

**WebSphere®** IBM WebSphere Process Server for  
Multiplatforms  
バージョン 7.0.0

## WebSphere Process Server のモニター





**WebSphere** IBM WebSphere Process Server for  
Multiplatforms  
バージョン 7.0.0

## WebSphere Process Server のモニター



本書は、WebSphere Process Server for Multiplatforms バージョン 7、リリース 0、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。

本書についてのご意見は、[doc-comments@us.ibm.com](mailto:doc-comments@us.ibm.com) へ E メールでお寄せください。皆様の率直なご意見をお待ちしています。

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。

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 7.0.0  
Monitoring WebSphere Process Server

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 2010.4

© Copyright IBM Corporation 2006, 2010.

---

# 目次

<b>第 1 章 サービス・コンポーネント・モニターの概説</b> . . . . .	<b>1</b>
モニターを使用する目的 . . . . .	1
モニター対象 . . . . .	2
モニターを使用可能にする方法 . . . . .	3
<b>第 2 章 サービス・コンポーネント・モニターの有効設定と構成</b> . . . . .	<b>7</b>
パフォーマンスのモニター . . . . .	7
Performance Monitoring Infrastructure 統計 . . . . .	8
Service Component Architecture のアプリケーション 応答測定統計 . . . . .	14
サービス・コンポーネント・イベントのモニター . . . . .	28
ビジネス・プロセス・イベントとヒューマン・タ スク・イベントのモニターの使用可能化 . . . . .	29
サービス・コンポーネント・イベントのロギング の構成 . . . . .	30
Common Event Infrastructure サーバーでのサービ ス・コンポーネントのモニター . . . . .	38
セッション・モニター . . . . .	42

<b>第 3 章 モニター対象イベントの表示</b> . . . . .	<b>45</b>
Tivoli Performance Viewer でのパフォーマンス・メ トリックの表示 . . . . .	45
サービス・コンポーネント・イベント・ログ・ファ イルの表示と解釈 . . . . .	47
<b>第 4 章 イベント・カタログ</b> . . . . .	<b>49</b>
Common Base Event の標準エレメント . . . . .	49
イベント内のビジネス・オブジェクト . . . . .	51
Business Process Choreographer イベント . . . . .	51
WebSphere Process Server イベント . . . . .	51
リソース・アダプター・イベント . . . . .	52
ビジネス・ルール・イベント . . . . .	53
ビジネス・ステート・マシン・イベント . . . . .	54
マップ・イベント . . . . .	55
メディエーション・イベント . . . . .	56
リカバリー・イベント . . . . .	57
Service Component Architecture イベント . . . . .	57
セレクター・イベント . . . . .	58



---

## 第 1 章 サービス・コンポーネント・モニターの概説

プロセス・サーバーでサービス・コンポーネントをモニターする理由、モニターするサービス・コンポーネント内のイベント・ポイントの選択、およびシステムのモニターの構成方法などを概念的に概説します。

WebSphere® Process Server には、サービス・コンポーネントをモニターする機能が  
あり、パフォーマンスの調整や問題判別などのシステム管理機能を支援します。さら  
に、こうした従来の機能だけでなく、必ずしも情報技術の専門家ではないユーザ  
ーに対して、システムにデプロイされたアプリケーション内のサービス・コンポー  
ネントの処理を継続的にモニターする機能も提供します。相互接続されたコンポー  
ネントの全体的な処理フローを監視することにより、システムで本来行われるべき  
処理が行われていることを確認することができます。

WebSphere Process Server は、WebSphere Application Server インストール済み環境  
の上部で稼働します。その結果、システム・パフォーマンスのモニターとトラブル  
シューティングで、アプリケーション・サーバー・インフラストラクチャーの機能  
の多くを使用します。また、プロセス・サーバーのサービス・コンポーネントをモ  
ニターするために設計された追加の機能も組み込まれています。このセクションで  
は、サーバー固有のサービス・コンポーネントのモニター方法を中心に説明しま  
す。このセクションは、WebSphere Application Server インフォメーション・センタ  
ーに掲載されたモニターおよびトラブルシューティングについてのトピックを補足  
することが目的です。そのため、製品を組み合わせて使用する場合の他のモニター  
機能の詳細については、このインフォメーション・センターを参照してください。

---

### モニターを使用する目的

WebSphere Process Server 内でサービス・コンポーネントをモニターする目的は、  
パフォーマンスを評価し、問題をトラブルシューティングして、システムにデプロ  
イされているアプリケーションを構成するサービス・コンポーネントの全体的な処  
理の進捗を評価することです。

サービス・コンポーネントは、WebSphere Process Server に組み込まれている重要  
な機能で、これを使用すると、社内で使用されているプロセスを反映したアプリケ  
ーションを、ご使用のシステム上で作成してデプロイできます。したがって、サー  
バーによって完了しようとするタスクを管理するには、それらのサービス・コンポ  
ネントを効果的にモニターすることが重要です。サーバーでサービス・コンポー  
ネントをモニターしなければならない理由は主に 3 つあります。

#### 問題判別

WebSphere Process Server の基盤となる WebSphere Application Server のロ  
ギング機能とトレース機能を使用して、特定のエラーを診断することができます。  
例えば、特定のアプリケーションが期待される結果を生み出していない場合は、  
アプリケーションを構成するサービス・コンポーネントの処理を  
モニターするようにロガーをセットアップできます。ログ出力をファイルに  
パブリッシュし、そのファイルを調べて問題の原因を特定することができます

す。トラブルシューティングは、システムのハードウェアとソフトウェアの保守に携わる管理者やその他の担当者にとって重要なタスクです。

### パフォーマンス調整

大半のプロセス・サーバー固有サービス・コンポーネントが作成する特定のパフォーマンス統計をモニターすることができます。この情報を使用して、システムの正常性を維持および調整し、アプリケーションが最適かつ効率的であるように調整されていることを確認します。また、1 つ以上のサービスが低レベルで実行されている状態を発見できます。この場合、システムに別の問題が存在する可能性があります。問題判別と同様、一般的にパフォーマンス調整は IT スペシャリストが実行する作業です。

### サービス・コンポーネントの処理の評価

問題判別とパフォーマンス調整は、特定の問題を解決するために短期的に実行するタスクです。システムにデプロイされているアプリケーションに組み込まれたサービス・コンポーネントを継続的にモニターするようにプロセス・サーバーをセットアップすることもできます。このタイプのサービス・コンポーネント・モニターは、設計、インプリメント、およびプロセスが設計目標を達成するようにする責任がある担当者にとって重要です。そうした担当者は、必ずしも IT 分野の熟練したスペシャリストであるとは限りません。

---

## モニター対象

処理中にサービス・コンポーネント・イベントが到達する特定のポイントを選択することによって、WebSphere Process Server のサービス・コンポーネント・イベントをモニターできます。各サービス・コンポーネントは、それらのイベント・ポイントを定義します。イベント・ポイントは、その所定のポイントでアプリケーションが処理を行ったときに、イベントを「送出」します。また、サービス・コンポーネント・イベントのパフォーマンス統計をモニターすることもできます。

サービス・コンポーネントで実行するモニターのタイプ (問題判別、パフォーマンス調整、またはプロセス・モニター) に関係なく、処理中に到達する特定のポイントをモニターします。このポイントは、イベント・ポイントと呼ばれ、これらのポイントをモニター対象として選択します。各イベント・ポイントは、サービス・コンポーネントの種類タグ、オプションの要素 種類 (サービス・コンポーネント・タイプの固有機能)、およびイベントの性質 をカプセル化します。これらのすべての因子により、モニターによって生成されるイベントのタイプが決定します。

イベント性質は、サービス・コンポーネントの処理中にイベントを生成するために必要な状態について説明します。これらの性質は、モニターの対象として選択するサービス・コンポーネントの論理構造におけるキーポイントです。サービス・コンポーネント・イベントの最も一般的な性質は ENTRY、EXIT、および FAILURE ですが、特定のコンポーネントや要素によっては他にも多くの性質があります。指定されたサービス・コンポーネントを持つアプリケーションが後で呼び出される場合には常に、サービス・コンポーネントの処理がイベント性質に対応するポイントを経由するたびにイベントが送出されます。

サービス・コンポーネント種類のイベントの定義方法の例として、MAP サービス・コンポーネント種類を挙げます。MAP サービス・コンポーネント種類は ENTRY、



EXIT、および FAILURE の性質を持つイベントを直接送出することができます。また、MAP コンポーネント種類内の特定タイプの機能を定義する「変換」というエレメント種類も含まれています。このエレメントも、ENTRY、EXIT、および FAILURE の性質を持つイベントを送出します。その結果、MAP サービス・コンポーネント種類は、指定するエレメントと性質の組み合わせに応じて、異なるイベントを 6 つまで送出することができます。イベント・カタログには、すべてのサービス・コンポーネントとそのエレメント、およびそのイベント性質のリストがあります。

モニターはアプリケーション処理の最上位にある機能の分離されたレイヤーであるため、サービス・コンポーネントの処理に干渉することはありません。モニターは、指定されたイベント・ポイントでアクティビティーを検出した場合のみサービス・コンポーネント処理に関係します。アクティビティーが検出されると、モニターによってイベントが送出されます。モニターは実行するモニターのタイプに基づいて、イベントの送付先およびイベントに含まれるデータを判断します。

#### パフォーマンス・メトリック

パフォーマンス・メトリックを収集するためにサービス・コンポーネントをモニターしている場合は、軽負荷イベントが Performance Monitoring Infrastructure に送出されます。サーバー固有のサーバー・コンポーネント用に生成される 3 種類のパフォーマンス統計の中から、1 つ以上のパフォーマンス統計をモニター対象として選択できます。

- 各 EXIT イベント性質のカウンター: 成功した計算をカウント。
- 各 FAILURE イベント性質のカウンター: 失敗した計算をカウント。
- 対応する ENTRY イベントと EXIT イベント (同期計算のみ) の間の処理所要時間。

また、アプリケーション応答測定 (ARM) 統計を使用することによって、Service Component Architecture (SCA) レベルでアプリケーションのパフォーマンスをモニターすることもできます。これらの指標を使用すると、アプリケーション内で、他のサービス・コンポーネント・イベントで使用可能な方法よりも詳細なレベルで、アプリケーションをモニターすることができます。この統計で SCA が使用されている場合、この統計を使用して、初期アプリケーション呼び出しとサービス応答との間の多数の異なるポイントをモニターできます。

#### ビジネス・オブジェクトを伴うサービス・コンポーネント・イベント

サービス・コンポーネントの指定されたイベント・ポイントでモニターによって送出されるイベントからデータを収集する場合、サーバーがイベントを生成し、そのデータが Common Base Event 形式でエンコードされるように構成します。各サービス・コンポーネント・イベント内で収集するビジネス・オブジェクト・データの詳細レベルを指定できます。これらのイベントは、ロガーか Common Event Infrastructure (CEI) バスのどちらかにパブリッシュできます。このパブリッシュでは、出力は特別に構成された CEI サーバー・データベースに送信されます。

---

## モニターを使用可能にする方法

行うモニターのタイプに応じて、モニター用サービス・コンポーネント・イベント・ポイントを指定する方法はいくつかあります。

## パフォーマンス統計

Performance Monitoring Infrastructure (PMI) 統計の場合、管理コンソールを使用して、モニターする特定のイベント・ポイントとそれに関連するパフォーマンス測定を指定します。モニター・サービス・コンポーネント・パフォーマンスを開始した後、生成された統計は一定の間隔で Tivoli® Performance Viewer にパブリッシュされます。このビューアーを使用すると、システムで結果が生成されてから見ることができます。また、オプションで、その結果をファイルにログ記録して、あとで同じビューアー内で表示および分析することもできます。

アプリケーション応答測定 (ARM) 統計の場合、管理コンソールの「要求メトリック」セクションを使用して、モニターする統計を指定します。

## 問題判別およびビジネス・プロセス・モニターを行う Common Base Event

アプリケーションの作成時、稼働中のサーバーにアプリケーションをデプロイした後にサービス・コンポーネント・イベント・ポイントが継続的にモニターされるように指定できます。また、イベントの詳細レベルも同時に指定できます。アプリケーションがデプロイされ、イベントが 1 回以上呼び出された後に、モニター対象のイベント・ポイントを選択することもできます。どちらの場合にも、モニターによって生成されるイベントは、Common Event Infrastructure (CEI) バスを介して送出されます。これらのイベントは、ログ・ファイルにパブリッシュすることも、構成済み CEI サーバーのデータベースにパブリッシュすることもできます。WebSphere Process Server では、問題判別およびビジネス・プロセス・モニターのために Common Base Event を使用可能にする以下の 2 種類の方法をサポートしています。

**静的** WebSphere Integration Developer ツールを使用して、アプリケーション内の特定のイベント・ポイントとその詳細レベルに、モニター用のタグを付けることができます。これらの選択項目が示す内容は、連続的なモニターの対象となるイベント・ポイント、およびアプリケーションとともに配布されデプロイされる .mon 拡張子付きのファイルに保管されるイベント・ポイントです。CEI サーバーを使用するように WebSphere Process Server を構成すると、指定したサービスが呼び出されるたびに、モニター機能によってサービス・コンポーネント・イベントの CEI サーバーへの送出が開始されるようになります。アプリケーションが WebSphere Process Server にデプロイされている限り、.mon ファイルに指定されているサービス・コンポーネント・イベント・ポイントは、そのアプリケーションが停止するまで常時モニターされます。実行中のアプリケーションでモニターされる追加イベントを指定したり、すでにモニターされているイベント・ポイントの詳細レベルを上げたりすることもできます。ただし、アプリケーションがアクティブである間は、デプロイ済みアプリケーションの .mon ファイルで指定されているモニター対象イベント・ポイントを停止したり、その詳細レベルを低くしたりすることはできません。

**動的** アプリケーションの処理中に、サーバーをシャットダウンしないで追加イベント・ポイントをモニターする必要がある場合は、動的モニターを使用できます。管理コンソールを使用してモニターするサービス・コンポーネント・イベント・ポイントを指定し、Common

Base Event に含まれる有効搭載量の詳細レベルを設定できます。サーバーの始動後に、処理済みサービス・コンポーネントが到達したイベント・ポイントのリストが作成されます。このリストから、モニターする個別のイベント・ポイントまたはイベント・ポイントのグループをロガーまたは CEI サーバー・データベースに送信されるサービス・コンポーネント・イベントとともに選択します。

