

# ソフトウェア分散開発 のためのテスト環境

Session-2

小野塚 荘一

日本IBM

ラショナル事業部・シニアITスペシャリスト

- 分散開発の課題
- 分散開発のためのテスト環境
  - テストツール
  - 変更とリリース管理 – 分散開発のインフラ
- まとめ

# ビジネスの現状と分散開発

## ビジネスの状況

- 企業買収、統合
- 状況の急激な変化
- 予算カット、経費削減
- スキルの高い人材不足
- 状況に応じて雇用調整が必要
- 開発のバックログ
- 短期開発、低コスト開発

グローバル開発  
と  
デリバリー

新しい問題

## 問題

- プロセスの誤解とミスマッチ
- コミュニケーションの問題
- 異文化理解の問題
- 生産性の低下
- やりなおし作業の増加
- 技術移転での誤解
- 調整の経費の増大
- プロジェクトの透明さ、コントロールの容易さ、問題対応の迅速さの低下
- 数値による状況把握が困難、達成基準が不明
- 人間関係の複雑化
- 情報漏えい対策が困難

# 分散開発の開発・テストサイト例

## コアサイト



### 調整役



### 製品品質とプロジェクト管理(高い技術スキル)



### システム開発のリソース管理(人材、機材、データ、予算)



### コアサイトにも開発のためのリソースは必要



### プロジェクトのデータ・リポジトリと管理が必要



### コアサイトにリモートからアクセス可能な環境を用意



## サブサイト



- リソースを確保
- メトリックスの値をベースにコアサイトと協業
- プロジェクト・データ・リポジトリとリモートアクセス環境を構築
- セキュリティレベルの設定

## 独立サイト



- プロジェクトの拡張
- 独自にプロジェクトおよびリソースの管理
- メトリックスの値はコアサイトへ伝える
- プロジェクト・データ・リポジトリとリモートアクセス環境を構築
- セキュリティレベルの設定

## リモートサイト



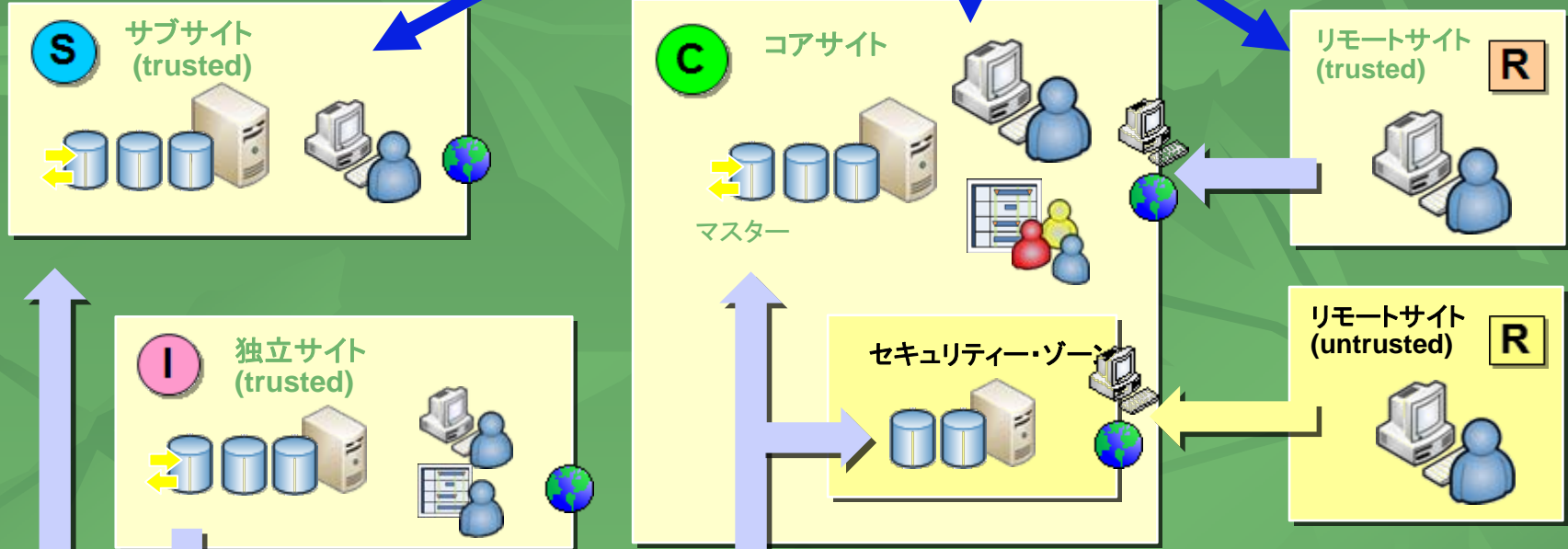
- 開発者はコアサイト、サブサイトのサービスをとおして、アセットへアクセス
- モバイルユーザー、少人数のグループ
- 最小限のインフラ構築
- セキュリティレベルの設定

