–開発の現場から学ぶ–

高信頼なWebシステム基盤構築のポイント

㈱日本総研ソリューションズ
技術本部 インフラ技術部
飯田 博記

日本総研ソリューションズ
JRI Solutions, Limited

会社概要

■名　　称 ： 株式会社日本総研ソリューションズ
　　JRI Solutions , Limited.
■設　　立 ： 2006年7月
■資 本 金 ： 50億円
■従 業 員 ： 1,300名
■株　　主 ： 株式会社日本総合研究所
■代 表 者 ： 代表取締役社長兼最高執行役員　小名木　正也
■所 在 地 ： 【東京本社】 東京都中央区晴海2-5-24　晴海センタービル
　　【大阪本社】 大阪市西区新町1-6-3
　　【支　　社】 名古屋
■U R L ： http://www.jri-sol.co.jp/

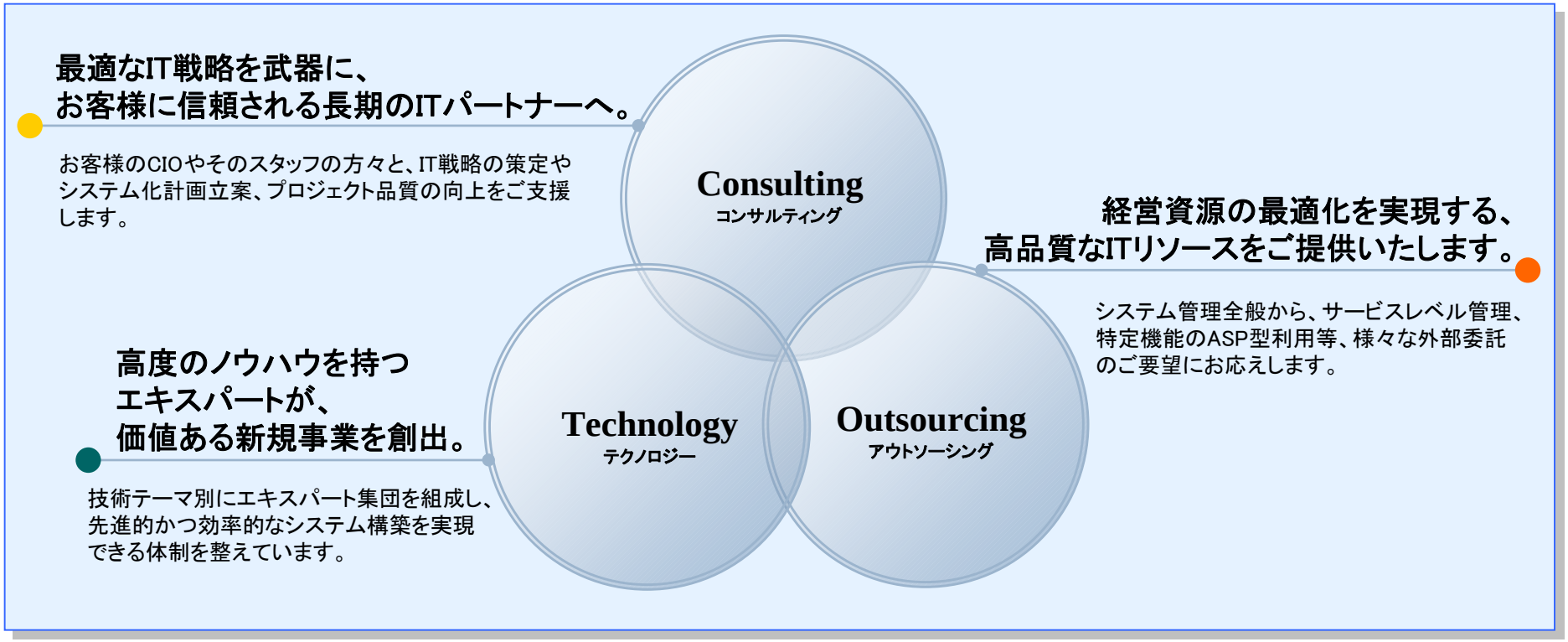
■主要事業所
〇システム開発
　東京：晴海
　大阪：四ツ橋、心斎橋、土佐堀
〇データセンター
　東京、大阪にそれぞれ設置

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

当社の強み

お客様の本質的な問題解決と未来の価値創造を。
私たちは信頼されるプロフェッショナル集団です。

複雑化するお客様のニーズに応えIT投資の効果を最大化する為、社員一人ひとりが、ITコンサルティングからシステム構築・運用アウトソーシングまで、各分野の高度な技術を背景に問題点を本質的に捉えスピーディに解決していく。
課題解決から価値創出へ繋ぐ架け橋として、いつの時代も信頼されるプロフェッショナル集団である。



Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

0. 高信頼なWebシステム基盤構築のポイント

セミナーのポイント

信頼性の高いWebシステム基盤を構築するには

- 信頼性とは
  - 定められた期間中に、システムが期待通りに動作すること。すなわち、要件に定められた機能、サービスを提供すること。

- 信頼性の高いWebシステムとは、
  - ピーク時にも安定して稼動できる
  - 障害が発生したときも影響を少なくできる
  - ビジネス規模の拡大に対応できる
  - 障害時に迅速に状況を把握できる