動的使用可能化の主な目的は、ログにパブリッシュされる関連サービス・コンポーネント・イベントを作成し、サービスでの問題判別を実行できるようにすることです。サービス・コンポーネント・イベントのサイズは大きくなる可能性があります (要求されるデータ量に応じて異なります)。また、イベントを CEI サーバーに送信する場合、データベース・リソースに負荷がかかることがあります。そのため、イベントのビジネス・データを読み取る必要がある場合、または読み取る必要はないがイベントのデータベース・レコードを保持する必要がある場合のみ、モニター済みイベントを CEI サーバーに動的にパブリッシュしてください。ただし、特定のセッションをモニターしている場合、そのセッションに関連するサービス・コンポーネント・イベントにアクセスするためには、CEI サーバー・データベースを使用する必要があります。

### 関連概念

7 ページの『パフォーマンスのモニター』

サービス・コンポーネント・イベント・ポイントに対してパフォーマンス測定を実行できます。パフォーマンス測定は、Performance Monitoring Infrastructure を使用して処理します。プロセス・サーバーでサービス・コンポーネント・イベント・ポイントからパフォーマンス・メトリックを収集するように構成します。また、Service Component Architecture 固有のパフォーマンス統計をアプリケーションのサーバー呼び出しから直接収集することもできます。

42 ページの『セッション・モニター』

Common Base Event ブラウザーを使用して、Common Event Infrastructure データベースで同じセッション ID 属性を持つすべてのイベントを検索することによって、同じセッションの一部である複数のイベントをモニターすることができます。

### 関連タスク

7 ページの『第 2 章 サービス・コンポーネント・モニターの有効設定と構成』

サービス・コンポーネントをモニターできるようにするには、最初にモニター機能を使用可能にする必要があります。次に、モニターするイベント、イベントから収集する情報、および結果のパブリッシュで使用する方法を指定する必要があります。

 Common Event Infrastructure の管理

 Common Base Events および監査証跡の使用可能化 (管理コンソール使用)

 要求メトリックからのパフォーマンス・データの取得



---

## 第 2 章 サービス・コンポーネント・モニターの有効設定と構成

サービス・コンポーネントをモニターできるようにするには、最初にモニター機能を使用可能にする必要があります。次に、モニターするイベント、イベントから収集する情報、および結果のパブリッシュで使用する方法を指定する必要があります。

### 関連概念

3 ページの『モニターを使用可能にする方法』

行うモニターのタイプに応じて、モニター用サービス・コンポーネント・イベント・ポイント指定する方法はいくつかあります。

---

### パフォーマンスのモニター

サービス・コンポーネント・イベント・ポイントに対してパフォーマンス測定を実行できます。パフォーマンス測定は、Performance Monitoring Infrastructure を使用して処理します。プロセス・サーバーでサービス・コンポーネント・イベント・ポイントからパフォーマンス・メトリックを収集するように構成します。また、Service Component Architecture 固有のパフォーマンス統計をアプリケーションのサーバー呼び出しから直接収集することもできます。

サービス・コンポーネントを調整して効率を最適化する場合も、ローパフォーマンスを診断する場合も、パフォーマンスの観点から、さまざまなランタイム・リソースおよびアプリケーション・リソースの動作を理解しておくことは重要です。

Performance Monitoring Infrastructure (PMI) では、ランタイム・リソースとアプリケーション・リソースの動作を説明する包括的なデータを提供します。PMI データを使用して、アプリケーション・サーバーにおけるパフォーマンスのボトルネックを識別し、修正できます。また、アプリケーション・サーバーの正常性をモニターする目的で PMI データを使用することもできます。

PMI は、ベースとなっている WebSphere Application Server インストールに組み込まれています。このセクションでは、WebSphere Process Server に固有のサービス・コンポーネントに関連するパフォーマンス・モニターについての補足的な情報のみを提供します。そのため、製品全体のほかの部分で PMI を使用方法については、WebSphere Application Server の資料の情報を参照してください。

PMI によってモニター可能な WebSphere Process Server に固有のサービス・コンポーネント・イベント・ポイントは、ENTRY、EXIT、および FAILURE というイベント性質を含むイベントです。このパターンに従って定義されていないイベント・リソースはサポートされません。サポートされるイベントには、測定可能なパフォーマンス統計の 3 つのタイプがあります。

- 成功した呼び出し。
- 失敗した呼び出し。
- イベント完了までの経過時間。

また、Application Response Measurement (ARM) 統計を使用することによって、アプリケーションのサービス呼び出しから取得したパフォーマンス統計をモニターすることもできます。これらの統計では、エンタープライズ・アプリケーションを構成するプロセス・サーバーのサービス・コンポーネント・イベントの基礎となる実際のランタイム・プロセスを測定します。これらの統計を使用して、アプリケーション処理のさまざまなパフォーマンス測定を取得することができます。

### 関連概念

3 ページの『モニターを使用可能にする方法』  
行うモニターのタイプに応じて、モニター用サービス・コンポーネント・イベント・ポイントを指定する方法はいくつかあります。

## Performance Monitoring Infrastructure 統計

Performance Monitoring Infrastructure を使用して、3 つのタイプのパフォーマンス統計をモニターできます。つまり、呼び出しの成功回数、失敗回数、およびイベント完了までの経過時間です。これらの統計は、ENTRY、EXIT、および FAILURE のタイプの性質を持つイベントにのみ使用可能です。

### 管理コンソールを使用した PMI の使用可能化

パフォーマンス・データをモニターするには、まずサーバーで Performance Monitoring Infrastructure を使用可能にする必要があります。

### このタスクについて

Performance Monitoring Infrastructure (PMI) は、管理コンソールを使用して使用可能にすることができます。

### 手順

1. 管理コンソールを開きます。
2. コンソールのナビゲーション・ツリーで、「サーバー」>「サーバー・タイプ」>「WebSphere Application Server」をクリックします。
3. *server\_name* をクリックします。

注: 管理コンソールから、「モニターおよびチューニング」>「Performance Monitoring Infrastructure (PMI)」>「*server\_name*」をクリックして同じパネルを開きます。

4. 「構成」タブをクリックします。
5. 「Performance Monitoring Infrastructure (PMI) を使用可能にする」チェック・ボックスを選択します。
6. オプション: 「順次カウンター更新を使用」チェック・ボックスを選択して、正確な統計の更新を使用可能に設定します。
7. サーバー名のリンクをクリックして、「サーバー PMI 構成 (server PMI configuration)」ページに戻ります。
8. 「適用」または「OK」をクリックします。
9. 「保管」をクリックします。
10. サーバーを再始動します。

## 次のタスク

変更内容は、サーバーを再始動するまで有効になりません。

### イベント・パフォーマンス統計

パフォーマンス・モニター統計は、大部分のサーバー・イベントについて使用可能です。パフォーマンス・モニター統計を使用して、成功および失敗した呼び出し要求の回数、およびイベントの完了に要した時間をモニターできます。

Performance Monitoring Infrastructure (PMI) を使用して、特定のサーバー・イベントにより生成される、以下の表に示す 3 つのパフォーマンス統計をモニターできます。

表 1. イベントの PMI 統計

統計名	タイプ	説明
BadRequests	カウンター	イベントの呼び出しが失敗した数。
GoodRequests	カウンター	イベントの呼び出しが成功した数。
ResponseTime	タイマー	イベント完了までの経過時間。

以下の統計は、ENTRY、EXIT、および FAILURE の性質を備えたエレメントを持つサービス・コンポーネント・イベントだけに限られます。各統計は、アプリケーションにおいて、所定のサーバー・イベント・タイプの 1 イベントごとに作成されます。すべてのパフォーマンス測定は、カウンター (指定されたイベント・ポイントの実行累積数) かタイマー (2 つのイベント・ポイント実行間隔の期間 (単位: ミリ秒)) のどちらかです。モニターできる各イベント種類 (および関連するエレメント) を以下にリストします。

表 2. イベント・パフォーマンス統計を生成できるイベント・タイプおよびエレメント

イベント・タイプ	エレメント
ビジネス・プロセス	Process Invoke Staff Receive Wait Compensate Pick Scope
ヒューマン・タスク	Task
ビジネス・ルール	Operation
ビジネス・ステート・マシン	Transition Guard Action EntryAction ExitAction
セレクター	Operation
マップ	Map Transformation

表2. イベント・パフォーマンス統計を生成できるイベント・タイプおよびエレメント (続き)

イベント・タイプ	エレメント
メディエーション	OperationBinding ParameterMediation
リソース・アダプター	InboundEventRetrieval InboundEventDelivery Outbound

## 関連資料

14 ページの『Service Component Architecture のアプリケーション応答測定統計』Service Component Architecture (SCA) レベルで、25 件のパフォーマンス統計をモニターできます。これらのアプリケーション応答測定 (ARM) 統計 (これらはカウンターかタイマーです) を使用して、さまざまなパターンのサービスの呼び出しと応答を測定することができます。

## モニターするパフォーマンス統計の指定

管理コンソールを使用することで、Performance Monitoring Infrastructure によるモニター用に 1 つの統計、複数の統計、または関連する統計のグループを指定することができます。

## 始める前に

このタスクを実行する前に、パフォーマンス・モニターを使用可能にしていること、およびモニター対象のイベントを少なくとも一度は必ず呼び出していることを確認します。

## 手順

1. 管理コンソールを開きます。
2. 「モニターおよびチューニング」 → 「Performance Monitoring Infrastructure (PMI)」を選択します。
3. モニターするイベント・ポイントを含んでいるサーバーまたはノード・エージェントを選択します。

**注:** クラスター上で統計のモニターを選択することはできません。それは特定のサーバーまたはノード上でのみ行うことができます。

4. WBIStats.RootGroup またはエンタープライズ Bean など、いくつかのグループを展開します。モニター可能なすべての統計は、リストされたグループ内にあります。サーバーの最後の始動以降呼び出されていないためにリストされない統計もあります。
5. パネルの左側にあるツリーからモニターする統計を選択し、右側で収集する統計を選択して、「使用可能にする」をクリックします。モニターするすべての統計に対して繰り返します。
6. サーバー名のリンクをクリックして、「サーバー PMI 構成 (server PMI configuration)」ページに戻ります。
7. 「適用」または「OK」をクリックします。
8. 「保管」をクリックします。



## タスクの結果

これで、Tivoli Performance Viewer で、選択した統計のパフォーマンスのモニターを開始できます。

**注:** 統計を参照するときに、カウンター・タイプの統計と期間タイプの統計を混合しないでください。カウンターは累積されるため、アプリケーションによってはグラフのスケールがすぐに大きくなる可能性があります。反対に、期間型統計はシステムが各イベントを処理するのにかかる時間の平均を示すため、一定の範囲に保たれます。このため、統計とその相対目盛りの不均衡が原因で、いずれかのタイプの統計がビューアーのグラフ上で偏ることがあります。

## チュートリアル: サービス・コンポーネントのパフォーマンス・モニター

このチュートリアルでは、パフォーマンス・モニターの設定例と、結果統計を表示する方法について説明します。

モニターするサービス・コンポーネント・イベント・ポイントについて、Performance Monitoring Infrastructure (PMI) にパブリッシュし、その結果のパフォーマンス統計を Tivoli Performance Viewer (TPV) に表示することができます。この演習では、サービス・コンポーネント・イベント・ポイントのパフォーマンス・モニターと、Common Event Infrastructure (CEI) サーバーおよびロガーを使用したモニターとの違いを示します。大きな違いは、パフォーマンス・モニターの場合、特定の性質を持つ個別のイベントを選択するのではなく、サービス・コンポーネント・エレメント全体を選択する点です。WebSphere Process Server がパフォーマンスをモニターできるのは ENTRY、EXIT、および FAILURE という性質のイベントを持つサービス・コンポーネント・エレメントのみであるため、モニター対象として選択できるのはそのようなサービス・コンポーネント・エレメントのみです。

ENTRY、EXIT、および FAILURE の各サービス・コンポーネント・イベント・ポイントがすべてのモニター・タイプで同じであるのに対し、サーバーのパフォーマンス・モニター機能では、CEI イベントに含まれる一部の情報を含む「最小化された」イベントを送出します。これらのイベントは PMI に送信され、対応するイベント・セットから次のパフォーマンス統計が計算されます。

- 正常な呼び出し - 対応する ENTRY イベントに続く、EXIT 性質のイベントの起動。
- 失敗した呼び出し - 対応する ENTRY イベントに続く、FAILURE 性質のイベントの起動。
- 正常終了の時間 - ENTRY イベントが起動してから、それに対応する EXIT イベントが起動するまでの経過時間。

PMI は TPV に統計をパブリッシュし、TPV では、成功した呼び出しと失敗した呼び出しの数を示す累積カウンターおよび完了応答時間の実行平均を表示します。

### このチュートリアルの目的

このチュートリアルを終了すると、次の操作ができるようになります。

- モニターするサービス・コンポーネント・エレメントのパフォーマンス統計を選択する。

- パフォーマンス統計結果を表示および解釈する。

## このチュートリアルを完了するのに必要な時間

このチュートリアルを完了するには、およそ 15 分から 20 分かかります。

## 前提条件

このチュートリアルを実行するには、次の条件を満たす必要があります。

- サーバーを構成および始動済みである。
- サーバーで PMI を使用可能に設定済みである。
- サーバーでサンプル・ギャラリー・アプリケーションがインストールおよび始動済みである。
- サーバーでビジネス・ルール・サンプル・アプリケーションがインストールおよび始動済みである。「サンプル・ギャラリー」ページの指示に従って、ビジネス・ルール・サンプル・アプリケーションをセットアップし、実行します。

これらのすべての前提条件が満たされたら、チュートリアルに進む前に、少なくとも一度サンプル・ギャラリーからビジネス・ルール・サンプル・アプリケーションを実行してください。

### 例: サービス・コンポーネントのパフォーマンスのモニター:

パフォーマンスのモニターでは、管理コンソールを使用して、モニター対象のサービス・コンポーネントを選択したり、パフォーマンス測定を表示したりすることができます。この例では、コンソールを使用して、パフォーマンス統計をモニターする方法を示します。

### このタスクについて

このシナリオでは、ビジネス・ルール・サンプル・アプリケーションを使用して、3 つのすべてのパフォーマンス統計 (成功、失敗、および応答時間) をモニターします。このアプリケーションが配置されている Web ページを開いておいてください。モニターの開始後にサンプルを数回実行するため、ページは開いたままにしておいてください。サンプルを少なくとも一度実行しておいてください。実行しておく、サンプルがモニター対象として選択可能な機能のリストに表示されます。

