TOSHIBA

〔W EB公開用〕

Redmine活用によるプロジェクトの見える化の実現

東芝インフラシステムズ株式会社
2019.03.12　　金子　博

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

本日のアジェンダ

0　自己紹介＆製品紹介

1　取組みの背景

2　問題対策とツール活用

3　まとめ

4　今後の展望（DevOpsを目指して）

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

0　自己紹介＆製品紹介

1　取組みの背景

2　問題対策とツール活用

3　まとめ

4　今後の展望（DevOpsを目指して）

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

開発の問題（社会インフラ系システムでのあるある）

老朽化

- システム寿命が長くアーキテクチャが長期利用される

- アーキテクチャの劣化
- コードの複雑化
- データ設計の陳腐化
- システム更新の難度化

ユーザ特異性

- ユーザー優位
- 追加仕様を押し込まれがち

- システム仕様の不整合
- オペレーションプロセスの複雑化
- 実装やデータ管理の複雑化

力業開発

- テストの実機依存
- アドホックな擦り合せ開発の常態化

- 工程後期での仕様変更
- テスト工数の増大
- バグ修正コストの増大

IoT時代の開発に逆行し「ガラパゴス化」へ

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

開発管理の問題

個人管理

- 成果物登録までは個人管理

個人ルール

- 共有フォルダは独自管理

都度収集

- 必要な情報は都度収集＆mail拡散

デジタルデータのアナログ管理が問題！

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

プロセス改善の問題

SPIって？

- 過去のSPI失敗により後向きモード化

やってるよ！

- データ入力してまーす！

無駄な努力

- 基幹システムは開発管理に不向きなのに…

手段先行や目的が正しく理解されないことが問題！

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

0　自己紹介&製品紹介

1　取組みの背景

2　問題対策とツール活用

3　まとめ

4　今後の展望（DevOpsを目指して）

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

問題対策

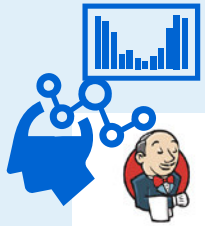
4つの視点で対策

1 開発資産・環境の整備

ソースコードを始めとした開発資産を
正しく運用できる様に教育、環境整備、
技術支援、ソースコード改善を実施

2 プロジェクト管理の改善

Redmineへのデータ集約と脱Excel
個人ファイル管理を実施
プロジェクトデータ分析ノウハウの自
動化と分析結果の見せる化

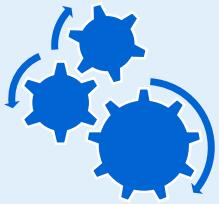


3 テストデータ収集の自動化

テスト管理ツール、バグ管理、自動テス
トツールの連携によるシームレステス
ト環境の構築と運用

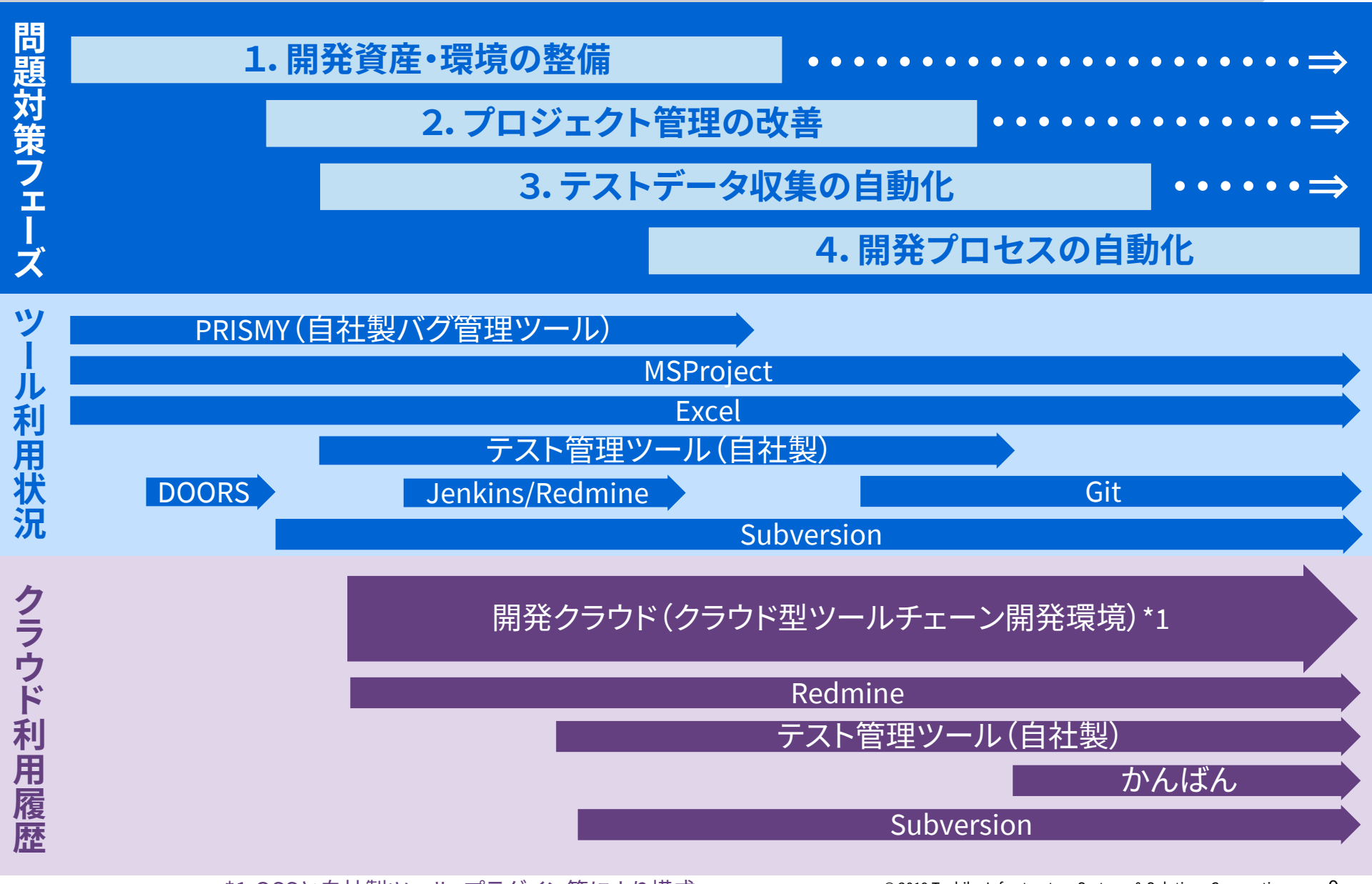
4 開発プロセスの自動化

属人化プロセスの形式知化とプロセ
スの自動実行



© 2019 Toshiba Infrastructure Systems & Solutions Corporation

対策フェーズとツール導入



*1:OSSと自社製ツール、プラグイン等により構成

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

ソフトウェアエンジニアリングの改善

教育とツール、グローバルリソース活用で既存資産改善！

1 教育の実施

正しいソフトウェアエンジニアリングの定着に向けた教育

- 要求管理
- システムソフトウェアエンジニアリング

2 コーディング技術の改善

ツール活用と作法の改善

- 静的解析*!の普及と義務化
- コーディング規約の普及展開
- コードレビュー支援

3 バグ修正プロセスの改善

バグ改善プロセスの指導やバグ修正指導の実施

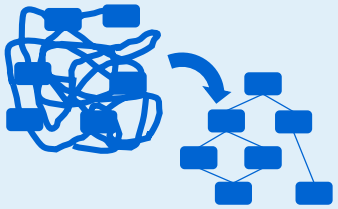
- Redmine適用推進
- 協力会社を巻き込んだ見える化推進
- PMO組織の設立と運営

REDMINE
flexible project management

4 アーキテクチャの改善

長期流用により劣化したアーキテクチャの改善

- アーキテクチャ改善
- モダナイゼーション化
- リファクタリング
- µITRON→Linux化



*1:社外製有償ツールを活用した社内向け静的解析サービス

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

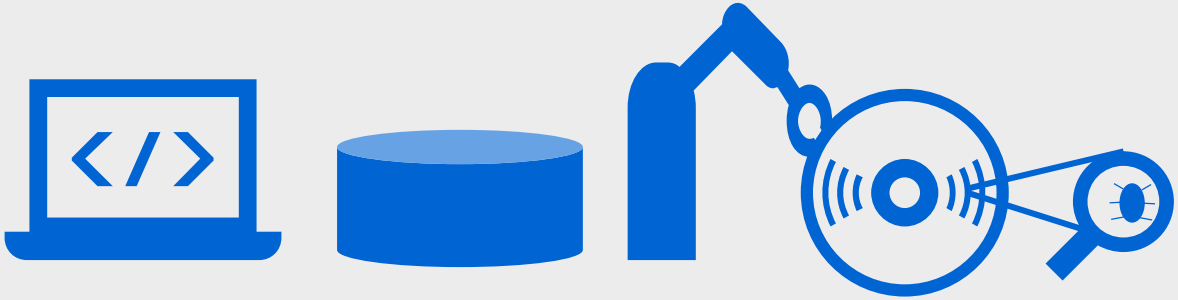
静的解析で品質の見える化とツール連携

1 ツールで見える化

- 社内の静的解析サービスを義務化
- 実績を重ねることで対策が推進
- メトリクスの自動収集
  - 複雑度、コード量他
- コードの問題指摘
  - コードクローン、メモリーリーク他