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

アジェンダ

1. システム基盤のアーキテクチャを選定する。

2. ピーク時にも安定して稼動する。 高性能

3. 障害が発生したときも影響を少なくできる。 可用性

4. ビジネス規模の拡大に対応できる。 拡張性

5. 障害を予防し障害時も迅速に対応できる。 運用性

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

アジェンダ

1. システム基盤のアーキテクチャを選定する。

2. ピーク時にも安定して稼動する。 高性能

3. 障害が発生したときも影響を少なくできる。 可用性

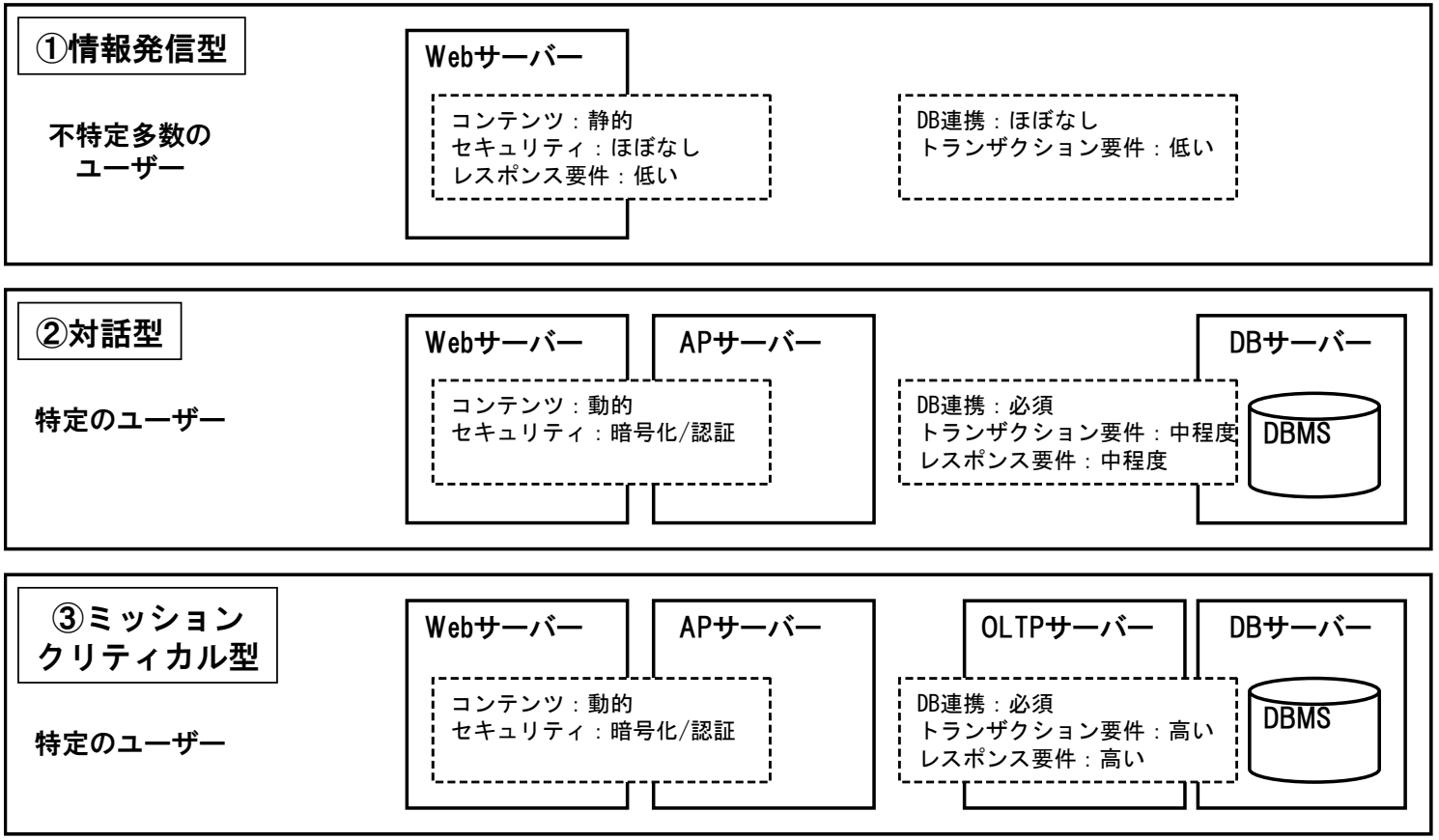
4. ビジネス規模の拡大に対応できる。 拡張性

5. 障害を予防し障害時も迅速に対応できる。 運用性

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

1-1. システム基盤アーキテクチャの選定

- Webシステムの目的ごとにアーキテクチャが異なる。



- 本セミナーでは、J2EEのAPサーバーを対象に、システム基盤構築のポイントを紹介する

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

アジェンダ

1. システム基盤のアーキテクチャを選定する。

2. ピーク時にも安定して稼動する。 高性能

（1）十分な能力を持ったハードウエアを選定する
（2）ガーベージコレクションを制御する
（3）リクエストの流量を制御し、スループットを確保する

3. 障害が発生したときも影響を少なくできる。 可用性

4. ビジネス規模の拡大に対応できる。 拡張性

5. 障害を予防し障害時も迅速に対応できる。 運用性

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

高性能

（1）十分な能力を持ったハードウエアを選定する

- Webシステム基盤に期待される性能要件
  - レスポンス時間
  - ピーク時のリクエスト量

- 性能要件を満たすようCPU性能・個数をサイジングする
  - 十分でなければ、レスポンス時間の要件を満たすことができない

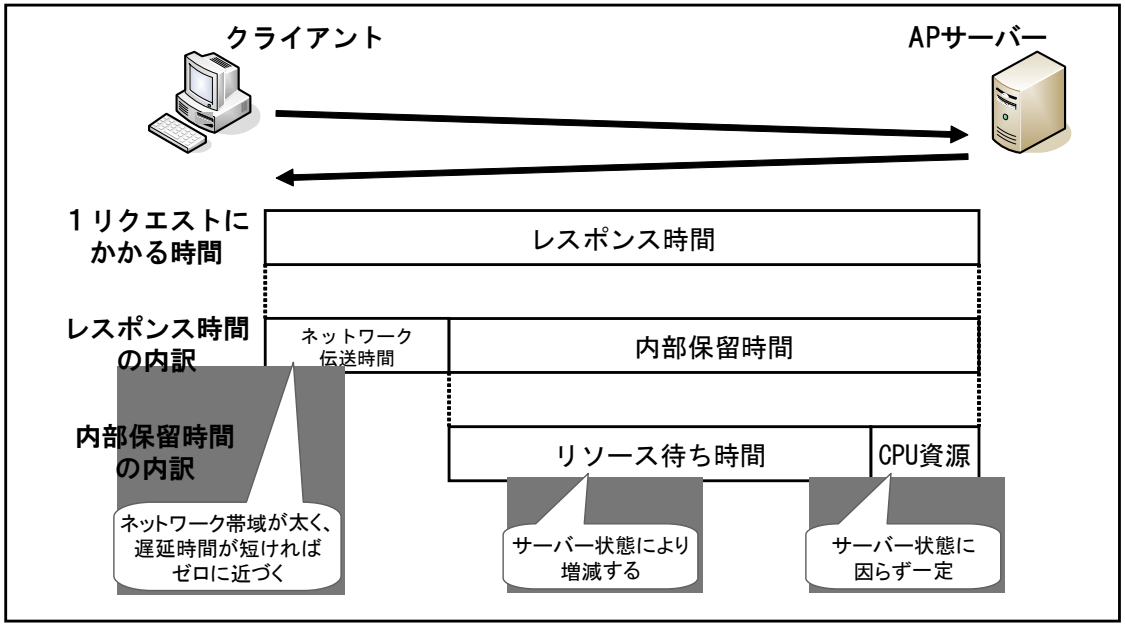
- CPUサイジング手順
  - 1リクエストを処理するのにかかるCPU資源はいくらか
  - ピーク時のリクエスト量を処理するのにかかるCPU資源の総量は
  - 必要となるCPUの性能・個数はいくらか