### 手順

1. 管理コンソールを開きます。
2. モニターするクラスターまたはサーバーを選択します。
  - クラスターをモニターするには、「サーバー」 → 「クラスター」 → 「WebSphere Application Server クラスター (WebSphere application server clusters)」 → 「*cluster\_name*」 をクリックします。
  - 単一のサーバーをモニターするには、「サーバー」 → 「サーバー・タイプ」 → 「WebSphere Application Server」 → 「*server\_name*」 をクリックします。
3. 「ランタイム (Runtime)」 タブをクリックします。
4. 「パフォーマンス」の下で、「Performance Monitoring Infrastructure (PMI)」 をクリックします。

5. 「カスタム」を選択します。
6. 「WBISStats.RootGroup」 → 「BR」 → 「brsample\_module.DiscountRuleGroup」 → 「操作」を展開します。
7. `_calculateDiscount` を選択します
8. BadRequests、GoodRequests、および ResponseTime の横にあるチェック・ボックスを選択します。
9. 「使用可能にする」をクリックします。
10. ナビゲーション・ペインで、「モニターおよびチューニング」 → 「Performance Viewer」 → 「現行アクティビティ」をクリックします。
11. 「*server\_name*」の横のチェック・ボックスを選択して、「モニターの開始」をクリックします。
12. *server\_name* をクリックします。
13. 「WBISStats.RootGroup」 → 「BR」 → 「brsample\_module.DiscountRuleGroup」 → 「操作」を展開します。
14. `_calculateDiscount` の横にあるチェック・ボックスを選択します。

### タスクの結果

空白のグラフが表示され、グラフの下には 3 つの統計の名前と値が表示されます。統計名の横のチェック・ボックスが選択されていない場合、それを選択します。これで、PMI では選択されたイベントのパフォーマンス・データのパブリッシュが、Tivoli Performance Viewer では結果の表示ができます。

ビジネス・ルール・サンプル・アプリケーションを数回実行し、パフォーマンス・ビューアーが定期的に最新表示されるのを確認します。グラフには、成功した要求の累積数と各成功した要求の平均応答時間を表す線が表示されます。また、グラフの下には、各統計の名前の横に値が表示されます。成功数を表す線はサンプルをさらに追加で呼び出すにつれて上昇するのに対し、応答時間の線は、最新表示が 2、3 回行われた後で水平になります。

この例を完了すれば、WebSphere Process Server がサービス・コンポーネントのパフォーマンス・モニターをインプリメントする方法を理解したことになります。モニターするサービス・コンポーネントを選択する方法や、パフォーマンス統計の計算方法についても理解できます。また、パフォーマンス・モニターを開始し、アプリケーション・パフォーマンスのリアルタイムの測定を表示することもできます。

### 次のタスク

パフォーマンス・モニターはシステム・リソースに負荷をかける可能性があります。そのため、このタスクが完了したらモニターを停止してください。モニターを停止するには、Tivoli Performance Viewer リンクをクリックし、ノードとサーバーの両方を選択して「モニターの停止」を押します。

## Service Component Architecture のアプリケーション応答測定統計

Service Component Architecture (SCA) レベルで、25 件のパフォーマンス統計をモニターできます。これらのアプリケーション応答測定 (ARM) 統計 (これらはカウンターかタイマーです) を使用して、さまざまなパターンのサービスの呼び出しと応答を測定することができます。

以下の表に示すアプリケーション応答測定 (ARM) 統計は、Service Component Architecture (SCA) 層に対する呼び出し元による呼び出しおよびサービスから戻される結果の、時間およびカウントの測定 (単純な方法での測定) です。事実、多数のサービス呼び出しパターンがあります。それは、遅延応答、結果取得、コールバック、および片方向呼び出しの同期実装と非同期実装とは異なります。しかし、パターンはすべて、呼び出し元による呼び出しとサービス、サービスからの応答、または場合によってはデータ・ソースとの間で、SCA 層を介して行われます。

モニターする ARM 統計を指定するには、管理コンソールで「モニターおよびチューニング」>「要求メトリック」パネルを開きます。要求メトリック情報は、後で取り出して分析できるようログ・ファイルに保存されるか、ARM エージェントへ送信されるか、あるいはその両方で処理されます。WebSphere Process Server には ARM エージェントは同梱されていませんが、ARM 4.0 に準拠したエージェントの使用はサポートされています。ユーザーは独自の ARM 実装プロバイダーを選択して、ARM 実装ライブラリーを入手できます。ARM プロバイダーからの指示に従い、ARM プロバイダーにある ARM API Java™ アーカイブ (JAR) ファイルをクラスパス上に置くようにして、WebSphere Process Server が必要なクラスをロードできるようにします。次に、以下の項目を各サーバーのシステム・プロパティーに追加する必要があります。そのためには、管理コンソールから「アプリケーション・サーバー」> *server\_name* >「プロセス定義」>「Java 仮想マシン」>「カスタム・プロパティー」を選択します。その後、サーバー (単数または複数) を再始動します。

- `Arm40.ArmMetricFactory` - ARM 実装プロバイダーのメトリック・ファクトリーの完全 Java クラス名。
- `Arm40.ArmTranReportFactory` - ARM 実装プロバイダーのトランザクション・レポート・ファクトリーの完全 Java クラス名。
- `Arm40.ArmTransactionFactory` - ARM 実装プロバイダーのトランザクション・ファクトリーの完全 Java クラス名。

ARM 統計を収集するためのサーバーの詳しい構成方法については、WebSphere Application Server の資料を参照してください。

表 3. ARM 統計を生成できるイベント・タイプおよびエレメント

イベント・タイプ	エレメント
ビジネス・プロセス	Process
ヒューマン・タスク	Task
ビジネス・ルール	Operation
ビジネス・ステート・マシン	Transition Guard Action EntryAction ExitAction

表 3. ARM 統計を生成できるイベント・タイプおよびエレメント (続き)

イベント・タイプ	エレメント
セレクター	Operation
マップ	Map Transformation
メディエーション	OperationBinding ParameterMediation
リソース・アダプター	InboundEventRetrieval InboundEventDelivery Outbound

表 4. 共通：以下の統計は、すべてのサービス呼び出しパターンに共通です。

統計名	タイプ	説明
<b>GoodRequests</b>	カウンター	例外が発生しなかったサービス呼び出しの数。
<b>BadRequests</b>	カウンター	例外が発生したサーバー呼び出しの数。
<b>ResponseTime</b>	タイマー	サーバー・サイドで測定される、要求の受信と結果の計算の間の時間。
<b>TotalResponseTime</b>	タイマー	呼び出し側で測定される、呼び出し側がサービスを要求してから呼び出し側に結果が戻るまでの時間。呼び出し側による結果の処理は含まれません。
<b>RequestDeliveryTime</b>	タイマー	呼び出し側で測定される、呼び出し側がサービスを要求してからサーバー・サイドの実装環境に要求が受け渡されるまでの時間。分散環境では、この測定の正確性はシステム・クロックの同期の正確性によって異なります。
<b>ResponseDeliveryTime</b>	タイマー	結果がクライアントに戻るまでに必要な時間。遅延応答の場合、この時間には結果取得時間は含まれません。分散環境では、この測定の正確性はシステム・クロックの同期の正確性によって異なります。

表 5. 参照：以下の統計は、サービスからの応答なしに、呼び出し側が SCA 層またはデータ・ソースを呼び出した場合に作成されます。

統計名	タイプ	説明
<b>GoodRefRequests</b>	カウンター	例外を発生させない SCA 層への呼び出し側による呼び出しの数。
<b>BadRefRequests</b>	カウンター	例外を発生させる SCA 層への呼び出し側による呼び出しの数。
<b>RefResponseTime</b>	タイマー	呼び出し側で測定される、呼び出し側が SCA 層に要求を出してから呼び出し側に呼び出しの結果が戻るまでの時間。
<b>BadRetrieveResult</b>	カウンター	例外を発生させるデータ・ソースへの呼び出し側による呼び出しの数。

表5. 参照 (続き)：以下の統計は、サービスからの応答なしに、呼び出し側が SCA 層またはデータ・ソースを呼び出した場合に作成されます。

統計名	タイプ	説明
<b>GoodRetrieveResult</b>	カウンター	例外を発生させないデータ・ソースへの呼び出し側による呼び出しの数。
<b>RetrieveResultResponseTime</b>	タイマー	呼び出し側で測定される、呼び出し側がデータ・ソースに要求を出してから呼び出し側にデータ・ソースの応答が戻るまでの時間。
<b>RetrieveResultWaitTime</b>	タイマー	タイムアウトが発生した場合に、呼び出し側で測定される時間。

表6. ターゲット：以下の統計は、サービスと SCA またはデータ・ソースとの間に発生する要求がある場合に作成されます。

統計名	タイプ	説明
<b>GoodTargetSubmit</b>	カウンター	例外を発生させないサービスへの SCA 呼び出しの数。
<b>BadTargetSubmit</b>	カウンター	例外を発生させるサービスへの SCA 呼び出しの数。
<b>TargetSubmitTime</b>	タイマー	サーバー・サイドで測定される、SCA がサービスに要求を出してからその呼び出しの結果が SCA に戻るまでの時間。
<b>GoodResultSubmit</b>	カウンター	例外を発生させないデータ・ソースへのサービス呼び出しの数。
<b>BadResultSubmit</b>	カウンター	例外を発生させるデータ・ソースへのサービス呼び出しの数。
<b>ResultSubmitTime</b>	タイマー	サーバー・サイドで測定される、サービスがデータ・ソースに要求を出してからその要求の結果がサービスに戻るまでの時間。

表7. コールバック：以下の統計は、呼び出し側にコールバック (元の呼び出しと同種の呼び出し) が存在する場合に作成されます。

統計名	タイプ	説明
<b>GoodCB</b>	カウンター	例外を発生させないコールバックへの SCA 呼び出しの数。
<b>BadCB</b>	カウンター	例外を発生させるコールバックへの SCA 呼び出しの数。
<b>CBTime</b>	タイマー	SCA がコールバックに要求を出してからコールバックからの結果が SCA に戻るまでの時間。
<b>GoodCBSubmit</b>	カウンター	サービスからコールバックを処理する SCA への例外を発生させない呼び出しの数。
<b>BadCBSubmit</b>	カウンター	サービスからコールバックを処理する SCA への例外を発生させる呼び出しの数。
<b>CBSubmitTime</b>	タイマー	サービスがコールバックを処理する SCA に要求を出してから、SCA からの結果がサービスに戻るまでの時間。

## 関連資料

8 ページの『Performance Monitoring Infrastructure 統計』

Performance Monitoring Infrastructure を使用して、3 つのタイプのパフォーマンス統計をモニターできます。つまり、呼び出しの成功回数、失敗回数、およびイベント完了までの経過時間です。これらの統計は、ENTRY、EXIT、および FAILURE のタイプの性質を持つイベントにのみ使用可能です。

## 関連情報



WebSphere Application Server 資料 (Network Deployment)

## 同期呼び出し

ここでは、サービスへの単純な Service Component Architecture (SCA) 呼び出しと、サービスからの応答から取得できるアプリケーション応答測定 (ARM) パフォーマンス統計について説明します。

## パラメーター

SCA コンポーネントのイベント・モニターには、黒で示されるイベント・ポイント



が表示されます。青で示されるイベント・ポイント

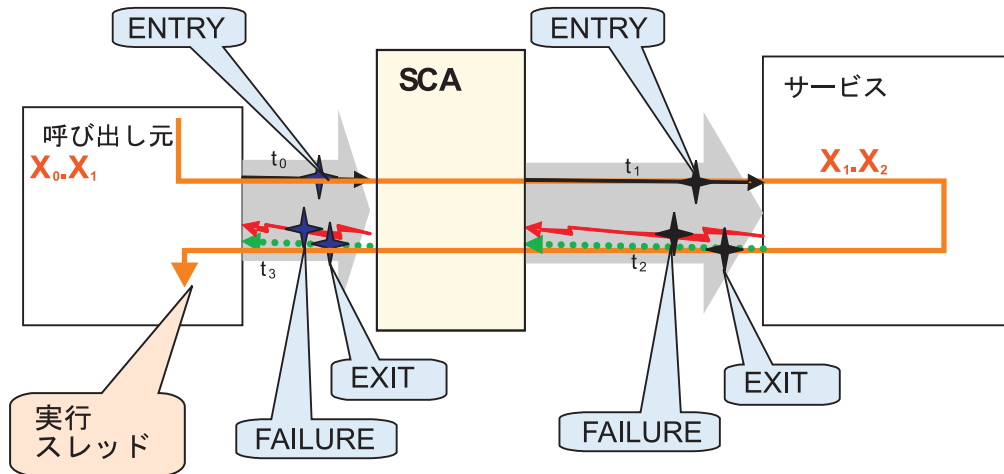


は、PMI/ARM 統計の計算と送出にのみ使用されます。

下記の表と図で、「現行」の ARM トランザクション ( $X_1$  として表記します) は、呼び出しサービス・コンポーネントが初めて呼び出されたときに作成されます。呼び出し元がサービス・コンポーネントではない場合は、現行の ARM トランザクションが使用されるか、新規のトランザクションが作成されます。次の表と図に  $X_n \cdot X_{n+1}$  として表記されているように、これが開始トランザクションでない場合は親が存在します。これらの表記は、トランザクションの系統を記述するために使用されます。すべての SCA 呼び出しは新規トランザクションを開始し、これは、呼び出し元の現行トランザクションが親となります。ユーザーは新規トランザクションを作成でき、現行トランザクションにアクセスできますが、それによって SCA トランザクションの系統が変更されることはありません。

表 8. SCA の同期呼び出しでの ARM 統計

統計	公式	ARM トランザクション
TotalResponseTime	$t_3 - t_0$	$X_0 \cdot X_1$
RequestDeliveryTime	$t_1 - t_0$	$X_1 \cdot X_2$
ResponseDeliveryTime	$t_3 - t_2$	
GoodRequests	Count <sub>EXIT</sub>	
BadRequests	Count <sub>FAILURE</sub>	
ProcessTime	$t_2 - t_1$	




