WebSphere Application Server V8.5 Workshop

その他新機能

日本アイ・ビー・エム システムズ・エンジニアリング株式会社
Webプラットフォーム
湯元 孝治

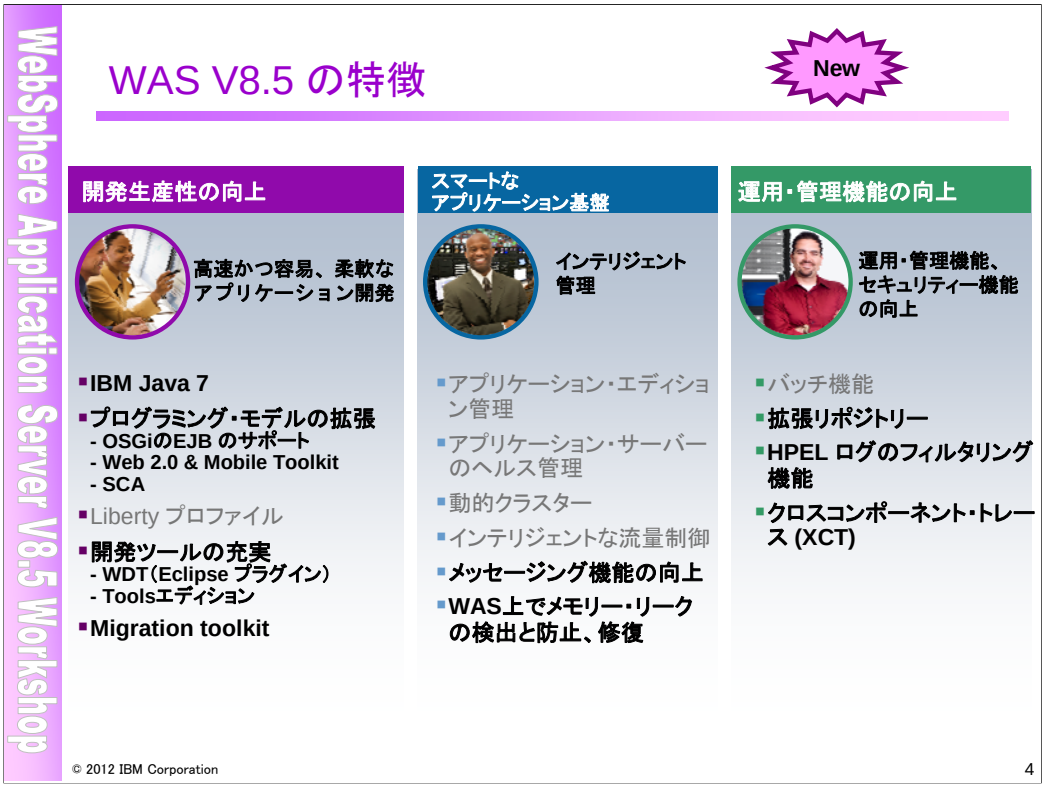
Agenda

1. 開発
2. 構成管理
3. メッセージング
4. 問題判別
5. マイグレーション
6. まとめ

参考文献

WebSphere Application Server V8.5 Workshop

1. 開発



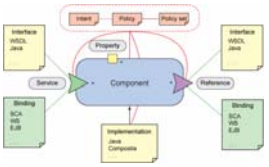
WAS V8.5では、Libertyプロファイルやインテリジェント管理、バッチ機能以外にも多くの新機能があります。

このセッションでは、これらの新機能についてご紹介していきます。

WebSphere Application Server V8.5 Workshop

豊富なプログラミング・インターフェイスのサポート

- **Java EE 6**
  - 軽量化、拡張性、開発容易性の向上
- New **Java SE 7**
  - 言語拡張、パフォーマンス、メモリー管理機能の向上
- **Web 2.0 & Mobile**
  - RIAやスマートフォン対応のアプリケーション開発を支援
- Update **Javaバッチ**
  - バッチ・アプリケーションを開発・実行するためのフレームワーク
- Update **OSGiアプリケーション**
  - Javaアプリケーションのモジュール化により柔軟性を向上
- **XML**
  - W3CのXML標準であるXPath 2.0、XSLT 2.0、XQuery 1.0に対応
- **SCA** (Service Component Architecture)
  - SOAのプログラミング・モデルの標準をサポート
- **SIPおよびCEA** (Communication Enabled Applications)
  - Webと音声を統合したアプリケーションの開発を支援



WASはこれまで業界標準のJavaやWebに関連する最新技術に対応し、多様なプログラミング・インターフェイスをサポートしてきました。

WAS V8.5では、最新のJava EE 6に対応しており、2011年にリリースされたJava SE 7にも新しく対応しています。

WAS V8.0から取り込まれているJavaバッチやOSGiなどの技術に関してもサポートを拡張しています。

そのほかにも、XMLファイルの操作APIやSOAプログラミング・モデルのサポート、音声アプリケーションの開発を支援するSIP/CEAを標準としてサポートしています。

WebSphere Application Server V8.5 Workshop

Java 7 – SDKの選択が可能に

New

- WAS V8.5 から Java SE 7 をサポート
  - ◆SDK6 と SDK7 を選択可能
    - デフォルトはSDK6
    - オプションとしてIBM SDK7を Installation Manager からインストール可能
    - 管理コンソール or manageSDK コマンド or wsadmin コマンド
  - ◆IBM Java 7
    - Project Coin
    - NIO.2（非同期 I/O）
    - JVM最適化
    - メモリー管理
      - 世代別 GC がデフォルト（WAS V8.0から実装）
      - Balanced GC （WAS V8.0から実装）
      - リアルタイムGC （WebSphere Real Time の評価版）
    - 共有クラスキャッシュ

WAS V8.5からJava 7を使用できるようになりました。デフォルトでは、SDK6がインストールされますが、オプションとしてSDK7をインストールし、どちらのSDKを使用するか選択することができます。IBM Java 7は以下のような特徴があります。

・Project Coin

Java言語としての比較的小さな拡張であるためProject Coinと名づけられています。内容としては、Switch文の条件分岐にStringが使用できるようになったことや例外ハンドリングの強化でExceptionのマルチキャッチなどがあります。

・NIO.2（非同期I/O）

NIO.2はNew I/OというJavaのファイル操作APIの拡張です。非同期でのファイル操作APIがサポートされています。

・JVM最適化

IBM SDK 7 には、IBM J9 V2.6 仮想マシン (JVM) が組み込まれています。メモリー使用量の削減とパフォーマンスの向上が期待される、新しいロック最適化も実装されています 。

・メモリー管理

メモリー管理に関しては、IBM Java 7からデフォルトのGCポリシー世代別GC（genconGC）になり、Balanced GCという新しいGCが追加されています。

ただし、WASでは先行してV8.0からこれらは取り込まれていました。

また、リアルタイムGCと呼ばれる、GCの停止時間を極限に短くするための機能がTechnology Preview（評価版）で利用できます。

商用で使用する場合には、WebSphere Real Timeという製品が提供されています。

・共有クラス・キャッシュ

共有クラス・キャッシュを使用することで、メモリーのフットプリントの削減が可能で、起動時間やパフォーマンスの向上が期待できます。

IBM Java 7では、JITデータをキャッシュに格納したり、共有クラス・キャッシュ内の検索や削除といった操作APIを使用できるようになっています。

Web 2.0 & Mobile Toolkitは、Feature Pack for Web 2.0 & Mobileの後継にあたります。

Web 2.0 & Mobile Toolkitにより、リッチ・インターネット・アプリケーション(RIA)とモバイル対応のアプリケーションの開発生産性が向上します。

