

バージョン 6.2.0



インストールの計画

バージョン 6.2.0



インストールの計画

お願い

本書および本書で紹介する製品をご使用になる前に、本書末尾の『特記事項』セクションに記載されている情報をお読みください。

本書は、WebSphere® Process Server for Multiplatforms バージョン 6、リリース 2、モディフィケーション 0 (製品番号 5724-L01) および新しい版で明記されていない限り、以降のすべてのリリースおよびモディフィケーションに適用されます。

お客様の環境によっては、資料中の円記号がバックスラッシュと表示されたり、バックスラッシュが円記号と表示されたりする場合があります。

原典： WebSphere® Process Server for Multiplatforms
Version 6.2.0
Planning the Installation

発行： 日本アイ・ビー・エム株式会社

担当： トランスレーション・サービス・センター

第1刷 2009.1

© Copyright International Business Machines Corporation 2007, 2008.

目次

第 1 章 概要: WebSphere Process Server の計画	1
第 2 章 ソフトウェアのニーズの判別	3
ビジネス要件の評価	3
使用可能なリソースの明確化	4
開発およびデプロイメントのバージョン・レベル	5
WebSphere Process Server と他の WebSphere Application Server 製品との間のインターオペラビリティの計画	6
インストールする製品の決定	7
データベースの選択	9
必要なセキュリティ権限の明確化	12
サーバーおよびクラスターでの Service Component Architecture サポートに関する考慮事項	13
第 3 章 セル内での複数のプラットフォームの使用	15
第 4 章 デプロイメント環境の計画	17
計画のためのシナリオ	19
WebSphere Integration Developer のインストール時に WebSphere Process Server もインストールする計画	19
WebSphere Integration Developer による使用のための WebSphere Process Server のインストール計画	21
デフォルトのスタンドアロン環境の計画	24
カスタム・スタンドアロン環境の計画	26
提供されたパターンの 1 つに基づくデプロイメント環境の計画	29
カスタム・デプロイメント環境の計画	33
プロファイル	37
サーバー	39
スタンドアロン・サーバー	39
Network Deployment	41
デプロイメント・マネージャー	42
管理対象ノードの概要	42
デプロイメント環境	43
デプロイメント環境パターンの選択	52
第 5 章 デプロイメント環境の実装	55

第 6 章 エラー防止およびリカバリーの計画	61
エラー防止とリカバリーの概要	61
エラー防止の計画	62
アプリケーション設計の一部としてのエラー防止	62
開発の一部としてのエラー防止	67
問題判別方法の文書	72
ソフトウェア適用状況	74
エラー処理方針とソリューション・リカバリー	74
安定した環境の保守	75
リカバリー方法の計画	76
高可用性 (HA)	76
リカバリーの環境および目的	77
トランザクションのプロパティとソリューション・リカバリー	80
ピア・リカバリー	82
エクスポート・バインディング	84
Failed Event Manager について	85
障害からのリカバリー	87
リカバリー・プロセスの概要	87
リカバリーのトリガー	87
システムの状態の評価	89
リカバリー: 問題の分析	92
リカバリー: ファースト・ステップ	93
失敗したイベントのロケーション: データの行き先	95
デプロイメント環境の再始動	105
サービス統合バスの表示	107
javacore の収集	111
サーバーおよびリカバリー・モード処理	112
保存キューと保留キュー	113
Business Process Choreographer の保守スクリプトとリカバリー・スクリプト	114
未確定トランザクションの解決	116
DB2 診断情報の確認	118
プロセス・リカバリーのトラブルシューティングのヒント	119
メッセージング・サブシステムのリカバリーについて	120
特記事項	121

第 1 章 概要: WebSphere Process Server の計画

WebSphere® Process Server などのミドルウェアでは、製品をインストールする前に、エンタープライズ情報システム (EIS) の多くの側面 (キャパシティーやセキュリティなど) を評価する必要があります。また、計画性を持ってエラー防止やリカバリーなどの WebSphere Process Server の機能を活用する必要があります。

以下の各質問への回答が、お客様のニーズに合わせてデプロイメント環境を設計するために役立ちます。

- ビジネス目標は何ですか。また、その目標を達成するために、このソフトウェアはどのように役立ちますか。
- どのようなアプリケーションを統合する必要がありますか。
- 重複した情報を除去しますか。
- システム応答時間と可用性の要件は何ですか。
- インストールを完了するためにどのような財務、ハードウェア、ソフトウェア、および人材の各資源が使用可能ですか。
- 他の部門のサービスが必要ですか。
- どのような作業を行う必要がありますか。それらの作業を行うのは誰でしょうか。
- どのような既存のハードウェアがインストールに必要ですか。
- ビジネス要件を達成するために追加のハードウェアが必要ですか。
- 既存のデータベースを使用できますか。それとも新規のデータベースが必要ですか。
- 既存のユーザー ID を WebSphere Process Server コンポーネントで使用できますか。それとも新規 ID が必要ですか。どのような権限が新規 ID に必要ですか。
- 購入可能な製品ライセンスの数を制限する財務上の考慮事項はありますか。
- お客様のシステムの増大度合いはどの程度ですか。例えば、今後、負荷の増加、およびより多数の同時ユーザーに対応することが必要になりますか。追加の要求を満たすために、リソースを今後さらに追加することが必要になりますか。
- お客様のシステムで、日々の要求の変動に対応するために動的にリソースを追加または除去することが必要になりますか。
- お客様のシステムで、負荷または同時ユーザー数の変動に定期的に対応する必要がありますか。

また、お客様の以下の現在の目標についても考慮します。テスト環境または実稼働環境を計画していますか。その環境は小規模ですか。それとも大規模ですか。使用する環境をデフォルト値で素早くセットアップしたいですか。それともカスタマイズしたいですか。このセクションの最後に、お客様が達成しようとするに基づき、さまざまなシナリオの計画のための提案があります。

第 2 章 ソフトウェアのニーズの判別

やり直しや障害を最小にするには、先に進んでインストールの決定を行う前に、現在の環境の把握に時間を掛けてください。現在のビジネス・ニーズと設計、既にインストールされているハードウェアとソフトウェア、および現在の長所と短所の分析は、デプロイメント環境に最適な設計を判別するのに役立ちます。また、この計画は、現在のニーズに必要な財務投資を最小化するのにも役立つ場合があります。

このセクションの情報により、現在および将来のニーズを分析し、それらのニーズに合う環境を開発します。

注: プラットフォーム固有のディスク・スペース所要量、サポートされているオペレーティング・システム、サポートされているデータベースのバージョン、およびオペレーティング・システムを準拠させるためにインストールする必要があるオペレーティング・システムのフィックスおよびパッチについての最新情報は、[WebSphere Process Server detailed system requirements](#)にある [WebSphere Process Server](#) の詳細なシステム要件を参照し、ご使用のバージョンの [WebSphere Process Server](#) へのリンクを選択してください。

ビジネス要件の評価

現在のビジネス要件は、ビジネス・コンポーネントの統合を合理化し、機能を向上させるための計画を立案する基礎になります。ビジネスの将来を見据えることにより、現在の業務を遂行するだけでなく、ビジネスの拡大も視野に入れた意思決定の指針を築くことができます。

始める前に

製品の製造および出荷、サービスの提供がどのように行われているかを把握する必要があります。

このタスクについて

計画プロセスの一部として、業務の仕組みを分析する必要があります。ここに示す手順では、この分析のためのフレームワークを提供します。

手順

1. 製品またはサービスの流れを、始めから終わりに至るまで図式化します。

プロセスは直線的な場合も、ループを持つ場合も、途中をバイパスする場合も、回避を行う場合もあります。ラフな図を描き、接続および相互の連絡関係を書き込みます。製品ライフ・サイクルの各セクションで、製品を次の段階に進めるために使用する手順を分析します。

- 手順では紙のフォームやメモを使用していますか、コンピューター化されていますか、紙とコンピューターの両方が混在していますか。
- コンピューター化されている場合は、どのようなソフトウェアが使用されていますか。どのようなハードウェアですか。

- プロセスに行き詰まりがありますか。混乱している箇所がありますか。例えば、手書きの文字が読みにくくて判読に時間がかかっている、スタッフが必要なコンピューター・スキルの習得に苦労しているなど。
 - このプロセスでどの領域がスムーズに行われていますか。強みは何ですか。
2. ステップ 1 (3 ページ) で作成した図の各セクションについて、それらの作業の実行方法を調べます。
 - すべてのセクションで同じソフトウェアを使用していますか。ハードウェアはどうですか。フォームはどうですか。
 - セクションごとに異なるソフトウェアを使用している場合、アプリケーション相互の関係が必要なときに、その関係がスムーズに行われていますか。
 - 各セクションは直前および直後のセクションのみとやり取りしていますか、それともサイクルの別の段階にあるセクションともやり取りしていますか。そうなっている場合、その理由は何ですか。それが原因で混乱や遅れが発生しませんか。
 - セクションが通信に使用するイントラネットが既に存在する場合に、そのイントラネットをバイパスするセクションがありますか。そうなっている場合、その理由は何ですか。イントラネットで、遅延やダウン時間が発生して他のプロセスに影響したことがありますか。
 - 相互連絡のどの領域がスムーズに行われていますか。ボトルネックはどの領域にありますか。重大度はどれくらいですか。
 3. 外部の調達先と相互連絡するプロセスについて考慮します。
 - お客様から、どのようなコメント (肯定的なものも否定的なものも含めて) が届きますか。苦情に一定の傾向がありますか。常に特定の領域が顧客を満足させていますか。
 - 他の企業とはどのようにやり取りしていますか。どの部門がコミュニケーションを担当していますか。コミュニケーションの方法は文書ですか? Web ベースですか? これらの調達先を図に追加します。スムーズに動く領域と、遅れや誤りが発生する可能性がある領域を突き止めます。
 4. 将来について計画します。
 - この先 1 年のビジネスをどのように展望していますか。5 年ではどうですか。10 年ではどうですか。
 - 新しい販路を開拓しますか。広告を増やしますか。顧客基盤を拡大しますか。
 - 競合他社を買収し、その製品およびサービスを自社に取り込む可能性はありますか。新しい分野の製品またはサービスを開拓する可能性はありますか。

次のタスク

使用可能なリソースを明確化します。

使用可能なリソースの明確化

資産を明確化し、既に使用可能なリソースを最大限に利用するとともに、購買の決定の通知を受け取ります。

始める前に

現在のハードウェアおよびソフトウェアに精通している必要があります。以下のようにして使用可能な資産のリストを準備します。

このタスクについて

現在のエンタープライズ情報システムを評価し、業務上のニーズを満たすためにハードウェアまたはソフトウェアを追加する必要があるかを判別します。

手順

1. それぞれの物理的ハードウェアを列挙します。
 - 実装済みメモリーの量
 - 実装済みマイクロプロセッサの数およびタイプ
 - 外部メディア
 - 特定の装置がアップグレード可能かどうか
2. 現在インストールされているソフトウェアおよびデータベース・アプリケーションを列挙します。
 - 機能
 - 企業内での使用範囲
 - セキュリティー要件
3. 現在の IT 担当者をリストし、WebSphere Process Server のインストールおよび保守の手段が適用できるか、およびデータベース管理に必要な専門家がいるかを確認します。すべての製品およびファイルを正常にインストールするための適切な権限を持つユーザー ID がすべての関係者に割り当てられていることを確認します。

関連概念



ハードウェアおよびソフトウェア要件

このトピックには、WebSphere Process Server のインストールに必要な、ハードウェア要件とソフトウェア相互要件および前提条件に関する追加情報へのリンクが含まれています。

開発およびデプロイメントのバージョン・レベル

ご使用の環境に必要な WebSphere Process Server のバージョン・レベルの決定は、アプリケーションが開発されたときのバージョン・レベルに依存します。一般に、前のバージョンの WebSphere Process Server にデプロイされたアプリケーションは、次に入手可能なバージョンの WebSphere Process Server 上で稼働します。

WebSphere Process Server バージョン 6.2 および WebSphere Integration Developer バージョン 6.2 は、以前のリリースとの間に以下に示す互換性があります。

- WebSphere Integration Developer バージョン 6.0.2.x または 6.1 から WebSphere Process Server 6.2 へのデプロイメントがサポートされています。
 - WebSphere Integration Developer 6.0.2.x または 6.1 を使用して作成および生成されたアプリケーションは、WebSphere Process Server 6.2 サーバーにパブリッシュできます。

- WebSphere Integration Developer 6.0.2.x または 6.1 で作成、生成、およびそこからエクスポートされたアプリケーションは、WebSphere Process Server 6.2 サーバーにインストールできます。

注: バージョン 6.0.1 WebSphere Adapters の場合、互換性を保つにはいくつかの追加ステップが必要です。詳細については、『WebSphere Process Server technotes』の製品技術情報を参照してください。

- WebSphere Process Server 6.2 成果物を WebSphere Process Server 6.0.2.x または 6.1 で実行することはサポートされていません。
 - WebSphere Integration Developer 6.2 で作成されたアプリケーションは、WebSphere Process Server 6.0.2.x または 6.1 (前のすべてのリリース) サーバーにパブリッシュまたはインストールすることはできません。このようなコンテンツは WebSphere Process Server 6.0.2.x または 6.1 で正しく稼働せず、コードの世代変更によってアプリケーションは WebSphere Process Server 6.0.2.x または 6.1 で正しく稼働しなくなります。
 - WebSphere Integration Developer 6.0.2.x または 6.1 で作成され、WebSphere Integration Developer 6.2 で生成されたアプリケーションは、WebSphere Process Server 6.0.2.x または 6.1 サーバーにパブリッシュまたはインストールできません。コードの世代変更によってアプリケーションは WebSphere Process Server 6.0.2.x または 6.1 で正しく稼働しなくなります。
 - serviceDeploy を使用して WebSphere Process Server 6.2 サーバーから生成されたアプリケーションは、WebSphere Process Server 6.0.2.x または 6.1 サーバーにインストールできません。コードの世代変更によってアプリケーションは WebSphere Process Server 6.0.2.x または 6.1 で正しく稼働しなくなります。

関連概念

『WebSphere Process Server と他の WebSphere Application Server 製品との間のインターオペラビリティの計画』

ソフトウェア環境を分析するときは、デプロイメント環境内に存在するさまざまなソフトウェア・レベル間で要求を受け渡すことができるかどうかを把握する必要があります。

関連情報

 [WebSphere Process Server へのマイグレーション](#)

WebSphere Process Server と他の WebSphere Application Server 製品との間のインターオペラビリティの計画

ソフトウェア環境を分析するときは、デプロイメント環境内に存在するさまざまなソフトウェア・レベル間で要求を受け渡すことができるかどうかを把握する必要があります。

最適なインターオペラビリティを維持するには、関連するすべての WebSphere Application Server サービスを適用し、該当するすべてのガイドラインに従った後、WebSphere Process Server の未解決のフィックスを適用してあることを確認してください。

関連概念

5 ページの『開発およびデプロイメントのバージョン・レベル』

ご使用の環境に必要な WebSphere Process Server のバージョン・レベルの決定は、アプリケーションが開発されたときのバージョン・レベルに依存します。一般に、前のバージョンの WebSphere Process Server にデプロイされたアプリケーションは、次に入手可能なバージョンの WebSphere Process Server 上で稼働します。

関連タスク



Update Installer を使用したフィックスパックおよびリフレッシュ・パックのインストール

IBM® Update Installer for WebSphere Software を使用して暫定修正、フィックスパック、およびリフレッシュ・パック (メンテナンス・パッケージと総称される) をインストールできます。Update Installer for WebSphere Software はまた、アップデート・インストーラー・プログラム、UpdateInstaller プログラム、およびアップデート・インストール・ウィザードと呼ばれています。

関連情報



相互運用 (WebSphere Application Server)

インストールする製品の決定

デプロイメント環境の設計には、必要になる可能性があるソフトウェア製品の数とタイプの判断が含まれます。製品の要件は、お客様のニーズに基づき、その環境に関連するコンピューターシステムによって異なる可能性があります。デプロイメント環境のすべてのサーバーにそれぞれ WebSphere Process Server ライセンスが必要なわけではありません。

始める前に

設計に関する以下の詳細情報が必要です。

- デプロイメント環境に組み込むクラスターおよびサーバー
- さまざまなサーバーを置く物理的ハードウェア
- 各クラスターがデプロイメント環境に提供する機能例えば、Web アプリケーション・コンポーネントのサポート、Java™ Platform Enterprise Edition コンポーネントのサポート、メディアエーション・モジュールのサポート、メッセージングのサポート、またはプロセス・サーバーのサポート。

このタスクについて

デプロイメント環境を設計した後、ソフトウェアを購入する前にこのタスクを使用して、ご使用のデプロイメント環境に必要な適切なソフトウェアを判断してください。

手順

1. デプロイメント環境のさまざまなコンポーネントをホストする専用のコンピューターの数のカウントします。

重要: 同じコンピューターシステム上で複数のサーバーを実行している場合、そのコンピューターシステム上で動作するすべてのサーバーに必要な機能をも

も多数提供するソフトウェアをインストールする必要があります。
このカウントには以下のものを含まず。

- 必要なデプロイメント・マネージャーの数。管理対象サーバー上で動作するソフトウェアにより、デプロイメント・マネージャーにインストールするソフトウェアが決まります。
 - WebSphere Process Server インスタンス
 - WebSphere ESB インスタンス: メディエーションのみをホストするための専用のインスタンス
 - まだカウントしていないメッセージング・エンジン: 固有の WebSphere Application Server インスタンスの数を表します
2. ソフトウェアのコストがプロジェクトの予算を超えるかどうかを判定します。
 3. オプション: 財務上の要件を満たすように設計を調整します。コストを下げるには、容量が最大のコンピューターで複数のサーバーをホストする必要があります。
 - 別個のコンピューターに別個のサーバー・インスタンスを作成するより、容量が大きいコンピューターに同じタイプの複数のサーバー・インスタンスを作成すると、インスタンスの数は同じでも必要なソフトウェアの数が減ります。
 - メッセージング・エンジンをホストする専用のコンピューターが必要かどうかを判断します。不要な場合は、それらを除去します。
 - 設計からアプリケーションを除去して、必要なアプリケーション・サーバー・インスタンスの数を減らします。

タスクの結果

以上で、設計の実装に必要なソフトウェアが判明しました。

次のタスク

必要なソフトウェアを発注します。

関連概念

43 ページの『デプロイメント環境』

デプロイメント環境とは、Service Component Architecture (SCA) の対話をホストするための環境を共同して提供する、構成済みのクラスター、サーバー、およびミドルウェアの集合のことです。例えば、デプロイメント環境には、メッセージの宛先用のホスト、ビジネス・イベントの処理プログラム、および管理プログラムが組み込まれている場合があります。

45 ページの『デプロイメント環境のクラスター』

クラスターを使用すると、アプリケーションの能力と可用性が単一サーバーの場合よりも高まります。

39 ページの『サーバー』

サーバーによって、WebSphere Process Server のコア機能が提供されます。プロセス・サーバーでは、アプリケーション・サーバーの機能が拡大または拡張されて、Service Component Architecture (SCA) モジュールが処理されます。他のサーバー (デプロイメント・マネージャーおよびノード・エージェント) は、プロセス・サーバーの管理に使用されます。



メッセージング宛先ホストまたはキュー宛先ホスト

メッセージング宛先ホストまたはキュー宛先ホストにより、サーバー内にメッセージング機能が提供されます。サーバーをメッセージング・ターゲットとして構成すると、サーバーはメッセージング宛先ホストになります。

データベースの選択

データベースの選択は、オペレーティング・システムによって、またご使用のフィーチャによって異なります。インストール手順で、ウィザードからデータベースを選択するというプロンプトが出されます。場合によっては、複数のテーブルを含む 1 つのデータベースだけで操作することが可能です。

始める前に

データベース構成の計画を立てるには、使用するコンポーネントがわかっている必要があります。

このタスクについて

このタスクを実行すると、構成するデータベースの数がわかります。

次の表は、各種の WebSphere Process Server コンポーネントと、そのコンポーネントに関連するテーブルが含まれる対応データベースとのマッピングを表しています。

i5/OS® プラットフォームでは、他の分散プラットフォームに対してここで示されている複数のデータベースの代わりに、固有の名前を持つデータベース・コレクションに同じコンポーネント・テーブルが常駐します。

手順

必要なデータベース表を判別するのにインストール・システムが使用するコンポーネントを選択します。表 1 は、コンポーネントと、そのコンポーネントに必要な対応データベース表のリストです。

プラットフォーム固有のディスク・スペース所要量、サポートされているオペレーティング・システム、およびオペレーティング・システムを準拠させるためにインストールする必要があるオペレーティング・システムのフィックスおよびパッチについての最新情報は、WebSphere Process Server detailed system requirementsにある WebSphere Process Server の詳細なシステム要件を参照し、ご使用のバージョンの WebSphere Process Server へのリンクを選択してください。

表 1. 個々のコンポーネントに必要なデータベース

コンポーネント	データベース (指定されている名前はデフォルトですが、使用する際に変更できます)
AppScheduler	共通データベース (WPRCSDB)
Business Process Choreographer	Business Process Choreographer データベース (BPCDB)

表 1. 個々のコンポーネントに必要なデータベース (続き)

コンポーネント	データベース (指定されている名前はデフォルトですが、使用する際に変更できます)
Business Process Choreographer Explorer レポート	Business Process Explorer レポート・データベース (OBSRVDB) 重要: 性能低下を防ぐには、レポート・データベースに専用のデータベースを用意してください。別のデータベースにテーブルだけを置くことは避けてください。
ビジネス・スペース	WPRCSDB (共通データベース)
Common Event Infrastructure (CEI)	CEI データベース (EVENT) 重要: パフォーマンスの低下を防ぐには、CEI が、単に別のデータベース内のテーブルを所有しているのではなく、独自のデータベースを所有していることを確認してください。
Enterprise Service Bus	WPRCSDB (共通データベース)
EventSequencing (LockManager)	WPRCSDB (共通データベース)
メディアエーション	WPRCSDB (共通データベース)
リカバリー	WPRCSDB (共通データベース)
リレーションシップ	WPRCSDB (共通データベース)
セクター/ビジネス・ルール	WPRCSDB (共通データベース)
Service Integration Bus	SIBDB (メッセージング・エンジンの構成時に作成)

関連概念

データベース構成

WebSphere Process Server では、情報の保持、格納、追跡のためにさまざまなデータベース表を使用します。これらのデータベース表の作成作業は、WebSphere Process Server の構成プロセスの一部になっています。これらのデータベース表は、プロファイル作成中に作成することも、スクリプトを使用して別途作成することもできます。

共通データベースの構成

共通データベースの構成には、サポートされるデータベース・タイプ、スクリプトとその場所、プロファイル作成の構成アクション、インストール・パラメーター、作成されるテーブルとユーザー ID の特権のタイプに関する情報が含まれません。

Common Event Infrastructure データベースの構成

Common Event Infrastructure データベース仕様には、サポートされるデータベースのタイプ、スクリプトの位置、プロファイル構成タイプ、および必要なユーザー ID の特権がリストされます。

Business Process Choreographer データベースの構成

Business Process Choreographer データベース仕様には、サポートされるデータベース・タイプ、スクリプトの位置、プロファイルの作成タイプ、データベースの制約事項、および必要なユーザー ID の特権がリストされます。

メッセージング・エンジン・データベースの構成

このメッセージング・エンジン・データベースの仕様には、サポートされるデータベース・タイプ、スクリプトとそれらの場所、プロファイル作成のタイプ、および必要なユーザー ID の特権がリストされています。

エンタープライズ・サービス・バスのロガー・メディエーション・データベースの構成

このエンタープライズ・サービス・バスのロガー・メディエーション・データベースの仕様を使用して、サポートされるデータベース・タイプ、スクリプト名とそれらの場所、プロファイル作成の構成操作、スキーマのアップグレード、およびユーザー ID の特権に関する情報を調べてください。

セレクター/ビジネス・ルール・グループ・データベース構成

このセレクター/ビジネス・ルール・グループ・データベースの仕様を使用して、サポートされるデータベース・タイプ、スクリプトとそれらの場所、プロファイル作成の構成操作、制限事項、テーブル名、およびユーザー ID の特権に関する情報を調べてください。

JDBC プロバイダー

アプリケーションは、JDBC プロバイダーによってリレーショナル・データベースと対話できます。

データ・ソース

データ・ソースは、アプリケーションとリレーショナル・データベースの間のリンクを提供します。

リモート z/OS[®] サーバー上の DB2[®] での Common Event Infrastructure リポジトリと共通データベース・リポジトリの作成

Common Event Infrastructure および共通データベースのリポジトリ用に DB2 をリモート z/OS マシンで使用する場合は、ユーザーまたはデータベース管理者 (DBA) は、関連するデータベースおよび正しいストレージ・グループを z/OS ワークステーション上に作成する必要があります。

関連タスク

プロファイルの作成

新規の WebSphere Enterprise Service Bus または WebSphere Process Server プロファイルを作成する方法について説明します。プロファイルの作成は、`manageprofiles` コマンドを使用してコマンド行から行うことも、プロファイル管理ツールのグラフィカル・ユーザー・インターフェース (GUI) を使用して対話式に行うこともできます。

既存のプロファイルの拡張

既存の WebSphere Application Server、WebSphere Application Server Network Deployment、または WebSphere Application Server Network Deployment with Web Services Feature Pack プロファイルを WebSphere Enterprise Service Bus や WebSphere Process Server プロファイルに拡張できます。あるいは、WebSphere Enterprise Service Bus プロファイルを WebSphere Process Server プロファイルに拡張することもできます。このトピックの説明を使用してプロファイルを拡張

します。プロファイルの拡張は、`manageprofiles` コマンドを使用してコマンド行から行うことも、プロファイル管理ツールのグラフィカル・ユーザー・インターフェース (GUI) を使用して対話式に行うこともできます。

必要なセキュリティ権限の明確化

サイトのセキュリティ・ポリシーによっては、設計を正常に実装するために、ファイルおよびフォルダーの作成、データベースへのアクセスなどのさまざまな作業を行えるユーザー ID およびパスワードが必要な場合があります。必要な権限を明確にすることで、サーバーが保護データにアクセスするときの問題を回避できます。

始める前に

- 設計を完了します。
- 使用する認証システム (例えば Lightweight Directory Access Protocol (LDAP)) を決定します。
- サイトのセキュリティ・ポリシーを確認し、適用されている制御のうち、どの制御が WebSphere Process Server のインストールに必要な権限に影響するかを判別します。
- 製品をインストールするシステムを識別します。

このタスクについて

サイトのセキュリティ・ポリシーではグローバル・セキュリティが有効になっており、ソフトウェアのインストール、データベースまたはテーブルの作成、およびデータベースへのアクセスを行うために所定の権限が必要になっています。正常に製品をインストールして運用するためには、ここに示すステップを行う必要があります。

- システムにソフトウェアをインストールするために十分な権限を持つユーザー ID およびパスワードを獲得するか、セキュリティ管理者に提供します。

必ずファイルおよびフォルダーを作成するための権限を持つ ID を使用して WebSphere Process Server のインストール・ウィザードを実行してください。

- システムの日常の運用に必要なユーザー ID、パスワード、およびロールを獲得するか、またはこれらをセキュリティ管理者に提供します。これらには、以下のようなものがあります。
 - 管理コンソールのユーザー ID と能力を制限するロール。ロールの構成、管理、またはモニター用のユーザー ID を所有することができます。
 - システム通信の認証に使用するシステム・バスごとのユーザー ID。
 - Business Flow Manager と Human Task Manager での認証のための Business Process Choreographer Container ごとの管理およびモニター用のユーザー ID またはグループ。
 - Business Flow Manager と Human Task Manager での認証のための同期呼び出し用のユーザー ID またはグループ。
- オプション: インストール中に WebSphere がデータベースまたはデータベース表を作成するために使用するユーザー ID およびパスワードを獲得するか、データベース管理者に提供します。


注: サイト・ポリシーによっては、この権限がデータベース管理者に制限されている場合があります。その場合は、生成されたスクリプトを管理者に渡してデータベースまたはデータベース表を作成してください。

- 運用中に WebSphere がデータベース表にアクセスするために使用するユーザー ID およびパスワードを獲得するか、データベース管理者に提供します。

タスクの結果

WebSphere サーバーをセキュアな環境にインストールして運用できます。

関連情報

 セキュリティー、ユーザー ID、および許可の計画

サーバーおよびクラスターでの Service Component Architecture サポートに関する考慮事項

サーバーおよびクラスターでは、Service Component Architecture (SCA) アプリケーション、アプリケーション宛先、またはその両方をサポートできます。

SCA アプリケーション (サービス・アプリケーションとも呼ばれる) では、自動的に作成される 1 つ以上のサービス統合バスを使用する必要があります。各アプリケーションでは、一連のメッセージング・リソース (宛先 と呼ばれる) が使用されます。これらの宛先には構成されたメッセージング・エンジンが必要であり、またこれらの宛先はアプリケーションと同じサーバーまたはクラスター、あるいはリモート・サーバーまたはリモート・クラスター上でホストすることができます。一般にメッセージング・エンジンはデータベース・データ・ソースを使用します。スタンドアロン・サーバー・プロファイルでは、プロファイル作成時にファイル・ストアを使用するオプションが選択されている場合、データベース・データ・ソースの代わりにファイル・ストアが使用される点に注意してください。

Network Deployment 環境または管理対象ノード環境内の新規のサーバーとクラスターは、デフォルトでは SCA アプリケーションとそれらの宛先をホストするようには構成されません。

注: スタンドアロン・サーバーでは、SCA サポートが自動的に構成されます。この構成を使用不可にすることはできません。

このサポートを有効にするには、管理コンソールの「Service Component Architecture」ページを使用します。サーバーの場合、アプリケーション・クラス・ローダー・ポリシーが「複数」に設定されていることを確認します。

Network Deployment 環境または管理対象ノード環境内にあるサーバーまたはクラスターに対して、SCA サポートを使用可能にする前に、以下の可能な構成のいずれを実装するかを決定します。

- **リモート・バス・メンバーの構成:** サーバーまたはクラスターでは SCA アプリケーションがホストされますが、宛先はリモート・サーバーまたはリモート・クラスター上でホストされます。このシナリオでは、宛先をホストするために必要なメッセージング・エンジンを使って、リモート・サービス統合バス・メンバーを構成する必要があります。

リモート・メッセージングの使用には、サービス統合バスとそのメンバーの計画を立てて構成するための初期投資が必要になりますが、この構成はアプリケーション・クラスター内の複数のメンバーで再利用できます。メッセージは、すべてのメンバーに配布されます。また、フェイルオーバー・サポートを提供するように初期構成を構造化することもできます。

- **ローカル・バス・メンバーの構成:** サーバーまたはクラスターでは SCA アプリケーションおよびアプリケーション宛先の両方がホストされます。必要なメッセージング・エンジンは、サーバーまたはクラスター上のローカル・バス・メンバーを使用して構成されます。




計画の各トピックを参照して、ご使用の環境にいずれの構成が適しているかを判断してください。

関連タスク

107 ページの『サービス統合バスの表示』

サービス統合バスを表示させるには、管理コンソールの Service Integration Bus Browser を使用します。

関連情報

-  [サーバーのクラス・ローダーの構成](#)
-  [サービス統合バスについて](#)
-  [メッセージング・エンジン](#)

第 3 章 セル内での複数のプラットフォームの使用

綿密に計画することによって、分散オペレーティング・システム、i5/OS オペレーティング・システム、および z/OS オペレーティング・システムの各プラットフォームのノードが含まれたデプロイメント・マネージャー・セルを作成することができます。

例えば、i5/OS ノード、z/OS ノード、Linux[®] ノード、UNIX[®] ノード、および Windows[®] ノードが含まれたデプロイメント・マネージャー・セルを作成できます。このような構成は、異機種混合 セルと呼ばれます。

異機種混合セルでは、しっかりと計画を立てることが必要です。異機種混合セルのセットアップでは、タスクの一部を自動化できないため、余計に時間がかかる場合もあります。『Heterogeneous Cells – cells with nodes on mixed operating system platforms』 ホワイト・ペーパーに、異機種混合セルを作成するのに必要な計画およびシステムの考慮事項が概説されています。

管理コンソールを使用して新規サーバーを作成する場合は、サーバーの初期構成設定を提供するサーバー・テンプレート を選択します。サーバーを作成する管理対象ノードを選択した後、ユーザーは、そのノードのオペレーティング・システム・プラットフォームに使用できるテンプレートを管理コンソール上で選択できます。

重要: セルは異機種混合にすることができますが、z/OS ノードをサーバー・クラスター内の他のノードと混合することはできません。

関連概念

42 ページの『デプロイメント・マネージャー』

デプロイメント・マネージャーとは、他のサーバーの論理グループ (セル) の操作を管理するサーバーです。デプロイメント・マネージャーは、サーバーおよびクラスターを管理するための中央の場所になっています。

42 ページの『管理対象ノードの概要』

管理対象ノードとは、デプロイメント・マネージャー・セルに統合されているノードのことです。管理対象ノードでは、管理対象サーバーを構成して実行できません。

関連情報



Heterogeneous Cells - cells with nodes on mixed operating system platforms

第 4 章 デプロイメント環境の計画

デプロイメント環境のセットアップには、物理サーバーの数から選択するパターンのタイプまで、あらゆる事柄に影響を与える多くの決定が関係しています。それぞれの決定はデプロイメント環境をセットアップする方法に影響を与えます。

始める前に

以下の作業が完了したことを確認してください。

- 使用可能なリソースの識別
- データベース・タイプの選択
- 必要な権限の識別

このタスクについて

相互接続サーバーのレイアウトを計画する場合、いくつかの決定を下す必要があります。こうした決定は、使用可能なハードウェアと物理接続の間で行われるトレードオフ、管理および構成の複雑さ、およびパフォーマンス、可用性、スケーラビリティ、分離機能、セキュリティ、安定度などの要件に影響を与えます。

手順

1. デプロイメント環境の目的を決定します。
2. デプロイメント環境の機能要件を明確化します。
 - a. デプロイするコンポーネント・タイプを決定します。

コンポーネント・タイプとコンポーネント間の対話を要件の一部として検討します。

- b. インポートおよびエクスポートの実装タイプとトランスポートを決定します。

データベースまたは Java Message Service (JMS) リソースに必要なリソース、およびビジネス・イベントとそれらの伝送手段に必要な事柄について考慮します。

- c. アプリケーションに関連しないすべての機能要件も明確化します。

セキュリティ・サーバー、ルーター、およびビジネス・イベントを処理するための他のすべてのハードウェア要件またはソフトウェア要件を検討します。

3. ご使用の環境に対する容量とパフォーマンスの要件を明確化します。
4. 各機能に必要な物理サーバーの数を決定します。
5. ご使用の環境に対する冗長度の要件を明確化します。
 - a. フェイルオーバーに必要なサーバーの数を決定します。
 - b. 必要なルーターの数を決定します。

ルーターの選択は、デプロイされたモジュールのエクスポート、サービス統合バス上で定義するキューのタイプ、Service Component Architecture (SCA) エクスポート、およびクラスター間に適用するロード・バランシングのタイプに左右されます。IBM では、Web Services エクスポートに使用される組み込みルーターを、Service Object Access Protocol(SOAP)/JMS トランスポートまたは JMS エクスポートとともに提供しています。しかし、IBM によって提供されるこの組み込みルーターを使用しないことを選択する場合、使用するテクノロジーに基づいてクラスター間でのロード・バランシングの方法を決定する必要があります。

