アプリケーション・ライフサイクルにおける「パフォーマンス」「セキュリティ」の品質向上アプローチ

2008年9月19日

株式会社サムライズ

SAMURAIZ

株式会社サムライズ　会社紹介

- 2006年7月24日設立
- ㈱アイ・ティ・フロンティア　ソフトウェア事業部の事業を継承
- 事業内容

  ソフトウェア販売事業（国内外ソフトウェア製品のマーケティング及び販売）
  サービス＆サポート事業（ソフトウェア製品の技術支援、保守、教育、各種サービス）
  IT関連の事業開発・事業開発支援（新規商材の発掘・事業戦略の策定支援、ローカライゼーション、業界動向の調査・市場開拓・営業支援・技術支援）

- 資 本 金　：　4,450万円

主要取扱製品

J2EEアプリケーションにおける パフォーマンステスト

技術グループ
プロダクトコンサルタント
富田 誠

SAMURAIZ

Agenda

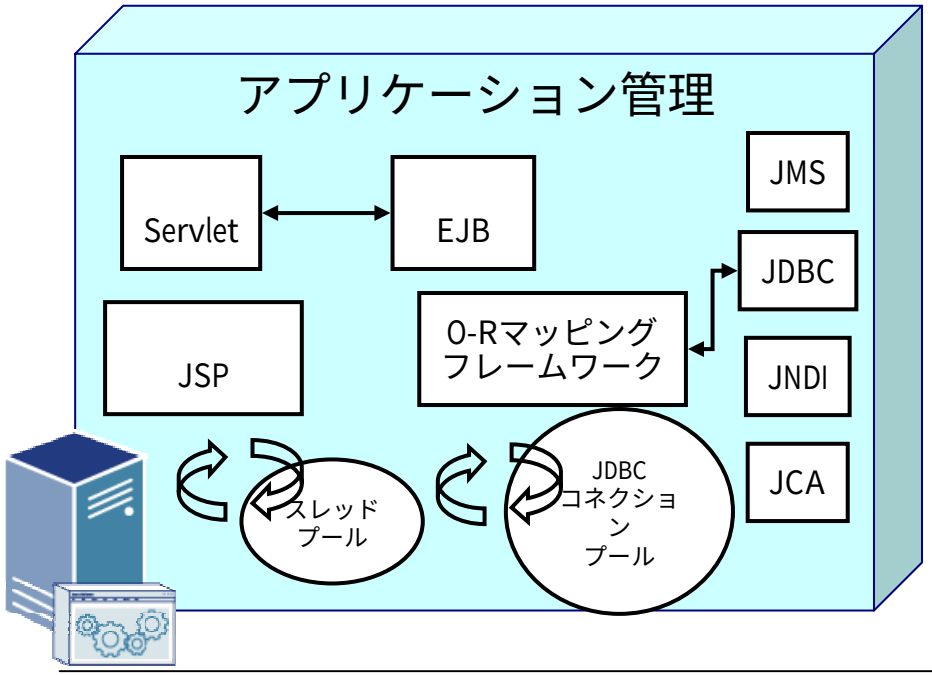
- 性能テストの現状
  - 実施すべき性能関連のテスト
  - 管理レベルの実状と問題
  - テスト工程における性能管理の現状
  - 改善ポイントを探る為に
- お客様事例
  - 概要
  - 課題
  - テスト方針
  - 直面した問題
  - プロダクトの導入
  - 開発環境
  - QA環境
  - ステージング検証環境
  - 本番環境
  - 追加開発
- まとめ

性能テストの現状

実施すべき性能関連のテスト

- 負荷テスト
  - 予想される通常負荷レベルを与えて、性能要件クリアするか検証する
- ストレステスト
  - 予想される高負荷を与えて、問題が発生しないか検証する
- ソークテスト
  - 予想以上の高負荷を与え、データの不整合などの重大事故が発生しないかを検証する
- ランニングテスト
  - 長時間、予想される通常負荷をあたえて、メモリリークなどの問題が発生しないかを検証する

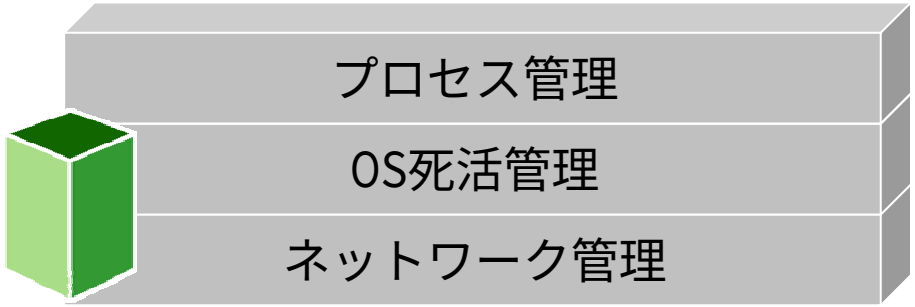
管理レベルの実状と問題



性能問題で原因となるのはアプリケーション管理レベル!見えない情報は管理も出来ない!!

実際のアプリケーションが稼動し、性能問題に対して、本当に管理しないとならないのは『アプリケーション管理』レベル

- アプリケーション(servlet , JSP , EJB , JDBC , etc)の応答速度、スループット、無応答箇所、エラー
- リソース(Heap , コネクション , etc)状況



実状管理できているのは、このレベル

- ps コマンドの実行
- 統合監視ツールの利用
- OS ログ
- ping コマンド

テスト工程における性能管理の現状

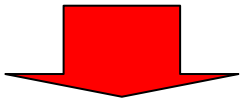
- 単純なテストシナリオ
  - ターンアラウンドタイム計測
  - スループット計測
- 目標値
  - どの処理、シナリオで？
  - ターンアラウンドタイムが、何秒以下
  - スループットは、どれぐらい？

- 診断レベル
  - JMeterなどの負荷ツールを使用して、外部からパフォーマンスを計測している
  - JVM、アプリケーションサーバが提供する、JMXを利用した情報を収集している

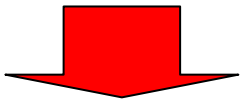
→ パフォーマンスの状況が判っても、どこから改善をすれば良いか良くわからない。

改善ポイントを探る為に

- 機能単位での実行時間が必要である。
- リクエスト毎のメソッドの実行数の解析が必要である。



- メソッドレベルで実行時間を ログに出力する。
- ログを集計して、グラフにする。



- ログ出力の為のプログラム変更が必要となる。
- 集計に時間がかかる為、100%満足できるまで、テストを繰り返すことができない
- 個別に収集した情報とのタイムスケールが合わない為、解析がしにくい。

まともにやると、何しろ大変！人も時間もかなりかかる。

コスト増加

お客様事例

プロジェクト概要

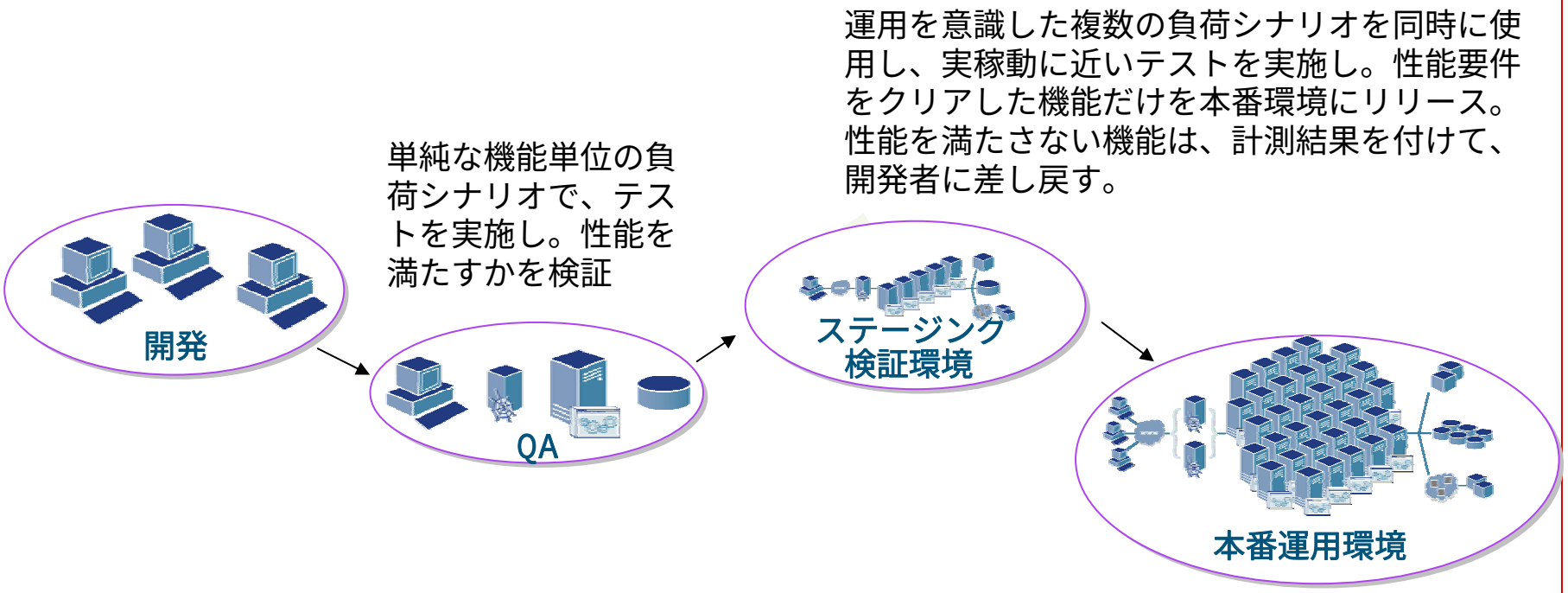
- ホストで稼動しているコボルで構築されて業務システムを、J2EEのアプリケーション上にリプレースする。
- 稼動している機能が多く、すべてを再設計している余裕が無い。
- システムが停止すると、業務全体に支障をきたす。