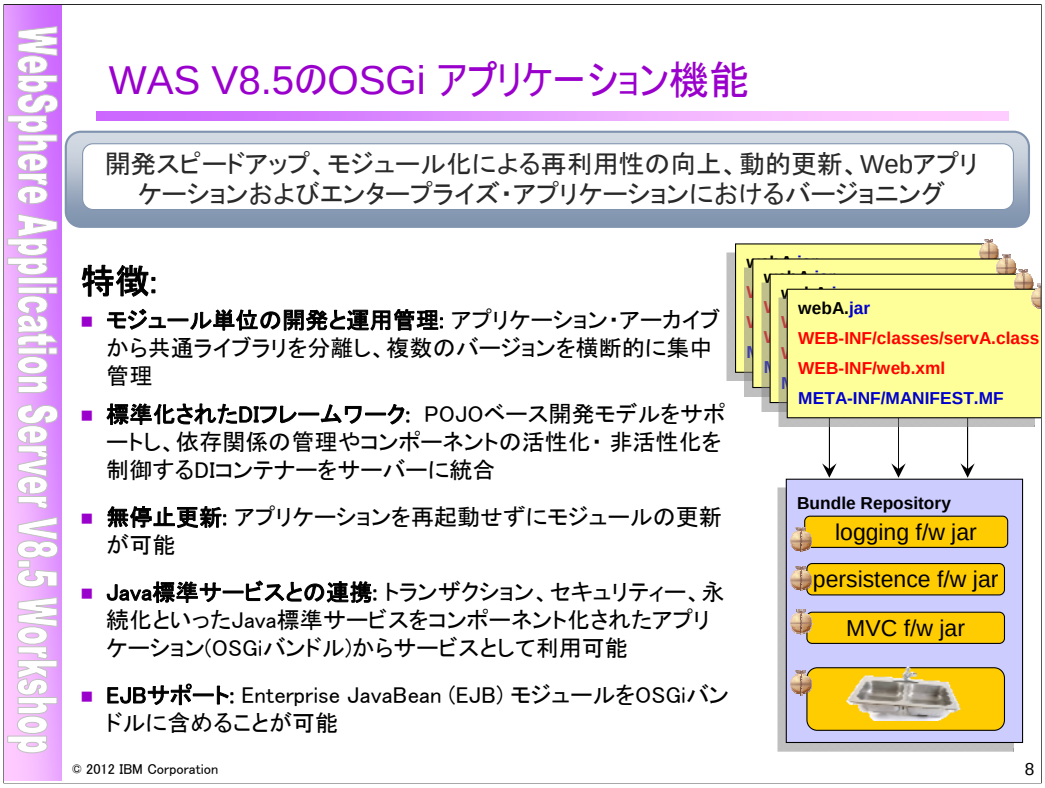
Web 2.0 & Mobile Toolkitの主な新機能としては、以下が挙げられます。

・IBM Dojo Toolkit：Dojo Toolkit 1.7.2 & 1.8（一部）をベースにしたツールキットで、RIAとモバイル・アプリケーションの作成を支援します。軽量のUI部品やデバイス向けのテーマを提供し、標準のHTML5、CSS3、JSを使用したリッチなモバイル向けアプリケーションが作成できます。

・IBM ILOG Dojo Diagrammer：ILOG JViewsの技術をベースに、より高度なUI部品を提供します。

・モバイル・アプリケーション・サービス：サーバー・サイドで利用可能なサービスとサンプル・アプリケーションを提供しています。

Feature Pack for Web 2.0 & Mobileで提供されていたAJAX Proxy、RPCアダプター、JSON4J、Webメッセージングは WASの標準機能として組み込まれています。



OSGiではJavaモジュールの動的な追加・更新・削除の仕組みを提供します。OSGiは、標準化団体であるOSGi Allianceによって仕様が策定されています。WAS V8.5でベースとなるOSGiの仕様は、エンタープライズ・アプリケーション向けの仕様として公開されているOSGi Enterprise Specification V4.2になります。

OSGiアプリケーションでは、JARファイルをバンドルとして管理します。OSGiではこのバンドル単位に、ロードやアンロードを行います。また、バンドルの構成情報は、バンドルのMETA-INF/MANIFEST.MFファイルにメタデータとして記述します。この中で、バンドル名やバージョン、そのバンドルがエクスポートまたはインポートするパッケージ等の情報を記載します。

OSGiアプリケーションでは、バンドルのバージョン管理が行えることもメリットの1つです。複数バージョンのバンドルを同時にアクティブにすることができるため、今まで複数アプリケーションから共通で使用するライブラリーのバージョン管理に苦労していたユーザーにとっては便利な機能となるでしょう。

WAS V8.5では、EJBモジュールをOSGiバンドルに含めることができるようになっています。

WAS V8.5での開発ツールについて紹介します。

まず、WDT (WAS Developer Tools for Eclipse) というツールです。その名の通り、Eclipseのプラグインとして導入できる開発ツールです。

対応するWASのバージョンは、 WAS V7.0/V8.0/V8.5/V8.5 Libertyプロファイルで、Java EEのアプリケーションを開発することができます。

ダウンロード・サイト、もしくはEclipseマーケットから無償でダウンロードすることができます。

他には、従来から提供されているRAD (Rational Application Developer) があります。

RADには、テストサーバーとしてWAS V7.0/V8.0/V8.5/V8.5（Liberty含む）が同梱されています。

WebSphere Portal / Javaバッチ開発などを備えた高機能の統合開発環境（IDE）です。

また、2011年11月からダウンロード開始になったIBM WebSphere Application Server - Tools EditionというWASとRADが同梱されたエディションも利用可能です。

WAS Developer Tools for Eclipse (WDT)

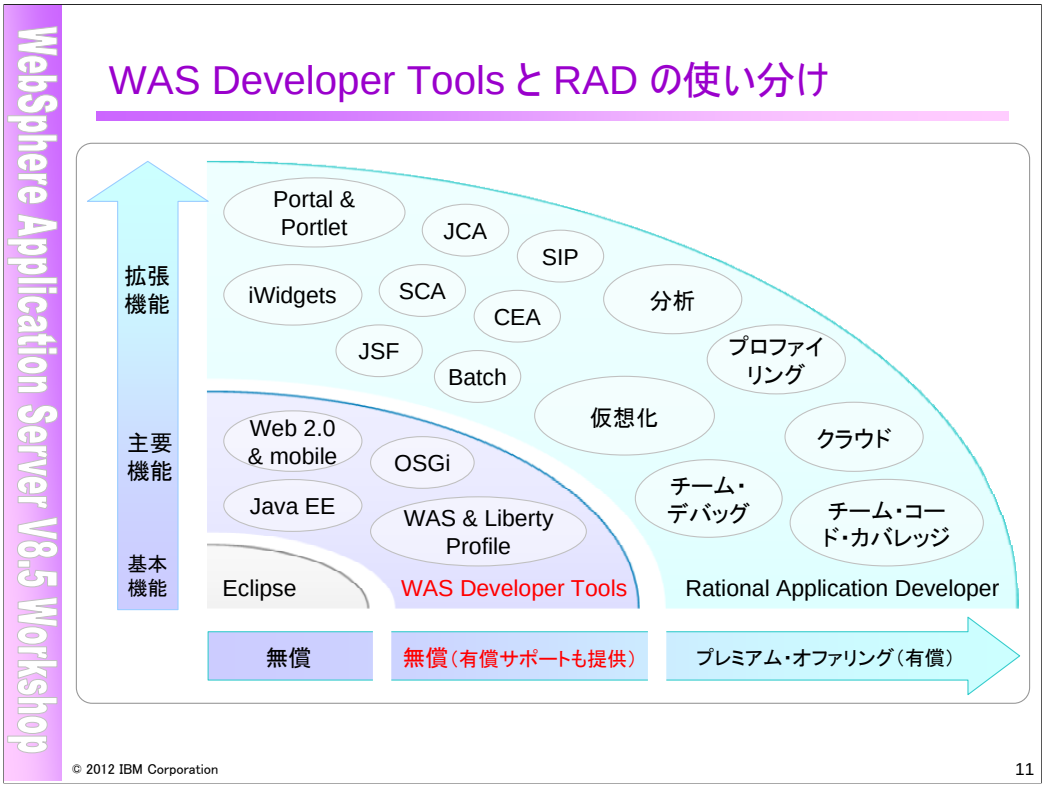
New

- Eclipse アドオンの無償開発ツール
  - WAS V7.0、V8.0、V8.5、Liberty Profileへの接続
  - WASの管理：起動、停止、コンソール起動、管理スクリプトの実行
  - ビジュアル・エディターを使用したWebアプリケーションの開発
  - Dojoベースのモバイル・アプリケーションの開発
  - OSGiアプリケーションの開発（V8.0/V8.5のみ）
  - IBM拡張を含むWAS用デプロイメント記述子の編集（GUI画面）
  - アプリケーションのデプロイ、テスト、デバッグ

開発者PC

もしくは

WAS Developer Tools for Eclipse (WDT) の特徴をまとめています。



WAS Developer ToolsとRADの使い分けについて記載しています。

一般的なWASのJavaEEアプリケーションの開発はWAS Developer Tools で十分実施できます。

さらに高度なPortalアプリケーションの開発やバッチ開発、チームでの開発を行う場合には、RADをはじめとするRational 製品群を使用することでより効率的に開発できます。

