JBoss Enterprise Application Platform 5

リリースノート 5.1.1

JBoss Enterprise Application Platform 5.1.1 の使用向け
エディッション 5.1.1

Last Updated: 2017-10-26

JBoss Enterprise Application Platform 5 リリースノート 5.1.1

JBoss Enterprise Application Platform 5.1.1 の使用向け
エディッション 5.1.1

Red Hat Documentation Group

法律上の通知

概要

これらのリリースノートには、JBoss Enterprise Application Platform 5.1.1 に関する重要な情報が含まれています。これらの情報は現在、製品マニュアルに含まれていない可能性があるため、JBoss Enterprise Application Platform 5.1.1をインストールする前に、すべてに目を通すようにしてください。

目次

第1章 はじめに

本リリースノートには、JBoss Enterprise Application Platform 5.1.1 関連の重要な情報が含まれており、新機能、本リリースで修正された問題、その他の既知の問題についての説明がなされています。

1.1. JBOSS ENTERPRISE APPLICATION PLATFORMについて

JBoss Enterprise Application Platformはオープンソースの企業ソフトウェアにおける進化の段階で次に進んでおり、pure Java Platform 上でリッチで性能の高いWeb 2.0 アプリケーションを開発する力強いツールとなっています。

JBoss Seam、Hibernate、CXF Web Services、JBoss Cache、JBoss Messagingなどの最高のオープンソースフレームワークを統合することで、本プラットフォームはオープンソースコミュニティーのイノベーションを有効活用しています。また、JBoss Enterprise Application Platformは、完全なテスト、サポートをRed Hatが行っており、主要なエンタープライズハードウェアやソフトウェア製品の多くで機能するとの認定を受けています。

1.2. 本リリースについて

JBoss Enterprise Application Platform 5.1.1 はマイナーリリースとなっています。マイナーリリースでは、以前のパッチおよび累積パッチ (CP: Cumulative Patch) のコンテンツを集約し、場合によっては新機能を追加することもあります。後続のパッチや累積パッチについては、それ以前のマイナーアップデートがインストールされていることが前提となっています。

Red Hatは、製品のメジャーバージョンのライフサイクルにわたりビジネスレベル相応の努力を行い、マイナーリリース、非同期パッチすべてにおいてAPIレベル互換を維持します。例えば、JBoss Enterprise Application Platform 5.1.1 は、JBoss Enterprise Application Platform 5 の最初のリリースであるJBoss Enterprise Application Platform 5.0.0とのAPIレベル互換を持たせるようにしています。このルールへの例外として考えられるのは、極めて重要なセキュリティー問題に対応した修正などです。

JBoss Enterprise Application Platform 5.1.1 がリリースされており、JBoss Enterprise Application Platform 5 を利用しているお客様はJBoss Enterprise Application Platform 5.1.1へアップデートするようにしてください。

『JBoss Enterprise ミドルウェア製品アップデートおよびサポートの方針』に関する詳細情報は、http://www.redhat.com/security/updates/jboss_notes/ を参照してください。

1.3. 本リリースでの新機能

コンポーネントレベルの新機能に関する情報は3章*新機能* を参照してください。

1.4. 除外、削除、廃止予定のアイテム

定義

除外されるアイテム

製品リリースの機能として組み込まれていないが同製品のオープンソースコンポーネントの一部となっているアイテム

廃止予定のアイテム

今後のリリース (通常、次のメジャーリリース) から削除予定のアイテム

削除されるアイテム

今まで製品リリースに含まれていたがなくなったアイテム。通常、削除前に廃止予定アイテムになります。

本JBoss Enterprise Application Platformはマイナーリリースですが、メジャーリリースと全マイナーリリースの間で互換性が維持されます。つまり、5.x リリースはすべて初期リリースの 5.0.0とバイナリ互換を維持することになります。このような理由から、本リリースで除外、削除、廃止予定のアイテムはありません。

第2章 インストール時の注意点

2.1. 対応の構成

最新の互換構成および認定済み構成に関する表
がhttp://www.jboss.com/products/platforms/application/supportedconfigurations/で入手できます。テスト済みの対応構成に関する情報は、この一覧を参照してください。

2.2. JBOSS ENTERPRISE APPLICATION PLATFORMのインストール

JBoss Enterprise Application Platform設定のインストールや確認に関する説明は、『インストールガイド』 を参照してください。

2.3. デフォルトの起動プロファイル

デフォルトの起動プロファイルは **`default`**で、Java EE 5 の基本的なサーバープロファイルでデフォルトのサービスが一式含まれています。このプロファイルには、Java EE 5 アプリケーションのデプロイ時に必要で、最も頻繁に利用されるサービスが含まれています。ただし、JAXRサービス、IIOPサービスあるいは、いずれのクラスタリングサービスも含まれていません。

`default` プロファイルは実稼働向けあるいは、ロード、負荷、可用性または性能テストを実行するため設計されておりません。

2.4. ソースファイル

ソースのZIPファイル

ftp://ftp.redhat.com/pub/redhat/jbeap/5.1.1/en/source/jboss-eap-src-5.1.1.zip

2.5. 製品サポート

本製品で問題を発見された場合は、カスタマーサポートポータル (https://access.redhat.com) から JBoss サポート案件として起票してください。

第3章 新機能

Enterprise Application Platform 5.1.1 の新機能全般について説明してから、コンポーネント別に見ていきます。

3.1. 全般

HornetQ

HornetQ は JBoss Enterprise Application Platform 5.1.1においてテクノロジープレビュー版 (TP: Technology Preview) として提供されています。カスタマーポータルにあるスタンドアローンのZIPを使い、JBoss Enterprise Application Platform 5.1.1とは別にHornetQをダウンロードしインストールする必要があります。詳細情報については『HornetQ ユーザーガイド』 を参照してください。

Red Hat Enterprise Linux 6 のRPM

JBoss Enterprise Application Platform 5.1.1 は、認定済みのRPMとして利用可能となっており、Red Hat Enterprise Linux 6 向けの認定済みインストレーション Zip ファイルに追加されます。

新規のデータベース認定

以下のデータベースおよびJDBCの組み合わせがJBoss Enterprise Application Platform 5.1.1で機能する認定を受けています。

PostgreSQL 8.4.x

JDBC4 PostgreSQL ドライバーの Version 8.4-702との互換認定

Sybase ASE 15.5

Sybase jConnect JDBC ドライバ v7 (Build 26502/EBF17993) との互換認定

Microsoft SQL Server 2008 R2

Microsoft SQL Server JDBC Driver 3.0.1301.101との互換認定

Apache httpd コネクター

Apache httpd コネクターの **mod_cluster** および **mod_jk**は、Red Hat Enterprise Linux 6に同梱されているApache httpd バージョンと互換があるとの認定を受けています。詳細情報については、https://issues.jboss.org/browse/JBPAPP-6195 を参照してください。

3.2. コンポーネント別

JCA

JBPAPP-4539

配備記述子で**<min-pool-size>** と **<max-pool-size>** に対しプロパティの代入ができませんでしたが、この問題は修正され、配備記述子にてどのプロパティに対してもプロパティの代入ができるようになりました。

Hibernate

JBPAPP-5022

プロパティ変数**hibernate.hbm2ddl.auto** が**create** あるいは **update**に設定されている場合、**SchemaUpdate** クラスは自動的にインデックスを作成するようになりました。以前は、この変数が**create**に設定されている場合のみ更新されるとの動作でした。

ネーミング

JBPAPP-5909

ORBGracefulShutdown と呼ばれる新しいプロパティが**iiop-service.xml** ファイルに追加されました。デフォルト値は**false**で、以前の動作を引き継ぎます。値が**true**の場合、シャットダウンする前に保留の呼び出すが完了するまでORBが待機するようになります。

Seam

JBPAPP-4771

複数の画像が同じページで利用されている場合、**s:graphicImage** を使い画像の変換を行っていましたが、ページのロードが遅いことがありました。これは**s:graphicImage** をキャッシュ可能にすることで解決されています。

JBPAPP-5766

以前は、**s:graphicImage**はBMP画像への対応がありませんでした。コンテンツタイプ**image/bmp**として、BMP画像も対応されるようになっています。

セキュリティ

JBPAPP-5882

キーストアあるいはトラストストアのパスワードをJaasSecurityDomain で公開するのはセキュアではありません。JSDから直接プライベートキーと証明書を取得するメソッドが2種類追加されていますので、外部のコンポーネントにより、これらを利用することが可能です。これらのメソッドは、**getKey** と **getCertificate**で、**getKey** メソッドはセキュリティトークンの提示が必要になります。

JBPAPP-5568

JBossWS の設定管理レイヤーはJaasSecurityDomain JNDI 名を参照することができず、セキュリティドメインにより通常公開されるキーストアやトランストストアにJBossWS セキュリティレイヤーがアクセスできなくなっていました。JBossWS 管理およびセキュリティレイヤーが更新され、代わりとなる**jboss-wsse-server.xml** や **jboss-wsse-client.xml** ファイルが認可されました。この代わりとなるXMLファイルにより、JaasSecurityDomain JNDI 名前を特定することが可能です。セキュリティドメインのキーストアやトラストストアが公開されることで、JBossWS トランザクションのセキュリティが強化されています。