### 同期実装での据え置き応答

要求の同期呼び出しが行われた場合に取得可能なアプリケーション応答測定 (ARM) 統計。戻り結果は、同期実装に対するデータ・ストアへの出力として送信されます。

### パラメーター

Service Component Architecture (SCA) コンポーネントのイベント・モニターには、

黒で示すイベント・ポイント  がありますが、青で示すイベント・ポイント



は、PMI/ARM 統計を計算して送出する場合にのみ使用します。

下記の表と図で、「現行」の ARM トランザクション ( $X_1$  として表記します) は、呼び出しサービス・コンポーネントが初めて呼び出されたときに作成されます。呼び出し元がサービス・コンポーネントではない場合は、現行の ARM トランザクションが使用されるか、新規のトランザクションが作成されます。次の表と図に  $X_n \cdot X_{n+1}$  として表記されているように、これが開始トランザクションでない場合は親が存在します。この表記を使用して、トランザクションの系統が表示されます。すべての SCA 呼び出しは新規トランザクションを開始し、これは、呼び出し元の現行トランザクションが親となります。ユーザーは新規トランザクションを作成したり現行トランザクションにアクセスしたりできますが、SCA トランザクションの系統を変更することはできません。



表 9. 要求の呼び出しと戻り結果

タイプ	統計	公式	ARM トランザクション
共通	TotalResponseTime	$t_3 - t_0$	$X_0 \cdot X_1$
	RequestDeliveryTime	$t'_0 - t_0$	$X_1 \cdot X_2$
	ResponseDeliveryTime	なし	なし
	GoodRequests	$\text{Count}_{\text{EXIT}}$	$X_1 \cdot X_2$
	BadRequests	$\text{Count}_{\text{FAILURE}}$	
	ResponseTime	$t'_1 - t'_0$	
参照 A	GoodRefRequest	$\text{Count}_{\text{EXIT}}$	$X_1 \cdot X_2$
	BadRefRequests	$\text{Count}_{\text{FAILURE}}$	
	RefResponseTime	$t_1 - t_0$	

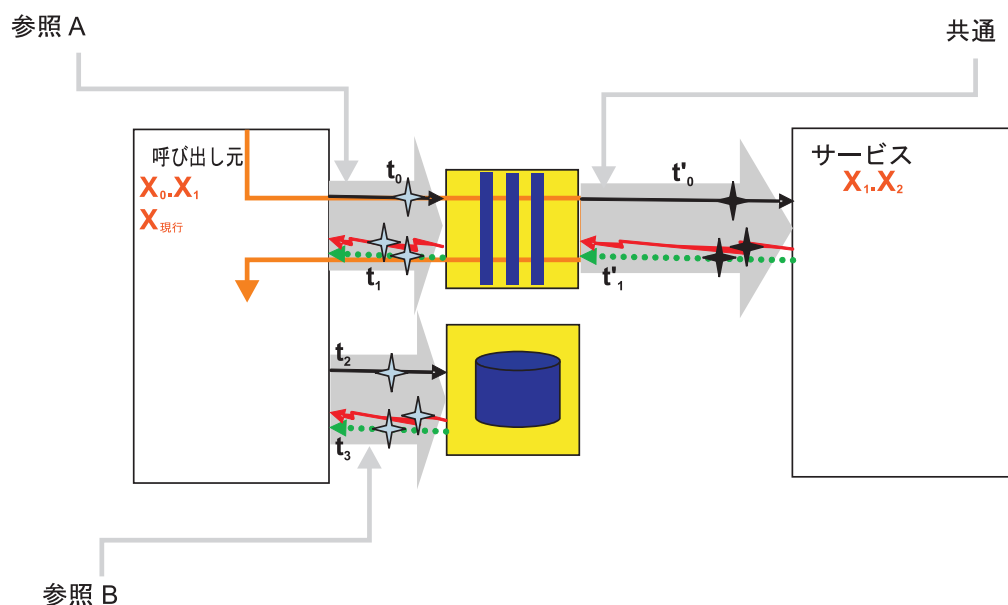


表 10. データ・ソースへの出力の呼び出し


タイプ	統計	公式	ARM トランザクション
参照 B	GoodRetrieveResult	$\text{Count}_{\text{EXIT}}$	$X_1 \cdot X_2$
	BadRetrieveResult	$\text{Count}_{\text{FAILURE}}$	
	ResultRetrieveResponseTime	$\sum t_3 - t_2$	
	ResultRetrieveWaitTime	$\sum \text{timeout}$	

### 非同期実装での据え置き応答

非同期実装で取得可能なアプリケーション応答測定 (ARM) 統計。ここでは、サービスの呼び出しと戻り結果が呼び出され、結果出力がサービスのターゲットからデータ・ストアへ送信される非同期実装からの統計を示します。

## パラメーター

Service Component Architecture (SCA) コンポーネントのイベント・モニターには、

黒で示すイベント・ポイント  がありますが、青で示すイベント・ポイント



は、PMI/ARM 統計を計算して送出する場合にのみ使用します。

下記の表と図で、「現行」の ARM トランザクション ( $X_1$  として表記します) は、呼び出しサービス・コンポーネントが初めて呼び出されたときに作成されます。呼び出し元がサービス・コンポーネントではない場合は、現行の ARM トランザクションが使用されるか、新規のトランザクションが作成されます。次の表と図に  $X_n \cdot X_{n+1}$  として表記されているように、これが開始トランザクションでない場合は親が存在します。この表記を使用して、トランザクションの系統が表示されます。すべての SCA 呼び出しは新規トランザクションを開始し、これは、呼び出し元の現行トランザクションが親となります。ユーザーは新規トランザクションを作成したり現行トランザクションにアクセスしたりできますが、SCA トランザクションの系統を変更することはできません。

表 11. 要求の呼び出しと戻り結果

タイプ	統計	公式	ARM トランザクション
共通	TotalResponseTime	$t_3 - t_0$	$X_0 \cdot X_1$
	RequestDeliveryTime	$t'_0 - t_0$	
	ResponseDeliveryTime	$t'_{03} - t'_2$	
	GoodRequests	Count <sub>EXIT</sub>	
	BadRequests	Count <sub>FAILURE</sub>	
	ResponseTime	$t'_3 - t'_0$	
参照 A	GoodRefRequest	Count <sub>EXIT</sub>	$X_0 \cdot X_1$
	BadRefRequests	Count <sub>FAILURE</sub>	
	RefResponseTime	$t_1 - t_0$	
ターゲット A	GoodTargetSubmit	Count <sub>EXIT</sub>	$X_1 \cdot X_2$
	BadTargetSubmit	Count <sub>FAILURE</sub>	
	TargetSubmitTime	$t'_1 - t'_0$	

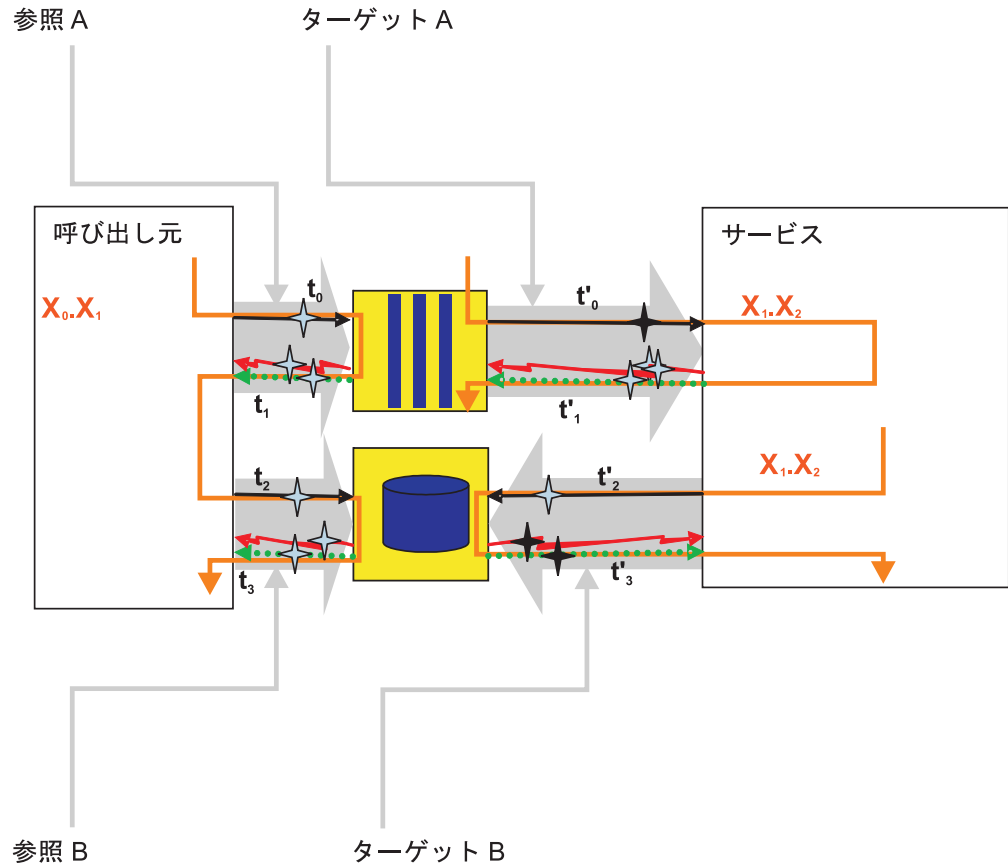


表 12. データ・ストアへの戻り結果の呼び出し


タイプ	統計	公式	ARM トランザクション
参照 B	GoodResultSubmit	$\text{Count}_{\text{EXIT}}$	$X_0 \cdot X_1$
	BadResultSubmit	$\text{Count}_{\text{FAILURE}}$	
	ResultResponseTime	$t'_3 - t'_2$	
ターゲット B	GoodResultRetrieve	$\text{Count}_{\text{EXIT}}$	$X_1 \cdot X_2$
	BadResultRetrieve	$\text{Count}_{\text{FAILURE}}$	
	ResultRetrieveResponseTime	$\sum t_3 - t_2$	
	ResultRetrieveWaitTime	$\sum \text{timeout}$	

### 非同期の結果取得での据え置き応答

アプリケーション応答測定 (ARM) トランザクションを使用して、ResultRetrieve ARM 統計を何らかのオリジナル要求に相関させることができるのは、 $X_{\text{PARENT-1}}$  と  $X_{\text{PARENT-2}}$  が共通の上位トランザクションを持っている場合だけです。要求の呼び出しと結果の取得は、異なるスレッド上で行われます。

## パラメーター

Service Component Architecture (SCA) コンポーネントのイベント・モニターには、

黒で示すイベント・ポイント  がありますが、青で示すイベント・ポイント



は、PMI/ARM 統計を計算して送出する場合にのみ使用します。

下記の表と図で、「現行」の ARM トランザクション ( $X_1$  として表記します) は、呼び出しサービス・コンポーネントが初めて呼び出されたときに作成されます。呼び出し元がサービス・コンポーネントではない場合は、現行の ARM トランザクションが使用されるか、新規のトランザクションが作成されます。開始トランザクションでないトランザクションには、親があります。これは、次の表と図に  $X_n \cdot X_{n+1}$  として表記されています。これらはトランザクションの系統を示すために使用されます。すべての SCA 呼び出しは新規トランザクションを開始し、これは、呼び出し元の現行トランザクションが親となります。ユーザーは新規トランザクションを作成でき、現行トランザクションにアクセスできますが、それによって SCA トランザクションの系統が変更されることはありません。

表 13. 要求の呼び出しと戻り結果

タイプ	統計	公式	ARM トランザクション
共通	TotalResponseTime	$t_3 - t_0$	$X_0 \cdot X_1$
	RequestDeliveryTime	$t'_0 - t_0$	$X_1 \cdot X_2$
	ResponseDeliveryTime	なし	なし
	GoodRequests	$\text{Count}_{\text{EXIT}}$	$X_1 \cdot X_2$
	BadRequests	$\text{Count}_{\text{FAILURE}}$	
	ResponseTime	具体的な図を参照	
参照 A	GoodReferenceRequest	$\text{Count}_{\text{EXIT}}$	$X_1 \cdot X_2$
	BadReferenceRequests	$\text{Count}_{\text{FAILURE}}$	
	ReferenceResponseTime	$t_1 - t_0$	

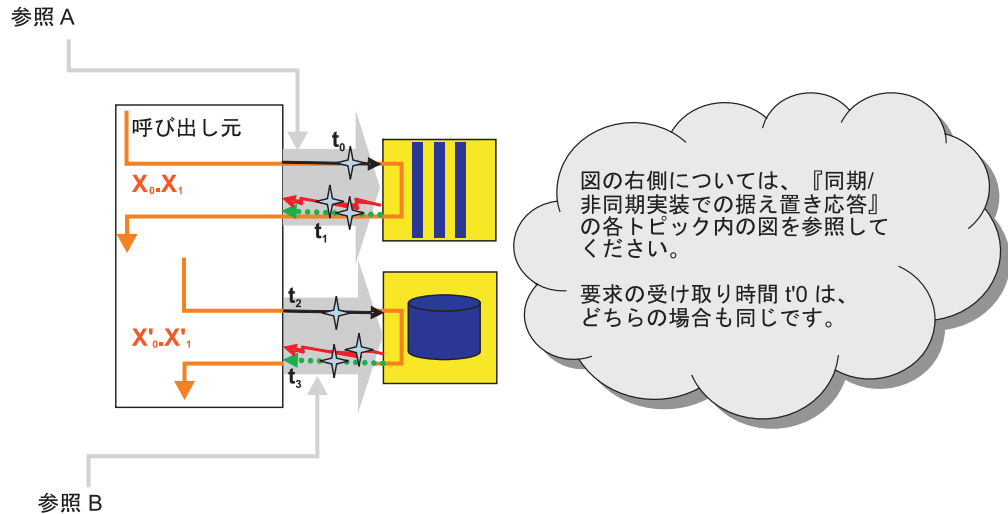


表 14. 要求の呼び出しと戻り結果


タイプ	統計	公式	ARM トランザクション
参照 B	GoodRetrieveResult	$\text{Count}_{\text{EXIT}}$	$X'_0 \cdot X'_1$
	BadRetrieveResult	$\text{Count}_{\text{FAILURE}}$	
	RetrieveResultResponseTime	$\sum t_3 - t_2$	
	RetrieveResultWaitTime	$\sum \text{timeout}$	