WebSphere Application Server V8.5 Workshop

2. 構成管理

WebSphere Application Server V8.5 Workshop

拡張リポジトリー・サービス

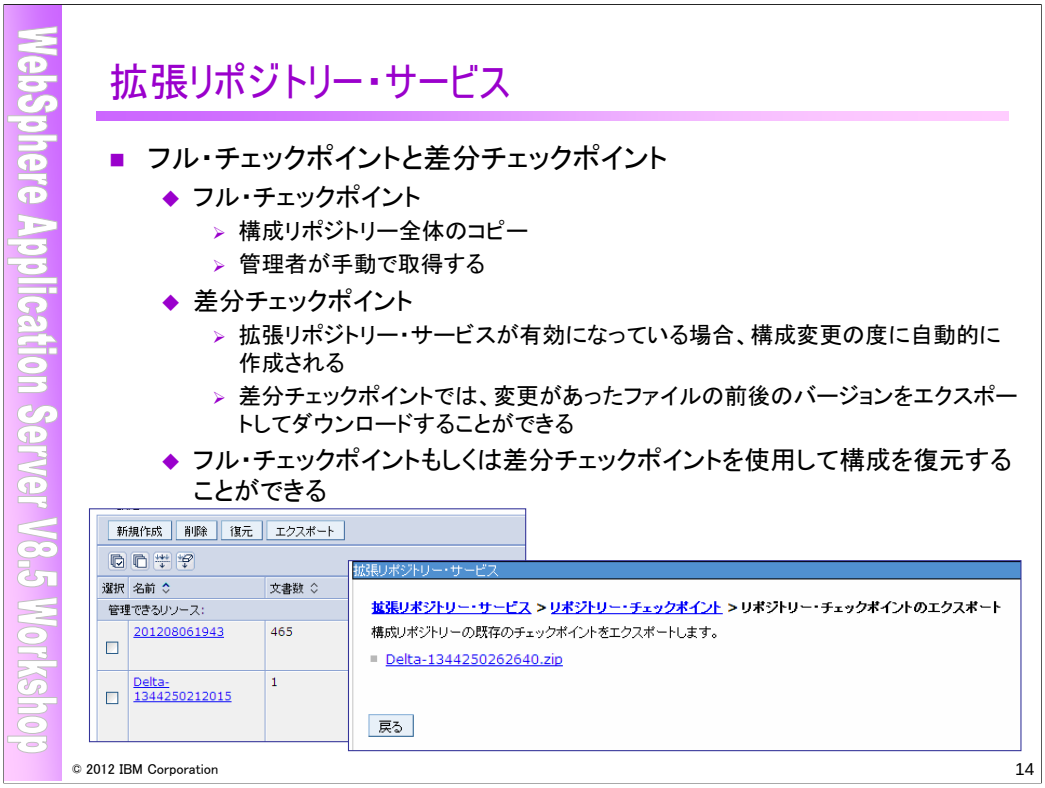
New

- WASの構成のチェック・ポイントを作成
  - WVE (WebSphere Virtual Enterprise) から統合された機能
  - 構成変更のリポジトリーを作成し、構成のロールバック等の管理が可能

構成管理の新機能として拡張リポジトリー・サービスが追加されました。

この機能はWVEから統合された機能ですが、NDエディションだけでなく、Base/Expressでも使用可能です。

拡張リポジトリー・サービスを有効にしている場合、構成変更の履歴をチェックポイントとして保存し、管理コンソールから確認、エクスポート、復元といった操作を行うことができます。



拡張リポジトリー・サービスのチェックポイントには、フル・チェックポイントと差分チェックポイントがあります。

フル・チェックポイントはその時点での構成リポジトリー全体のコピーです。
フル・チェックポイントは、管理者が任意のタイミングで手動で取得した場合のみ生成されます。

差分チェックポイントは、拡張リポジトリー・サービスが有効になっている場合に、構成変更と保管が行われると自動的に作成されます。
差分チェックポイントでは、Deltaから始まるエントリーが作成され、zip形式のファイルをダウンロードすることができます。ダウンロードしたファイルを解凍すると、beforeとafterというフォルダが含まれそれぞれに構成変更前後の構成ファイルが保管されています。

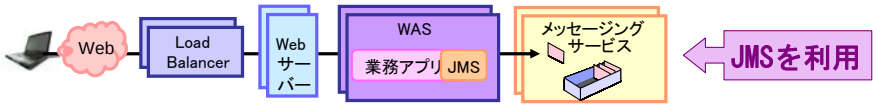
これらのチェックポイントを使用して構成を復元することが可能です。

WebSphere Application Server V8.5 Workshop

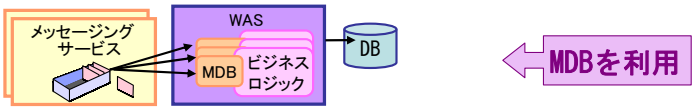
3. メッセージング

【参考】WASにおけるメッセージング処理形態

- WAS上のアプリケーションからメッセージング・サービスにメッセージを送受信する形態
  - アプリケーションはサーブレットやEJBとして稼働
    - WebからLoad Balancer, Webサーバーを通してリクエストを受信し、キュー・マネージャーへメッセージを送受信

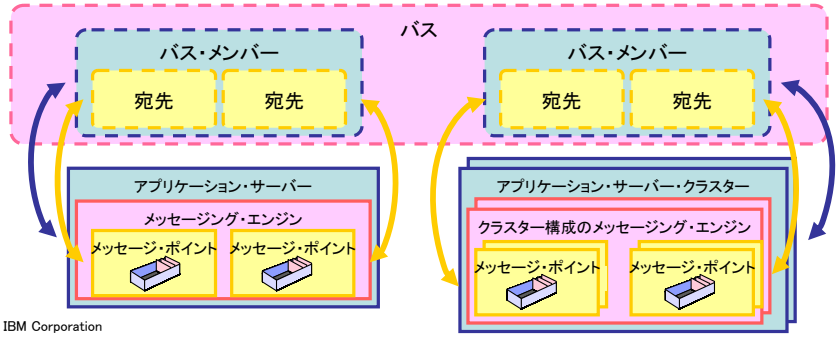


- メッセージング・サービスからWAS上のアプリケーションがメッセージを受信する形態
  - アプリケーションはMDBとして稼働
    - メッセージの到着をトリガーにMDBが起動し、ビジネス・ロジックを実行



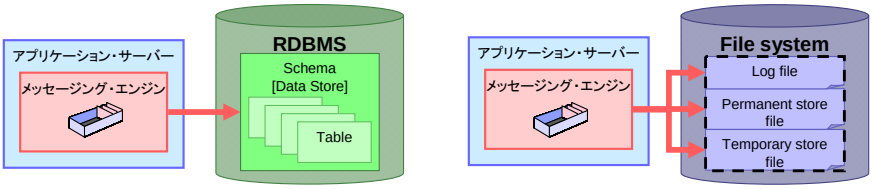
【参考】サービス統合バス(SIBus)とメッセージング・エンジン

- サービス統合バス(SIBus：Service Integration Busの略)とは・・・
  - WASのデフォルト・メッセージング・プロバイダー機能を論理的に定義したもの
  - 管理コンソールでメッセージング関連リソースをまとめて設定する
  - 実行時には、アプリケーションに対してメッセージング処理を隠蔽し、一つのJMSプロバイダーとしてサービスを提供する
- メッセージング・エンジン(ME)とは・・・
  - SIBusのメッセージング機能を実現するための主要コンポーネント
  - バス・メンバー(アプリケーション・サーバーまたはクラスター)上で稼働する
  - バス中に複数のMEがある場合には相互に連携してメッセージング機能を実現する