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

高性能

1リクエストを処理するのにかかるCPU資源はいくらか

- レスポンス時間の内訳を確認しておく
  - ネットワーク伝送時間、リソース待ち時間は状況により変化する。
  - 1リクエストの処理に必要となるCPU資源の量は、**サーバー状態に因らず一定**



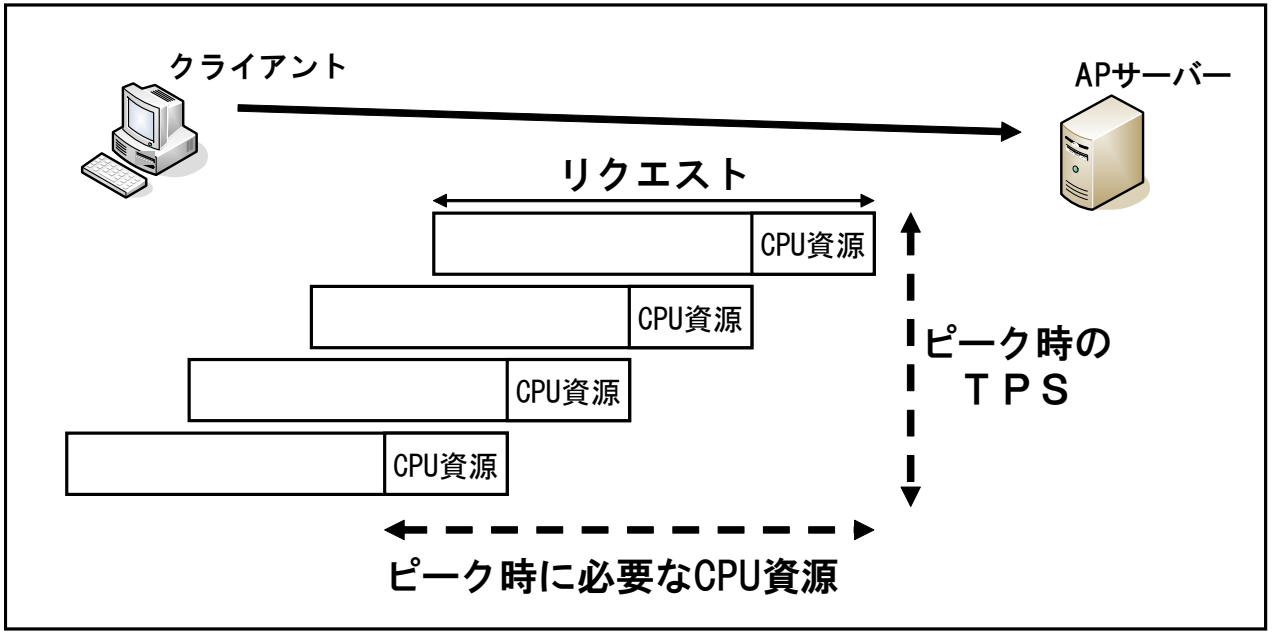
- CPU資源の量はCPU性能（モデル・クロック）とアプリケーションアーキテクチャ（EJB利用有無など）により決まる。