## 同期実装での非同期コールバック

同期実装で、コールバック要求とコールバック実行で異なるスレッドを使用する場合に取得可能なアプリケーション応答測定 (ARM) 統計。

### パラメーター

Service Component Architecture (SCA) コンポーネントのイベント・モニターには、

黒で示すイベント・ポイント  がありますが、青で示すイベント・ポイント



は、PMI/ARM 統計を計算して送出する場合にのみ使用します。

下記の表と図で、「現行」の ARM トランザクション ( $X_1$  として表記します) は、呼び出しサービス・コンポーネントが初めて呼び出されたときに作成されます。呼び出し元がサービス・コンポーネントではない場合は、現行の ARM トランザクションが使用されるか、新規のトランザクションが作成されます。次の表と図に  $X_n \cdot X_{n+1}$  として表記されているように、これが開始トランザクションでない場合は親が存在します。この表記を使用して、トランザクションの系統が表示されます。すべての SCA 呼び出しは新規トランザクションを開始し、これは、呼び出し元の現行トランザクションが親となります。ユーザーは新規トランザクションを作成したり現行トランザクションにアクセスしたりできますが、SCA トランザクションの系統を変更することはできません。

表 15. 要求の呼び出しと戻り結果

タイプ	統計	公式	ARM トランザクション
共通	TotalResponseTime	$t_2 - t_0$	$X_0 \cdot X_1$
	RequestDeliveryTime	$t'_0 - t_0$	$X_1 \cdot X_2$
	ResponseDeliveryTime	$t_2 - t'_1$	
	GoodRequests	$\text{Count}_{\text{EXIT}}$	
	BadRequests	$\text{Count}_{\text{FAILURE}}$	
	ResponseTime	$t_3 - t_2$	
参照	GoodRefRequest	$\text{Count}_{\text{EXIT}}$	$X_1 \cdot X_2$
	BadRefRequests	$\text{Count}_{\text{FAILURE}}$	
	RefResponseTime	$t'_1 - t'_0$	

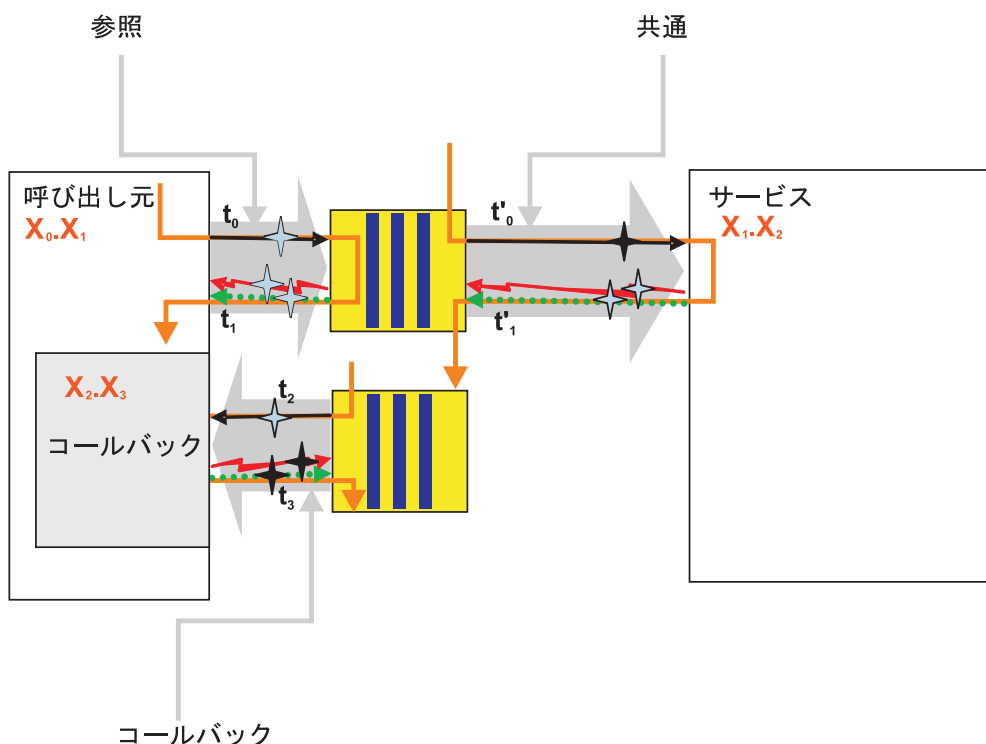


表 16. コールバックの呼び出し


タイプ	統計	公式	ARM トランザクション
コールバック	GoodCB	$\text{Count}_{\text{EXIT}}$	$X_1 \cdot X_3$
	BadCB	$\text{Count}_{\text{FAILURE}}$	
	CBTime	$t_3 - t_2$	

### 非同期実装での非同期コールバック

非同期実装で、異なるスレッドを使用したコールバック要求とコールバック実行について使用可能なアプリケーション応答測定 (ARM) 統計。

## パラメーター

Service Component Architecture (SCA) コンポーネントのイベント・モニターには、

黒で示すイベント・ポイント  がありますが、青で示すイベント・ポイント



は、PMI/ARM 統計を計算して送出する場合にのみ使用します。

下記の表と図で、「現行」の ARM トランザクション ( $X_1$  として表記します) は、呼び出しサービス・コンポーネントが初めて呼び出されたときに作成されます。呼び出し元がサービス・コンポーネントではない場合は、現行の ARM トランザクションが使用されるか、新規のトランザクションが作成されます。次の表と図に  $X_n \cdot X_{n+1}$  として表記されているように、これが開始トランザクションでない場合は親が存在します。この表記を使用して、トランザクションの系統が表示されます。すべての SCA 呼び出しは新規トランザクションを開始し、これは、呼び出し元の現行トランザクションが親となります。ユーザーは新規トランザクションを作成したり現行トランザクションにアクセスしたりできますが、SCA トランザクションの系統を変更することはできません。

表 17. 要求の呼び出しと戻り結果

タイプ	統計	公式	ARM トランザクション
共通	TotalResponseTime	$t_2 - t_0$	$X_0 \cdot X_1$
	RequestDeliveryTime	$t'_0 - t_0$	$X_1 \cdot X_2$
	ResponseDeliveryTime	$t_2 - t'_2$	
	GoodRequests	$\text{Count}_{\text{EXIT}}$	
	BadRequests	$\text{Count}_{\text{FAILURE}}$	
	ResponseTime	$t'_3 - t'_0$	
参照 A	GoodRefRequest	$\text{Count}_{\text{EXIT}}$	$X_0 \cdot X_1$
	BadRefRequests	$\text{Count}_{\text{FAILURE}}$	
	RefResponseTime	$t_1 - t_0$	
ターゲット A	GoodTargetSubmit	$\text{Count}_{\text{EXIT}}$	$X_1 \cdot X_2$
	BadTargetSubmit	$\text{Count}_{\text{FAILURE}}$	
	TargetSubmitTime	$t'_1 - t'_0$	

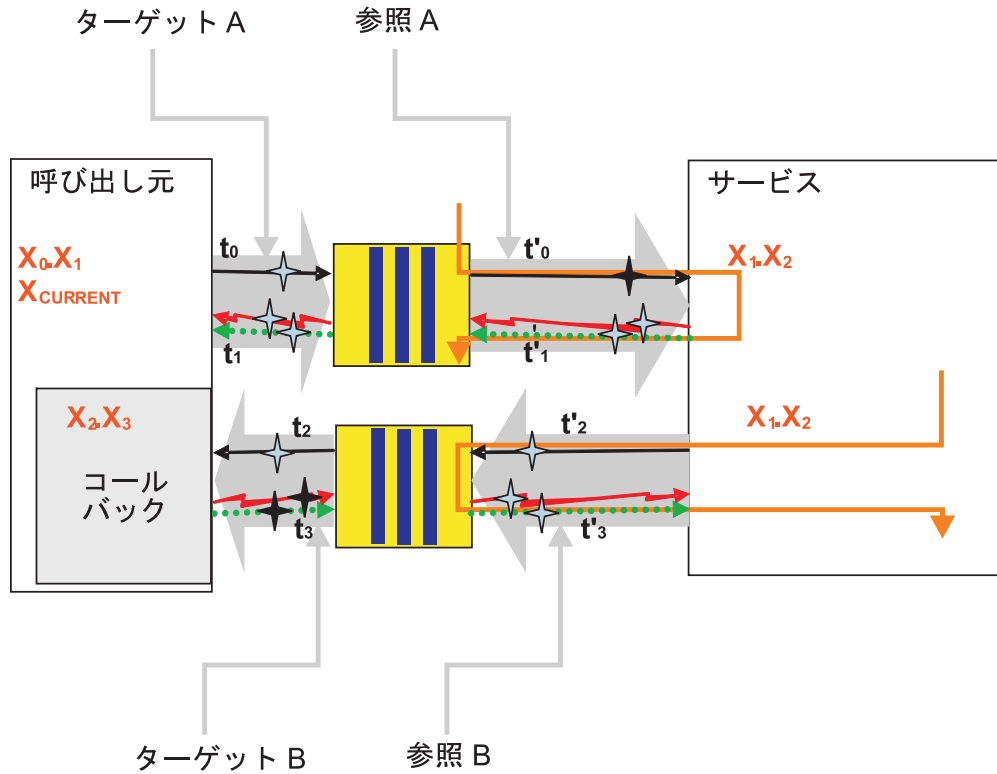


表 18. コールバックの呼び出し


タイプ	統計	公式	ARM トランザクション
参照 B	GoodCBSubmit	$\text{Count}_{\text{EXIT}}$	$X_1 \cdot X_2$
	BadCBSubmit	$\text{Count}_{\text{FAILURE}}$	
	CBSubmitTime	$t'_3 - t'_2$	
ターゲット B	GoodCB	$\text{Count}_{\text{EXIT}}$	$X_0 \cdot X_1$
	BadCB	$\text{Count}_{\text{FAILURE}}$	
	CBTime	$t_3 - t_2$	

### 同期実装環境での非同期片方向呼び出し

同期実装で、呼び出しが実行依頼（応答不要送信）された場合に取得可能なアプリケーション応答測定（ARM）統計。

### パラメーター

Service Component Architecture (SCA) コンポーネントのイベント・モニターには、

黒で示すイベント・ポイント  がありますが、青で示すイベント・ポイント



は、PMI/ARM 統計を計算して送出する場合にのみ使用します。

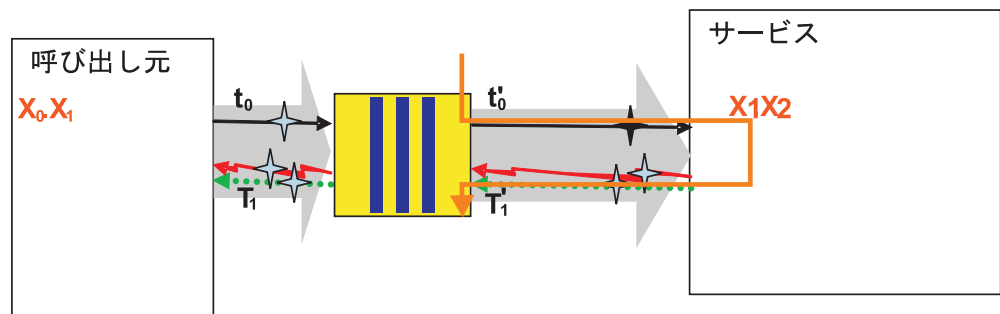
下記の表と図で、「現行」の ARM トランザクション ( $X_1$  として表記します) は、呼び出しサービス・コンポーネントが初めて呼び出されたときに作成されます。呼



呼び出し元がサービス・コンポーネントではない場合は、現行の ARM トランザクションが使用されるか、新規のトランザクションが作成されます。次の表と図に  $X_n \cdot X_{n+1}$  として表記されているように、これが開始トランザクションでない場合は親が存在します。この表記を使用して、トランザクションの系統が表示されます。すべての SCA 呼び出しは新規トランザクションを開始し、これは、呼び出し元の現行トランザクションが親となります。ユーザーは新規トランザクションを作成したり現行トランザクションにアクセスしたりできますが、SCA トランザクションの系統を変更することはできません。

表 19. 要求の呼び出しと戻り結果

タイプ	統計	公式	ARM トランザクション
共通	TotalResponseTime	$t_1 - t_0$	$X_0 \cdot X_1$
	RequestDeliveryTime	$t'_0 - t_0$	$X_1 \cdot X_2$
	ResponseDeliveryTime	なし	なし
	GoodRequests	$\text{Count}_{\text{EXIT}}$	$X_1 \cdot X_2$
	BadRequests	$\text{Count}_{\text{FAILURE}}$	
	ResponseTime	$t'_1 - t'_0$	




### 非同期実装環境での非同期片方向呼び出し

非同期実装で、呼び出しが実行依頼（応答不要送信）された場合のアプリケーション 応答測定（ARM）統計。

#### パラメーター

Service Component Architecture (SCA) コンポーネントのイベント・モニターには、

黒で示すイベント・ポイント  がありますが、青で示すイベント・ポイント



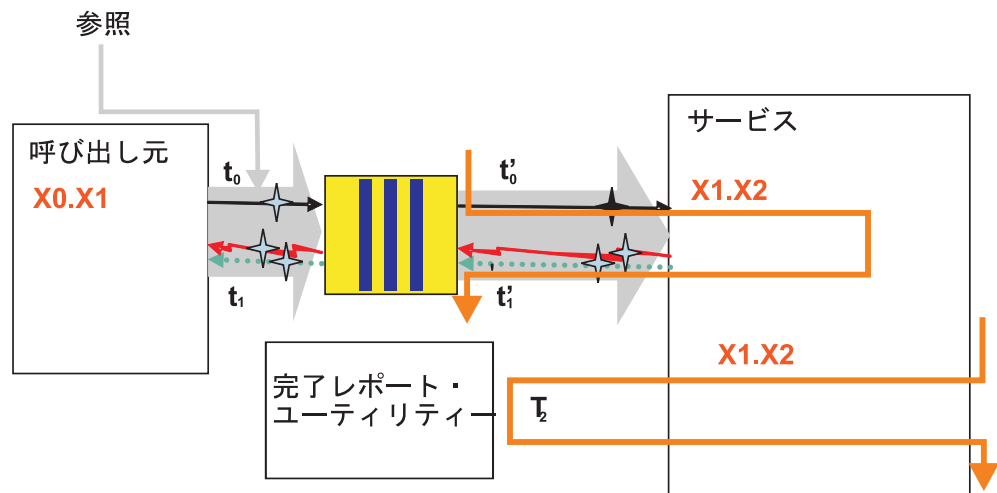
は、PMI/ARM 統計を計算して送出する場合にのみ使用します。

下記の表と図で、「現行」の ARM トランザクション ( $X_1$  として表記します) は、呼び出しサービス・コンポーネントが初めて呼び出されたときに作成されます。呼び出し元がサービス・コンポーネントではない場合は、現行の ARM トランザクションが使用されるか、新規のトランザクションが作成されます。開始トランザクションでないトランザクションには、親があります。この親子関係は、次の表と図に