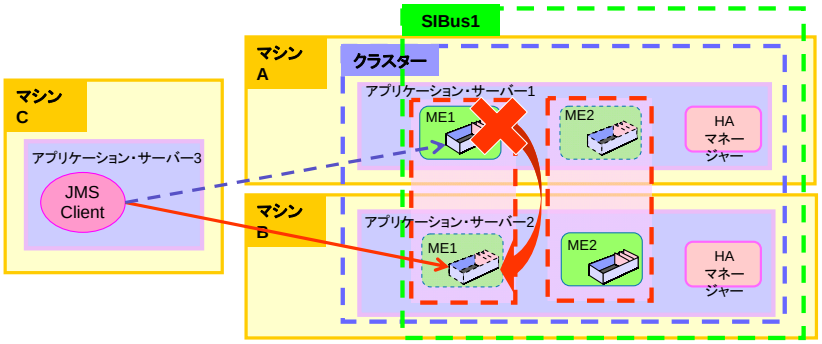
【参考】メッセージ・ストア

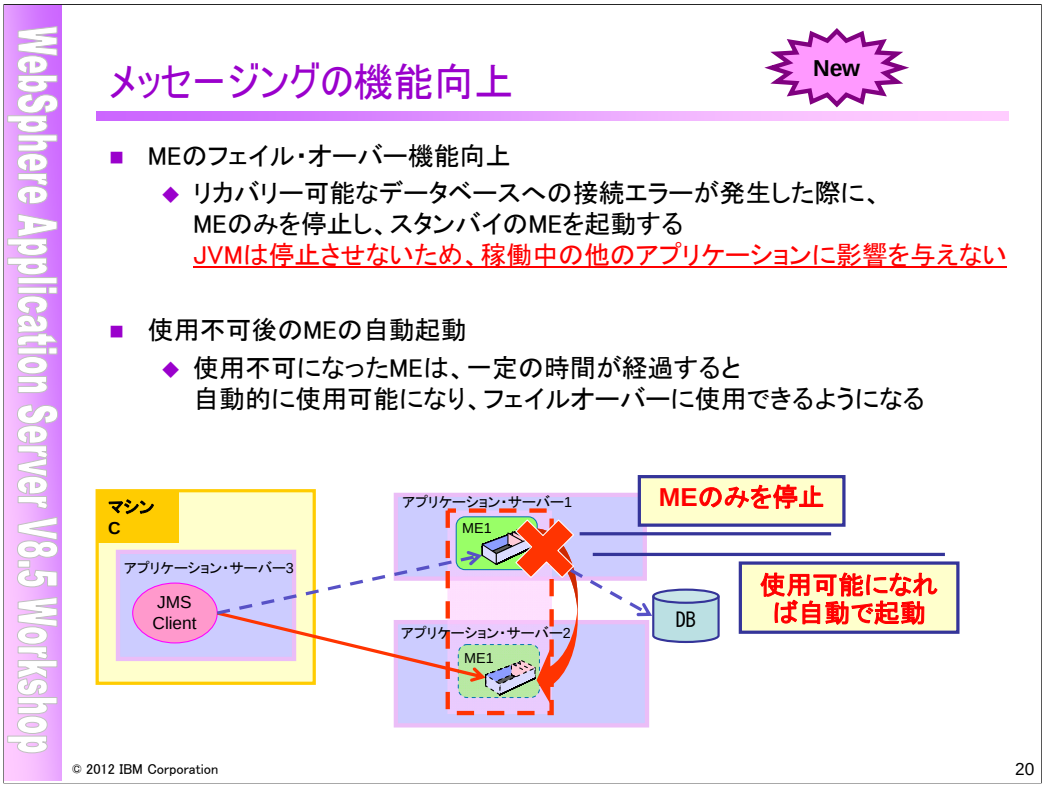
- データ・ストア
  - RDBMSを利用したメッセージ・ストア
  - MEとデータ・ストアのスキーマは1：1に対応
  - パーシステント・メッセージや仕掛かり中のトランザクションの情報などを格納する
  - MEの構成情報の一部（UUID）を保持・・・特定のMEからのみ利用可能となる
- ファイル・ストア
  - フラット・ファイルを利用したメッセージ・ストア
  - 以下の3種類のファイルを保持
    - ログ・ファイル・・・ローカル・トランザクションのログ
    - 永続ストア・ファイル・・・パーシステント・メッセージ保管用のファイル
    - 一時ストア・ファイル・・・一時データ保存用のファイル



【参考】SIBusのトポロジー構成

- HAマネージャーによるMEのフェールオーバー構成(Active-Standby)
  - クラスター構成のメッセージング・エンジンのフェールオーバーが可能
  - メッセージ・ストアを共有ディスク上に配置することでパーシステント・メッセージの引継ぎが可能になる
- MEのパーティショニング(クラスタリング)機能による負荷分散構成(Active-Active)
  - バス・メンバー内に複数のメッセージング・エンジンを起動し、処理の負荷分散が可能



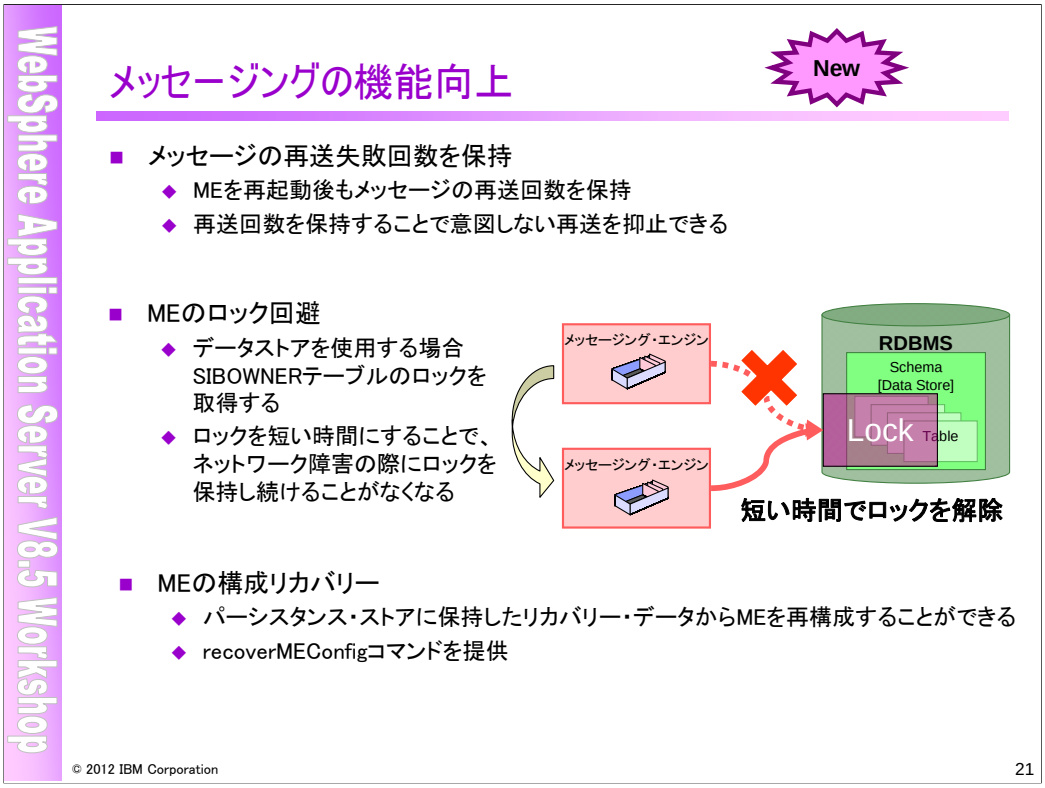


WAS V8.5では、メッセージング関連の機能向上が為されています。

ME（メッセージング・エンジン）がアクティブースタンバイの冗長構成をとっている場合、アクティブなMEからデータベースへの接続エラーが発生するとフェイル・オーバーが発生し、スタンバイのMEに処理が引き継がれます。ただし、V8.0まではMEを停止する際に、JVMも停止していたためJVM上で稼働するすべてのアプリケーションが影響を受ける可能性がありました。

V8.5では、フェイル・オーバーの際に、MEのみを停止し、スタンバイのMEを起動させるような挙動になりました。この場合JVMは停止させないため、稼働中の他のアプリケーションに影響を与えずフェイル・オーバーを実現できます。

また、これまでは障害等により使用不可になったMEは手動で起動させる必要がありましたが、一定時間後に自動的に使用可能になり、フェイル・オーバーに使用可能なスタンバイ状態になります。



その他のメッセージング関連の新機能です。

MEは、バス宛先に対してメッセージ送信が失敗すると「失敗送信の最大数」まで再送信しますが、試行に失敗した回数をサーバー再始動後も保持できるようになりました。そのため、1メッセージあたりの再送失敗回数を正確に指定し、回数を超えた際には、例外宛先に送信するなどの処理を行うことができます。

また、データストアを構成している際に、MEはSIOWNERテーブルのロックを取得しますが、使用不可になったMEが長時間ロックを保持することを可否するため、短い時間でロックが解除されるようになっています。