課題

- パフォーマンス目標値クリアのチェック方法？
- Java VM上に構築されたアプリケーションは、業務毎の稼働状況が監視できるのか?
- アプリケーションサーバに実装されている管理画面では、リアルタイムのアプリケーションサーバリソースしか、監視できない。

テスト方針

- 多重化されていないリクエストで、性能要件をクリアできていないプログラムは、サーバリソースが大きくなっても、簡単には、性能要件をクリアできない。
- パフォーマンスに対して、早い段階から細かく検証を実施することで、手戻りを少なくする。



直面した問題

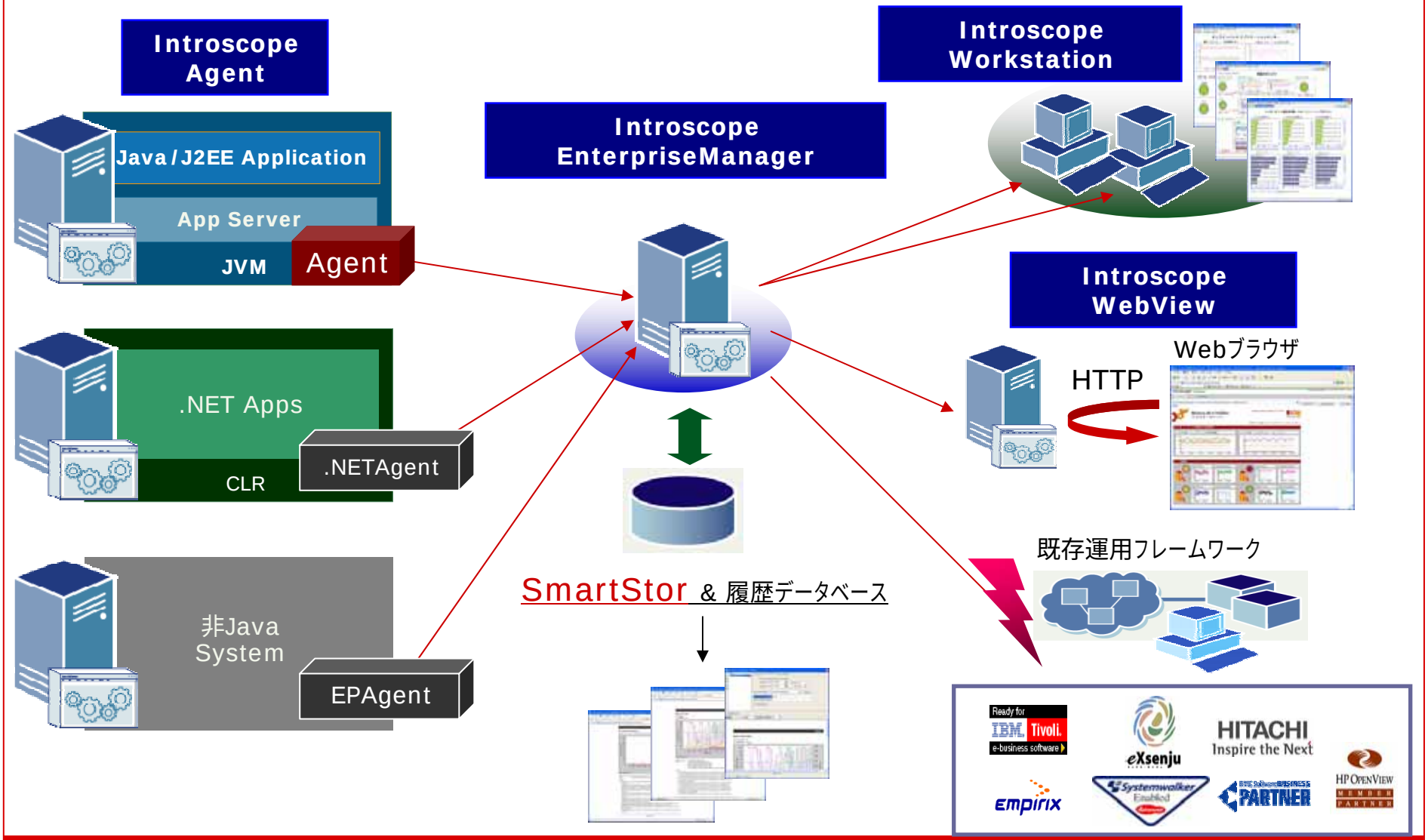
- 最初は、プロファイラーで測定していたが、開発が進み機能が増えるに伴い、アプリケーションの起動に時間がかかるようになり、テストサイクルに支障が出てきた。
- プロファイラーを使用した場合、計測時間に対するペナルティが大きく、実際のレスポンスタイムを測定できない。
- 業務毎の稼動状況など、本番運用を意識したデータ収集が難しい。

プロダクトの導入

- 弊社取扱製品『CA Wily Introscope』を使用することで、問題を解消

- CA Wily Introscopeの主な特徴
  - Java、.Netのソースを変更することなく、メソッドの平均応答時間、最小応答時間、最大応答時間、実行回数が取得できる。
  - 取得したデータを、リアルタイムでグラフ化できる。
  - 取得したデータを蓄積し、過去にさかのぼって、グラフ化できる。
  - 取得したデータをCSV形式で、取り出すことが出来る。
  - 実行時間やリソース状況を監視して、アラートを仕掛けることが出来る。
  - 監視画面が、カスタマイズできる。
  - 低負荷で、24時間365日情報取得が出来る。