$X_n \cdot X_{n+1}$  として表記されています。この表記を使用して、トランザクションの系統が表示されます。すべての SCA 呼び出しは新規トランザクションを開始し、これは、呼び出し元の現行トランザクションが親となります。ユーザーは新規トランザクションを作成したり現行トランザクションにアクセスしたりできますが、SCA トランザクションの系統を変更することはできません。

表 20. 要求の呼び出しと戻り結果

タイプ	統計	公式	ARM トランザクション
共通	TotalResponseTime	$t_1 - t_0$	$X_0 \cdot X_1$
	RequestDeliveryTime	$t'_0 - t_0$	$X_1 \cdot X_2$
	ResponseDeliveryTime	なし	なし
	GoodRequests	Count <sub>EXIT</sub>	$X_1 \cdot X_2$
	BadRequests	Count <sub>FAILURE</sub>	
	ResponseTime	$t_2 - t_0$	
参照	GoodRefRequest	Count <sub>EXIT</sub>	$X_0 \cdot X_1$
	BadRefRequest	Count <sub>FAILURE</sub>	
	RefResponseDuration	$t_1 - t_0$	



## サービス・コンポーネント・イベントのモニター

WebSphere Process Server モニターでは、特定のイベント・ポイントでサービス・コンポーネントのデータを取り込むことができます。ログ・ファイル内の個々のイベントを表示するか、もっと多用途の Common Event Infrastructure サーバーのモニター機能を使用することができます。

プロセス・サーバーにデプロイされているアプリケーションには、アプリケーションが実行されている限りモニターされるサービス・コンポーネント・イベントの仕様が含まれています。WebSphere Integration Developer を使用してアプリケーションを開発した場合、サービス・コンポーネント・イベントを継続的にモニターするように指定できます。この仕様はアプリケーションの一部として組み込まれており、アプリケーションのデプロイ時にプロセス・サーバーによって読み取られる .mon

拡張子を持つファイルの形式で提供されます。アプリケーションを開始したら、.mon ファイルに指定されているサービス・コンポーネントのモニターをオフにすることはできません。WebSphere Process Server の資料では、このタイプの継続モニターについては扱っていません。この件についての詳細は、WebSphere Integration Developer の資料を参照してください。

WebSphere Process Server を使用して、アプリケーションの .mon ファイルに指定されていないサービス・コンポーネント・イベントをモニターすることができます。プロセス・サーバーでイベント・モニターの出力をログ・ファイルや Common Event Infrastructure サーバー・データベースに送信するように構成することができます。モニター対象イベントは、Common Base Event 標準を使用してフォーマット設定されますが、各イベントで保持する情報の量を規制することができます。WebSphere Process Server のモニター機能を使用して、問題の診断、アプリケーションのプロセス・フローの分析、またはアプリケーションの使用法の監査を行うことができます。

## ビジネス・プロセス・イベントとヒューマン・タスク・イベントのモニターの使用可能化

ビジネス・プロセスおよびヒューマン・タスクのサービス・コンポーネントのモニターをサポートするには、それらのサービス・コンポーネント種類のモニターを実際に開始する前に、WebSphere Process Server を構成する必要があります。

### 始める前に

ビジネス・プロセス・コンテナとヒューマン・タスク・コンテナがプロセス・サーバー上に事前に作成されている必要があります。

### このタスクについて

以下のタスクを実行して、WebSphere Process Server での Common Event Infrastructure モニターのサポートを使用可能に設定します。

### 手順

1. 管理コンソールを開きます。
2. Business Process Choreographer が単一サーバー上に構成されている場合は、以下のステップを実行して、サーバーがビジネス・プロセス・イベントを生成できるようにします。
  - a. Human Task Manager に対してビジネス・プロセス・イベントを使用可能にする場合、「サーバー」 → 「サーバー・タイプ」 → 「WebSphere Application Server」 → 「server\_name」 をクリックし、「ビジネス・インテグレーション」の「構成」タブで「Business Process Choreographer」を展開し、「Human Task Manager」をクリックします。「状態監視」セクションで、「Common Event Infrastructure のロギングを使用可能に設定」、「監査ロギングを使用可能に設定」、「タスク履歴を使用可能にする」の各チェック・ボックスが選択されていることを確認します。これらのチェック・ボックスが選択されていない場合は、これらを選択し、サーバーを再始動する必要があります。

- b. ビジネス・フロー・マネージャーに対してビジネス・プロセス・イベントを使用可能にする場合、「サーバー」 → 「サーバー・タイプ」 → 「WebSphere Application Server」 → *server\_name*をクリックし、「ビジネス・インテグレーション」の「構成」タブで「Business Process Choreographer」を展開し、「Business Flow Manager」をクリックします。「状態監視」セクションで、「Common Event Infrastructure のロギングを使用可能に設定」と「監査ロギングを使用可能に設定」の各チェック・ボックスが選択されていることを確認します。selected. これらのチェック・ボックスが選択されていない場合は、これらを選択し、サーバーを再始動する必要があります。
3. Business Process Choreographer がクラスター上に構成されている場合は、以下のステップを実行して、クラスターがビジネス・プロセス・イベントを生成できるようにします。
  - a. Human Task Manager に対してビジネス・プロセス・イベントを使用可能にする場合、「サーバー」 → 「クラスター」 → 「WebSphere Application Server クラスター (WebSphere application server clusters)」 → 「*cluster\_name*」をクリックし、「ビジネス・インテグレーション」の「構成」タブで「Business Process Choreographer」を展開し、「Common Event Infrastructure のロギングを使用可能に設定」、「監査ロギングを使用可能に設定」、「タスク履歴を使用可能にする」の各チェック・ボックスが選択されていることを確認します。これらのチェック・ボックスが選択されていない場合は、これらを選択し、サーバーを再始動する必要があります。
  - b. Business Flow Manager に対してビジネス・プロセス・イベントを使用可能にする場合、「サーバー」 → 「クラスター」 → 「WebSphere Application Server クラスター (WebSphere application server clusters)」 → 「*cluster\_name*」をクリックし、「ビジネス・インテグレーション」の「構成」タブで「Business Process Choreographer」を展開し、「Business Flow Manager」をクリックします。「状態監視」セクションで、「Common Event Infrastructure のロギングを使用可能に設定」と「監査ロギングを使用可能に設定」の各チェック・ボックスが選択されていることを確認します。selected. これらのチェック・ボックスが選択されていない場合は、これらを選択し、サーバーを再始動する必要があります。

## 次のタスク

これらのボックスのいずれかを選択する必要があった場合は、サーバーまたはクラスターを再始動して、変更を有効にする必要があります。

## サービス・コンポーネント・イベントのロギングの構成

プロセス・サーバー・モニターによって送出されるサービス・コンポーネント・イベントを収集するには、WebSphere Application Server のロギング機能を使用する方法を選択できます。アプリケーションの処理に関する問題を診断する場合は、ロガーを使用してイベント内のデータを表示します。

WebSphere Process Server では、基盤となる WebSphere Application Server の拡張ロギング機能を使用して、サービス・コンポーネント・イベント・ポイントでサーバー・モニターによって送出されるイベントを収集できます。管理コンソールを使用して、モニターする特定のサービス・コンポーネント・イベント・ポイント、結

果のサービス・コンポーネント・イベントに含まれる有効搭載量の詳細、および結果をパブリッシュする方法 (特定形式のファイルにパブリッシュしたり、コンソールに直接パブリッシュしたりするなど) を指定できます。モニター・ログには Common Base Event 形式でエンコードされたイベントが記録されており、イベント・エレメントに含まれている情報を使用して、サービス・コンポーネントの処理に関する問題をトレースすることができます。

WebSphere Application Server ログ機能とトレース機能については、WebSphere Application Server の資料で詳細に文書化されており、製品全体でのログ機能およびトレースの使用法についても詳細に説明されています。このセクションでは、WebSphere Process Server に固有のサービス・コンポーネントに関連するログ機能について、補足情報のみを提供します。製品全体の他のコンポーネントでログ機能およびトレースを使用する場合については、WebSphere Application Server の資料を参照してください。

## 診断トレース・サービスの使用可能化

このタスクを使用して、診断トレース・サービスを使用可能に設定します。このサービスは、サービス・コンポーネント・イベントに含む詳細情報の量を管理できるログ機能・サービスです。

### 始める前に

Common Event Infrastructure (CEI) ログ機能と監査ログ機能を許可するには、ビジネス・プロセス・コンテナとヒューマン・タスク・コンテナを構成しておく必要があります。

### このタスクについて

診断トレース・サービスは、サービス・コンポーネント・イベントのエレメントに含まれる詳細情報を取り込むために必要な詳細レベルを指定可能な唯一のロガー・タイプです。イベントをログに記録するには、プロセス・サーバーを始動する前に、診断トレース・サービスを使用可能にする必要があります。また、管理コンソールを使用して、CEI サーバーを使用してモニターするサービス・コンポーネント・イベント・ポイントを選択する場合も、このサービスを使用可能にする必要があります。

### 手順

1. ナビゲーション・ペインで、「サーバー」 → 「サーバー・タイプ」 → 「WebSphere Application Server」をクリックします。
2. 作業で使用するサーバーの名前をクリックします。
3. 「トラブルシューティング」の下で、「診断トレース・サービス (Diagnostic Trace service)」をクリックします。
4. 「構成」タブの「ログの使用可能化 (Enable log)」を選択します。
5. 「適用」をクリックし、次に「保管」をクリックします。
6. 「OK」をクリックします。

## 次のタスク

サーバーが既に始動している場合は、再始動して変更内容を有効にする必要があります。

### 管理コンソールを使用したロギング・プロパティの構成

このタスクを使用して、モニター機能がサービス・コンポーネント・イベントをロガー・ファイルにパブリッシュするように指定できます。

#### このタスクについて

アプリケーションがモニター済みイベントをログに記録できるようにするためには、その前に、モニターするサービス・コンポーネント・イベント・ポイント、各イベントに要求する詳細のレベル、およびイベントをログにパブリッシュするために使用する出力の形式を指定する必要があります。管理コンソールを使用して、次の操作を実行できます。

- 特定のイベント・ログを使用可能または使用不可にする。
- ログの詳細レベルを指定する。
- ログ・ファイルの保管場所、保持するログ・ファイルの数、およびログ出力の形式を指定する。

ログ構成は、静的または動的に変更することができます。静的構成変更は、アプリケーション・サーバーの始動または再始動時にアプリケーションに反映されます。動的構成変更つまりランタイムの構成変更は、即座に適用されます。

ログの作成時、構成データに基づいてログのレベル値が設定されます。特定のログ名に対応する構成データがない場合、ログ・レベルの値はそのログの親から取得されます。親ログの構成データが存在しない場合は、さらにその親のログがチェックされ、ヌル以外のレベル値を持つログが検出されるまでツリーをさかのぼって同じ操作が実行されます。ログのレベルを変更すると、その変更はログの子に伝搬され、必要に応じてさらにその子に伝搬されます。

#### 手順

1. ロギングを使用可能にし、ログの出力プロパティを設定します。
2. ナビゲーション・ペインで、「サーバー」>「サーバー・タイプ」>「WebSphere Application Server」をクリックします。
3. 作業で使用するサーバーの名前をクリックします。
4. 「トラブルシューティング」の下で、「ロギングおよびトレース (Logging and tracing)」をクリックします。
5. 「ログ詳細レベルの変更 (Change Log Detail levels)」をクリックします。
6. コンポーネント、パッケージ、およびグループのリストに、実行中のサーバー上で現在登録されているすべてのコンポーネントが表示されます。このリストには、呼び出し回数が 1 回以上のサーバー・イベントのみが表示されます。ログに記録可能なイベント・ポイントを保持したすべてのサーバー・コンポーネントが **WBILocationMonitor.LOG** という名前が始まるコンポーネントのいずれかの下にリストされます。
  - 構成を静的に変更するイベントを選択するには、「構成」タブをクリックします。

- 構成を動的に変更するイベントを選択するには、「ランタイム (Runtime)」タブをクリックします。
7. ログに記録するイベントまたはイベント・グループを選択します。
  8. イベントまたはイベント・グループごとにロギング・レベルを設定します。

注: CEI イベントのロギングでは、FINE、FINER、および FINEST レベルのみが有効です。

9. 「適用」をクリックします。
10. 「OK」をクリックします。
11. 静的な構成変更を有効にするには、サーバーを停止してから再始動します。

## タスクの結果

デフォルトでは、ロガーは `trace.log` というファイルに出力をパブリッシュします。このファイルは `install_root/profiles/profile_name/logs/server_name` フォルダにあります。

## チュートリアル: サービス・コンポーネント・イベントのロギング

モニター対象のサービス・コンポーネント・イベント・ポイントでは、イベントを基礎となる WebSphere Application Server のロギング機能にパブリッシュできます。このチュートリアルでは、ロギングを使用したモニターの設定例と、ログ・ファイルに格納されているイベントを表示する方法について説明します。

この例のシナリオにより、サーバーにすでにデプロイされ稼働しているアプリケーションで、モニターするサービス・コンポーネント・イベント・ポイントを選択する方法を示します。ここでは、アプリケーションの処理がそれらのイベント・ポイントのいずれかに到達した場合常にモニター機能がイベントを送出する仕組みについて知ることができます。送られる各イベントは標準化された Common Base Event 形式をとり、XML スtringとしてログ・ファイルに直接パブリッシュされます。

## このチュートリアルの目的

このチュートリアルを完了すれば、次の操作を実行できるようになります。

- モニターするサービス・コンポーネント・イベント・ポイントおよびサーバーのロガーにパブリッシュされる出力を選択する。
- ログ・ファイルに保管されているイベントを表示する。

## このチュートリアルを完了するのに必要な時間

このチュートリアルを完了するには、およそ 15 分から 20 分かかります。

## 前提条件

このチュートリアルを実行するには、次の条件を満たす必要があります。

- サーバーを構成および始動済みである。
- Common Event Infrastructure を構成済みである。
- サーバーの診断トレース・サービスが使用可能に設定されている。