JBPAPP-5578

キーストアあるいはトラストストアのパスワードをJaasSecurityDomain で公開するのはセキュアではありません。JSDから直接プライベートキーと証明書を取得するオプションが2つ追加されていますので、外部のコンポーネントにより、これらを利用することが可能です。これらのメソッドは、**getKey** と **getCertificate**で、**getKey** メソッドはセキュリティトークンの提示が必要になります。

JBPAPP-5434

JaasSecurity ドメインには**clientAlias** と **serverAlias** のオプションが含まれるようになりました。**keyStoreAlias**と同じ方法でこれらを設定することができます。

第4章 修正済の問題

Enterprise Application Platform 5.1.1 で修正された問題をコンポーネント別に一覧にまとめています。

ビルド

JBPAPP-4621

以下のような不必要なファイルがEnterprise Web Platform のインストレーションに含まれていました。これらはEnterprise Application Platformにのみ関連するものです。

- **`jboss-ewp-5.1/jboss-as-web/server/default/conf/props/messaging-users.properties`**
- **`jboss-ewp-5.1/jboss-as-web/server/default/conf/props/messaging-roles.properties`**

これらのファイルはインストールの手順から削除されています。

JBPAPP-4970

JBoss Enterprise Application Platform 5 で配信されていた Seam のバージョンには、Hyper Structured Query Language Database (HSQLDB)へ必要のない依存関係が存在していましたが、これは削除されています。

JBPAPP-5155

JBoss Native Zipは、zip アーカイブの中に sha256sum の値が含まれていませんでした。この値は別のファイルに含まれており、ネーティブの zip アーカイブと別でダウンロードする必要がありました。これにより、リリースプロセスで問題を引き起こし、正確な sha256sum リストを保持できなくなっていました。現在はsha256sum の値は zip アーカイブに含まれています。

クラスタリング

JBPAPP-3549

マルチキャストアドレス (**`mcast_addr`**) が誤って設定された場合、ログに表示される警告に古いURLが含まれていましたが、このURLは更新されています。

JBPAPP-3795

`org.jboss.system.server.profileservice.repository.clustered.local.file.AbstractFileWriteAction` クラスにはメンバー変数**`tempFile`**が含まれており、これが**`getTempFile()`**にて新規作成ファイルに設定されています。**`writeBytes()`**から呼び出される**`getOutputStream()`**がこのファイルを呼び出すのですが、このファイルにデータがある場合しか**`writeBytes`** が呼び出されません。

このファイルが空の場合、**`tempFile`** が設定されず **`AbstractFileWriteAction.modifyTarget`** が null の **`File`** パラメーターを**`FileUtil.localMove`**に渡し、 **`NullPointerException`** が発生します。

`AbstractFileWriteAction.modifyTarget` は**`getTempFile()`**を呼び出すようになり、**`tempFile`**を直接使用するのではなく、随時ファイルを作成します。

JBPAPP-4947

JGroups の **encrypt** クラスには Cipher ルーチンのスレッド化に関する問題がありましたが、この問題は対処されています。

JBPAPP-5171

ソフトウェアのエラーが原因で、ログインページにサービスを提供するプロセス、あるいは**POST** を **j_security_check** へ処理しても**ClusteredSessionValve**への呼び出しが行われませんでした。**ClusteredSessionValve**コードは、データベースに保存することでセッションの複製をトリガーするリクエストパイプラインの一部です。**FormAuthenticator** は、直接ログインページに送られ自身で**j_security_check** を処理していました。

結果、ログインページにサービスを提供する前に作成されたこのセッションは永続化されませんでした。認証が完了し、元にあったURLへのリクエストが入ってくると、このセッションマネージャは分散ストアをチェックし、ローカルセッションのコピーの有効期限が切れていないか確認していました。また、**DataSourcePersistentManager**で、**VersionBasedOutdatedSessionChecker**に対してこのチェックを行いますが、**VersionBasedOutdatedSessionChecker**はデータベースにこのセッションが見つからないため、true を返し、この戻り値**true** が原因で、セッションマネージャはローカルセッションを利用することができませんでした。さらに、永続ストアから読み込みを試行しますが、永続ストアにも存在しないため、新規セッションが作成されてしまっていました。

この問題に対応するため、**VersionBasedOutdatedSessionChecker** は、永続化されたセッションが見つからない場合セッションの**getLastReplicated** メソッドを確認するようになりました。値が 0 の場合は、セッションは新しく永続化されていないため、期限が切れることはありません。そのため、この場合は false を返します。

JBPAPP-5406

スーパークラスコンストラクタの操作順が原因でリモートクライアントの作成時に誤ったサブシステムが設定されており、**NullPointerExceptions** と **ClassCastExceptions**が発生していました。正しいサブシステムを設定後クライアントが初期化されるようにコンストラクタが変更されました。

JBPAPP-5843

あるグループにおいて全チャネルで共有されているタイマーは、そのチャネルの 1 つが閉じられても停止されませんでした。チャネルがグループから退出した時点で、OOBメッセージが渡され、チャネルが閉じられた後、メッセージがRetransmitter に追加されました。これは、Retransmitter に追加されたメッセージが**Retransmitter.reset()** が完了した後も引き続きリクエストを出しつづけるため発生していました。ノードがクラスタから離れると同時にノードからのメッセージが処理される際に再送リクエストが出される状態を修正することで、この問題は解決されています。

JBPAPP-5844

Microsoft Windowsでは、ネットワークケーブルを抜き差しするなど、ネットワークインターフェースが中断されるとJGroups が**NoRouteToHostException** タイプの例外をメッセージ送信を試行する度に受け取っていました。これは、サーバーが再起動されるまで継続されていましたが、この問題は、ソケットをインターフェースに再構築するというJGroupsの新しい動作により、JGroups がメッセージを再送できるようになり解決しています。

JBPAPP-5851

以前は、**KeyGenerator**ルックアップのみが、ENCRYPT アルゴリズムの**asym_provider** および **sym_provider** セキュリティプロバイダーオプションを使っていましたが。これらのオプションが **Cipher** ルックアップにも実装されています。

JBPAPP-5855

JGroups **FD_SOCK** プロトコルは、ソケットをベースにする障害検知プロトコルです。**FD_SOCK** TCP 接続のクライアント側が特定の IP アドレスとバインドしませんでした。個別アプリケーションぞれぞれに固有のファイアウォール接続を開こうとすると、アプリケーション間で必要のないクロストークが行われていました。**client_bind_port**、 **port_range**、**bind_addr** が **FD_SOCK**に追加され、特定のIPアドレスおよび/あるいはポート範囲への送信接続をバインドできるようになっています。

JBPAPP-5900

JGroups **FLUSH** プロトコルの競合状態が原因で、マスターノードがクラスターから退出時にメッセージが誤った順番に処理されたため、(新しいマスターで) 新規ビューを送信する前にマスターノードが退出していました。そのため、他のノードがこのマスターが存在しない点を検出しない、あるいは障害検知プロトコルによりマスターが存在しない点を検出するまで新しいマスターを選択していました。メッセージの順番が修正され、競合状態が発生なくなりました。

JBPAPP-5912

非同期シリアル化するよう構成されたJBoss Cache インスタンスがクラスタからシュンされると、それ以降の複製がキャッシュが再度立ち上がるまでの間、失敗していました。原因は、メソッドと親クラスのネーミングで矛盾が起こっていたためです。これは、**CommandAwareRpcDispatcher.stop()** メソッドの名称を**stopDispatcher()**に変更することで解決されています。

JBPAPP-6011

バインディングマネージャのJGroups 設定は、Enterprise Application Platform 5.1の JGroups 設定に追加されたシステムプロパティを実装するようになりました。さらにハードコード化されたポートが削除され、デフォルトのマルチキャストアドレスが同プラットフォームのバージョン 5.0 で使われているものと同じ値が設定されています。

コンソール

JBPAPP-3928

ユーザーが Admin Console ログインページに入った後ログインするまで長時間待機すると、セッションがタイムアウトし例外が発生していました。しかし、分かりやすいタイムアウトメッセージが表示されてるようになっています。

JBPAPP-4791

トピックあるいはキューが既存のトピックあるいはキューと同名で作成された場合、新規のトピック/キューが古いものを上書きしていました。これは、同名のトピックあるいはキューの作成を試行すると失敗しエラーが生成されるようにすることで解決されました。

JBPAPP-4886

ApplicationServerDiscoveryComponentクラスは、パスが**jar** にハードコード化されているためJBoss 変数を使用しません。JBoss ライブラリの場所とライブラリのJBoss 変数 (**jboss.lib.url**, **jboss.common.lib.url** and **jboss.server.lib.url**)が変更されると、JBoss Admin Console が開きませんでした。この問題はAdmin Console に**BootstrapAction.createPluginContainerConfiguration()**を追加することで解決されました。