### 考慮点:

- ▶ 分散環境のガバナンス
- ▶ 分散環境のデリバリー・チェーン
- ▶ 分散環境のリポジトリ
- ▶ セキュリティゾーンの設定
- ▶ リモートアクセスの提供

# 分散開発のパターン例

テスト環境 負荷テスト、回帰テストなど



プロジェクト・リポジトリ 構成管理、変更管理、要求管理、プロジェクト管理、ビルド管理、  
テスト管理、品質管理

# しかし、...

- 品質低下をどのように防止するか
- コミュニケーションが不安
- 要件、仕様書のすり合わせで誤解はないか
- 納期を本当に守れるか
- ...

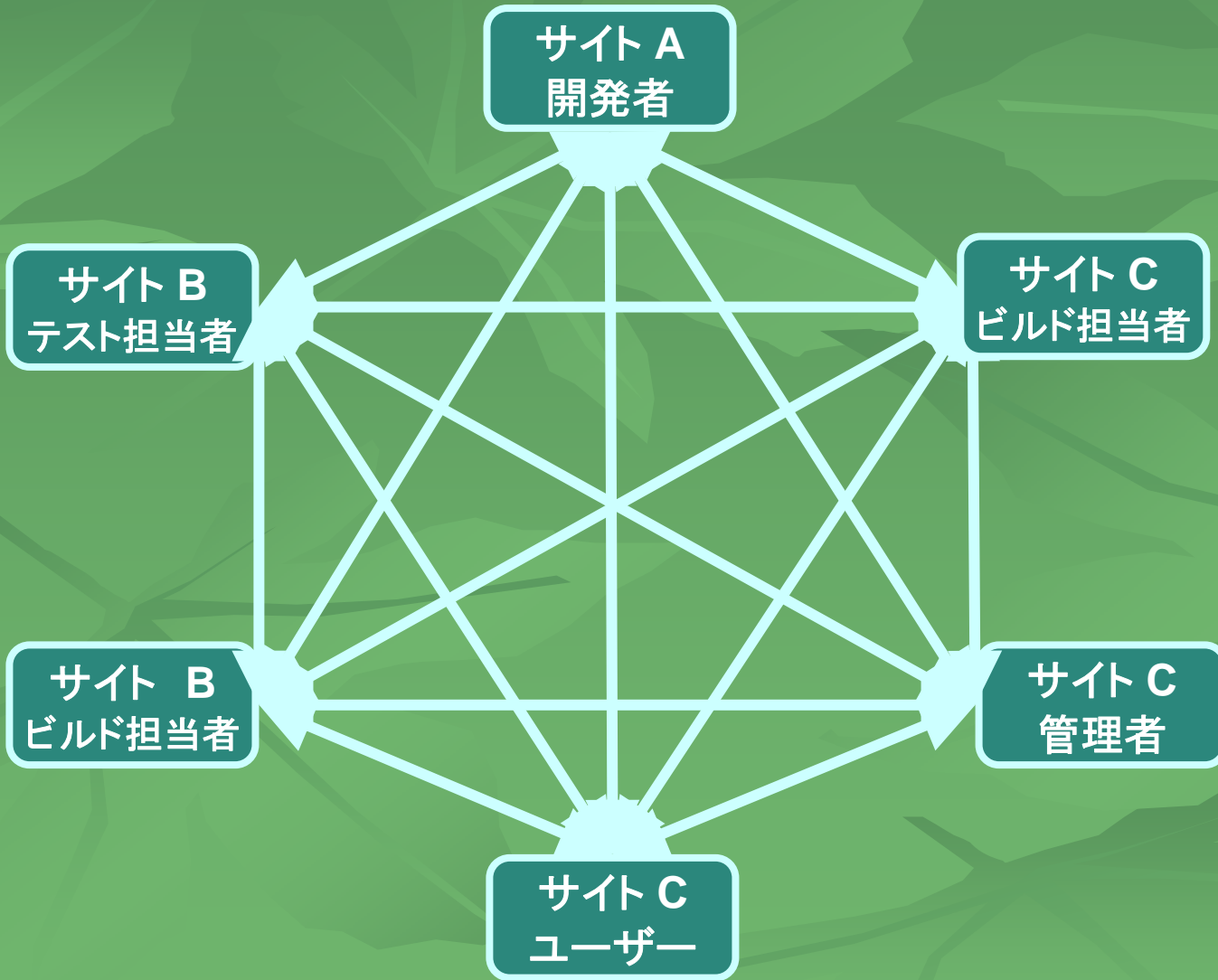


- ・ 日本の受注ソフトウェア業界の一つの特徴である、元請け・下請け構造では分散開発・テストにおいても生産性向上が困難。
- ・ コアサイト・サブサイトは元請け・下請けではない。
- ・ 作業工程(プロセス)、役割、作業範囲、作業開始・完了基準等の標準化が重要。