2 構成管理ツールと連携

- ソースコードがアップデートされると自動で解析実施
- 指摘件数推移なども容易に見ることができる



有識者レビューが有効も、最低レベル引き上げに有効

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

モダナイゼーションの分類

1 リビルド	現行システム仕様を元に新たに開発する手法。現行システムの問題となっている非機能要件を満たす仕組みを実現できるが、時間とコストが掛かり、運用実績不足による品質への不安伴う。
2 リプレース	新しいベンダー製品等をベースにシステムを置き換える手法。移行先製品の機能差分によっては、新たな開発が必要で、コストや安定稼働までの時間などに不安が伴う。
3 リライト	コンバージョンツールなどを用いて新言語環境にコード変換、書き換えをする手法。従来のプラットフォームやフレームワークなどの継続利用が困難な場合や、発展的に新しい環境に適用させる場合がある。
4 ラッピング	現行システムから大きな変更をせずに、I／Fを中心に機能の一部を利用できる様にする手法。容易に対応できる反面、システムの構造が複雑になる可能性が高く、一次的な対応に留めることが望ましい。
5 リインターフェース	新デバイスに対応するため、ユーザーインターフェースを改良する手法。現行システムの機能を変更せずに利用性を向上させられる。
6 リホスト	CPUやOS、ミドルウェアなどの刷新、保守切れに伴い、現行システムを移行する手法。新しい環境に対応する為に、設計の見直しが広範囲にわたる可能性もあり、コストと品質に影響がでる可能性がある。
7 リファクタリング	長く差分開発を行ったソースコード名での保守性や性能改善を目的として、ソースコードを修正する手法。I／Fと機能を維持しつつ、クローンコード対策やロジック効率化などが一般的な方法。
8 リドキュメント	現行システムのドキュメントを再整備する手法。差分開発が続くことでメンテナンスが不足しがちなベースドキュメントの整備が肝となるが、当時の設計者が居ない場合もあり、コストと時間が掛かる。

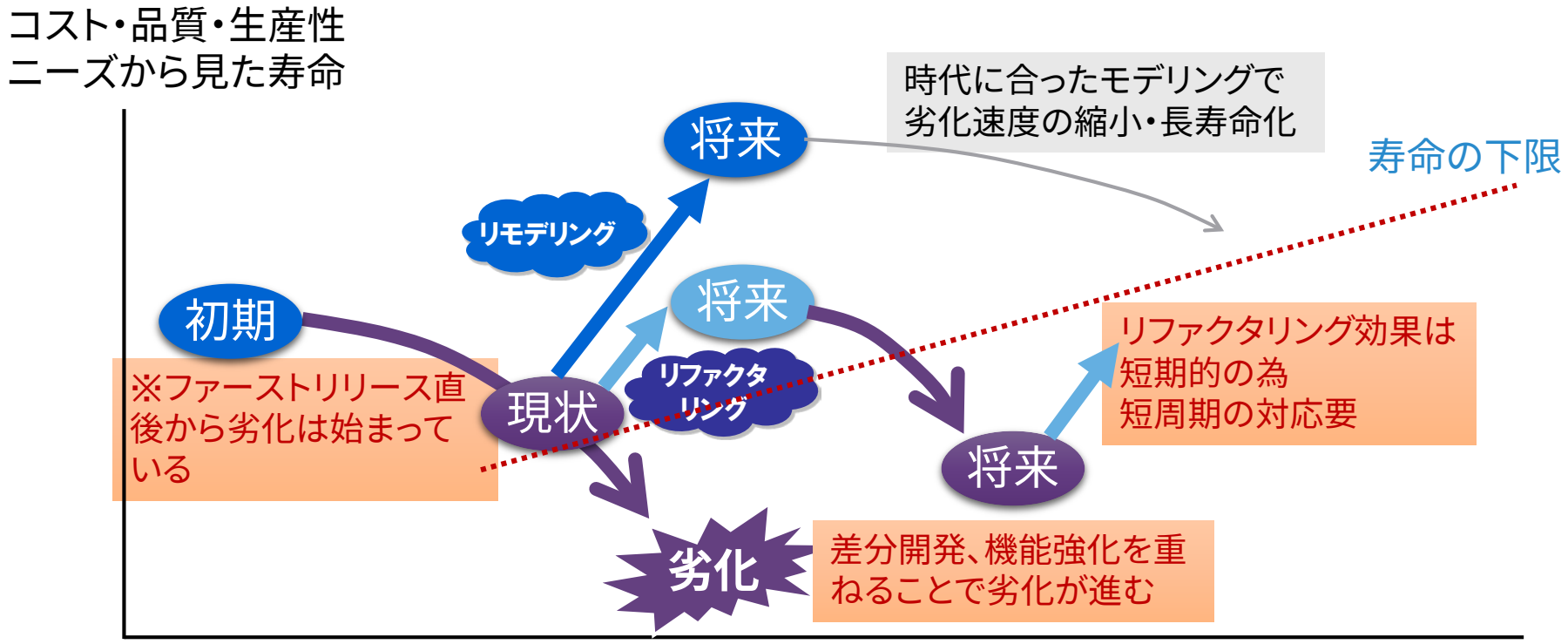
© 2019 Toshiba Infrastructure Systems & Solutions Corporation

過去のしがらみからの解放と価値を生む開発へのシフト

新しい価値への適応性対応が必要

長期の差分開発による不良資産化
- 場当り的差分開発により類似機能が重複/分散
- 個別顧客対応で構成管理が複雑/崩壊
- アーキテクチャの劣化による品質・コストの悪化

技術的負債



© 2019 Toshiba Infrastructure Systems & Solutions Corporation

既存資産の整備

大きなコストを掛けずに今後の寿命を延ばすための施策を実施→品質・工期への無用な不安を回避

Before

差分開発の積み重ねで改修コストが徐々に増加

- クローンコードの増殖
- アーキテクチャの陳腐化
- 専用エンジニア不足(μITRON)
- パッチの積み重ねによるバグの深度化
- 旧型機器依存

After

モダナイゼーションの実施で延命化
段階的進化で不安を回避

- リホスト：Linu化
- リモデリング：アーキテクチャの一部を見直しシステム責務変更を実施
- リファクタリング：コードの正常化
- リインターフェース

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

ＩPA：モダナイゼーションの解説本

組込み系システムでもソフトウェアの範囲はほぼ同じ

https://www.ipa.go.jp/sec/reports/20180214.html

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

2. プロジェクト管理の改善

データの一元管理化（脱Excel化）と分析の自動化！

1	2	3	4
プロジェクトデータの集約	テスト管理データの集約	AI*管理の集約	データ分析ノウハウの自動化
プロジェクトデータのRedmineへの集約（脱Excel化） ・バグデータ ・成果物管理 ・課題管理	テスト進捗管理の脱Excel化 ・テスト計画/実施管理ツール活用 ・テスト管理ツールデータのRedmine連携	ActionItem管理の脱Excel化 ・Redmine連携機能提供 ・Redmine活用方法の普及展開	PJデータ分析ノウハウの自動化と見える化 ・re'dashの活用 ・Redmineへの組込み