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

高性能

ピーク時のリクエスト量を処理するのにかかるCPU資源の総量は

- ピーク時のリクエスト量を、1秒あたり(TPS)で捉える
  - ピーク時のTPSは性能要件で定められる
  - TPS:1秒あたりのリクエスト量



- APサーバーで必要となるCPU資源の総量は

ピーク時のTPS×1リクエストあたりのCPU資源

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

2-1. ハードウエア選定

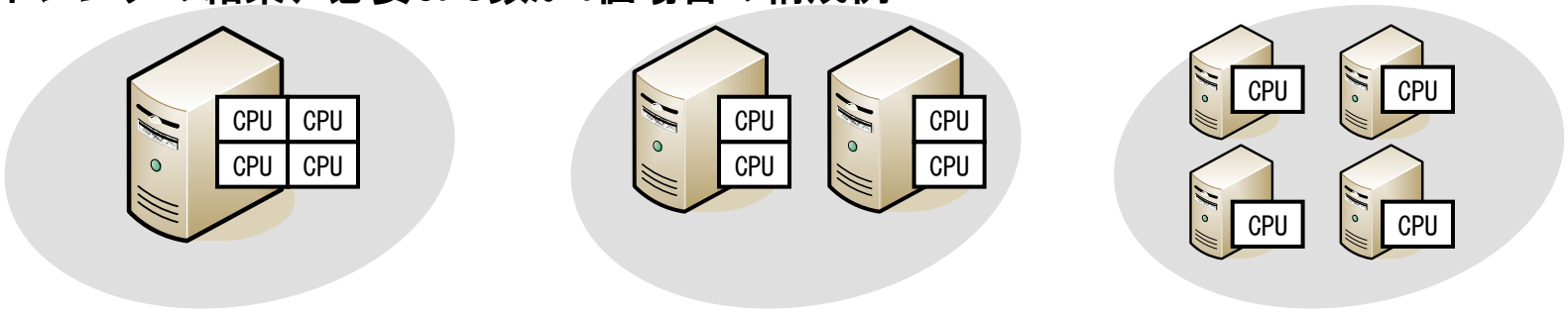
高性能

必要となるCPUの性能・個数はいくらか

- 必要となるCPU資源の総量に、CPU使用率を考慮して CPU性能と個数を求める

- 算出したCPU個数にもとづいて、サーバー構成を決定する

サイジングの結果、必要CPU数が4個場合の構成例



4CPU 構成x 1サーバー　　2CPU構成 x 2サーバー　　1CPU構成 x 4サーバー

低 ← 負荷分散・冗長性 → 高

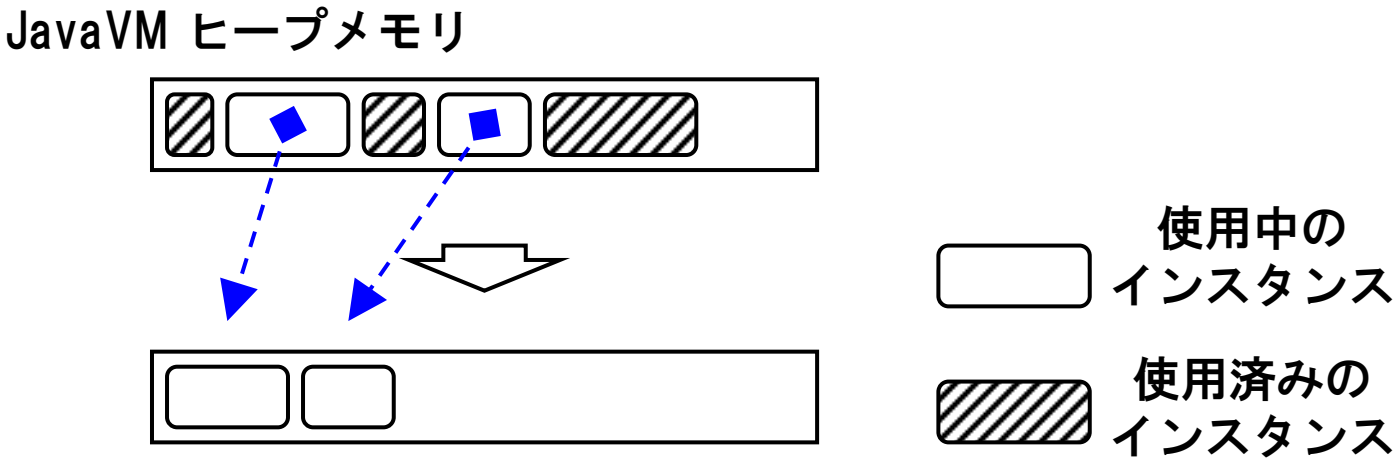
低 ← 運用管理コスト → 高

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

高性能

(2)ガーベージコレクションを制御する

- Javaの自動メモリー管理
  - メモリー確保・解放のプログラム不要
  - 使用済みメモリーの解放と未使用領域の確保はGCが行う



- GCには、検討しておくべき課題がある。
  - GC中に一時的に処理が停止する、GCが頻発する、

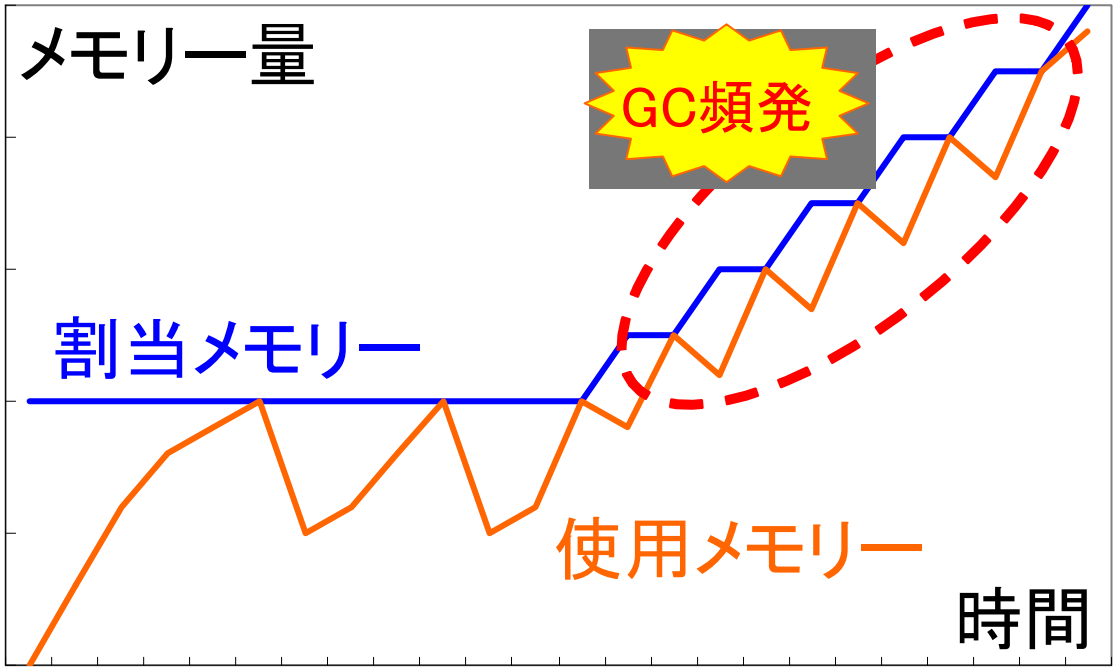
Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

2-2. ガーベージコレクション制御

高性能

ガーベージコレクション(GC)の課題

- GC中は一時的に処理が停止する
- GCを実行してもメモリーが解放されない場合、頻繁にGCが発生し、性能が劣化する
  - 使用中のインスタンスは、GCを実行してもメモリーが解放されない。長期間使用されるインスタンスが多いと問題になる。



Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

高性能

GCの処理速度を改善する－世代別GC

- Javaのインスタンスの多くは一時的なものが占める

	例	割合	生存期間	GC
短命 インスタンス	リクエストーレスポンス間の処理に使用されるもの	多い	短い。 数秒程度	Copy GC （高速）
長命 インスタンス	セッション情報に保存されるもの。ユーザーID、権限情報など	少ない	長い。 数分程度	Full GC （全領域）