Introscope のシステム構成図

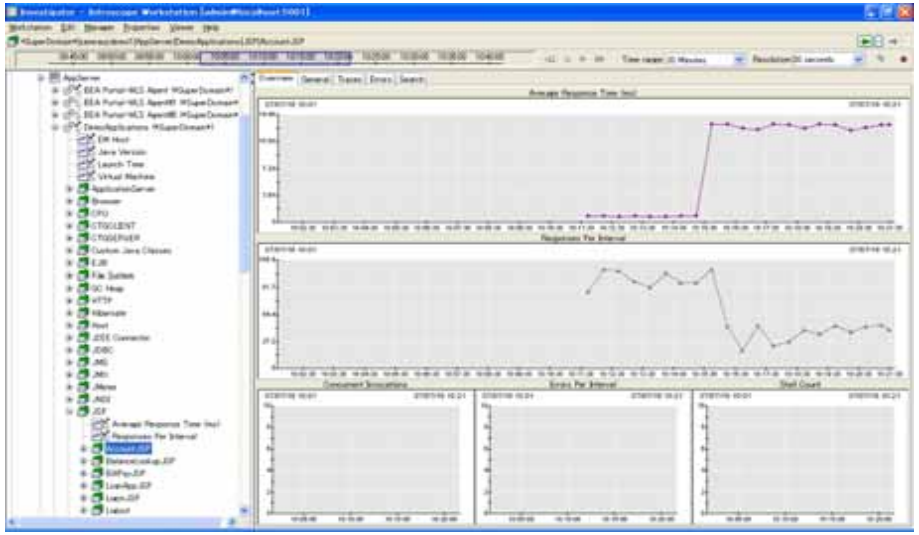


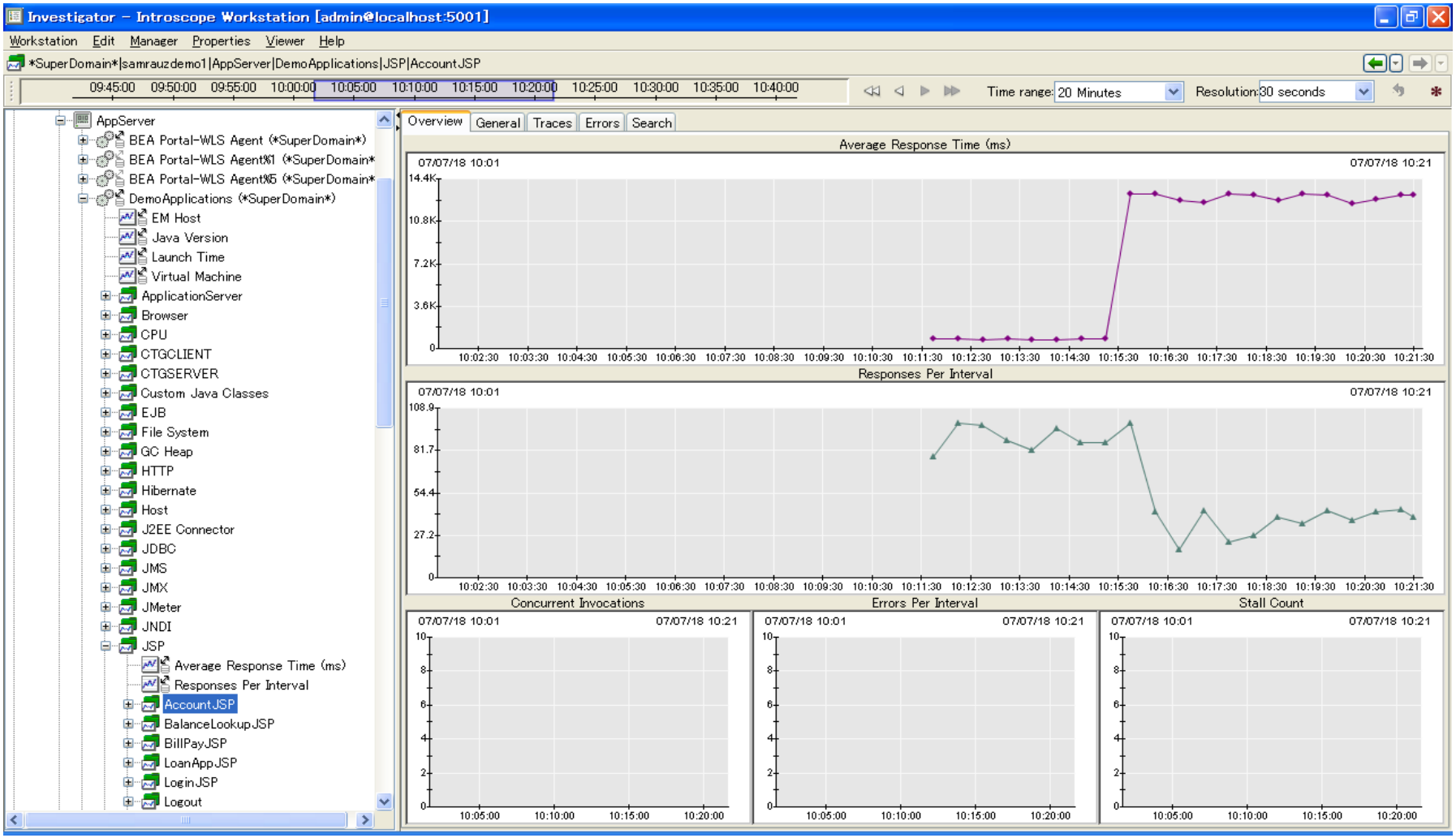
開発環境

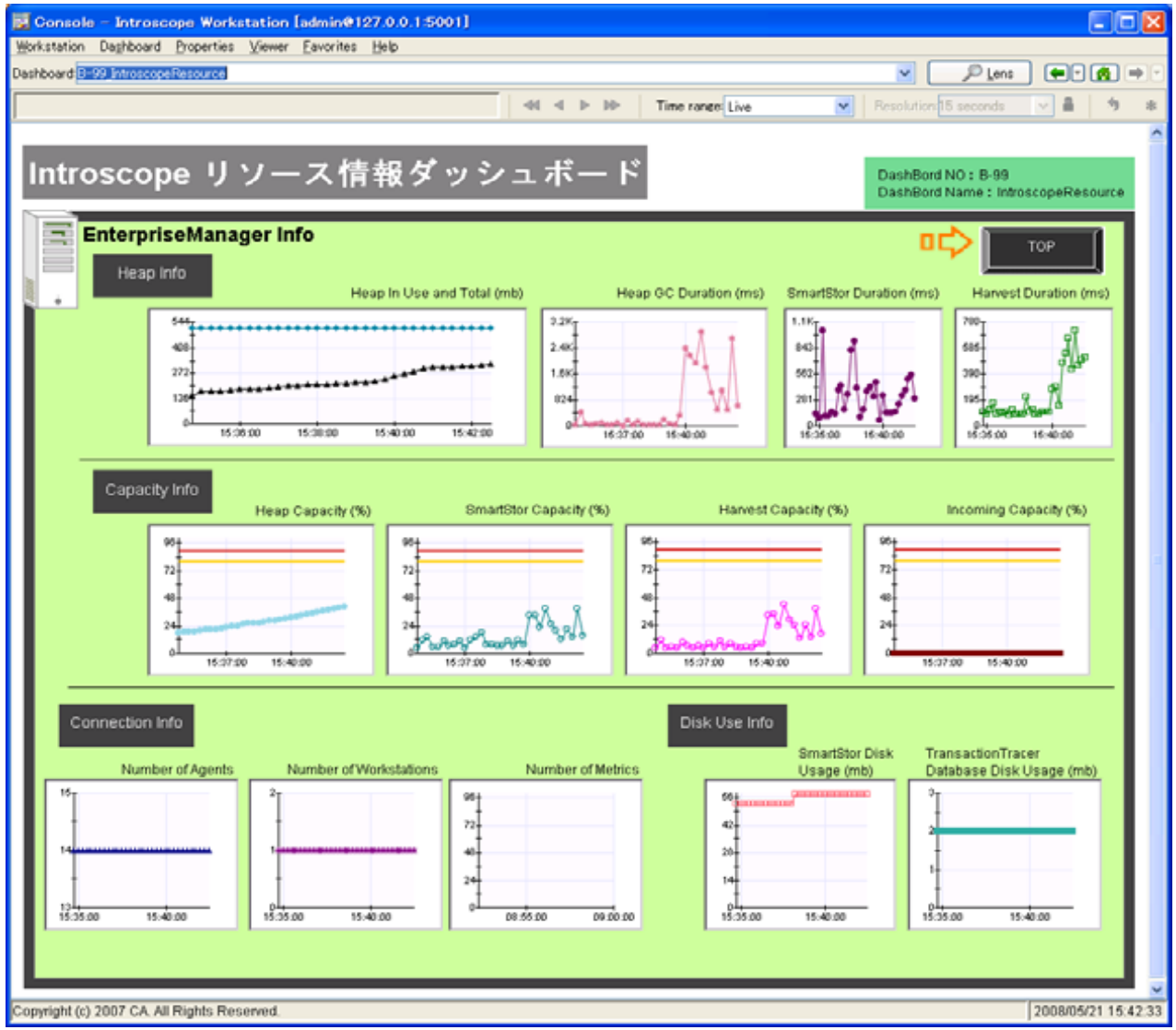
- エンジニアが、開発用環境でログによるタイムスタンプやプロファイラを使用して、単体機能を検証する。
- 最小構成のアプリケーションなので、プロファイラを使用しても起動にかかる時間を抑えられる。
- プロファイラなので、細部まで確認が出来る。
- 性能、機能の目標を満たしたコードを、QA環境へのリリースする。

QA環境

- ツール導入により、パフォーマンスへのペナルティは数パーセントに下がったので、実スピードを計測することが出来る。
- 負荷テスト実施後に、集計をすることなくパフォーマンスの問題箇所を解析をすぐに実施できる。
- 開発者は後から、問題になった箇所とその前後のトランザクションやサーバリソースを含めて確認できる為、JVM内で起きている事象を把握しやすくなる。