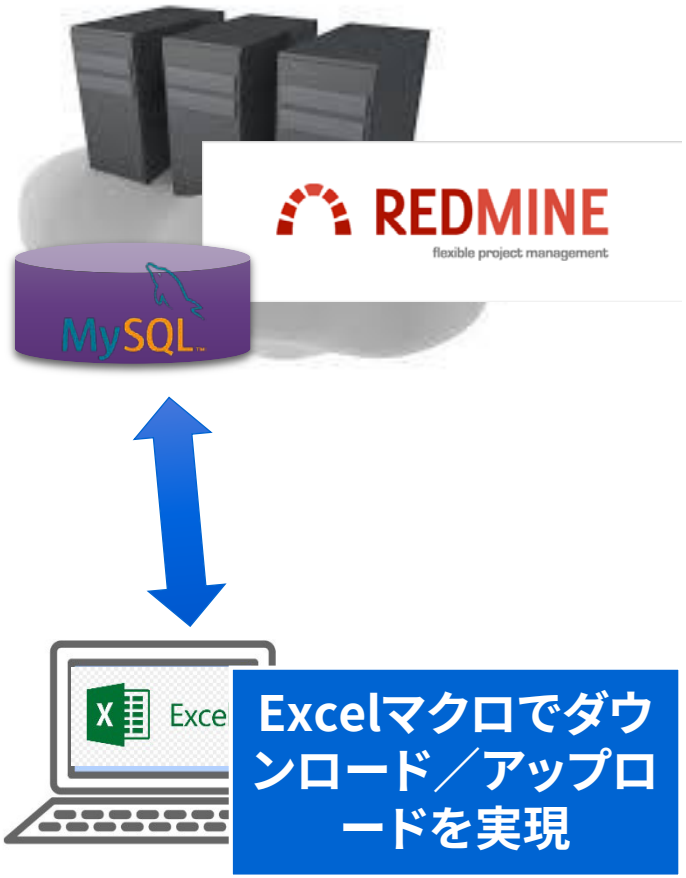
*AI:Action Item

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

脱Excel施策

◆ チケットドリブンの課題（管理者視点）

- リーダーにチケット処理依頼が集中し停滞
- チケット毎に処理が必要なのが手間
- ステークホルダーとの共有にはexcel必須



- Excelとチケットをリンクするマクロを作成
  - 双方向で利用可能
- 顧客AIとチケット表現の違いを吸収
- ログイン処理を簡素化

Excelマクロでダウンロード／アップロードを実現

手厚いサポートで何とか普及！

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

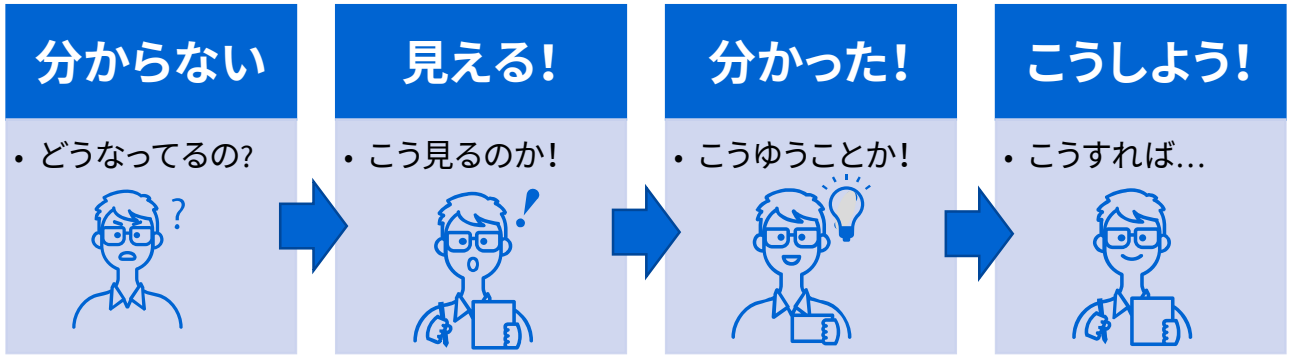
「見える化」、「見せる化」で変わる！

- チケット発生、終息推移
- 担当者別チケット状況
- 工程別WT実績集計
- バグ曲線(テスト管理ツール,Redmine連携)
- テスト進捗率
- カテゴリ別バグ状況一覧
- カテゴリ別バグ分布（一覧＋円グラフ）
- バグ原因別分布（一覧＋円グラフ）
- ソースコード修正箇所へのリンク（SVN連携）

…

ツール連携効果を具体的に提示

見える化（グラフ化）すると分かることがいっぱい

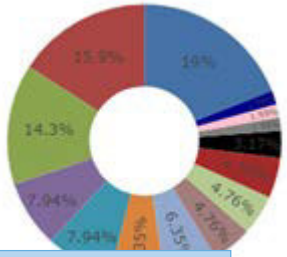


© 2019 Toshiba Infrastructure Systems & Solutions Corporation

プロダクト品質メトリクス分析「ノウハウの形式知化」を推進

繰り返しやることを自動化!