文書

JBPAPP-4387

『Seam Reference User Guide』 ソースコードブロックで構文がハイライトされず、ソースコードの解読が困難でした。関連のコードが更新され正しくハイライトされるようになっています。

JBPAPP-4616

『Microcontainer ユーザーガイド』 に、旧版のMaven アーチファクトへの参照がいくつか含まれていました。更新後の情報については、現在のリリースで同梱されている文書を参照してください。

JBPAPP-4958

『Hibernate Core ユーザーガイド』には誤ったPOMファイルに参照がされていました。本リリースの文書ではこの参照は修正されています。

JBPAPP-5223

当ガイドの『JMS クラスタリングノート』 の項にある構成情報が誤っていました。本ガイドは正しい設定例に更新されこの問題は解決しています。

JBPAPP-5584

PostgreSQL versions 8.2 および 8.4 にて **`max_prepared_transactions`** データソースプロパティのデフォルト値を変更すると、XA Transactions が拒否される結果になっていました。『管理設定ガイド』の『添付資料 A.7 PostgreSQL』 にあるXAデータソースの例に重要な警告が追加され要件を強調しています。

JBPAPP-5907

Enterprise Application Platform の以前のバージョンでリリースされた 『セキュリティガイド』 にはデータソースパスワードの説明が抜けていました。本文書には、この手順が現在は含まれています。以下のコマンドを実行することで**`server.password`** ファイルの設定をする方法：

```
java -cp jboss-as/common/lib/jbosssx.jar
org.jboss.security.plugins.FilePassword \
          SALT COUNT MASTER_PASSWORD PASSWORD_FILE
```

JBPAPP-6199

『HTTP コネクターガイド』 はJBoss Enterprise Application Platform 5.1.1の新たなユーザーガイドで、『mod_cluster ユーザーガイド』 および『設定管理ガイド』のHTTPサービスの項をベースに改善され、これらの文書より優先されます。

EJB

JBPAPP-3392

EJB3 クライアントは、後続の呼び出しにて既存のソケット接続を使用せず、呼び出し毎に新規接続を作成し接続が完了すると破棄されていました。遅延を追加し接続が開いた状態を保つことで後続の呼び出しで利用できるようにしました。

JBPAPP-5167

２つのケースで、**`UnifiedClassLoader`** を分離EARのクラスロードの代わりに**`UnifiedClassLoader`** を使っていました。1番目は、EJB2 Entity Bean が分離EAR内でパッケージされ、さらにEJBクライアントが同じ分離EARでパッケージされた場合に発生していました。EJB Entity Bean がパッシベーションされた場合、**`ClassNotFound`** の例外がスローされました。これは、EARの分離クラスローダーではなく、**`UnifiedClassLoader`** が利用されることで起こっていました。2番目は、デプロイされた EJB2 Entity Bean に**`UnifiedClassLoader`**にあるインター

フェースと同名のローカルインターフェースがあることが原因で**ClassCastException**が発生していました。両ケースで、正しいクラスローダーを利用することで、エラーが発生しなくなりました。

JBPAPP-5476

org.jboss.ejb.plugins.SecurityInterceptor のバグが原因で、元々runasでデプロイされていないステートレスセッションEJBにてrunas-identity context メソッドの呼び出し設定ができませんでした。EJBに送信された認証コンテキストのアイデンティティを呼び出しが利用するため、EJBがrunasでデプロイされたように呼び出しが実行されてしまいました。この動作を止める唯一の方法は、サーバーを再起動することです。**SecurityInterceptor** は、元のEJBにあるrun-as ロールを参照し、EJBからの呼び出しにrunAsRole が利用し宣言的なセキュリティチェックができるようにしています。

JBPAPP-5618

EJB Timer サービスが**GeneralPurposeDatabasePersistencePlugin**を利用し、**CachedConnectionManager**をデバッグに設定していると、以下の警告がサーバーログに記録されます。

```
WARN [org.jboss.resource.adapter.jdbc.WrappedConnection] Closing a
result set you left open! Please close it yourself.
```

これは、 **GeneralPurposeDatabasePersistencePlugin** が内部で利用した**ResultSet**を終了しないため発生し、**CachedConnectionManager**は**ResultSet** を終了し、正しくユーザーにエラー通知を行っていました。**ResultSet**が終了するように**GeneralPurposeDatabasePersistencePlugin**が修正され、警告が表示されなくなりました。

JBPAPP-6102

JDK6 のアップデート19 以降でアプリケーションが**java.io.File** オブジェクトを送信あるいは受信した場合、シリアル化の例外が発生する可能性があります。また、この例外は、**java.io.File**を含むステートフルセッションでクラスタリングを利用していると発生します。

この例外を阻止するために新たなプロパティが追加されました。相互通信を行う全 JBoss インスタンスに対して、このシステムプロパティ**-Dorg.jboss.serial.SYNC_SERIALIZATION_BINARY_FORMATS=true**を設定します。

Hibernate

JBPAPP-4175

Hibernate が**ResultTransformer**を使いキャッシュ可能なクエリを実行すると、**ResultTransformer**適用後、結果をキャッシュしようとしていました。しかし、Hibernate が解読できないようにデータが変更される可能性があり、この場合、結果をキャッシュしようとすると、**ClassCastException**が発生する可能性があります。

この問題は、APIに新たなクラスを導入することで解決されています。

- **org.hibernate.transform.AliasedTupleSubsetResultTransformer**
- **org.hibernate.transform.CacheableResultTransformer**

- **`org.hibernate.transform.TupleSubsetResultTransformer`**

詳細については Hibernate 向けの Javadoc を参照してください。

JBPAPP-4738

`org.hibernate.dialect.SQLServer2008Dialect` の方言が Hibernate に追加され、Microsoft SQL Server version 2008で導入されたMicrosoft SQL 方言の変更に対応しています。

JBPAPP-4895

二次キャッシュが有効で、**`insert()`**の直後に**`refresh()`** メソッドが呼び出されると、エンティティが二次キャッシュに挿入されました。しかし、**`refresh()`** が問題なくコミットされても、キャッシュしたデータは自動的に削除されませんでした。これは、**`refresh()`**がエンティティのステータスを追跡しなかったためです。**`refresh()`** がエンティティーのステータスを追跡し、コミットに成功するとキャッシュされたデータが削除されるようになっています。

JBPAPP-4904

JBoss Marshalling は**`org.hibernate.impl.SessionImpl`**のシリアル化ロジックに不備があるため、セッションのシリアル化に失敗していました。このクラスは、http://java.sun.com/javase/6/docs/platform/serialization/spec/output.html#86に記載されているJava Serialization 仕様に従うようになり、シリアル化が正しく予想どおりに行われるようになっています。

JBPAPP-4905

反映することでenum 値を取得するのは高価なため、**`org.hibernate.type.EnumType`**は静的マップを利用してenum 値をキャッシュしていました。しかし、この実装は、状況によっては（特にアプリケーションサーバーでHibernate を利用時など）メモリリークを引き起こす可能性がありました。これはキャッシュ化オブジェクトがガーベッジコレクションを行わないため、キャッシュの代わりとなっているMap が無限に増大し続けるためです。

グローバルのenum キャッシュを利用する代わりに、Hibernate はenum 値をキャッシュする**`EnumType`** インスタンス毎に一時アレイを使うようになりました。こうすることでパフォーマンスが向上しメモリリークがなくなります。

JBPAPP-4926

割り当てられたPKと親テーブルをリンクするオートインクリメント Primary Key (PK) テーブルを**`cascade-save`** 操作が誤処理していました。修正を行うことで**`cascade-save`** 操作がテーブル間の親子関係を処理できるようになりました。

JBPAPP-5394

`QueryPlanCache` の問題がバインドされていない**`SoftLimitMRUCache`**とのソフト参照が原因でメモリリークを引き起こしていることが判明しました。最終的に、**`SoftLimitMRUCache`** は、主な stop-the-world ガベージコレクションが**`SoftLimitMRUCache`** のソフト参照を消去する必要が出てくるまで充填していました。

キャッシュにあるバインドされていないソフト参照および強参照の量を制限することで解決します。LRUポリシーを使うか、ソフト参照の場合はGCからのメモリプレッシャーでエントリを削除します。

この問題を解決するための設定オプションが2種類あります。

- **`hibernate.query.plan_cache_max_strong_references`** (デフォルト値 128)

- **hibernate.query.plan_cache_max_soft_references** (デフォルト値 2048)

以前の動作をエミュレーションしたい場合は、**hibernate.query.plan_cache_max_soft_references** をInteger.MAX_VALUEに設定してください。

JBPAPP-5405

JPA 永続化仕様に従い、コレクション宣言に対し**AS** キーワードはオプションとなっています。例は以下の通りです。

```
collection_member_declaration ::= IN (collection_valued_path_expression)
[AS] identification_variable
```

以下の構文例のようにHQL/JPQA は解析例外を引き起こしていました。

```
SELECT o FROM EntityBean AS o, IN (o.items) AS l WHERE l.itemValue = '1'
```