6. デプロイメント環境を設計します。

パターンを決定します。3 つの確立されたクラスター・パターンの中から選択できます。必要を満たすパターンがこれらの 3 つのうちでない場合、独自のカスタム・デプロイメント環境を作成できます。

- 単一クラスター
- リモート・メッセージング
- リモート・メッセージングおよびリモート・サポート

各パターンとそれらの間の違いについては、『デプロイメント環境パターン』を参照してください。

7. デプロイメント環境をインストールする方法を決定します。

管理コンソールのウィザードで、単一メッセージング、リモート・メッセージング、およびリモート・メッセージング・クラスターとリモート・サポート・クラスターをインストールできます。カスタム・デプロイメント環境は、管理コンソールのウィザードを使用するか、または管理コンソールによってその環境を自分で作成して、インストールすることができます。すべてまたは一部のインストールに、コマンド行またはサイレント・インストールを使用することもできます。

次のタスク

お客様の状況に最適な計画のシナリオを選択して、それに従ってください。

関連概念

39 ページの『スタンドアロン・サーバー』

スタンドアロン・サーバーは、Service Component Architecture (SCA) モジュールを 1 つのサーバー・プロセスにデプロイするための環境を提供します。このサーバー・プロセスには、管理コンソール、デプロイメント・ターゲット、メッセージング・サポート、ビジネス・ルール・マネージャー、および Common Event Infrastructure サーバーが含まれます (ほかのものが含まれる場合もあります)。

43 ページの『デプロイメント環境』

デプロイメント環境とは、Service Component Architecture (SCA) の対話をホストするための環境を共同して提供する、構成済みのクラスター、サーバー、およびミドルウェアの集合のことです。例えば、デプロイメント環境には、メッセージの宛先用のホスト、ビジネス・イベントの処理プログラム、および管理プログラムが組み込まれている場合があります。

46 ページの『デプロイメント環境パターン』

デプロイメント環境パターンは、デプロイメント環境に含まれるコンポーネントとリソースの制約と要件を指定します。デプロイメント環境パターンは、お客

様がデプロイメント環境を最も簡単な方法で作成できるよう支援することを目的とし、大半のビジネス要件を満たせるように設計されています。

45 ページの『デプロイメント環境のクラスター』

クラスターを使用すると、アプリケーションの能力と可用性が単一サーバーの場合よりも高まります。



WebSphere Process Server 用のサービス統合バス

サービス統合バスとは、同期および非同期メッセージングによってサービス統合をサポートする、管理された通信メカニズムです。バスは、バス・リソースを管理する相互接続メッセージング・エンジンで構成されます。サービス統合バスは、WebSphere Process Server の基盤となる WebSphere Application Server テクノロジーの 1 つです。



サービス・コンポーネント

IBM WebSphere Process Server 上で稼働するすべての統合成果物 (ビジネス・プロセス、ビジネス・ルール、ヒューマン・タスクなど) は、適切に定義されたインターフェースを持つコンポーネントとして表されます。

関連資料

105 ページの『デプロイメント環境の再始動』

リカバリー・プロセスの 1 ステップとして、デプロイメント環境の再始動が必要となる場合があります。

計画のためのシナリオ

デプロイメント環境を計画する方法は、デプロイメント環境を使用する方法によって決まります。以下の各シナリオをすべて読んで、デプロイメント環境を使用する方法に最もよく一致するシナリオを見つけてください。

WebSphere Integration Developer のインストール時に WebSphere Process Server もインストールする計画

このシナリオは、アプリケーション開発者が WebSphere Integration Developer を使用してデプロイメント環境にアクセスする場合で、デフォルトの構成が要件を満たす場合に使用します。

始める前に

WebSphere Integration Developer インフォメーション・センターで説明されているインストール処理についてよく理解しておいてください。そこに記述されている要件は、WebSphere Process Server の要件に対する追加の要件です。

このタスクについて

WebSphere Integration Developer をインストールする前に、WebSphere Process Server をインストールして、開発者がテスト・アプリケーションを使用するためのサーバーを提供することにメリットがあるかどうかを検討します。開発チームを最初からテスト機能を提供する環境に移動させることで、開発チームの生産性が迅速に高まる可能性があります。

小規模のテスト用のサーバーで要件を満たせる場合は、WebSphere Process Server を WebSphere Integration Developer と共にインストールすることを検討します。

手順

1. 開発環境とテスト環境を設計します。
 - a. WebSphere Integration Developer に対する要件を決定します。
 - b. テスト用のサーバーの要件を決定します。

開発チームと話し合っ、可用性、容量、およびセキュリティーに関するチームの意見を確認します。多くの場合、実稼働環境から隔離された単一サーバーで開発チームの使用には十分間に合います。
 - c. ターゲット・サーバーがこれらのニーズを十分満たすことが可能なハードウェアであることを確認します。
2. セキュリティー管理者に連絡して、インストールを完了するために必要なすべてのユーザー ID と権限を取得します。
3. オプション: サイト・ポリシーによってデータベースの作成と中央の部門へのアクセスが制限される場合は、担当のデータベース管理者に連絡してください。
4. WebSphere Integration Developer および WebSphere Process Server のインストールを、開発コミュニティへの影響を最小限に抑えるように調整してスケジュールします。

次のタスク

テスト用のサーバーをステップ 1 で指定したサーバーにインストールするオプションを選択して、ハードウェアと WebSphere Integration Developer をインストールし、その環境が予想通りに稼働していることを検査します。

関連概念

39 ページの『スタンドアロン・サーバー』

スタンドアロン・サーバーは、Service Component Architecture (SCA) モジュールを 1 つのサーバー・プロセスにデプロイするための環境を提供します。このサーバー・プロセスには、管理コンソール、デプロイメント・ターゲット、メッセージング・サポート、ビジネス・ルール・マネージャー、および Common Event Infrastructure サーバーが含まれます (ほかのものが含まれる場合もあります)。

39 ページの『サーバー』

サーバーによって、WebSphere Process Server のコア機能が提供されます。プロセス・サーバーでは、アプリケーション・サーバーの機能が拡大または拡張されて、Service Component Architecture (SCA) モジュールが処理されます。他のサーバー (デプロイメント・マネージャーおよびノード・エージェント) は、プロセス・サーバーの管理に使用されます。

関連タスク

 デフォルト値を使用したプロファイルの構成

プロファイル管理ツールを使用して、デフォルトの構成設定でプロファイルを作成または拡張する方法について説明します。

 既存のプロファイルの拡張

既存の WebSphere Application Server、WebSphere Application Server Network

Deployment、または WebSphere Application Server Network Deployment with Web Services Feature Pack プロファイルを WebSphere Enterprise Service Bus や WebSphere Process Server プロファイルに拡張できます。あるいは、WebSphere Enterprise Service Bus プロファイルを WebSphere Process Server プロファイルに拡張することもできます。このトピックの説明を使用してプロファイルを拡張します。プロファイルの拡張は、manageprofiles コマンドを使用してコマンド行から行うことも、プロファイル管理ツールのグラフィカル・ユーザー・インターフェース (GUI) を使用して対話式に行うこともできます。

関連資料

データベースのユーザーおよびスキーマ

WebSphere Process Server のインストール時に、データベースをインストールする際にデフォルトのスキーマ名およびユーザー ID 特権を使用することができます。ただし、データベース設計によっては、別のユーザー ID およびスキーマ名特権が必要になる場合があります。提供されている 3 つのシナリオを検討して、WebSphere Process Server のインストール時に別のスキーマ名およびユーザー ID 特権を構成するタイミングと方法を決定できます。

関連情報

ソフトウェアのインストール

WebSphere Process Server 製品の入手方法は 2 つあります。1 つは製品パッケージ内のディスクから入手する方法で、もう 1 つは Passport Advantage® サイトからインストール・イメージをダウンロードする方法です (この場合は、そのためのライセンスが必要です)。ソフトウェアをインストールするには、グラフィカル・インターフェース・モードまたはサイレント・モードでインストール・ウィザードを使用します。サイレント・モードでは、インストール・ウィザードはグラフィカル・インターフェースを表示せずに、応答ファイルから応答を読み取ります。

Business Process Choreographer の構成

WebSphere Integration Developer による使用のための WebSphere Process Server のインストール計画

このシナリオは、アプリケーション開発者が WebSphere Integration Developer を使用してデプロイメント環境にアクセスする場合で、デフォルトの構成が要件を満たさない場合に使用します。

始める前に

WebSphere Integration Developer インフォメーション・センターで説明されているインストール処理についてよく理解しておいてください。そこに記述されている要件は、WebSphere Process Server の要件に対する追加の要件です。

このタスクについて

この手順は、開発チームのテスト用のサーバーに対するニーズを満たすと想定されるサーバーが存在している場合に使用します。

このシナリオを使用する場合の例としては、以下のようなケースがあります。

- DB2 などのリモート・データベースを使用する場合。
- 特定のセキュリティー・リポジトリを使用する場合。
- 複数の環境でテストする場合。例えば、アプリケーションを製品の前のバージョンと現行バージョンの両方でテストする場合。

手順

1. 開発チームのニーズを調べます。
2. 開発環境を設計します。
3. テスト環境を設計します。 実働アプリケーション環境から隔離されたサーバーを使用します。テスト環境の隔離により、ビジネス・データの汚染が防止されま

ロケーション	考慮事項
開発用のサーバーとテスト用のサーバーが同じ	<ul style="list-style-type: none"> • このサーバーに、両方のワークロードに対応する能力があることを確認します。 • すべての開発者がこのサーバーにアクセスできることを確認します。 • WebSphere Integration Developer のインストールと同時に WebSphere Process Server をインストールすることを検討します。
開発用のサーバーとテスト用のサーバーが異なる	<ul style="list-style-type: none"> • 両方のサーバーが通信可能なことを確認します。 • すべての開発者がこのサーバーにアクセスできることを確認します。

4. セキュリティー管理者に連絡して、インストールを完了するために必要なすべてのユーザー ID と権限を取得します。
5. オプション: サイト・ポリシーによってデータベースの作成と中央の部門へのアクセスが制限される場合は、担当のデータベース管理者に連絡してください。
6. WebSphere Integration Developer および WebSphere Process Server のインストールを、開発コミュニティへの影響を最小限に抑えるように調整してスケジュールします。
7. WebSphere Process Server を選択したテスト用のサーバーにインストールします。
8. WebSphere Integration Developer を選択した開発用のサーバーにインストールします。

次のタスク

WebSphere Integration Developer を構成して、隔離したサーバーを使用します。

関連概念

39 ページの『スタンドアロン・サーバー』

スタンドアロン・サーバーは、Service Component Architecture (SCA) モジュールを 1 つのサーバー・プロセスにデプロイするための環境を提供します。このサーバー・プロセスには、管理コンソール、デプロイメント・ターゲット、メッセ

ージング・サポート、ビジネス・ルール・マネージャー、および Common Event Infrastructure サーバーが含まれます (ほかのものが含まれる場合もあります)。

39 ページの『サーバー』

サーバーによって、WebSphere Process Server のコア機能が提供されます。プロセス・サーバーでは、アプリケーション・サーバーの機能が拡大または拡張されて、Service Component Architecture (SCA) モジュールが処理されます。他のサーバー (デプロイメント・マネージャーおよびノード・エージェント) は、プロセス・サーバーの管理に使用されます。

関連タスク

製品のインストールの検査

WebSphere Process Server が正常にインストールされ、スタンドアロン・サーバー・プロファイルまたはデプロイメント・マネージャー・プロファイルが正常に作成されていることを確認するには、インストール検査ツールを使用します。プロファイルは、デプロイメント・マネージャーまたはサーバー用のランタイム環境を定義するファイルから構成されます。installver_wbi 検査合計ツールを使用してコア製品ファイルを検査します。インストール検査テスト (IVT) ツールを使用して、各プロファイルを検査します。

デフォルト値を使用したプロファイルの構成

プロファイル管理ツールを使用して、デフォルトの構成設定でプロファイルを作成または拡張する方法について説明します。

既存のプロファイルの拡張

既存の WebSphere Application Server、WebSphere Application Server Network Deployment、または WebSphere Application Server Network Deployment with Web Services Feature Pack プロファイルを WebSphere Enterprise Service Bus や WebSphere Process Server プロファイルに拡張できます。あるいは、WebSphere Enterprise Service Bus プロファイルを WebSphere Process Server プロファイルに拡張することもできます。このトピックの説明を使用してプロファイルを拡張します。プロファイルの拡張は、manageprofiles コマンドを使用してコマンド行から行うことも、プロファイル管理ツールのグラフィカル・ユーザー・インターフェース (GUI) を使用して対話式に行うこともできます。

アプリケーション・デプロイメント・ターゲット・クラスターの始動の確認

アプリケーション・デプロイメント・ターゲット・クラスターが始動可能なことを確認するには、デプロイメント環境の 3 つのクラスターをすべて始動する必要があります。このセクションでは、3 つのクラスターで構成されるデプロイメント環境についての事例を説明します。

関連資料

データベースのユーザーおよびスキーマ

WebSphere Process Server のインストール時に、データベースをインストールする際にデフォルトのスキーマ名およびユーザー ID 特権を使用することができます。ただし、データベース設計によっては、別のユーザー ID およびスキーマ名特権が必要になる場合があります。提供されている 3 つのシナリオを検討して、WebSphere Process Server のインストール時に別のスキーマ名およびユーザー ID 特権を構成するタイミングと方法を決定できます。

関連情報

ソフトウェアのインストール

WebSphere Process Server 製品の入手方法は 2 つあります。1 つは製品パッケージ内のディスクから入手する方法で、もう 1 つは Passport Advantage サイトからインストール・イメージをダウンロードする方法です (この場合は、そのためのライセンスが必要です)。ソフトウェアをインストールするには、グラフィカル・インターフェース・モードまたはサイレント・モードでインストール・ウィザードを使用します。サイレント・モードでは、インストール・ウィザードはグラフィカル・インターフェースを表示せずに、応答ファイルから応答を読み取ります。

Business Process Choreographer の構成

デフォルトのスタンドアロン環境の計画

このシナリオは、デプロイメント環境を他の環境から分離する必要がある場合に使用します。この環境で実行されるアプリケーションは、必要なものをそれ自体が完備し、Web サービス SOAP/HTTP などの限られたインポート・プロトコルを使用する必要があります。このシナリオは、インストールおよびセットアップの容易さが高可用性に対する要件よりも重要な場合にも使用します。

始める前に

- デプロイメント環境を設計します。
- すべてのビジネス要件を単一サーバーで満たすことができることを確認します。
- スタンドアロン・プロファイルの概念についてよく理解してください。

このタスクについて

ニーズを満たすためにデフォルトの単一のサーバー環境をインストールする必要がある設計があります。

手順

1. 設計をサポートするために必要なハードウェアおよびソフトウェアを決定します。
2. インストールを完了するために必要な権限を持つすべてのユーザー ID を特定または作成します。
3. オプション: サイト・ポリシーによってデータベースの作成と中央の部門へのアクセスが制限される場合は、担当のデータベース管理者に連絡してください。

重要: 将来の計画に、この環境をデプロイメント・マネージャー・セルに統合することが含まれている場合は、リモート・アクセスをサポートするデータベースとデータベース・ドライバーを使用していることを必ず確認してください。これらのタイプの製品の例としては、Derby Network と Java Toolbox JDBC があります。

4. WebSphere Integration Developer および WebSphere Process Server のインストールを、開発コミュニティへの影響を最小限に抑えるように調整してスケジュールします。

次のタスク

ソフトウェアをインストールします。

関連概念

39 ページの『スタンドアロン・サーバー』

スタンドアロン・サーバーは、Service Component Architecture (SCA) モジュールを 1 つのサーバー・プロセスにデプロイするための環境を提供します。このサーバー・プロセスには、管理コンソール、デプロイメント・ターゲット、メッセージング・サポート、ビジネス・ルール・マネージャー、および Common Event Infrastructure サーバーが含まれます (ほかのものが含まれる場合もあります)。

39 ページの『サーバー』

サーバーによって、WebSphere Process Server のコア機能が提供されます。プロセス・サーバーでは、アプリケーション・サーバーの機能が拡大または拡張されて、Service Component Architecture (SCA) モジュールが処理されます。他のサーバー (デプロイメント・マネージャーおよびノード・エージェント) は、プロセス・サーバーの管理に使用されます。



ハードウェアおよびソフトウェア要件

このトピックには、WebSphere Process Server のインストールに必要な、ハードウェア要件とソフトウェア相互要件および前提条件に関する追加情報へのリンクが含まれています。

関連タスク



製品のインストールの検査

WebSphere Process Server が正常にインストールされ、スタンドアロン・サーバー・プロファイルまたはデプロイメント・マネージャー・プロファイルが正常に作成されていることを確認するには、インストール検査ツールを使用します。プロファイルは、デプロイメント・マネージャーまたはサーバー用のランタイム環境を定義するファイルから構成されます。installver_wbi 検査合計ツールを使用してコア製品ファイルを検査します。インストール検査テスト (IVT) ツールを使用して、各プロファイルを検査します。



既存のプロファイルの拡張

既存の WebSphere Application Server、WebSphere Application Server Network Deployment、または WebSphere Application Server Network Deployment with Web Services Feature Pack プロファイルを WebSphere Enterprise Service Bus や WebSphere Process Server プロファイルに拡張できます。あるいは、WebSphere Enterprise Service Bus プロファイルを WebSphere Process Server プロファイルに拡張することもできます。このトピックの説明を使用してプロファイルを拡張します。プロファイルの拡張は、manageprofiles コマンドを使用してコマンド行から行うことも、プロファイル管理ツールのグラフィカル・ユーザー・インターフェース (GUI) を使用して対話式に行うこともできます。

関連資料




データベースのユーザーおよびスキーマ

WebSphere Process Server のインストール時に、データベースをインストールする際にデフォルトのスキーマ名およびユーザー ID 特権を使用することができます。ただし、データベース設計によっては、別のユーザー ID およびスキーマ名

特権が必要になる場合があります。提供されている 3 つのシナリオを検討して、WebSphere Process Server のインストール時に別のスキーマ名およびユーザー ID 特権を構成するタイミングと方法を決定できます。

関連情報

 Business Process Choreographer の構成

 ソフトウェアのインストール

WebSphere Process Server 製品の入手方法は 2 つあります。1 つは製品パッケージ内のディスクから入手する方法で、もう 1 つは Passport Advantage サイトからインストール・イメージをダウンロードする方法です (この場合は、そのためのライセンスが必要です)。ソフトウェアをインストールするには、グラフィカル・インターフェース・モードまたはサイレント・モードでインストール・ウィザードを使用します。サイレント・モードでは、インストール・ウィザードはグラフィカル・インターフェースを表示せずに、応答ファイルから応答を読み取ります。

カスタム・スタンドアロン環境の計画

このシナリオは、分離された環境を必要としているが、ビジネス要件のためにデフォルトの単一サーバー環境を使用できない場合に利用します。

始める前に

- デプロイメント環境を設計します。
- すべてのビジネス要件を単一サーバーで満たすことができることを確認します。
- スタンドアロン・プロファイルの概念についてよく理解してください。

このタスクについて

ニーズを満たすためにデフォルトの単一のサーバー環境をインストールする必要がある設計があります。

手順

1. デプロイメント環境をサポートするためのデータベース製品を選択します。

z/OS や i5/OS などの一部のシステムには、メッセージング・エンジンと Common Event Infrastructure (CEI) 用のデータベースとテーブルを作成するための自動化方法がありません。これらのシステム用のデータベースを作成する際には、データベース定義スクリプトを正常に実行するために必要な十分な権限があることを確認します。

重要: 将来の計画に、この環境をデプロイメント・マネージャー・セルに統合することが含まれている場合は、リモート・アクセスをサポートするデータベースとデータベース・ドライバーを使用していることを必ず確認してください。これらのタイプの製品の例としては、Derby Network と Java Toolbox JDBC があります。

2. データベース表の作成方法を決定します。

製品のインストール時にテーブルを作成するか、製品のインストール・プロセスにテーブルを作成するためのスクリプトを作成させるか、またはこの手順を実行するスクリプトを自分で作成します。

3. クライアントにデプロイメント環境内のアプリケーションにアクセスさせる方法を決定します。

アクセスの方法には、ユーザーのニーズに基づき、Web サービス (SOAP/HTTP と SOAP/JMS)、同期または非同期の Service Component Architecture (SCA) 要求、Java Message Service (JMS)、MQ (JMS またはネイティブ)、またはアダプターを介してなど多数の方法があります。これらの選択は、インストールする必要のある他のソフトウェアとリソースに影響を与えます。

4. アプリケーションで必要となるリソースにアプリケーションがアクセスする方法を決定します。

アクセスの方法には、ユーザーのニーズに基づき、Web サービス (SOAP/HTTP と SOAP/JMS)、同期または非同期の Service Component Architecture (SCA) 要求、Java Message Service (JMS)、MQ (JMS またはネイティブ)、またはアダプターを介してなど多数の方法があります。これらの選択は、インストールする必要のある他のソフトウェアとリソースに影響を与えます。

5. ソフトウェアのインストール方法、サーバーの作成および構成方法を決定します。

ソフトウェアのインストール時にサーバーを作成して構成でき、またプロファイル管理ツールを使用してサーバーを作成して構成することもできます。さらに、管理コンソールを使用してサーバーを作成して構成することもできます。経験を積んだインストール担当者は、これらのタスクを処理するためにスクリプトを使用することもできます。すべての方法の利点と欠点を理解してから、選択を行うようにしてください。

6. インストールを完了するために必要な権限を持つすべてのユーザー ID を特定または作成します。
7. オプション: サイト・ポリシーによってデータベースの作成と中央の部門へのアクセスが制限される場合は、担当のデータベース管理者に連絡してください。

重要: 将来の計画に、この環境をデプロイメント・マネージャー・セルに統合することが含まれている場合は、リモート・アクセスをサポートするデータベースとデータベース・ドライバーを使用していることを必ず確認してください。これらのタイプの製品の例としては、Derby Network と Java Toolbox JDBC があります。

8. WebSphere Integration Developer および WebSphere Process Server のインストールを、開発コミュニティへの影響を最小限に抑えるように調整してスケジュールします。

次のタスク

ソフトウェアをインストールします。

関連概念

39 ページの『スタンドアロン・サーバー』

スタンドアロン・サーバーは、Service Component Architecture (SCA) モジュール

を 1 つのサーバー・プロセスにデプロイするための環境を提供します。このサーバー・プロセスには、管理コンソール、デプロイメント・ターゲット、メッセージング・サポート、ビジネス・ルール・マネージャー、および Common Event Infrastructure サーバーが含まれます (ほかのものが含まれる場合もあります)。

39 ページの『サーバー』

サーバーによって、WebSphere Process Server のコア機能が提供されます。プロセス・サーバーでは、アプリケーション・サーバーの機能が拡大または拡張されて、Service Component Architecture (SCA) モジュールが処理されます。他のサーバー (デプロイメント・マネージャーおよびノード・エージェント) は、プロセス・サーバーの管理に使用されます。



ハードウェアおよびソフトウェア要件

このトピックには、WebSphere Process Server のインストールに必要な、ハードウェア要件とソフトウェア相互要件および前提条件に関する追加情報へのリンクが含まれています。

関連タスク



製品のインストールの検査

WebSphere Process Server が正常にインストールされ、スタンドアロン・サーバー・プロファイルまたはデプロイメント・マネージャー・プロファイルが正常に作成されていることを確認するには、インストール検査ツールを使用します。プロファイルは、デプロイメント・マネージャーまたはサーバー用のランタイム環境を定義するファイルから構成されます。installver_wbi 検査合計ツールを使用してコア製品ファイルを検査します。インストール検査テスト (IVT) ツールを使用して、各プロファイルを検査します。



既存のプロファイルの拡張

既存の WebSphere Application Server、WebSphere Application Server Network Deployment、または WebSphere Application Server Network Deployment with Web Services Feature Pack プロファイルを WebSphere Enterprise Service Bus や WebSphere Process Server プロファイルに拡張できます。あるいは、WebSphere Enterprise Service Bus プロファイルを WebSphere Process Server プロファイルに拡張することもできます。このトピックの説明を使用してプロファイルを拡張します。プロファイルの拡張は、manageprofiles コマンドを使用してコマンド行から行うことも、プロファイル管理ツールのグラフィカル・ユーザー・インターフェース (GUI) を使用して対話式に行うこともできます。

9 ページの『データベースの選択』

データベースの選択は、オペレーティング・システムによって、またご使用のフィーチャーによって異なります。インストール手順で、ウィザードからデータベースを選択するというプロンプトが出されます。場合によっては、複数のテーブルを含む 1 つのデータベースだけで操作することが可能です。

7 ページの『インストールする製品の決定』

デプロイメント環境の設計には、必要になる可能性があるソフトウェア製品の数とタイプの判断が含まれます。製品の要件は、お客様のニーズに基づき、その環境に関連するコンピューターシステムによって異なる可能性があります。デプロイメント環境のすべてのサーバーにそれぞれ WebSphere Process Server ライセンスが必要なわけではありません。

4 ページの『使用可能なリソースの明確化』

資産を明確化し、既に使用可能なリソースを最大限に利用するとともに、購買の決定の通知を受け取ります。

関連資料

データベースのユーザーおよびスキーマ

WebSphere Process Server のインストール時に、データベースをインストールする際にデフォルトのスキーマ名およびユーザー ID 特権を使用することができます。ただし、データベース設計によっては、別のユーザー ID およびスキーマ名特権が必要になる場合があります。提供されている 3 つのシナリオを検討して、WebSphere Process Server のインストール時に別のスキーマ名およびユーザー ID 特権を構成するタイミングと方法を決定できます。

関連情報

Business Process Choreographer の構成

ソフトウェアのインストール

WebSphere Process Server 製品の入手方法は 2 つあります。1 つは製品パッケージ内のディスクから入手する方法で、もう 1 つは Passport Advantage サイトからインストール・イメージをダウンロードする方法です (この場合は、そのためのライセンスが必要です)。ソフトウェアをインストールするには、グラフィカル・インターフェース・モードまたはサイレント・モードでインストール・ウィザードを使用します。サイレント・モードでは、インストール・ウィザードはグラフィカル・インターフェースを表示せずに、応答ファイルから応答を読み取ります。

提供されたパターンの 1 つに基づくデプロイメント環境の計画

このシナリオは、Service Component Architecture (SCA) アプリケーションに対するスケーラビリティ、可用性、およびサービス品質の要件がある場合で、IBM 提供のパターンの 1 つでそれらの要件を満たすことができる場合に使用します。

始める前に

これらのトピックと関連したトピックに関する情報についてよく理解してください (まだ理解していない場合)。

- サーバー
- クラスタ
- プロファイル
- データベースの選択
- デプロイメント環境
- デプロイメント環境の機能
- デプロイメント環境パターン

ご使用のデプロイメント環境で使用しているハードウェアの図表を作成し、各機器がホストするサーバーを示します。また、サーバーをクラスタ化する方法についてより明確な認識を持てるように、デプロイメント環境機能を提供するサーバーも明確化します。

このタスクについて

ビジネス・ニーズの分析を完了して、ニーズを満たすには単一サーバーでは不十分であることが判明しています。高可用性およびフェイルオーバーを提供するために、複数のサーバーを必要としています。お客様の設計が IBM 提供のデプロイメント環境パターンの 1 つに一致しています。

手順

1. 設計をサポートするために必要なハードウェアおよびソフトウェアを決定します。
2. デプロイメント環境をサポートするためのデータベース製品を選択します。

z/OS や i5/OS などの一部のシステムには、メッセージング・エンジンと Common Event Infrastructure (CEI) 用のデータベースとテーブルを作成するための自動化方法がありません。これらのシステム用のデータベースを作成するには、データベース定義スクリプトを正常に実行するために必要な十分な権限があることを確認します。

重要: 将来の計画に、この環境をデプロイメント・マネージャー・セルに統合することが含まれている場合は、リモート・アクセスをサポートするデータベースとデータベース・ドライバーを使用していることを必ず確認してください。これらのタイプの製品の例としては、Derby Network と Java Toolbox JDBC があります。

3. データベース表の作成方法を決定します。

製品のインストール時にテーブルを作成するか、製品のインストール・プロセスにテーブルを作成するためのスクリプトを作成させるか、またはこの手順を実行するスクリプトを自分で作成します。

4. IBM 提供のいずれのパターンが設計に最適かを判断します。
5. 各サーバーを、設計で指定した機能を提供するクラスターのメンバーとしてマップします。

選択したパターンにより、ノードがクラスターにマップされ、メンバー数とそれらの配分が決まります。

6. クライアントにデプロイメント環境内のアプリケーションにアクセスさせる方法を決定します。

アクセスの方法には、ユーザーのニーズに基づき、Web サービス (SOAP/HTTP と SOAP/JMS)、同期または非同期の Service Component Architecture (SCA) 要求、Java Message Service (JMS)、MQ (JMS またはネイティブ)、またはアダプターを介してなど多数の方法があります。これらの選択は、インストールする必要がある他のソフトウェアとリソースに影響を与えます。

7. アプリケーションで必要となるリソースにアプリケーションがアクセスする方法を決定します。

アクセスの方法には、ユーザーのニーズに基づき、Web サービス (SOAP/HTTP と SOAP/JMS)、同期または非同期の Service Component Architecture (SCA) 要求、Java Message Service (JMS)、MQ (JMS またはネイティブ)、またはアダプ

ターを介してなど多数の方法があります。これらの選択は、インストールする必要のある他のソフトウェアとリソースに影響を与えます。

8. ソフトウェアのインストール方法、サーバーの作成方法、および作成したサーバーの構成方法を決定します。

ソフトウェアのインストール時にサーバーを作成して構成でき、またプロファイル管理ツールを使用してサーバーを作成して構成することもできます。さらに、管理コンソールまたはスクリプトを使用してサーバーを作成して構成することもできます。すべての方法の利点と欠点を理解してから、選択を行うようにしてください。

9. 同じハードウェア上に作成するすべてのサーバーがそのシステム上のリソースを共有する方法を決定します。

ソフトウェアを別々の場所にインストールでき、また異なるプロファイルを使用することもでき、さらに i5/OS の場合は、異なるロジカル・パーティションを使用してこの共有を達成することもできます。

10. インストールを完了するために必要な権限を持つすべてのユーザー ID を特定または作成します。

次のタスク

デプロイメント環境をインストールします。

関連概念

39 ページの『サーバー』

サーバーによって、WebSphere Process Server のコア機能が提供されます。プロセス・サーバーでは、アプリケーション・サーバーの機能が拡大または拡張されて、Service Component Architecture (SCA) モジュールが処理されます。他のサーバー (デプロイメント・マネージャーおよびノード・エージェント) は、プロセス・サーバーの管理に使用されます。

42 ページの『デプロイメント・マネージャー』

デプロイメント・マネージャーとは、他のサーバーの論理グループ (セル) の操作を管理するサーバーです。デプロイメント・マネージャーは、サーバーおよびクラスターを管理するための中央の場所になっています。

45 ページの『管理対象サーバー』

管理対象サーバーとは、管理対象ノード内に構成されるサーバーのことです。管理対象サーバーにより、アプリケーションが実行されるデプロイメント環境内に特定のリソースが提供されます。

45 ページの『デプロイメント環境のクラスター』

クラスターを使用すると、アプリケーションの能力と可用性が単一サーバーの場合よりも高まります。

46 ページの『デプロイメント環境パターン』

デプロイメント環境パターンは、デプロイメント環境に含まれるコンポーネントとリソースの制約と要件を指定します。デプロイメント環境パターンは、お客様がデプロイメント環境を最も簡単な方法で作成できるよう支援することを目的とし、大半のビジネス要件を満たせるように設計されています。

50 ページの『デプロイメント環境機能』

堅固なデプロイメント環境を設計するためには、各クラスターが特定の IBM 提

供のデプロイメント環境パターンまたはカスタム・デプロイメント環境に提供できる機能について理解しておく必要があります。この知識は、ニーズに最も合うデプロイメント環境パターンを正しく判別するのに役立ちます。

カスタム・デプロイメント環境のレイアウト構成

この概要では、カスタム・デプロイメント環境の構成における 2 つの主要な考慮事項について説明します。環境で使用するクラスターおよび単一サーバーの選択と、デプロイメント環境構成の指定です。これらの考慮事項を理解すれば、デプロイメント環境を効率的に計画および実装できます。

6 ページの『WebSphere Process Server と他の WebSphere Application Server 製品との間のインターオペラビリティの計画』

ソフトウェア環境を分析するときは、デプロイメント環境内に存在するさまざまなソフトウェア・レベル間で要求を受け渡すことができるかどうかを把握する必要があります。

74 ページの『エラー処理方針とソリューション・リカバリー』

WebSphere Process Serverには、リカバリーのために利用できるエラー処理機能とツールが組み込まれています。

77 ページの『実稼働環境でのリカバリー』

実稼働環境での目標は、整然とした一貫性のある方法でシステムに入力された要求すべてを処理することです。この環境ではデータを保持する必要があり、システムを使用できなったり、データを損失したりする状況を最小限に抑えるための手段をすべて実施する必要があります。

関連タスク

17 ページの『第 4 章 デプロイメント環境の計画』

デプロイメント環境のセットアップには、物理サーバーの数から選択するパターンのタイプまで、あらゆる事柄に影響を与える多くの決定が関係しています。それぞれの決定はデプロイメント環境をセットアップする方法に影響を与えます。

9 ページの『データベースの選択』

データベースの選択は、オペレーティング・システムによって、またご使用のフィーチャーによって異なります。インストール手順で、ウィザードからデータベースを選択するというプロンプトが出されます。場合によっては、複数のテーブルを含む 1 つのデータベースだけで操作することが可能です。

4 ページの『使用可能なリソースの明確化』

資産を明確化し、既に使用可能なリソースを最大限に利用するとともに、購買の決定の通知を受け取ります。

7 ページの『インストールする製品の決定』

デプロイメント環境の設計には、必要になる可能性があるソフトウェア製品の数とタイプの判断が含まれます。製品の要件は、お客様のニーズに基づき、その環境に関連するコンピューターシステムによって異なる可能性があります。デプロイメント環境のすべてのサーバーにそれぞれ WebSphere Process Server ライセンスが必要なわけではありません。

関連資料

データベースのユーザーおよびスキーマ

WebSphere Process Server のインストール時に、データベースをインストールする際にデフォルトのスキーマ名およびユーザー ID 特権を使用することができます

す。ただし、データベース設計によっては、別のユーザー ID およびスキーマ名特権が必要になる場合があります。提供されている 3 つのシナリオを検討して、WebSphere Process Server のインストール時に別のスキーマ名およびユーザー ID 特権を構成するタイミングと方法を決定できます。

関連情報

 Network Deployment のインストールの計画

 概要: クラスタ

 Business Process Choreographer の構成

カスタム・デプロイメント環境の計画

このシナリオは、サービス品質要件があるか、または IBM 提供のパターンで定義されたデプロイメント環境よりも複雑なデプロイメント環境が必要な場合に使用します。

始める前に

重要: カスタム・デプロイメント環境のインストールは、デフォルトのデプロイメント環境のインストールよりも複雑であり、Network Deployment、クラスタ化、および他の WebSphere Process Server の機能の理解が必要になります。IBM では、デプロイメント環境の各部分を個々に計画して、段階的に実装することをお勧めします。

これらのトピックと関連したトピックに関する情報についてよく理解してください (まだ理解していない場合)。

- サーバー
- クラスタ
- プロファイル
- カスタム・デプロイメント環境とそれらの機能
- Business Process Choreographer コンポーネントおよび構成