グラフは定期的に更新



Plugin/Jenkinsを駆使して

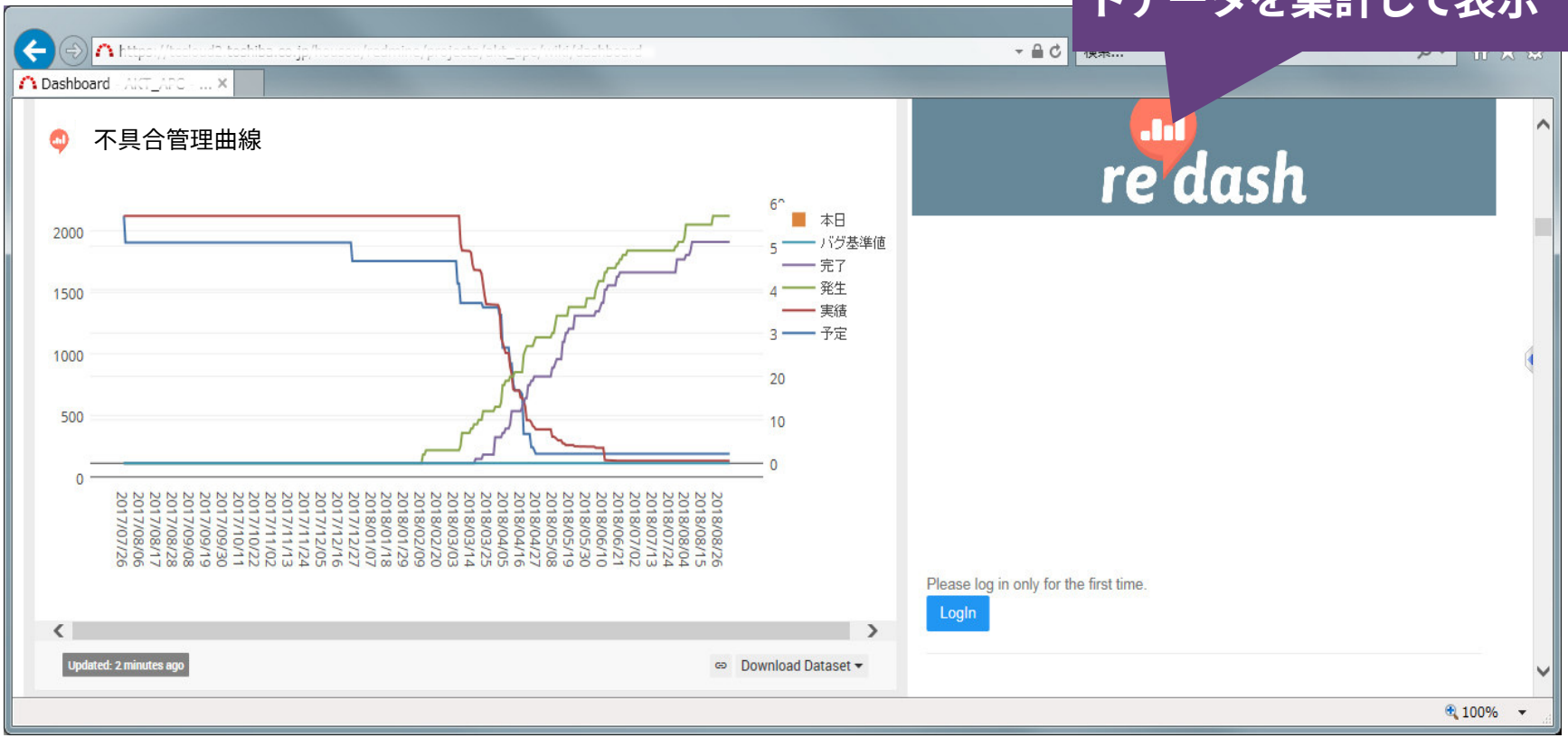
Redmine+αで開発データを自動的に集約

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

テスト進捗管理連携

テスト進捗管理

テスト管理ツールデータと
Redmine上の不具合チケットデータを集計して表示



© 2019 Toshiba Infrastructure Systems & Solutions Corporation

バグ分析グラフ

目的別グラフ分析

■作成される分析の例

- 問題点カテゴリ別
- 機能分類別
- 原因分類別
- 装置分類別
- テスト工程別

など

■分析表示の例

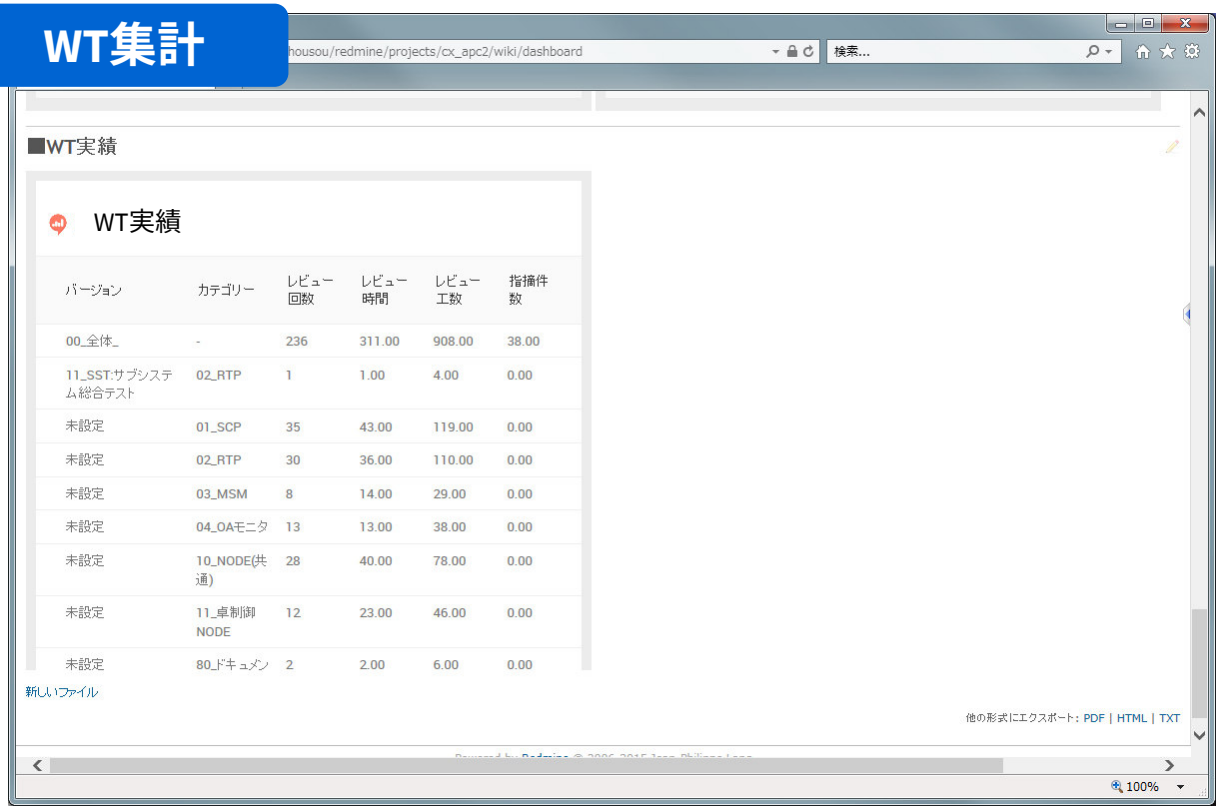
- 円グラフ
- リスト
- ファイル出力(CSV,XML)

分析して欲しいグラフを定型化

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

WT議事録から自動集計

WT集計



バージョン	カテゴリー	レビュー回数	レビュー時間	レビュー工数	指摘件数
00_全体_	-	236	311.00	908.00	38.00
11_SST:サブシステム総合テスト	02_RTP	1	1.00	4.00	0.00
未設定	01_SCP	35	43.00	119.00	0.00
未設定	02_RTP	30	36.00	110.00	0.00
未設定	03_MSM	8	14.00	29.00	0.00
未設定	04_OAモニタ	13	13.00	38.00	0.00
未設定	10_NODE(共通)	28	40.00	78.00	0.00
未設定	11_卓制御NODE	12	23.00	46.00	0.00
未設定	80_ドキュメン	2	2.00	6.00	0.00

■集計項目

- レビュー回数
- レビュー時間
- レビュー工数
- 指摘件数

など

■分類

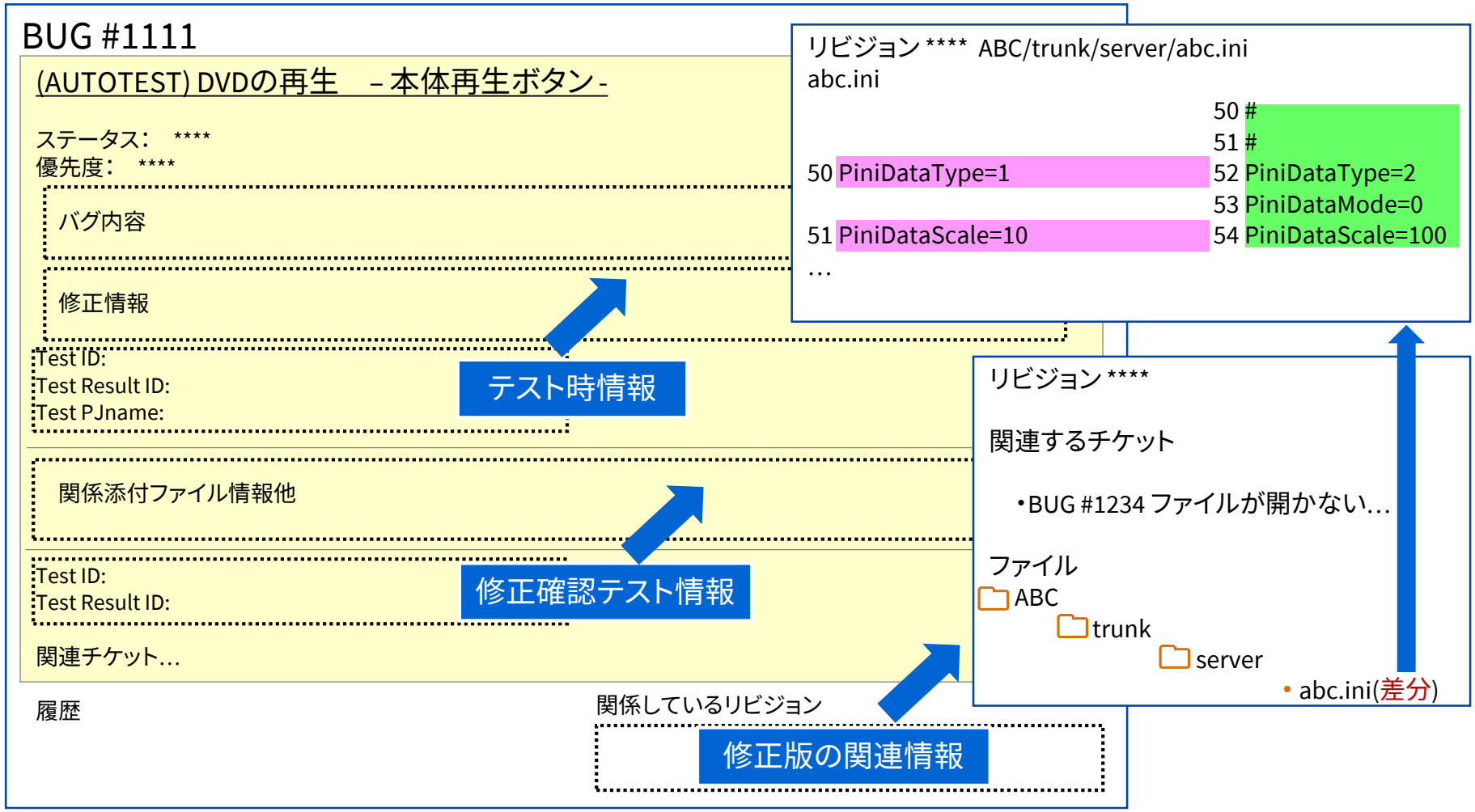
- 開発工程
- サブシステム名

など

議事録チケットフォームを定型化して集計を楽に

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

テスト情報（独自）、ソースコード情報（基本機能）を連携



関連情報を連携して、レビュアーや承認者の負荷を軽減！

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

PJ実行中や節目レビュー時に活用

最低限の分析を定常的に実施して対策遅れを防止

Before

- 問題が大きくなってから分析
- 節目レビューの直前に分析（時間切れによる不十分な対策）
- 毎回同じ指摘を受け続ける

対策遅れでQCDに影響
（QA部門の頭痛の種）

After

- 定期レビュー時にダッシュボードを見ながら対策を検討
- メトリクスの目標クリアに向けた事前対策の常態化
- **データ白書データの自動収集**

プロアクティブなプロジェクト管理でQCD改善

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

シームレステスト管理までの道のり

関係者理解と環境整備により自動化環境を整備!