これは、Hibernateが**AS** キーワードを実装しないため発生していました。Hibernateはこの分野の仕様に従い、オプションキーワードが可能になりました。

JBPAPP-5409

Hibernateの **ByteCodeHelper.readByteCode()** は以前は409600バイトまでの制限がありましたが、現在はエンティティサイズにおいて特にサイズの制限なく処理可能です。

JBPAPP-5478

AliasToBeanResultTransformerが**ChainedPropertyAccessor** を利用しており、**ChainedPropertyAccessor** はシリアル化できませんでした。そのため、**AliasToBeanResultTransformer** を使ってキャッシュ可能なクエリがキャッシュの複製中に壊れてしまいました。例えば、このクエリは複製、あるいはディスクへのキャッシュができませんでした。

```
session.createQuery("select foo").setResultTransformer(new
AliasToBeanResultTransformer(SimpleCount.class)).setCacheable(true).list
();
```

AliasToBeanResultTransformerのこの動作は変更され、**AliasToBeanResultTransformer**の作成/シリアル化解除がなされると、キャッシュされた**resultClass**に従い**ChainedPropertyAccessor** インスタンスが再作成されるようになっています。そのため、**AliasToBeanResultTransformer**を使うことでキャッシュ可能なクエリはキャッシュ複製中も継続して機能するようになっています。

JBPAPP-5479

AliasToBeanResultTransformer.hashCodeは**propertyAccessor.hashCode()**依存しており、**PropertyAccessor** の実装は**hashCode()** あるいは **equals()**を上書きしません。そのため、**AliasToBeanResultTransformer**を使った**QueryKey** 関連のキャッシュルックアップは、キャッシュミスを引き起こしていました。

AliasToBeanResultTransformer オブジェクトの2つは同等であるべきで、さらに**resultClass** が同等である場合は同じハッシュコードを利用する必要があります。引数として提示されているエイリアスによりゲッターが決定されるため、同等性について決定する必要があり

ません。セッターを使い**equals()** と **hashCode()**を算出する必要があります。しかし、セッターは、セッターの実装により**hashCode()** あるいは **equals()**を上書きしません。

修正方法ですが、セッターに対応するエイリアスをキャッシュし、このキャッシュを利用し同等性をチェックしています。**AliasToBeanResultTransformer** を使って**QueryKey** 関連のキャッシュルックアップを行っても、キャッシュミスが発生しなくなっています。

JBPAPP-5481

Hibernateが**ResultTransformer**を使ってキャッシュ可能なクエリを実行した場合、その結果をキャッシュしようとしていました。しかし、このデータは変更される可能性があるため、Hibernateは読み込みできませんでした。

以下のすべてが真の場合、 **PropertyAccessException** がスローされていました。

- クエリに**ResultTransformer**がある
- 結果が変換される前にキャッシュされる
- 各結果の値が 1 つである

この問題は、新しいAPI呼び出しを3つ導入することで解決されています。

- **org.hibernate.transform.AliasedTupleSubsetResultTransformer**
- **org.hibernate.transform.CacheableResultTransformer**
- **org.hibernate.transform.TupleSubsetResultTransformer**

詳細はHibernate Javadoc を参照してください。

JBPAPP-5581

Hibernate Core **EntityMetamodel entityNameByInheritenceClassNameMap** フィールドが一貫性なく利用されていました。マップへのput にクラスを利用し、マップからのget にはメソッドを利用していました。特定のエンティティー名でサブクラスのインスタンスを保存すると、例外がスローされていました。**EntityModel** クラスが更新され、**getName()** メソッドではなく、**InheritenceClass** クラスを使うようになっています。このような場合にサブクラスのインスタンスを保存しても、例外がスローされなくなっています。

JBPAPP-5763

同じデータベースにて、**org.hibernate.id.enhanced.OptimizerFactory.PooledOptimizer** を複数のJVMで使うと、シーケンス値が重複して生成されるリスクがありました。これは、メソッドが2度呼び出され2種の値 (value and hiValue)が初期化されるため、発生していました。これらの呼び出し間で別のJVMがシーケンス値をリクエストすると、このシーケンスは2番目の呼び出しに関連付けられました。最初の読み込みでHibernateに**initialValue**を渡す際にオプティマイザの初期化へ二重読み込みを制限することで、この問題は修正されています。

JBPAPP-5765

エンティティがproperty-ref-based キーでコレクションを cacheable と定義すると、コレクションをロードし二次キャッシュエリアに置くことができませんでした。Hibernateはエンティティを所有するのに誤ったキーを使っており、該当のID値ではなく参照されたプロパティ値を使おうとしていました。

Hibernateは、コレクションキーがproperty-refで定義されているかをチェックするようになっています。チェック時にこのキーが存在する場合、コレクション自体に紐付けられたオーナーインスタンスのキーを使います。HibernateはPersisitence Context のオーナーに対してコレクションのオーナーを解決します。

JBPAPP-5814

Mapへのアクセスを同期すると、パフォーマンスが悪化していました。現在、**ReentrantReadWriteLock** と **ConcurrentHashMap**が**Map** の同期を管理しているため、パフォーマンスが改善されています。

JBPAPP-5817

\t\t解析済みのHQL文字列の各要素に対して、ツリーノード毎にうまくフォーマットおよびインデントされた**String**が構築されていました。目的は**log.trace()**の呼び出しをフィードすることでした。ログレベルに関係なく情報が追加されるため、余計なアウトプットが発生し、ログファイルが増大していました。Hibernateはログレベルをチェックするようになり、ログレベルが**TRACE**の時のみフォーマットされたアウトプットを含めるようになりました。

JBPAPP-5898

ScrollableResults JoinFetchで問題を報告した場合、ScrollableResults は、最初の親オブジェクトで正しく子コレクションを設定しますが、2つ目の親オブジェクト以降、子コレクションの1つ目の要素のみが含まれていました。

ScrollableResults ロジックが改善され順番に読み込むようになっています。新たな親に遭遇すると、ScrollableResults は以前の親の行がすべて処理済みであると想定します。

重要

この修正は、結果が絶えず分類されている必要があります。データベースによっては分類データ (H2 など) を返さないものもあるため、ScrollableResults の結果順序に対して「order by」を明示的に適用する必要があります。

IOPP

JBPAPP-3134

jboss-log4j.xmlが更新され、 **jacorb.config** ログレベルが含められるようになっています。優先順位が**ERROR** に設定され、サーバー起動時に必要のないメッセージがコンソールに出力されないようになりました。

インストーラー

JBPAPP-2724

グラフィックインストーラーでは、ディレクトリの選択ダイアログは、インストールの最初にユーザーがリクエストした言語ではなく、インストーラーが動作する環境にて指定されている言語が使われていました。インストーラーは更新され、ディレクトリの選択画面ではインストールプロセスの最初にユーザーが選択した言語を使われるようになりました。

JBPAPP-4262

Solaris では Enterprise Application Platform のインストール時に root パスワードが聞かれていました。これは、インストーラーがオプションのショートカット (シンボリックリンク) を作成時に権限

の問題が原因で発生していました。この問題は、Solaris プラットフォームのインストーラー経由でショートカット作成できる機能を削除することで、改善されています。必要であれば、手動でシンボリックリンクを継続して作成することも可能です。

JBPAPP-5049

以前はJBoss Enterprise Application Platform のインストール時に、同Platform のインストール先のオペレーティングシステムに制限されるのではなく、全オペレーティングシステムに対するスクリプトが含まれていました。インストーラーが更新され、適切なスクリプトのみが本プラットフォームのbin/ ディレクトリに含まれるようになっています。Linux および UNIX システムに対してはShellスクリプトがインストールされ、Microsoft の環境にはバッチスクリプトが含まれます。

JBPAPP-5087

Red Hatの文書ページがhttp://www.redhat.com/docs から http://docs.redhat.com/へ移動されました。全文書にてこのリンクが修正されています。

JBPAPP-5110

グラフィックインストーラーでは、コンソールや呼び出し側のセキュリティを確保するため、カスタムの JAAS セキュリティドメインを指定することが可能です。しかし、以前はTomcatコンソールは JAAS セキュリティドメイン**`jmx-console`**が存在しない場合でも、常にこのドメインでセキュリティが追加されていました。

この問題は修正され、Tomcatコンソールはインストール時に指定したセキュリティドメインでセキュリティが確保されるようになっています。

JBPAPP-5116

カスタムのJAAS セキュリティドメインがグラフィックインストールのプロセス中に作成されると、カスタムの JAAS セキュリティドメインが継続して**`jmx-console-users.properties`** と **`jmx-console-roles.properties`**を利用していました。この誤った動作は変更され、*NAME*と呼ばれるカスタムのJAAS セキュリティドメインは、代わりに***`NAME`*`-users.properties`** と ***`NAME`*`-roles.properties`** を利用するようになっています。