ご使用のデプロイメント環境で使用しているハードウェアの図表を作成し、各機器がホストするサーバーを示します。また、サーバーをクラスタ化する方法についてより明確な認識を持てるように、デプロイメント環境機能を提供するサーバーも明確化します。

設計では、どのクラスタがメッセージング、Common Event Infrastructure、およびアプリケーション・サポートをデプロイメント環境に提供するかを指定する必要があります。

このタスクについて

以下のステップは、お客様の設計が IBM 提供のパターンのいずれにも一致しないか、または既存のデプロイメント環境を拡張する場合に使用します。すべての複雑さを最小限に抑えるために、一度にデプロイメント環境の 1 つの部分の追加、構成、および検証を行うだけになるように、反復手法の使用を検討してください。

手順

1. デプロイメント環境をサポートするためのデータベース製品を選択します。

z/OS や i5/OS などの一部のシステムには、メッセージング・エンジンと Common Event Infrastructure (CEI) 用のデータベースとテーブルを作成するための自動化方法がありません。これらのシステム用のデータベースを作成する際には、データベース定義スクリプトを正常に実行するために必要な十分な権限があることを確認します。

重要: 将来の計画に、この環境をデプロイメント・マネージャー・セルに統合することが含まれている場合は、リモート・アクセスをサポートするデータベースとデータベース・ドライバを使用していることを必ず確認してください。これらのタイプの製品の例としては、Derby Network と Java Toolbox JDBC があります。

2. データベース表の作成方法を決定します。

製品のインストール時にテーブルを作成するか、製品のインストール・プロセスにテーブルを作成するためのスクリプトを作成させるか、またはこの手順を実行するスクリプトを自分で作成します。

3. このデプロイメント環境にデプロイするアプリケーションを分析して、それらのアプリケーションのサポートに必要なクラスターを決定します。
4. デプロイメント環境の物理レイアウトを設計します。
5. 各サーバーを、設計で指定した機能を提供するクラスターのメンバーとしてマップします。

デプロイメント環境により提供される機能、およびどのノードが各クラスターに関連するかを決定します。

6. クライアントにデプロイメント環境内のアプリケーションにアクセスさせる方法を決定します。

アクセスの方法には、ユーザーのニーズに基づき、Web サービス (SOAP/HTTP と SOAP/JMS)、同期または非同期の Service Component Architecture (SCA) 要求、Java Message Service (JMS)、MQ (JMS またはネイティブ)、またはアダプターを介してなど多数の方法があります。これらの選択は、インストールする必要のある他のソフトウェアとリソースに影響を与えます。

7. アプリケーションで必要となるリソースにアプリケーションがアクセスする方法を決定します。

アクセスの方法には、ユーザーのニーズに基づき、Web サービス (SOAP/HTTP と SOAP/JMS)、同期または非同期の Service Component Architecture (SCA) 要求、Java Message Service (JMS)、MQ (JMS またはネイティブ)、またはアダプターを介してなど多数の方法があります。これらの選択は、インストールする必要のある他のソフトウェアとリソースに影響を与えます。

8. ソフトウェアのインストール方法、サーバーの作成方法、および作成したサーバーの構成方法を決定します。

制約事項: 単一のセル内のカスタム・デプロイメント環境の場合、インストーラーまたはプロファイル管理ツールを使用してサーバーを作成することはできません。

9. インストールを完了するために必要な権限を持つすべてのユーザー ID を特定または作成します。
10. オプション: サイト・ポリシーによってデータベースの作成と中央の部門へのアクセスが制限される場合は、担当のデータベース管理者に連絡してください。

重要: 将来の計画に、この環境をデプロイメント・マネージャー・セルに統合することが含まれている場合は、リモート・アクセスをサポートするデータベースとデータベース・ドライバーを使用していることを必ず確認してください。これらのタイプの製品の例としては、Derby Network と Java Toolbox JDBC があります。

11. WebSphere Integration Developer および WebSphere Process Server のインストーラーを、開発コミュニティへの影響を最小限に抑えるように調整してスケジュールします。

次のタスク

デプロイメント環境をインストールします。

関連概念

39 ページの『サーバー』

サーバーによって、WebSphere Process Server のコア機能が提供されます。プロセス・サーバーでは、アプリケーション・サーバーの機能が拡大または拡張されて、Service Component Architecture (SCA) モジュールが処理されます。他のサーバー (デプロイメント・マネージャーおよびノード・エージェント) は、プロセス・サーバーの管理に使用されます。

42 ページの『デプロイメント・マネージャー』

デプロイメント・マネージャーとは、他のサーバーの論理グループ (セル) の操作を管理するサーバーです。デプロイメント・マネージャーは、サーバーおよびクラスターを管理するための中央の場所になっています。

45 ページの『管理対象サーバー』

管理対象サーバーとは、管理対象ノード内に構成されるサーバーのことです。管理対象サーバーにより、アプリケーションが実行されるデプロイメント環境内に特定のリソースが提供されます。

45 ページの『デプロイメント環境のクラスター』

クラスターを使用すると、アプリケーションの能力と可用性が単一サーバーの場合よりも高まります。

50 ページの『デプロイメント環境機能』

堅固なデプロイメント環境を設計するためには、各クラスターが特定の IBM 提供のデプロイメント環境パターンまたはカスタム・デプロイメント環境に提供できる機能について理解しておく必要があります。この知識は、ニーズに最も合うデプロイメント環境パターンを正しく判別するのに役立ちます。

カスタム・デプロイメント環境のレイアウト構成

この概要では、カスタム・デプロイメント環境の構成における 2 つの主要な考慮事項について説明します。環境で使用するクラスターおよび単一サーバーの選

択と、デプロイメント環境構成の指定です。これらの考慮事項を理解すれば、デプロイメント環境を効率的に計画および実装できます。

6 ページの『WebSphere Process Server と他の WebSphere Application Server 製品との間のインターオペラビリティの計画』

ソフトウェア環境を分析するときは、デプロイメント環境内に存在するさまざまなソフトウェア・レベル間で要求を受け渡すことができるかどうかを把握する必要があります。

関連タスク

17 ページの『第 4 章 デプロイメント環境の計画』

デプロイメント環境のセットアップには、物理サーバーの数から選択するパターンのタイプまで、あらゆる事柄に影響を与える多くの決定が関係しています。それぞれの決定はデプロイメント環境をセットアップする方法に影響を与えます。

9 ページの『データベースの選択』

データベースの選択は、オペレーティング・システムによって、またご使用のフィーチャーによって異なります。インストール手順で、ウィザードからデータベースを選択するというプロンプトが出されます。場合によっては、複数のテーブルを含む 1 つのデータベースだけで操作することが可能です。


4 ページの『使用可能なリソースの明確化』

資産を明確化し、既に使用可能なリソースを最大限に利用するとともに、購買の決定の通知を受け取ります。

7 ページの『インストールする製品の決定』

デプロイメント環境の設計には、必要になる可能性があるソフトウェア製品の数とタイプの判断が含まれます。製品の要件は、お客様のニーズに基づき、その環境に関連するコンピューターシステムによって異なる可能性があります。デプロイメント環境のすべてのサーバーにそれぞれ WebSphere Process Server ライセンスが必要なわけではありません。


関連資料

 データベースのユーザーおよびスキーマ

WebSphere Process Server のインストール時に、データベースをインストールする際にデフォルトのスキーマ名およびユーザー ID 特権を使用することができます。ただし、データベース設計によっては、別のユーザー ID およびスキーマ名特権が必要になる場合があります。提供されている 3 つのシナリオを検討して、WebSphere Process Server のインストール時に別のスキーマ名およびユーザー ID 特権を構成するタイミングと方法を決定できます。

関連情報

 Network Deployment のインストールの計画

 概要: クラスタ

 Business Process Choreographer の構成

プロフィール

プロフィールでは、個別のコマンド・ファイル、構成ファイル、ログ・ファイルを持つ固有のランタイム環境を定義します。プロフィールでは、スタンドアロン・サーバー、デプロイメント・マネージャー、および管理対象ノードの 3 つのタイプの環境を定義します。

プロフィールを使用すると、WebSphere Process Server バイナリー・ファイルの複数のコピーをインストールしなくても、1 つのシステムに複数のランタイム環境を保持することができます。

最初のプロフィールは、WebSphere Process Server のインストール時に自動的に作成されます。後で、バイナリー・ファイルの 2 つめのコピーをインストールしなくても、プロフィール管理ツールまたは `manageprofiles` コマンドを使用して、同じシステム上に追加のプロフィールを作成することができます。

注: 分散プラットフォームでは、各プロフィールには固有の名前があります。z/OS では、すべてのプロフィールに「default」という名前が付けられます。

プロフィール・ディレクトリー

システム内の各プロフィールには、それぞれのファイルをすべて収容するための独自のディレクトリーがあります。プロフィールの作成時に、プロフィール・ディレクトリーの場所を指定します。デフォルトでは、WebSphere Process Server がインストールされたディレクトリーの `profiles` ディレクトリーになります。例: Dmgr01 プロフィールは `C:\Program Files\IBM\WebSphere\ProcServer\profiles\Dmgr01` です。

ファースト・ステップ・コンソール

Linux **UNIX** **Windows** **i5/OS** システム内のすべてのプロフィールには、ファースト・ステップ・コンソールがあります。このコンソールは、スタンドアロン・サーバー、デプロイメント・マネージャー、または管理対象ノードを熟知するためのユーザー・インターフェースです。

デフォルト・プロフィール

WebSphere Process Server の 1 つのインストール環境内に作成する最初のプロフィールは、デフォルト・プロフィールです。デフォルト・プロフィールは、WebSphere Process Server がインストールされたディレクトリー内の `\bin` ディレクトリーから出されるコマンドのデフォルトのターゲットです。システム上にプロフィールが 1 つしかない場合は、すべてのコマンドがそのプロフィールに対して作用します。プロフィールをもう 1 つ作成すると、そのプロフィールをデフォルトにすることができます。

注: デフォルト・プロフィールは、必ずしも「default」という名前のプロフィールではあるとは限りません。

プロファイルの拡張

WebSphere Application Server Network Deployment または WebSphere ESB 用に作成されたデプロイメント・マネージャー、カスタム・プロファイル、またはスタンドアロン・サーバーが既にある場合は、既存の機能のほかに WebSphere Process Server をサポートするように、そのプロファイルを拡張 できます。プロファイルを拡張するには、最初に WebSphere Process Server をインストールします。次にプロファイル管理ツールまたは `manageprofiles` コマンドを使用します。

制約事項: プロファイルが、既にデプロイメント・マネージャーに統合済みの管理対象ノードを定義する場合は、そのプロファイルを拡張できません。

関連概念

39 ページの『スタンドアロン・サーバー』

スタンドアロン・サーバーは、Service Component Architecture (SCA) モジュールを 1 つのサーバー・プロセスにデプロイするための環境を提供します。このサーバー・プロセスには、管理コンソール、デプロイメント・ターゲット、メッセージング・サポート、ビジネス・ルール・マネージャー、および Common Event Infrastructure サーバーが含まれます (ほかのものが含まれる場合もあります)。

42 ページの『デプロイメント・マネージャー』

デプロイメント・マネージャーとは、他のサーバーの論理グループ (セル) の操作を管理するサーバーです。デプロイメント・マネージャーは、サーバーおよびクラスターを管理するための中央の場所になっています。

42 ページの『管理対象ノードの概要』

管理対象ノードとは、デプロイメント・マネージャー・セルに統合されているノードのことです。管理対象ノードでは、管理対象サーバーを構成して実行できません。

複数プロファイル環境のプロファイル・コマンド

サーバーに複数のプロファイルが存在する場合、特定のコマンドでは、コマンドが適用するプロファイルを指定する必要があります。これらのコマンドでは、`-profileName` 属性を使用して、アドレス指定するプロファイルを指定します。コマンドごとに `-profileName` 属性を指定する必要をなくするためには、各プロファイルの `bin` ディレクトリーに存在するバージョンのコマンドを使用します。

関連タスク

プロファイル管理ツールを使用したプロファイルの作成

プロファイル管理ツールのグラフィカル・ユーザー・インターフェース (GUI) を使用して、スタンドアロン・サーバー・プロファイル、デプロイメント・マネージャー・プロファイル、またはカスタム・プロファイルを作成します。

`manageprofiles` コマンドを使用したプロファイルの作成

`manageprofiles` コマンドおよびプロパティ・ファイルを使用してコマンド行からプロファイルを作成する方法について説明します。

関連情報

ファースト・ステップ・コンソールの開始

WebSphere Process Server をインストールしたら、ファースト・ステップ・コンソールを使用して、製品ツールの開始、製品資料へのアクセス、個別プロファイ

ルに関連するサーバーおよび管理コンソールなどのエレメントへの指示を行います。汎用バージョンのコンソールと、インストール内のプロファイルごとのバージョンが使用可能です。

サーバー

サーバーによって、WebSphere Process Server のコア機能が提供されます。プロセス・サーバーでは、アプリケーション・サーバーの機能が拡大または拡張されて、Service Component Architecture (SCA) モジュールが処理されます。他のサーバー (デプロイメント・マネージャーおよびノード・エージェント) は、プロセス・サーバーの管理に使用されます。

プロセス・サーバーは、スタンドアロン・サーバー または管理対象サーバー のいずれかです。管理対象サーバーは、必要に応じてクラスター のメンバーになることが可能です。管理対象サーバー、サーバーのクラスター、および他のミドルウェアの集合を、デプロイメント環境 と呼びます。デプロイメント環境では、管理対象サーバーまたはクラスターが、それぞれデプロイメント環境内の特定の機能 (例えば、宛先ホスト、アプリケーション・モジュール・ホスト、または Common Event Infrastructure サーバー) に構成されます。スタンドアロン・サーバーは、必要なすべての機能を提供するように構成されます。

サーバーによって、Service Component Architecture (SCA) モジュール、それらのモジュールで使用されるリソース (データ・ソース、アクティベーション・スペック、および JMS 宛先)、および IBM 提供のリソース (メッセージ宛先、Business Process Choreographer Container、および Common Event Infrastructure サーバー) にランタイム環境が提供されます。

ノード・エージェント とは、システムに対するノードを表し、そのノード上のサーバーを管理する管理エージェントのことです。ノード・エージェントによって、ホスト・システム上のサーバーがモニターされ、管理要求がサーバーに送付されます。ノード・エージェントは、ノードがデプロイメント・マネージャーに統合されると作成されます。

デプロイメント・マネージャー とは、複数のサーバーとクラスターに一元管理ビューを提供する管理エージェントのことです。

スタンドアロン・サーバーはスタンドアロン・プロファイルによって定義され、デプロイメント・マネージャーはデプロイメント・マネージャー・プロファイルによって定義され、管理対象サーバーは管理対象ノード 内に作成され、管理対象ノードはカスタム・プロファイルによって定義されます。

スタンドアロン・サーバー

スタンドアロン・サーバーは、Service Component Architecture (SCA) モジュールを 1 つのサーバー・プロセスにデプロイするための環境を提供します。このサーバー・プロセスには、管理コンソール、デプロイメント・ターゲット、メッセージング・サポート、ビジネス・ルール・マネージャー、および Common Event Infrastructure サーバーが含まれます (ほかのものが含まれる場合もあります)。

スタンドアロン・サーバーはセットアップが容易であり、ファースト・ステップ・コンソールを備えています。このコンソールからはスタンドアロン・サーバーの開始および停止が可能であり、サンプル・ギャラリーおよび管理コンソールを開くことができます。WebSphere Process Server サンプルをインストールし、サンプル・ギャラリーを開くと、サンプル・ソリューションがスタンドアロン・サーバーにデプロイされます。このサンプルに使用されているリソースは、管理コンソールで探索できます。

ユーザー独自のソリューションをスタンドアロン・サーバーにデプロイすることは可能ですが、スタンドアロン・サーバーでは、実稼働環境で一般的に必要とされる容量、スケーラビリティ、または頑強性を提供できません。実稼働環境では、Network Deployment 環境を使用する方が適切です。

最初はスタンドアロン・サーバーから始めて、後でそれを Network Deployment 環境に取り込むことは可能です。このためには、スタンドアロン・サーバーをデプロイメント・マネージャー・セルに統合しますが、このセルに他のノードが統合されていないことが前提になります。1つのセルに複数のスタンドアロン・サーバーを統合することはできません。スタンドアロン・サーバーを統合するには、デプロイメント・マネージャーの管理コンソールまたは **addNode** コマンドを使用します。addNode コマンドを使用してスタンドアロン・サーバーを統合する場合は、スタンドアロン・サーバーが稼働してはなりません。

スタンドアロン・サーバーは、スタンドアロン・サーバー・プロファイルで定義されています。


関連概念

37 ページの『プロファイル』

プロファイルでは、個別のコマンド・ファイル、構成ファイル、ログ・ファイルを持つ固有のランタイム環境を定義します。プロファイルでは、スタンドアロン・サーバー、デプロイメント・マネージャー、および管理対象ノードの3つのタイプの環境を定義します。

 メッセージング宛先ホストまたはキュー宛先ホスト

メッセージング宛先ホストまたはキュー宛先ホストにより、サーバー内にメッセージング機能が提供されます。サーバーをメッセージング・ターゲットとして構成すると、サーバーはメッセージング宛先ホストになります。


 データ・ソース

データ・ソースは、アプリケーションとリレーショナル・データベースの間のリンクを提供します。

 WebSphere Process Server 用のサービス統合バス

サービス統合バスとは、同期および非同期メッセージングによってサービス統合をサポートする、管理された通信メカニズムです。バスは、バス・リソースを管理する相互接続メッセージング・エンジンで構成されます。サービス統合バスは、WebSphere Process Server の基盤となる WebSphere Application Server テクノロジーの1つです。

関連タスク

 スタンドアロン・サーバー・プロファイルのデプロイメント・マネージャーへの統合

addNode コマンドを使用して、スタンドアロン・サーバー・プロファイルをデプロイメント・マネージャー・セルに統合する方法を学習します。統合の後に、ノード・プロセス・エージェント・プロセスが作成されます。このノード・エージェントおよびサーバー・プロセスの両方とも、デプロイメント・マネージャーにより管理されます。スタンドアロン・サーバー・プロファイルを統合し、サーバー・アプリケーションをすべて組み込むと、統合動作によりデプロイメント・マネージャーにアプリケーションがインストールされます。スタンドアロン・サーバー・プロファイルは、他に統合されたプロファイルがない場合にのみ統合できます。

Network Deployment

Network Deployment では、容量、スケーラビリティ、および一般に実稼働環境に要求される頑強性が提供されます。Network Deployment では、ワークロード・バランシングとフェイルオーバーを提供するために、サーバーのグループを共同して使用することができます。各サーバーは、単一の管理コンソールを使用して一元管理されます。

WebSphere Process Server 内の Network Deployment は、WebSphere Application Server Network Deployment に実装された Network Deployment 機能をベースにしています。WebSphere Application Server Network Deployment での Network Deployment に詳しい場合は、その概念は同じなので理解しやすいはずですが。WebSphere Process Server では、デプロイメント環境の概念が Network Deployment に加わります。

Network Deployment に関して読んでおく必要があることは、WebSphere Application Server Network Deployment をアップグレードするのか、それとも WebSphere Application Server Network Deployment に関する経験がない状態で WebSphere Process Server を実装するのかによって異なります。

WebSphere Application Server Network Deployment のアップグレード

WebSphere Application Server Network Deployment では、その名前が示すように、アプリケーションの Network Deployment がサポートされます。WebSphere Process Server でアップグレードする WebSphere Application Server Network Deployment のインストール済み環境が既に存在する場合は、Network Deployment の概念に詳しいことでしょう。それぞれにデプロイメント・マネージャーと管理対象ノードを持つ Network Deployment セルが、おそらく 1 つ以上存在していることでしょう。WebSphere Process Server のプロファイル管理ツールを使用して、それらのプロファイルを拡張して、WebSphere Process Server をサポートすることができます。拡張の後も、サーバーは引き続きアプリケーション・サーバーとして機能しますが、それらは Service Component Architecture (SCA) モジュールをサポートすることも可能です。

WebSphere Process Server の Network Deployment の実装

Network Deployment では、WebSphere Process Server を 1 つ以上のホスト・システムにインストールした後、デプロイメント環境を作成します。IBM では、クラスター、サーバー、および Service Component Architecture (SCA) モジュールをホストするために必要なミドルウェアを構成するために役立ついくつかのデプロイメント環境パターンを提供しています。

関連情報

 WebSphere Application Server Network Deployment と単一サーバー (すべてのオペレーティング・システム) のインフォメーション・センター

デプロイメント・マネージャー

デプロイメント・マネージャーとは、他のサーバーの論理グループ (セル) の操作を管理するサーバーです。デプロイメント・マネージャーは、サーバーおよびクラスターを管理するための中央の場所になっています。

デプロイメント環境を作成する場合、最初に作成するプロファイルは、デプロイメント・マネージャー・プロファイルです。デプロイメント・マネージャーには、ファースト・ステップ・コンソールがあります。このコンソールからは、デプロイメント・マネージャーの開始および停止が可能であり、その管理コンソールを開始できます。セル内のサーバーおよびクラスターを管理するには、デプロイメント・マネージャーの管理コンソールを使用します。管理作業には、サーバーおよびクラスターの構成、クラスターへのサーバーの追加、サーバーおよびクラスターの開始および停止、およびサーバーおよびクラスターへの Service Component Architecture (SCA) モジュールのデプロイが含まれます。

デプロイメント・マネージャーは一種のサーバーですが、デプロイメント・マネージャー自体にモジュールをデプロイすることはできません。

関連概念

15 ページの『第 3 章 セル内での複数のプラットフォームの使用』

綿密に計画することによって、分散オペレーティング・システム、i5/OS オペレーティング・システム、および z/OS オペレーティング・システムの各プラットフォームのノードが含まれたデプロイメント・マネージャー・セルを作成することができます。

37 ページの『プロファイル』

プロファイルでは、個別のコマンド・ファイル、構成ファイル、ログ・ファイルを持つ固有のランタイム環境を定義します。プロファイルでは、スタンドアロン・サーバー、デプロイメント・マネージャー、および管理対象ノードの 3 つのタイプの環境を定義します。

管理対象ノードの概要

管理対象ノードとは、デプロイメント・マネージャー・セルに統合されているノードのことです。管理対象ノードでは、管理対象サーバーを構成して実行できます。

管理対象ノード上で構成されているサーバーが、デプロイメント環境のリソースを形成しています。これらのサーバーの作成、構成、開始、停止、管理、および削除は、デプロイメント・マネージャーの管理コンソールを使用して行われます。ノー

ドが統合されると、ノード・エージェント・プロセスが自動的に作成されます。プロファイルの構成を管理できるようにするには、このノード・エージェントが稼働していなければなりません。例えば、以下の作業を実行する場合などです。

- サーバー・プロセスの開始および停止
- デプロイメント・マネージャー上の構成データとノード上のコピーとの同期化

ただし、アプリケーションがノード内のリソースを実行または構成するようにしたい場合は、ノード・エージェントが稼働している必要はありません。

管理対象ノードには 1 つ以上のサーバーを配置できます。それらのサーバーは、デプロイメント・マネージャーによって管理されます。管理対象ノード内のサーバーにソリューションをデプロイできますが、その管理対象ノードにはサンプル・アプリケーション・ギャラリーは含まれません。管理対象ノードは、カスタム・プロファイルで定義されており、ファースト・ステップ・コンソールを備えています。

関連概念

15 ページの『第 3 章 セル内での複数のプラットフォームの使用』

綿密に計画することによって、分散オペレーティング・システム、i5/OS オペレーティング・システム、および z/OS オペレーティング・システムの各プラットフォームのノードが含まれたデプロイメント・マネージャー・セルを作成することができます。

37 ページの『プロファイル』

プロファイルでは、個別のコマンド・ファイル、構成ファイル、ログ・ファイルを持つ固有のランタイム環境を定義します。プロファイルでは、スタンドアロン・サーバー、デプロイメント・マネージャー、および管理対象ノードの 3 つのタイプの環境を定義します。

デプロイメント環境

デプロイメント環境とは、Service Component Architecture (SCA) の対話をホストするための環境を共同して提供する、構成済みのクラスター、サーバー、およびミドルウェアの集合のことです。例えば、デプロイメント環境には、メッセージの宛先用のホスト、ビジネス・イベントの処理プログラム、および管理プログラムが組み込まれている場合があります。

デプロイメント環境の計画では、容量、可用性、スケーラビリティ、およびフェイルオーバー・サポートに対するビジネス・ニーズを満たすことができるように、デプロイメント環境の物理的なレイアウト (トポロジー) を設計することが必要になります。設計の重要な部分としては、デプロイメント環境を構成するハードウェア上のサーバーの数と相対的な配置があります。

スタンドアロン環境

Service Component Architecture (SCA) モジュールをスタンドアロン・サーバーにデプロイすることができます。この環境はセットアップが最も容易ですが、スタンドアロン・サーバーは他のサーバーへ接続されないため、その容量は同一コンピューター・システム上のリソースに限られ、フェイルオーバー・サポートも組み込まれません。

スタンドアロン・サーバーで提供される容量、スケーラビリティ、可用性、またはフェイルオーバー・サポートを超えるものが必要な場合は、相互接続サーバーのデプロイメント環境を検討する必要があります。

相互接続サーバー

デプロイメント環境とは、以下のような WebSphere Process Server のアプリケーション・コンポーネントをサポートする相互接続サーバーの集合のことです。

- Business Process Choreographer
- ビジネス・ルール
- メディエーション
- リレーションシップ

この環境では、WebSphere Enterprise Service Bus と WebSphere Application Server ベースのサーバーもサポートされます。

デプロイメント環境内のサーバーは、1 つ以上のホスト・システム上で稼働することができます。複数のサーバーを、ロード・バランシングとフェイルオーバーをサポートするためにクラスター にグループ化することができます。

スタンドアロン・サーバーでは提供できないパフォーマンス、可用性、スケーラビリティ、分離機能、セキュリティ、および安定度の特性に加えて、相互接続サーバーまたはクラスターのデプロイメント環境には、集中化されたデプロイメント・マネージャー からすべてのサーバーまたはクラスターを管理できるというさらなる利点もあります。

デプロイメント環境パターン

提供されるデプロイメント環境パターンのいずれかを使用する場合、それに応じた要件と計画を把握していれば、デプロイメント環境の構成は簡単です。パターンには、以下の 3 つがあります。

- 単一クラスター
- リモート・メッセージング
- リモート・メッセージングおよびリモート・サポート

これらのパターンのいずれもが要件を満たさない場合は、ユーザー独自のカスタマイズしたデプロイメント環境を計画して作成することもできます。

デプロイメント環境の作成タイミングの決定

デプロイメント環境の計画のほかに、その作成タイミングを決定する必要もあります。以下のオプションのいずれか 1 つを選択できます。

1. インストール・ウィザードまたはサイレント・インストールを使用して、ソフトウェアのインストール時にデプロイメント環境を作成します。
2. ソフトウェアを使用するホスト・システム上にインストールします。次に、プロファイル管理ツールまたは `manageprofiles` コマンドを使用して、デプロイメント環境を作成します。
3. ソフトウェアを使用するホスト・システム上にインストールします。プロファイル管理ツールまたは `manageprofiles` コマンドを使用して、デプロイメント・マネ

ージャーとカスタム・プロファイルを作成します。次に、デプロイメント・マネージャーの管理コンソールを使用して、デプロイメント環境を作成します。

選択するオプションは、デプロイメント環境の複雑さによって決まります。提供されているデプロイメント環境パターンのいずれかが要件を満たす場合は、オプション 1 (44 ページ) または 2 (44 ページ) を選択し、提供されているいずれのパターンも要件を満たさない場合は、オプション 3 (44 ページ) を選択します。

デプロイメント環境の作成に使用する方法に関係なく、デプロイメント環境のいくつかの面は、管理コンソールを使用して引き続き管理することができます。(例えば、ノードをデプロイメント環境にさらに追加することができます。) ただし、オプション 1 (44 ページ) または 2 (44 ページ) を使用してデプロイメント環境を作成した場合は、一部変更できないものがあります。(例えば、データベース・タイプを変更することはできません。)

管理対象サーバー

管理対象サーバーとは、管理対象ノード内に構成されるサーバーのことです。管理対象サーバーにより、アプリケーションが実行されるデプロイメント環境内に特定のリソースが提供されます。

管理対象サーバーは、必要に応じてクラスターのメンバーになることが可能です。堅固で実動規模のプロセス・サーバーを提供するには、管理対象サーバーのクラスターを含むデプロイメント環境を構成します。

サーバーとクラスターの構成および管理には、デプロイメント・マネージャーの管理コンソールを使用します。

デプロイメント環境のクラスター

クラスターを使用すると、アプリケーションの能力と可用性が単一サーバーの場合よりも高まります。

クラスターとは、高可用性およびワークロード・バランシングをアプリケーションに提供する管理対象サーバーの集合のことです。クラスターのメンバーは、各種ホスト上にあるサーバーの場合と、同じホスト (同じノード) 上にあるサーバー場合があります。高可用性とワークロード・バランシングを最大限に達成するには、各クラスター・メンバーを異なるホスト・マシンに配置します。

クラスター環境には、以下の利点があります。

- **ワークロード・バランシング:** 複数のサーバー上でアプリケーション・イメージを実行することにより、クラスターはクラスター内のサーバー全体のアプリケーション・ワークロードのバランスを取ります。
- **アプリケーションの処理能力:** サーバーのハードウェアをアプリケーションをサポートするクラスター・メンバーとして構成することにより、アプリケーションの処理能力を増強できます。
- **アプリケーションの可用性:** サーバーに障害が発生した場合、アプリケーションの処理はクラスター内の他のサーバー上で続行されます。これにより、リカバリー作業をアプリケーション・ユーザーに影響を与えることなく進めることができます。

- 保守容易性: アプリケーションの処理を停止することなく、計画された保守のためにサーバーを停止できます。
- 柔軟性: デプロイメント・マネージャーの管理コンソールを使用することにより、必要に応じて容量を追加または除去することができます。

デプロイメント環境パターン

デプロイメント環境パターンは、デプロイメント環境に含まれるコンポーネントとリソースの制約と要件を指定します。デプロイメント環境パターンは、お客様がデプロイメント環境を最も簡単な方法で作成できるよう支援することを目的とし、大半のビジネス要件を満たせるように設計されています。

インストール・プロセスを簡単にするために、パターンをインプリメントするためのガイド付きインストール・ウィザードがあります。

3つのデプロイメント環境パターンのそれぞれが、特定の要件のまとまりに対応しています。ほとんどの要件のまとまりは、これらのパターンのいずれかを使用することで対処できます。

以下の記述は、インストールについて説明するためのものではありません。これらのパターンのいずれかに当てはまるデプロイメント環境を作成するには、インストールまたはプロファイルの作成時、あるいは管理コンソールで選択を行います。

単一クラスター・パターン

単一クラスター・パターンは、アプリケーションの実行および同期呼び出しに焦点を当てたシナリオを対象にしています。このパターンでは、メッセージング要件は最小に維持します。Service Component Architecture (SCA) の内部非同期呼び出し、Java Message Service (JMS) と MQ のメッセージングのバインディングは、同じクラスターで複数のメッセージング・エンジンをサポートしません。モジュールでこれらのいずれかが必要な場合は、メッセージング・インフラストラクチャーがアプリケーション・デプロイメントのターゲットとは別個のクラスターにある、ほかのパターンを選択してください。

すべてのコンポーネントは以下の単一クラスター上で実行されます。

- Service Component Architecture (SCA) アプリケーション・バス・メンバー
- SCA システム・バス・メンバー
- Business Process Choreographer バス・メンバー
- Explorer などの Business Process Choreographer の各コンポーネント
- Business Process Choreographer Container
- Common Event Interface (CEI) バス・メンバー
- CEI サーバー
- ビジネス・ルール・マネージャー
- アプリケーション・デプロイメント・ターゲット

アプリケーション・デプロイメント・ターゲットを構成して、SCA アプリケーションと Business Process Choreographer コンポーネントをサポートします。

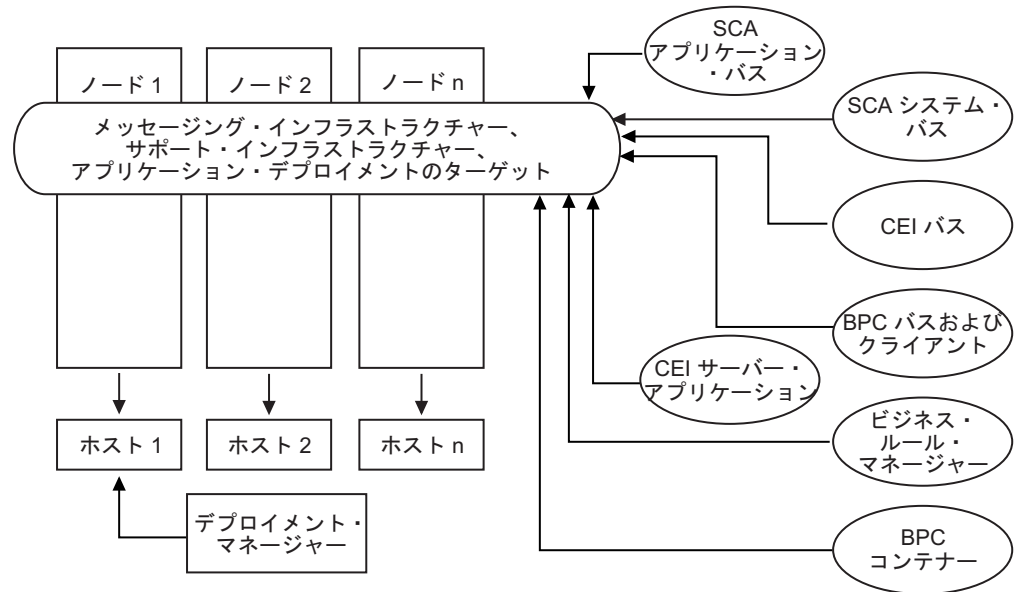


図1. 単一クラスター・パターン

リモート・メッセージング・パターン

リモート・メッセージング・パターンは、メッセージング・ロール用に別のクラスターを提供します。このパターンは、負荷に対してクラスターを拡張できるため、非同期呼び出しを必要とするシナリオに適しています。各コンポーネントは、2つのクラスター間で分割されます。

リモート・メッセージング・クラスター

- Service Component Architecture (SCA) アプリケーション・バス・メンバー
- SCA システム・バス・メンバー
- Business Process Choreographer (BPC) バス・メンバー
- Common Event Interface (CEI) バス・メンバー

サポート・インフラストラクチャーおよびアプリケーション・デプロイメントのターゲット・クラスター

- CEI サーバー・アプリケーション
- ビジネス・ルール・マネージャー
- Explorer などの Business Process Choreographer の各コンポーネント
- アプリケーション・デプロイメント・ターゲット

アプリケーション・デプロイメント・ターゲットを構成して、SCA アプリケーションと Business Process Choreographer コンポーネントをサポートします。