参照先: 情報処理推進機構 第29回 「情報処理産業経営実態調査報告書」

<http://www.ipa.go.jp/software/hosyo/kankobutu.html>

# 複雑化するコミュニケーション



共通の関心事の  
コミュニティー

- テスト手法
- 負荷テストツール
- 要件定義
- 組み込み開発  
など

# グローバル分散開発の成功要因

作業の曖昧さ排除

標準化

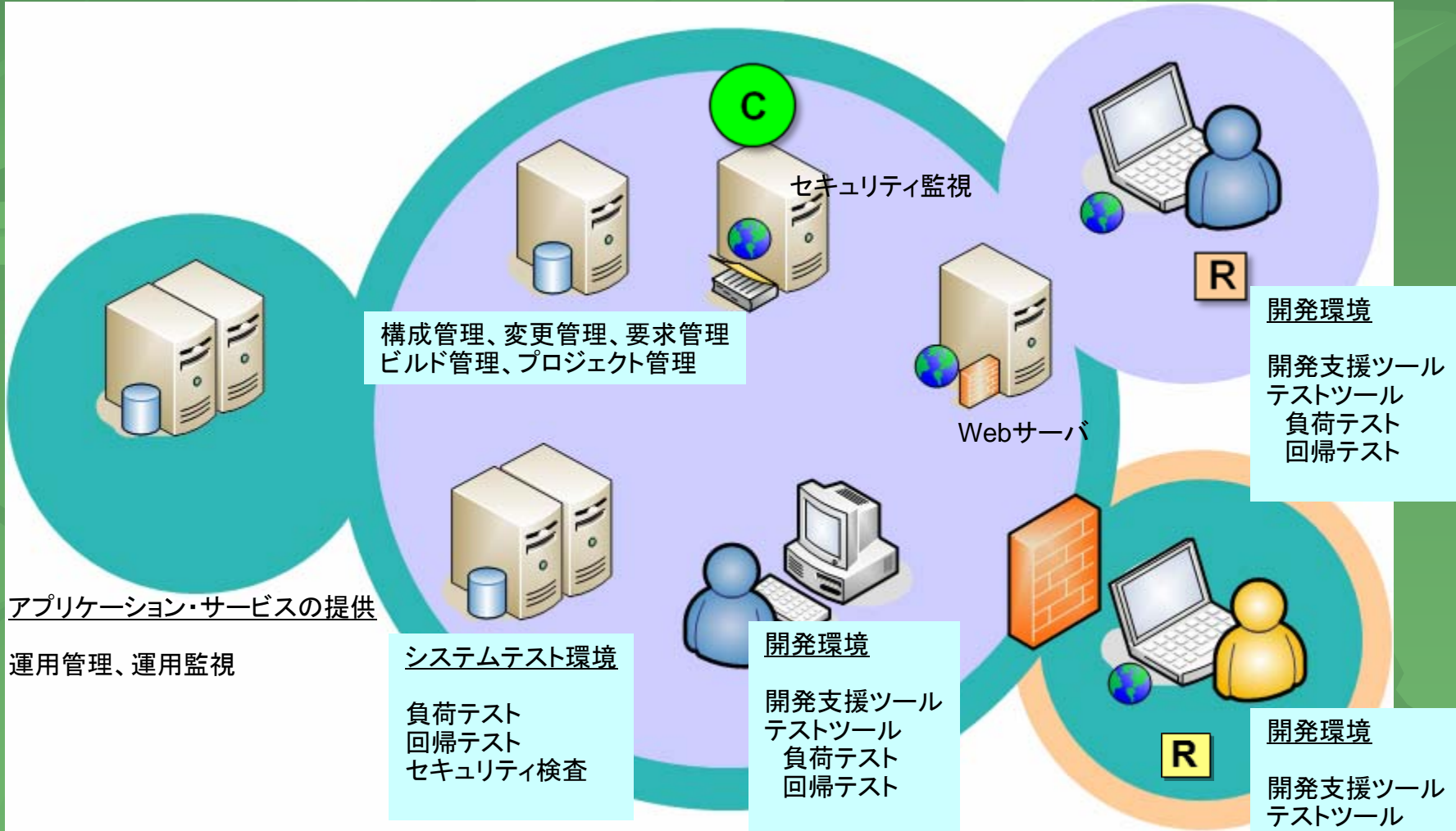
コミュニケーション

ツール活用

リスク管理

ガバナンス

# 品質の見える化のインフラ



## ・ ビルド - テスト - デプロイのインフラ構築

# メトリックス(測定)

## ライフサイクル全般でガバナンスの確立

製品品質と進捗状況を把握

テスト作業  
保守作業

- 1、障害管理
- 2、予算管理

進捗  
(ビルド作業)  
の安定性

製品(アプリケーション)品質  
の見える化

開発状況  
の見える化

# 自動化と再利用

## ■ 複雑なテスト環境の管理、効率的なテスト

開発チーム

リリース管理チーム



開発者



ビルド担当者



実装担当者



テスト担当者



プロジェクト  
マネージャー



リリース  
マネージャー



実装担当者  
(本番環境)