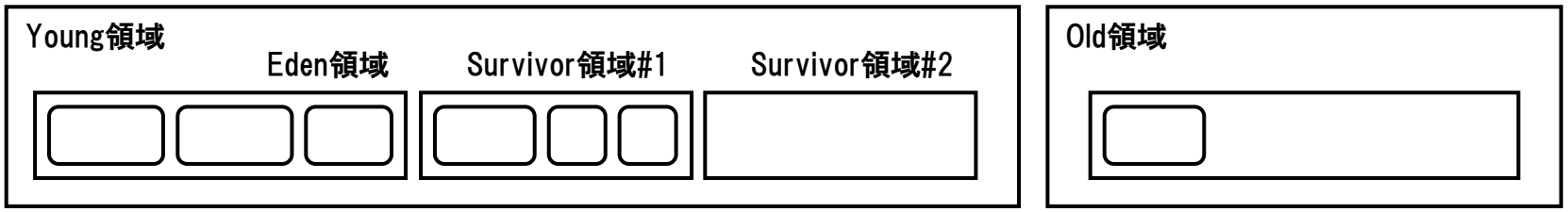
- 世代別GCでは、インスタンスを生存期間で分け、短命インスタンスを対象として高速なGC（Copy GC）を行う
- Full GC の発生頻度を制御することが重要

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

高性能

Full GCの発生頻度を制御する

- JavaVMのヒープ領域をYoung領域とOld領域に分ける



  - インスタンスは、Young領域に生成される。Copy GCを繰り返した結果、長命インスタンスがOld領域に配置される。
  - Full GCは、Old領域が一杯になったときに行われる。発生頻度は、長命インスタンス＝セッション情報が生成されるスピードによって決まる。
  - これは1秒あたりの想定ログイン数、セッション情報サイズから求まる。

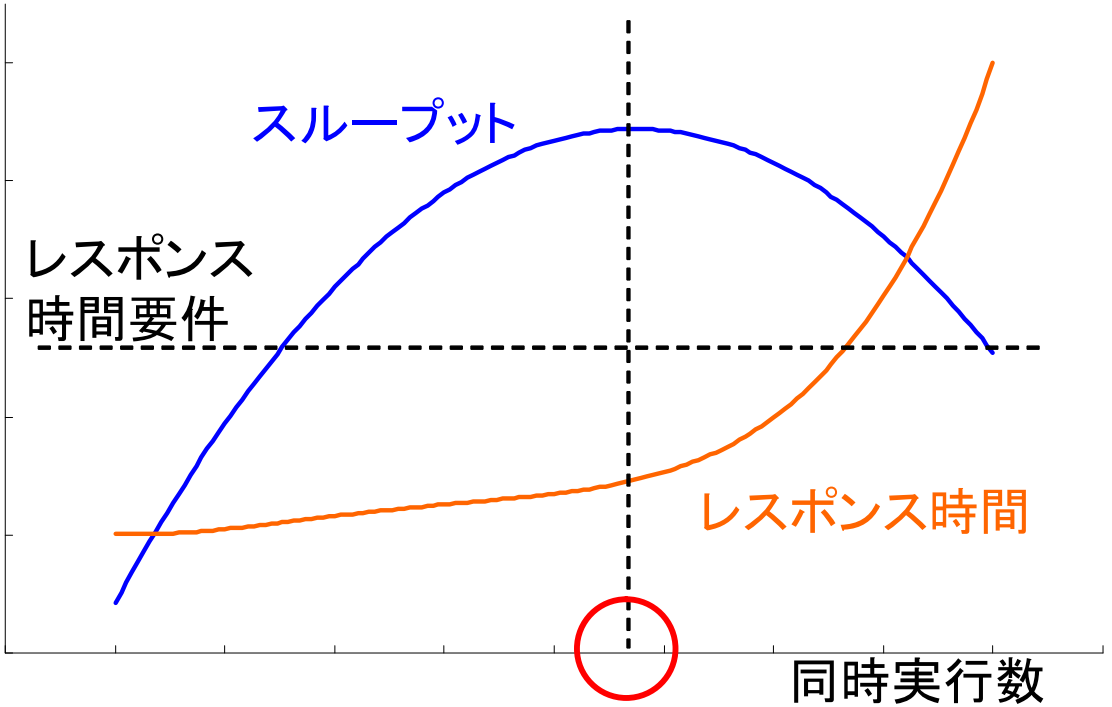
- ログイン数が多い場合、セッション情報サイズが大きい場合は、Old領域が一杯になるスピードが速い。Full GCの発生頻度が多くなる
- セッションタイムアウトが長い場合は、セッション情報が解放されない。Full GC多発の原因となる

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

高性能

(3)リクエストの流量を制御し、スループットを確保する

- 同時実行数は、レスポンス時間、スループットと関係がある
  - 同時実行数が少なすぎれば、スループットが低くなる
  - 多すぎれば、レスポンス時間が悪化し、スループットも低下する
- スループットが高くなるように、リクエストの流量＝同時実行数を設定する