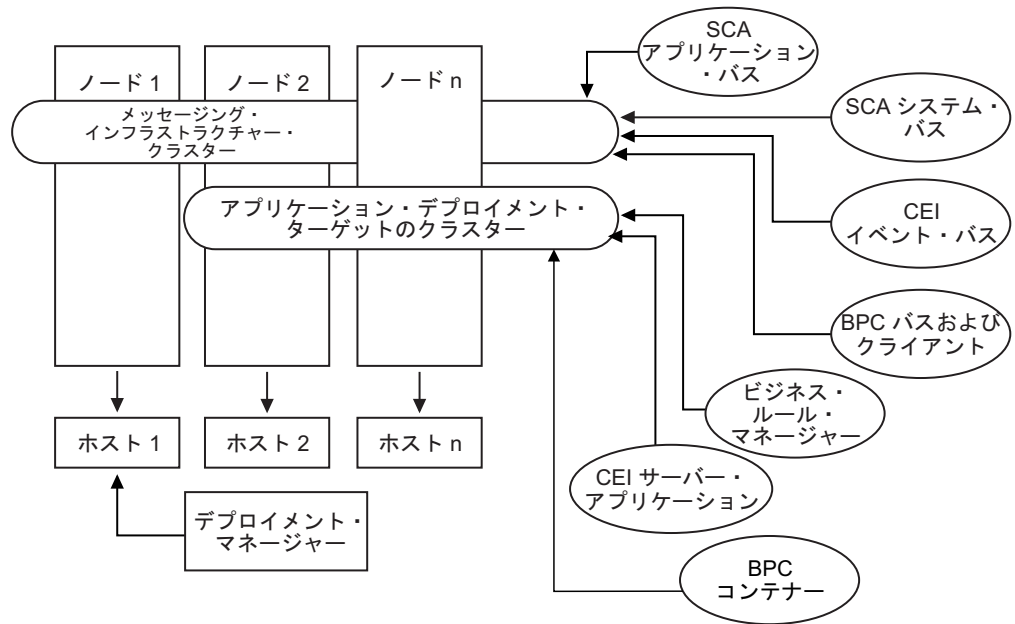


図2. リモート・メッセージング・パターン

リモート・メッセージングおよびリモート・サポート・パターン

この 3 クラスター・パターンでは、最も高い負荷を処理するクラスターにリソースが割り振られます。このパターンは最も柔軟で用途が広く、ほとんどのユーザーが好みます。各コンポーネントは、3 つのクラスター間で分割されます。

リモート・メッセージング・インフラストラクチャー・クラスター

- Service Component Architecture (SCA) アプリケーション・バス・メンバー
- SCA システム・バス・メンバー
- Business Process Choreographer (BPC) バス・メンバー
- Common Event Interface (CEI) バス・メンバー

リモート・サポート・インフラストラクチャー・クラスター

- CEI サーバー・アプリケーション
- ビジネス・ルール・マネージャー
- Explorer などの Business Process Choreographer の各コンポーネント

アプリケーション・デプロイメント・クラスター

- アプリケーション・デプロイメント・ターゲット
- Business Process Choreographer Container

アプリケーション・デプロイメント・ターゲットを構成して、SCA アプリケーションと Business Process Choreographer コンポーネントをサポートします。

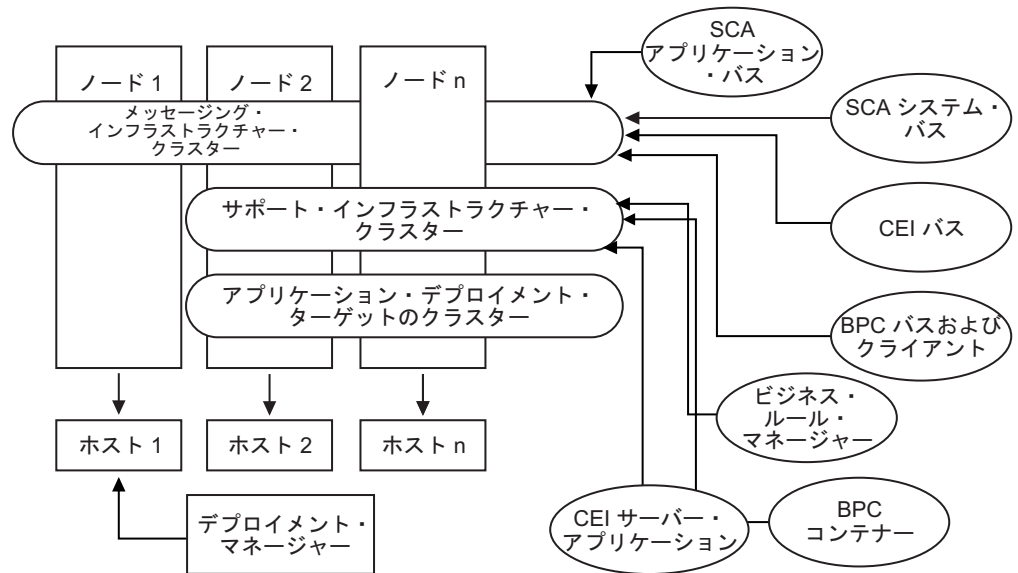


図3. リモート・メッセージングおよびサポート・パターン

リソース割り振りの例

次の図は、リソースがリモート・メッセージングおよびリモート・サポート・パターンを使用して割り振られる場合がある 1 つの例を示しています。このインストール済み環境に対する最も重い負荷はアプリケーションでの使用であるため、アプリケーション・デプロイメントのターゲット・クラスター (クラスター 3) にほかの機能よりも多くのリソース (サーバー 1、サーバー 2、およびサーバー 6) が割り振られています。

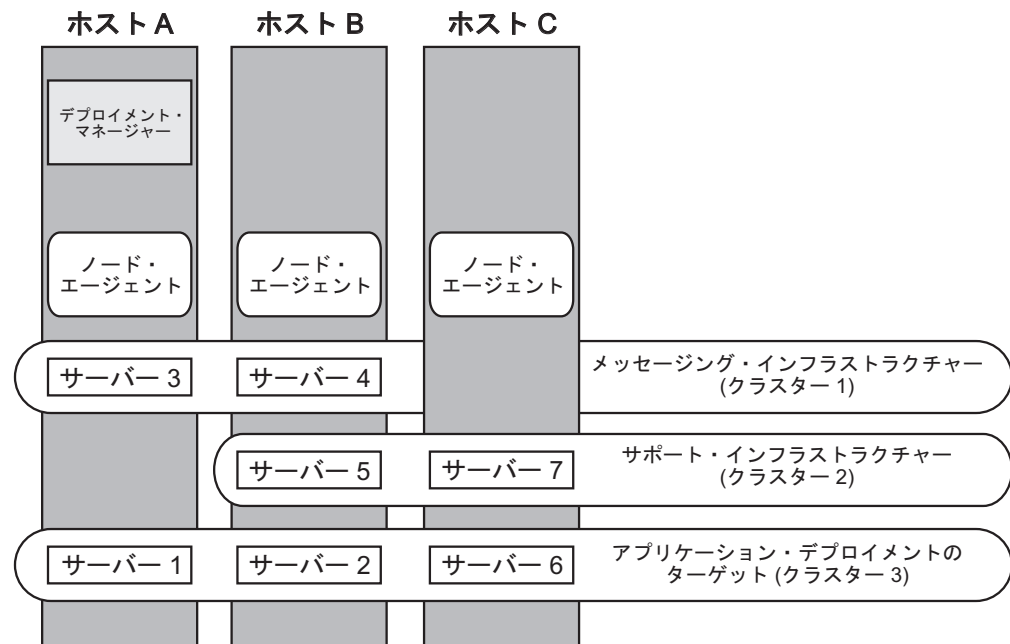


図4. リソース割り振りの例

関連概念

エラー防止とリカバリーの概要

エラー防止とリカバリーの情報では、システム障害を引き起こす問題を回避する方法について説明し、通常の状態と異常な状況の両方で発生する可能性があるシステム障害からリカバリーする方法について、情報を提供しています。

デプロイメント環境機能

堅固なデプロイメント環境を設計するためには、各クラスターが特定の IBM 提供のデプロイメント環境パターンまたはカスタム・デプロイメント環境に提供できる機能について理解しておく必要があります。この知識は、ニーズに最も合うデプロイメント環境パターンを正しく判別するのに役立ちます。

ネットワーク・デプロイメントの場合、クラスターは環境に特定の機能を提供するために共同で作業することができます。お客様の要件に合わせて、デプロイメント環境内の各クラスターに特定の機能を割り当てて、パフォーマンス、フェイルオーバー、および容量を提供します。

IBM 提供のデプロイメント環境パターン

デプロイメント環境パターンで構成されたクラスターでは、以下の機能が提供されます。

アプリケーション・デプロイメント・ターゲット

アプリケーションをインストールする先の 1 つのクラスターから構成されます。選択したデプロイメント環境パターンによっては、アプリケーション・デプロイメント・ターゲットによって、メッセージング・インフラストラクチャーおよびサポート・インフラストラクチャーの機能が提供される場合があります。デプロイするアプリケーションのタイプに基づいて適切な製品を選択します。

- アプリケーションにヒューマン・タスクまたはビジネス・プロセスの成果物が含まれる場合、WebSphere Process Server をインストールします。
- アプリケーションにメディエーション・モジュールのみが含まれる場合、WebSphere Enterprise Service Bus をインストールします。

単一のクラスター・パターンにおいて、アプリケーション・デプロイメントのターゲットはデプロイメント環境の機能全体を提供します。

サポート・インフラストラクチャー

Common Event Infrastructure (CEI) サーバーをホスティングする 1 つのクラスター、およびご使用のシステムの管理に使用されるその他の各種インフラストラクチャー・サービスから構成されます。各種インフラストラクチャー・サービスには、以下のものがあります。

- ビジネス・ルール
- セレクター
- ヒューマン・タスク
- ビジネス・プロセス

重要: このノードに対して、アプリケーション・デプロイメント・ターゲット・クラスターに対して使用した製品機能と同じ製品機能のカスタム・プロファイルを使用する必要があります。

メッセージング・インフラストラクチャー

メッセージング・エンジンが配置されている 1 つのクラスターから構成されます。メッセージング・エンジンによって、デプロイメント環境内のノード間の通信が可能になります。クラスターがメッセージング機能のみを提供する場合は、このクラスターは WebSphere Process Server ではなく WebSphere Application Server で作成されたノード上のメンバーでも構成することができます。

カスタム・デプロイメント環境

カスタム・デプロイメント環境では、更に多様なトポロジーが可能になります。アプリケーションでさらに処理能力を必要とするか、さらに多数のクラスターに対してサポート・インフラストラクチャー機能を拡大する必要があるか、いくつかのサーバーまたはクラスター用のサポート・インフラストラクチャーを 1 つのクラスターに統合する必要がある場合は、カスタム・デプロイメント環境を使用してこれらを実現することができます。

クラスター間で機能を分けるには、コラボレーション単位を使用します。コラボレーション単位により、お客様のニーズに基づいて、各機能を 1 つの単位として連携する複数のクラスターとサーバーに分散して、独立性、機能統合、スループット能力、およびフェイルオーバー機能をさらに増大させることができます。

管理コンソールでは、コラボレーション単位を以下のようにグループ化します。

メッセージング

メッセージング単位では、IBM 提供のデプロイメント環境パターン用のメッセージング・インフラストラクチャーと同じサポートが提供されます。ローカルのメッセージング・エンジンが含まれたサーバーがクラスター内にあり、この単位内のその他のサーバーとクラスターは、そのメッセージング・エンジンをメッセージの宛先として使用します。

Common Event Infrastructure

Common Event Infrastructure 単位は、CEI サーバーをホスティングするサーバー、および CEI 機能をサポートするその他のクラスターとサーバーから構成されます。単位内の各クラスターまたはサーバーで受信された Common Base Event は、CEI サーバーをホスティングするサーバーに送信されます。ご使用のデプロイメント環境で、さまざまなイベント・ソースからのイベントを分離するためにさらに多数の CEI サーバーをホスティングするのに必要な分だけのコラボレーション単位を使用します。

アプリケーション・サポート

アプリケーション・サポート単位は、IBM 提供のデプロイメント環境パターン用のサポート・インフラストラクチャーと似ています。これらの単位は、アプリケーションをデプロイするクラスターとサーバーをグループ化します。これらの単位は、コラボレーション単位をさらに定義して、複数のビジネス・コンテナまたは Service Component Architecture (SCA) サポート・クラスターをデプロイメント環境内に定義できるという点が異なります。1 つの単位により、その単位内の同一または異なるクラスターに 1 つのビジネス・プロセス・クラスター、および 1 つ以上の SCA サポート・クラスターとサポート・アプリケーションが定義されます。

デプロイメント環境パターンの選択

IBM 提供のいずれかのパターンを選択するか、独自のカスタム・デプロイメント環境を作成することによって、デプロイメント環境を構成できます。このトピックでは、IBM 提供の各パターンでサポートされる機能をリストします。

始める前に

以下のための情報に精通している必要があります。

- ビジネス要件の評価
- 使用可能なリソースの明確化

このタスクについて

デプロイメント環境の設計を完了し、さまざまな製品ウィザードでサポートされる IBM 提供のパターンがニーズを満たすかどうかを判断する必要があります。

重要: デプロイメント環境で z/OS システムまたはクラスターを使用する場合は、サーバーまたはクラスターが提供する機能を判断してください。同じクラスターで z/OS システムと他のシステムを混在させることはできないため、設計時にはこの点を考慮に入れる必要があります。

手順

1. ビジネスのニーズに最適な IBM 提供のパターンを判断します

デプロイメント環境パターン	機能
単一クラスター	メッセージング、アプリケーション・デプロイメント・ターゲット、およびアプリケーション・サポート機能を単一のクラスターに含めます。このパターンは、同期メッセージング環境、PoC (概念検証) 環境、またはアプリケーション・テスト環境に役立ちます。
リモート・メッセージング	このパターンでは、メッセージング環境をアプリケーション・デプロイメント・ターゲットおよびアプリケーション・サポート機能から分離します。このパターンは、日常の運用でメッセージのスループットが重要な要件となる場合に使用します。このパターンは、非同期メッセージング・システムおよびトランザクション・システムの場合に強くお勧めします。

デプロイメント環境パターン	機能
リモート・メッセージングおよびリモート・サポート	このパターンでは、メッセージング、Common Event Infrastructure (CEI)、アプリケーション・デプロイメント・ターゲット、およびアプリケーション・サポート機能を別個のクラスターに分離します。ほとんどの業務では、このパターンを使用してデプロイメント環境をサポートできます。理由は、このパターンはパフォーマンスを重視し、トランザクション処理をメッセージングなどのサポート機能から分離して設計されているためです。

2. オプション: メディエーション・サービスのみを提供する必要がある場合は、WebSphere Process Server の代わりに Enterprise Service Bus をインストールします。
3. IBM 提供のいずれのパターンもビジネスのニーズを満たさない場合は、カスタム・デプロイメント環境を実装できます。

注: カスタム・パターンを実装するには、デプロイメント環境の動作に関する詳細な知識を持ち、サーバーおよびクラスターの正しい構成方法を理解している必要があります。

次のタスク

製品をインストールおよび構成します。

関連概念

46 ページの『デプロイメント環境パターン』

デプロイメント環境パターンは、デプロイメント環境に含まれるコンポーネントとリソースの制約と要件を指定します。デプロイメント環境パターンは、お客様がデプロイメント環境を最も簡単な方法で作成できるよう支援することを目的とし、大半のビジネス要件を満たせるように設計されています。

74 ページの『エラー処理方針とソリューション・リカバリー』

WebSphere Process Serverには、リカバリーのために利用できるエラー処理機能とツールが組み込まれています。

関連資料

105 ページの『デプロイメント環境の再始動』

リカバリー・プロセスの 1 ステップとして、デプロイメント環境の再始動が必要となる場合があります。

第 5 章 デプロイメント環境の実装

デプロイメント環境を設計したら、設計を実現するための作業を行います。デプロイメント環境の実装に使用する方法にかかわらず、同じ汎用の手順を行います。

始める前に

- トポロジを計画し、以下に関する決定事項を記録します。

- 必要なサーバーおよびクラスター。
- 必要なデータベースの数。

注: **i5/OS** i5/OS システムにインストールする場合、システムに置くことができる DB2 Universal Database は 1 つだけです。i5/OS システムには、複数のデータベースの代わりにデータベース・コレクションと一意に呼ばれるコンポーネント・テーブルが常駐します。

- どのデータベース表がどのデータベースに属するか
- 必要なユーザー ID および認証ロール
- デプロイメント環境に関係する各クラスターが提供する機能
- デプロイメント環境の実装に使用する方法
- 製品のインストール先システムがハードウェア要件およびソフトウェア要件を満たすことを確認します。
- インストールできるようにオペレーティング・システムを準備します。
- **Windows** **Linux** **UNIX** 製品資料に従って、データベース製品をインストールして構成します。必ず以下を行ってください。
 - 製品をサーバーとして構成します。
 - データベース内のデータおよびテーブルへのアクセスに使用する WebSphere Process Server のユーザー ID を定義します。
 - **オプション:** WebSphere Process Server 共通データベース (デフォルトの名前は WPRCSDB) を作成します。

このデータベースを製品のインストール中に作成した場合、またはプロファイル管理ツールによって作成した場合は、このステップはスキップしてください。

- 構成に必要なその他すべてのデータベースを作成します。特定の機能向けにデータベースを作成しない場合、システムは WebSphere Process Server 共通データベースを使用します。
- **i5/OS** **オプション:** WebSphere Process Server 共通データベース・スキーマを作成します。スキーマ名は、システム内で固有の名前にする必要があります。
- すべてのサーバーでシステム・クロックを同期させます。同じタイム・ゾーンに調整する場合は、クロックの差が相互に 5 分以内でなければなりません。
- トポロジに関係するすべてのサーバーが、IP アドレスおよびドメイン・ネーム・サーバー (DNS) 名で位置指定できることを確認します。

- すべてのシステムでディレクトリーおよびファイルを作成できる適切な権限を持つユーザー ID を用意します。
- 他の製品と共存して所定の冗長度を確保するために必要な準備を行います。

このタスクについて

以上でデプロイメント環境の計画が完了し、前提条件タスクをすべて実行しました。この後は、設計で必要になるサーバーおよびクラスターをインストールして構成します。どのような方法でデプロイメント環境を実装するかにかかわらず、以下のステップに従って設計の単一のセルを作成します。

注: ここに示す手順には、デプロイメント環境の実装に必要なすべてのステップを記載しています。インストール方法によっては、手順が多少前後する場合があります。

手順

1. デプロイメント環境に関連するすべてのシステムに製品バイナリーをインストールし、ソフトウェアが正常にインストールされたことを確認します。
2. デプロイメント・マネージャーを作成します。
3. デプロイメント・マネージャーを始動します。
4. 必要な数の管理対象ノードを作成します。
5. ステップ 4 のノードを、ステップ 2 で作成したデプロイメント・マネージャーに統合します。
6. セルを構成します。

重要: デプロイメント環境によっては、構成の処理に時間がかかる場合があります。プロセスがタイムアウトになるのを防ぐため、デプロイメント・マネージャーの SOAP 要求のタイムアウト値を大きい値 (例えば 1800 秒) に設定します。WebSphere Application Server インフォメーション・センターの『Java Management Extensions コネクター・プロパティ』を参照してください。

このために、設計で定義した機能を実行するクラスターを作成した後、そのクラスターにメンバーを追加する必要があります。

パターンに基づくデプロイメント環境を実装する設計の場合は、必要なすべてのクラスターの作成とクラスター・メンバーの定義が自動的に行われ、必要な機能がすべて提供されます。この機能には、選択したデプロイメント環境パターンに応じ、アプリケーション・デプロイメント、メッセージング・サポート、およびインフラストラクチャー・サポートのための各クラスターが含まれます。

カスタムのデプロイメント環境を実装する設計の場合は、必要な機能を提供するためのクラスターをすべて独自に作成する必要があります。この機能には、アプリケーション・デプロイメントのメッセージング・サポート、アプリケーション・サポート、および Common Event Infrastructure サポートがあります。

7. テーブルの作成延期を選択した場合は、トポロジーで必要になるデータベースまたはデータベース表を構成します。

構成作業は、延期オプションの選択によって生成された各種スクリプトを実行することで進めます。

- a. 共通のデータベース表を構成します。このテーブルは共通データベース内のものです。
- b. メッセージング・エンジンのデータベース表を構成します。このテーブルは共通データベース内のものです。
- c. オプション: Business Process Choreographer のデータベース表を構成します。

システムでビジネス・プロセスもヒューマン・タスクも使用しない場合は、このステップを省略してください。このテーブルは、Business Process Choreographer で使用するように構成したデータベース (デフォルトの名前は BPEDB) に配置するものです。

Business Process Choreographer Explorer レポート機能を使用する場合は、Business Process Choreographer Explorer レポート・データベース (OBSRVDB) も構成する必要があります。

- d. エンタープライズ・サービス・バスのロギング・メディエーション・データベース表を作成します。このテーブルは共通データベース内のものです。
 - e. Common Event Infrastructure データベースを構成します。
8. ルーティング・サーバーをインストールおよび構成します。これは、IBM HTTP Server などの任意のサーバーにすることができます。このサーバーによって、クライアントはこのトポロジー内のアプリケーションにアクセスできるようになります。
 9. テスト・アプリケーションをインストールして実行し、インストールを検証します。

次のタスク

- 必要に応じて別のセルを作成します。
- このデプロイメント環境で実行するアプリケーションをデプロイします。

関連概念

50 ページの『デプロイメント環境機能』

堅固なデプロイメント環境を設計するためには、各クラスターが特定の IBM 提供のデプロイメント環境パターンまたはカスタム・デプロイメント環境に提供できる機能について理解しておく必要があります。この知識は、ニーズに最も合うデプロイメント環境パターンを正しく判別するのに役立ちます。

46 ページの『デプロイメント環境パターン』

デプロイメント環境パターンは、デプロイメント環境に含まれるコンポーネントとリソースの制約と要件を指定します。デプロイメント環境パターンは、お客様がデプロイメント環境を最も簡単な方法で作成できるよう支援することを目的とし、大半のビジネス要件を満たせるように設計されています。

43 ページの『デプロイメント環境』

デプロイメント環境とは、Service Component Architecture (SCA) の対話をホストするための環境を共同して提供する、構成済みのクラスター、サーバー、およびミドルウェアの集合のことです。例えば、デプロイメント環境には、メッセージの宛先用のホスト、ビジネス・イベントの処理プログラム、および管理プログラムが組み込まれている場合があります。

45 ページの『デプロイメント環境のクラスター』

クラスターを使用すると、アプリケーションの能力と可用性が単一サーバーの場合よりも高まります。

39 ページの『サーバー』

サーバーによって、WebSphere Process Server のコア機能が提供されます。プロセス・サーバーでは、アプリケーション・サーバーの機能が拡大または拡張されて、Service Component Architecture (SCA) モジュールが処理されます。他のサーバー (デプロイメント・マネージャーおよびノード・エージェント) は、プロセス・サーバーの管理に使用されます。

42 ページの『デプロイメント・マネージャー』

デプロイメント・マネージャーとは、他のサーバーの論理グループ (セル) の操作を管理するサーバーです。デプロイメント・マネージャーは、サーバーおよびクラスターを管理するための中央の場所になっています。

 メッセージング宛先ホストまたはキュー宛先ホスト

メッセージング宛先ホストまたはキュー宛先ホストにより、サーバー内にメッセージング機能が提供されます。サーバーをメッセージング・ターゲットとして構成すると、サーバーはメッセージング宛先ホストになります。

関連タスク

 製品のインストールの検査

WebSphere Process Server が正常にインストールされ、スタンドアロン・サーバー・プロファイルまたはデプロイメント・マネージャー・プロファイルが正常に作成されていることを確認するには、インストール検査ツールを使用します。プロファイルは、デプロイメント・マネージャーまたはサーバー用のランタイム環境を定義するファイルから構成されます。installver_wbi 検査合計ツールを使用してコア製品ファイルを検査します。インストール検査テスト (IVT) ツールを使用して、各プロファイルを検査します。

 デフォルト値を使用したプロファイルの構成


プロファイル管理ツールを使用して、デフォルトの構成設定でプロファイルを作成または拡張する方法について説明します。

 カスタマイズした値を使用したプロファイルの構成


プロファイル管理ツールを使用して、カスタマイズした構成設定でプロファイルを作成または拡張する方法について説明します。


 デプロイメント環境に対するプロファイルの構成


新規または既存のデプロイメント環境パターンで使用する、カスタマイズした構成設定のプロファイルを作成または拡張する方法について説明します。プロファイル管理ツールを使用して、プロファイルを構成します。


 デプロイメント・マネージャーの停止と再始動

デプロイメント・マネージャーに対して任意の構成変更を行った場合は、デプロイメント・マネージャーを停止後に再始動して、それらの変更を有効にする必要があります。


 **カスタム・ノードのデプロイメント・マネージャーへの統合**
addNode コマンドを使用して、カスタム・ノードをデプロイメント・マネージャー・セルに統合できます。以下の説明に従って、カスタム・ノードの統合およびデプロイのプロセスを実行します。

 **プロファイルの作成**
新規の WebSphere Enterprise Service Bus または WebSphere Process Server プロファイルを作成する方法について説明します。プロファイルの作成は、manageprofiles コマンドを使用してコマンド行から行うことも、プロファイル管理ツールのグラフィカル・ユーザー・インターフェース (GUI) を使用して対話式に行うこともできます。


 **既存のプロファイルの拡張**
既存の WebSphere Application Server、WebSphere Application Server Network Deployment、または WebSphere Application Server Network Deployment with Web Services Feature Pack プロファイル WebSphere Enterprise Service Bus や WebSphere Process Server プロファイルに拡張できます。あるいは、WebSphere Enterprise Service Bus プロファイル WebSphere Process Server プロファイルに拡張することもできます。このトピックの説明を使用してプロファイルを拡張します。プロファイルの拡張は、manageprofiles コマンドを使用してコマンド行から行うことも、プロファイル管理ツールのグラフィカル・ユーザー・インターフェース (GUI) を使用して対話式に行うこともできます。

 **デプロイメント環境の検証**
実動アプリケーションを新しい環境に移動する前に、テストを行ってすべてのコンポーネントが正常に動作することを確認してください。

関連情報


 **ソフトウェアのインストール**
WebSphere Process Server 製品の入手方法は 2 つあります。1 つは製品パッケージ内のディスクから入手する方法で、もう 1 つは Passport Advantage サイトからインストール・イメージをダウンロードする方法です (この場合は、そのためのライセンスが必要です)。ソフトウェアをインストールするには、グラフィカル・インターフェース・モードまたはサイレント・モードでインストール・ウィザードを使用します。サイレント・モードでは、インストール・ウィザードはグラフィカル・インターフェースを表示せずに、応答ファイルから応答を読み取ります。




 **Business Process Choreographer の構成**

 **Web サーバーとの通信**

 **IBM HTTP Server のインストール**

 **wsadmin ツール**

 **ノード・エージェントの管理**

-  クラスターの開始
-  クラスターの停止
-  Java Management Extensions コネクター・プロパティー

第 6 章 エラー防止およびリカバリーの計画

システムおよびアプリケーションのエラーの影響を最小化するために、エラー防止およびリカバリーの方法を作成することができます。

『エラー防止およびリカバリーの計画』の各トピックには、インフォメーション・センター・トピック、技術記事および IBM Redbooks® などのさまざまなリソースへのリンクが含まれています。これらには WebSphere のシステム・リカバリー機能を活用するように設計された開発プロセスおよびシステム構成パターンに関する詳細な情報が提供されています。

エラー防止とリカバリーの概要

エラー防止とリカバリーの情報では、システム障害を引き起こす問題を回避する方法について説明し、通常の状態と異常な状態の両方で発生する可能性があるシステム障害からリカバリーする方法について、情報を提供しています。

WebSphere Process Server は、ビジネス・プロセス管理 (BPM) ソリューションとサービス指向アーキテクチャー (SOA) ソリューションの実行および管理を可能にするために最適化されたミドルウェア・サーバーです。WebSphere Process Server は、WebSphere Application Server の基盤となる機能を土台として作成されています。

ミドルウェア・システムはさまざまな条件下で実行されますが、従来、それらのすべてが『良好なパス (good path)』条件になっているわけではありません。WebSphere Process Server の主要な機能の多くは、普通に見える動作から発生する可能性がある、不確実さに対処するためのものです。

想定および予期

『エラー防止およびリカバリーの計画』セクションの記載に従ってシステム障害およびリカバリーの情報を使用する前に、以下の想定リストを読んでください。

- WebSphere Process Server およびこの製品の基礎となっている基本的なアーキテクチャー上の原則、およびこの製品で実行される基本的な種類のアプリケーションに精通している。
- 統合プロジェクトの計画および実装方法を含め、統合プロジェクトについての基礎的な理解を得ている。
- 特に指定されない限り、システム障害およびリカバリーに関する情報は、バージョン 6.1.0 以降の WebSphere Process Server に関連しています。

注: 『エラー防止およびリカバリーの計画』セクションに含まれている推奨事項は、リモート・メッセージングおよびリモート・サポートのパターンを想定しており、WebSphere Process Server 用に 1 つ、メッセージング・エンジンおよび CEI イベント・サーバー用にそれぞれ 1 つずつの 3 つの別個のクラスターで構成されています。

関連概念

46 ページの『デプロイメント環境パターン』

デプロイメント環境パターンは、デプロイメント環境に含まれるコンポーネントとリソースの制約と要件を指定します。デプロイメント環境パターンは、お客様がデプロイメント環境を最も簡単な方法で作成できるよう支援することを目的とし、大半のビジネス要件を満たせるように設計されています。

関連資料

82 ページの『ピア・リカバリー』

ピア・リカバリーは、同じクラスター内の別のメンバーによって実行されるリカバリーであり、手動でも自動でも開始できます。ピア・リカバリー処理 (自動ピア・リカバリーまたは手動ピア・リカバリー) は、WebSphere の高可用性環境と緊密に連携しています。

エラー防止の計画

あらゆる IT 処理の場合と同様、極端なシチュエーションに対して計画を行って実行すれば、正常にリカバリーする確率は高くなります。

システムおよびアプリケーションのリカバリーを準備することに関しては、必須の考慮事項が数多くあります。これらの考慮事項は、次の 2 つのカテゴリーに分類できます。

- アプリケーション設計の一部としてのエラー防止手段
- 開発プロセスの一部としてのエラー防止手段

アプリケーション設計の一部としてのエラー防止

アプリケーション設計の一部としてのエラー防止手段を組み込むことは、特定の設計技法を実装し、製品の機能を活用して、システムおよびアプリケーションのエラーを防止することを意味します。

アプリケーションを正しく構築するためには、アーキテクチャーおよび設計のガイドラインと適切な標準と、その組み合わせとなるレビューおよびチェックポイントが完備された強力なシステム・ガバナンスが不可欠です。

アプリケーション設計の一部としてのエラー防止手段には、以下の要素が含まれています。

- 例外および障害に対する設計上の考慮事項の実装
- 既存の WebSphere Process Server エラー処理機能とツールを利用するエラー処理方針の実装
- 接続グループの作成とモジュール・アプリケーション設計技法の活用

接続グループ

接続グループは、SCA モジュール内に見られる特定の動作パターンを表します。

システムで考えられる要求ソースを表す接続グループを作成することがベスト・プラクティスです。

接続グループでは以下を行います。

- インバウンド・データを取得するためのすべてのロジックを 1 つのモジュールに配置します。

これはアウトバウンド・データが、外部システムまたはレガシー・システムに送信される場合にも該当します。

- データを接続および変換するためのすべてのロジックを 1 つのモジュールに配置します。

他のすべてのモジュールもインターフェースの標準セットを使用できるようになったため、さらなる変換を考慮する必要はありません。

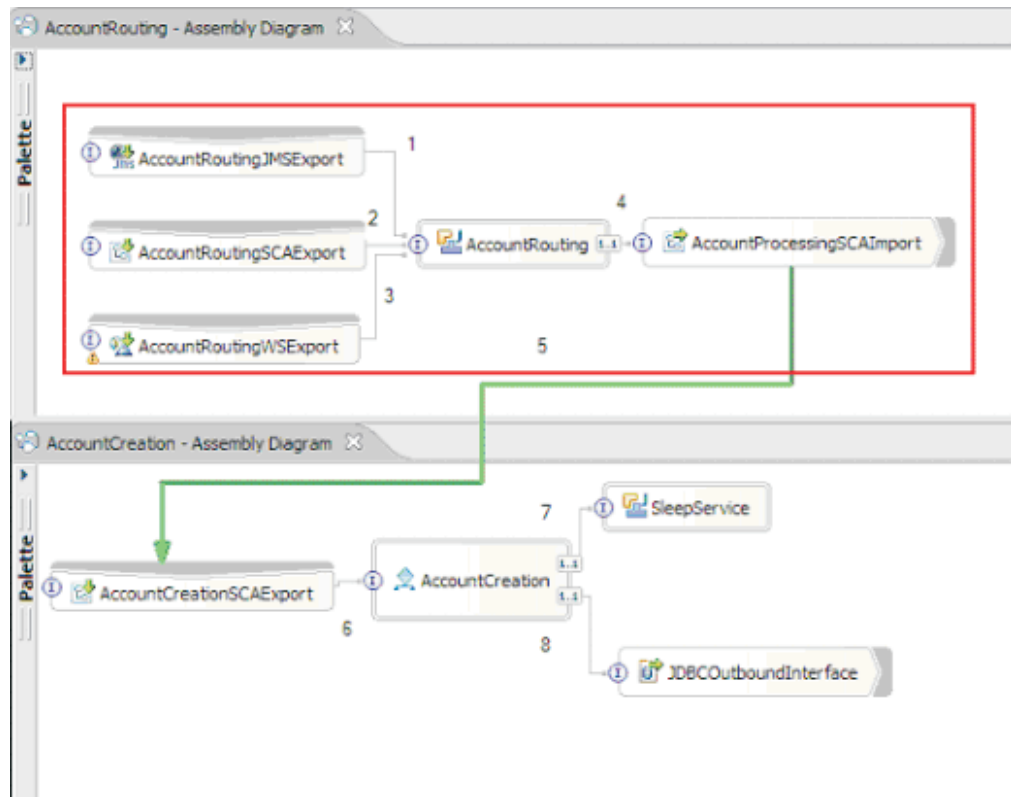
接続グループには、長期実行ビジネス・プロセスやビジネス・ステート・マシンのようなステートフル・コンポーネント・タイプが含まれません。これらの接続グループにより、特定のエンドポイントの統合要件をカプセル化して分離できます。一般に、WebSphere ESB メディエーション・モジュールがこの目的のために使用されるのは、「インフラストラクチャー」関連タスクを実装する便利な方法だからです。

接続グループの概念に従い、リカバリーの必要がある場合に、システムを休止状態にするための便利な方法として使用することもできます。接続グループ・モジュールはステートレスであるため、モジュールを一時的に停止して、システムが保持するイベントの処理を完了させる間、新規イベントのインバウンド・フローを遮断することができます。

注: インバウンド・イベントのフローを停止する場合、接続モジュールはインバウンドおよびアウトバウンドを同じモジュール内でサポートしてはいけません (同じ EIS システムがインバウンドとアウトバウンドの両方を持っている場合でも該当します)。インバウンドおよびアウトバウンドが同じモジュールでサポートされている場合、アウトバウンドはインバウンドと一緒にオフになります。このため、内部処理の実行が停止することがあります。このケースでは、インバウンドとアウトバウンドを分離することを検討してください。