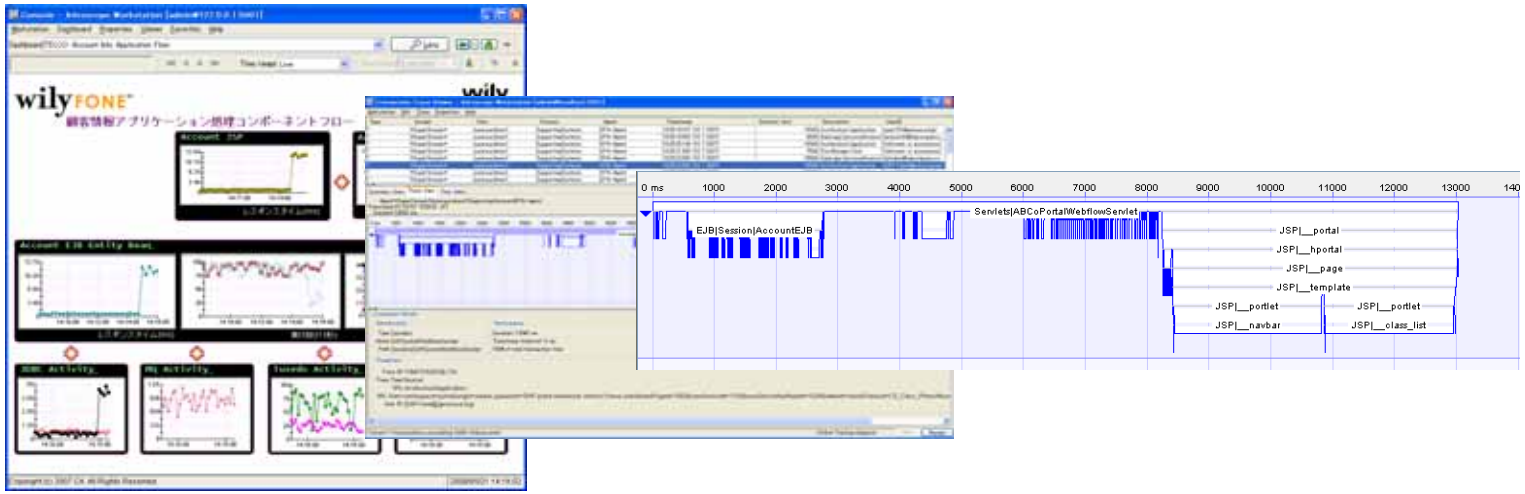


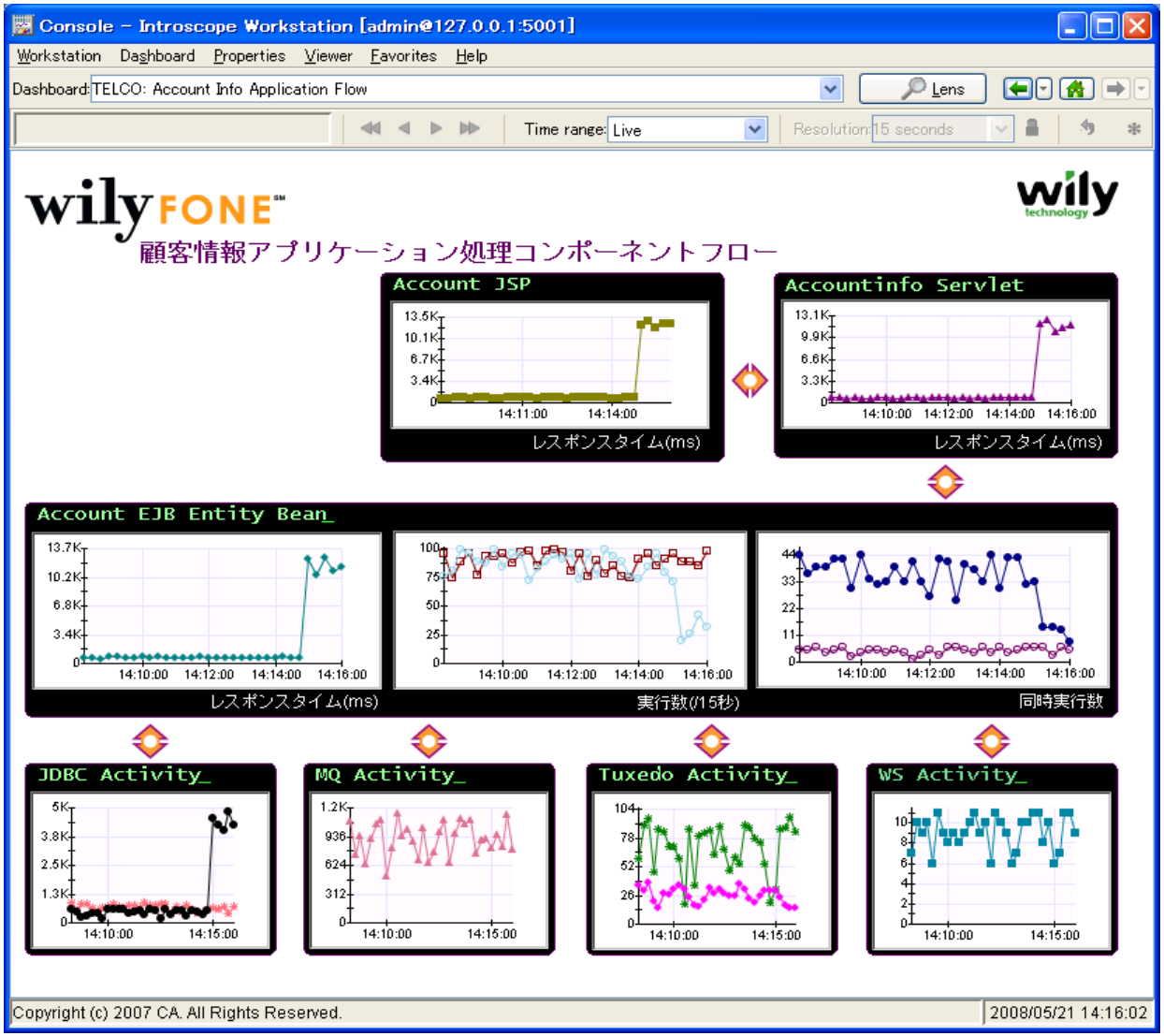


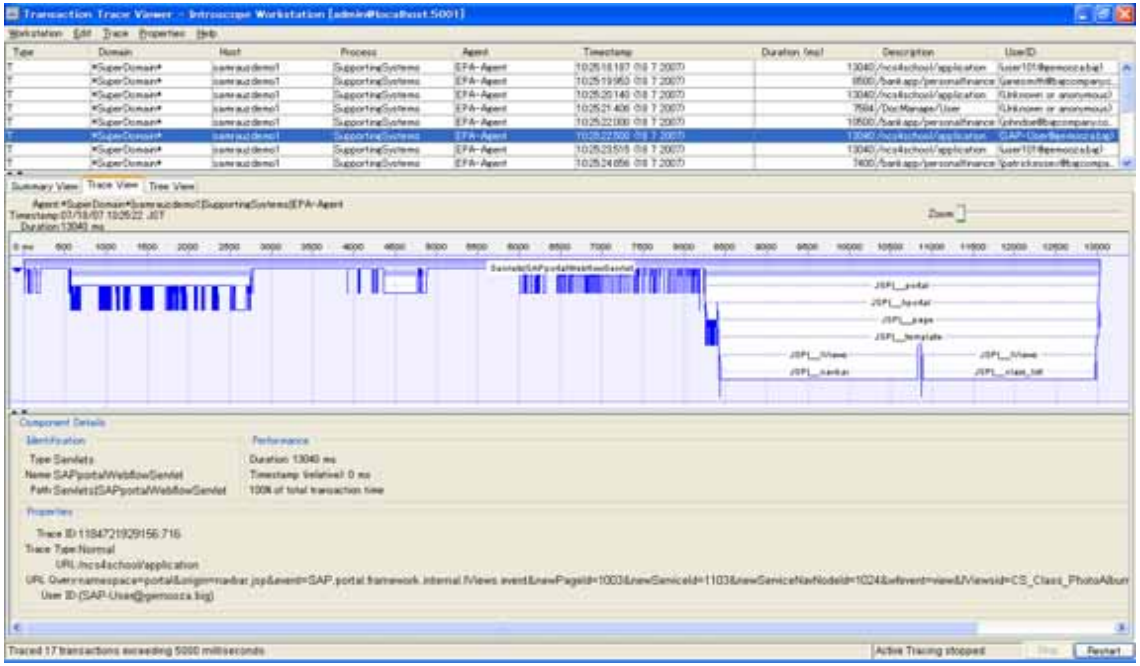


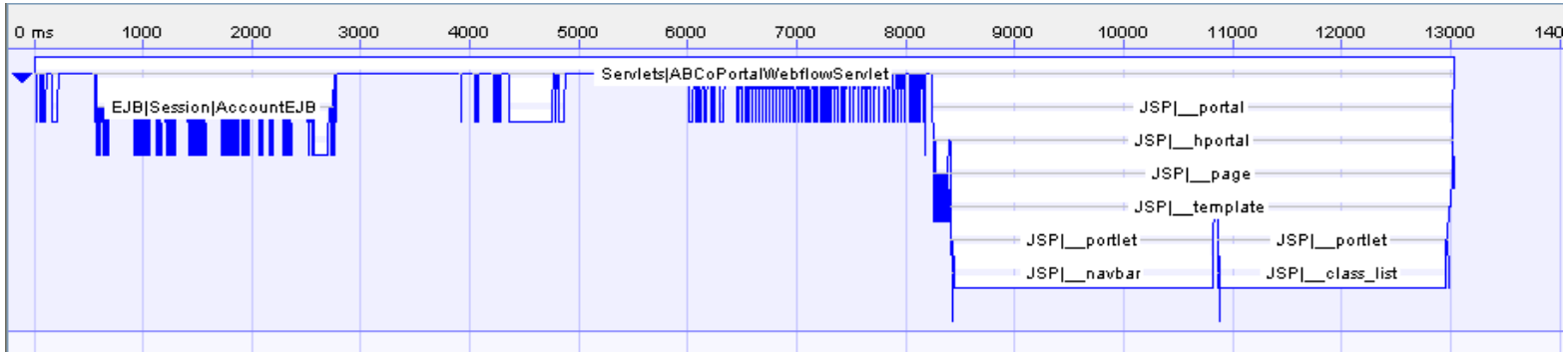
ステージング検証環境

- 本番運用を想定した負荷シナリオでテストが実施が可能。
- 性能要件を満たさないトランザクションを捕まえて、アプリケーションの動きを解析できる。
- 多重化によるサーバーリソースの消費が、アプリケーションの動きに合わせて確認できる為、リソースの増強、環境のチューニング、プログラムのチューニングのいずれの方法で改善するのか、判断がしやすくなる。
- アプリケーションの特性が見える。
- 業務量の増加に対して、レスポンスタイムあるいは処理時間はどのように変化しているか見える。
- 業務量の増加に対して、CPU使用率などリソースはどのように変化しているかが、見える。