実装

ビルド

自動化

システムテスト

個人の経験に依存しない



正しいビルドか？

プロビジョニング  
と評価

アセットの  
再利用

承認

承認

電子認証

障害

稼動

電子認証

障害・新機能番号、重要度・優先度

- 分散開発の課題
- 分散開発のためのテスト環境
  - テストツール
  - 変更とリリース管理 – 分散開発のインフラ
- まとめ

# ツールの活用 – テストツール

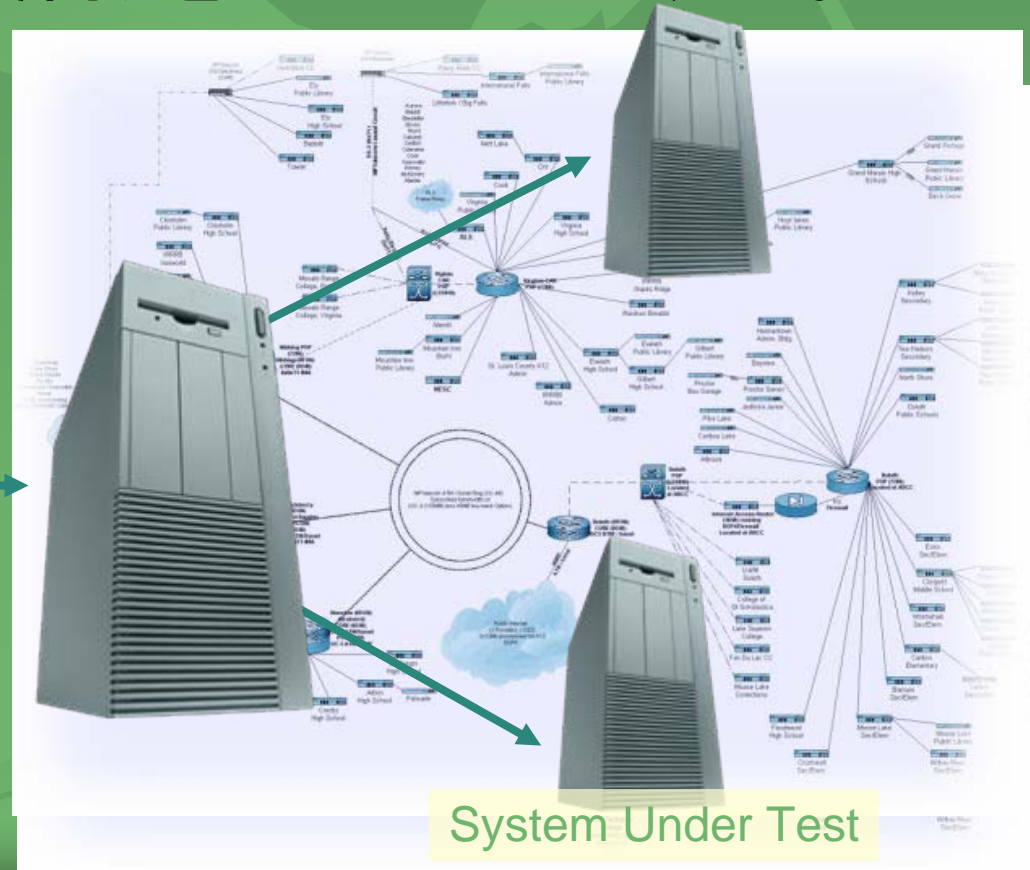
- 負荷テストツール
  - IBM Rational Performance Tester
    - Web, SAP, Cirix, Siebelに対応
- GUI機能テストツール(回帰テストの自動化)
  - Rational Functional Tester
    - Web, Java, .NET, SAP, Siebel, 3270/5250 Terminalに対応
- Webアプリケーションのセキュリティー検査ツール  
IBM Rational AppScan

# 負荷テストツール

## IBM Rational Performance Tester

- システムボトルネックの発見を目的とし、実際のユーザー操作(負荷)をエミュレートする。

コントローラ



# 負荷テストの作成 実行と分析

- 数台のPCから実行
  - 負荷試験の実行時に応答時間のモニターを行う
- 分析のポイント
  - 遅いページを見つける
  - 応答時間明細によりボトルネックを見つける
- 多彩なレポートの作成
  - 時間によるフィルタリング
  - リソースとパフォーマンス情報の重ね合わせ