- サーバーでサンプル・ギャラリー・アプリケーションがインストールおよび始動済みである。
- サーバーでビジネス・ルール・サンプル・アプリケーションがインストールおよび始動済みである。「サンプル・ギャラリー」ページの指示に従って、ビジネス・ルール・サンプル・アプリケーションをセットアップし、実行します。

これらのすべての前提条件が満たされたら、チュートリアルに進む前に、少なくとも一度サンプル・ギャラリーからビジネス・ルール・サンプル・アプリケーションを実行してください。

#### 例: ロガーによるイベントのモニター:

ロギングによるモニターでは、管理コンソールを使用して、イベント・タイプの詳細を管理できます。この例では、コンソールを使用して一部のイベント・タイプで記録された詳細のレベルを変更する方法や、テキスト・エディターを使用して `trace.log` ファイルを開き、個々のイベントの情報を表示する方法を示します。

#### このタスクについて

このシナリオではビジネス・ルール・サンプル・アプリケーションを使用するため、このアプリケーションが配置されている Web ページをあらかじめ開いてください。この Web ページは開いたままにしておいてください。モニター・パラメーターを指定した後、このサンプルを実行します。サンプルがモニター対象として選択可能な機能リストに表示されるように、少なくとも一度実行されていることを確認してください。

#### 手順

1. 管理コンソールを開きます。
2. ナビゲーション・ペインで、「サーバー」 → 「アプリケーション・サーバー」をクリックします。
3. `server_name` をクリックします。
4. 「トラブルシューティング」の下で、「ロギングおよびトレース (Logging and tracing)」をクリックします。
5. 「ログ詳細レベルの変更 (Change Log Detail levels)」をクリックします。
6. 「ランタイム (Runtime)」タブを選択します。
7. **WBILocationMonitor.LOG.BR** のツリーを展開します。  
**WBILocationMonitor.LOG.BR.brsample.\*** エレメントの下に 7 つのイベント・タイプが表示されます。最初のイベントは **WBILocationMonitor.LOG.BR.brsample\_module.DiscountRuleGroup** と呼ばれ、次の性質を持つ **Operation.\_calculateDiscount** という名前の 1 つの関数が含まれています。
  - ENTRY
  - EXIT
  - FAILURE
  - SelectionKeyExtracted
  - TargetFound
8. 各イベントをクリックして 「**finest**」 を選択します。



9. 「OK」をクリックします。
10. 「ビジネス・ルール・サンプル・アプリケーション (business rules sample application)」ページに切り替え、アプリケーションを一度実行します。
11. テキスト・エディターを使用して、システムの `profile_root/logs/server_name` フォルダーに格納されている `trace.log` ファイルを開きます。

### タスクの結果

サンプル・アプリケーションを実行したときにモニターによって送出されたビジネス・ルール・イベントを含むログ内の行が表示されます。これを見て気付く大きな点は、出力が Common Base Event 標準に準拠した長い未解析 XML ストリングで構成されていることです。ENTRY イベントと EXIT イベントを調べると、ビジネス・オブジェクトが 16 進形式でエンコードされていることがわかります (詳細レベルの「finest」を選択したため、このビジネス・オブジェクトが記述されています)。この出力を、Common Event Infrastructure サーバーにパブリッシュされたイベントと比較してください。このサーバーは、XML を読み取り可能な表に解析し、ビジネス・オブジェクト・データを読み取り可能な形式にデコードします。この演習内の前のステップに戻り、詳細レベルを「finest」から「fine」または「finer」に変更して、イベント間の違いを比較することもできます。

この演習を完了すれば、ロガーを使用してモニターするサービス・コンポーネント・イベント・ポイントを選択する方法を理解したことになります。このタイプのモニターで送出されるイベントは標準形式で、結果は未加工の XML 形式のストリングとしてログ・ファイルに直接パブリッシュされることがわかりました。パブリッシュされたイベントを表示するには、テキスト・エディターでログ・ファイルを開き、個別のイベントの内容を復号します。

### 次のタスク

ビジネス・ルール・サンプル・アプリケーションのモニターを終了する場合は、ここで概要を示したステップを逆戻りして、サンプル・イベントの詳細レベルを「info」にリセットしてください。

## ビジネス・ルールおよびセレクトターの監査ロギング

WebSphere Process Server で、ビジネス・ルールやセレクトターへの変更が自動的にログに記録されるようにセットアップすることができます。

サーバーで自動的にビジネス・ルールやセレクトターへの変更を検出したり、変更の詳細を説明するログ・ファイルにエントリーを作成するように構成することができます。

標準 JVM SystemOut.log ファイルと指定したカスタム監査ログ・ファイルのどちらかにログ・エントリーを書き込むかを選択できます。変更方法に応じて、各ビジネス・ルールやセレクトターを変更したプロセス・サーバーでは以下の内容がログに記録されます。

- 変更を行ったユーザーの名前
- 変更要求が出された場所
- 旧ビジネス・ルール・オブジェクトまたはセレクトター・オブジェクト
- 旧オブジェクトから置き換えられた新規ビジネス・ルールまたはセレクトター

ビジネス・ルール・オブジェクトおよびセレクター・オブジェクトは、置き換えられたビジネス・ルールまたはセレクターの場合でも、置き換えた新規バージョンの場合でも、完全なビジネス・ルール・セット、デシジョン・テーブル、ビジネス・ルール・グループ、またはセレクターです。ログを調べて (監査出力は Common Event Infrastructure データベースに出力できません)、旧/新ビジネス・ルールまたはセレクターを比較して変更箇所を判別できます。以下のシナリオで、ロギングが発生する状況 (ロギングが構成されている場合) と、ログ・エントリーの内容について説明します。

シナリオ	結果	ログ・エントリーの内容
ビジネス・ルール・マネージャーを使用したビジネス・ルールのパブリッシュ	要求	ユーザー ID、サーバー名 (該当する場合、セルとノードを含む)、旧ビジネス・ルール・ルールセット、新規ルールセット。
	失敗	ユーザー ID、サーバー名 (該当する場合、セルとノードを含む)、旧ビジネス・ルール・ルールセット、新規ルールセット。
リポジトリ・データベースの更新とコミット (ビジネス・ルール・マネージャーを使用したパブリッシュから)	成功	ユーザー ID、旧ルールセット、新規ルールセット。
	失敗	ユーザー ID、新規ルールセット。
セレクター・グループまたはビジネス・ルール・グループのエクスポート	要求	ユーザー ID、セレクター、またはビジネス・ルール・グループ名。
	成功	ユーザー ID、サーバー名 (該当する場合、セルとノードを含む)、エクスポートされたセレクター・グループまたはビジネス・ルール・グループのコピー。
	失敗	ユーザー ID、サーバー名 (該当する場合、セルとノードを含む)、セレクター・グループまたはビジネス・ルール・グループ名。

シナリオ	結果	ログ・エントリーの内容
セレクター・グループまたはビジネス・ルール・グループのインポート	要求	ユーザー ID、新規セレクター・グループまたはビジネス・ルール・グループのコピー。
	成功	ユーザー ID、サーバー名 (該当する場合、セルとノードを含む)、インポートされたセレクター・グループまたはビジネス・ルール・グループのコピー、インポートされたバージョンに置き換えられたセレクター・グループまたはビジネス・ルール・グループのコピー。
	失敗	ユーザー ID、サーバー名 (該当する場合、セルとノードを含む)、インポートされる予定だったセレクター・グループまたはビジネス・ルール・グループのコピー。
アプリケーションのインストール	成功	ユーザー ID、サーバー名 (該当する場合、セルとノードを含む)、セレクター・グループまたはビジネス・ルール・グループ名。
	失敗	ユーザー ID、サーバー名 (該当する場合、セルとノードを含む)、セレクター・グループまたはビジネス・ルール・グループ名。
アプリケーションの更新 (管理コンソールまたは wsadmin コマンドから)	成功	ユーザー ID、サーバー名 (該当する場合、セルとノードを含む)、新規セレクター・グループまたはビジネス・ルール・グループのコピー、旧セレクター・グループまたはビジネス・ルール・グループのコピー。
	失敗	ユーザー ID、サーバー名 (該当する場合、セルとノードを含む)、新規セレクター・グループまたはビジネス・ルール・グループのコピー。

シナリオ	結果	ログ・エントリーの内容
既存のビジネス・ルールかセレクター、またはその両方が開始されている以前にアプリケーションをデプロイ	成功	サーバー名 (該当する場合、セルとノードを含む)、セレクター・グループまたはビジネス・ルール・グループのコピー。
	失敗	サーバー名 (該当する場合、セルとノードを含む)、セレクター・グループまたはビジネス・ルール・グループのコピー。

## Common Event Infrastructure サーバーでのサービス・コンポーネントのモニター

サービス・コンポーネントのモニター結果を Common Event Infrastructure (CEI) サーバーにパブリッシュするように選択することができます。サービス・コンポーネント・イベント・ポイントを CEI サーバーによるモニター対象として指定できます。モニターは、アプリケーション・フローの表示および管理を行う永続ベースか、問題のトラブルシューティングを行う一時ベースかのどちらかです。

モニターを使用すると、CEI バスを介して送出されるサービス・コンポーネント・イベント内部のサービス・コンポーネント・イベント・ポイントでデータを公開できます。このモニター方法によって、システムでのサービス・コンポーネントのアクティビティの分析をより柔軟に行うことができます。また、Common Base Event ブラウザーなどの CEI イベント用に最適化されたブラウザーを使用することもできます。

イベントは、ロガーに送信されるイベントと一致した内容で構造化されていますが、サービス・コンポーネント・イベントの分析のために特別に設計されたビューアーがアクセスできるように、データベースに保管されます。アプリケーションの作成時にサービス・コンポーネント・イベント・ポイントをアプリケーション内に指定して、アプリケーションがデプロイされ、サーバー上で稼働するようになった後にモニターが常時継続的に行われるようにします。これは「静的」モニターと呼ばれる方法です。システムでのコンポーネント処理のフローを適切にするために特に重要なサービス・コンポーネント・イベント・ポイントについて、静的モニターを実行してください。この情報により、システムで実行されているサービス・コンポーネント・プロセスのアクションおよびプロセス間の対話を、容易に監督することができます。また、これらのプロセスの通常フローからの逸脱を素早く検出することもできます。この場合は、サービス・コンポーネントが正常に作動していない可能性があります。

サービス・コンポーネントの静的モニターを構成するには、WebSphere Integration Developer を使用して、アプリケーションのサービス・コンポーネント・イベント・ポイントを選択します。選択されたサービス・コンポーネント・イベント・ポイントは、アプリケーションとともにデプロイされる、.mon という拡張子を持つ XML ファイルの形式で指定されます。稼働中のサーバーにデプロイした後は、アプリケーションの .mon ファイルに指定されているイベント・モニターの詳細レベルをオ

フにしたり、レベルを下げたりすることはできません。このモニターを停止するには、サーバーを停止し、アプリケーションをアンデプロイする必要があります。

また、「動的」モニターのサービス・コンポーネント・イベント・ポイントも選択することができます。これは、実行中のサーバーに既にデプロイされているアプリケーション上で使用可能にしたり使用不可にしたりすることができます。CEI サーバーを使用して動的モニターを実行する仕組みは、ロギングによってシステムの問題を診断およびトラブルシューティングする場合と基本的に同じです。出力はログャーにパブリッシュされた出力と基本的に同じです。CEI バスを通して送出されるイベントごとに構造を構成する Common Base Event エlementを使用します。また、ロギング・データと同様、詳細レベルの違いはイベント内でエンコードされる有効搭載量にのみ影響します。

## 管理コンソールを使用したサービス・コンポーネント・イベント・モニターの構成

管理コンソールを使用して、モニター機能がサービス・コンポーネント・イベントを Common Event Infrastructure サーバーにパブリッシュするように動的に指定することができます。

### 始める前に

ログャーの場合と同様に、診断トレース・サービスを使用可能にする必要があります。サーバーの再始動後、モニターするイベントを 1 回呼び出し、モニター可能なイベントのリストに表示されるようにします。

### このタスクについて

モニター対象のイベントを選択する方法は、プロセス・サーバーにデプロイ済みのアプリケーションに対して使用します。アプリケーションとともにプロセス・サーバーにデプロイされた .mon ファイルに指定されているイベントは、ここでの変更に関係なく、Common Event Infrastructure (CEI) データベースによってモニターされます。このようなイベントの場合は、CEI データベースに収集およびパブリッシュするための詳細レベルの値を大きくするだけで済みます。CEI データベースにパブリッシュされる出力は、ログャーによってパブリッシュされる出力に非常によく似ています。

### 手順

1. 管理コンソールから、「トラブルシューティング」>「ロギングおよびトレース」をクリックします。
2. 「ログ詳細レベルの変更 (Change Log Detail levels)」をクリックします。
3. コンポーネント、パッケージ、およびグループのリストに、稼働中のサーバーに現在登録されているすべてのコンポーネントが表示されます。少なくとも 1 回呼び出されたプロセス・サーバー・イベントのみがこのリストに表示されます。ログ可能なすべてのプロセス・サーバー・イベントは、**WBILocationMonitor.CEI** という名前が始まるいずれかのコンポーネントの下にリストされます。
  - 構成を静的に変更するには、「構成」タブをクリックします。
  - 構成を動的に変更するには、「ランタイム (Runtime)」タブをクリックします。

4. モニターするイベントまたはイベント・グループを選択します。
5. イベントごとに収集する情報の詳細レベルをクリックします。

注: CEI イベントの場合、FINE、FINER、および FINEST レベルのみが有効です。

6. 「適用」をクリックし、次に「保管」をクリックします。
7. 「OK」をクリックします。
8. 構成を静的に変更した場合は、変更を有効にするためにプロセス・サーバーを再始動する必要があります。

## タスクの結果

モニター対象イベントの結果は、Common Base Event Browser で表示できます。

## チュートリアル: イベント・モニターでの Common Event Infrastructure サーバーの使用

このチュートリアルでは、CEI サーバーを使用したモニターの設定例と、データベースに格納されているイベントを表示する方法について説明します。

モニター対象のサービス・コンポーネント・イベント・ポイントでは、イベントを Common Event Infrastructure (CEI) サーバーに公開して CEI サーバー・データベースに格納することができます。イベントが収集された後、格納されたそれらのイベントを表示するには、Common Base Event ブラウザーを使用します。このシナリオで使用する場合では静的モニターは使用しないため、.mon ファイルを使用してデプロイされたアプリケーションは継続して特定のサービス・コンポーネント・イベント・ポイントをモニターします。静的モニターの実行方法について詳しくは、IBM® WebSphere Integration Developer インフォメーション・センターを参照してください。