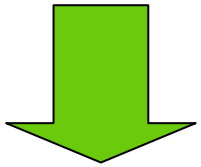




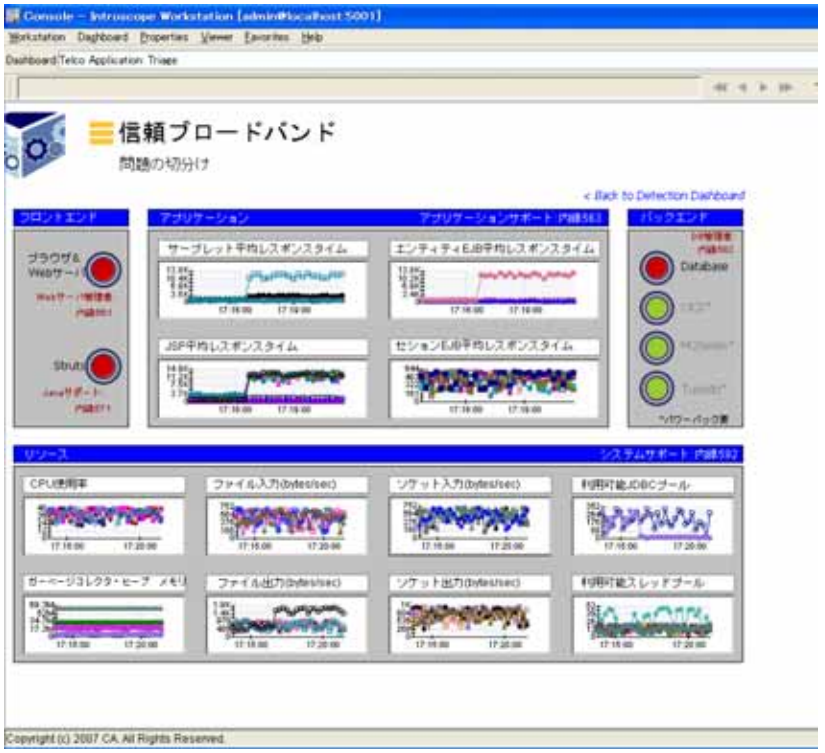


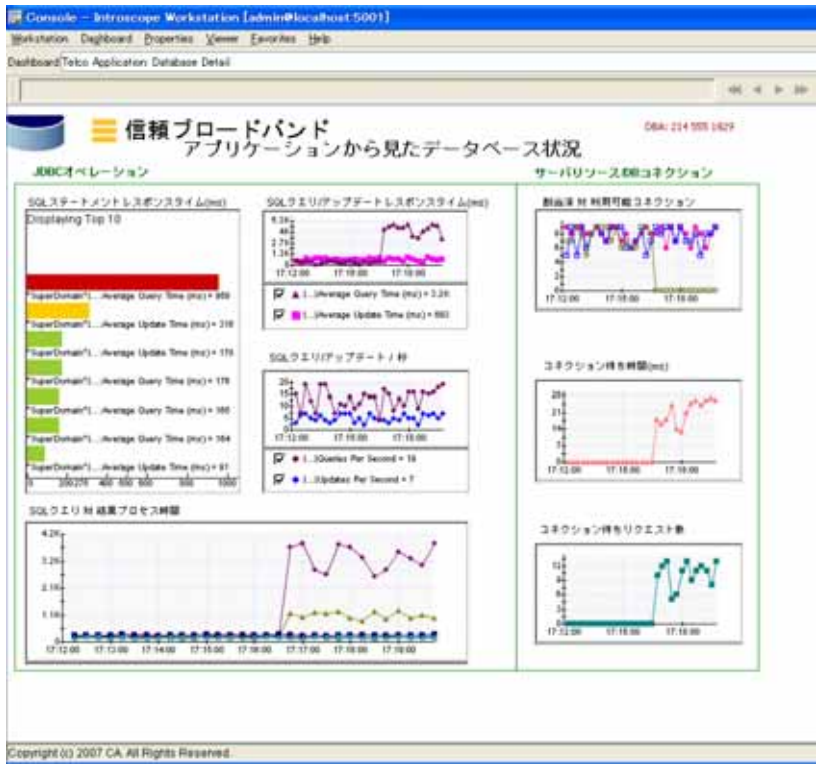
本番環境

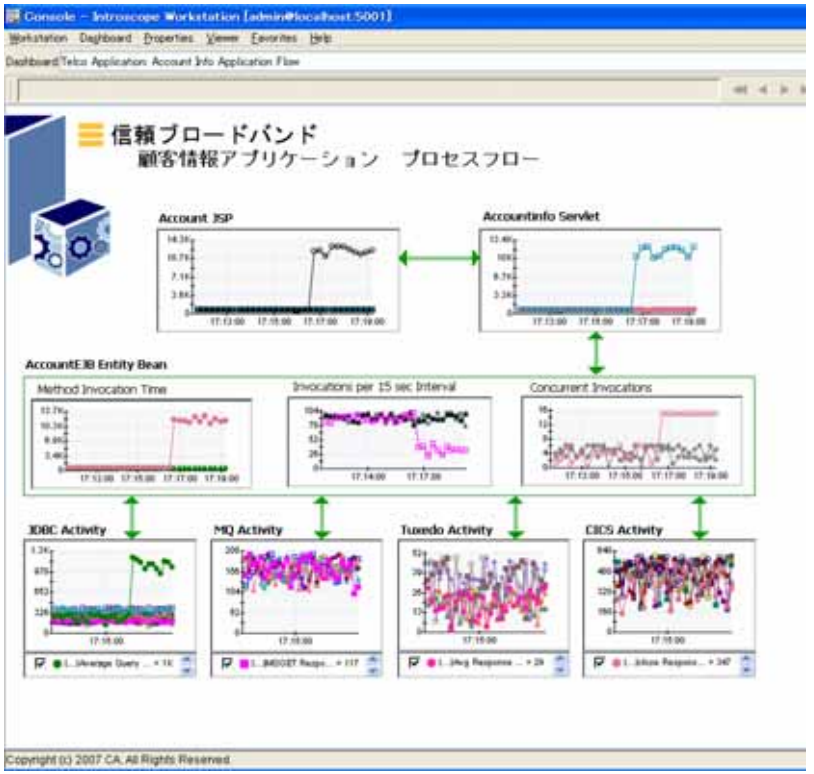
- 監視・解析の為のポイントを絞った監視画面を作成しておくことで、問題発生時の分析、報告の時間が短縮できる。
- 閾値による、アラートを設定して、すでにある監視フレームワークに通知することで、既存の監視プロセスに大きなインパクトを与えずに、導入できる。
- 正常時との比較によって障害の予兆を見つけることが出来る。
- 予兆検知の為のアラートを設定することで、問題がサービスに影響する前に、回避手段の検討が可能になる。
- リソース増強スケジュールへの反映