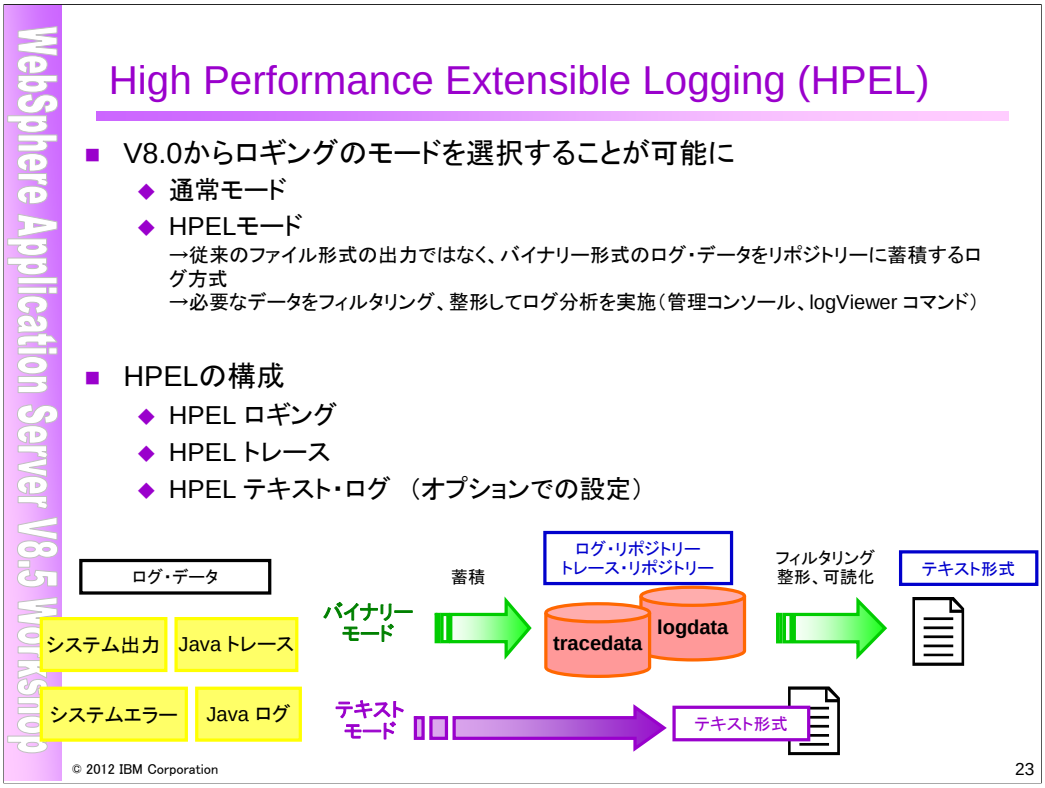
障害でMEの構成が壊れた場合に、パーシスタンス・ストアに保持したリカバリー・データからMEを再構成することができます。このメッセージ・ストアとしては、前のメッセージング・エンジンが接続されていたデータベースまたはファイル・ストア・システムが可能です。

recoverMEConfig コマンドが提供されています。

recoverMEConfig コマンド

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/rjk_recoverme_config.html

4. 問題判別



WAS V8.0から、High Performance Extensible Logging (HPEL) という新しいロギングのモードを選択することができるようになりました。

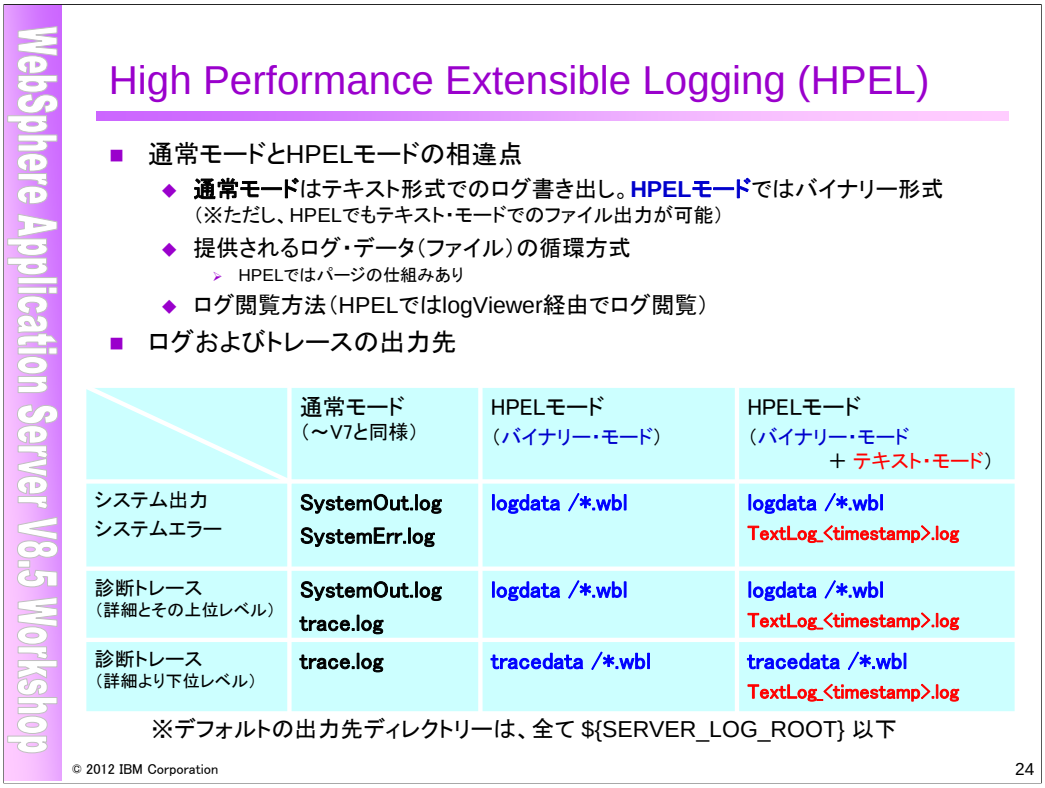
HPEL モードとは、従来のファイル形式の出力とは異なり、生成されるログ・データをバイナリー形式でリポジトリーと呼ばれるディレクトリーに蓄積していくログ方式です。

HPEL モードでは、バイナリー形式でファイルに出力されるため、ログを閲覧する場合には logViewer ツールもしくは管理コンソールで表示させます。

従来のログ方式が置き換わるわけではなく、通常モードとHPELモードを選択することができます。デフォルトは従来の通常モードとなっています。

HPEL　　モードは、HPELロギング、HPELトレース、HPELテキスト・ログを構成します。
HPEL ロギングでは、システム出力、システム・エラーといったログ・データをログ・リポジトリーにバイナリー形式で蓄積します。同様にHPEL トレースでは、トレースのデータをトレース・リポジトリーに蓄積します。

HPEL モードにすると必ずバイナリー・モードで出力されますが、オプションでテキスト・モードで出力させることも可能です。



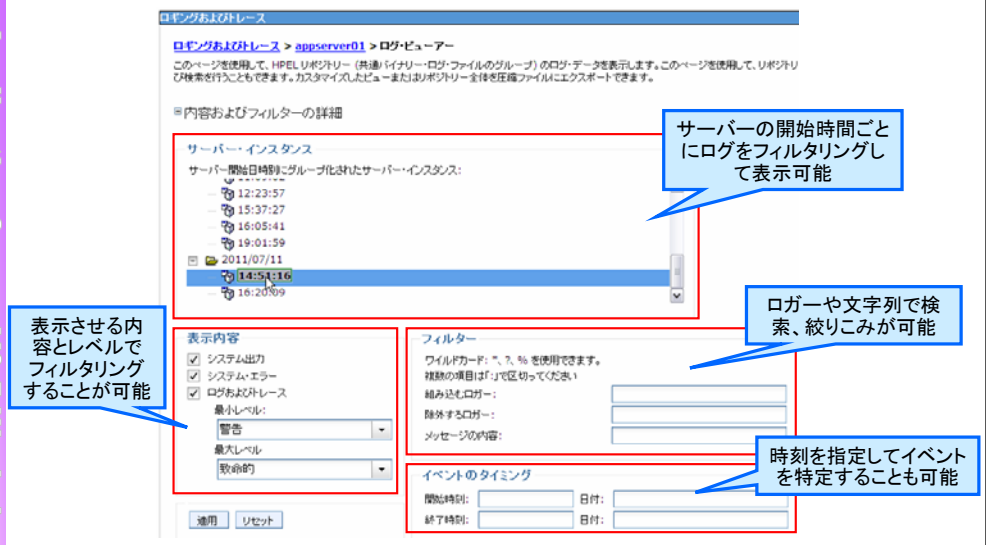
通常モードとHPELモードの相違点をまとめています。通常モードはテキスト形式のファイルへの出力、HPELモードではバイナリー形式でlogdataディレクトリーもしくはtracedataディレクトリーへログ出力されます。HPELモードでは、テキスト・モードを有効にすることでテキスト・ファイル形式での出力も可能ですが、システム出力、システム・エラー、トレースデータすべて1つのファイルへ書き出されます。

循環方式について、従来のロギング・モードではファイルベースの循環方式でしたが、HPELでは古いデータがパージされる仕組みとなっています。設定項目、方法については後で詳しく紹介します。

また、閲覧方法も異なります。HPEL モードのバイナリー形式のログはテキスト・エディター等で開けないため、ツールを使用して閲覧する必要があります。

HPEL の出力確認方法 (1/3)