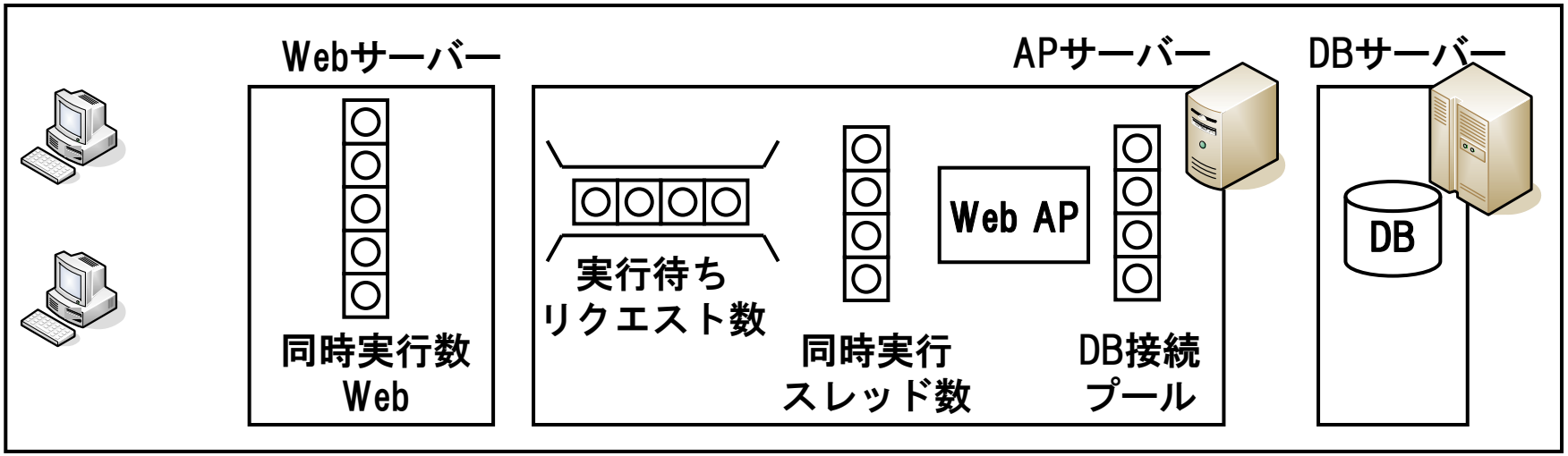


Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

2-3. リクエストの流量を制御する

高性能

流量制御の設定ポイント



同時実行スレッド数	APサーバーの同時実行数。少ないとスループットが低くなる。多いとレスポンス時間悪化、スループット低下。
DB接続プール	DB高速化のため、接続をプールしておく。同時実行スレッド数より大きく設定する。少ないとDB接続プール待ちが発生する
実行待ちリクエスト数	APサーバーの実行キュー。少ないとキューあふれエラーの発生頻度が高くなる。多いとレスポンス時間が長くなった場合にもエラーとできない
同時実行数Web	Webサーバーの同時実行数。同時実行スレッド数＋実行待ちリクエスト数より多くする

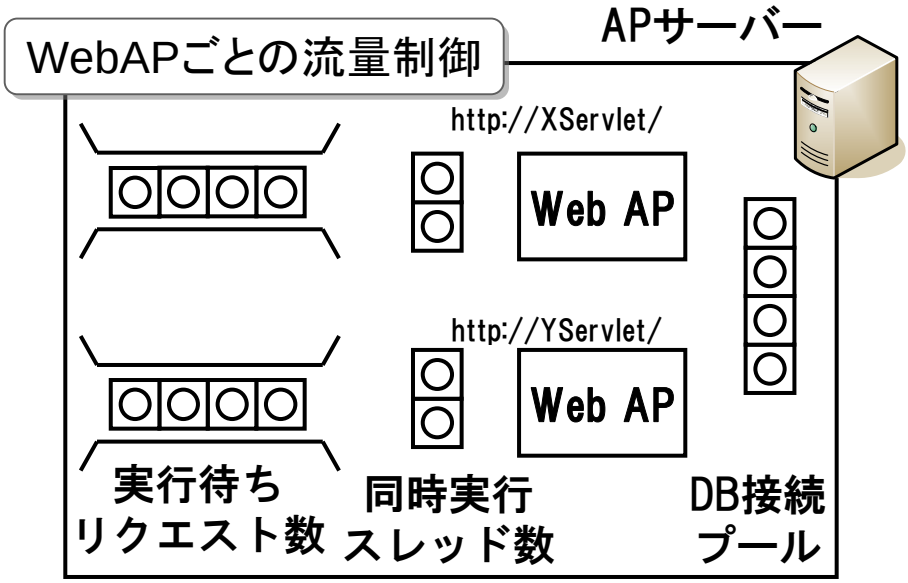
Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

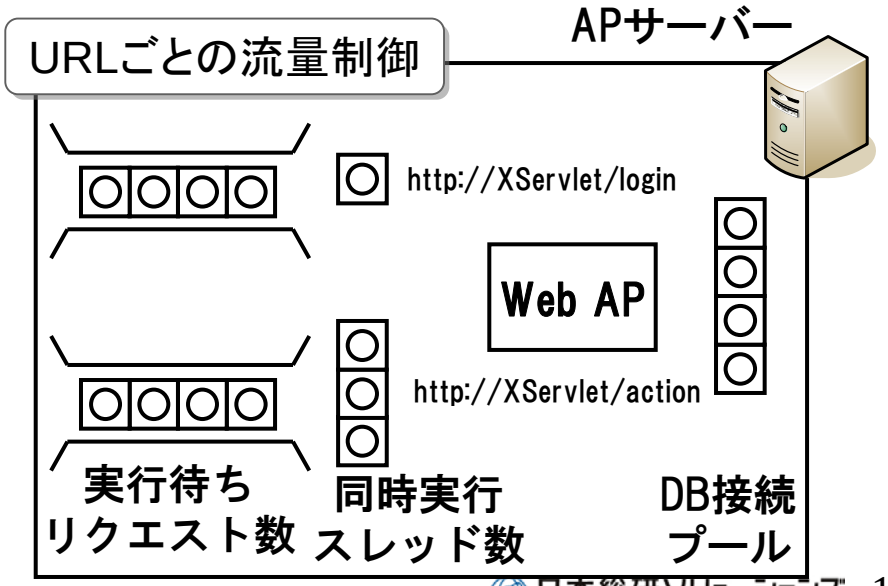
2-3. リクエストの流量を制御する

高性能

アプリケーション特性にあわせた流量制御

- Webコンテナに複数のWebアプリを配置して運用する場合
  - 一方のWebアプリにリクエストが集中すると、他方に影響がある
  - Webアプリケーションごとに流量制御を行う Cosminexus
- 処理時間の長いプログラムが含まれる場合
  - 特定のプログラムが実行スレッドを占有してしまう
  - プログラムごと(URLごと)に流量制御を行う Cosminexus





Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

アジェンダ

1. システム基盤のアーキテクチャを選定する

2. ピーク時にも安定して稼動する。 高性能

3. 障害が発生したときも影響を少なくできる。 可用性

4. ビジネス規模の拡大に対応できる。 拡張性

（1）機器を多重化し、冗長化構成とする
（2）グローバルトランザクション方式の考慮点

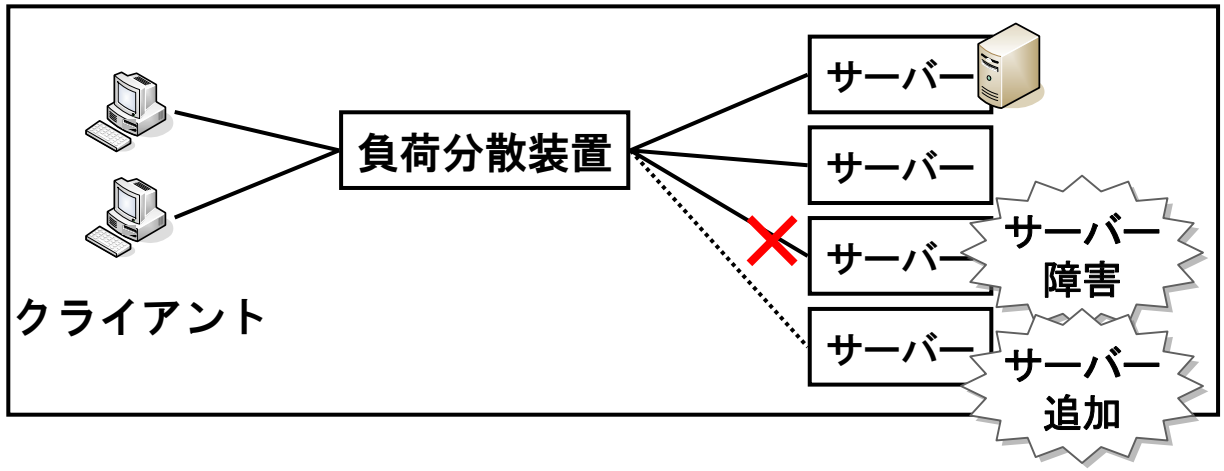
5. 障害を予防し障害時も迅速に対応できる。 運用性

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

可用性 拡張性

（1）機器を多重化し、冗長化構成とする

負荷分散装置を利用した水平負荷分散



- 平常時は複数のサーバーで分散して処理
- サーバー障害時は稼動サーバーで処理継続
- サーバーを追加でき、スケーラビリティを確保

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

負荷分散装置を用いた水平負荷分散の考慮点

- サーバーへの振り分け方法
  - サーバーごとの処理量を平準化する方式を選択する

ラウンドロビン | 最小接続数 | 最小応答時間 | IPアドレス

- セッションを維持する方法
  - 既存のセッションは同じサーバーに振り分ける

Cookie | URLリライト | IPアドレス

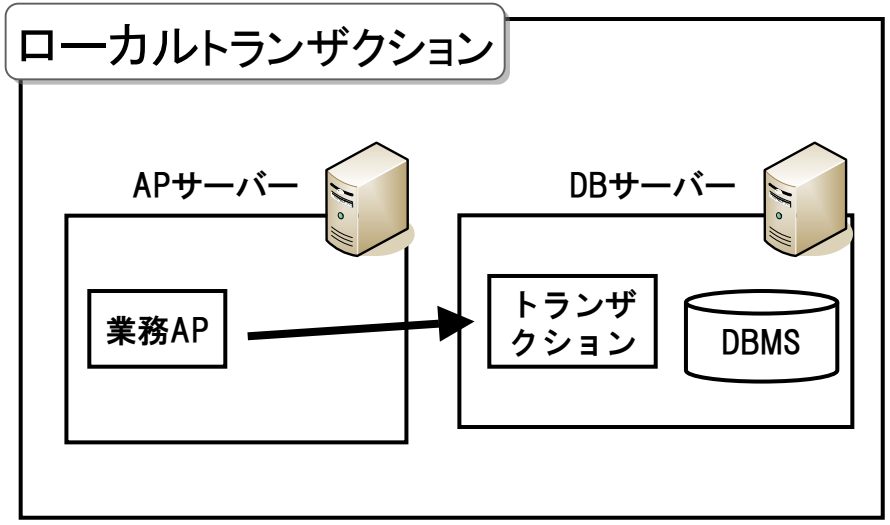
- サーバーの死活監視
  - 確実性と負荷、運用性とのバランス

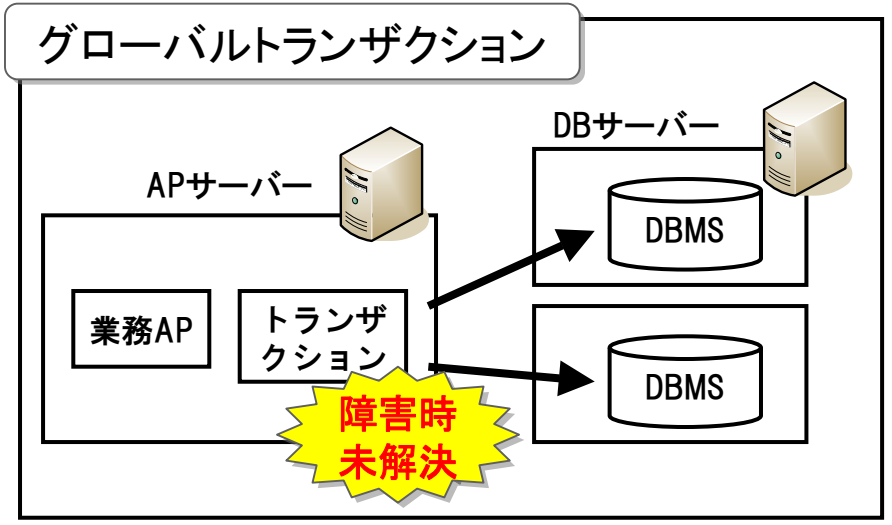
Ping | HTTP | アプリケーション

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

(2)グローバルトランザクション方式の考慮点

- ローカルトランザクション
  - トランザクションはDBサーバー管理
  - APサーバー障害時はDBサーバーがリカバリー
- グローバルトランザクション
  - トランザクションはAPサーバー管理
  - APサーバー障害時はトランザクションが未解決となる

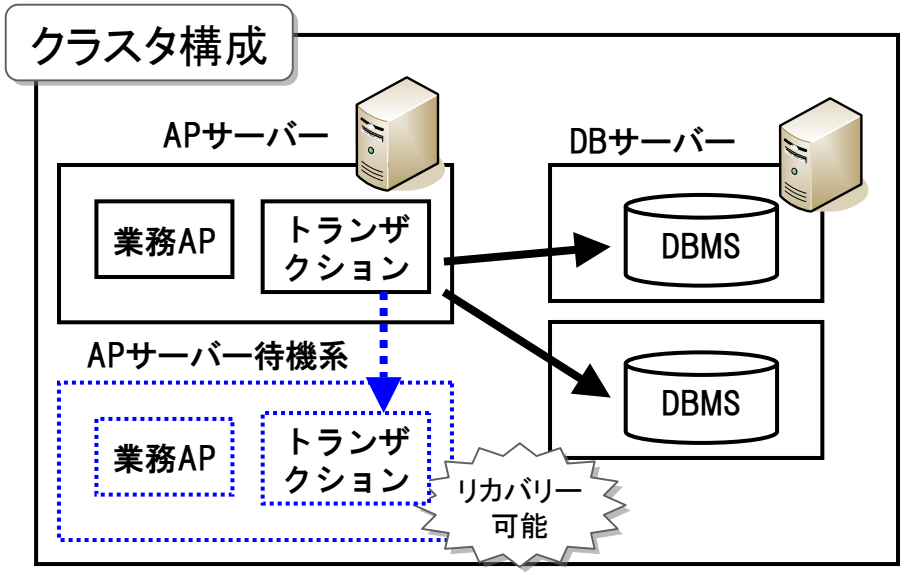




Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

APサーバー障害時にリカバリーするには

- クラスタ構成
  - ディスクを共有し、稼動系障害時は待機系がリカバリー
  - 稼動サーバーごとに待機・リカバリー用のサーバーが必要となり、リソースの無駄が多い