1	2	3	4	5
バグ管理の改善	テスト管理の改善	テスト自動化	テスト戦略の実践	シームレステスト管理実践
バグ管理ツールの正しい運用とタイムリーな登録への改善	テスト仕様書管理と実行管理の改善	自動テストの実施とテスト環境の構築	ベテランテスターノウハウの形式知化と効率的テストスクリプトの流用	テスト計画から実行までの連携と、バグ発見時自動起票までのシームレス連携
・バグ管理ツール運用の改善 ・Redmineへの変更	・テスト管理ツール導入と運用改善	・Ranorex導入 ・オフショアテスト環境の構築 ・仮想実行環境の構築	・スクリプト変換ツール開発 ・ラルフチャート導入	
REDMINE flexible project management		Ranorex		

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

テスト自動化への取組み

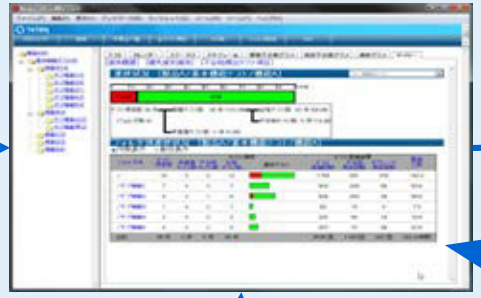
テスト仕様・スクリプト・結果の一元管理で可視性・保守性を向上

・テスト管理ツールのIDと紐付けてテスト仕様を管理し保守性を向上

テスト仕様書

テスト仕様

テスト管理ツール（自社製）



テスト成績書

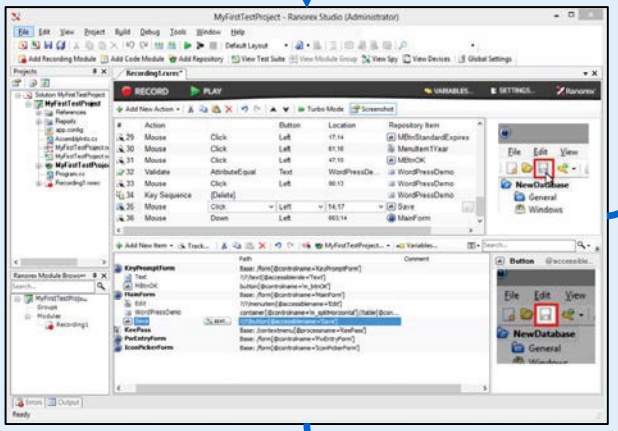
仕様＋結果

・サマリー機能でテスト進捗を即座に確認

テスト
スクリプト

自動テストツール

Ranorex



テスト
結果

・Ranorexの実行結果をテスト管理ツールに自動登録

Bug管理
（チケット）

REDMINE
flexible project management

テスト実行ログを元に、RedmineにBugチケットを自動登録

テスト
実行ログ

Bugチケットサンプル

チケット参照者が確認し易い情報を自動登録

Bug　#***
(AUTOTEST) DVDの再生　– 本体再生ボタン -

ステータス：　New　　Test ID：　112233
優先度：　Normal　　Test Result ID：　332211
…

SUCCESS：
✔ VERIFY：　DVDがTV画面で再生すること。　期待値：True　実際値：True
✔ VERIFY：　プレーヤーの状態が再生中であること。　期待値：True　実際値：True

FAILURE：
✘ #1：　VERIFY：　タイムレコードがカウント開始すること。　期待値：True　実際値：False

合否判定

合否判定判断が容易

[発生事象]
・[入力切替]ボタンを押す。
・[HDMI1]を選択する。
・DVDを挿入する。
・[再生]ボタンを押す。
・VERIFY：　DVDがTV画面で再生すること。　期待値：True　実際値：True
・VERIFY：　プレーヤーの状態が再生中であること。　期待値：True　実際値：True
・FAILURE #1
　- VERIFY：　タイムレコードがカウント開始する。　期待値：True　実際値：False
・[停止]ボタンを押す。
・DVDを取り出す。

テスト実行履歴

テスト実行手順確認が容易

[発生日時]
SCRIPT START TIME：　2019-10-10　10:10:10
- FAILURE #1：　OCCURRENCE AFTER 0:25:28

テスト実行時刻

テスト実行時の機器運用情報との参照が容易

実機のテスターにも使いやすい環境を提供!

◆ 面倒なことを自動化
- テスト管理ツール設定を定型化
- チケットフィールドの自動設定
- テスト管理ツール登録の自動化
- 試験結果（Excelデータ）とRedmineチケット発行の連動

現場

Excel

Excelファイルを
自動で取り込み

テスト管理
ツール

Excelでリリース

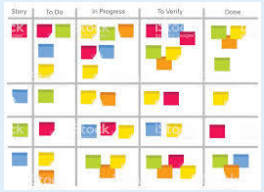
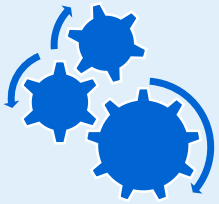
Jenkins

設計

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

属人化プロセスの改善（自動化とDX）

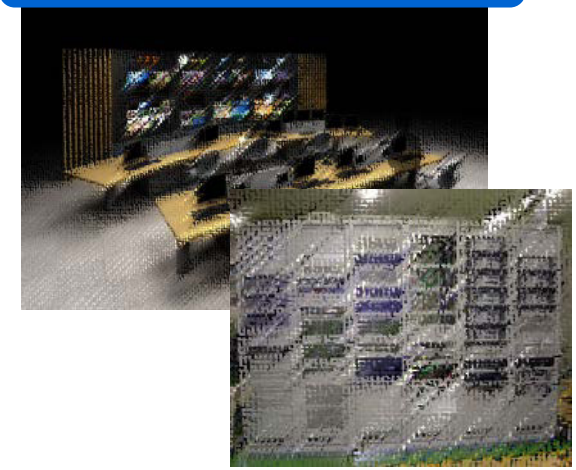
非効率化（属人化）課題の抽出と対策による効率化！

1	2	3	4
業務プロセスの改善	手作業の自動化	作業の見える化	シームレスなツールプロセス化
「かんばん」の実践と慣れ	データ変換ツールの作成や連携の拡大 ●データフォーマットバリエーション対応 ✖指定フォームへの対応拒否	属人化作業の見える化 ・PFD作成 ・手順書作成（かんばん活用）	繰り返し実行が可能な自動化（DX）の実現
			

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

仮想テスト環境整備とオフショア活用

大規模組込みシステム



クラウド上に仮想化

REDMINE
flexible project management

docker

主要監視装置群をリアルに再現

オフショアからの利用

- 実機レス(輸送不要)
- 版管理の容易化
- テストデータ連係
- 時差の有効活用
- バグ分析の容易化
- 実行環境整備コスト削減

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

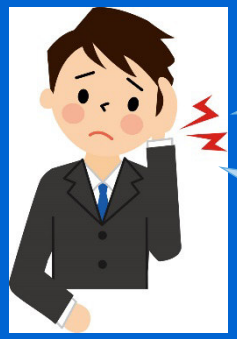
設計補助作業に“かんばん”プラグインを活用！

実は、これがRedmineの食わず嫌いのハードルを下げたかも？

設計補助担当者への依頼を看板で運用

Before

- 複数の依頼者が想定されるため作業負荷状況が見えづらい
- 依頼しようとしてもいつまでにやって貰えるのか分からない
- 短時間作業が多く常に作業を抱えている様に見える