システムがリカバリーされ、新規の作業を処理できるようになったら、これらのモジュールを再開できます。

以下の画面取りに外観が示されているモジュールは、接続グループの一部と見なされます。



接続グループは、SAP または CICS® などの外部ソースまたはレガシー・システムからの入力用に使用できます。または、Web ブラウザー・ベースのクライアントからの新規作業のためにも使用できます。

関連概念

ユース・ケース: 失敗イベントからのデータのリカバリー (recovering data from failed events)

ユース・ケースは、リカバリー・シナリオでのコンテキストとして使用されます。このユース・ケースでのビジネスには、新規アカウントを作成する要求を受け取るアプリケーションがあります。

関連資料

84 ページの『エクスポート・バインディング』

システムを完全に停止するには、使用可能なエクスポート・バインディングによってサポートされるさまざまなタイプの要求呼び出しを考慮する必要があります。

例外および障害に対するアプリケーション設計上の考慮事項

アプリケーション設計では、WebSphere Process Server のエラー処理機能と障害処理機能の利点を活用することを考慮する必要があります。

包括的なエラー処理方法を作成するには、ソリューション設計者は、WebSphere Process Server および WebSphere ESB が宣言済みまたは宣言済みでない例外をどのように表すかを理解する必要があります。

SCA プログラミング・モデルには以下の 2 つのタイプの例外が提供されています。

- サービス・ビジネス例外

サービス・ビジネス例外は、ビジネス・メソッドの機能シグニチャー (WSDL 障害または Java スロー) で宣言されるチェック例外です。サービス・ビジネス例外では、アプリケーションまたはサービスによって予期されるエラー状態を識別します。これらの例外は、「チェック例外」と呼ばれることもあります。

例えば、株価サービスでの `InvalidSymbolException` があります。そのような例外は `ServiceBusinessException` によってラップされ、クライアントに戻されます。

- サービス・ランタイム例外

「システム例外」としても知られています。サービス・ランタイム例外は、メソッド・シグニチャーでは宣言されません。一般にこの例外は、Java コンポーネントでの `NullPointerException` などのように、アプリケーションによって予期されていないエラー状態を表します。

これらの例外は、`ServiceRuntimeException` によってラップされ、クライアントに戻されます。クライアントは `ServiceRuntimeException` について問い合わせる原因を判別できます。

注: SCA レベルで動作するとき、これらの例外は障害と呼ばれることもあります。ただし、Java コードを使用するときは、これらはたいてい例外と呼ばれません。

サービス・ビジネス例外の処理:

サービス・ビジネス例外は、アプリケーションまたはサービスによって予期される既知および宣言済みの例外を表します。

サービス・ビジネス例外は、サービス・インターフェースで定義されます。

コンポーネント開発者は、スローされる可能性がある例外を宣言することに注意を払い、消費側のサービスでそれらの例外を処理できるようにしてください。例えば、銀行用アプリケーションのビジネス障害には、「無効な口座番号」または「資金不足」がビジネス例外として含まれる場合があります。したがって、サービス呼び出すアプリケーションには、無効な口座番号が渡された場合や、\$100 を振り替えようとして口座に \$50 しかなかった場合などの状況を処理するロジックを組み込んでおく必要があります。これらは、呼び出し側のアプリケーションが処理するように設計されたタイプのビジネス・エラーです。WebSphere Process Server のビジネス例外は、`catch` して適切に処理するクライアントに戻されます。

ビジネス・サービス例外を処理する場合、サービス・コンシューマーは、宣言済みのビジネス例外の場合に以下のアクションのいずれかを実行するようにクライアントを実装します。

1. 例外を `catch` して、呼び出し側のアプリケーションに適したサービス・ビジネス例外を作成します。

これは、元の例外を新規例外に含める (ラッピングする) ことを意味する場合があります。これが最も頻繁に行われるのは、呼び出し側のモジュールが、呼び出

し先のサービスと同じビジネス例外を持たない場合です。例外を catch して、呼び出し側のアプリケーションのためにサービス・ビジネス例外を作成するフローの例を以下に示します。

- a. Module A は SBE 「MoneyTransferFailed」を持つ
 - b. Module B は SBE 「InsufficientFunds」を持つ
 - c. Module A は Module B を呼び出し、「InsufficientFunds」例外を取得する
 - d. Module A は、資金の不足という元のエラーを定義するストリングを格納できる場所を持つ、新規例外 「MoneyTransferFailed」を作成する必要がある。
2. 例外を catch して代替ロジックを実行します。

関連概念

ユース・ケース: 失敗イベントからのデータのリカバリー (recovering data from failed events)

ユース・ケースは、リカバリー・シナリオでのコンテキストとして使用されます。このユース・ケースでのビジネスには、新規アカウントを作成する要求を受け取るアプリケーションがあります。

サービス・ランタイム例外の処理:

サービス・ランタイム例外は、未宣言の例外です。一般にこの例外は、アプリケーションによって予期されていないエラー状態を表します。

サービス・ランタイム例外を使用して、実行時の予期せぬ状態を知らせます。

コンポーネント開発者は、次の方法でサービス・ランタイム例外を処理できます。

1. それらの例外を catch し、何らかの代替ロジックを実行します。

例えば、あるパートナーが要求を処理できなくても、別のパートナーでは処理できることもあります。

2. 例外を catch し、クライアントに「再スロー」します。
3. その例外をビジネス例外に再マップします。

例えば、パートナーがタイムアウトになると、ビジネス例外が生成される可能性があります。その例外では、ほとんどの要求が処理されたものの、完了していない 1 つの要求が残されており、その要求を後で再試行するか、異なるパラメータを指定して試行すべきことが示されます。

例外が catch されない場合、例外は現行コンポーネントを呼び出したコンポーネントに渡されます。この呼び出しチェーンは、チェーンの最初の呼び出し元に戻るまで続きます。例えば、Module A が Module B を呼び出し、Module B が Module C を呼び出して、Module C が例外をスローすると、Module B は例外を catch する場合も、またはしない場合もあります。Module B が例外を catch しない場合、例外は Module A まで戻されます。

注: ランタイム例外はインターフェースの一部として宣言されていないため、コンポーネント開発者は、例外の解決を試みて、クライアントがユーザー・インターフェースである場合にランタイム例外がクライアントまで不意に伝搬されるのを防止してください。

一般に、サービス・ランタイム例外が発生すると、サービスのトランザクションのロールバックが生じます。クライアントとサービス・プロバイダーの間で非同期呼び出しパターンが使用された場合は、障害を表すために失敗したイベントが作成されます。

以下に、`ServiceRuntimeException` の現行サブクラス 4 つを示します。

1. `ServiceExpirationRuntimeException`

この例外を使用して、非同期 SCA メッセージの有効期限が切れたことを示します。有効期限は、サービス参照で `RequestExpiration` 修飾子を使用して設定できます。

2. `ServiceTimeoutRuntimeException`

この例外を使用して、非同期要求への応答を構成された期間内に受信しなかったことを示します。有効期限は、サービス参照で `ResponseExpiration` 修飾子を使用して設定できます。

3. `ServiceUnavailableException`

この例外を使用して、インポートを介して外部サービスを呼び出し中にスローされた例外が存在することを示します。

4. `ServiceUnwiredReferenceRuntimeException`

この例外を使用して、コンポーネントでのサービス参照が正しくワイヤーされていないことを示します。

関連概念

ユース・ケース: 失敗イベントからのデータのリカバリー (recovering data from failed events)

ユース・ケースは、リカバリー・シナリオでのコンテキストとして使用されます。このユース・ケースでのビジネスには、新規アカウントを作成する要求を受け取るアプリケーションがあります。

開発の一部としてのエラー防止

開発プロセスの一部として、エラー防止手段を含めることができます。

開発プロセスの一部としてのエラー防止手段では、プロジェクトをロールアウトするために設置されるガバナンスと開発プロセスに焦点を合わせており、主に、テスト、チューニング、測定、および再テストのアクティビティが関係しています。

開発プロセスの一部としてのエラー防止手段には、以下の要素が含まれています。

- 総合的なテストによる問題の防止
- 継続的かつ定期的にスケジュールされた環境調整
- インフラストラクチャー・モニター

エラー防止: 総合的なテスト

総合的な機能およびシステム・テスト計画を実装することにより、リカバリーが必要になる問題の発生を防止できます。

一般に、デプロイされたソリューションのテストは、以下のように分類できます。

- 機能テスト

機能テストは、アプリケーションに実装された機能が、指定のビジネス要件を満たすか確認します。機能テストは、ビジネス・ユーザーとアプリケーション設計者によって作成されます。

- システム・テスト

システム・テストは、パフォーマンス、高可用性、およびリカバリーのサービス・レベル・アグリーメントを検証するように設計されています。

システム・テストでは、パフォーマンス・テストと高可用性テストのような要素を組み合わせ、極端な実稼働状況でのシステムのリカバリーを評価することが重要です。

機能テストとシステム・テストの両方に関して、自動化することを強くお勧めします。自動化テストは、組織にとって回帰バグが入り込むのを阻止する有効な方法です。

関連概念

93 ページの『リカバリー: ファースト・ステップ』

管理者は一般的な手法のファースト・ステップ・チェックリストに従うことによって、ソリューション・リカバリー・プロセスを容易に行うことができます。

関連情報



Problem determination in WebSphere Process Server

エラー防止: 環境のチューニング

チューニング演習は、システム開発ライフ・サイクルの正規の部分です。大規模なアプリケーションのデプロイメントごとに、パフォーマンス評価をスケジュールに入れてください。

実稼働環境へのソリューションのデプロイに対する前提条件として、実動前環境でソリューションの評価とテストを実行してください。これにより、既存のアプリケーションおよび現在のシステム・パラメーターとリソースに対する新規ソリューションの影響を測定できます。実動前環境でのソリューションの評価およびテストを怠ると、そのソリューションがリカバリーに関して問題を持つようになる確率が高くなります。






パフォーマンス・テスト計画のプロセスおよび実行について説明した多くのリソースが、一般に入手可能になっています。そのような資料を調べ、ご使用のアプリケーションとトポロジーに適切なテスト計画を作成してください。

WebSphere Process Server のパフォーマンスとチューニングについての情報が掲載されている IBM Redbooks、および WebSphere Process Server のパフォーマンスとチューニングに関するテクニカル・ホワイト・ペーパーを参照してください。また、Business Process Management (BPM) の新規リリースと IBM の Connectivity 製品すべてに付属している、パフォーマンス・レポートも参照してください。

関連情報



チューニング

-  IBM WebSphere Business Process Management Performance Tuning
-  Endurance testing with WebSphere Process Server
-  WebSphere Business Integration V6.0.2 Performance Tuning
-  Performance Tuning Automatic Business Processes for Production Scenarios with DB2
-  WebSphere Process Server V6 – Business Process Choreographer Performance Tuning of Human Workflows Using Materialized Views

エラー防止: インフラストラクチャーのモニター

インフラストラクチャーのモニター、およびインフラストラクチャー・モニター・ツールを使用することは、実動システムの要件の 1 つです。

ITCAM for SOA および *Tivoli® Performance Viewer* のようなモニター・ツールにより、システム管理者は、重要なシステムの振る舞いをモニターして、停止状態を引き起こしかねない問題を検出することができます。

実動システムでの基本的なレベルの IT モニターは、可用性サービス・レベル・アグリーメントに適合するために欠かせません。

サービス・コンポーネント・イベントのパフォーマンスおよびビジネス・プロセスのモニターについては詳しくは、WebSphere Process Server インフォメーション・センターの『モニター』のセクションを参照してください。

関連情報

-  モニター

IBM Tivoli Composite Application Manager Family for SOA:

IBM Tivoli Composite Application Manager Family (ITCAM) for SOA を使用して、WebSphere Process Server をモニターできます。また、ITCAM for SOA を使用して、問題メディエーションを自動化し、ソリューションの構成およびデプロイメントを管理することもできます。

ITCAM for SOA には以下の機能が組み込まれています。

SOA サービスの管理

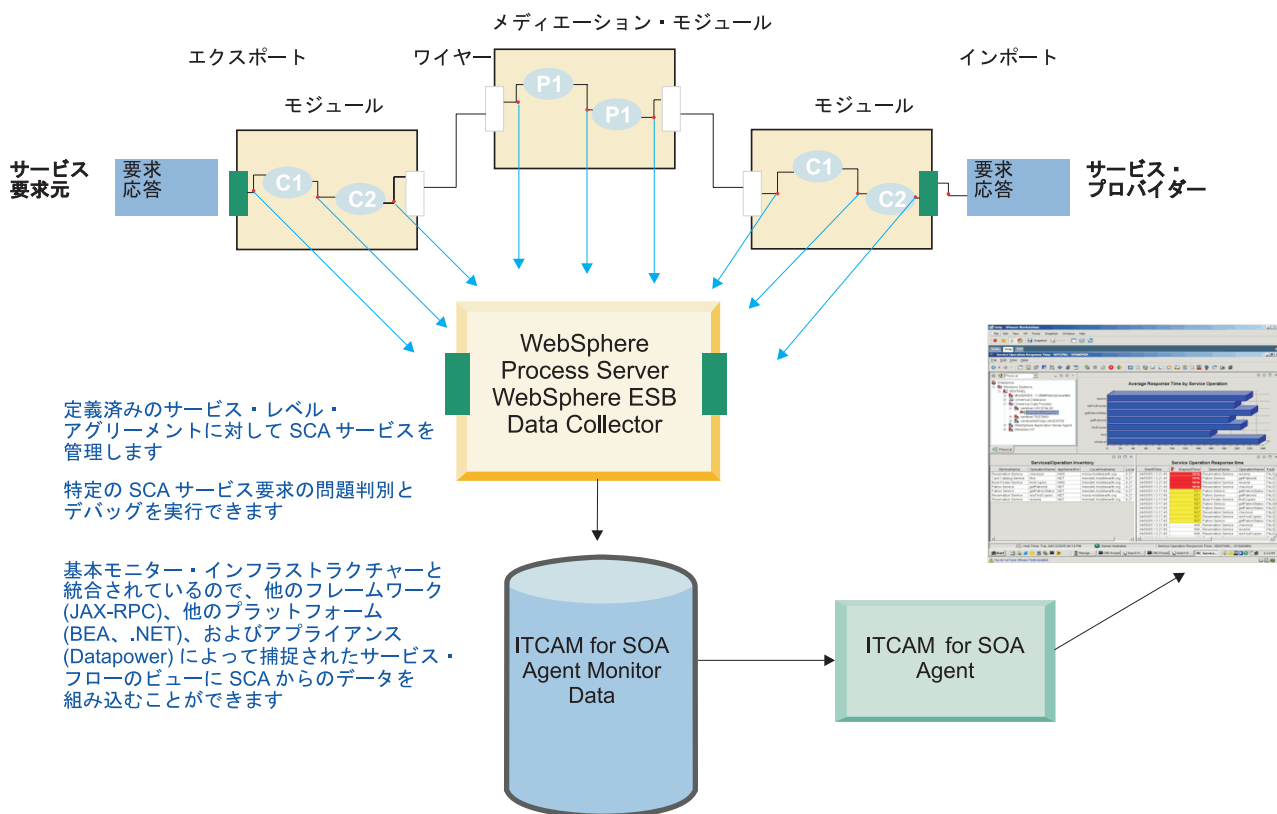
- SOA サービス対話内の可視性
- メッセージ・コンテンツ・パターンとトランザクション・フロー・パターン内の可視性
- テクノロジーとプラットフォームの境界を越えてパフォーマンスのボトルネックを識別して分離する能力
- 軽量で、業界標準の、ARM ベース・パフォーマンス計測
- ポリシーの高いパフォーマンスと柔軟な施行
- 統合が容易な標準ベースの計測

ビジネス・プロセスのモニター

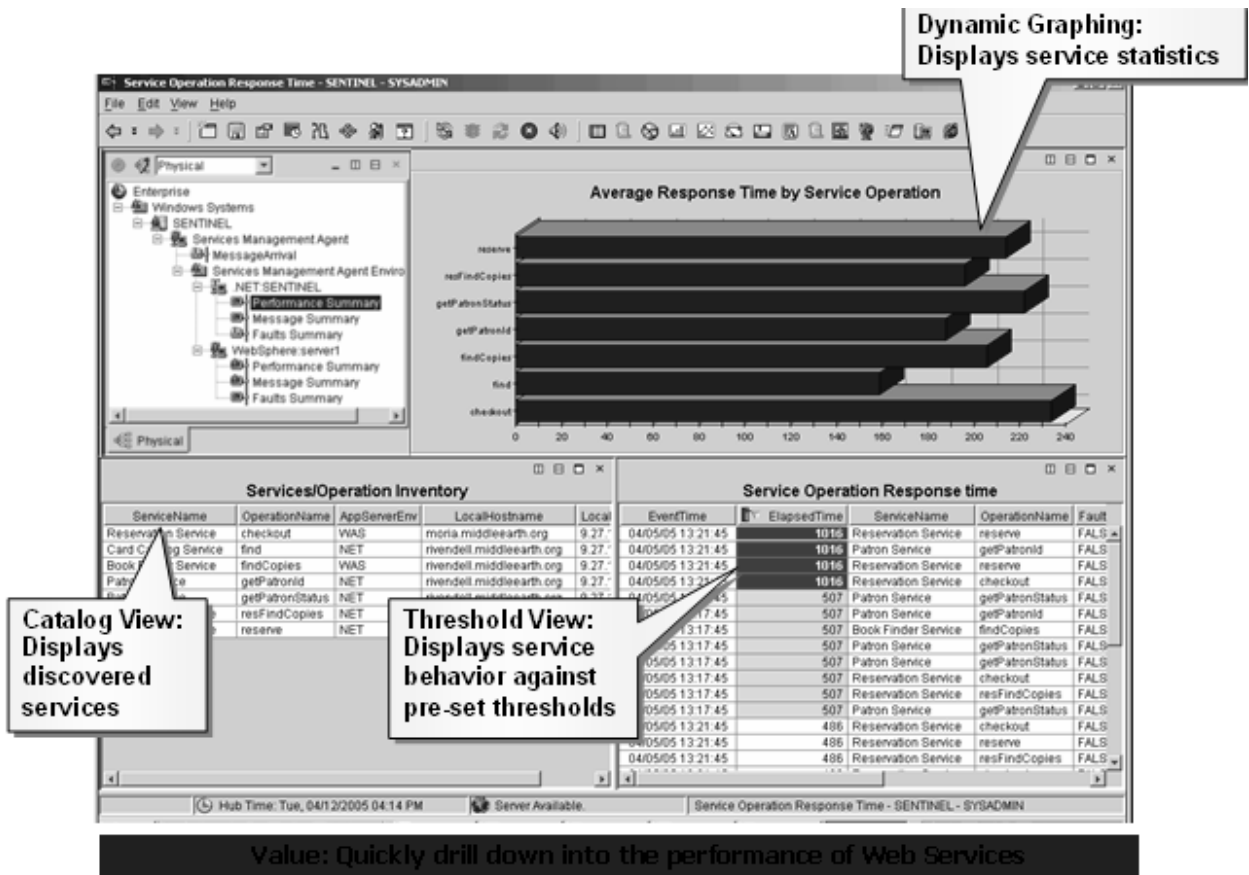
- 未完了プロセスの管理
- アクティブ・プロセスのビジネス・パフォーマンスのモニター
- ビジネス・シチュエーションの検出およびアクションの実行
- 収集されたプロセス・データからのビジネス・インテリジェンスの収集
- 停止した、または動作速度の遅いアプリケーションを識別して素早く修正する総合的な deep-dive モニター
- リアルタイム・メトリックおよびヒストリカル・データ分析

IBM Tivoli Composite Application Manager Family (ITCAM) for SOA の例

以下の例は、IBM Tivoli Composite Application Manager Family(ITCAM) for SOA による、サービス、応答時間、メッセージ数、およびメッセージ・サイズのモニター方法を示しています。

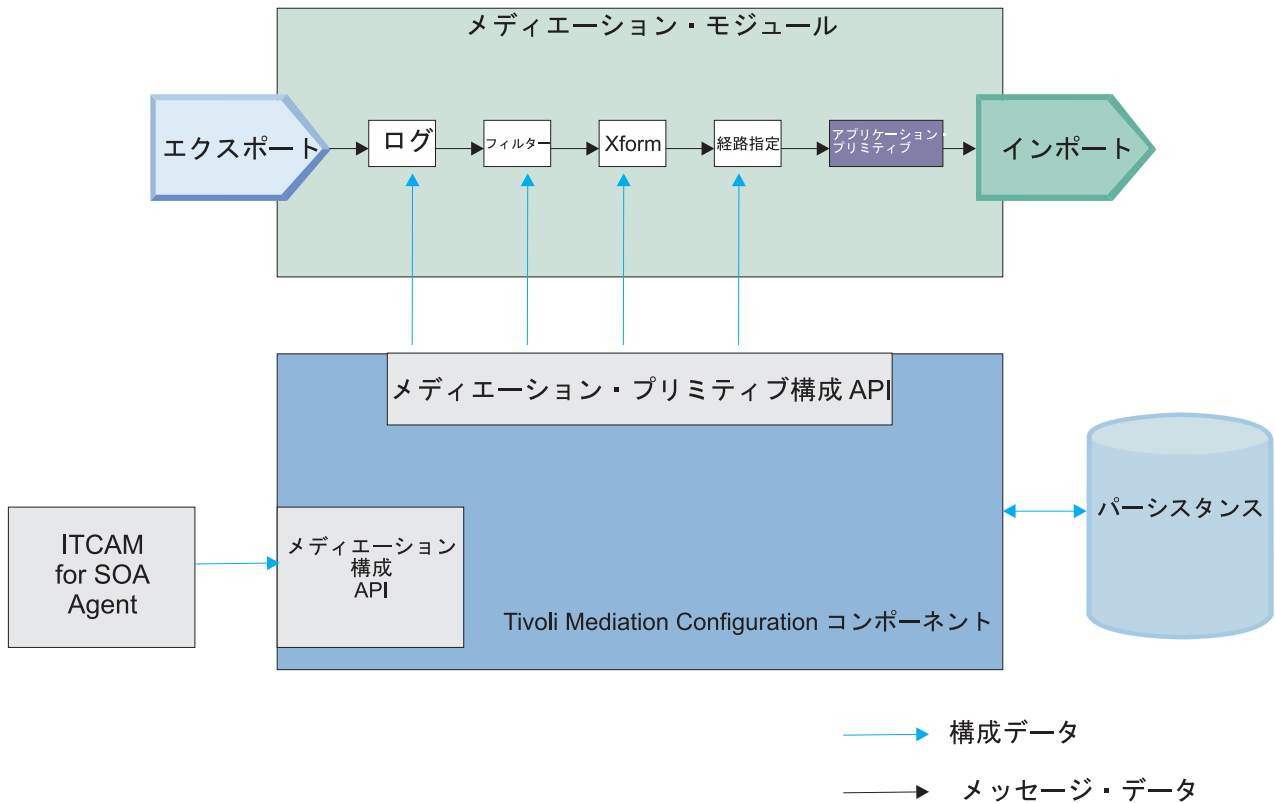


以下の例は、IBM Tivoli Composite Application Manager Family (ITCAM) for SOA に表示される、操作あたりの画面測定統計と、問題を検出するために設定可能なしきい値を示しています。



前述のとおり、IBM Tivoli Composite Application Manager Family (ITCAM) for SOA には、WebSphere ESB と連携してメディエーション・フロー構成を動的に変更する特別な機能が組み込まれています。

以下の図は、IBM Tivoli Composite Application Manager Family (ITCAM) for SOA のメディエーション・フロー構成機能を示しています。



関連情報

IBM Tivoli Composite Application Manager Family Installation, Configuration, and Basic Usage

問題判別方法の文書

実稼働環境にデプロイするソリューションについて、明確で分かりやすい問題判別の方法論を確立します。

これは、問題判別方法の文書を維持し、文書化された方法を一貫して実行することを意味します。

操作マニュアルにおいて、ソリューション固有の問題判別方法論を文書化することをお勧めします。この操作マニュアルには、ソリューション固有の問題判別に関係した以下の種類の情報を含めてください。

- 問題判別中に観察結果を記録するための確立された形式

確立された形式を使用することにより、観察結果を記録する際の一貫性を保つことができます。Excel 形式のスプレッドシートは、一般的な「観察レポート作成ツール」です。

- トレース情報のリスト

ソリューション固有の問題判別用に、以下のトレース情報を含めてください。

- 有効にするトレースのリスト
- トレースを有効にするサーバーのリスト

- トレースを有効にするときの条件の記述

トレースを実装する前に、トレースによって問題が悪化しないことを確認してください。「すべてを有効にする」ことは適切ではありません。トレースの仕様は観察された状況に対して適切なものになっている必要があるため、トレースを有効にする場合は慎重に行ってください。インテリジェント・シチュエーション分析を使用して正しい診断情報を収集します。適切なトレース・レベルを実装する方法が不明確な場合は、IBM サポートにお問い合わせください。

- 冗長なガーベッジ・コレクション (verbosegc) を有効にする

冗長なガーベッジ・コレクション (GC) データには、特定のアプリケーションでの GC の実行方法に関する詳細が含まれています。この情報は、パフォーマンス問題の分析や、アプリケーションの GC 設定を調整する場合に役立ちます。

- ヒープ・ダンプの生成

ヒープ・ダンプ機能は、Java ヒープにあるすべてのオブジェクトのレコードをテキスト・ファイルに出力する IBM JVM の機能です。

各オブジェクトのサイズおよびアドレスが記録されるとともに、そのオブジェクトが参照するすべてのオブジェクトのアドレスが記録されます。この情報により、メモリーの大きな部分を占めているオブジェクトを把握することができます。

- Java.core の作成

javacore ファイルを分析して問題判別を実行することは、IBM Java 仮想マシン (JVM) で発生している可能性があるエラー状態の根本原因を判別するのに効果的です。

- 問題管理レコード (PMR) を開く前に収集する必要がある、ログの場所と種類。IBM の「must gather」スクリプトの正しい使用法を定義します。
- すべての保守パッケージ情報を含むバージョン情報 (versionInfo) の収集
- さまざまな問題が発生するときにデータベースによって記録されるログおよび情報を収集するための、データベース固有手順

作成するソリューション固有の問題判別文書は頻繁に更新される文書 として扱い、機能およびシステム・テストによる観察で新しい手法が見つかるたびに、毎回保守および更新してください。

注: 問題判別および問題報告で活用可能な IBM Support Assistant を始めとする他のツールに精通し、使用してください。上に述べた情報を収集することは、その情報を含めることで PMR のサイクル・タイムが著しく短縮されるので、すべての新規 PMR を開く場合の前提条件としてください。

関連情報



Generating an IBM Heap



製品のバージョン情報および履歴情報



WebSphere Application Server で冗長なガーベッジ・コレクション (verbosegc) を有効にする

 IBM Support Assistant

 クロス・コンポーネント・トレースの有効化


ソフトウェア適用状況

ソフトウェア適用状況は、デプロイされているソリューションの最新ソフトウェアを維持するための手法です。

デプロイされているソリューションのソフトウェア適用状況を維持することは重要です。

IBM は、製品ベースで存在するプログラム診断依頼書 (APAR) の適用を支援するフィックスパックを定期的に作成しています。このサービス・パッケージには、必須のコード変更が必ず含まれています。詳しくは、公開されている APAR フィックスのリストを参照してください。

関連情報

 お客様サポートとの連絡

 フィックスの入手

 WebSphere Process Server サポート

エラー処理方針とソリューション・リカバリー

WebSphere Process Serverには、リカバリーのために利用できるエラー処理機能とツールが組み込まれています。

ソリューションを構築するアーキテクチャー・チームは、エラー処理とリカバリーにおいて WebSphere Process Server のツールと機能を利用する方法を理解する必要があります。

アーキテクチャー・チームには、アプリケーション開発チームが遵守しなければならないエラー処理標準を作成する責任があります。

プロジェクトのエラー処理方針には、以下の点を含める必要があります。

- 作業単位の適切な使用法 (トランザクションおよびアクティビティー・セッション)
- 障害および ServiceBusinessExceptions の宣言および使用法
- 特に BPEL やメディエーション・フローのコンポーネントなど、すべてのコンポーネント・タイプで一貫した障害処理
- 再試行論理および「エラーの継続」Business Process Choreographer 機能の使用法
- 完了したプロセス・インスタンス削除の適切な設定
- 同期および非同期の呼び出しパターンの正しい使用法
- インポート・タイプとエクスポート・タイプの適切な使用法
- メディエーション・フローでの再試行機能の正しい使用法

アーキテクチャー・チームは、上の点に加えて、WebSphere Process Server の組み込みリカバリー機能 (Failed Event Manager など) が適切に利用される設計パターンを作成する必要があります。

関連タスク

52 ページの『デプロイメント環境パターンの選択』

IBM 提供のいずれかのパターンを選択するか、独自のカスタム・デプロイメント環境を作成することによって、デプロイメント環境を構成できます。このトピックでは、IBM 提供の各パターンでサポートされる機能をリストします。

29 ページの『提供されたパターンの 1 つに基づくデプロイメント環境の計画』

このシナリオは、Service Component Architecture (SCA) アプリケーションに対するスケーラビリティ、可用性、およびサービス品質の要件がある場合で、IBM 提供のパターンの 1 つでそれらの要件を満たすことができる場合に使用します。

関連情報



ビジネス・プロセスでの障害処理および補正処理



ビジネス・プロセスでのフォールトの処理

安定した環境の保守

安定した環境を実現し、システムおよびアプリケーションの障害が発生する確率を低く抑えるために実行できる、いくつかの追加ステップがあります。

以降のセクションでは、ソリューションの安定性とシステムのリカバリーに影響する手動プロセスの数を減らすために、インフラストラクチャー・チームが活用できる手法について説明します。

自動化環境作成

スクリプト・フレームワークは、環境の作成時の整合性に寄与します。

管理コンソールから実行可能なすべてのアクションは、スクリプトを使用して実行することもできます。特定のニーズに応じて使用およびカスタマイズすべき既存の IBM サービス・アセットがあります。これらのスクリプトは、調整を実行するたびに保守することができます。テスト環境で作業する場合、環境の再作成が必要になることが多くあります。テスト環境の作成などの反復アクションを実装するには、スクリプトがもっとも効率的な方法です。テスト・システムのスクリプトは後で変更して、実動システムの作成に使用できます。

IBM Software Services for WebSphere (ISSW) の担当者と自動化デプロイメントについて話し合うか、ご使用の WebSphere Application Server 実稼働環境で活用されているのと同様の手順で構築してください。

関連情報



スクリプトの使用 (wsadmin)



コマンドおよびスクリプト

自動化アプリケーション・デプロイメント

自動化スクリプトを使用して、適切な環境へのアプリケーションまたはソリューション・グループのデプロイメントを支援します。

優れた設計の「ビルド、パッケージ、およびデプロイ」モデルには数多くの利点があり、例えば、開発者の生産性向上、ビルドおよびコード修正にかかるターンアラウンド・タイムの短縮、アプリケーション・コードにおける整合性の向上、およびデプロイメント・ポリシーの強化などが挙げられます。

アプリケーションまたはソリューション・グループのデプロイで使用する自動化スクリプトは、ご使用の環境を作成する際の自動化プロセスを補完します。

スクリプトを使用する自動化アプリケーション・デプロイメントでは、環境への手操作による介入が減り、再デプロイメントやリカバリーにヒューマン・エラーが入り込む余地も少なくなります。

IBM Software Services for WebSphere (ISSW) の担当者と自動化デプロイメントについて話し合うか、ご使用の WebSphere Application Server 実稼働環境で活用されているのと同様の手順で構築してください。

関連情報



スクリプトを使用したアプリケーションのデプロイ



Sample Scripts for WebSphere Application Server

リカバリー方法の計画

リカバリー方法を計画すれば、正常にリカバリーする確率が高くなります。

高可用性 (HA)

高可用性 (HA) とは、どのような障害が発生しても処理を続行し、事前定義されたあるサービス・レベルに応じた処理機能を維持する IT サービスの能力のことです。

ソリューション・リカバリーを促進するために行うことができる重要な作業の 1 つは、高可用性 (HA) を付与してシステムを構成することです。対象となる障害には、保守およびバックアップなどの計画イベントと、ソフトウェア障害、ハードウェア障害、電源障害、および災害などの未計画イベントの両方が含まれます。クラスター環境は、本質的に高可用性を備えています。これは、クラスター化されたシステムは、ノードまたはデーモンの障害が発生すると再構成され、作業負荷をクラスター内の残りのノードに再配分することができるからです。

高可用性ソリューションは、ハードウェア、ソフトウェア、およびサービスを組み合わせられており、リカバリー・プロセスが完全に自動化され、ユーザー・アクティビティを中断させません。HA ソリューションは、リカバリー時間が最短になる直前のリカバリー・ポイントを見つける必要があります。

高可用性ソリューションでは、アプリケーション・サーバーが問題を検出すると、そのトランザクションと関連するデータが別のサーバー (同じデータ・センター

内、または災害の場合は別の地理的位置にあるサーバー) に自動的に移動されます。トランザクションおよび関連データを別のサーバーに移すことを、**ピア・リカバリー** と呼びます。

関連資料

82 ページの『ピア・リカバリー』

ピア・リカバリーは、同じクラスター内の別のメンバーによって実行されるリカバリーであり、手動でも自動でも開始できます。ピア・リカバリー処理 (自動ピア・リカバリーまたは手動ピア・リカバリー) は、WebSphere の高可用性環境と緊密に連携しています。

関連情報



WebSphere Application Server Network Deployment V6: High Availability Solutions

リカバリーの環境および目的

リカバリー・スペクトルの範囲は、テスト環境と実稼働環境、および異なるリカバリー目的 (システム・リカバリーとアプリケーション・リカバリー) に及びます。リカバリーの目標および目的は、リカバリー元となる環境に応じて異なります。

関連概念

80 ページの『トランザクションのプロパティとソリューション・リカバリー』

WebSphere Process Server は WebSphere Application Server をベースとしているため、ビジネス・トランザクションを実行するトランザクション・モデル をサポートしています。

関連情報



Selecting your deployment pattern



トランザクションの高可用性



Asynchronous replication of WebSphere Process Server and WebSphere Enterprise Service Bus for disaster recovery environments

実稼働環境でのリカバリー

実稼働環境での目標は、整然とした一貫性のある方法でシステムに入力された要求すべてを処理することです。この環境ではデータを保持する必要があり、システムを使用できなかつたり、データを損失したりする状況を最小限に抑えるための手段をすべて実施する必要があります。

実稼働環境に関する次の側面を考慮してください。

- トポロジー・タイプ

ご使用の実稼働環境に最適なトポロジー・タイプを理解しておく必要があります。ご使用の環境に適切なトポロジーを見つけるには、アプリケーションのプロパティと機能に無関係の要件を分析する必要があります。

トポロジー・タイプについて詳しくは、WebSphere Process Server インフォメーション・センターのトピック『提供されたパターンの 1 つに基づくデプロイメント環境の計画』を参照してください。

- リカバリーする必要がある条件についての理解と洞察

例えば、クラスターに複数のクラスター・メンバーがある場合は、リカバリーする必要がある唯一のコンポーネントが単一クラスター・メンバーであり、ワークロード管理機能によって処理がすでに「実行中のサーバー」に転送済みになっている可能性があります。このような場合は、サーバーの再始動によってリカバリーが強制され、そのサーバーはクラスターに再び参加します。

