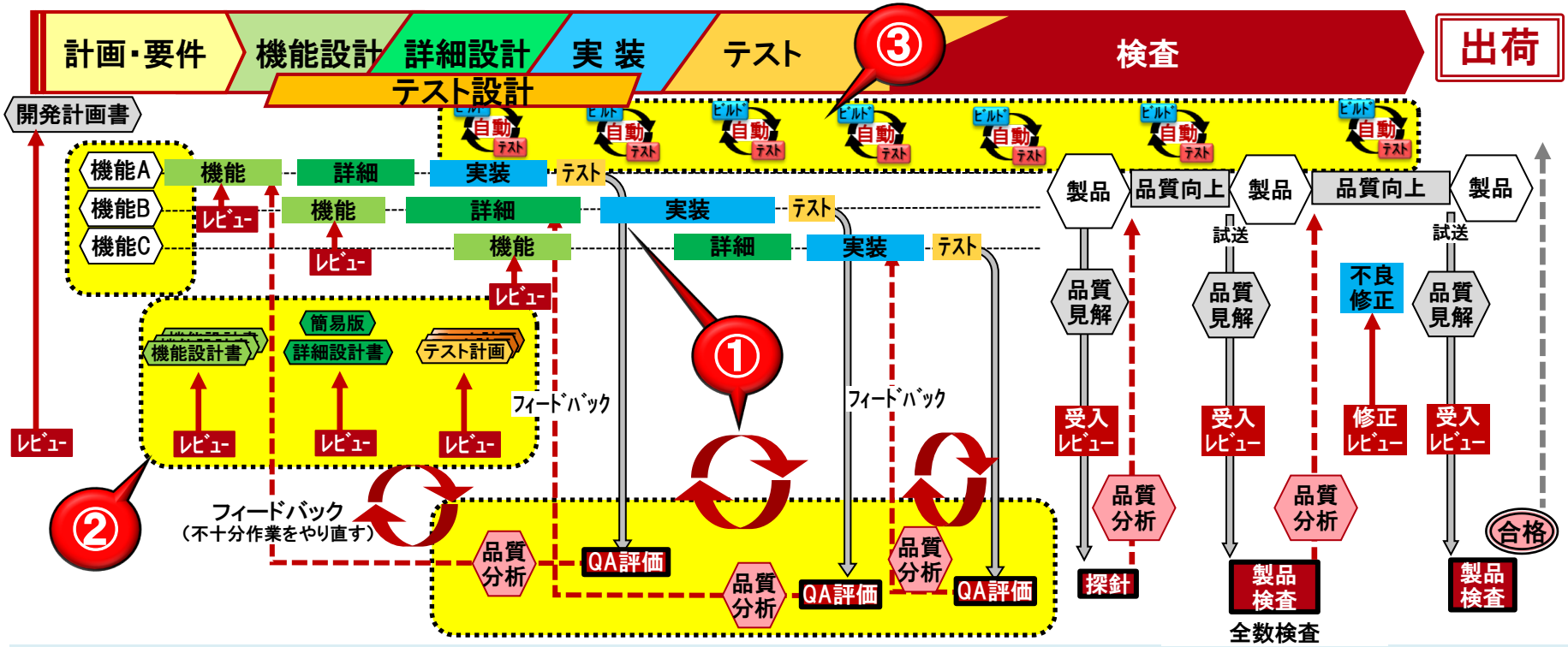
➡設計書を完璧に作成しない事によるリスクをテスト強化・早期の検査評価で軽減

（参考）レビュー以外での取り組み紹介

HITACHI
Inspire the Next

ウォーターフォール型開発にアジャイル的思考を加えた開発モデル

ドキュメントを減らす品質リスクを、テスト強化、早期の検査評価で軽減。



① 重要機能は先に開発。QAが事前評価テストを実施。結果を早くﾌｨｰﾄﾞﾊﾞｯｸ。検査ｼﾞｮﾌﾞ自動化。

② 重要度/難易度の高い機能のみ詳細設計書を作成（他は簡易版）。

③ 自動ビルド・テスト、検査ジョブ自動化によるテスト/検査繰返し・効率向上

メリット　➔比較的短期で開発。早い段階で問題発見でき、難しい対策も可能。

デメリット➔品質目標が決め難い。詳細設計をしない悪影響（バグ）も散見。・・・更なる改善が必要。

5. 最後に

HITACHI
Inspire the Next

私はこう思っています。

本発表の纏め

HITACHI
Inspire the Next

私はこう思っています。

ソフトウェア開発の要は、所詮「人」。

人の考え あっての、人の行動。

当然、自分はやる、

でも、他人にもやってもらうためには

どうするべきか。

本発表の纏め

HITACHI
Inspire the Next

私はこう思っています。

■ 組織の品質意識の醸成

品質への「責任感」と「危機感」を持って行動すること。

■ 他人と異なる視点や思考

積極的な顧客対応や、他業種知識、市場ニーズ、トレンドの取込み。

「書かれていないところに問題は潜む」ことを意識した設計。

そういう「人」を育成することが重要。

一番大事

ご静聴ありがとうございました。

少しでも参考になれば幸いです。

ぜひ、ご意見をお聞かせください。

HITACHI

Inspire the Next