- 管理コンソール

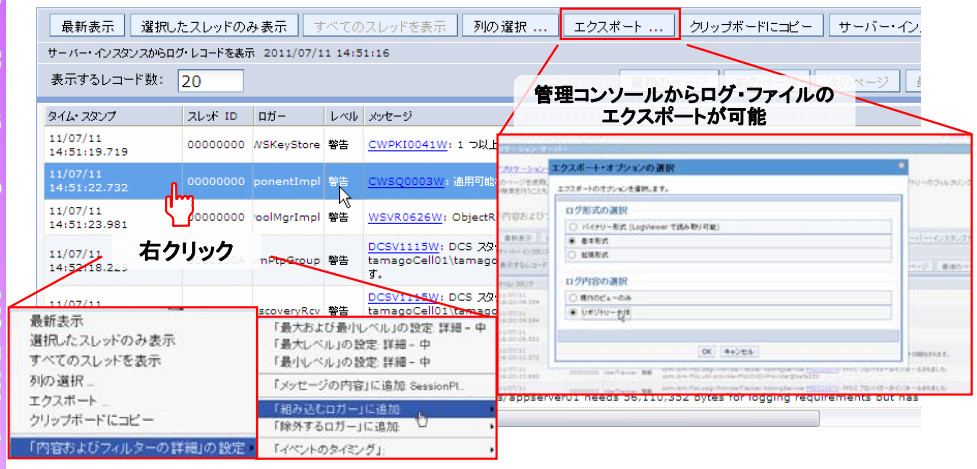


HPELのログ・レコードを閲覧する方法は、管理コンソールで表示させる方法とlogViewerコマンドで出力させる方法の2通りあります。

管理コンソールでは、ナビゲーション・ツリーから「トラブル・シューティング」-「ログおよびトレース」をクリックし、対象のサーバー名を選択します。関連項目から「HPEL ログおよびトレースの表示」をクリックするとログ・ビューアーの画面になります。「内容およびフィルターの詳細」を展開すると、表示させるログの絞り込みのオプションを選択できます。

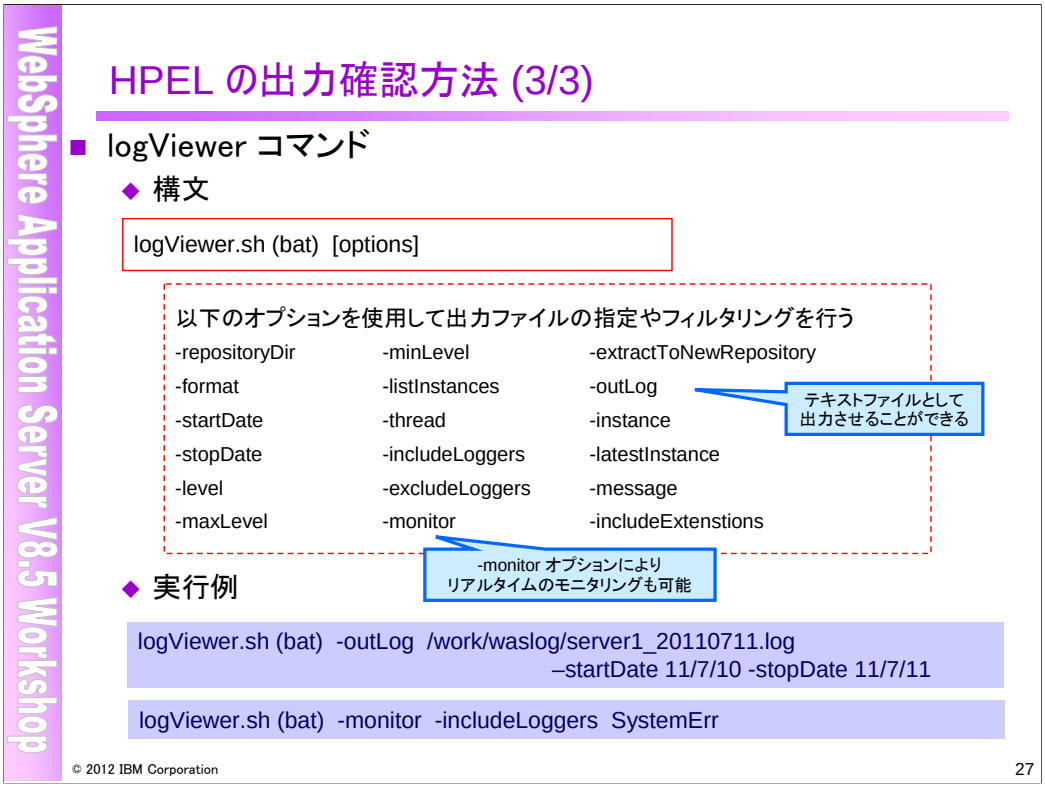
HPEL の出力確認方法 (2/3)

- 管理コンソール



表示させたログ・レコードの任意の行で右クリックすると、いくつかのオプションを選択できます。特定のログのメッセージやロガーを「内容およびフィルターの詳細」に追加することが可能です。

また、管理コンソールからログ・ファイルをエクスポートすることも可能です。ログ形式の選択で「バイナリー形式」、「基本形式」、「拡張形式」から、ログ内容の選択で「現行のビューのみ」、「リポジトリー全体」から選択します。

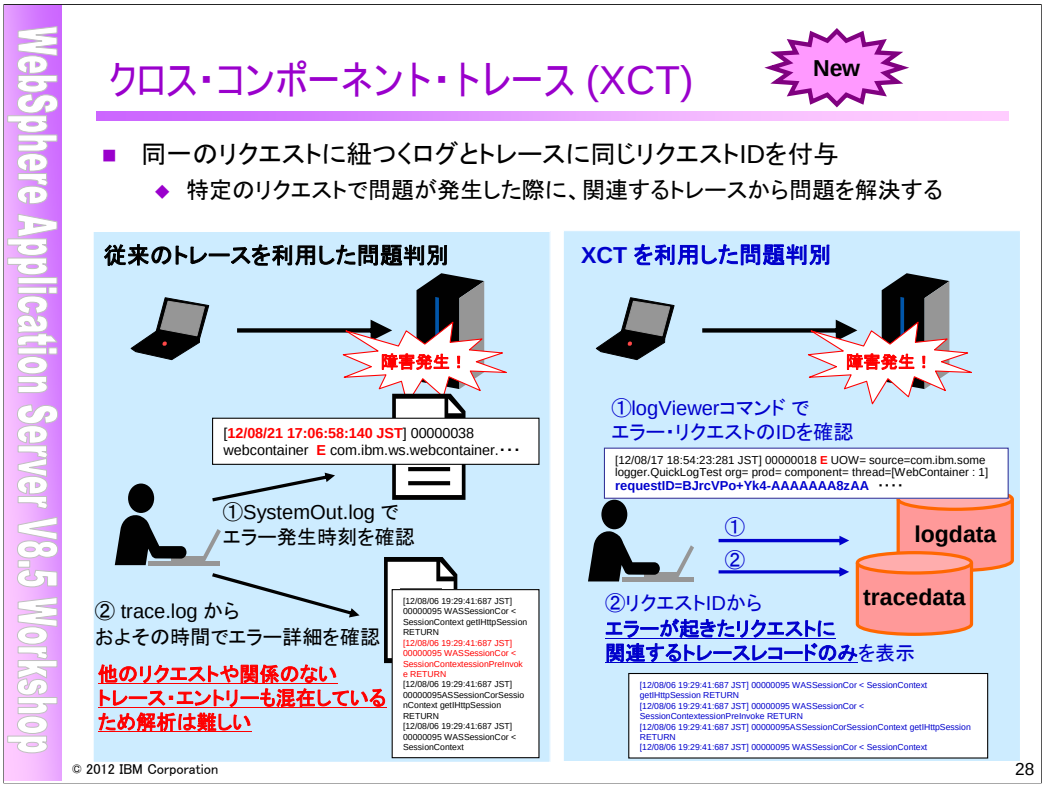


HPELログを閲覧するための管理コマンドのlogViewerの使用方法です。

logViewerコマンドではオプションを使用してフィルタリングやテキストファイル出力など、自在にログを扱うことができます。上記の各オプションについては、InformationCenterもしくはコマンドのhelpオプションを使用して確認してください。

InformationCenter 「LogViewer コマンド行ツール」

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/rtrb_logviewer.html



クロス・コンポーネント・トレース（XCT）は同一のリクエストに対して関連するログとトレースに同じリクエストIDを付与し、問題発生時の根本原因の究明と解決を手助けします。特にリクエストの処理が複数のスレッドやプロセスをまたがるような複雑なケースで、処理の流れをたどることができます。

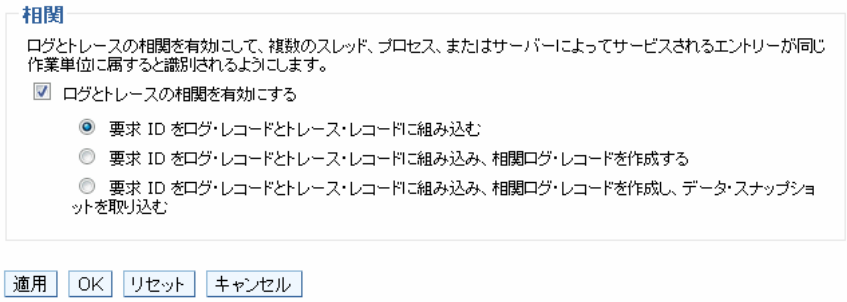
従来の問題判別の手順では、標準ログやエラーログの情報を元に、膨大なトレースの中から関連すると考えられるメッセージやエントリーを探す必要がありました。