JBPAPP-5129

Seam の例は**`build.properties`** ファイルのパスデリミターの問題で、**`ant explode`** を使うと正しくデプロイされませんでした。これにより、Seam 例のデプロイメントに失敗していました。**`build.properties`** ファイルは現在、正しいパスデリミターを利用するようになっています。

JBPAPP-5132

Microsoft Windows での Seam例では、スタートメニューの**`Deploy Hotel Booking Seam Demo`** のショートカットがJBoss Enterprise Application Platform設定の 1 つ上のディレクトリを参照していました。これが原因で、 **`NoClassDefFoundError`** タイプのエラーがログに表示され、この例がデプロイされませんでいた。このパスは修正され、例もショートカットからデプロイされるようになっています。

JBPAPP-5435

グラフィックインストーラーのセキュリティ設定画面には、誤った文書ポータルやユーザー文書へのリンクが含まれていましたが、すべて修正されています。

JCA

JBPAPP-4964

JBAS-5929の修正によりもたらされた再発バグが原因で、JCAプールがチェックアウト接続を検索せずにシャットダウンされることがありました。この問題は修正され、JCAプールはシャットダウン前にチェックアウト接続を検索するようになっています。

JBPAPP-5119

XAManagedConnectionFactory クラスはプロパティの取得、設定に**is** methodName を形成することができませんでした。これが原因で、プロパティがboolean タイプの場合XA DataSources で問題が発生していました。現在、**XAManagedConnectionFactory** は**is** フォーマットが可能となり、エラーなしにboolean タイプを処理するようになっています。

JBPAPP-5374

NewMsgsWorker#run() クラスの**Thread.sleep** メソッドにマイナスのタイムアウト値が渡されると、**Mail**メッセージ駆動 bean が新規メッセージの確認をしなくなりました。マイナスのタイムアウト値を許可しないようにすることで、これは修正されました。

JBPAPP-5596

JCAコードが正しくマルチスレッドアクセスを処理しませんでした。いくつか考えられた原因はJBC操作とJBossTS トランザクション reaper スレッド関連のデッドロックです。JCAコードは修正されておりこれらの問題が修正されているため、JCAでマルチスレッドアクセスが正しく機能するはずです。

メタデータ

JBPAPP-4041

ServiceMetaDataParser.parseValueFactoryParameter() メソッドは**<parameter>** 要素の最初の子のみを考慮していました。**<null/>** 要素がキャリッジリターン (CR) で囲まれている場合、ノードはテキスト値として処理され、それ以外の場所でパラメーターが正しく代入されませんでした。以下の例は失敗していましたが、今は正しく機能するようになっています。

```
<parameter>
   <null/>
</parameter>
```

JMX

JBPAPP-5690

JMXOpsAccessControlFilter クラスは、ロールが**DeploymentRolesMappingProvider**で設定されている場合、ユーザーに対するロールマッピングを保持しませんでした。フィルタークラスが修正され、マッピングが予想どおりに保持されるようになっています。

メッセージング

JBPAPP-5241

デバッグコードが誤って**ChannelSupport.afterRecoveryEx()**に残されており、このコードがログファイルに表示されていました。これは削除され、ログファイルをさらに明確かつ簡単に解読できるようにしています。

JBPAPP-5280

MessageSucker はクラスター内の別メンバーとの間でメッセージを移行します。**onMessage** ルーチンは、他のタスクが存在する中、メッセージをローカルキューに配信しようとします。配信に失敗すると、メッセージが紛失したように表示されていました。これらのメッセージはデータベースに存在しているにも拘らず、再送信するのが困難でした。

これは、JBoss Messaging コンポーネントの問題で、このコンポーネント内で修正されました。この修正は複雑でJBMESSAGING-1822にて説明されていますが、手短に言うと信頼性の新規レイヤーがメッセージの配信ロジックに追加されました。

JBPAPP-5415

トランザクションのコミット中にセッションが終了すると、競合状態が発生する可能性がありました。この場合、例外がスローされました。コミットされたメッセージがキャンセルされるようにチェックが追加されています。

mod_cluster

JBPAPP-3463

コネクターの**pause()** メソッドの問題が原因で、アプリケーションのアンデプロイ時に競合状態が発生する可能性がありました。アンデプロイの通知を受け取る前にmod_cluster によりセッションがアプリケーションサーバーへ送られると、**error 503 - This application is not currently available**が発生する可能性がありました。このメソッドは修正され、これらの問題に対応しています。

JBPAPP-5048

mod_cluster マネージャーのステータスページがフェールオーバー時に更新されないため、ワーカーノードに問題が発生した後も有効かつ利用可能であると表示されていました。このステータスページはノードに問題が発生すると、更新されるようになっています。

JBPAPP-5237

/webappなどのサブディレクトリにデプロイされているアプリケーションでフェールオーバーの設定がされているサーバーや**/**でデプロイされている別のアプリケーションが原因で時折、エラーが発生することがありました。これは、フェールオーバーしたアプリケーションが**/webapp**の代わりに**/**をデプロイしようとしたためでした。この状況は修正され、予想通りにフェールオーバーが起こるようになっています。

JBPAPP-5283

クラスターのエイリアスの最大長が**40** から **64**へ増やされ長いエイリアスに対応するようになっています。

JBPAPP-5315

クラスタ化されたノードは、ワーカーの再試行タイムアウト (６０秒に設定) の後にのみ終了していました。これにより、この期間に正しくノードが終了する可能性が高い場合、ノードが６０秒間エラー状態であったとのエラーメッセージが出されていました。約１０秒おきにノードが配信するSTATUS メッセージをノードが受信すると、終了するようになっています。

JBPAPP-5511

JBoss Enterprise Application Platformで**ROOT** コンテキストがデプロイされていない場合、**mod_rewrite** はhttpd のルート (/) から再書き込みができませんでした。これは、mod_clusterが**mod_rewrite**提示のURIではなく、元のURIを使おうとしていたため発生ていました。**ROOT** コン

テキストが同プラットフォームにデプロイされるか否かに拘らず、再書き込みのルールが機能するようになっています。

その他

JBPAPP-3083

AopC が利用中の場合、**ArrayIndexOutOfBoundsException**タイプの例外でレポート生成が失敗していました。レポートが予想どおりに生成されるように解決されており、例外もスローされなくなっています。

JBPAPP-3308

分離デプロイメントでアプリケーションサーバーが Timer に紐付けられたinfo オブジェクトをデシリアル化しようとすると、**ClassNotFoundException** がスローされました。これは、誤ったクラスローダー (**threadContextClassLoader**) を使ってオブジェクトをデシリアル化したため発生しました。このバグは修正され、正しいクラスローダーが使用されるようになっています。

JBPAPP-5148

以前は正しいパスかについてデプロイメントファイル名がチェックされませんでした。不完全に作成されたファイル名が原因で予期せぬかたちでファイルが削除あるいは変更されることがありました。デプロイメントファイルは正しいパスかをチェックされるようになり、不正パスが使われている場合は例外がスローされるようになっています。

JBPAPP-5232

long がintにキャストされているキャスティングの問題が原因で、1 ミリ秒などの短時間**org.jboss.varia.scheduler.ScheduleManager** を利用すると負の値が繰り返される場合がありました。これはintではなくlong を使うことで修正されています。

プロファイルサービス

JBPAPP-2698

Profile サービスのバグが原因で、デプロイメントが非常に早く開始し停止された場合、デプロイメントのステータスが誤ってレポートされていました。このバグは未だに存在していますが、コンソールに修正が加えられ、短時間の遅延後ステータスを更新するか、ページを更新することでこの問題を回避することができます。通常利用では、ステータスは予想どおりに表示されます。

リモーティング

JBPAPP-5748

WSクライアントから複数のリクエストを送信し、**fastinfoset** を使うことで、**CLOSE_WAIT** のステータスのソケット数が増え誤ってシャットダウンすることがありました。**org.jboss.ws.client.remoting.disconnect.after.use** JVM プロパティを追加することで、クライアントリモートの接続が即座に解除され、この問題が修正されています。このプロパティはデフォルトで有効になっています。無効にすると、**HttpURLConnections** は開いたままになります。

RESTEasy

JBPAPP-2993

spring-hibernate-contacts の例はエラーで失敗していました。

```
java.lang.IllegalArgumentException: object is not an instance of
declaring class
```

これは、**ContactServiceImpl** クラスの **getContactById(@PathParam("id") Long id)** メソッドが @GET アノテーションに存在しないため、発生していました。このエラーは修正され、コード例は予想どおり機能するようになっています。

スクリプトおよびコマンド

JBPAPP-5403

service.batを使いサーバーを開始すると、オプションが**run.conf.bat**から継承されませんでした。これは、**service.bat** が**JAVA_OPTS** をプレースホルダーの値へ設定し、チェックに不具合があることが原因で**run.conf.bat**に設定された値が上書きされないため、発生していました。この問題は修正され、**service.bat**でサーバーを起動すると**run.conf.bat**に設定されたオプションが利用されるようになっています。