After

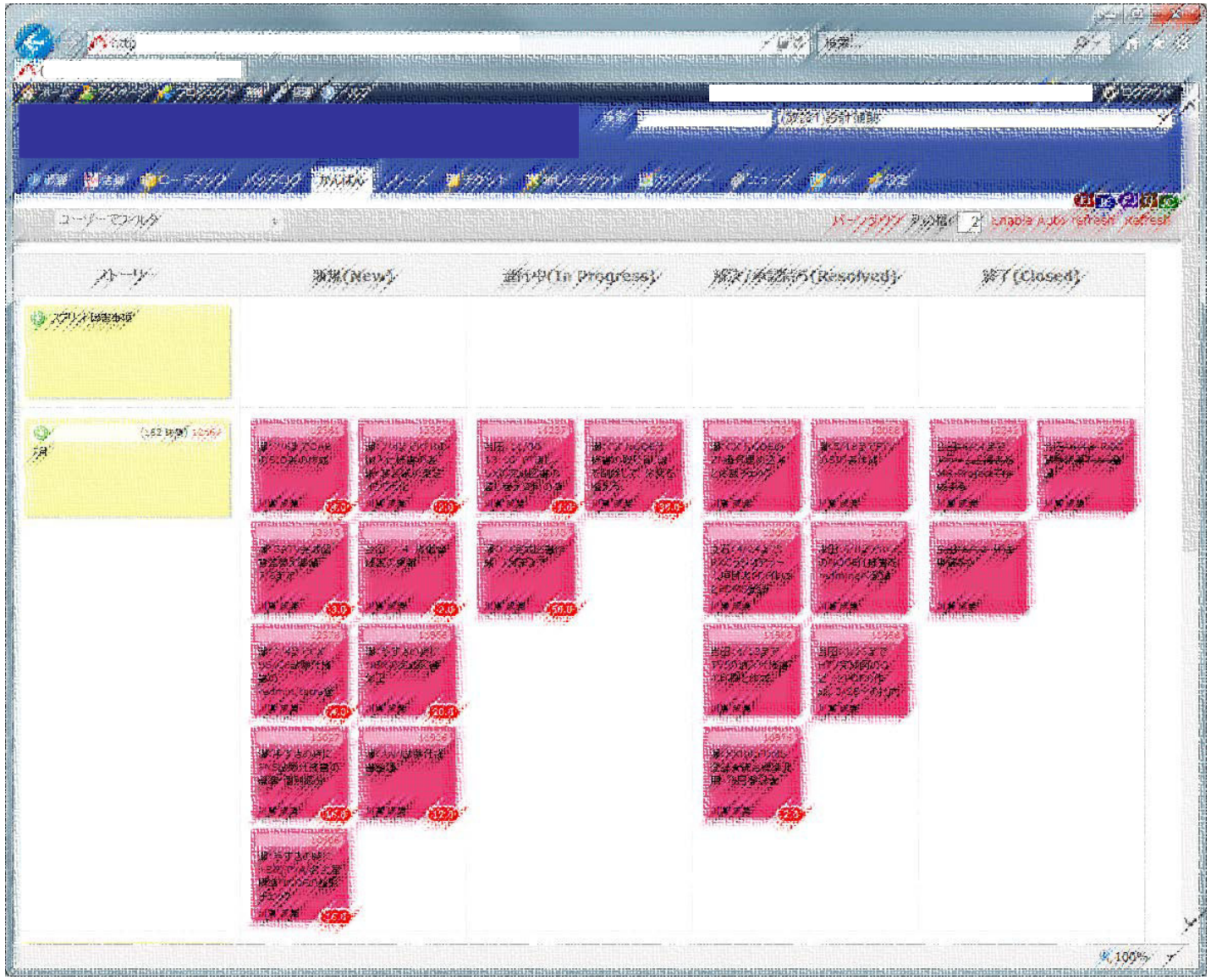
負荷状況が一目瞭然！
進捗も分かりやすい！
毎日の作業内容が分かって
仕事がし易くなった！

Win-Win

毎日の作業内容が
分かって進めやすい！

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

タスクの進捗と負荷状態が一目瞭然

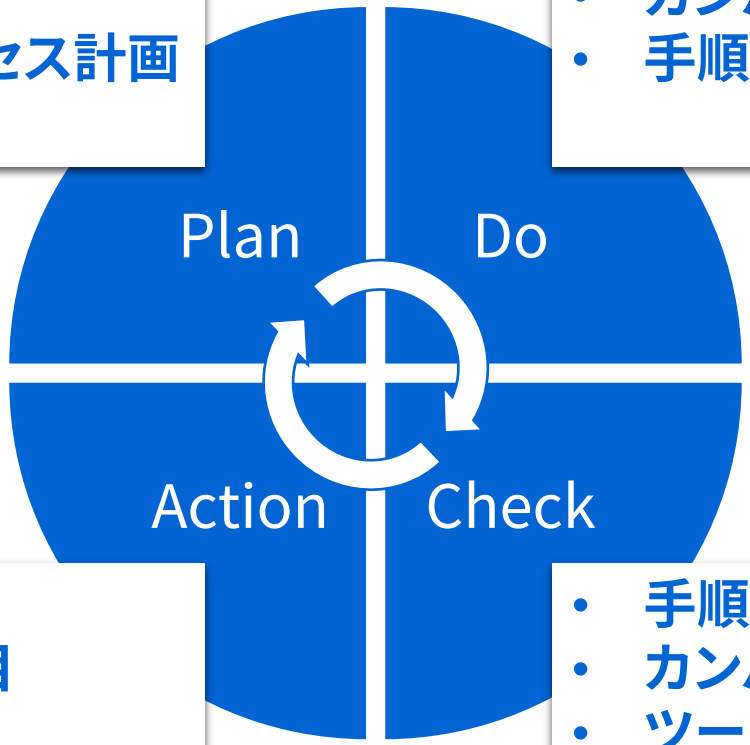


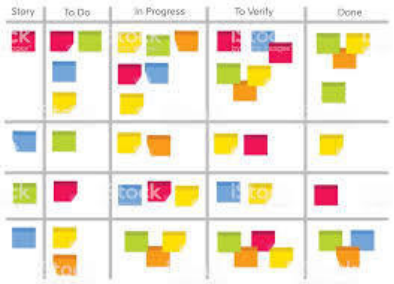
© 2019 Toshiba Infrastructure Systems & Solutions Corporation