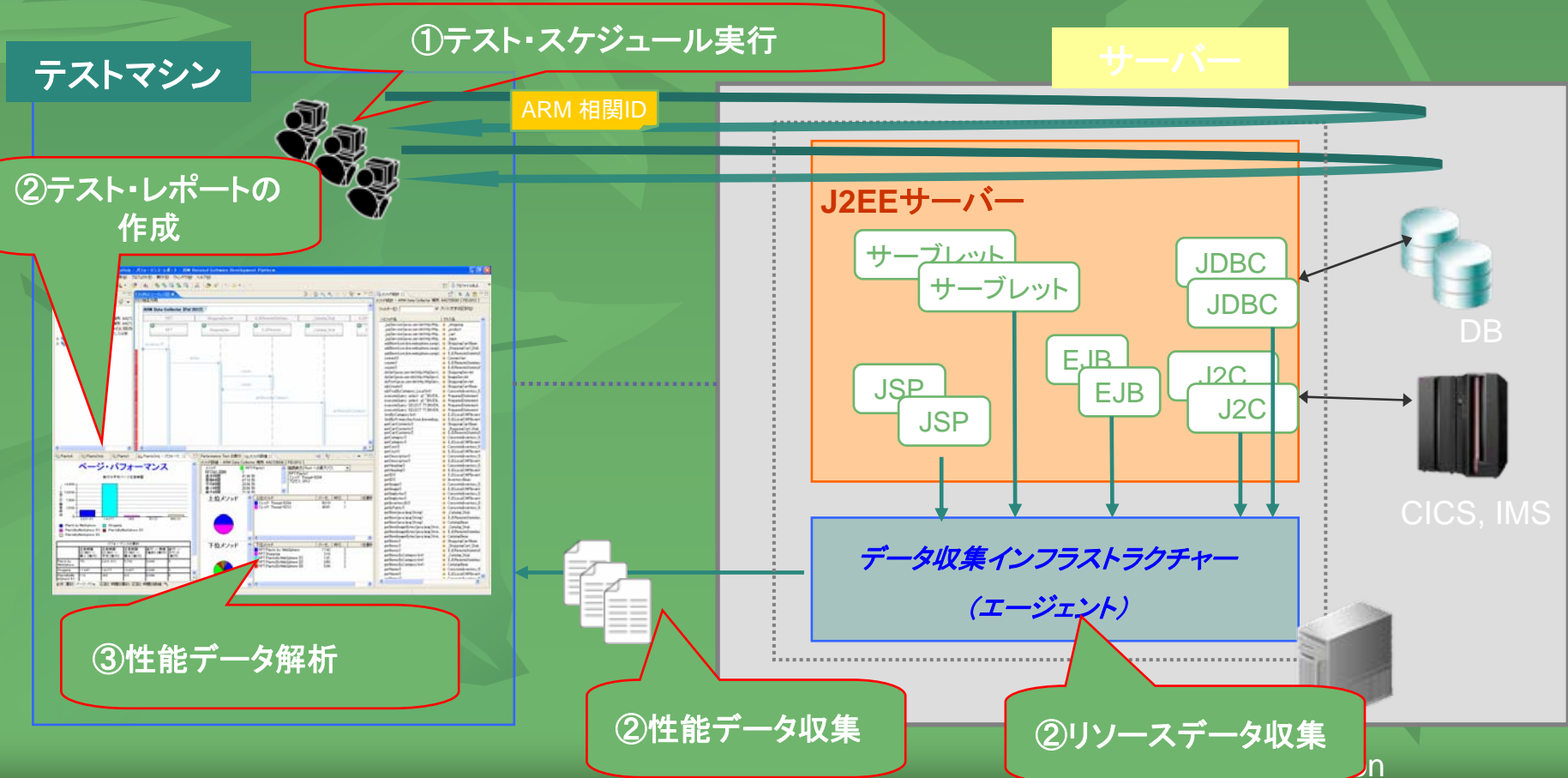


(参考) 標準レポート

- パフォーマンス・レポート
- ページエレメント・レポート
- 百分位数レポート
- トランザクション・レポート
- 検査ポイント・レポート

# 根本原因分析機能とは

- 負荷テスト実施時にJ2EEサーバー側のパフォーマンスデータを収集し、パフォーマンスのボトルネックとなっている箇所を特定をサポートする機能



# IBM Rational Functional Tester

## 自動テストツール



- Web, JAVA, .Net アプリケーションに対する回帰テストの自動化
- 主な特徴
  - マウス、キー操作を記録、自動再生
  - アプリケーションの画面変更に柔軟に対応