一部の高可用性 (HA) 構成では、別のところにある 1 つのサーバーから失敗したトランザクションをリカバリーできます (ピア・リカバリー)。

実動データのリカバリーでは、システムおよびアプリケーションの 2 つのレベルで成功する必要があります。

関連タスク

29 ページの『提供されたパターンの 1 つに基づくデプロイメント環境の計画』このシナリオは、Service Component Architecture (SCA) アプリケーションに対するスケーラビリティ、可用性、およびサービス品質の要件がある場合で、IBM 提供のパターンの 1 つでそれらの要件を満たすことができる場合に使用します。

テスト環境でのリカバリー

テスト環境の目標および前提の数は、実稼働環境の目標や前提と異なります。

テスト環境での目標は、新規テストをできるだけすぐに実施できるような方法でシステムをリカバリーすることです。データの保持は不要で、システム内のすべての要求は破棄可能という前提があります。

注: これは、「リカバリー」テストとは異なります。リカバリー・テストでは、実動シナリオとして提供される推奨事項を利用し、プロジェクトのシステム・テスト・フェーズ中に実行されます。

システム・リカバリー

システム・リカバリーとは、ソリューションのインフラストラクチャーへの悪影響を修正するために (手動または自動で) 実行されるオペレーションのことです。

WebSphere Process Server のソリューションは、基本的なインフラストラクチャーの要件に依存しています。WebSphere Process Server は、以下の条件のいずれかにより悪影響を受ける可能性があります。

- 電源異常
- ネットワークの破損
- データベース障害
- ハードウェア障害

これらのタイプのいずれかの中断が発生した場合は、WebSphere Process Server システムのリカバリー前に、それに対処して修正する必要があります。

基本的なインフラストラクチャー要件における障害または中断に対処するとすぐ、WebSphere Process Server は、WebSphere からの継承機能に基づいてアプリケーション・リカバリーを開始します。

関連概念

93 ページの『リカバリー: ファースト・ステップ』

管理者は一般的な手法のファースト・ステップ・チェックリストに従うことによって、ソリューション・リカバリー・プロセスを容易に行うことができます。

アプリケーション・リカバリー

アプリケーション・リカバリーとは、未完了ビジネス・トランザクションのリカバリーと解決を意味します。

システムで障害 (電源異常など) が発生すると、アクティブなトランザクションの多くが影響を受けます。それらのトランザクションすべては、プロセス・フローのさまざまな段階にあります。ここでは、システムのリカバリー・プロセスの一環としてシステムがそれらのトランザクションを処理する方法を説明します。

アプリケーション・リカバリーを最後まで完全に実行するには、それらのアプリケーション自体が、設定されている防止手段を遵守する必要があります。

リカバリーおよびトランザクションの適用範囲を念頭に、ベスト・プラクティスに従ってアプリケーションが開発されていない場合、アプリケーション・リカバリーが最後まで完全に実行される確率は低くなります。

設計に問題がある、または「未調整」のシステムまたはアプリケーションでは、それ以外のアプリケーションが新規イベントの処理を開始した後も未解決のまま残る、未完了のトランザクションまたはプロセスの割合が高くなります。これは、WebSphere Process Server だけでなく、すべての J2EE アプリケーションとアプリケーション・サーバーの場合にも当てはまります。

注: 「未調整」という語句は、パフォーマンスの考慮事項やエラー処理手法について考慮せずに、すべてのコンポーネントでデフォルト設定を使用するソリューションを指します。

未解決イベントは、実行状態のままのプロセスや再実行依頼できない失敗イベントなど、さまざまな形態で発生する可能性があります。完全リカバリーのためにアプリケーション内部で必要とされる変更を判別するには、これらのイベントのリカバリー後分析を行うことが必要になります。そのような変更は、総合的な機能/システム・テスト計画の実行時に発見する必要があります。

関連概念

93 ページの『リカバリー: ファースト・ステップ』

管理者は一般的な手法のファースト・ステップ・チェックリストに従うことによって、ソリューション・リカバリー・プロセスを容易に行うことができます。

関連情報

 失敗イベントの管理

トランザクションのプロパティとソリューション・リカバリー

WebSphere Process Server は WebSphere Application Server をベースとしているため、ビジネス・トランザクションを実行するトランザクション・モデル をサポートしています。

WebSphere Process Server はこのトランザクション・モデルに基づいて構築されており、疎結合の SOA アプリケーションおよび BPM アプリケーションを提供します。

これは、技術的に 2 つの事柄を意味します。

1. WebSphere Process Server は、トランザクション・アプリケーション実行パターンを実現するためにデータベース・システムおよびメッセージング・システムに依存しています。
2. トランザクションは、メッセージング・システムおよびデータベース・システムにおいて重要な役割を担っています。

トランザクションは、ACID プロパティに対応しています。トランザクションは、原子性、一貫性、独立性、および耐久性を持つときに、ACID 準拠と見なされます。

WebSphere Process Server ではデータベース・システムおよびメッセージング・システムを使用して、「疎結合された」パターンを実現します。WebSphere Process Server はデータベースを更新してメッセージを送信します。データベースの更新およびメッセージの処理は同じトランザクションでコミットされます。

「疎結合」パターンの別の特徴は、メッセージング・システムからメッセージを取り出して、データベースを更新することです。この処理中に障害が発生すると、イベントはあたかも未読であったかのようにメッセージ・キューに戻ります。WebSphere Process Server には再試行メカニズムが存在し、5 回試行した後でイベントは Failed Event Manager に渡されます。「疎結合」という句は、すべての処理が 1 つの大きなトランザクション内で発生しなくてもよいことを表しています。

システム障害イベントでのデータ損失の回避

利用可能なリソース・マネージャーを適切に調整および構成しておけば、システムのある部分に障害があってもデータは失われません。ロールバックおよびリカバリーのメカニズムなどのトランザクションの健全性は、障害が発生してもデータが失われないようにする、WebSphere のキー・コンポーネントです。

WebSphere のロールバックおよびリカバリー・メカニズムを機能させるには、リソース・マネージャー (データベースおよびメッセージング) を正しくセットアップする必要があります。例えば、データベースのロック・タイムアウトを適切に設定し、サーバーのリカバリー時に、ロック状態になることなく、データベースでコミットまたはロールバックのどちらかを完了できるようにする必要があります。

WebSphere Process Server は、WebSphere Application Server の機能を拡張する機能を追加することで、予期しない障害からデータを回復する完全なソリューションを提供します。

リカバリー機能の使用可能化の概要

WebSphere Process Server のコア・リカバリー・モデルの基本は作業単位です。システムは、システム操作中に発生する障害を、実行される単一の作業単位に注目して処理および回復できるため、中断することなくサービスを提供できます。このタイプのリカバリーは、一連の再試行メカニズムとエラー・キューによって行われます。アプリケーション設計の一部に、システム・エラーをアプリケーション・エラーと区別する機能を組み込んでください。システム・エラーは、呼び出し側コンポーネントをサポートするインフラストラクチャーに戻されます。そこでは、追加のシステム・レベル・リカバリーが試みられたり、より一般的なビジネス例外への変換が行われたりします。自動的に実行されるさまざまな再試行メカニズムを構成できます。また WebSphere Process Server では、必要な場合に人間の詳細な介入を可能にする一連のコンソールおよび対応するプログラミング・インターフェースが提供されています。これらの機能およびこれらが扱う障害の多くは、この作業を実行するサーバーが新しい要求の処理を継続している間も活用できます。

使用不可サーバー - 概要

障害によって、高い可用性を持つ WebSphere クラスタ内の 1 つまたは複数のサーバーが使用不可になった場合、システム内の追加のリカバリー機能が以下のように呼び出されます。

1. インバウンド処理を経路指定して障害システムから引き離す

これは、基盤となる WebSphere Application Server のワークロード管理機能を使用して実行されます。この機能は、プロトコル、トポロジー、および構成によって異なります。

2. 管理者がアクションを開始する

システムは、全体としてアクティブで使用可能な状態を続けますが、管理者はリカバリー操作を実行できます。

管理者のアクションの目的は、基本的な優先順位を決定し、停止しているサーバーを再始動することです。この再始動によってトランザクション・ログが再生され、ほとんどのサーバー・ダウンの状況がクリーンアップされます。

完全リカバリーを管理するには、WebSphere Process Server で提供されているエラー処理メカニズムを使用することが必要な場合があります。

使用不可クラスター - 概要

サーバー・クラスター全体が使用できないか、または応答しない場合、より複雑な一連のリカバリー・アクションが必要です。例えば、データベースなどの共用リソースが利用できない場合は、クラスター内のどのサーバーでも一様に処理の完了が難しくなります。

共用リソース・リカバリーを処理する手順は、障害が発生している共用リソースによって決まります。さまざまな WebSphere の技法を適用して、全体的なダウン時間を最小化し、停止した処理を再開することができます。

壊滅的な障害 - 概要

壊滅的な状況では、マシン全体が利用できないか、またはサーバーがリカバリー可能ではありません。このような事例では、WebSphere の拡張機能を使用して、サーバー障害のリカバリーを同じクラスター内の別のサーバー上で実行することができます。この機能では、ログを共有するためのネットワーク接続されたストレージまたはその他のメカニズムを用意するという前提条件を満たすことで、この種類のリカバリーも可能になります。同じクラスターの別のメンバーによる障害サーバーのリカバリーについては、『ピア・リカバリー』を参照してください。

関連概念

95 ページの『ユース・ケース: 失敗イベントからのデータのリカバリー (recovering data from failed events)』

ユース・ケースは、リカバリー・シナリオでのコンテキストとして使用されます。このユース・ケースでのビジネスには、新規アカウントを作成する要求を受け取るアプリケーションがあります。

77 ページの『リカバリーの環境および目的』





リカバリー・スペクトルの範囲は、テスト環境と実稼働環境、および異なるリカバリー目的 (システム・リカバリーとアプリケーション・リカバリー) に及びます。リカバリーの目標および目的は、リカバリー元となる環境に応じて異なります。

関連資料

『ピア・リカバリー』

ピア・リカバリーは、同じクラスター内の別のメンバーによって実行されるリカバリーであり、手動でも自動でも開始できます。ピア・リカバリー処理 (自動ピア・リカバリーまたは手動ピア・リカバリー) は、WebSphere の高可用性環境と緊密に連携しています。

関連情報

-  サーバーおよびクラスターの管理
-  イベントの使用
-  ビジネス・プロセスのトランザクションの振る舞い
-  ビジネス・プロセスでの補正処理 (Compensation handling in business processes)

ピア・リカバリー

ピア・リカバリーは、同じクラスター内の別のメンバーによって実行されるリカバリーであり、手動でも自動でも開始できます。ピア・リカバリー処理 (自動ピア・リカバリーまたは手動ピア・リカバリー) は、WebSphere の高可用性環境と緊密に連携しています。

HA マネージャー

WebSphere では、HA マネージャー・コンポーネントを使用して、アプリケーション・サーバーにより提供されるサービスをモニターします。これらのサービスとしては、メッセージング、トランザクション・マネージャー、ワークロード管理コントローラー、およびクラスター内の他のアプリケーション・サーバーがあります。

さらに HA マネージャー・コンポーネントでは、Network Attached Storage (NAS) デバイスを使用して、クラスター内の各アプリケーション・サーバーからのトランザクション・ログを保管します。

HA マネージャーは、定義された HA クラスターで障害が発生したサーバーに関して、未確定 トランザクションと未完了 トランザクションの両方の自動ピア・リカバリーを実行します。未確定トランザクションとは、例えばノードの除去によりメッセージング・エンジンが破壊されるなどの例外的な状況が原因で、いつまでも未確定状態のまま動かなくなったトランザクションです。未確定トランザクション状態が発生するのは、データベースがフェーズ 1 のコミット処理を終えてからフェーズ 2 を開始するまでの間です。未完了トランザクションとは、コミット・プロセスの「準備フェーズ」をまだ完了していないトランザクションです。この場合、トランザクションまたはメッセージはどこかに残っており、その場所でトランザクションまたはメッセージをリカバリーすることができます。HA マネージャーによって実行される自動リカバリー機能により、クラスターは 1 つ以上のクラスター・メンバーに障害が発生した場合にバランスを取り戻すことができます。

自動ピア・リカバリーと手動ピア・リカバリー

自動ピア・リカバリー は、デフォルトのピア・リカバリー開始方式です。アプリケーション・サーバーに障害が発生した場合、WebSphere Application Server は、そのアプリケーション・サーバーの代わりにピア・リカバリー処理を実行するサーバーを自動的に選択します。高可用性を有効にしたり、各クラスター・メンバーのリカバリー・ログ・ロケーションを構成したりするほかには、このモデルを使用するために追加の WebSphere Application Server 構成ステップを行う必要はありません。

手動ピア・リカバリー は、明示的に構成する必要がある特定のピア・リカバリー方式です。アプリケーション・サーバーに障害が発生した場合、オペレーターは管理コンソールを使用して、そのアプリケーション・サーバーの代わりにリカバリー処理を実行するサーバーを選択することができます。

ピア・リカバリー参照情報

『IBM WebSphere 開発者用技術ジャーナル: WebSphere Application Server V6 におけるトランザクションの高可用性とデプロイメントに関する考慮事項 (IBM WebSphere Developer Technical Journal: Transactional high availability and deployment considerations in WebSphere Application Server V6)』という表題の記事で、自動および手動によるピア・リカバリーの要件、セットアップ、管理について説明しています。

追加の文書については、WebSphere Application Server インフォメーション・センターおよび『WebSphere Application Server V6 Scalability and Performance Handbook』を参照してください。

- WebSphere Application Server V6 Scalability and Performance Handbook
- WebSphere Application Server インフォメーション・センターの『トランザクション・プロパティの、ピア・リカバリー用の構成』
- WebSphere Application Server インフォメーション・センターの『トランザクション・サービスの手動ピア・リカバリーの管理』

関連概念

87 ページの『リカバリーのトリガー』
ソリューション・リカバリーは、さまざまなトリガーの結果として必要になり
ます。

80 ページの『トランザクションのプロパティとソリューション・リカバ
リー』

WebSphere Process Server は WebSphere Application Server をベースとしている
ため、ビジネス・トランザクションを実行するトランザクション・モデル をサ
ポートしています。

61 ページの『エラー防止とリカバリーの概要』

エラー防止とリカバリーの情報では、システム障害を引き起こす問題を回避する
方法について説明し、通常の状態と異常な状態の両方で発生する可能性があるシ
ステム障害からリカバリーする方法について、情報を提供しています。

87 ページの『リカバリーのトリガー』

ソリューション・リカバリーは、さまざまなトリガーの結果として必要になり
ます。

76 ページの『高可用性 (HA)』

高可用性 (HA) とは、どのような障害が発生しても処理を続行し、事前定義され
たあるサービス・レベルに応じた処理機能を維持する IT サービスの能力のこと
です。

エクスポート・バインディング

システムを完全に停止するには、使用可能なエクスポート・バインディングによっ
てサポートされるさまざまなタイプの要求呼び出しを考慮する必要があります。

SCA 呼び出しパターン

次の表は、各種エクスポート・バインディングに使用される SCA 呼び出しパター
ンのタイプを示したものです。

表2. EIS エクスポート・バインディングと関連する呼び出しパターン

エクスポート・バインディング	操作のタイプ	パフォーマンス属性と対話スタイル	呼び出しスタイル
EIS	片方向	非同期	非同期 (デフォル ト)
		同期	同期
	要求応答	任意の値	同期

表3. エクスポート・バインディングおよび関連する操作タイプと呼び出しスタイル

エクスポート・バインディング	操作のタイプ	呼び出しスタイル
EIS	片方向または要求応答	同期
MQ または MQ JMS	片方向	非同期
SCA JMS	片方向	非同期
	要求応答	コールバック付き非同期
Web サービス (soap/http) また は (soap/jms)	片方向または要求応答	同期

使用するアプリケーションやトポロジーに応じて、さまざまな方法で同期通信を停止することができます。同期通信の停止方法を定義する場合は、使用するエクスポートやトポロジーの固有の特性に基づいて定義することを強くお勧めします。


関連概念

62 ページの『接続グループ』

接続グループは、SCA モジュール内に見られる特定の動作パターンを表します。

関連情報

 [エクスポートとエクスポート・バインディング](#)

 [エクスポートの使用](#)

 [バインディング](#)

Failed Event Manager について

Failed Event Manager は、呼び出しの失敗を処理および再サブミットするための Web ベースのクライアントです。

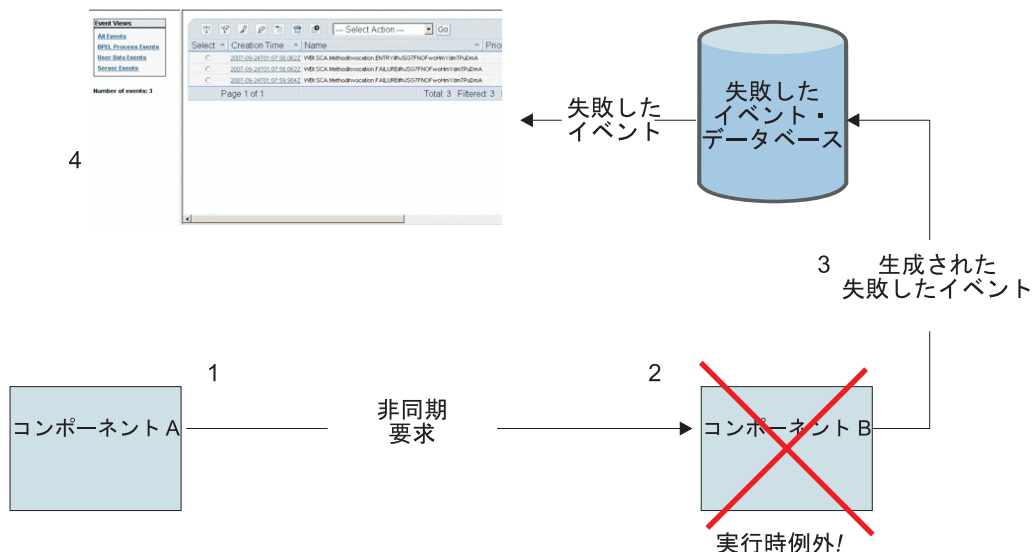
Failed Event Manager は、インテグレーション・アプリケーションで、管理コンソールで使用できます。

このアプリケーションによって、失敗イベントの数が表示され、多数の検索機能が提供されます。

失敗イベントを、日付、最後の成功イベントまたは失敗イベントなどさまざまな基準を使用して、例外テキストによって、またはこれらを組み合わせて、照会できます。

以下の図は、WebSphere Process Server 例外処理の高位での説明、および Failed Event Manager との関係を示しています。図の後に、番号付きでステップの説明があります。

失敗したイベント・マネージャー



1. コンポーネント A は非同期でコンポーネント B を呼び出しています。
2. コンポーネント B にランタイム例外が発生し、失敗イベント・レコードが生成されました。
3. 障害リカバリー・サービスはこの障害を捕捉し、失敗イベントのデータベースに格納します。
4. システム管理者は Failed Event Manager を開き、問題を調査します。

Failed Event Manager を使用した失敗イベントの再サブミットについて

Failed Event Manager で入力した検索基準と合致するイベントが表示されます。単一または複数の失敗イベントを再サブミットできます。再サブミット中、ペイロードを変更することもできます。例えば、何らかの不適切なデータを渡したことが原因で、障害が発生したとします。この場合、ペイロードは Failed Event Manager 内から更新され、再サブミットできます。メモリーに格納されているデータのみが更新されるため、データのオリジナル・ソースは訂正されません。再サブミットされたイベントが失敗した場合、Failed Event Manager で新規の失敗イベントとして表示されます。また、単一または複数イベントを削除することが可能で、障害発生時点以降データは無効になっているため、これは多くの場合適切なアクションとなります。

関連概念

95 ページの『ユース・ケース: 失敗イベントからのデータのリカバリー (recovering data from failed events)』

ユース・ケースは、リカバリー・シナリオでのコンテキストとして使用されます。このユース・ケースでのビジネスには、新規アカウントを作成する要求を受け取るアプリケーションがあります。

関連情報

 失敗イベントの管理

障害からのリカバリー

障害からのリカバリーのためには、障害時における標準のシステム処理と、障害の原因と思われる問題の分析方法を理解する必要があります。

リカバリー・プロセスの概要

リカバリー・プロセスは、分析とプロシーチャーの両方を含む一連のタスクで構成されます。

障害からリカバリーするときに行うステップの概要を以下に示します。

- 発生する障害の種類について学習します。詳しくは、『リカバリーのトリガー』を参照してください。
- システムの状態を評価します。詳しくは、『システムの状態の評価』を参照してください。
- 何が問題かについて仮説を立てます。
- データを収集して分析します。
- このインフォメーション・センターの他のトピックを参照して、問題を修正するための指示を探します。

関連概念

『リカバリーのトリガー』

ソリューション・リカバリーは、さまざまなトリガーの結果として必要になります。

89 ページの『システムの状態の評価』

異常状態が発生したときに取るべき最初の行動は、システム全体の動向を調べ、システムは実際にどの程度稼働しているのか、および何であれその状態を引き起こした外部要因によってどれほどのシステム要素が「サービス停止」となったのかを把握することです。

リカバリーのトリガー

ソリューション・リカバリーは、さまざまなトリガーの結果として必要になります。

ソリューション・リカバリーが必要なシチュエーション

ソリューション・リカバリーとは、操作を再開可能な状態にシステムを戻すプロセスのことです。これには、予期できない状況によってトリガーされる可能性があるシステム障害またはシステムの不安定性に対処するための一連のアクティビティーが含まれます。

以下の状況では、ソリューション・リカバリー・アクティビティーを実行する必要があります。

- ハードウェア障害

異常終了またはシステム・ダウンは、電源異常または壊滅的なハードウェアの故障が原因で発生します。これはシステム（すべてではないとしてもほとんどの JVM）が停止する原因となります。

壊滅的なハードウェア障害の場合、デプロイされたソリューションは、再始動時に一貫性を欠いた状態になることがあります。

ハードウェアの障害や環境の問題は、他の要因ほどではありませんが、予定外のダウン時間の原因ともなります。

自己最適化リソース調整による最新の LPAR 機能、(システムの過負荷を回避する) Capacity on Demand、および (単一機器の故障がシステム全体の故障となるのを防止する) システム内での冗長ハードウェアなどの機能を使用することにより、ハードウェア障害と環境の問題が発生する可能性を低くすることができます。

- **システム応答なし**

新規要求がシステム内に流れ込み続けていますが、表面上は、すべての処理が停止したように見えます。

- **システムで新規プロセス・インスタンスを開始できない**

システムは応答しており、データベースは正常に稼働しているように見えます。しかし、新規プロセス・インスタンスの作成に失敗します。

- **データベース、ネットワーク、またはインフラストラクチャーの障害**

基盤となるインフラストラクチャーの障害の場合、ソリューションでは、そのインフラストラクチャー障害が解決された後にビジネス・トランザクションの再開/再実行依頼を管理することが必要になる場合があります。

- **不十分なチューニングまたはキャパシティー・プランニングの不足**

システムは機能していても深刻な過負荷状態になっている。トランザクションのタイムアウトが報告され、計画容量からオーバーフローしている証拠がありません。

キャパシティー・プランニングまたはパフォーマンス・チューニングが不完全であると、このタイプのソリューションの不安定性の原因となる場合があります。

- **アプリケーション・モジュール開発での欠陥**

カスタム開発ソリューションの一部になっているモジュールには、バグが含まれる可能性があります。これらのバグにより、ソリューションが不安定になったり、サービスの実行に失敗したりします。

カスタム開発ソリューション内のバグは、以下を含む (ただし、それらに限定されることなく) さまざまなシチュエーションが原因となります。

- アプリケーション設計で計画されなかった、または予測されなかったビジネス・データ。
- アプリケーション設計での不完全なエラー処理方針。

詳細なエラー処理設計を行うことにより、ソリューションの不安定性を削減することができます。

• WebSphere ソフトウェアの欠陥

WebSphere 製品の欠陥により、イベントのバックログが処理またはクリアされません。

関連概念

92 ページの『リカバリー: 問題の分析』

すべての未計画システム・イベントの場合に、識別する時点で一組の基本リカバリー手順を活用できます。

87 ページの『リカバリー・プロセスの概要』

リカバリー・プロセスは、分析とプロシーチャーの両方を含む一連のタスクで構成されます。

関連資料


82 ページの『ピア・リカバリー』


ピア・リカバリーは、同じクラスター内の別のメンバーによって実行されるリカバリーであり、手動でも自動でも開始できます。ピア・リカバリー処理 (自動ピア・リカバリーまたは手動ピア・リカバリー) は、WebSphere の高可用性環境と緊密に連携しています。

82 ページの『ピア・リカバリー』

ピア・リカバリーは、同じクラスター内の別のメンバーによって実行されるリカバリーであり、手動でも自動でも開始できます。ピア・リカバリー処理 (自動ピア・リカバリーまたは手動ピア・リカバリー) は、WebSphere の高可用性環境と緊密に連携しています。

関連情報

 [ビジネス・プロセスのチューニング](#)

 [Exception handling in WebSphere Process Server and WebSphere Enterprise Service Bus](#)

 [エラー処理方針とソリューション・リカバリー](#)

システムの状態の評価

異常状態が発生したときに取るべき最初の行動は、システム全体の動向を調べ、システムは実際にどの程度稼働しているのか、および何であれその状態を引き起こした外部要因によってどれほどのシステム要素が「サービス停止」となったのかを把握することです。

事前定義した質問セットに答えて、障害の範囲を見積もります。以下に、適切な情報の収集に役立つように考案された事前定義の質問の例を示します。

1. このシステムは依然として稼働しているか?

システムが依然として稼働しているかどうかを判別します。多くの場合、システムは稼働可能でも、過負荷、不適切なチューニング、あるいはその両方の理由により、タスクを迅速に完了していないか、実際には失敗する処理を実行しようとしています。

これらの質問のそれぞれに対するリトマス試験は、デプロイされているソリューションの性質に固有のものです。

2. アプリケーションに組み込まれている特別なエラー処理サポート機能は何か?

数多くの自動化再試行およびさまざまなサポート・ロジックが存在する場合は、アプリケーション自体が、一部のエラーを隠して IT オペレーターに明示されないようにすることがあります。

このような状態は、リカバリー・チームが参照できるよう、周知および文書化される必要があります。

システムの状態を見積もるのに役立つタスクを以下に示します。

1. サーバーが少なくとも動作しているかどうかを調べます。

管理コンソールを介して、PID が表示されたか、または Deployment Manager から肯定のフィードバックを取得しましたか?

2. データベース内にロックが存在するかどうか、または異常なデータベース・トランザクションが存在するかどうかを調べます。

ほとんどのデータベースには、ロックを検出する機能があります。デプロイメント・トポロジーに応じて、複数のデータベースが存在する可能性があります。

- メッセージング・エンジン・データベース
- Business Process Container データベース
- WebSphere Process Server 共通データベース (失敗したイベントおよびリレーションシップ・データ)

3. メッセージング・システムの状況を調べます。

以下の場所にイベントまたはメッセージがあるかどうかを確認します。

- Business Process Choreographer の保留宛先と保存宛先
- 失敗したイベントの数
- ソリューション・モジュール宛先のメッセージの数

4. データベースが機能しているかどうかを調べます。

アンロックされたデータで、何らかの単純な SELECT オペレーションを妥当な時間内に実行することができますか?

5. データベース・ログにエラーがあるかどうかを調べます。

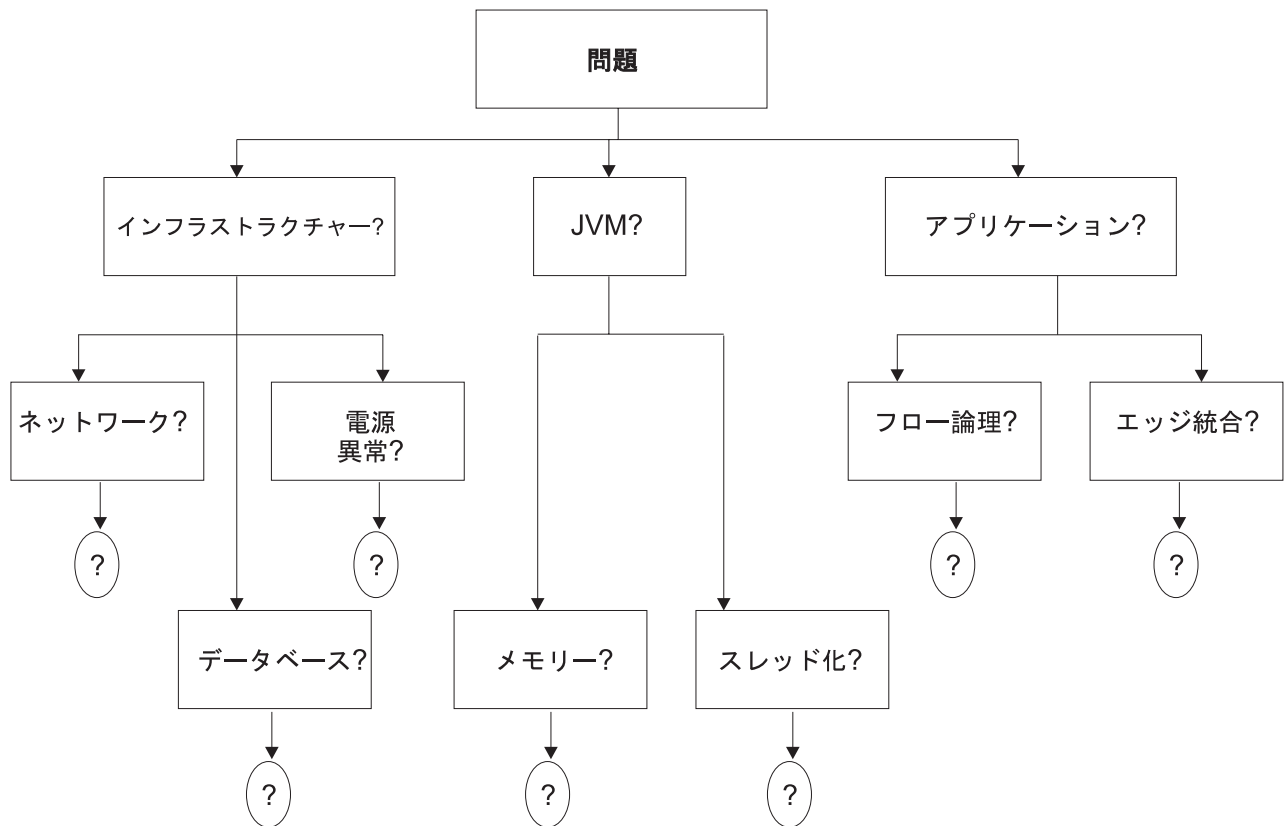
データベースが正常に稼働していない場合は、データベースをリカバリーする (これにより、少なくともロック解除され、単純な選択オペレーションを実行できるようになります) ことがシステムのリカバリーにとっても重要です。

メッセージング・システムが正常に稼働していない場合は、メッセージング・サブシステムをリカバリーして最低でも表示および管理できるようにすることが、システムのリカバリーにとっても重要です。

注: 「ボトムアップ」方式が必ずしも確実な方法とは限りません。しかし、リカバリーを正常に実行できる確率は、これらの基本的なアクティビティーに基づいています。

このような基本的な手順および正常性検査の類に含まれるアクティビティーから始めて、いくつかの固有のシチュエーションを探し出す必要があります。パターンについて説明され、仕様が定められ、水面下で進行している状況に関する洞察が与えられます。

このシチュエーション分析は、読み取り専用アクティビティーであることを認識してください。この分析により、適切なりカバー・アクションを判別するための重要な情報が提供されますが、検討中のシステムの状態は変更されません。システム障害について考えられる原因すべてを予測し、アクションの規定を設けることは不可能です。例えば、以下のデジジョン・ツリーについて考慮してみます。



計画外の停止イベントでは、調査するカテゴリーが広範囲に及びます。これらの広範なカテゴリーには、サブカテゴリーなども含まれています。各ノードとその後続ノードについて規定のアクションを定義することは、各調査の結果に依存します。このタイプのリレーションシップは文書形式で伝達することが難しいため、*IBM Guided Activity Assist* などのサポート・ツールを活用し、調査および決定プロセス全体を対話式に処理することをお勧めします。最上位から各子ノードに進むにつれ、適切なレベルのシチュエーション分析を実施することが重要になります。

関連概念

92 ページの『リカバリー: 問題の分析』

すべての未計画システム・イベントの場合に、識別する時点で一組の基本リカバリー手順を活用できます。

87 ページの『リカバリー・プロセスの概要』

リカバリー・プロセスは、分析とプロシーチャーの両方を含む一連のタスクで構成されます。

リカバリー：問題の分析

すべての未計画システム・イベントの場合に、識別する時点で一組の基本リカバリー手順を活用できます。

シチュエーション分析に対しては、十分に定義されたいくつかのステップがあります。以下にそのステップを示します。

1. 質問を定義する
2. 情報とリソースを収集する (観察)
3. 仮説を立てる
4. 実験を実行しデータを収集する
5. データを分析する
6. データを解釈し、新仮説の開始点となる結論を導く

実稼働環境のシナリオごとに、リカバリー・アクションを開始させる症状は異なります。

状況分析のガイドラインに従い、発生した症状に関連する修正アクションを実行することは重要です。

関連概念

89 ページの『システムの状態の評価』

異常状態が発生したときに取るべき最初の行動は、システム全体の動向を調べ、システムは実際にどの程度稼働しているのか、および何であれその状態を引き起こした外部要因によってどれほどのシステム要素が「サービス停止」となったのかを把握することです。

87 ページの『リカバリーのトリガー』

ソリューション・リカバリーは、さまざまなトリガーの結果として必要になります。

シチュエーション分析

シチュエーション分析とは、科学的手法を周期的に実行することで、リカバリー手順が開始されることになるさまざまなシチュエーションを考慮に入れることができます。

以下に、リカバリー手順が開始されることになるさまざまなシチュエーションのタイプを示します。

- 異常終了またはシステム停止

電源異常または壊滅的なハードウェアの障害により、システムが停止しました (ほとんどの JVM ではないとしてもすべて)。

- システム応答なし

新規要求がシステム内に流れ込み続けていますが、表面上は、すべての処理が停止したように見えます。

- システムは機能していても深刻な過負荷状態になっている

トランザクションのタイムアウトが報告され、計画容量からオーバーフローしている証拠があります。

- システムで新規プロセス・インスタンスを開始できない

システムは応答しており、データベースは正常に稼働しているように見えます。しかし、新規プロセス・インスタンスの作成に失敗します。

関連概念

120 ページの『メッセージング・サブシステムのリカバリーについて』
メッセージング・システムに問題が発生した場合、基礎となるメッセージング・サブシステムをリカバリーしなければならない可能性があります。

関連情報

 [トラブルシューティング](#)

リカバリー: ファースト・ステップ

管理者は一般的な手法のファースト・ステップ・チェックリストに従うことによって、ソリューション・リカバリー・プロセスを容易に行うことができます。

以下に、ソリューションのリカバリーを試みる際に、通常の状態では**実行すべきでない**アクションを示します。

注: 以下に示すアクションのいずれかを実行しなければならない特殊な状況もあります。ただし、必ず最初に WebSphere Process Server のサポート組織に問い合わせるから、これらのアクションのいずれかを開始するようにしてください。

- トランザクション・ログ・ファイルを削除しないこと

トランザクション (tranlog) ログ・ファイルには、データベースに書き込まれる重要なトランザクション・データが保管されています。これは、WebSphere Application Server が未完了トランザクションの管理に使用し、万が一サーバーが破損した場合は、リカバリーを試みるために使用する内部ファイルです。