カンバンによる属人化プロセスの改善例：実践型かんばん

- 問題分析
- カンバンプロセス計画
- 第三者実施

- カンバン1巡目の実施
- 手順書作成





- ツール改善
- カンバン2巡目
- 手順書確認

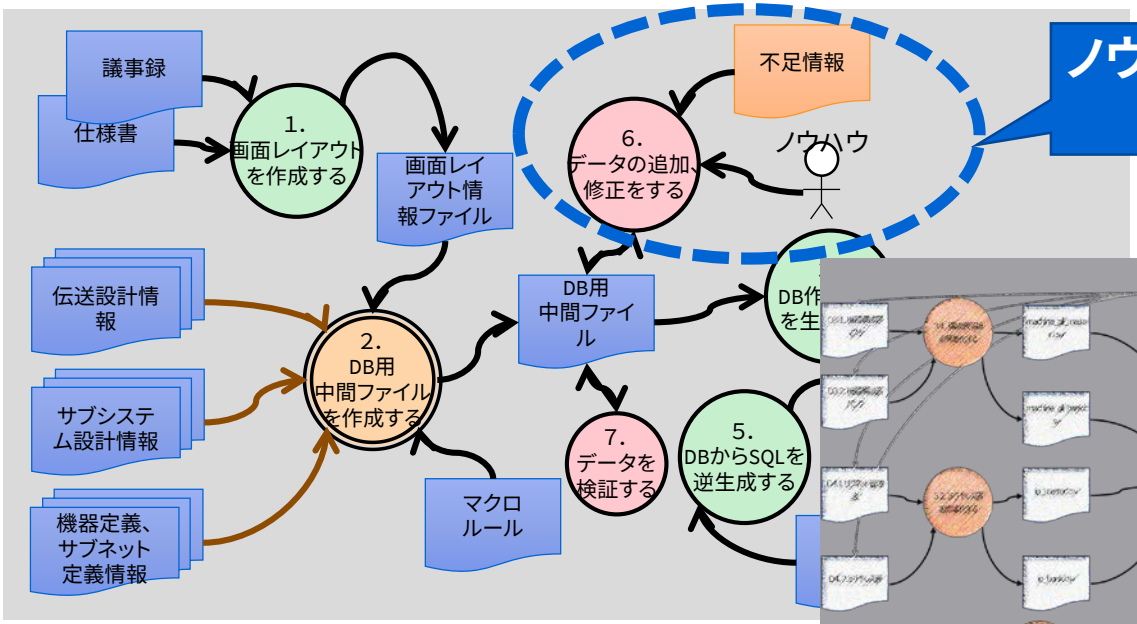
- 手順書分析
- カンバンプロセス分析
- ツール効果分析

かんばん2巡で改善を実感！

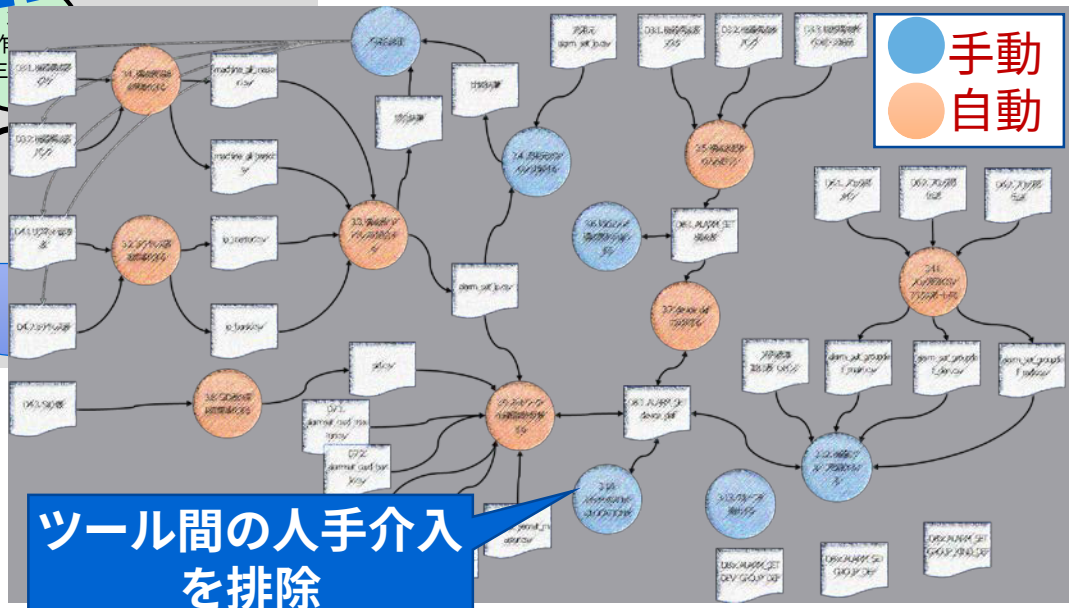
© 2019 Toshiba Infrastructure Systems & Solutions Corporation

かんばんで業務効率化を実現！

- 第三者により「個人プロセス」の「見える化」を実現！
- 既存ツールのボトルネックを発見！
- 人手の介入削減で効率アップ！
- 属人化作業の自動化による効率化を実現！



ノウハウ／属人性のあぶり出し



ツール間の人手介入を排除

© 2015 Toshiba Infrastructure Systems & Solutions Corporation

0　自己紹介＆製品紹介

1　取組みの背景

2　問題対策とツール活用

3　まとめ

4　今後の展望（DevOpsを目指して）

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

まとめ…スマートに書くと

- **内部推進者が必要**
  - 担当者が使わなければ効果が無いし、効果を実感すれば継続して使われる
- **但し目標と改善のステップの明確化と共有**
  - 目的が曖昧で、関係者の合意がとれていないと効果が得られない
  - どうなるべきかのイメージを事前に擦り込む
    - 好事例の共有、経験者の意見を聞く
- **計画的な地ならし**
  - 先ずは慣れること（しかし我流が定着しない範囲で）
  - 現場毎の課題をあぶり出して対策
- **効果が出るまで手厚いサポート**
  - 野放しは我流が横行、結局使われなくなる
  - レスポンス良くサポート（作業を請け負ってはダメ）
- **組織的バックアップも必要**
  - 管理職、役職者などへの理解を得るべく根回しが必要（使わなくても効果を理解してもらうには…?）
  - 仕掛けを作る時間と費用の確保が重要

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

まとめ...実際は苦労だらけ

◆理解者を増やすのが大変...**なにくそ**
- 改革・改善には痛みが伴うがケアが大切（怒鳴られたり、けなされたり...）
- 少しずつ効果を実感してもらい広める
- 「やってみなはれ！」（面倒な理由の正当化、保守的な評論家対策が必要）

◆完璧なツールはない！カスタマイズ最高！...**プロは自分で**
- 現状に合わせたカスタマイズや臨機応変な運用が大切
- Redmineはプラグインが豊富＆アイデアが勝負！Jenkinsにお任せ！

◆Project管理が実は大変...**さぼると**
- 使い方
- FORMAT
- アクセス権

◆製造／FTに使って貰えると成功...**やっぱり**
- 製品と一緒に現地に移動でコミュニケーションが困難な職場にフィット

変革への強い意志と献身的なケアで効果を実感！

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

今後の期待

1 品質分析機能の強化

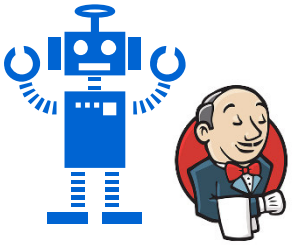
- 分析グラフの追加
- 入力データ精度の向上
- 自動集計（ツール連携）の強化

品質課題の提示と早期対策体制の構築と、分析ノウハウの形式知化

2 日常業務の自動化

- 属人化作業の自動化
- 共通作業の自動化
- 基幹システムデータとの連携

により、作業効率化の推進



3 コミュニケーションロスの改善

- mail依存からの脱却
- タイムリーな情報共有の実現

Slackなどのチャットツールの導入などにより、コミュニケーションの促進と内容の記録を推進

業務効率化とノウハウの共有を目指して

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

0　自己紹介＆製品紹介

1　取組みの背景

2　問題対策とツール活用

3　まとめ

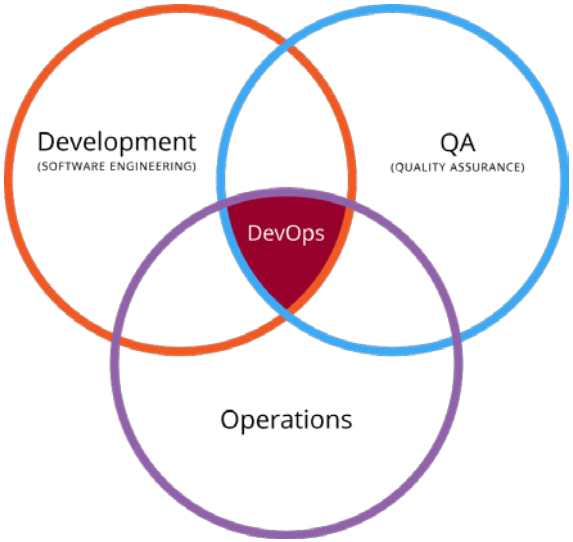
4　今後の展望（DevOpsを目指して）

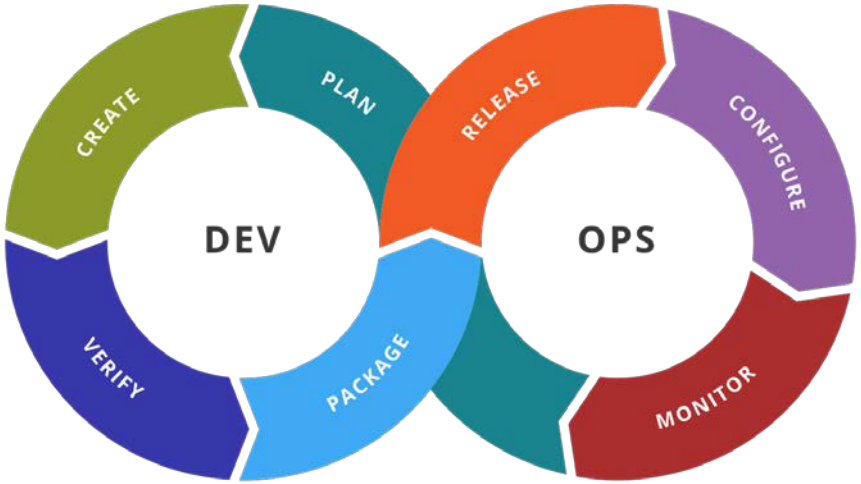
© 2019 Toshiba Infrastructure Systems & Solutions Corporation

展望：DevOps

「10+ Deploys Per Day: Dev and Ops Cooperation at Flickr」@Velocity 2009 – O'Reilly Conferencesがオリジナル*1

DevOps（デブオプス）は、ソフトウェア開発手法の一つ。開発 (Development) と運用 (Operations) を組み合わせたかばん語であり、開発担当者と運用担当者が連携して協力する（さらに両担当者の境目もあいまいにする）開発手法をさす。厳密な定義は存在しておらず、抽象的な概念に留まっている。ソフトウェアのビルド、テスト、そしてリリースの文化と環境を以前よりも迅速に、頻繁に、確実に発生する確立を目指している。





ウィキペディアより：https://ja.wikipedia.org/wiki/DevOps

*1:https://www.slideshare.net/jallspaw/10-deploys-per-day-dev-and-ops-cooperation-at-flickr

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

DevOpsのメリット

DevOpsは真の働き方改革を実現する!