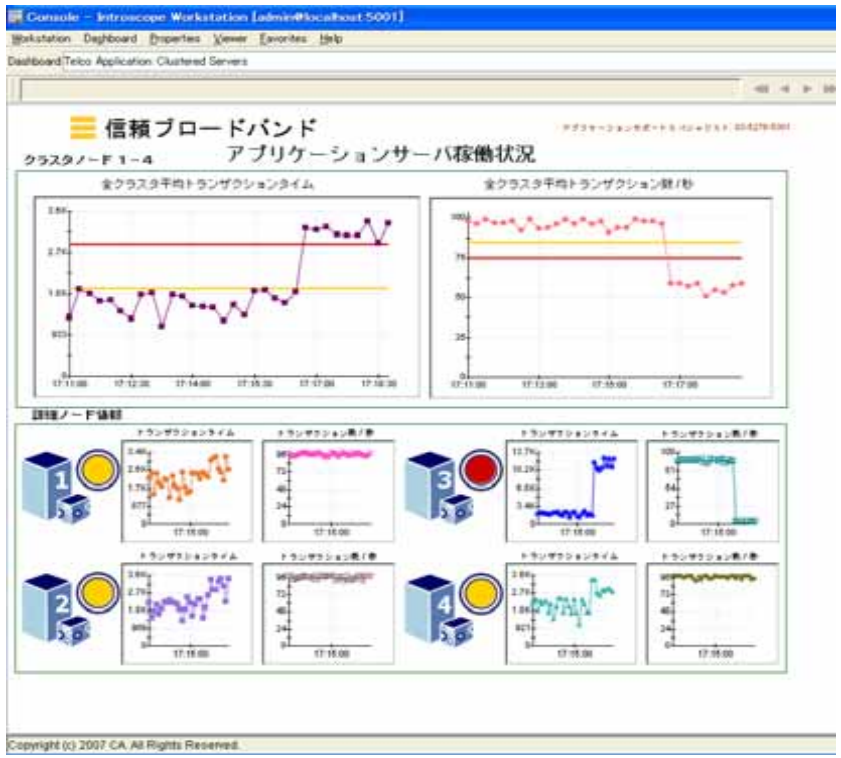


問題がなくても、繰り返すことが大事



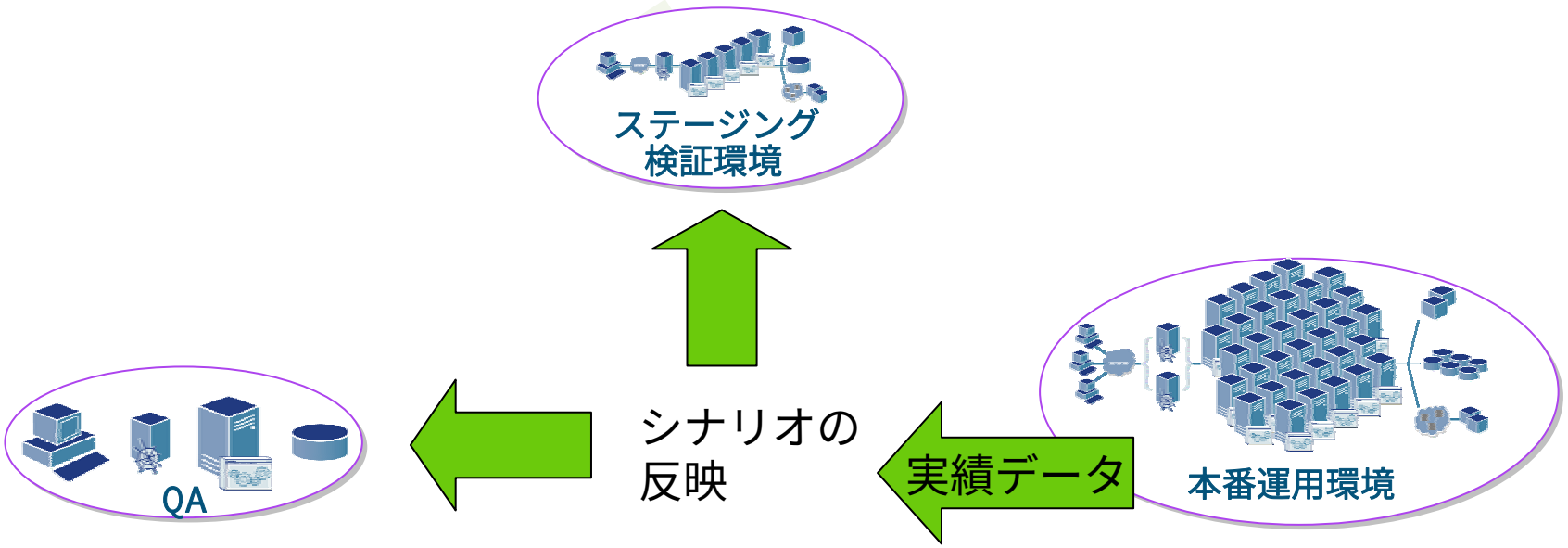






追加開発など

- 開発からリリースまでは、新規開発と同様のプロセスを実施する。
- 本番の稼動状況をステージングの負荷シナリオに、反映したテストを実施。



より、稼動に近いシナリオでテストを実施

- 機能の利用割合、多重度、など

まとめ

- 数値化されることで、結果が誰にでもわかり易くなる。
- 細かく性能評価をすることで、手戻りも少なく、問題箇所も絞りやすい。
- リソースの拡張計画を立てやすくなる。
- アプリケーションの動きの可視化することで、運用に入る前に、運用による対策を立てることが出来る。

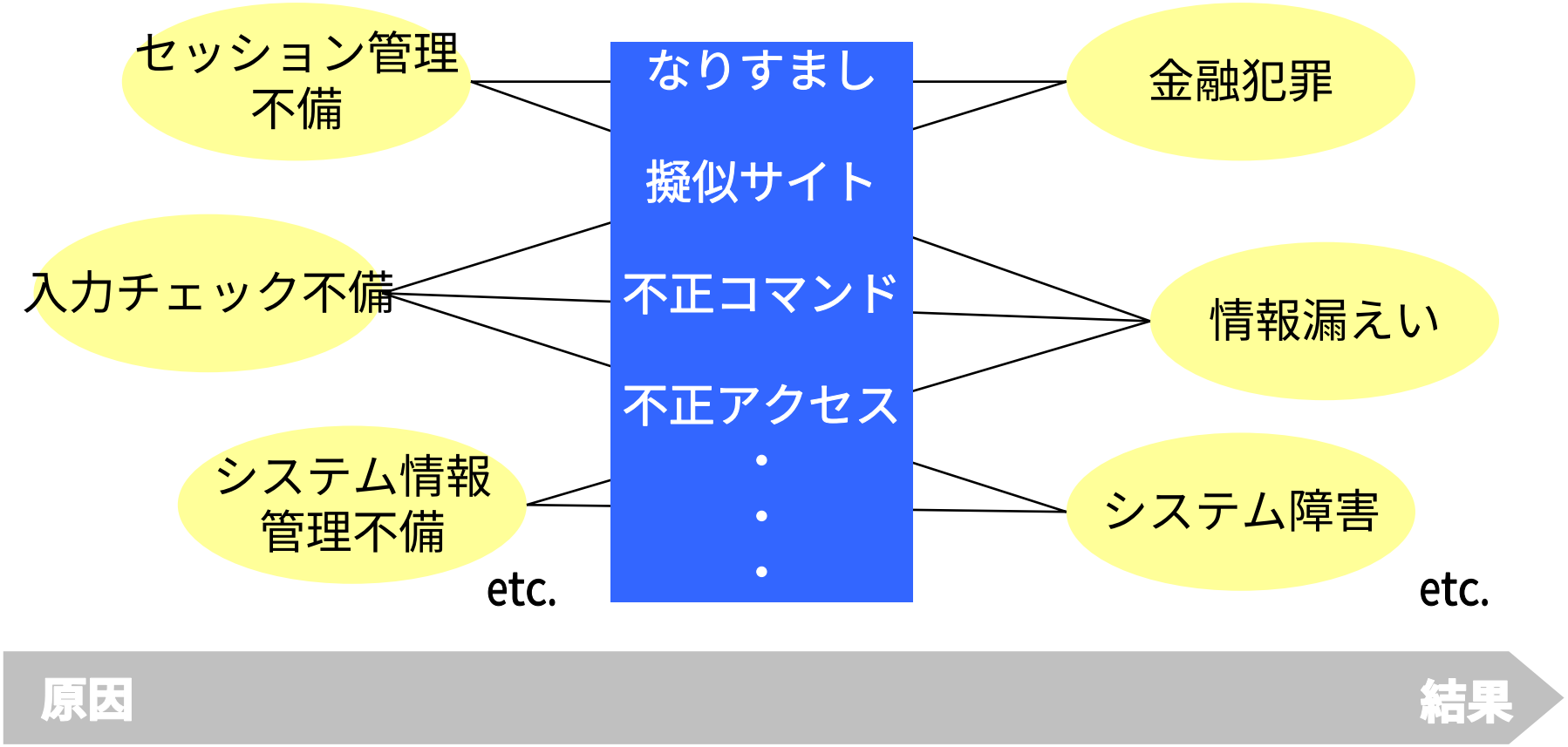
アプリケーションライフサイクルと脆弱性対策

技術グループ
プロダクトコンサルタント
藤井 義隆

SAMURAIZ

背景

アプリケーションの脆弱性を利用した事件・事故は増加の一途を辿っています！



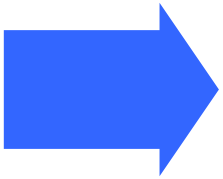
ターゲットは確実にアプリケーション層へ

完全な防御は難しい！

結局サーバにつながらなければシステムは利用できない。

通過可能な全てのパケットを完全に分析・防御するのは難しい。

- ファイアーウォール、パケット監視など運用による水際での防御だけでは充分な効果が見込めない。

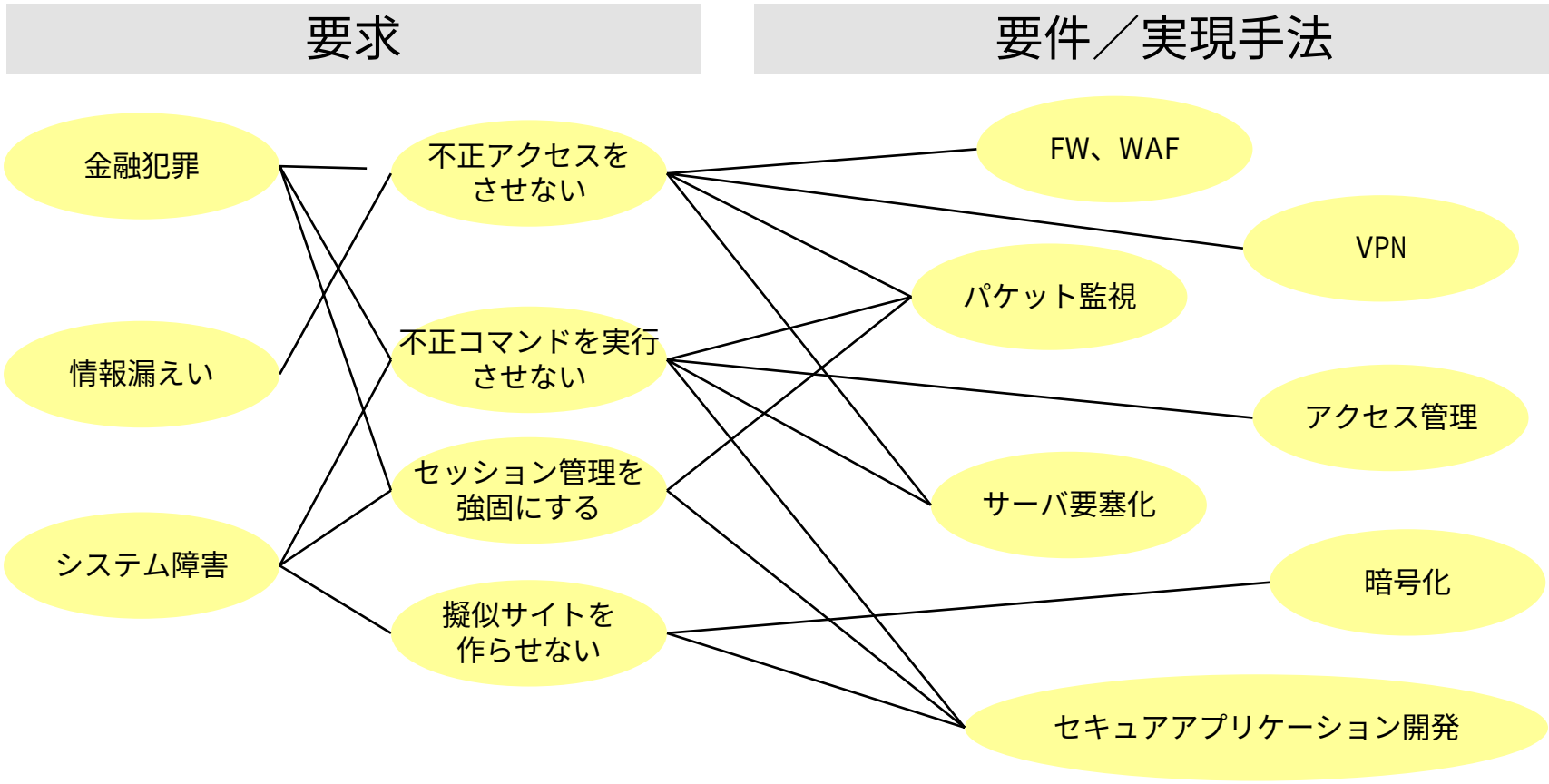


侵入されることを前提とした、

アプリケーション開発やシステム全体の構築が不可欠に！

セキュリティ対策としてすべきことの定義

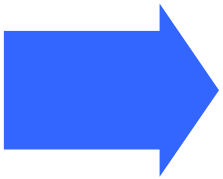
- セキュリティ対策として、セキュアなアプリケーションの構築が必要とされてきている。