Seam

JBPAPP-3520

JBPMの**JpdlParser** にはハードコード化されたXSDのファイル名が含まれており、インターネットの場所を参照しクラスパスのXSDを無視していました。これらのインターネットの場所が利用不可の場合、問題が発生していました。ハードコード化されたパスが削除され、このクラスパスのXSDが検索されるようになっています。

JBPAPP-4231

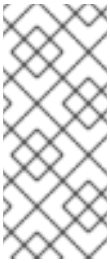
Seam 例をアンデプロイあるいはアンインストールすると**NullPointerException** が発生していました。アプリケーションで問題を引き起こすわけではありませんがエラーが間違っていました。この問題は、『nestedbooking』、『dvdstore』、『itext』、『excel』 の例で発生していましたが、『ui』 の例では発生しませんでした。原因はEJB3 デプロイヤのバグで、このコンポーネントをアップグレードすることで当問題が解決されています。結果、上記の例で**NullPointerException** は発生しなくなりました。

JBPAPP-4508

<h:dataTable>内の**<s:fileUpload>**でファイルをアップロードすると、エラーが発生しファイル名が同じであるのにコンテンツが誤っていました。これはローカルの値が**getLocalValue()** メソッドにより返されるため発生していました。この問題は、メソッドを修正することで修正され、ファイル名とコンテンツの問題は発生しなくなりました。

JBPAPP-4582

IBM JVM v1.6 はランタイム時にUNKNOWNとなっているアノテーションを処理しませんでした。これは、Chatroom の例で問題を引き起こしていました。IBM JVM バージョン1.6.0 (SR9 FP1) 側で根本的な問題が解決されました。



注記

チャットルームの例をバージョン1.6.0 (SR9 FP1)以前のIBM JVM で機能させるには、http://repository.jboss.org/maven2/net/jcip/jcip-annotations/1.0/jcip-annotations-1.0.jar を*JBOSS_HOME*/server/*PROFILE*/libへコピーします。

指定のファイルを追加した後、サーバーを再起動します。

JBPAPP-5013

Seampay の例は例で出てくる最初の支払いと2番目の支払いの期間に問題がありました。処理の早いハードウェア、Java 6 ランタイム、Windows Server 2003 の組み合わせを利用している場合、支払い間の遅延設定が短すぎました。この例は変更され 2 つの支払いの間隔を伸ばし、問題が修正されています。

JBPAPP-5015

`jta.jar` が seam-gen プロセスによりEclipse のサードパーティ依存として含まれていました。**`jboss-transaction-api.jar`** はseam-gen プロセスに含まれていましたが、この2つのファイルが原因で**`.classpath`** のコンフリクトを引き起こしていました。Hibernate の依存関係がすべて更新され**`jta.jar`** ファイルを除外し依存性の問題を解決しています。

JBPAPP-5056

Enterprise Platform の一部として提供されているSeamBay の例は、Internet Explorer 8との互換性がありませんでした。これは解決され、この例は再度Internet Explorer 8とでも機能するようになっています。

JBPAPP-5078

Enterprise Platform 内の Seam ディストリビューションに全く同じ**`ant.jar`** ライブラリが2つ存在しました。

- **`/seam/lib/ant.jar`**
- **`/seam/lib/gen/ant.jar`**

この問題は**`seam/lib/`**から**`ant.jar`** を削除することで解決されています。

JBPAPP-5410

トランザクション機能を必要とするJBPM の同期呼び出しが原因で、トランザクションが開始されていない場合同期アクションが実行されませんでした。例外がスローされ、処理が継続されませんでした。これは、JBPM同期処理の開始前に、トランザクションが必要かをチェックし、アクティブでない場合は新規トランザクションを開始することで解決されています。

JBPAPP-5469

トランザクション中にアプリケーション例外がスローされると、トランザクションはコミットもロールバックもされず、そのままメモリに保持されていました。例外がスローされるとトランザクションは正しくロールバックされ、メモリから削除されるようになっています。

JBPAPP-5496

`jbpm.cfg.xml` 内の永続サービス設定が『Todo』 と 『DvdStore』 の例で更新され、以下の値**`<service name="persistence" factory="org.jbpm.persistence.jta.JtaDbPersistenceServiceFactory" />`**が利用できるようになっています。

JBPAPP-5517

jbpm-jpdl 処理定義時にSeam EL 表現が評価され例外が発生した場合、JBPMのトランザクション処理はトランザクションをロールバックすべきであるのにコミットしてしまうことがありました。Seam がJBPMプロセスのコミット中にJBPM Contextを終了しようとするため、Seam 側のみでロールバックが起こっていました。Seam はJPBMコンテキストをトランザクションのロールバック直後に終了し、遅延コミットが発生しなくなりました。

JBPAPP-5590

EntityQuery.resultCount() を呼び出すと、**select count(entity)...**などのクエリになっていました。しかし、Hibernate は**count**クエリで複合キーをサポートしないため、永続プロバイダーがHibernateで、Entity に複合キーがある場合、例外 **[ERROR JDBCExceptionReporter] Operand should contain 1 column(s)**がトリガーされます。

この制限により、Hibernateを利用しており、かつ複合キーのあるエンティティーが存在する場合、seam-gen とJBoss Developer Studio を使ったCRUDの生成ができませんでした。Hibernateは、生成コードで**EntityQuery.resultCount()** を広範に利用し、このクラスが例外をトリガーしていました。

Seam が更新され、永続プロバイダーがHibernate の場合、Seam は **select count (entity)**ではなく**select count(*)** の表記を使いクエリを構築するようになっています。このメソッドは複合キーがあっても機能します。この修正をあてると、**EntityQuery.resultCount()** は全エンティティおよび永続プロバイダ、また生成CRUDに対して機能するようになります。

JBPAPP-5823

Seam-RESTEasy の統合モジュールにより、JAX-RS リクエストの呼び出し中に例外が発生した場合、アクティブでないセッションリクエストを開いたままにすることができました。リクエストで不正な認証情報が渡されたとしても、以前に認証済みのセッションにアクセスすることが可能でした。セッションを取り消すコードがJava **finally** ブロックに含まれるようになっており、この修正でアクティブでないセッションリクエストが開いた状態を保持できなくなっています。

JBPAPP-6283

s:link を含むページが過去にJBoss Enterprise Platform インスタンスでレンダリングされたことがない場合、**s:link** が機能せずIllegalStateException: Unable to read <page>が発生しました。これは、 Seam UI ページアクションへのバインディングメソッドが他のサーバー上でSafeActionsに追加なかったためです。**viewId** は各検索パスで Web コンテキスト root へのパスを含むようになり、ノード間でワークロードを移動している間ページが有効な状態を保持することができるようになりました。

JBPAPP-6362

seam-genで、**@Version** によりアノテーションされたEntity フィールドがInteger データ型として設定されていました。Update の操作はInteger データ型と互換性がなく、**NullPointerException**で終了されていました。**@Version**のデータ型がInteger からint に変更され、Entity Update の操作は問題なく完了するようになっています。

JBPAPP-6387

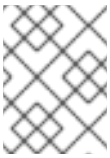
JBoss Seam がページの例外処理時にJBoss Expression Language (EL) コンストラクトへのアクセスを正しく阻止せず、任意の Java メソッドを実行できると判明しました。遠隔の攻撃者は、この不具合を利用し、JBoss Seam 2 フレームワークをベースとした特定のアプリケーションへ特に巧妙に作り上げられたURL経由で任意のコードを実行することができました。

注記

正しく設定され、有効なJava Security Manager により、この不具合の悪用を阻止することができます (CVE-2011-1484)。

JBPAPP-6770

CVE-2011-1484 (JBoss Seam 2 は、ページの例外処理にて悪意のあるJBoss Expression Language (EL) コンストラクトすべてへのアクセスを阻止せず、任意の Java メソッドを実行することができた件) への修正は不完全であることが判明しました。遠隔の攻撃者は、この不具合を利用し、JBoss Seam 2 フレームワークをベースとした特定のアプリケーションへ特に巧妙に作り上げられたURL経由で任意のコードを実行することができました。



注記

正しく設定され、有効なJava Security Manager により、この不具合の悪用を阻止することができます (CVE-2011-2196)。

セキュリティー

JBPAPP-2598

JBAS-7049 の回避策を適用すると、OpenJDK 6 を使いセキュリティマネージャを実行しているサーバーがEnterprise Application Platformを起動することができません。これは、OpenJDKにおけるアップストリームの問題でした。これらの問題は、他社製品であるOpenJDKソフトウェアにて対処されています。そのため、OpenJDK をアップグレードすることで、Enterprise Application Platform は影響を受けたシステム上でも予想どおり起動するようになっています。

JBPAPP-5459

サービス起動の順番におけるエラーが原因で、**`JNDIContextEstablishment`** は予想よりも遅れて起動する可能性があり、**`java:/jaas`** コンテキストがJNDIにバインドされていないため、**`NamingExceptions`**を引き起こしてしまいます。このバインドは**`JaasSecurityManagerService`**で処理されるようになり、**`deploy/`** ディレクトリにて他のサービスより先に起動しています。