- クラスタ・メンバーにローカルでトランザクション・ログを保管しないこと

トランザクション・ログは共用ドライブで保管してください。これは、リカバリー中のダウン時間を最小化するピア・リカバリーを可能にする唯一の方法です。

- 結果セットが追加リソース競合を形成するのに十分なほど大きい場合 (OutOfMemory) に、データベース操作を試みないこと
- 大きな結果セットを返す Business Process Choreographer Explorer 操作を実行するのを避けること
- 結果セットのサイズを考慮せずにプロセス・インスタンスで管理スクリプトを実行するのを避けること
- 実稼働環境でデータベースをドロップまたは再作成しないこと
- 標準のリカバリー手順の一部としてアプリケーションをアンインストールしないこと

アプリケーションは、IBM サポート組織の指示がある場合にのみアンインストールしてください。

- システムが過負荷状態の場合に有効にするトレースの量を多くしすぎないこと

トレースの量が多すぎると、システムのスループットが低下し、トランザクションがタイムアウトになる可能性があります。また、トレースの量が多すぎると、元の問題を解決する方法に関して洞察するというよりも、対処しなければならない問題が増えるということが多くなります。

正しいトレース仕様を定義するには、IBM サポートの支援をすぐに受けてください。

- 実動システムで新規のスクリプトやコマンドを実験的に使用したり試したりしないこと
- 実動サーバーを **開発モード** で実行しないこと

「**開発モードで実行**」オプションを有効にすると、アプリケーション・サーバーの起動時間が短くなる場合があります。これには、バイトコード検証を無効にし、JIT コンパイル・コストを減らすなどの、JVM の設定が含まれる場合があります。



以下に、リカバリーの場合に推奨されるアクションの説明を示します。

- 常に、構成ツリー、問題のアプリケーションの PI ファイル、および使用可能なログのスナップショットを取得してください。

ログは、構成に応じてそれ自体を上書きしている場合があります。セットを早めにかつ頻繁に捕捉することは、事後分析の重要なステップの 1 つです。この種のアクティビティで役立つ IBM Support Assistant については、「*IBM Support Assistant (ISA)*」のトピックを参照してください。

- 特に、データベース・トランザクションのログ・ファイルのサイズ、接続プール、およびロック・タイムアウトなどのデータベースの設定を常に把握しておいてください。

関連概念

IBM Support Assistant

IBM Support Assistant は、IBM ソフトウェア製品に関する疑問や問題を解決するのに役立つ無償のローカル・ソフトウェア保守容易性ワークベンチです。

67 ページの『エラー防止: 総合的なテスト』

総合的な機能およびシステム・テスト計画を実装することにより、リカバリーが必要になる問題の発生を防止できます。

78 ページの『システム・リカバリー』

システム・リカバリーとは、ソリューションのインフラストラクチャーへの悪影響を修正するために (手動または自動で) 実行されるオペレーションのことです。


79 ページの『アプリケーション・リカバリー』

アプリケーション・リカバリーとは、未完了ビジネス・トランザクションのリカバリーと解決を意味します。

関連情報

 トランザクション・ログ・ファイル (Transaction log file)

 アプリケーション・サーバー設定

 サービス・コンポーネント・イベントのロギングの構成

失敗したイベントのロケーション: データの行き先

すべてのリカバリー・アクティビティー (実動およびテスト) の場合に、イベントが累積されるソリューション内の場所の数は限定されています。

『エラー防止およびリカバリーの計画』に記載されているガイドラインおよび防止手段に従い、すべてのビジネス・イベントおよび関連するデータは、これらのいずれかの場所に安全に累積されます。

アーキテクチャーとアプリケーション開発のための優れた手法に従わない場合は、未完了イベントの割合が一貫性を欠いた状態になり、それらのイベントからのリカバリーを達成できなくなります。そのような状況では (想定されるのはテスト・サイクルの期間)、その後のリカバリー・アクティビティーが最後まで完全に実行されるようにするため、リカバリー後調査とクリーンアップを実施して問題を修正する必要があります。

以下のシナリオを正確に記述するためには、ユース・ケースのコンテキストで情報を提供することが重要です。

ユース・ケース: 失敗イベントからのデータのリカバリー (recovering data from failed events)

ユース・ケースは、リカバリー・シナリオでのコンテキストとして使用されます。このユース・ケースでのビジネスには、新規アカウントを作成する要求を受け取るアプリケーションがあります。

このソリューションは、モジュールのベスト・プラクティスによって推奨されている複数のモジュールで構成されています。

最初のモジュールは、要求を仲介し、処理をアカウント作成プロセスに委任します。以下の例では、別々のモジュールとしてソリューションを実装しています。要求は、SCA インポート/エクスポートを介してメディエーション・モジュール (AccountRouting) と処理モジュール (AccountCreation) の間で渡されます。2 つのモジュールの説明については、以下の画面取りを参照してください。

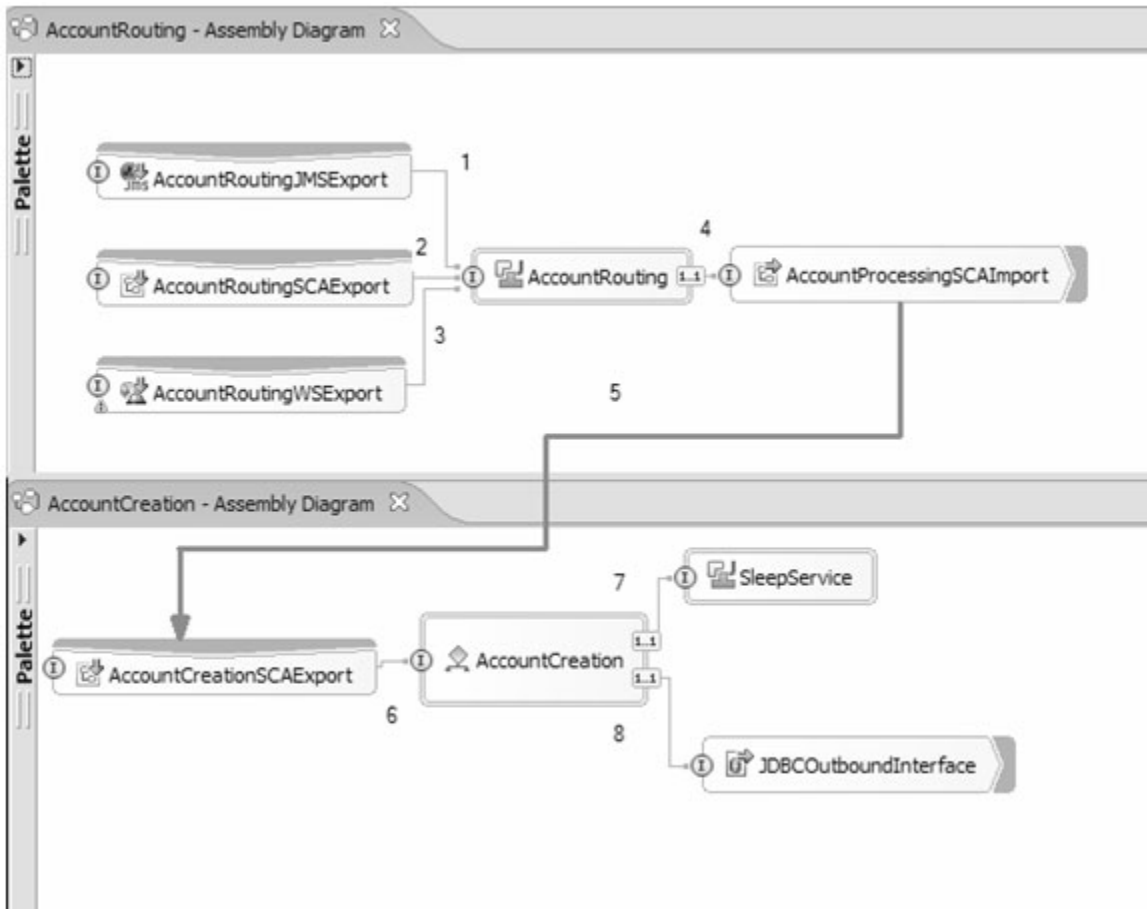


図5. アカウント・ルーティング・プロセスのアセンブリ・ダイアグラム

図5に示されるアセンブリ・ダイアグラムから始めて、障害が発生した可能性があるフロー内の場所を表示することができます。アセンブリ・ダイアグラム内の呼び出しポイントすべては、トランザクションを伝搬したり関係させたりすることができます。フロー内には、アプリケーションまたはシステムの障害の結果としてデータが集まる領域がいくつかあります。

一般に、トランザクション境界は、コンポーネントおよびインポート/エクスポート・バインディングとそれらの関連付けられた修飾子との間の対話 (同期および非同期) によって作成および管理されます。ビジネス・データは、トランザクション障害、デッドロック、またはロールバックのために、ほとんどの場合に特定のリカバリー・ロケーションに累積します。

WebSphere Application Server 内のトランザクション機能により、WebSphere Process Server は、サービス・プロバイダーによるトランザクションを参加させることができます。このような参加による対話は、インポートおよびエクスポートのバインデ

イングについて理解する上で特に重要です。特定のビジネス・ケース内でのインポートとエクスポートの使用方法を理解することは、リカバリーする必要があるイベントが累積している場所を判別するために重要です。

エラー処理方針では、対話パターン、使用するトランザクション、インポートおよびエクスポートの使用をアプリケーションの開発前に定義する必要があります。ソリューションの設計者は、アプリケーションが作成されるときに使用される、使用する設定およびガイドラインを識別する必要があります。例えば設計者は、同期呼び出しまたは非同期呼び出しをいつ使用するか、また BPEL 障害処理をいつ使用するかなどを理解する必要があります。設計者は、すべてのサービスがトランザクションに参加できるかどうか、また参加できないサービスについては、問題が発生したときに補正をどう処理するかについて知っておく必要があります。

また、96 ページの図 5 のアセンブリー・ダイアグラムに示されるアプリケーションでは、接続グループとモジュール開発のベスト・プラクティスを活用しています。このパターンを活用することにより、AccountRouting モジュールを停止して、新規イベントのインバウンド・フローを停止することができるようになっています。

以下のセクションでは、障害およびリカバリーにおけるビジネス・データのロケーションを説明します。

Business Flow Manager または Human Task Manager

このビジネス・ケースでは、AccountCreation プロセスで BPEL プロセスを活用します。

リカバリーに関しては、BPEL およびヒューマン・タスク管理について次のような質問について考えてみる必要があります。

1. 実行中のプロセスのタイプはどれか (短期実行または長期実行、ビジネス・ステート・マシン、ヒューマン・タスク)?

短期実行プロセスはマイクロフローとして知られています。

2. プロセスは正しく作成され、障害処理を使用してデータ保全性を向上させているか?
3. トランザクション境界の予測と制御のために、呼び出しパターンと作業単位プロパティはどのように構成されているか?

これらの質問に対する答えを知ることは、以下の画面取りで強調表示されているアセンブリー・ダイアグラムの呼び出し 7 および 8 のリカバリー方針に影響を与えます。

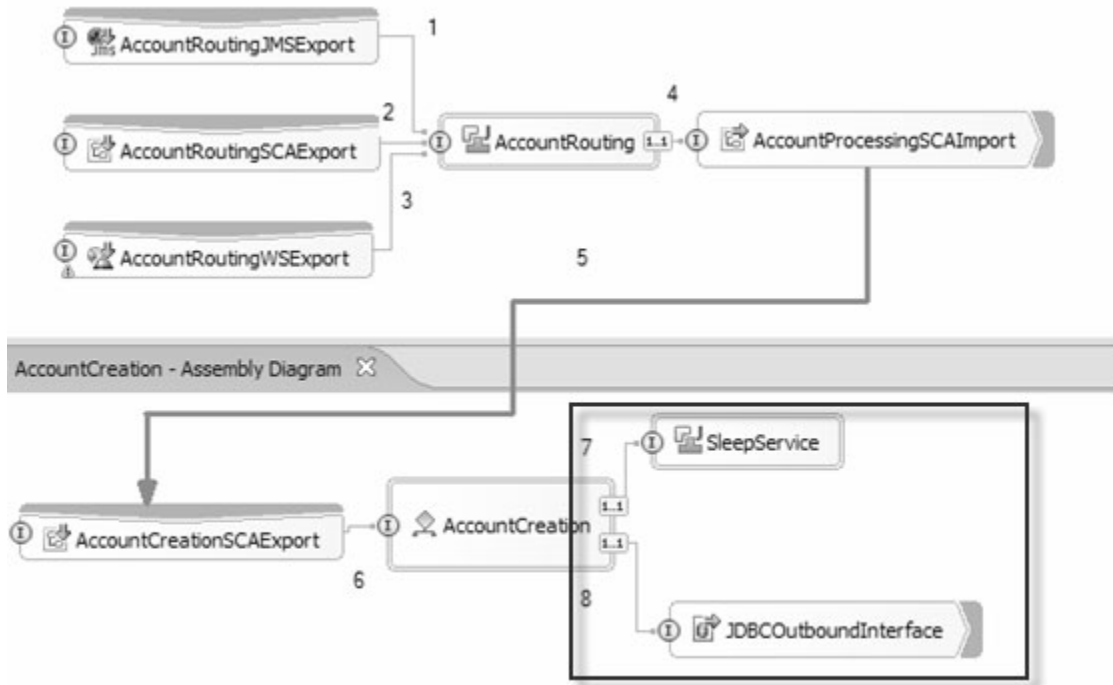


図6. アカウント・ルーティングのアセンブリ・ダイアグラム - 呼び出し 7 および 8

長期実行 BPEL プロセスやビジネス・ステート・マシンなどのステートフル・コンポーネントには、プロセス・アクティビティ・トランザクションと状態変更がデータベースに対してコミットされる、数多くのデータベース・トランザクションが関係しています。作業は、データベースを更新し、次に実行する内容を記述したメッセージを内部キューに格納することによって進行します。マクロ・フロー・トランザクションについての詳しい情報は、インフォメーション・センターの『長期実行プロセスのトランザクションの動作 (Transactional behavior of long-running processes)』で提供されています。

Business Flow Manager の内部でメッセージの処理に問題が発生すると、それらのメッセージは、保存キューに移動されます。システムは、メッセージの処理を続行しようとします。後続メッセージが正常に処理されると、保存キューのメッセージが再実行依頼されて処理されます。同じメッセージが保存キューに 5 回格納された場合、そのメッセージは保留キューに移されます。使用される内部キューや、これらのキューでの再試行アルゴリズムなどの情報については、インフォメーション・センターの『インフラストラクチャー障害からの回復』というトピックで詳しく説明されています。

メッセージ数の表示とメッセージの再生についての追加情報は、『保存キュー/保留キューからのメッセージの再生 (Replaying Messages from the Retention Queue / Hold Queue)』で提供されています。

Failed Event Manager

Failed Event Manager (FEM) は、ほとんどのコンポーネント・タイプ間で非同期に行われるイベントまたはサービス呼び出し要求を再生するために使用されます。

失敗したイベントは、AccountRouting コンポーネントが SCA インポート・バインディング AccountCreationSCAImport を非同期で呼び出し、ServiceRuntimeException が戻された場合に作成されます。

重要な点として、BPEL がサービス対話においてクライアントになっている場合は、そのほとんどにおいて失敗したイベントが生成されません。つまり、(98 ページの図 6 に示される) 7 および 8 の呼び出しでは、通常は失敗したイベントになりません。BPEL には、障害をモデル化するために、障害ハンドラーやその他の方法が用意されています。このため、「JDBCOutboundInterface」を呼び出す ServiceRuntimeException (SRE) 障害が発生しても、SRE は BPEL に戻されて処理されます。プロジェクトのエラー処理方針では、BPEL において一貫した方法で実行時例外を処理する方法を定義してください。

ただし、インフラストラクチャー障害のためにプロセス・インスタンスにメッセージを配信できない場合には、BPEL クライアントへの非同期応答メッセージに対して、失敗したイベントが作成されることを念頭においてください。

以下の図は、失敗したイベント・マネージャー・コンポーネントの動作方法を示しています。図では、番号付けされた各ステップに関連する処理の説明を示しています。

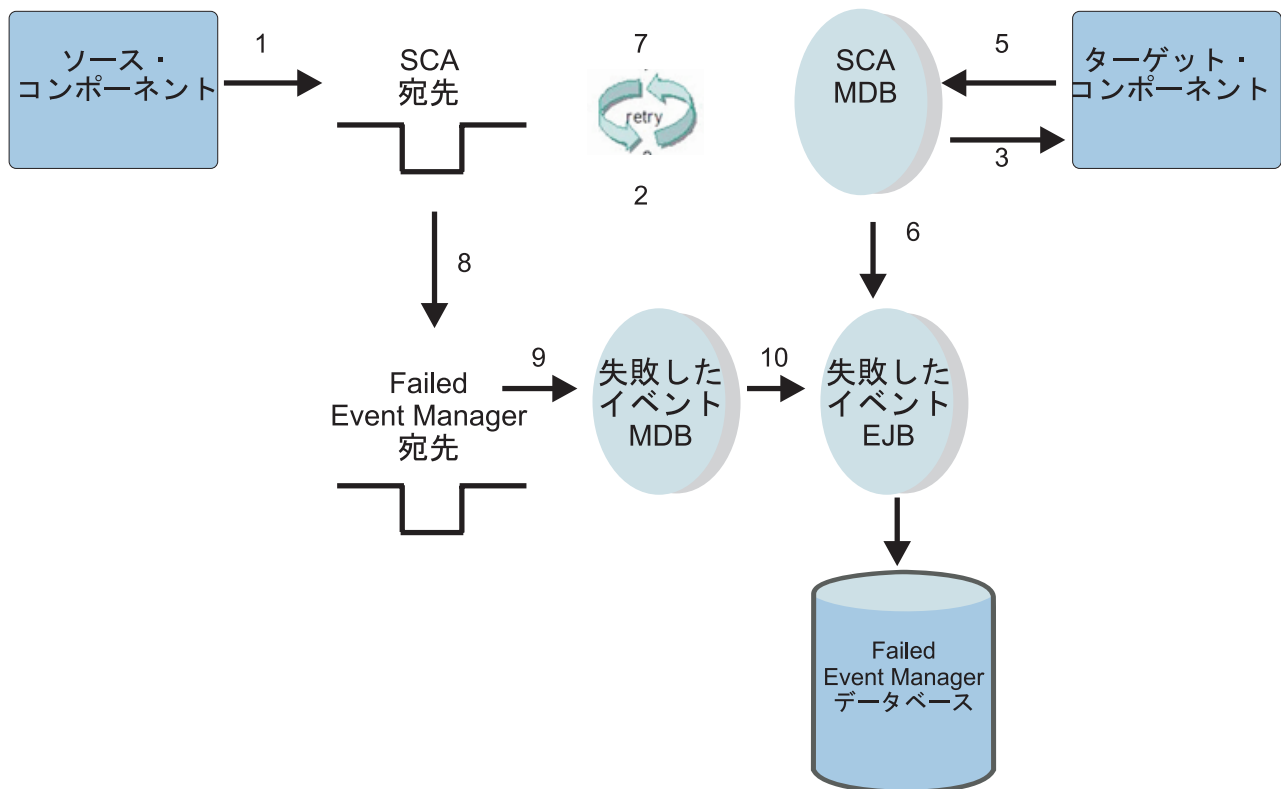


図 7. 失敗したイベント・マネージャーの処理

失敗したイベント・マネージャーの処理

1. ソース・コンポーネントが非同期呼び出しパターンを使用して呼び出しを行う
2. SCA MDB が SCA 宛先からメッセージを選出する

3. SCA MDB が正しいターゲット・コンポーネントに対して呼び出しを行う
4. ターゲット・コンポーネントが `ServiceRuntimeException` をスローする
5. SCA MDB トランザクションが SCA 宛先にロールバックする
6. 例外情報が、未確認 という状況を指定されて、失敗したイベント・マネージャー・データベースに保管される
7. SIBus が呼び出しの再試行を n 回行う

再試行制限のデフォルト値は 5 (最初の呼び出しの 1 回および再試行 4 回) です。このデフォルト値は、管理コンソールで変更できます。例えば、M という SCA モジュールの場合は、「バス」 → 「SCA.SYSTEM.<CELL>.BUS」 → 「宛先」 → 「sca/M」にナビゲートし、「最大デリバリー失敗数」フィールドで値を変更します。

8. 再試行回数が指定された制限値に達すると、メッセージが FEM 宛先に移動される
9. 失敗したイベント・マネージャー・データベースがメッセージを選出する
10. 失敗したイベント・マネージャー・データベースが、データベース内の失敗したイベントを更新し、状況が失敗 に設定される

「失敗したイベント」はいつ作成されるか?

すでに述べたように、失敗したイベントは、同期呼び出しで作成されるのでも、標準的に両方向ビジネス・プロセス対話で作成されるのでもありません。

失敗したイベントは、通常、クライアントが非同期呼び出しパターンを使用し、サービス・プロバイダーが `ServiceRuntimeException` をスローしたときに作成されません。

すべてのイベントを同期をとって同じトランザクションで実行すると、データはどこにも収集されません。代わりに、呼び出し元のクライアントにすべてロールバックされます。データは、コミットが発生した場所に収集されます。すべての呼び出しが同期的に実行される一方で複数のコミットが実行された場合は、コミットの問題が発生します。

一般的に、複数のトランザクションが必要な場合は、非同期処理呼び出しまたは長期実行 BPEL を使用する必要があります。このため、各 ASYNC 呼び出しはデータを収集する機会となります。長期実行 BPEL プロセスは、データ収集のポイントです。

表 4. 呼び出しパターンと失敗したイベントの作成との関連: サービス・ビジネス例外

呼び出しパターン	失敗したイベントが作成されるかどうか (Y/N)?	注意
同期	いいえ	失敗したイベントは、サービス・ビジネス例外の場合、または同期パターンを使用する場合には作成されません

表 4. 呼び出しパターンと失敗したイベントの作成との関連: サービス・ビジネス例外 (続き)

呼び出しパターン	失敗したイベントが作成されるかどうか (Y/N)?	注意
非同期 - 片方向	いいえ	定義により、片方向呼び出しでは障害を宣言できません。つまり、ServiceBusinessException をスローするのは不可能です。
非同期 - 据え置き応答	いいえ	サービス・ビジネス例外では失敗したイベントが作成されません
非同期 - コールバック	いいえ	サービス・ビジネス例外では失敗したイベントが作成されません

表 5. 呼び出しパターンと失敗したイベントの作成との関連: サービス・ランタイム例外

呼び出しパターン	失敗したイベントが作成されるかどうか (Y/N)?	注意
同期	いいえ	失敗したイベントは、サービス・ランタイム例外の場合、または同期パターンを使用する場合には作成されません
非同期 - 片方向	はい	
非同期 - 据え置き応答	はい	
非同期 - コールバック	はい	
BPEL - 双方向	いいえ	ソース・コンポーネントがビジネス・プロセスの場合、失敗したイベントは作成されません。 注: 非同期呼び出しでは、応答を BPEL に返すことができない場合に、失敗したイベントが作成されます。
BPEL - 片方向	はい	

追加情報については、インフォメーション・センターの『失敗イベントの管理』というトピックを参照してください。

失敗したイベントの表示および再実行依頼に関する追加情報については、セクション『失敗したイベントの再サブミット』を参照してください。

サービス統合バスの宛先

処理待ちのメッセージは、少数のサービス統合バス (SIBus) 宛先に累積されます。これらの宛先の大部分は「システム」宛先です。これらの宛先内のメッセージは、通常、次の 3 つのタイプのメッセージで混成されています。

- 処理の非同期要求
- 要求に対する非同期応答
- 非直列化または関数セクター解決に失敗した非同期メッセージ

注: 非同期応答は、有効なビジネス・オブジェクトであったり、要求の結果として返された障害であったりします。

SCA モジュール宛先

再び、ビジネス・ケースに戻ります。

ソリューションには、次の 2 つの「SCA モジュール」宛先があります。

- sca/AccountRouting
- sca/AccountCreation

これらの宛先は、モジュールがアプリケーション・サーバーまたはクラスターにデプロイされるときに作成されます。

これらの宛先にメッセージが累積されることはまれです。これらの場所にメッセージが累積されるということは、パフォーマンス上の問題やアプリケーションの問題が発生している可能性が非常に高いことを示します。すぐに調査してください。メッセージのバックアップによってシステムが停止したり、リサイクル時間が延長されたりすることになるので、(選択した IT モニター・ソリューションによって) モジュールの宛先の深さをモニターすることは重要です。

これらは、生成される名前が「sca/」付きのモジュール名と同じになるため、「SCA モジュール」宛先と呼びます。これらの宛先は、SCA 非同期呼び出し (要求と応答のプロローカリング) の機能において中心的な役割を果たします。SCA.SYSTEM バスへのアプリケーション・インストール中に生成される追加宛先の数はさまざまですが、ここでの説明では「SCA モジュール」宛先の重要性を扱います。

システム統合バス再試行

上述のとおり、FEM には SCA message driven bean (MDB) による再試行メカニズムが組み込まれています。この再試行動作は、モジュール宛先の「最大デリバリー失敗数」属性を変更することによって制御できます。

注: 通常は、この再試行機能を調整する必要はありません。ここでは、すべての状況について説明しておきます。

ここでのビジネス・ケースを例にとると、非同期通信をサポートするため、SCA により多くの SI バス宛先が作成されています。

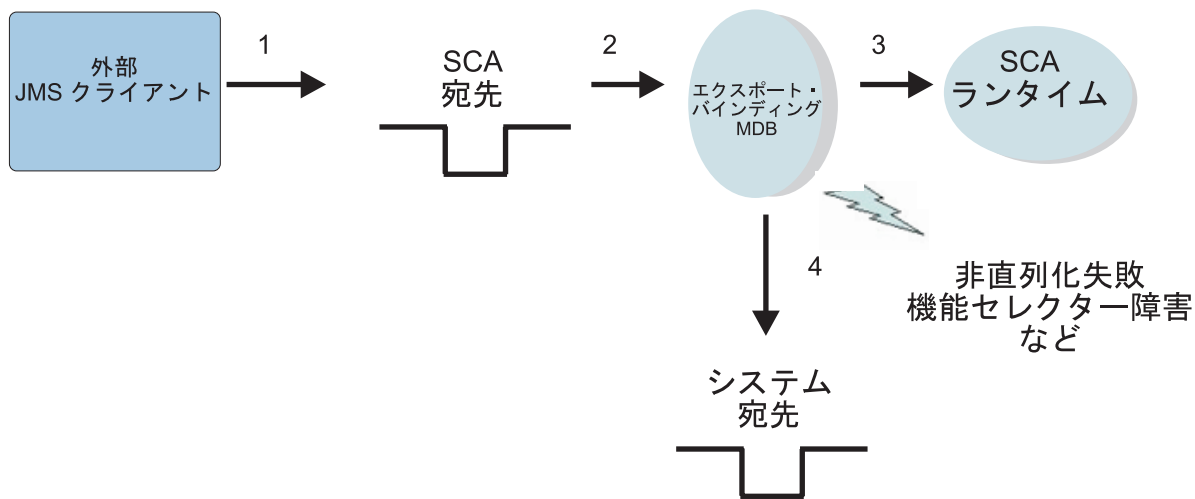
すでに学んだとおり、これらの宛先の 1 つは「sca/AccountRouting」と呼ばれます。非同期サービス呼び出しの `ServiceRuntimeException` 時に実行される再試行回数は、管理コンソールを介して「最大デリバリー失敗数」プロパティの値を変更することによって調整できます。ただし、BPEL プロセスによるモジュールで、2 未満の値を設定することはできません。`ServiceRuntimeExceptions` を BPEL に戻して処理するには、2 回目のデリバリーが必要になります。

システム例外宛先

失敗したイベント・マネージャーは、管理障害を探すために調べることができる場所の 1 つです。JMS または EIS ベースのインポートおよびエクスポートを処理する場合は、別の重要な場所について考慮する必要があります。

SCA.Application バスの宛先は、失敗したメッセージを、そのバスの SIB システム例外宛先に経路指定するように構成されます。したがって、JMS エクスポートが SCA.Application バスからメッセージを選出し、ロールバック・シチュエーションに入ると、失敗したメッセージは、WBI リカバリー例外宛先ではなく、その SIB システムの例外宛先に経路指定されます。このシナリオは、SCA.Application バスでメッセージの非直列化に失敗しても失敗したイベントが生成されない、上述の失敗したイベントの説明とは異なります。ソリューション内のすべてのバスには、システム例外宛先があります。これらの宛先は、ほとんど MQ インフラストラクチャーに共通の「送達不能キュー」のようにして、モニターおよび管理する必要があります。

以下のシナリオについて考慮してください。



外部の JMS クライアントは、JMS エクスポートによって公開されるインバウンド・キューにメッセージを格納します。JMS エクスポート・バインディング MDB は、処理するメッセージを選出します。ここから、以下の 2 つのいずれかの状態になります。

1. JMS エクスポートは、メッセージを正常に解析し、処理のためにメッセージが SCA ランタイムに送信されるときに呼び出されるインターフェース上の操作を判別します。
2. JMS エクスポートは、有効なビジネス・オブジェクトとしてメッセージ本文を認識することに失敗するか、または JMS エクスポート・バインディングがメッセージ本文を非直列化しても、インターフェースで呼び出す適切な操作を判別できません。この時点でメッセージは、バスのシステム例外宛先に格納されます。

この種の失敗は、AccountRoutingJMSExport (1) から要求を受け取ろうとするときに発生する可能性があります。このエクスポートは JMS エクスポートの 1 つで、イベントが SCA.Application.Bus のシステム例外宛先に累積する可能性があります。選択した IT モニター・ソリューションを使用して、この宛先の深さを観察してください。

失敗したイベント・マネージャーと SIB の宛先

WebSphere Process Server では、例外宛先が WebSphere Process Server の例外宛先キューに設定されます。このキューは、次の命名規則に従います。

ノード名: WPSNode
 サーバー名: server1
 リカバリー例外宛先: WBI.FailedEvent.WPSNode.server1

一般に、SCA.System バスで作成されるすべての宛先は、失敗したメッセージがリカバリー例外宛先に経路指定されるように構成されます。

システム障害が発生すると、この例外宛先に失敗したメッセージが捕捉されることに加え、WebSphere Process Server リカバリー機能により、システム・エラーを表す失敗したイベントが生成され、この文書のセクション『失敗したイベント・マネージャー』で説明されるリカバリー・データベースに保管されます。

要約

要約すると、WebSphere Process Server により、基盤となる WebSphere Application Server プラットフォーム以上の管理機能を提供できます。これらの機能を理解して使用するため、『エラー防止とリカバリーの計画』のセクション『エラー防止の計画』で説明されているガイダンスに従って、適切な手段を作成してください。

表 6. 失敗の管理に役立つ管理機能

管理機能	WebSphere Process Server にバンドルされているか (Y/N)?	要約
Business Process Choreographer Explorer	はい	読み取り/書き込み/編集/削除アクセス。これは、ビジネス・プロセスとヒューマン・タスクを管理するための中心的な場所です。
失敗したイベント・マネージャー	はい	読み取り/編集/削除アクセス。これは、サービス・ランタイム例外と他の形態のインフラストラクチャーの障害を管理するための中心的な場所です。
Service Integration Bus Browser	はい	読み取り/削除。管理コンソールの「Service Integration Bus Browser」を使用して、サービス統合バスで日次の運用タスクを表示して実行します。

注: これらのツールによって同時に管理可能なイベントまたはレコードの数は、メモリー割り振り、結果セットと DB チューニング、および接続タイムアウトなどの外部要因によって異なります。テストを実行し、例外 (OOM、TransactionTimeout) を回避する適切なしきい値を設定してください。

関連概念

65 ページの『サービス・ビジネス例外の処理』

サービス・ビジネス例外は、アプリケーションまたはサービスによって予期される既知および宣言済みの例外を表します。

66 ページの『サービス・ランタイム例外の処理』

サービス・ランタイム例外は、未宣言の例外です。一般にこの例外は、アプリケーションによって予期されていないエラー状態を表します。

80 ページの『トランザクションのプロパティとソリューション・リカバリー』

WebSphere Process Server は WebSphere Application Server をベースとしているため、ビジネス・トランザクションを実行するトランザクション・モデル をサポートしています。

62 ページの『接続グループ』

接続グループは、SCA モジュール内に見られる特定の動作パターンを表します。







85 ページの『Failed Event Manager について』

Failed Event Manager は、呼び出しの失敗を処理および再サブミットするための Web ベースのクライアントです。

113 ページの『保存キューと保留キュー』

メッセージの処理中に問題が発生すると、そのメッセージは保存キューまたは保留キューに移されます。

関連情報

-  [ビジネス・プロセスのトランザクションの振る舞い](#)
-  [失敗イベントの再サブミット](#)
-  [管理コンソールによる障害メッセージの照会と再生](#)
-  [Service Integration Bus Browser](#)
-  [失敗イベントの管理](#)
-  [ビジネス・プロセスでのフォールトの処理](#)

デプロイメント環境の再始動

リカバリー・プロセスの 1 ステップとして、デプロイメント環境の再始動が必要となる場合があります。

デプロイメント環境の再始動について

デプロイメント環境を再始動する手順は、トポロジーによって異なります。トポロジーはシステム構成パターンに基づいており、各パターンは特定のビジネス要件を満たすように設計されています。

WebSphere Process Server は、事前定義されたデプロイメント環境構成パターン一式をサポートしています。これらのパターンのいずれもが要件を満たさない場合は、ユーザー独自のカスタマイズしたデプロイメント環境を計画して作成することもできます。

どのデプロイメント環境構成パターンにも、JVM プロセスとして稼働する複数のサーバーが存在します。通常、次の 4 種類のサーバーがあります。

- メッセージング・サーバー

サービス統合バス (SIB) のメッセージング・インフラストラクチャーを提供するサーバーです。

- WebSphere ESB Servers

メディアエーション・モジュールだけをホストして実行することができるプロファイルを持つサーバーです。

- WebSphere Process Servers

すべてのモジュール・タイプをホストして実行することができるプロファイルを持つサーバーです。このプロファイルは、Business Process Choreographer コンポーネントをホストします。

- サポート・サーバー

サポートの提供と Common Event Infrastructure (CEI) などのサービスのモニターを行うサーバーです。

最も費用対効果の高い方法でビジネス要件に対応できるようにするため、デプロイメントのパターンは、すべての機能コンポーネントの分類方法と編成方法によって異なります。より高度で可用性の高い環境では、複数の物理リソースに分散したクラスター内にサーバーが存在します。

リカバリー操作の一部としてサーバーを再始動する一般的な手順

通常のサーバー始動の場合、メッセージング・サーバー、サポート・サーバー、WebSphere Process Server サーバーの順に始動します。各アプリケーション・アーキテクチャーのアプリケーション・コンポーネント間には、注意が必要な固有の依存関係が存在する場合があります。

サーバーをシャットダウンする場合、基本的には始動手順の逆になります。つまり、最初にアプリケーション・サーバー・クラスターをシャットダウンし、最後にメッセージング・インフラストラクチャーが静止して未完了トランザクションを処理した後に、メッセージング・インフラストラクチャーをシャットダウンします。

関連タスク

52 ページの『デプロイメント環境パターンの選択』

IBM 提供のいずれかのパターンを選択するか、独自のカスタム・デプロイメント環境を作成することによって、デプロイメント環境を構成できます。このトピックでは、IBM 提供の各パターンでサポートされる機能をリストします。

17 ページの『第 4 章 デプロイメント環境の計画』

デプロイメント環境のセットアップには、物理サーバーの数から選択するパターンのタイプまで、あらゆる事柄に影響を与える多くの決定が関係しています。それぞれの決定はデプロイメント環境をセットアップする方法に影響を与えます。

関連情報



WebSphere Process Server and WebSphere Enterprise Service Bus deployment patterns: Selecting your deployment pattern

サービス統合バスの表示

サービス統合バスを表示させるには、管理コンソールの Service Integration Bus Browser を使用します。

始める前に

SCA システム・バスがどのように使用されるかを理解する必要があります。

このタスクについて

Service Integration Bus Browser は、サービス統合バスにおける日常の操作タスクの参照および実行を行うための単一ロケーションを提供します。

サービス統合バスの表示は、メッセージが SCA モジュール宛先に蓄積されているかどうかを判別するのに便利です。

SCA モジュール宛先にメッセージが累積している場合、パフォーマンス上の問題やアプリケーションの問題が存在する可能性が高いということになります。

メッセージを定期的に表示し、長期間にわたってロックされているメッセージがないか調べることをお勧めします。こうしたメッセージが残っている場合、「未確定トランザクション」が存在する可能性があります。

手順

1. 管理コンソールで「サービス統合」を展開します。
2. 「バス」を選択します。



3. このサービスの適切なメッセージング・バスを選択します。次の例では、SCA.System.cleanup1cell101.bus という名前のメッセージング・バスが強調表示されています。この cleanup1cell101 は、セルの名前です。

Preferences

New Delete

Select	Name	Description	Security
<input type="checkbox"/>	BPC.cleanup1Cell01.Bus	Messaging bus for Process Choreographer	Enabled
<input type="checkbox"/>	CommonEventInfrastructure Bus	CommonEventInfrastructure Bus	Enabled
<input type="checkbox"/>	SCA.APPLICATION.cleanup1Cell01.Bus	Messaging bus for Service	Enabled
<input type="checkbox"/>	SCA.SYSTEM.cleanup1Cell01.Bus	Messaging bus for Service	Enabled

Total 4

4. 「宛先」を選択します。

Configuration Local Topology

General Properties

Name
SCA.SYSTEM.cleanup1Cell01.Bus

UUID
88D12C35D81C8E5C

Description
Messaging bus for Service

Topology

- Bus members
- Messaging engines
- Foreign buses

Destination resources

- Destinations
- Mediations

5. 関連する情報を確認します。ここでは、sca/XYZ という宛先を確認する必要があります (XYZ はモジュール名)。例えば、AccountRouting と AccountCreation というモジュールの場合は、以下の宛先を確認します。

<input type="checkbox"/>	sca/AccountCreation
<input type="checkbox"/>	sca/AccountCreation/component/AccountCreation
<input type="checkbox"/>	sca/AccountCreation/component/SleepService
<input type="checkbox"/>	sca/AccountCreation/export/AccountCreationSCAExport
<input type="checkbox"/>	sca/AccountCreation/exportlink/AccountCreationSCAExport
<input type="checkbox"/>	sca/AccountCreation/import/JDBCOutboundInterface
<input type="checkbox"/>	sca/AccountCreation/import/sca/dynamic/import/scaimport
<input type="checkbox"/>	sca/AccountCreation/import/sca/dynamic/import/vsimport
<input type="checkbox"/>	sca/AccountRouting

- 表示したい宛先のリンク・テキストを選択します。

この操作により、表示したい宛先の一般プロパティ・ページに移動します。

- 宛先の一般プロパティ・ページで「キュー・ポイント」を選択します。

Configuration

General Properties

Identifier:

UUID:

Type:

Description:

Message points

[Queue points](#)

[Mediation points](#)

Additional Properties

[Context properties](#)

[Mediation execution points](#)

- 「キュー・ポイント」ページで、メッセージ・ポイントのリンクを選択します。

Buses > SCA.SYSTEM.cleanup1Cell01.Bus > Destinations > sca/AccountCreation > Queue points

The message point for a queue, for point-to-point messaging.