セキュアアプリケーション開発

- 要件としての実現項目を明確に定義する　検収基準への組み入れ
- 実現プロセスを計画（工程）に組み込む　標準化
- チェックプロセスを随時設けることで、精度を向上させる。

対策の実施

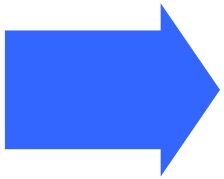


機能要件と同様に計画し、実装していくことが必要です。

しかしながら

ー起こりうる課題ー

- 膨大なソースコード全体ではなく、重要な機能など部分的なチェックになりがち！
- チェックした結果と以降のステータスを管理することが難しい、あるいは負担が大きすぎる！
- プロジェクト内でのセキュリティに関する意識や知識にばらつきがあるためムラがでやすい！
- 侵入テストなどが行われるリリース直前に、修正依頼がくることが多い
  - 手戻り工数が多い！
  - そもそも間に合わない！



物理的な限界とのトレードオフが発生

そこで・・・

セキュアアプリケーション開発実現のために

- 検査項目の定義
- 解析結果の可視化／管理
- セキュリティナレッジの補填

時間、人的資源などへの負担増をツールによって削減

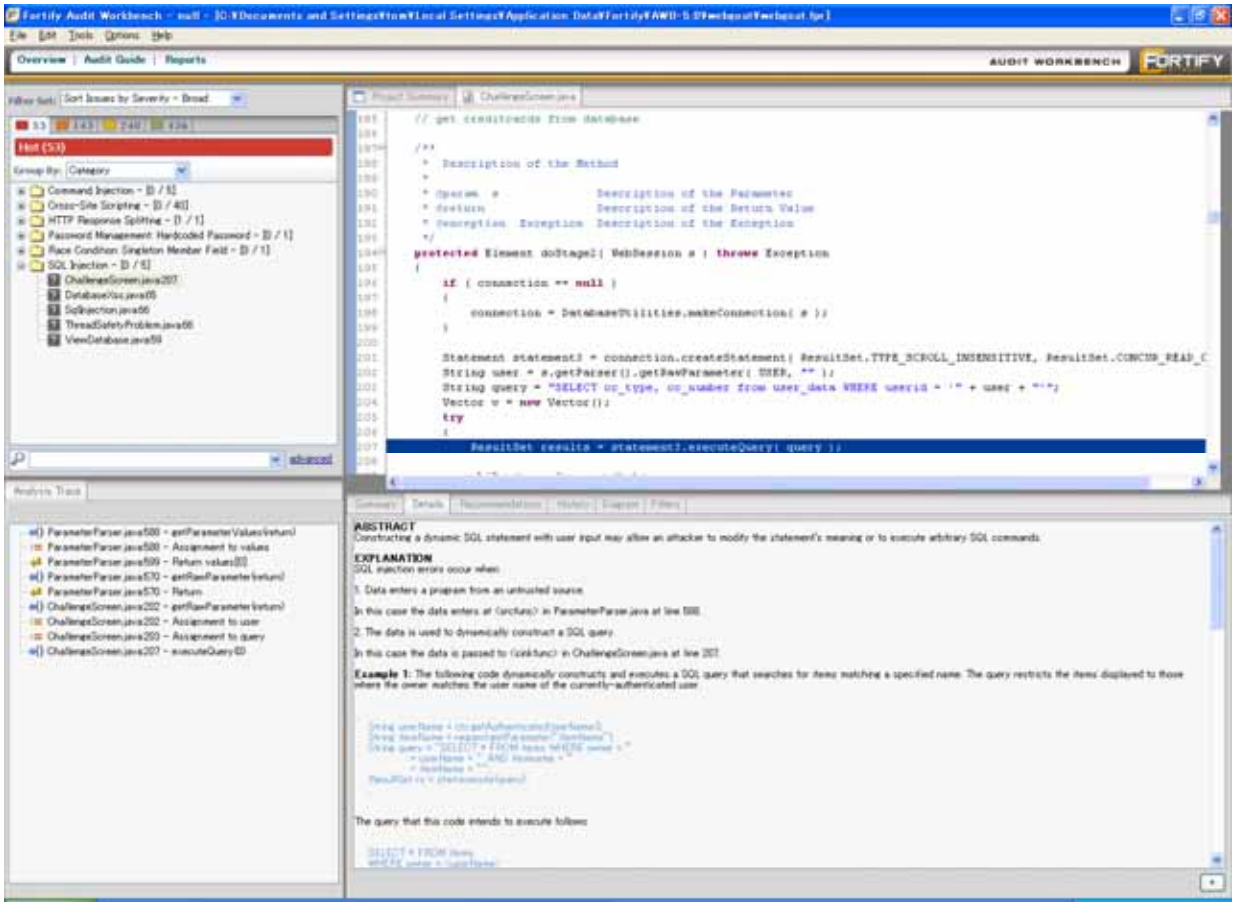
FORTIFY SCA概要

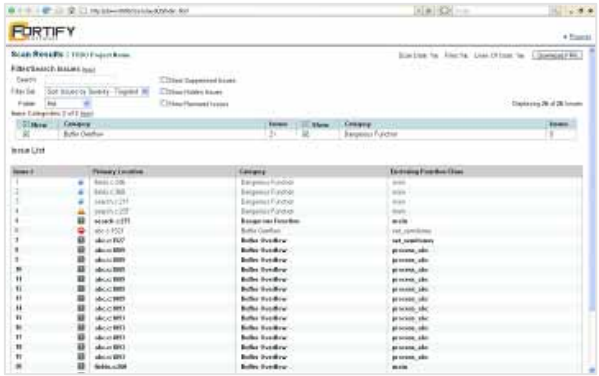
アプリケーションの脆弱性について、セキュリティの観点から、
ソースコードの解析を実施するツールです。

◆ソースコードを分析して脆弱性に関わる問題点を発見・可視化
◆問題の原因、概要、推奨情報の提供
◆最新のセキュリティノウハウを研究、調査し、ＳＣＡに反映
◆対応言語の多さ
　C/C++、JAVA、JSP、C#、VB.NET、ASP.NET、ColdFusion、SQL
　PHP、Java Script
◆対応プラットフォームの多さ
　Windows, Linux, Solaris, HP-UX, AIX, Mac
◆複数の言語が混在したシステムも一括分析
　（例）ＪＳＰ＋ＪＡＶＡ＋ＳＱＬ
◆何百万のソースコード全体を短時間にチェック
◆開発時における手戻り工数の削減

Audit Workbench インターフェース

Audit Workbench





分析結果をGUIにて視覚的、直感的にとらえることができます。

豊富なルール

【SCAが提供するチェックルールの一部】

- SQL Injection
- Cross-Site Scripting
- Buffer Overflows
- Access Control
- Process Control
- No Null Termination
- Setting Manipulation
- Resource Injection
- Password Management
- Unreleased Resource
- Format String Issues
- EJB Resource Permission
- EJB Bad Practices
- J2EE Bad Practices
- Struts Form Field Validation
- Double Memory Free
- Null Pointer Dereference
- Directory Restriction
- Object Model Violation
- Often Misused
- Poor Style
- Access Control
- Insecure Randomness
- Least Privilege Violation
- Code Correctness
- Poor Error Handling
- Dead Code
- J2EE Misconfiguration
- Memory Leak
- Portability Flaw
- Obsolete
- System Information Leak
- Trust Boundary Violation
- ASP.NET Misconfiguration
- Privacy Violations
- Native Callout
- Unsafe Memory Operation
- Unchecked Return Value
- Always Unsafe Functions
- Race Conditions
- Uninitialized Variable
- Session-ID Length
- Entity Bean Configuration
- Information Leakage
- Log Forging
- Integer Overflow