1 組織連携の強化

機能リリースサイクルが早くなるため、顧客ニーズの反映や、戦略の変更に柔軟に対応できるようになります

2 単純作業からの解放

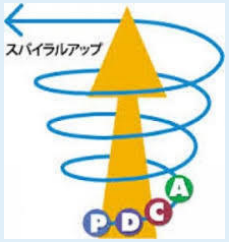
リリースやデプロイの自動化が欠かせないため、自動化が前提となり、これらに関わる作業はツールが支援することになり、価値の高い作業に注力できます

3 人為ミスの削減

属人化作業の撲滅が前提となり、リリースやデプロイ作業における人為ミスがなくなり、後戻り作業がなくなり前向きな作業に集中できます

4 製品価値を高める

リリースまでの作業が高速化されるため、短いサイクルで機能追加・改善ができ、顧客価値を継続的に高めることができます



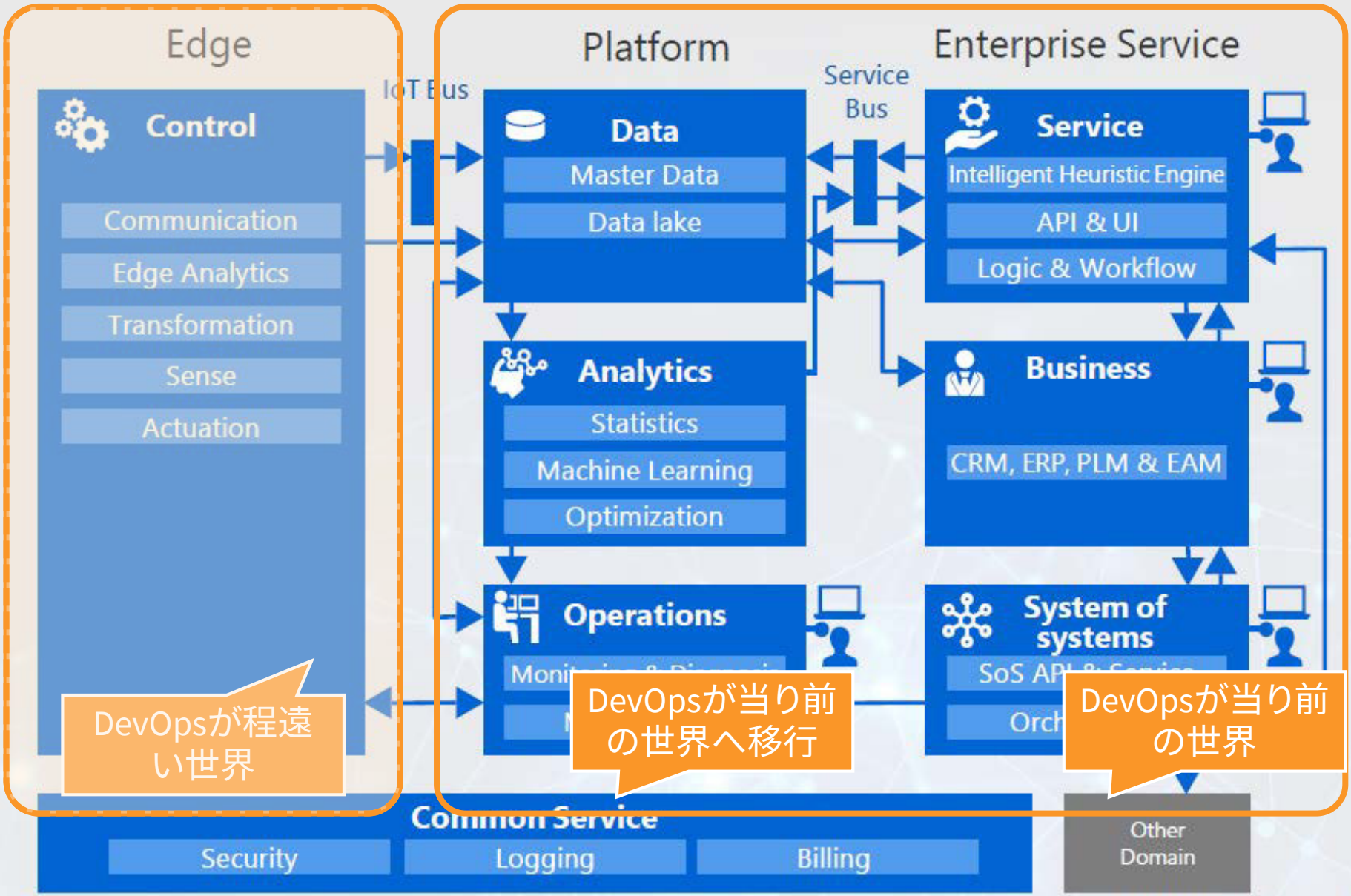
© 2019 Toshiba Infrastructure Systems & Solutions Corporation

DevOps実現の条件

01	開発手法	アジャイル、反復型開発、かんばん
02	構成管理	Git,Subversion,ansible,...
03	CI環境	Jenkins,自動テスト,仮想実行環境,...
04	CD環境	自動アップデート、Jenkins,ansible
05	コミュニケーション	Redmine,チャット（グループウェア）

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

Toshiba IoT Reference Architecture Ver2.0 (3 Tier Architecture)



© 2018 Toshiba Corporation

DevOps化を目指して

手作業

- 人に聞く
- 人が頑張る
- Excelで個人管理
- メールでばらまく

手作業の自動化(属人化脱却)

- 設定作業の自動化
- グラフ化(PJ)
- テスト管理連携
- セットアップ自動化
- テストスクリプト化
- 仮想実行環境構築
- トレーサビリティー管理（USDM）
- かんばん

CI化

- テスト自動実行と結果の自動収集
- コード管理改善
- 静的解析ツールの自動実行
- 仮想実行環境のDockerコンテナ化

CD化

- デリバリー自動化
- 仮想環境と自動テスト連携

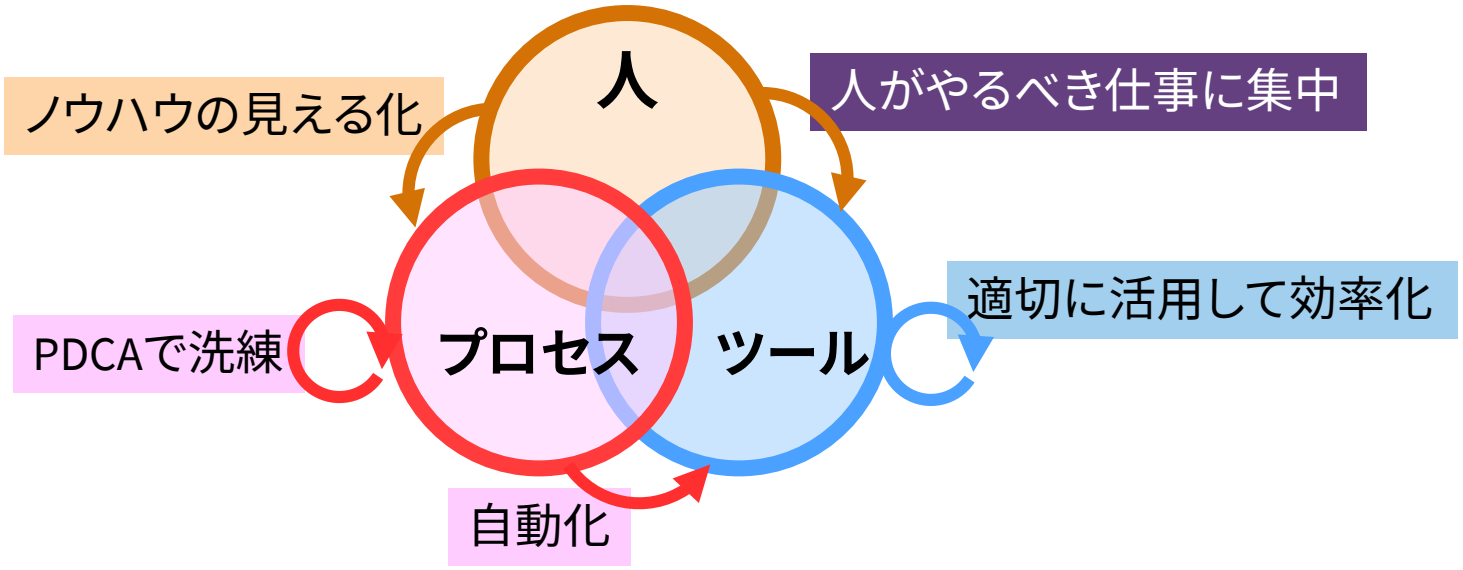
DevOps化

- アジャイル
- CPS検証
- テスト仕様書自動生成

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

我々の目指す姿

- 個人管理（属人化）の撲滅
- 繰り返し作業の自動化
- データの自動収集環境の整備
- コミュニケーションロスの削減
- 分析ノウハウの自動化（QA、PMノウハウの適用）



DevOps化(ゴール)を目指して自動化を推進！

© 2019 Toshiba Infrastructure Systems & Solutions Corporation

TOSHIBA