XCTを使用可能にしている場合、エラー・メッセージに付与されているリクエストIDを元にHPELのログ・リポジトリーとトレース・リポジトリーをそのリクエストIDに関連するメッセージをフィルタリングできるため、問題の追跡を容易に行うことができます。

WebSphere Application Server V8.5 Workshop

XCT ログの設定方法

- クロス・コンポーネント・トレース (XCT) は HPELモードが前提
  - 「サーバー名」>「診断トレース・サービス」>「ログ詳細レベルの変更」から「ログとトレースの相関を有効にする」にチェックを入れる
  - 「構成」もしくは「ランタイム」で設定



XCTログの設定方法です。

WebSphere Application Server V8.5 Workshop

XCT ログの確認方法

- クロス・コンポーネント・トレース (XCT) は HPELモードが前提
  - スレッドの開始/終了やリクエストがスレッドを移動することを確認するための XCT ログ・レコードをログ・ファイルに追加できる
  - ログの内容は HPEL logViewer コマンド および XCT Log Viewer Tool で確認

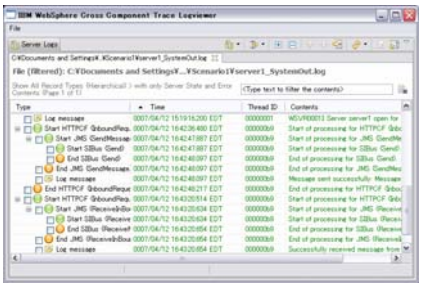
```
[12/08/17 18:54:23:281 JST] 000000a5 XCT  I  BEGIN AACQobp5E6n-AAAAAAAAABP
00000000000-ccccccccc2 HTTPCF(InboundRequest /hitcount RemoteAddress(127.0.0.1)
RequestContext(1083409441))
```

HPEL logViewer コマンド

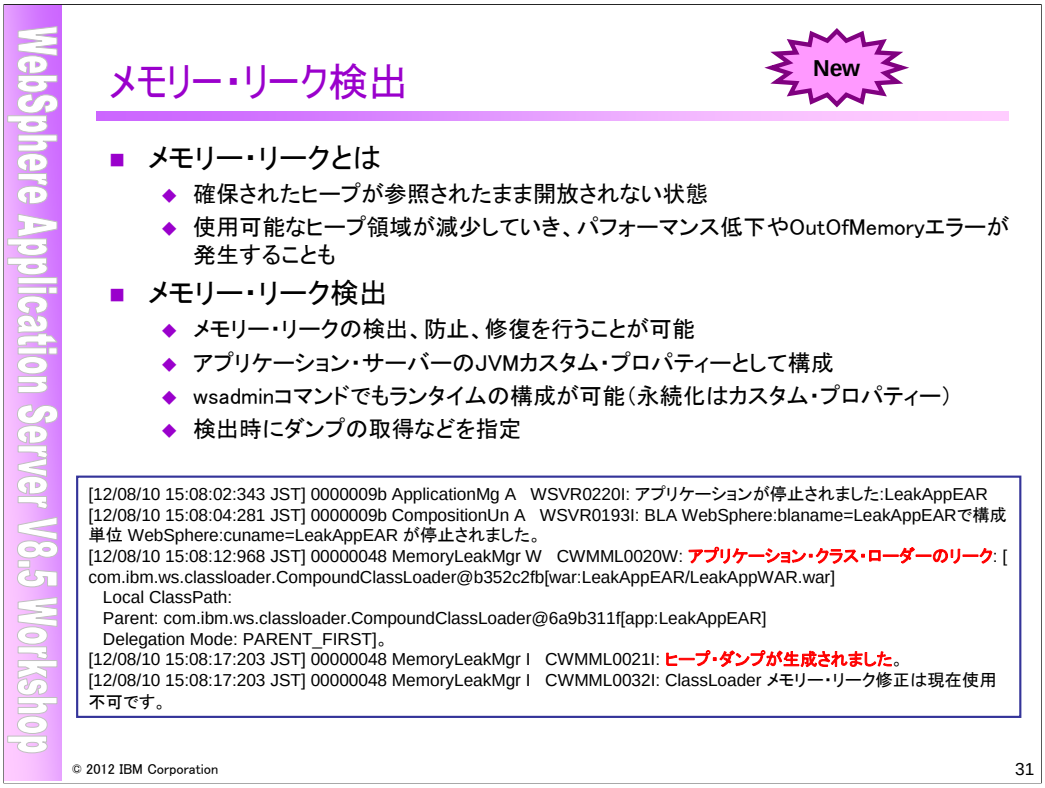
```
C:¥IBM¥WebSphere85¥AppServer¥profiles¥App
Srv01¥bin>logViewer.bat  -includeExtensions
requestID=AACRm/EWltR-AAAAAAAAAAI

C:¥IBM¥WebSphere85¥AppServer¥profiles¥App
Srv01¥logs¥server1 をリポジトリー・ディレクトリーとして使用しています。

*****  略 *****

[12/08/17 19:28:58:656 JST] 000000a6 XCT
I   BEGIN AACRm/EWltR-AAAAAAAAAAI
00000000000-ccc
```

XCT Log Viewer

(IBM Support Assistant アドオン)



XCTログの確認方法です。



メモリー・リーク検出は、稼働中のアプリケーションのメモリー・リークの疑わしいパターンを監視し、検出、防止、修復を行うことが可能な機能です。

メモリー・リークはJREによって、もしくはアプリケーションによって引き起こされます。JREによるメモリー・リークとしては、JREがコンテキスト・クラスローダーを使用してシングルトン・インスタンスをロードする場合に発生し、アプリケーションによるメモリー・リークとしては、独自のThreadLocalクラスを使用する場合、アプリケーションによって作成されたコンテキスト・クラスローダーを使用する場合、静的クラス変数などが考えられます。

WASでとり得るメモリー・リークに対するポリシーを以下に示します。

検出:

メモリー・リークを検出した際に、Warningを出力します。検出ポリシーはデフォルトでは使用不可になっています。

使用可能にする場合は、JVMのカスタム・プロパティーで以下を設定してください。

com.ibm.ws.runtime.component.MemoryLeakConfig.detectAppCLLeaks=true

防止:

防止はデフォルトで有効になっており、JREによって引き起こされるリークに対して有効です。防止措置として、アプリケーション・サーバーのクラス・ローダーがコンテキスト・クラス・ローダーである場合には、サーバー始動時に singleton を初期化 します。

修復:

メモリー・リークに対する予防措置的なアクションで、リークを修正します。これらのアクションは適切なデフォルト値に設定されており、ケースに合わせて調整することができます。

ポリシーとカスタム・プロパティーは次のページをご覧ください。

WebSphere Application Server V8.5 Workshop

【参考】リーク修正ポリシー

リークの原因	修正方法	使用可能にするため、および制御するための Java Virtual Machine プロパティー
Threadlocal	構成可能な期間、スレッド・プールでスレッドを更新します。スレッドをプールから取り出すと、スレッドおよびスレッド・ローカルをガーベッジ・コレクションできます。	• com.ibm.ws.runtime.component.MemoryLeakConfig.clearReferencesThreadLocal • com.ibm.ws.runtime.component.MemoryLeakConfig.renewThreadPoolNames • com.ibm.ws.runtime.component.MemoryLeakConfig.threadPoolRenewalDelayFactor • com.ibm.ws.util.ThreadPool.DEFAULT_THREAD_RENEWAL_DELAY
HttpClient キープアライブ・スレッド	スレッドを親クラス・ローダーに切り替えます。	• com.ibm.ws.runtime.component.MemoryLeakConfig.clearReferencesHttpClientKeepAliveThread
タイマー・スレッド	リフレクションを使用して、スケジュールされている可能性がある新規タスクを停止します。	• com.ibm.ws.runtime.component.MemoryLeakConfig.clearReferencesStopTimerThreads
非 JVM 制御スレッド	スレッドが executor を使用して開始された場合は、executor をシャットダウンするか、スレッドを中断します。	• com.ibm.ws.runtime.component.MemoryLeakConfig.clearReferencesInterruptThreads
JDBC ドライバー	Web アプリケーションによって登録されたが忘れられた JDBC ドライバーを登録抹消します。	• com.ibm.ws.runtime.component.MemoryLeakConfig.preventJreMemoryLeak
ResourceBundle	ResourceBundle キャッシュから、このクラス・ローダー、またはこのクラス・ローダーが親クラス・ローダーであるすべてのクラス・ローダーによってロードされたすべてのバンドルをクリアします。	• com.ibm.ws.runtime.component.MemoryLeakConfig.preventJreMemoryLeak
RMI ターゲット	リフレクションを使用して、sun.rmi.transport.ObjectTable.implTable および sun.rmi.transport.ObjectTable.objTable の値をクリアします。	• com.ibm.ws.runtime.component.MemoryLeakConfig.preventJreMemoryLeak
静的クラス変数	WebSphere Application Server は、アプリケーションまたはモジュールのクラス・ローダーによってロードされたクラスのすべての静的クラス変数の値を無効にします。	• com.ibm.ws.runtime.component.MemoryLeakConfig.clearReferencesStatic com.ibm.ws.runtime.component.MemoryLeakConfig.filterPrefixes