JBPAPP-5691

`unauthenticatedIdentity` のオプションは、呼び出されたメソッドが特定のロールを必要としない場合でも**`LdapExtLoginModule`** クラスと連携しませんでした。セキュリティライブラリが更新されこの問題は修正されています。

JBPAPP-5729

以前は**`org.jboss.security.plugins.FilePassword`** ファイルはパスワードの解読にも書き込みの権限が必要でした。また、ファイルにアクセスできないため通常ユーザーに対してファイルを読み取り専用に変更できませんでした。この問題は、ファイルの権限を読み取り専用に変更することで修正されています。

JBPAPP-5940

`LdapExtLoginModule` は、返された分類名をチェックするのではなく、**`roleNameAttributeID`** に対してLDAPのクエリを行っていました。この動作が遅く、パフォーマンスを改善するため に**`parseRoleNameFromDN`**オプションがこのモジュールに追加されています。

システム

JBPAPP-5608

JBoss Transactions の**`QueuedPessimisticEJBLock`** は常に待機中のトランザクションがタイムアウトしたか検出するわけではありませんでした。そのため、ロックを開放すべきときでさえ、ロックが有効のままでした。そのため、スレッドプール全体のロックとなり、全トランザクションがロックが開放されるまで待機し、ロックされたトランザクションはタイムアウトとして登録される結果となっていました。**`isTxExpired()`**が変更されタイムアウトが登録されトランザクションにロックの通知がいくようになっています。

トランザクション

JBPAPP-5175

JBossTS TransactionReaper にはバグが含まれており、動的モードで実行されている場合、間隔を置いて実行されるのではなく継続的に実行されてしまっていました。これによりパフォーマンスの低下が引き起こされていました。この問題を修正するため、JBossTS が更新され、Reaper は間隔を置いて実行されパフォーマンスも改善されています。

JBPAPP-5195

Transactions Recovery Manager は、リカバリが必要なXIDを検索するものの、リカバリ可能なXIDを見落とすことがありました。これは、特定のリソースマネージャーに対してリカバリスキャンを連続で行い、対象になる可能性のあるXIDで同等でないものの一覧を返した場合に発生していました。照合アルゴリズムが強化されリカバリ可能なXIDが処理されるようになっています。

XA対応のJDBCドライバーTX Resource リソースマネージャを利用する場合、XIDがなくなり、ヒューリスティックな例外が発生するリスクがあるため、Resource Manager が返すXIDの順番に依存するアンチパターンとなっており、このようなケースではトランザクションを手動でコミットあるいはロールバックする必要があります。

JBPAPP-5775

トランザクション処理が有効であるにも拘らずノードが予期せず終了した場合、ゼロ長のトランザクションジャーナルファイルが作成される可能性があります。ノードが再起動されると、該当のトランザクションをリカバリするために利用可能な情報が含まれていないにも拘らず、これらのファイルが処理され警告メッセージがログに記録されます。今回の修正でこの状況に対してログが残されないようになっています。

Varia

JBPAPP-5566

`org.jboss.mail.SessionObjectFactory` クラスでのバグにより、JNDIツリーに複数のサービスが設定され存在する場合でもメールサービス 1 つしか解決されませんでした。このクラスはそのように構成されている場合、複数のメールサービスを解決するようになっています。

Web

JBPAPP-4960

`sendNotification`
は**`org.jboss.web.tomcat.service.deployers.TomcatServices.java`**にありましたが無効となっていたため、**`jboss.tomcat.connectors.started notification`** がありませんでし

た。Barrier Controller はこの通知に依存しており、サブスクリプション管理に利用することができませんでした。**sendNotification**を復活させ、この問題は解決されました。

JBPAPP-5168

Oracle WebLogic から移行しているお客様は、webapp以外のコンテキストでURLをリライトする機能が必要であったため、この機能が実装されました。コンテキストをオーバーライドするには、**-Dorg.apache.catalina.connector.Response.REWRITE_CONTEXT_CHECK=false**を設定してください。

JBPAPP-5293

HTTPSコネクタはSSLハンドシェイクを検証するために**addHandshakeCompletedListener()**を利用していますが、このHTTPSコネクタがコネクタとリスナー間で問題なく交渉されていました。通知スレッドが動作を開始した後に**addHandshakeCompletedListener()** がリスナーに追加されると、SSLハンドシェイクを再度交渉する必要がありました。現在、**setEnabledCipherSuites(new String[0])** を利用することでSSLハンドシェイクの交渉を検証しており、この問題は解決されています。この問題はCVE-2009-3555への修正で紹介されています。

JBPAPP-5813

org.apache.tomcat.util.http.ServerCookie.VERSION_SWITCH=falseを使う場合、\, **(**, **)**, **:**, **<**, **=**, **>**, **?**, @, **[**, \\, **]**, **{**, **}**, **(**, **:** の文字を含むクッキーが上記の文字のうち1つが出てきたところで断ち切られていました。上記の文字をコードで処理する方法が変更され、これらの文字を含むクッキーが誤って省略されないようになっています。

Web Services

JBPAPP-4346

署名や暗号化違反がJBoss WSに追加されました。これらの機能は、署名あるいは暗号化要素に**includeFaults="true"** を指定することで有効化できます。クライアント側で **<requires>**タグ内で暗号化および署名要素の**includeFaults="true"** を指定し、署名/暗号化違反の強制が可能になります。

JBPAPP-4506

JBoss Web Services Nativeにサービスをデプロイすると、相対URLでインポートするXML Schema Declaration (XSD) がWSDLサービスコントラクトを起点とした絶対URLで書き直されていました。これは、インポートしたXSDが相対パスを使い2番目のXSDをインポートしない限り、問題なく機能していました。しかし、２つ目のXSDをインポートした場合、2つ目のXSDのパスがWSDLサービスコントラクトではなく、誤って最初のXSDを起点として書き直されていました。この問題は解決され、パスはWSDLサービスコントラクトを起点に書き直されるようになっています。

JBPAPP-4564

soapMessage.getMimeHeaders().addHeader("Transfer-Encoding", "disabled");を呼び出すことで、SAAJ API を使いエンコーディングをまとめて無効にすることができるようになりました。

JBPAPP-4920

JAX-WS向けにWSDLを生成すると、WSDLのメッセージ部分にある名前空間のプレフィックスが不正となっていました。@WebFaultアノテーションで使われる別の名前空間に例外がマッピングされると、この間違った名前空間のプレフィックスが参照されていました。この問題は、WSDLの生成

方法を修正することで解決しており、正しい名前空間のプレフィックスが利用されるようになっています。

JBPAPP-5450

JBossWS ライブラリにあるSAAJ実装のエラーが原因で、以前はSOAPノードと **<Envelope>**ノード が同じプリフィックスを持っていました。これは修正され、SOAPノードは独自のプレフィックスを持つようになっています。

JBPAPP-5494

DescriptorDeploymentAspectはクラスパスに**/cxf.xml**をロードし、CSFエンドポイント配備記述子としてデプロイメントのアタッチメントに置かれていました。これにより、**META-INF/jbosssws-cxf.xml** あるいは **WEB-INF/jbosssws-cxf.xml**がロードされませんでした。これらはCSFバスを作成する**/cxf.xml**と共存するように作られています。このコードが更新され、上記のファイルが正しくロードされるようになっています。

JBPAPP-5545

Element を**SOAPBody** へ追加することで、**java.lang.IllegalArgumentException**が発生していました。このコードが修正されたため、**Element**を追加すると**Element**が**SOAPElement**に変換され例外がスローされてなくなっています。

JBPAPP-5577

WSDL URL から URL への変換に失敗する時の**MessageContextJAXWS** のロギングレベルが高すぎました。これが原因でログメッセージが過剰になっていました。これらのメッセージのロギングレベルが下げられ、重要度が適切に反映されています。

JBPAPP-5710

application/fastinfoset content-type ヘッダーを持つリクエストが**FastInfoset** Web サービスに送信されると、誤ったヘッダー**application/soap+xml** あるいは **text/xml**でレスポンスが買えされていました。この問題はJBoss Web Servicesのコンポーネントをアップグレードすることで修正されます。

JBPAPP-5826

JBossWS は、複数のリクエストをSSLでJBoss Enterprise Application Platformに送信する場合、確立済みのSSL接続を再利用しません。つまり、リクエスト毎にSSLハンドシェイクが発生していました。この問題は修正され、ローカルのWSDLが利用されている場合は、SSLハンドシェイクが1度発生しその後のリクエストで再利用されるようになっています。

第5章 既知の問題

Enterprise Application Platform 5.1.1 での既知の問題をコンポーネント別に一覧にまとめています。

クラスタリング

JBPAPP-4541

MCBean:ServerConfig Profile Service が管理するコンポーネント上でpartitionNameプロパティを使う場合、**null** 値を返していましたが、**MCBean:HAPartition** 管理コンポーネントからのpartionName を使ってください。こちらが正しいプロパティになっています。