【拡張ルール】

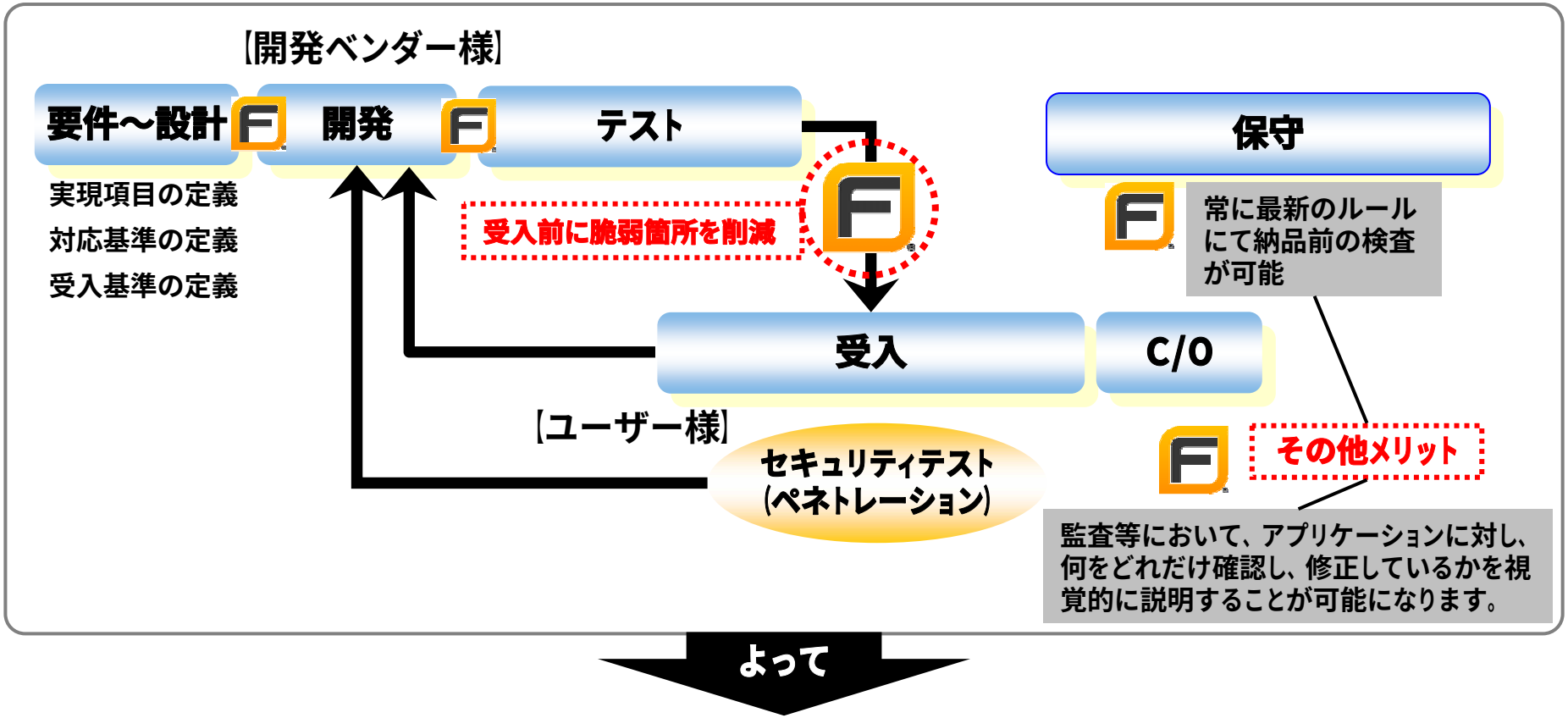
.NET	NLOG and Log4Net、 Microsoft ASP.NET AJAX (Atlas)
C/C++	MFC™ and ATL™ Glib、 Microsoft Windows API、 Pthread、 Sun RPC
Configuration	J2EE and EJB configuration files、 ASP.NET、 and BEA WebLogic™ configuration files
Java	JMS、 JNDI、 J2EE、 Apache Commons、 Log4J、 ORO、 Struts、 ATG Dynamo™、 Hibernate、 Spring、 and iBatis Google Web Toolkit (GWT). Direct Web Remoting (DWR)
JSP	JSTL Core、 JSTL SQL、 and Apache Struts HTML EL
SQL	MOD PLSQL

- OWASPなどからの最新の情報をもとに随時更新（年に数回）

FORTIFYの効果

- 要件定義フェーズにおいて、ソースコードにどのようなチェックをかけて修正していくかを明示することが可能になります。
- プロトタイプ実装から開発フェーズまで、随時、解析＆修正（擬似レビュー）を実施することが可能になります。
- 上記により、ソースコード検収時、リリース直前のセキュリティテストなどで大量の手戻りの発生を削減することが可能になります。
- ルールの更新により、リリース後も、最新の動向にあわせた解析を継続していくことが可能になります。

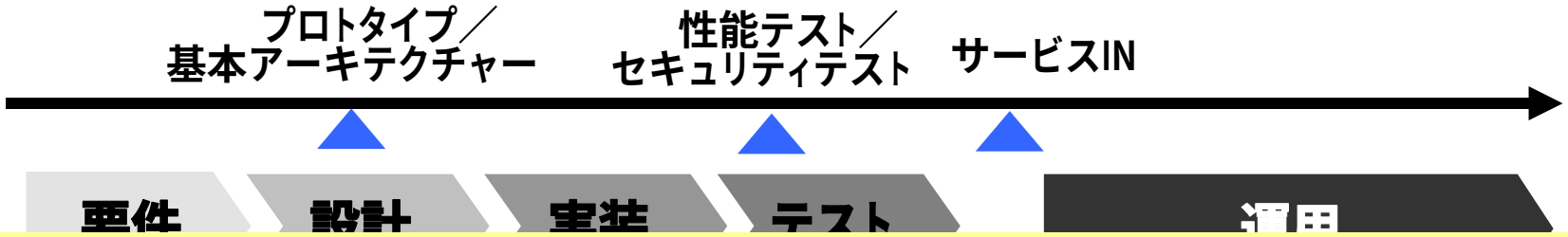
アプリケーションライフサイクルにおける効果



★手戻り工数が、６分の１に減少　★修正時間が、２分の１に減少
★脆弱箇所に関するレビュー工数が、５分の２に減少

（メーカーサンプル顧客での実績値より）

まとめ



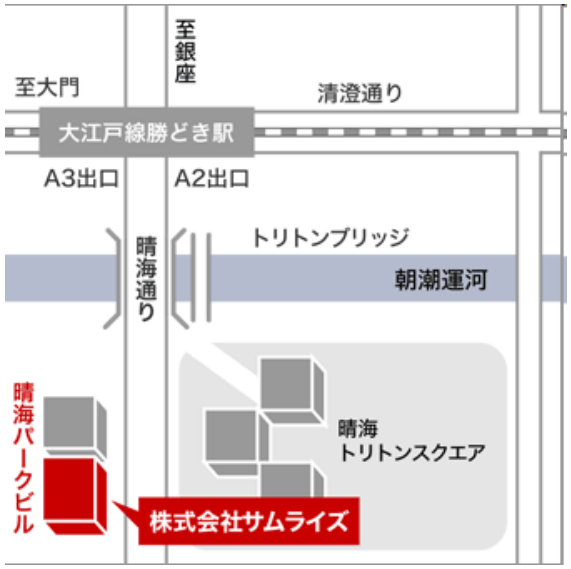
開発期間の短縮に反して、機能要件が複雑化する中、
性能やセキュリティなどの非機能要件への対処は、
プロジェクトにとって非常に大きな負担となっております。

- 機能要件と同様に非機能要件も重要な要素として扱われる必要があります。

「Fortify SCA ハンズオン紹介セミナー 」

本セミナーではFortify SCAを実際にご覧いただき、お客様が自由にご操作いただく事により本製品の最適な利用イメージを体感していただきます。

日時	2008年10月2日（木） 14:00～16:00 2008年11月6日（木） 14:00～16:00 受付（13:30～）
場所	株式会社サムライズ　トレーニングルーム
主催	株式会社サムライズ
費用	無料
定員	若干名

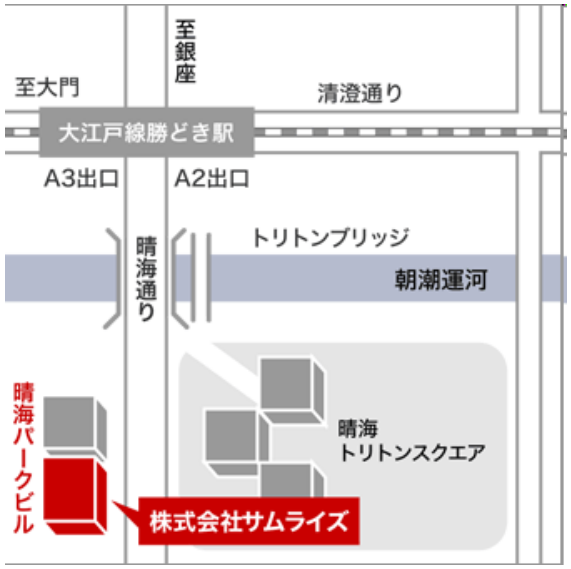


お申込はこちらから
http://www.samuraiz.co.jp/event/index.html

「CA Wily Introscope ハンズオン紹介セミナー 」

本セミナーではCA Wily Introscopeをお客様が自由にご操作いただく事により、本製品による性能管理のイメージを体感していただきます。

日時	2008年10月21日（火） 14:00～17:00 受付（13:30～）
場所	株式会社サムライズ　トレーニングルーム
主催	株式会社サムライズ
費用	無料
定員	若干名



お申込はこちらから
http://www.samuraiz.co.jp/event/index.html

■2008年9月23日より受付開始

私たちSAMURAIZは、
次世代エンタープライズWebの基盤となる
「技術・製品・サービス」を
国内・海外から紹介し、市場の創造、販売、ソリューションの
ご提案までをトータルに提供するために
生まれた企業です。

《お問い合わせ先》

株式会社 サムライズ
http://www.samuraiz.co.jp
〒104-0053　東京都中央区晴海3-2-22　晴海パークビル

TEL ： 03-5548-8820
CA Wily製品関して：introscope_info@samuraiz.co.jp
Fortify製品に関して：fortify@samuraiz.co.jp