  - ミドルウエアによっては、一台で複数のサーバーのリカバリーが可能な専用サーバが利用できる（N+1クラスタ、Cosminexus ）

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

アジェンダ

1. システム基盤のアーキテクチャを選定する

2. ピーク時にも安定して稼動する。 高性能

3. 障害が発生したときも影響を少なくできる。 可用性

4. ビジネス規模の拡大に対応できる。 拡張性

5. 障害を予防し障害時も迅速に対応できる。 運用性

（1）ログ・トレースの収集
（2）ログ・トレースの解析

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

運用性

運用性確保のポイントとは

- 平常時の運用
  - システムの起動、計画停止
  - 稼動状況の監視、稼動統計の確認
  - モジュール入替、パッチ適用、定義変更
- 障害時の運用
  - 情報取得、ログ・トレース収集
  - ログ・トレース解析
  - フェールオーバー、フェールバック

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

運用性

(1)ログ・トレースの収集

- タイムリーに、必要なログを収集する
  - 出力場所は多岐に渡る
  - 迅速に収集しなければ情報が失われる

1. 顧客からの問い合わせ、または障害検知
2. 現象確認(再現性)
3. エラーログ、メッセージの確認
4. サーバーの稼動状況の確認(CPU、メモリ)
5. 問題点の一次切り分け(サーバー、業務アプリ)
6. 業務復旧(システム暫定復旧)
7. ベンダーへの問い合わせ(指示待ち)
8. 調査用の各種ログの送付
9. 対策(業務アプリ修正、製品パッチなど)
10. 障害対応完了

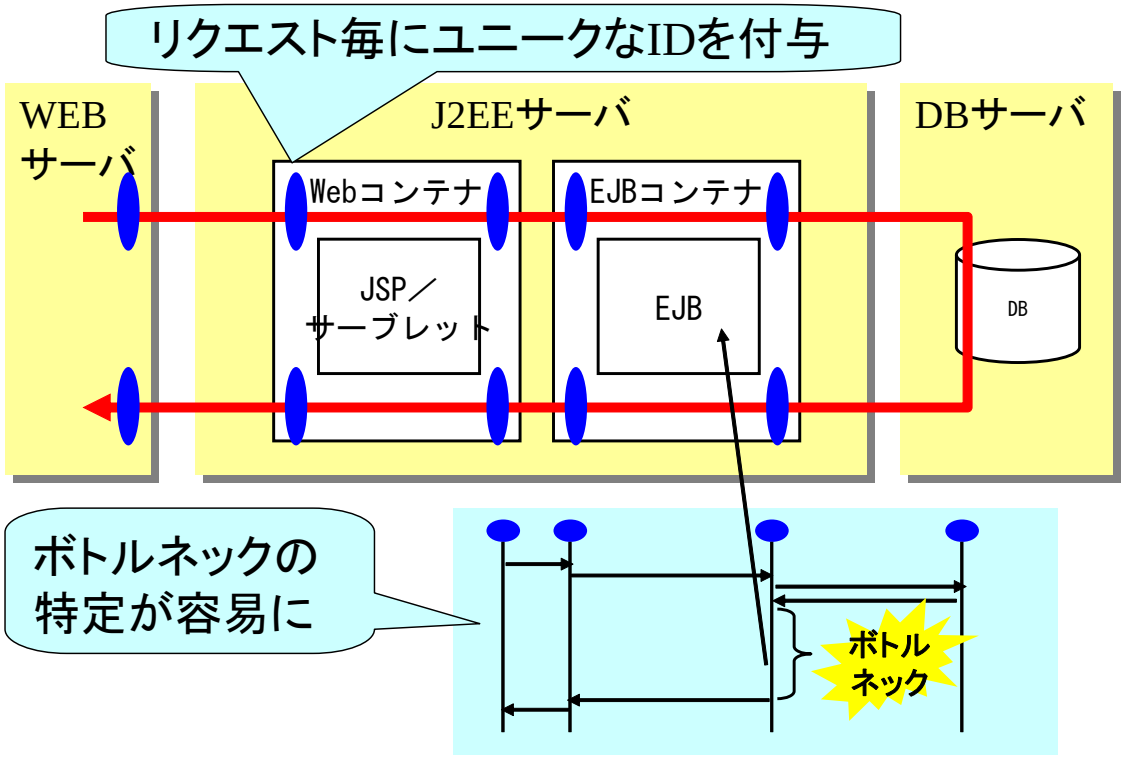
- 必要なログを**一括で効率よく収集**できるよう工夫する
  - ミドルウエアの機能を利用する(Cosminexus)

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

運用性

(2)ログ・トレースの解析

- 状況を確認するには、ログ・トレースから、リクエストの処理シーケンスを解析する
  - 複数のログにまたがるため解析が困難
  - ミドルウエアの機能を利用する(PRFトレース、Cosminexus)



Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

まとめ

- 高信頼なシステム基盤を構築するには
- 性能確保
  - CPUサイジング、ガーベージコレクションの制御、流量制御
- 可用性・拡張性
  - 冗長化構成、水平負荷分散、クラスタ構成、N+1クラスタ
- 運用性
  - ログ収集、トレース解析

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved

終わりに

本日のご説明内容他の詳細については

日経BP社:「システム基盤の統合ノウハウ」もご参照ください
(2008年1月刊行予定)

※本文中に記載されている会社名・商標名は、各社の商標または登録商標です。

Copyright © 2007 JRI Solutions, Ltd. All Rights Reserved