JBPAPP-5464

InitialState フェーズでHibernate SecondLevel Cache を使うと、**NullPointerException**が発生します。この問題を回避するには、クエリキャッシュを無効にしてください。クエリキャッシュを無効にする方法ですが、デフォルトではクエリキャッシュが無効となっているため、**persistence.xml**を編集することで、以下のように明示的に **query_cache** を **false**に設定するか、あるいはこの行をすべて削除します。

```
<property name="hibernate.cache.use_query_cache" value="false"/>
```

コンソール

JBPAPP-5285

Admin Consoleを使ってデータベースプロパティを変更すると、変更が永続化されます。しかし、この変更を元に戻しアプリケーションサーバーを再起動すると以前の値が永続されてしまいます。この問題はまだ解決されていませんが、**server/*PROFILE*/data/attachments**にある関連の添付ファイルを編集しデータソース設定が新規設定を強制的に使用させることで回避可能です。

EJB

JBPAPP-4899

jboss-ejb3-core-1.3.5 をデフォルト設定 **LANG=C** でコンパイルすると、マッピングできない文字が表示されASCIIのエンコーディングエラーとなります。**LANG=en_US.UTF-8** を使うことでこの問題を回避してください。

JBPAPP-5121

永続ユニットがEARの外側にあり、永続ユニットの注入を試行する bean がEAR内にある場合、永続ユニットをEARにデプロイできません。永続ユニットが見つからないため、この注入は失敗します。EJB3仕様では予期される動作で、EJB3仕様に厳密に準拠するには、永続ユニットはEARにパッケージする必要があります。JBoss 固有の動作により、永続ユニットがEARの外側にも存在可能となっています。これは JBoss AS サーバープロファイルの下にある**deployers/ejb3.deployer/META-INF/jpa-deployer-jboss-beans.xml** ファイルで設定されおり、該当箇所は以下の通りです。

```
<bean name="PersistenceUnitDependencyResolver"
class="org.jboss.jpa.resolvers.DynamicPersistenceUnitDependencyResolver"
/>
```

デフォルトでは、DynamicPersistenceUnitDependencyResolverが使われており、JMX Console を使いMBean 経由で仕様に準拠した動作となるよう制御可能です。仕様に準拠しない JBoss Search Strategy は、こちらにあります。

Hibernate

JBPAPP-6395

LEFT OUTER JOIN でCriteria API を利用し、子に基準を追加する場合、子のコレクションにはこの基準と一致する子しか含まれなくなります。この動作は、フィルターを利用しない場合でも適用されます。例えば、

```
criteria.createCriteria("children", JoinFragment.LEFT_OUTER_JOIN)
```

この問題に対する回避策はありません。

JBPAPP-6475

エンティティーに以下の条件が含まれている場合、org.hibernate.PropertyAccessException がスローされる可能性があります。

1. @EmbeddedIdを使う場合
2. コレクションあるいは関連プロパティ/フィールドで@JoinTable を利用し、このエンティティーの別プロパティー/フィールドを参照している場合

例：

```
@EmbeddedId
private MyPk id;
private Long name;
@CollectionOfElements
@JoinTable(
name="GLOBAL_NOTES",
joinColumns=@JoinColumn(name="text_id", referencedColumnName="name"))
private Set<String> globalNotes = new HashSet<String>();
```

この問題に関する回避策ですが、@EmbeddedIdを利用しないという策しかありません。

メッセージング

JBPAPP-3904

Oracle 11g R1を利用している場合、Oracle JDBC ドライバ version 11.1.0.7.0 が原因で、JBoss Messaging Transaction Service が**`SQLException`** ("Bigger type length than Maximum")で失敗します。

これは、Oracle JDBC driver 11.1.0.7.0における再発バグにより起こっています。

Oracle 11g R1、Oracle 11g R2、Oracle RAC 11g R1、Oracle RAC 11g R2で利用する場合は、Oracle JDBC ドライバー version 11.2.0.1.0 を推奨しています。

JBPAPP-4668

Oracle 11g R1、R2、RAC 上で最新のJDBCドライバー version 11.2.0.2.0を使うと、JBoss Messaging Test Suite テスト2つが失敗します。

- QueueManagementTest.testDestroyDestinationProgrammatically
- QueueManagementTest.testDestroyDestinationProgrammaticallyWithParams

これらのテストは、fullSize キュー設定パラメータに大きな値を使い、**`java.sql.PreparedStatement`**で**`setFetchSize`** メソッドへ渡します。JDBCドライバの問題は、**`executeQuery()`** が呼び出されると、通常のメモリ以上の量を消費していることになります。その結果、**`java.lang.OutOfMemoryError`**が発生しテストが失敗してしまいます。

JBPAPP-5124

JDBC driver Sybase jConnect JDBC driver v7 (Build 26502)でSybase データベースを使うと、このドライバの**`PreparedStatement`** クラスの制限により、**`sybase-persistence-service.xml`** 設定ファイルの**`MaxParams`** 属性を **`481`** より大きく設定できなくなります。 **`MaxParams`** 属性が**`481`**よりも大きい値に設定されると、予期せず失敗してしまう可能性があります。この問題を回避するには、**`MaxParams`** を **`481`**よりも小さい値にします。

JBPAPP-5537

メッセージ駆動型 beanがデフォルト設定 **`useDLQ=true, DLQMaxResent=5`** でデプロイされ、メッセージの再送が促されると、メッセージがデッドレターキューに送信された後でもメッセージが**`delivering`** の状態でキューに残ります。この問題を回避するには、**`useDLQ=false`**に設定することでMDB側で処理するデッドレターキューを無効にします。

ネットワーキング

JBPAPP-5591

Enterprise Platformのバージョン5ではIPv6に対応していません。

RESTEasy

JBPAPP-4665

`java.lang.SecurityException`が原因で、Resteasy-guice アプリケーションがデプロイに失敗します。以下のようなエラーメッセージが表示されます。

```
java.lang.SecurityException: class
"org.jboss.resteasy.examples.guice.hello.DefaultGreeter$$FastClassByGuic
e$$70fd68d0"'s signer information does not match signer information of
other classes in the same package
```

これは、JBoss Enterprise Application Platform の **`cglib.jar`** が署名され、**`cglib-instrumented`** プロキシがアプリケーションターゲットクラスの署名者情報ではなく、**`cglib.jar`** の署名者情報を使うためです。

JBPAPP-4995

2010年8月え基本認証メソッドをTwitter が廃止するとのことで、TwitterClient の例は RESTEasy 1.2.x で廃止予定となっています。そのため、アプリケーションはすべてOAuthを使う必要があります。RESTEasy 2.x には TwitterClient の例を編集しOAuthが入ったものが含まれるようになっています。テスト目的で利用いただく場合、RESTEasy 2.x から例をダウンロードしてください。

JBPAPP-5038

jettison.jar ファイルが **jboss-eap-noauth-5.1.0.CR3.zip**の**jboss-eap-5.1/resteasy/lib** ディレクトリに含まれていません。この問題を回避するには、Seam ディストリビューションに含まれている**jettison.jar** ファイル、**jboss-eap-5.1.1/seam/lib/jettison.jar**を利用してください。

スクリプト

JBPAPP-5003

jboss_init_redhat.sh のスクリプトがディストリビューションから削除されました。代わりに、ご自身のスクリプトを使うことで、Enterprise Platformのスタートアップをカスタマイズできます。

Seam

JBPAPP-5039

Microsoft Internet Explorer 8とSeam のタスク例との互換がないことで、タスクに添付されたテキストがタスク一覧に空白の行として表示されてしまいのですが、この問題の回避策はありません。

JBPAPP-6366

jboss-seam2-examples ディストリビューションに含まれる例のうち、JBoss Enterprise Application Platform サーバーの機能を紹介することを目的に設計されているものもあります。完全なApplication Platform サーバーを必要とするこれらの例は、JBoss Enterprise Web Platform サーバーでは機能しません。

TCK

JBPAPP-3929

java.sql.Date.valueOf が日付の形式 yyyy-mm-dd を解析しようとすると、TCKテストがjava.lang.IllegalArgumentExceptionをスローしていました。これは、最新のSun JVM、Sun JDK 1.6.0_24の再発バグが原因となっています (詳細はhttp://bugs.sun.com/bugdatabase/view_bug.do?bug_id=6898593 を参照)。Sun JDK 1.6.0_17にダウングレードすることで、この問題を回避できます。

Varia

JBPAPP-4912

webapp から依存されているデータソースが再起動すると、NullPointerException がスローされていました。このエラーが IllegalStateExceptionに変更され、"WebModules cannot be restarted, and must be redeployed"と買えされるようになりました。

付録A 改訂履歴

改訂 5.1.1-2.400	**2013-10-31**	**Rüdiger Landmann**
Rebuild with publican 4.0.0		
改訂 5.1.1-2	**2012-07-18**	**Anthony Towns**
Rebuild for Publican 3.0		
改訂 5.1.1-105	**Mon Jul 18 2011**	**Jared Morgan**
JBoss Enterprise Application Platform 5.1.1 GAに対する変更を盛り込む		