# スクリプト記録

## 動的データの検査

**ClassicsCD.com** CD Order Placed

**Catalog** **Go**    Your order has been placed. For future reference, your order ID is 230 at ClassicsCD.com!

**Shopping Cart** **Go**

**Cashier** **Go**

**Order Status** **Go**

Order ID は発注処理が行われるごとに採番される

###

###

###

###

###

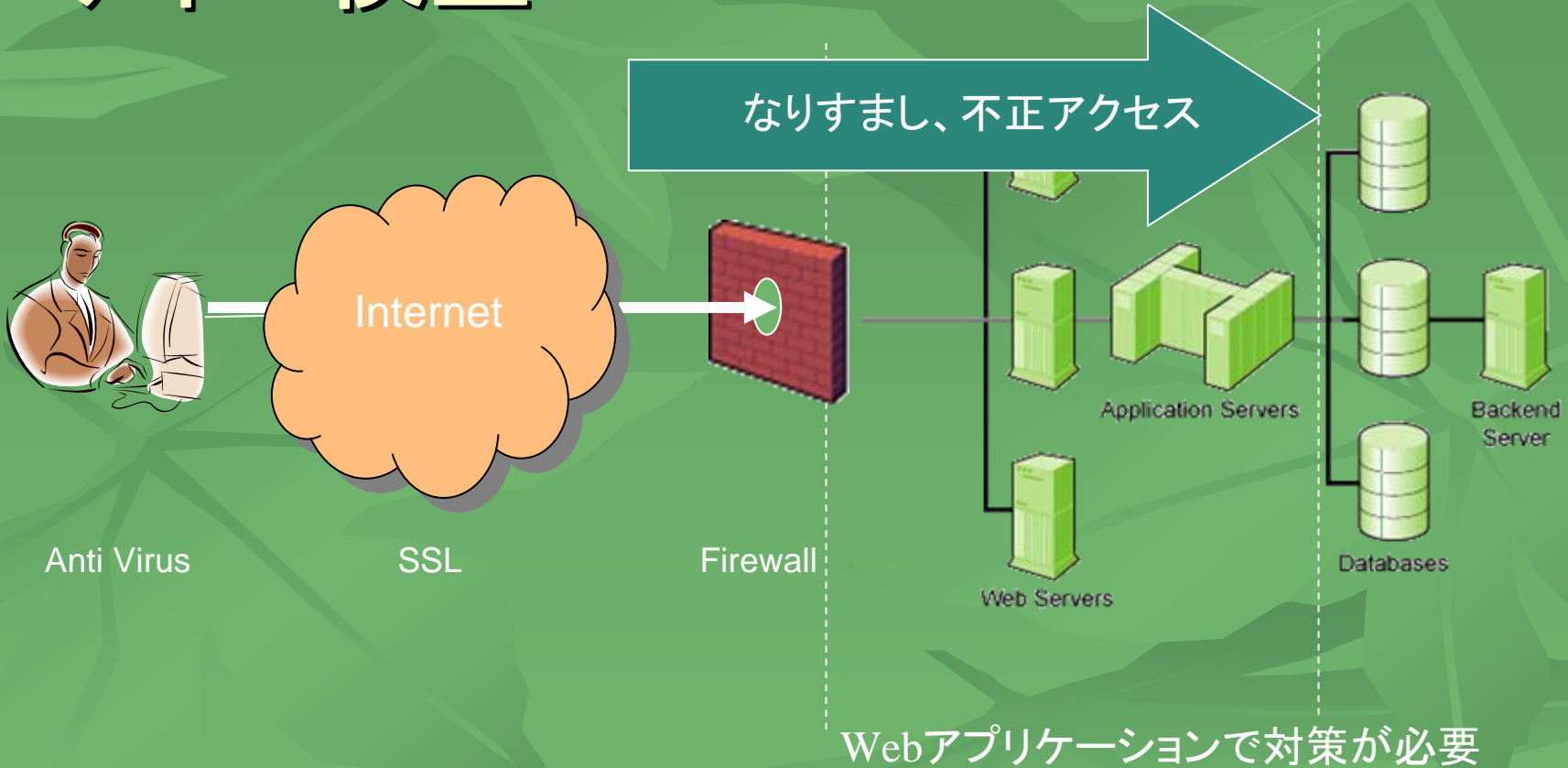
###

動的データをパターン・ベースラインと比較することで検査する

- パターン・マッチングにより、動的データを検査し、広範なテストが可能
  - 例：“注文番号 230”,を検査する代わりに“注文番号 ###” などとする

# IBM Rational AppScan

## Web アプリケーションセキュリティ ティ-検査



Anti Virus

SSL

Firewall

Application Servers

Web Servers

Databases

Backend Server

# Web アプリケーションに対する攻撃

- パラメータの改竄 (Parameter Tampering)
- Hiddenフィールドの不正操作 (Hidden Field Manipulation)
- クッキーの濫用 (Cookie Poisoning)
- バッファオーバーフロー (Buffer Overflow)
- 強制ブラウジング (Forceful Browsing)
- ステルスコマンド (Stealth Command)
- SQLインジェクション (SQL Injection)
- クロスサイトスクリプティング (Cross-Site Scripting – XSS)
- バックドア(デバッグオプションなど) (Debug Option and Backdoor)
- 設定ミスや既知の脆弱性を利用 (Know Vulnerability)

# IBM Rational AppScan

- Webアプリケーション セキュリティテスト ツール
  - Webアプリケーションの脆弱性と、インフラ(OS、Webサーバー等)の設定ミス、既知の問題を検知
  - テストを自動化し、手作業に比べて圧倒的な時間とコストの削減が可能