Java Platform, Enterprise Edition アプリケーションでのメモリー・リーク

http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/topic/com.ibm.websphere.nd.multiplatform.doc/ae/ctrb_memleakdetection.html

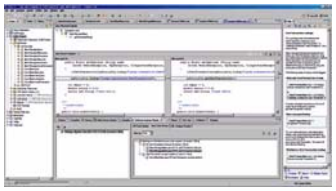
WebSphere Application Server V8.5 Workshop

5. マイグレーション

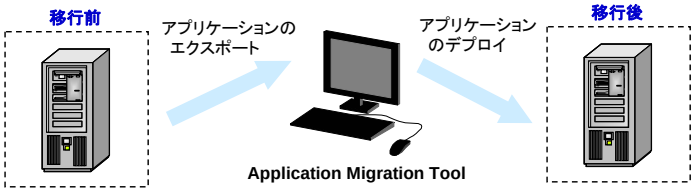
WebSphere Application Server Migration Toolkit V3.5

Update

- 他社ASもしくは古いバージョンのWAS上のアプリケーションを新しいバージョンのWASにマイグレーションするための支援ツール
  - 無償のEclipseのプラグイン
  - 以下のASで稼働するアプリケーションのコード・レビューとクイック・フィックスを提供
    - Apache Tomcat (New)
    - JBoss
    - WebLogic
    - Oracle Application Server
    - WAS V5.1、V6.0、V6.1、V7.0

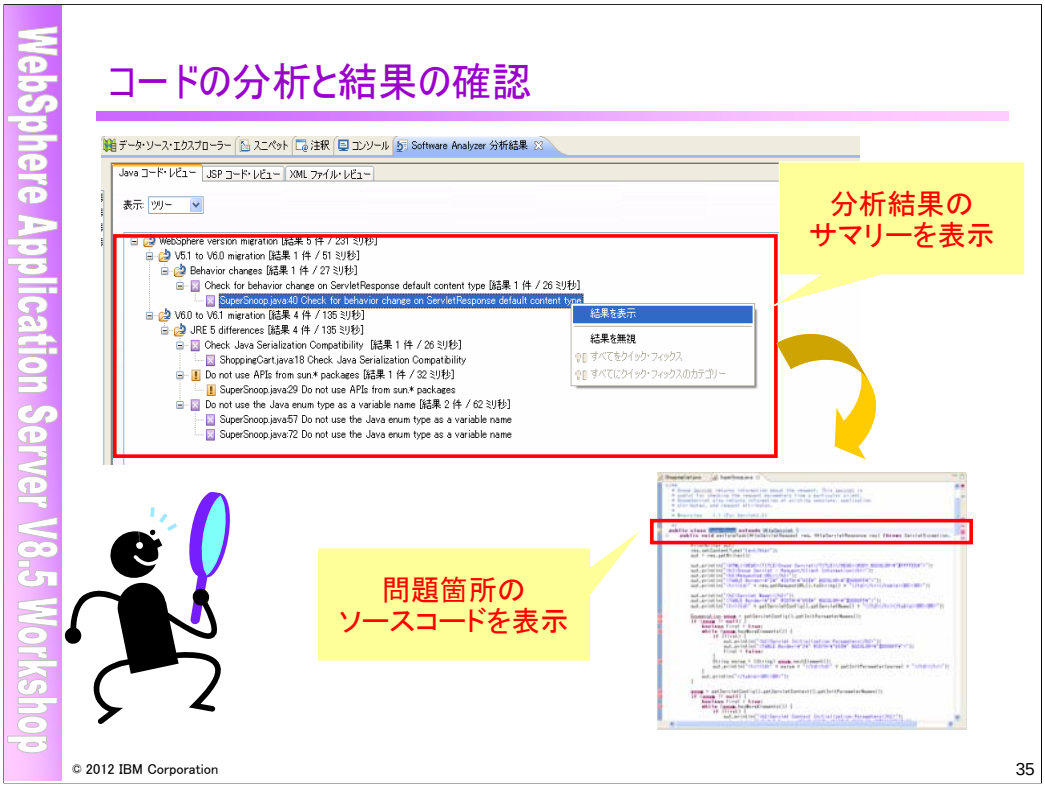


- IBM Rational Software Analyzer がベースのツール
  - Rational Software Analyzer のスキャン機能を利用して，旧環境のアプリケーション・サーバー固有のデータを探し，WASでアプリケーションを実行できるように データを変更



WebSphere Application Server Migration Toolkit は、他社ASや旧バージョンのWAS上のアプリケーションを新しいバージョンのWASにマイグレーションするための支援ツールです。Eclipseもしくは RADのプラグインとして導入し、インポートしたアプリケーションをスキャンして、ツールで定義されているルールに基づき非互換の箇所を検出します。修正すべきコードの内容をクイック・フィックスとして提供しているものもあり、アプリケーションの迅速な移行をサポートします。

2012年9月現在の最新であるV3.5では、新しくTomcatからのマイグレーションをサポートしました。



こちらは設定したルールに従って実行されたアプリケーションの分析結果です。分析結果はサマリーとして表示されますが，ここの分析結果を選択することで修正すべきソースコードを表示することも可能であり，アプリケーションの修正をスムーズに行うことが可能です。

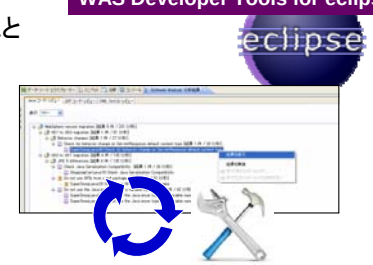
WebSphere Application Server V8.5 Workshop

マイグレーションの効率化を実現

- 例：Tomcatからの移行シナリオ
  - 現行アプリケーション（Tomcat）
  - Eclipse上にApplication Migration ToolkitとWAS Developer Tools を導入
  - ツールによるコードレビュー
クイックフィックスの適用
  - Liberty（WAS for Developer）上でテスト、修正、追加開発
  - 検証環境/本番環境のWASフルプロファイルにデプロイ

Application Migration Toolkit

WAS Developer Tools for eclipse



充実したツール類を活用することで
円滑にアプリケーションのマイグレーションを遂行

例として、Tomcatからの移行例を示します。

Application Migration Toolkit により迅速な移行を実現するのはもちろん、WebSphere Application Server Developer Toolsを使用した追加開発、Libertyプロファイル上での稼働確認により、マイグレーションから追加要件への対応まで効率的に実施することができます。

6. まとめ

まとめ

- 豊富なAPIのサポートによる開発生産性が向上
  - JDK7のサポート
  - OSGi やモバイル、Web2.0 といった先進技術のサポート
- 柔軟かつ便利な構成管理機能
  - 拡張チェックポイント・リポジトリーにより構成の変更管理を容易に実施
- メッセージング機能向上
  - MEのフェイル・オーバー機能の向上
  - データベースロック回避などの障害時対応
- 強力な問題判別機能
  - HPELとXCTで問題を迅速に解決
  - メモリー・リークの検出でプロアクティブにメモリー問題の対応が可能！
- 円滑なマイグレーション
  - WebSphere Application Server Migration Toolkit でコード変換を支援
  - Tomcat / WebLogic / JBoss / 旧WAS からの変換ルールを提供

参考文献

- WebSphere Application Server V8.5 Information Center
  - http://pic.dhe.ibm.com/infocenter/wasinfo/v8r5/index.jsp
- WebSphere Application Server: New Features in V8.5
  - http://www.redbooks.ibm.com/redpieces/abstracts/redp4870.html
- Explore IBM Java 7
  - http://www.ibm.com/developerworks/training/kp/j-kp-java7/index.html
- IBM WebSphere Application Server Migration Toolkit
  - http://www.ibm.com/developerworks/websphere/downloads/migtoolkit/
- WebSphere Application Server Migration Toolkit Version 3.5
  - https://www.ibm.com/developerworks/mydeveloperworks/blogs/wasdev/entry/websphere_application_server_migration_toolkit_version_3_522