Preferences

Identifier ⇅

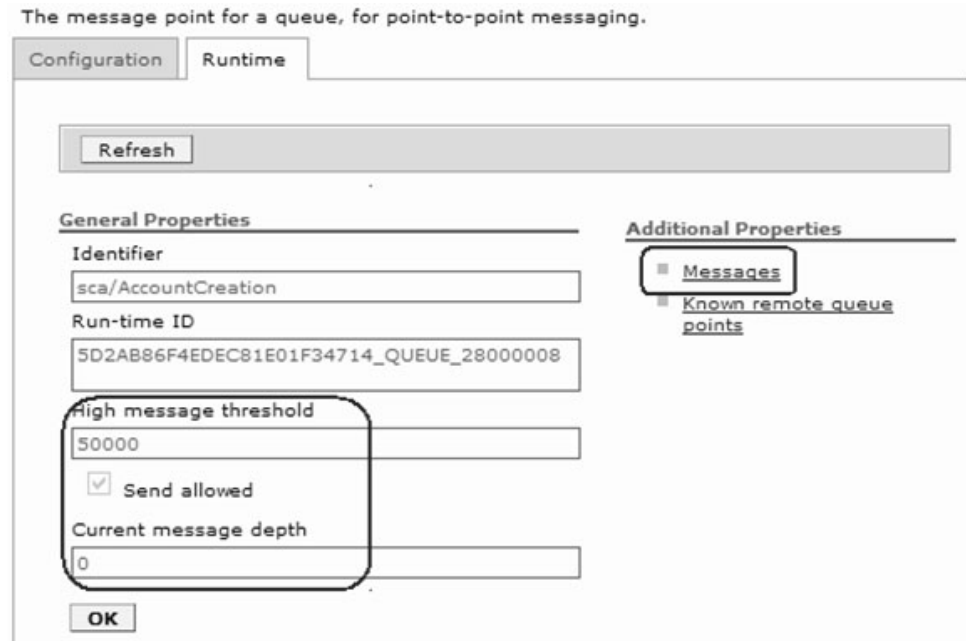
[sca/AccountCreation@default.Messaging.000-SCA.SYSTEM.cleanup1Cell01.Bus](#)

Total 1

- 「ランタイム」タブを選択します。

この画面から、現行メッセージの「深さ」としきい値がわかります。

「メッセージ」リンクを選択すると、メッセージの内容を実際に表示することができます。



適切な IT モニター・ツールを使用して、これらの宛先のアラートしきい値を設定することをお勧めします。このしきい値は、アプリケーションのパフォーマンス・テスト・フェーズ中に設定されます。

SCA L3 チームによる明確な指示がない限り、実動システムのメッセージは削除しないでください。

関連概念

SCA システム・バス

SCA システム・バス とは、Service Component Architecture (SCA) モジュールのキュー宛先をホストするために使用するサービス統合バスのことです。メディアエーション・モジュールをサポートする SCA ランタイムは、システム・バス上のキュー宛先をインフラストラクチャーとして使用して、コンポーネントとモジュール間の非同期対話をサポートします。

WebSphere Process Server 用のサービス統合バス

サービス統合バスとは、同期および非同期メッセージングによってサービス統合をサポートする、管理された通信メカニズムです。バスは、バス・リソースを管理する相互接続メッセージング・エンジンで構成されます。サービス統合バスは、WebSphere Process Server の基盤となる WebSphere Application Server テクノロジーの 1 つです。

関連タスク

116 ページの『未確定トランザクションの解決』

ノードの削除によってメッセージング・エンジンが破壊されるなどの例外的な状況が原因で、トランザクションが未確定状態のままになることがあります。

関連情報

 SCA リソース

13 ページの『サーバーおよびクラスターでの Service Component Architecture サポートに関する考慮事項』

サーバーおよびクラスターでは、Service Component Architecture (SCA) アプリケーション、アプリケーション宛先、またはその両方をサポートできます。

 doc/cadm_sibbrowser.dita

javacore の収集

IBM JDK の javacore や、IBM 以外の JDK のスレッド・ダンプを収集する場合、いくつかの方法があります。

javacore の収集

javacore ダンプ (スレッド・ダンプとも呼ばれる) は、アプリケーション・サーバーが作成する主要な問題判別文書の 1 つです。

1. 以下のように、wsadmin を使用してプロファイル・ディレクトリーに javacore を生成します。

a. Windows の場合:

```
<PROFILE_DIR>%bin%wsadmin.bat [-host host_name] [-port port_number]
[-user userid -password password] -c
"$AdminControl invoke [$AdminControl queryNames WebSphere:name=JVM,process=server1,*]
dumpThreads"
```

b. Unix の場合 (IBM JDKs):

```
<PROFILE_DIR>>/bin/wsadmin.sh[-host host_name]
[-port port_number] [-user userid -password password] -c
"%AdminControl invoke [%AdminControl queryNames WebSphere:name=JVM,process=server1,*]
dumpThreads"
```

注: AdminControl queryNames コマンドを囲む中括弧 [] は、実際のコマンドの一部です。ホスト、ポート、ユーザーを囲む中括弧のようにオプション・パラメーターを示す括弧ではありません。プロセス名 server1 は、ご使用の構成に合うように変更する必要があります。

2. 以下のように、サーバー・プロセスに信号を送信することができます。

a. Windows の場合:

起動スクリプトを使用してサーバー・プロセスを開始し、プロセスに信号を渡す必要があります。そのためには、サーバーを始動する前に特別なセットアップを行う必要があります。

1) <PROFILE_DIR>%bin%startServer.bat server1 -script SERVER1.bat

2) b. SERVER1.bat

サーバー・プロセスがコマンド・ウィンドウで開始されます。通常は中間 JVM プロセスによってサーバー・プロセスが開始されますが、ここでは

この中間 JVM プロセスは使用されないため、ログを調べて、サーバーが正常に始動したことを確認する必要があります。

3) <CTRL><BREAK>

サーバー・プロセスが稼働しているコマンド・ウィンドウで <CTRL><BREAK> を実行します。javacore が生成されます。

b. UNIX (すべての JDK) の場合: kill -3 <pid>

この <pid> は、WebSphere Process Server のプロセス ID です。IBM JDK の場合、javacore は <PROFILE_DIR> ディレクトリーに生成されます。

IBM 以外の JDK の場合、スレッド・ダンプは native_stdout.log に書き込まれます。

3. Windows コア・ファイルは、jvmdump を使用してダンプすることもできます。

この方法の場合、サーバーを始動する前に特別なセットアップを行う必要はありません。ただし、JVM チームから専用の実行可能プログラムを入手する必要があります。このプログラム (jvmdump.exe) が必要な場合は、jvmcookbook@uk.ibm.com までご連絡ください。このプログラムを使用すると、JVM 内部で実行されているネイティブ・コードに関する追加情報を取得することができます。ダンプの形式は、IBM javacore とは異なります。

- jvmdump.exe <PID>
- <WAS_HOME>%java%jre%bin%jextract.exe <core.name.dmp>
- <WAS_HOME>%java%jre%bin%jdumpview.exe
 - set dump <core.name.dmp>.zip
 - display thread

ダンプ中に実行されている現行スレッドを表示します。

- c. display thread *

ダンプからすべてのスレッドを表示します。

jdumpview ユーティリティーについて詳しくは、IBM Developer Kit and Runtime Environment, Java Technology Edition, バージョン 5.0 の「Diagnostics Guide」を参照してください。

関連情報

 ハング検出ポリシーの構成

サーバーおよびリカバリー・モード処理

障害の後、アクティブ・トランザクションでアプリケーション・サーバー・インスタンスを再始動する際、トランザクション・サービスはリカバリー・ログを使用して、リカバリー・プロセスを完了します。

これらのリカバリー・ログは、各トランザクションのリソースが保持しており、あらゆる未確定トランザクションを戻し、システム全体を自己矛盾のない状態に戻すために使用されます。未確定トランザクションは、コミット処理中に環境によるエラー、またはその他のエラーが発生したトランザクションを指します。通常の未完

了トランザクションについてもログは記録されますが、コミット処理が成功した後
にこれらのログ・エントリーは削除されます。

このリカバリー・プロセスは、アプリケーション・サーバー内のすべての必要なサ
ブシステムがサーバー始動中に使用可能になると即時に、開始されます。アプリケ
ーション・サーバーは、リカバリー・モードで再始動されていない場合、サーバー
が使用可能な状態になると即時に新しい作業の受け入れを開始できます。そのた
め、リカバリー作業が完了する前に、その作業が発生する可能性があります。多く
の場合、これは問題ありませんが、ここではより保守的なオプションを提示しま
す。つまりこれは、サーバーが「通常」の始動モードで始動された場合でも、サー
バー再始動と同時にリカバリーが実行されるようにすることです。

サーバーの始動方法について詳しくは、 WebSphere Process Server インフォメーシ
ョン・センターの『サーバーの始動 (Starting a server)』トピックを参照してくださ
い。

関連情報

 [doc/tadm_start_man_server.dita](#)

 [doc/cadm_log_files.dita](#)

保存キューと保留キュー

メッセージの処理中に問題が発生すると、そのメッセージは保存キューまたは保留
キューに移されます。

保存キューと保留キューのメッセージに対して、管理コンソールまたはスクリプト
を使用して管理アクションを実行することができます。

場合によっては、保存キューまたは保留キューのメッセージの表示と再生は、リカ
バリー手順の一部になる可能性があります。


関連概念

95 ページの『ユース・ケース: 失敗イベントからのデータのリカバリー
(recovering data from failed events)』

ユース・ケースは、リカバリー・シナリオでのコンテキストとして使用されま
す。このユース・ケースでのビジネスには、新規アカウントを作成する要求を受
け取るアプリケーションがあります。

関連情報

 [ビジネス・プロセス: インフラストラクチャー障害からの回復](#)

 [Failed Event Manager コンソールのヘルプ・フィールドの説明 \(Failed event manager console help field descriptions\)](#)

 [doc/recovery/cadm_failedoverview.dita](#)

 [管理コンソールによる障害メッセージの照会と再生](#)

 [管理スクリプトによる障害メッセージの照会と再生](#)

Business Process Choreographer の保守スクリプトとリカバリー・スクリプト

Business Process Choreographer 用の保守関連のスクリプトがあります。こうした保守スクリプトを、データベースのパフォーマンスを維持するための一般的な保守ポリシーの一部として、または必要なリカバリー・プロセスの一部として実行します。

これらのスクリプトを実行して、テンプレートおよび関連オブジェクトをデータベースから除去する必要があります。また、WebSphere 構成リポジトリで対応する有効なアプリケーションに含まれていない完了済みのプロセス・インスタンスも除去する必要があります。

また、無効なプロセス・テンプレートが存在する可能性もあります。この状態は、ユーザーによってアプリケーションのインストールが取り消されたか、構成リポジトリに保管されなかった場合に発生します。

また、WebSphere Process Server には、Business Process Choreographer のクリーンアップを自動化するサービスも用意されています。サービスは、管理コンソールから実行できます。

Business Process Choreographer のリカバリー・メンテナンスには、以下のスクリプトを使用します。

- `deleteInvalidProcessTemplate.py`

Business Process Choreographer データベースから、有効でなくなったビジネス・プロセス・テンプレートを削除するには、このスクリプトを実行します。

注: これらのテンプレートは、通常影響はありません。これらのテンプレートは、Business Process Choreographer Explorer では表示されません。

このスクリプトを使用して、データベースから有効なアプリケーションのテンプレートを削除することはできません。対応するアプリケーションが有効な場合、この状態は検査され、`ConfigurationError` 例外がスローされます。

- `deleteInvalidTaskTemplate.py`

このスクリプトは、無効になったヒューマン・タスク・テンプレートを Business Process Choreographer データベースから削除するときに実行します。

このスクリプトを使用して、データベースから有効なアプリケーションのテンプレートを削除することはできません。対応するアプリケーションが有効な場合、この状態は検査され、`ConfigurationError` 例外がスローされます。

- `deleteCompletedProcessInstances.py`

このスクリプトは、完了したプロセス・インスタンスをすべて削除する必要がある場合に実行します。

最上位のプロセス・インスタンスは、以下のいずれかの終了状態になったときに完了したものと見なされます。

- 終了

- 強制終了
- 終了
- 失敗

最上位のプロセス・インスタンスと、そのすべての関連データ (アクティビティ・インスタンス、子プロセス・インスタンス、インライン・タスク・インスタンスなど) をデータベースから選択して削除する場合の基準を指定することができます。

注: これらのスクリプトをコマンド行から実行する場合は、要求された操作を WAS 管理クライアントで完了できるように、SOAP クライアントのタイムアウトに十分な値を設定してください。

完了したプロセス・インスタンスの割り当ての削除

開発環境のプロセス・インスタンスの割り当てを削除することができます。

提供された `deleteCompletedProcessInstances.py` のラッパー・スクリプトの使用

このラッパー・スクリプトで正しいユーザー名、パスワード、パスを編集して定義することにより、開発環境からプロセス・インスタンスの割り当てを削除することができます。

適切なタイム・スライスを慎重に選択することにより、デプロイメント・マネージャーと通信する際の SOAP タイムアウト例外を防ぐことができます。

管理可能なインスタンスの「適切なタイム・スライス」とは、以下に示すように多くの要因によって異なりますが、これらの要素だけに限定されるわけではありません。

- JVM のチューニングとメモリーの割り当て
- データベース・サーバーのトランザクション・ログの構成
- SOAP の接続タイムアウト構成

例

例えば、以下のようにスクリプトを変更してコマンドを実行します。

```
wsadmin.<bat|sh> -user<USERNAME> -password<PASSWORD> -f loopDeleteProcessInstances.py 2008-04-02T21:00:00 3600
```







これは、タイム・スタンプ前の完了時間を、毎実行後に 1 時間 (60 分 * 60 秒) 増加しながら、`deleteCompletedProcessInstances.py` を実行します。

`deleteCompletedProcessInstances.py` スクリプトは、タイム・スタンプ・パラメーターを保有しており、このパラメーターを使用して、削除するインスタンスの数を制御できます。時間の間隔が小さいほど、`deleteCompletedProcessInstances.py` を呼び出すたびに削除されるインスタンスも少なくなります。複数のプロセス・インスタンスを削除するとトランザクション・タイムアウトが発生する場合、このパラメーターを指定すると便利です。プロセスの削除中にトランザクション・タイムアウトが発生する最も一般的な原因には、以下のものがあります。

- データベースのチューニング不足

- システムの過負荷
- 大量のプロセス・インスタンスを一度に削除しようとした場合

関連情報

-  プロセス・インスタンス
-  スクリプトによる Business Process Choreographer の管理
-  未使用のプロセス・テンプレートの削除
-  完了したプロセス・インスタンスの削除
-  未使用のヒューマン・タスク・テンプレートの削除
-  クリーンアップ・サービスおよびクリーンアップ・ジョブの構成
(Configuring the cleanup service and cleanup jobs)

未確定トランザクションの解決

ノードの削除によってメッセージング・エンジンが破壊されるなどの例外的な状況が原因で、トランザクションが未確定状態のままになることがあります。

始める前に

この手順を使用して未確定のトランザクションを解決するのは、他の手順 (サーバーをリカバリー・モードで再始動するなど) を行ったが失敗した場合に限ります。

このタスクについて

トランザクションが未確定状態のままになった場合は、影響を受けたメッセージング・エンジンが処理を続行できるように、トランザクションをコミットまたはロールバックする必要があります。

管理コンソールを使用してメッセージ・ポイント上のメッセージをリスト表示することにより、問題の原因となったメッセージを表示することができます。

未確定トランザクションに関するメッセージがある場合は、そのメッセージに関連するパネルにトランザクションの ID が表示されます。この情報を基に、以下のいずれかの方法でトランザクションを解決します。

- サーバーのトランザクション管理パネルを使用する
- メッセージング・エンジンの MBean のメソッドを使用する

最初に、アプリケーション・サーバーのトランザクション管理パネルを使用して未確定トランザクションを解決します。この方法で解決できない場合は、メッセージング・エンジンの MBean のメソッドを使用します。両方の手順について、以下で説明します。

手順

1. アプリケーション・サーバーのトランザクション管理パネルを使用して未確定トランザクションを解決する

- a. 管理コンソールのトランザクション管理パネルにナビゲートします。
「サーバー」 → 「アプリケーション・サーバー」 → [目次ペイン] → 「*server-name*」 → [コンテナ設定] 「コンテナ・サービス」 → 「トランザクション・サービス」 → 「ランタイム」 → 「インポートされた準備済みトランザクション - 検討」をクリックします。
- b. 結果のパネルにトランザクション ID が表示されたトランザクションについては、コミットまたはロールバックすることができます。

トランザクション ID が表示されている場合は、そのトランザクションをコミットまたはロールバックするオプションを選択してください。

トランザクション ID がパネルに表示されていない場合は、そのトランザクションがサーバーのトランザクション・サービスにリストされていない状態になっています。この場合に限り、MBean のメソッド (次のステップで説明します) を使用して、メッセージング・エンジンによって直接管理される未確定トランザクション ID のリストを表示する必要があります。

2. メッセージング・エンジンの MBean のメソッドを使用して未確定トランザクションを解決する

注意:

このステップは、サーバーのトランザクション管理パネルを使用してトランザクション ID を表示することができなかった場合だけ実行してください。

- a. 以下に示すメッセージング・エンジンの MBean のメソッドを使用すると、トランザクション ID (xid) のリストの取得や、トランザクションのコミットおよびロールバックを実行することができます。

- `getPreparedTransactions()`
- `commitPreparedTransaction(String xid)`
- `rollbackPreparedTransaction(String xid)`

- b. メソッドを呼び出すには、`wsadmin` コマンドを使用します。例えば、メッセージング・エンジンの MBean から未確定トランザクション ID のリストを取得する場合は、次の形式のコマンドを使用します。

```
wsadmin> $AdminControl invoke [$AdminControl queryNames
type=SIBMessagingEngine,*] getPreparedTransactions
```

あるいは、以下のようなスクリプトを使用して MBean のメソッドを呼び出すこともできます。

```
foreach mbean [$AdminControl queryNames type=SIBMessagingEngine,*] {
  set input 0

  while {$input >=0} {
    set xidList [$AdminControl invoke $mbean getPreparedTransactions]

    set meCfgId [$AdminControl getConfigId $mbean]
    set endIdx [expr {[string first "(" $meCfgId] - 1}]
    set me [string range ${meCfgId} 0 $endIdx]

    puts "----Prepared Transactions for ME $me ----"
    set index 0
    foreach xid $xidList {
      puts "  Index=$index XID=$xid"
      incr index
    }
  }
}
```

```

puts "----- End of list -----"
puts "Select index of XID to commit/rollback (-1 to continue) :"
set input [gets stdin]

if {$input < 0 } {
puts "No index selected, going to continue."
} else {
set xid [lindex $xidList $input]
puts "Enter c to commit or r to rollback XID $xid"
set input [gets stdin]
if {$input == "c"} {
puts "Committing xid=$xid"
$AdminController invoke $mbean commitPreparedTransaction $xid
}
if {$input == "r"} {
puts "Rolling back xid=$xid"
$AdminController invoke $mbean rollbackPreparedTransaction $xid
}
}
puts ""
}
}

```

このスクリプトにより、トランザクション ID が索引と共にリストされます。このリストから索引を選択し、その索引に対応するトランザクションをコミットまたはロールバックすることができます。

タスクの結果

未確定トランザクションを特定して解決する手順を、以下にまとめます。

1. 管理コンソールを使用して、未確定トランザクションのトランザクション ID を探します。
2. トランザクション管理パネルにトランザクション ID が表示された場合は、必要に応じてトランザクションをコミットまたはロールバックします。
3. トランザクション ID がトランザクション管理パネルに表示されない場合は、メッセージング・エンジンの MBean のメソッドを使用します。例えば、スクリプトを使用して、未確定トランザクションのトランザクション ID のリストを表示します。各トランザクションに対して、以下の操作を実行します。
 - a. トランザクション ID の索引を入力します。
 - b. トランザクションをコミットする場合は、c と入力します。
 - c. トランザクションをロールバックする場合は、r と入力します。
4. トランザクションが未確定状態ではなくなったことを確認するには、サーバーを再始動してトランザクション管理パネルを表示するか、メッセージング・エンジンの MBean のメソッドを使用して確認します。

関連タスク

107 ページの『サービス統合バスの表示』

サービス統合バスを表示させるには、管理コンソールの Service Integration Bus Browser を使用します。

DB2 診断情報の確認

テキスト・エディターを使用して、問題が発生したと思われるマシンの DB2 診断ログ・ファイルを表示します。最新のイベント記録は、ファイルの一番下に表示されます。

このタスクについて

システムが正常に機能しない場合は、DB2 の診断情報を確認してください。これにより、ログ・ファイルがいっぱいかどうかを調べます。

手順

UNIX で、以下のコマンドを入力します。 `tail -f /home/db2inst1/sqllib/db2dump/db2diag.log`

データベースが応答しない場合は、以下のような情報が表示されます。

```
2008-04-03-11.57.18.988249-300 I1247882009G504    LEVEL: Error
PID      : 16020                                TID : 3086133792  PROC : db2agent (WPRCSDB) 0
INSTANCE: db2inst1                            NODE : 000        DB   : WPRCSDB
APPHDL   : 0-658                               APPID: 9.5.99.208.24960.080403084643
AUTHID   : DB2INST1
FUNCTION: DB2 UDB, data protection services, sqlpWriteLR, probe:6680
RETCODE  : ZRC=0x85100009=-2062548983=SQLP_NOSPACE
          "Log File has reached its saturation point"
          DIA8309C Log file was full.
```

```
2008-04-03-11.57.18.994572-300 E1247882514G540    LEVEL: Error
PID      : 16020                                TID : 3086133792  PROC : db2agent (WPRCSDB) 0
INSTANCE: db2inst1                            NODE : 000        DB   : WPRCSDB
APPHDL   : 0-658                               APPID: 9.5.99.208.24960.080403084643
AUTHID   : DB2INST1
FUNCTION: DB2 UDB, data protection services, sqlpgResSpace, probe:2860
MESSAGE  : ADM1823E The active log is full and is held by application handle
          "274". Terminate this application by COMMIT, ROLLBACK or FORCE
          APPLICATION.
```

上記の例で DB 行を見ると、WPRCSDB のトランザクション・ログがいっぱいになっていることがわかります。

次のように、DB2 ユーザーとしてログインして `db2diag` を実行しても、`db2diag` ログを表示することができます。

```
su -l db2inst1
db2diag | less
```

関連情報



診断ログ・ファイルの各項目の解釈

プロセス・リカバリーのトラブルシューティングのヒント






Business Process Choreographer Explorer を使用すると、プロセスのリカバリー作業を簡単に実行することができます。

Business Process Choreographer Explorer には、管理者がビジネス・プロセスおよびヒューマン・タスクを管理するためのユーザー・インターフェースが提供されています。

Business Process Choreographer Explorer を使用して、Business Process Choreographer データベース (BPEDB) の状況を確認することができます。Business Process Choreographer Explorer でデータベース情報を取得できない場合や、Business Process Choreographer からデータベース情報が返されるのが遅い場合は、データベースに問題が発生している可能性があります。

パフォーマンスやデータベースの問題が疑われる場合、大量のプロセス・インスタンスやタスクを取得することは避けてください。この場合は、あまり多くのデータを取得しないビュー（「ユーザーのプロセス・テンプレート」など）を選択するか、取得するデータ量を少なくすることをお勧めします。

関連情報

-  [プロセスおよびアクティビティの修復](#)
-  [Business Process Choreographer Explorer の構成](#)
-  [Business Process Choreographer Explorer の開始](#)
-  [Business Process Choreographer Explorer の概要](#)
-  [Business Process Choreographer Explorer の調整](#)

メッセージング・サブシステムのリカバリーについて

メッセージング・システムに問題が発生した場合、基礎となるメッセージング・サブシステムをリカバリーしなければならない可能性があります。



一般的に、これにはさまざまなキューの状態チェックが含まれますが、統合バス・インフラストラクチャーの分析も含めることができます。

メッセージング・サブシステムのリカバリーに関する詳細な情報は、[WebSphere Application Server インフォメーション・センター](#)を参照してください。

関連概念

92 ページの『シチュエーション分析』
シチュエーション分析とは、科学的手法を周期的に実行することで、リカバリー手順が開始されることになるさまざまなシチュエーションを考慮に入れることができます。

関連情報

-  [サービス統合メッセージに関する問題のトラブルシューティング](#)
-  [doc/covw_esb.dita](#)

特記事項

本書は米国 IBM が提供する製品およびサービスについて作成したものです。

本書に記載の製品、サービス、または機能が日本においては提供されていない場合があります。日本で利用可能な製品、サービス、および機能については、日本 IBM の営業担当員にお尋ねください。本書で IBM 製品、プログラム、またはサービスに言及していても、その IBM 製品、プログラム、またはサービスのみが使用可能であることを意味するものではありません。これらに代えて、IBM の知的所有権を侵害することのない、機能的に同等の製品、プログラム、またはサービスを使用することができます。ただし、IBM 以外の製品とプログラムの操作またはサービスの評価および検証は、お客様の責任で行っていただきます。

IBM は、本書に記載されている内容に関して特許権 (特許出願中のものを含む) を保有している場合があります。本書の提供は、お客様にこれらの特許権について実施権を許諾することを意味するものではありません。実施権についてのお問い合わせは、書面にて下記宛先にお送りください。

〒106-8711
東京都港区六本木 3-2-12
日本アイ・ビー・エム株式会社
法務・知的財産
知的財産権ライセンス渉外

以下の保証は、国または地域の法律に沿わない場合は、適用されません。IBM およびその直接または間接の子会社は、本書を特定物として現存するままの状態を提供し、商品性の保証、特定目的適合性の保証および法律上の瑕疵担保責任を含むすべての明示もしくは黙示の保証責任を負わないものとします。国または地域によっては、法律の強行規定により、保証責任の制限が禁じられる場合、強行規定の制限を受けるものとします。

この情報には、技術的に不適切な記述や誤植を含む場合があります。本書は定期的に見直され、必要な変更は本書の次版に組み込まれます。IBM は予告なしに、随時、この文書に記載されている製品またはプログラムに対して、改良または変更を行うことがあります。

本書において IBM 以外の Web サイトに言及している場合がありますが、便宜のため記載しただけであり、決してそれらの Web サイトを推奨するものではありません。それらの Web サイトにある資料は、この IBM 製品の資料の一部ではありません。それらの Web サイトは、お客様の責任でご使用ください。

IBM は、お客様が提供するいかなる情報も、お客様に対してなんら義務も負うことのない、自ら適切と信ずる方法で、使用もしくは配布することができるものとします。

本プログラムのライセンス保持者で、(i) 独自に作成したプログラムとその他のプログラム (本プログラムを含む) との間での情報交換、および (ii) 交換された情報の相互利用を可能にすることを目的として、本プログラムに関する情報を必要とする方は、下記に連絡してください。

IBM Corporation
1001 Hillsdale Blvd., Suite 400
Foster City, CA 94404
U.S.A.

本プログラムに関する上記の情報は、適切な使用条件の下で使用することができますが、有償の場合もあります。

本書で説明されているライセンス・プログラムまたはその他のライセンス資料は、IBM 所定のプログラム契約の契約条項、IBM プログラムのご使用条件、またはそれと同等の条項に基づいて、IBM より提供されます。

この文書に含まれるいかなるパフォーマンス・データも、管理環境下で決定されたものです。そのため、他の操作環境で得られた結果は、異なる可能性があります。一部の測定が、開発レベルのシステムで行われた可能性があります。その測定値が、一般に利用可能なシステムのもと同じである保証はありません。さらに、一部の測定値が、推定値である可能性があります。実際の結果は、異なる可能性があります。お客様は、お客様の特定の環境に適したデータを確かめる必要があります。

IBM 以外の製品に関する情報は、その製品の供給者、出版物、もしくはその他の公に利用可能なソースから入手したものです。IBM は、それらの製品のテストは行っておりません。したがって、他社製品に関する実行性、互換性、またはその他の要求については確認できません。IBM 以外の製品の性能に関する質問は、それらの製品の供給者にお願います。

IBM の将来の方向または意向に関する記述については、予告なしに変更または撤回される場合があります、単に目標を示しているものです。

本書には、日常の業務処理で用いられるデータや報告書の例が含まれています。より具体性を与えるために、それらの例には、個人、企業、ブランド、あるいは製品などの名前が含まれている場合があります。これらの名称はすべて架空のものであり、名称や住所が類似する企業が実在しているとしても、それは偶然にすぎません。

著作権使用許諾:

本書には、様々なオペレーティング・プラットフォームでのプログラミング手法を例示するサンプル・アプリケーション・プログラムがソース言語で掲載されています。お客様は、サンプル・プログラムが書かれているオペレーティング・プラットフォームのアプリケーション・プログラミング・インターフェースに準拠したアプリケーション・プログラムの開発、使用、販売、配布を目的として、いかなる形式においても、IBM に対価を支払うことなくこれを複製し、改変し、配布することができます。このサンプル・プログラムは、あらゆる条件下における完全なテストを経ていません。従って IBM は、これらのサンプル・プログラムについて信頼性、利便性もしくは機能性があることをほめかしたり、保証することはできません。

それぞれの複製物、サンプル・プログラムのいかなる部分、またはすべての派生的創作物にも、次のように、著作権表示を入れていただく必要があります。(C) (お客様の会社名) (西暦年)。このコードの一部は、IBM Corp. のサンプル・プログラムから取られています。(c) Copyright IBM Corp. 年を入れる。 All rights reserved.

この情報をソフトコピーでご覧になっている場合は、写真やカラーの図表は表示されない場合があります。

プログラミング・インターフェース情報

プログラミング・インターフェース情報がある場合、それらはこのプログラムを使用してアプリケーション・ソフトウェアを作成する際に役立つよう提供されています。

一般使用プログラミング・インターフェースにより、お客様はこのプログラム・ツール・サービスを含むアプリケーション・ソフトウェアを書くことができます。

ただし、この情報には、診断、修正、および調整情報が含まれている場合があります。診断、修正、調整情報は、お客様のアプリケーション・ソフトウェアのデバッグ支援のために提供されています。

警告: 診断、修正、調整情報は、変更される場合がありますので、プログラミング・インターフェースとしては使用しないでください。

商標

IBM、IBM ロゴ、および ibm.com は、International Business Machines Corporation の米国およびその他の国における商標または登録商標です。これらおよび他の IBM 商標に、この情報の最初に現れる個所で商標表示 (R または TM) が付されている場合、これらの表示は、この情報が公開された時点で、米国において、IBM が所有する登録商標またはコモン・ロー上の商標であることを示しています。このような商標は、その他の国においても登録商標またはコモン・ロー上の商標である可能性があります。現時点での IBM の商標リストについては、www.ibm.com/legal/copytrade.shtml の「Copyright and trademark information」をご覧ください。

Microsoft および Windows は、Microsoft Corporation の米国およびその他の国における商標です。

Linux は、Linus Torvalds の米国およびその他の国における商標です。

Java は、Sun Microsystems, Inc. の米国およびその他の国における商標です。

UNIX は The Open Group の米国およびその他の国における登録商標です。

他の会社名、製品名およびサービス名等はそれぞれ各社の商標です。

この製品には、Eclipse Project (<http://www.eclipse.org>) により開発されたソフトウェアが含まれています。



IBM WebSphere Process Server for Multiplatforms バージョン 6.2



Printed in Japan