  - 脆弱性の指摘、修正方法の提示、レポートの作成
  - ブラックボックス テスト
    - ハッカーの視点でセキュリティ問題を検査 - "Hacker in the box"
    - "現実的な" 問題点の指摘 - "Low hanging fruits"

- 分散開発の課題
- 分散開発のためのテスト環境
  - テストツール
  - 変更とリリース管理 – 分散開発のインフラ
- まとめ

# 変更とリリース管理 – 分散開発のインフラ

## IBM Rational ClearCase

- ソースコード変更の記録
- 変更後のビルド、リリース作業の自動化

自動化、統合化、モジュール化

## IBM Rational ClearQuest

- 現状確認、可視化、メンバーの連携
- リリース毎の障害情報

## IBM Rational Build Forge

- ビルドその理由の関連付け
- 一連のビルド作業の自動化

- 分散開発の課題
- 分散開発のためのテスト環境
  - テストツール
  - 変更とリリース管理 – 分散開発のインフラ
- まとめ

# まとめ

- 役割を整理し、どこで何をするのかを明確にする。(標準化)
- 要求(要件)からテストまで、作業状況を見える化し、品質をトレースする。
- テストの自動化ツールを活用する。
- 分散開発のプロジェクト・インフラを構築する。
- 反復的に分散開発を拡充する。

THANK YOU

The text "THANK YOU" is rendered in large, light blue, 3D block letters. Each letter serves as a frame for a different portrait of a person. The 'T' shows a man in a white shirt and orange tie. The 'H' shows a woman with dark hair. The first 'A' shows a man with a textured, metallic-looking skin. The 'N' shows a woman with dark hair. The 'K' shows a man with glasses. The 'Y' shows a man in a light blue shirt. The 'O' shows a man in a white shirt and orange tie. The 'U' shows a woman with dark hair.