その代わりに、この例で使用するシナリオでは、サーバーにすでにデプロイされて実行されているアプリケーションのサービス・コンポーネントで、モニター・イベント・ポイントを選択する方法を示します。ここでは、アプリケーションの処理がそれらのイベント・ポイントのいずれかに到達した場合常にモニター機能がイベントを送出する仕組みについて知ることができます。送出される各イベントは CEI サーバーにパブリッシュされます。CEI サーバーにはデータベースに関するイベント情報が保管されます。イベントを表示するには、Common Base Event ブラウザーを使用します。

## このチュートリアルの目的

このチュートリアルを完了すれば、次の操作を実行できるようになります。

- モニターするサービス・コンポーネント・イベント・ポイントを、CEI サーバーにパブリッシュされるイベントとともに選択する。
- 保管されているイベントを Common Base Event ブラウザーによって表示する。

## このチュートリアルを完了するのに必要な時間

このチュートリアルを完了するには、およそ 15 分から 20 分かかります。

## 前提条件

このチュートリアルを実行するには、次の条件を満たす必要があります。

- サーバーを構成および始動済みである。
- CEI およびそのデータベースを構成済みである。
- サーバーの診断トレース・サービスが使用可能に設定されている。
- サーバーでサンプル・ギャラリー・アプリケーションがインストールおよび始動済みである。
- サーバーでビジネス・ルール・サンプル・アプリケーションがインストールおよび始動済みである。「サンプル・ギャラリー」ページの指示に従って、ビジネス・ルール・サンプル・アプリケーションをセットアップし、実行します。

これらのすべての前提条件が満たされたら、チュートリアルに進む前に、少なくとも一度サンプル・ギャラリーからビジネス・ルール・サンプル・アプリケーションを実行してください。

### 例: Common Event Infrastructure サーバーによるモニター:

CEI サーバーによるモニターでは、管理コンソールを使用して、イベント・タイプの詳細を管理したり、記録されたイベントを Common Base Event ブラウザーで表示したりすることができます。この例では、コンソールを使用して一部のイベント・タイプで記録された詳細のレベルを変更する方法や、Common Base Event ブラウザーを使用して個々のイベントの情報を表示する方法を示します。

### このタスクについて

このシナリオではビジネス・ルール・サンプル・アプリケーションを使用するため、このアプリケーションが配置されている Web ページをあらかじめ開いてください。この Web ページは開いたままにしておいてください。モニター・パラメーターを指定した後、このサンプルを実行します。サンプルを少なくとも一度実行しておいてください。実行しておく、サンプルがモニター対象として選択可能な機能のリストに表示されます。

### 手順

1. 管理コンソールを開きます。
2. ナビゲーション・ペインで、「サーバー」 → 「サーバー・タイプ」 → 「WebSphere Application Server」をクリックします。
3. *server\_name* をクリックします。
4. 「トラブルシューティング」の下で、「ロギングおよびトレース (Logging and tracing)」をクリックします。
5. 「ログ詳細レベルの変更 (Change Log Detail levels)」をクリックします。
6. 「ランタイム (Runtime)」タブを選択します。
7. **WBILocationMonitor.CEI.BR** のツリーを展開します。  
**WBILocationMonitor.CEI.BR.ersample.\*** エレメントの下に以下の 5 つのイベント・タイプが表示されます。各イベント・タイプには、**Operation.\_calculateDiscount** 関数によって付加された名前 **WBILocationMonitor.CEI.BR.ersample\_module.DiscountRuleGroup** と、次の性質が含まれています。

- ENTRY
  - EXIT
  - FAILURE
  - SelectionKeyExtracted
  - TargetFound
8. 各イベントをクリックして「**finest**」を選択します。
  9. 「**OK**」をクリックします。
  10. 「ビジネス・ルール・サンプル・アプリケーション (business rules sample application)」ページに切り替え、アプリケーションを一度実行します。
  11. 管理コンソールに戻り、ナビゲーション・ペインから、「**統合アプリケーション**」 → 「**Common Base Event ブラウザー**」を選択します。
  12. Network Deployment 環境内のノードでサーバーを実行している場合は、「**イベント・データ・ストア**」フィールドにサーバーとノードの名前が含まれるように変更する必要がある場合もあります。cell/nodes/node\_name/servers/server\_name/ejb/com/ibm/events/access/EventAccess というストリングを入力します。
  13. 「**イベントの取得**」を押します。

### タスクの結果

Common Base Event ブラウザーの上側のペインに、サンプル・アプリケーションを実行したときに CEI サーバーにパブリッシュされた 4 つのビジネス・ルール・イベントのリストが表示されます。イベントの 1 つを選択してください。下側のペインにイベントの内容が表示されます。このイベントとロガーにパブリッシュされたイベントを比較してください。CEI サーバーにパブリッシュされた元の XML ストリングはブラウザーによって解析済みで、ENTRY イベントと EXIT イベントのビジネス・オブジェクト・コードは、元の 16 進形式から読み取り可能な XML に変換されています。この演習内の前のステップに戻り、詳細レベルを「**finest**」から「**fine**」または「**finer**」に変更して、イベント間の違いを比較することもできます。

この演習を完了すれば、CEI サーバーを使用してモニターするサービス・コンポーネント・イベント・ポイントを選択する方法を理解したことになります。このタイプのモニターで送出されるイベントは標準形式で、結果はデータベースにパブリッシュされることがわかりました。また、Common Base Event ブラウザーを使用してデータベースからイベントを取得し、個別のイベント情報を解析済み表形式でブラウザー上に表示することもできるようになります。

### 次のタスク

ビジネス・ルール・サンプル・アプリケーションのモニターを終了する場合は、ここで概要を示したステップを逆戻りして、サンプル・イベントの詳細レベルを「**info**」にリセットしてください。

## セッション・モニター

Common Base Event ブラウザーを使用して、Common Event Infrastructure データベースで同じセッション ID 属性を持つすべてのイベントを検索することによって、同じセッションの一部である複数のイベントをモニターすることができます。



WebSphere Process Server には、単一セッションの一部であるすべてのサービス・コンポーネント・イベントを識別できる拡張機能があります。Common Base Event の標準エレメントには、contextDataElement エレメントの下の WBISessionID という名前の属性があります。各セッションの固有 ID はこの属性に保管されているので、そのセッションの一部であったすべてのサービス・コンポーネント・イベントを識別できます。Common Base Event ブラウザーの「**SessionID**」フィールドを使用して、Common Event Infrastructure (CEI) データベースに保管されていて指定したセッション ID と一致するイベントを検索できます。この機能を使用すると、すべてのサービス・コンポーネント・イベントのプロセス・フローおよびプロセス・コンテンツを簡単に確認することができます。この情報を使用して、アプリケーションの効率性を評価したり、特定の環境の下でのみ発生する問題を診断したりできます。

Common Base Event ブラウザーを使用して、イベントおよび関連するコンテンツに関する戻されたリストを表示できます。「すべてのイベント」ビューをクリックすると、リンクの列が表示され、イベントの詳細を参照できます。特定のイベントの「失敗」列にリンクがある場合、そのリンクをクリックして失敗イベントの詳細を表示できます。同様に、特定のイベントに関連付けられた「ビジネス・プロセス」にリンクがある場合、そのリンクをクリックして Business Process Choreographer Explorer を開き、ビジネス・プロセス・イベントまたはヒューマン・タスク・イベントの詳細を表示できます。

#### 関連概念

3 ページの『モニターを使用可能にする方法』

行うモニターのタイプに応じて、モニター用サービス・コンポーネント・イベント・ポイントを指定する方法はいくつかあります。



---

## 第 3 章 モニター対象イベントの表示

モニター対象イベントのパブリッシュ結果を表示する方法は多数あり、使用するモニターのタイプによって異なります。このセクションでは、パフォーマンス・データ、イベント・ログ、および Common Event Infrastructure データベースに格納されているサービス・コンポーネント・イベントを表示する方法を説明します。

---

### Tivoli Performance Viewer でのパフォーマンス・メトリックの表示

Tivoli Performance Viewer を使用したパフォーマンス・モニターの開始および停止、Performance Monitoring Infrastructure のデータをグラフ形式または表形式でリアルタイムに表示、およびオプションで、後で同じビューアーで再確認できるファイルにデータを記録することができます。

#### 始める前に

Tivoli Performance Viewer でパフォーマンス・メトリックを表示するには、以下の条件が満たされている必要があります。

- モニターするサーバーがノード上で稼働している。
- Performance Monitoring Infrastructure (PMI) が使用可能になっている。
- モニターするサービス・コンポーネント・イベント・ポイントが 1 回以上呼び出されており、ビューアー内で選択可能である。

#### このタスクについて

Tivoli Performance Viewer (TPV) は、サーバーのパフォーマンスに関するさまざまな詳細情報を表示できる強力なアプリケーションです。WebSphere Application Server インフォメーション・センターの『Tivoli Performance Viewer を使用したパフォーマンスのモニター』というセクションには、さまざまな目的でのこのツールの使用法の詳細が記載されており、このプログラムの詳細な使用方法に関するリソースも含まれています。このセクションでは、WebSphere Process Server の固有イベントのパフォーマンス・データの表示についてのみ説明します。

Performance Viewer により、管理者およびプログラマーは、WebSphere Process Server の現在の正常性をモニターできます。データの収集と表示はプロセス・サーバーで行われるため、パフォーマンスに影響があります。パフォーマンスへの影響を最小限に抑えるには、モニターするアクティビティーが行われるサーバーのみをモニターしてください。

注: 統計を参照するときに、カウンター・タイプの統計と期間タイプの統計を混合しないでください。カウンターは累積されるため、アプリケーションによってはグラフのスケールがすぐに大きくなる可能性があります。反対に、期間型統計はシステムが各イベントを処理するのにかかる時間の平均を示すため、一定の範囲に保たれます。このため、統計とその相対目盛りの不均衡が原因で、いずれかのタイプの統計がビューアーのグラフ上で偏ることがあります。

## 手順

- 現在のパフォーマンス・アクティビティの表示
  - 管理コンソールのナビゲーション・ツリーで、「**モニターおよびチューニング**」 → 「**Performance Viewer**」 → 「**現行アクティビティ**」をクリックします。
  - 「**サーバー**」を選択し、モニターするアクティビティがあるサーバーの名前をクリックします。別の方法として、モニターするアクティビティがあるサーバーのチェック・ボックスを選択し、「**モニターの開始**」をクリックすることもできます。複数のサーバーのモニターを同時に開始するには、複数のサーバーを選択し、「**モニターの開始**」をクリックします。
  - 「**パフォーマンス・モジュール**」を選択します。
  - 表示する各パフォーマンス・モジュール名の横にあるチェック・ボックスを選択します。パフォーマンス統計を生成し、少なくとも 1 回呼び出された WebSphere Process Server イベントが、WBISStats.RootGroup 階層の下にリストされます。ツリーを展開する場合はノードの横の + をクリックし、縮小する場合はノードの横の - をクリックします。
  - 「**モジュールの表示**」をクリックします。ページの右側に、要求されたデータを示すグラフまたは表が表示されます。デフォルトではグラフが表示されず。

各モジュールには、複数のカウンターが関連付けられています。カウンターは、データのグラフまたは表の下にある表に表示されます。選択されたカウンターは、グラフまたは表内に表示されます。カウンターの横にあるチェック・ボックスを選択またはクリアすることによって、グラフや表にカウンターを追加またはグラフや表からカウンターを削除できます。デフォルトでは、モジュールごとに最初の 3 つのカウンターが表示されます。

カウンターは 20 個まで選択可能で、「現行アクティビティ」モードの TPV に表示できます。

- オプション: グラフまたは表からモジュールを削除するには、モジュールの横のチェック・ボックスをクリアし、再度「**モジュールの表示**」をクリックします。
  - オプション: データを表形式で表示するには、カウンターを選択する表で「**表の表示**」をクリックします。切り替えてグラフに戻すには、「**グラフの表示**」をクリックします。
  - オプション: グラフの凡例を表示するには、「**凡例の表示**」をクリックします。凡例を非表示にするには、「**凡例の非表示**」をクリックします。
  - イベントのパフォーマンスのモニターが終了したら、「**Tivoli Performance Viewer**」をクリックし、モニターしていたサーバーを選択して、「**モニターの停止**」をクリックします。
- パフォーマンス統計のログ記録

サーバー上でモニターがアクティブな間は、現在使用可能なすべての PMI カウンターのデータをログに記録し、結果を TPV ログ・ファイルに記録することができます。毎回 20 個までのカウンターを組み合わせ、特定期間の TPV ログ・ファイルを複数回表示することができます。サーバーにおける特定期間のさまざまなパフォーマンス測定値の間の関係を柔軟に監視できます。

1. サマリー・レポートまたはパフォーマンス・モジュールを表示する場合は、「**ロギング開始**」をクリックします。
2. 終了したら、「**ロギング停止**」をクリックします。デフォルトでは、ログ・ファイルは、サーバーが稼働しているノードの `profile_root/logs/tpv` ディレクトリーに保管されます。TPV は保管スペースへの書き込みが終了すると、ログ・ファイルを自動的に圧縮します。各圧縮ファイルに含まれるログ・ファイルが 1 つのみで、その名前は圧縮ファイルと同じになっている必要があります。
3. ログを表示するには、管理コンソールのナビゲーション・ツリーで、「**モニターおよびチューニング**」 → 「**Performance Viewer**」 → 「**ログの表示**」をクリックします。

---

## サービス・コンポーネント・イベント・ログ・ファイルの表示と解釈

このトピックでは、サービス・コンポーネント・モニターによって生成されたログ・ファイルにある情報を解釈する方法について説明します。ログ・ファイルは、管理コンソールのログ・ビューアーで表示したり、任意の個別のテキスト・ファイル・エディターで表示したりできます。

サービス・コンポーネント・モニターによってロガーに送出されるイベントは、Common Base Event 形式でエンコードされます。ログ・ファイルにパブリッシュされる場合、イベントは XML タグ付け形式での 1 行の長いテキストとして記述され、これにはロガー固有のフィールドもいくつか含まれます。ログに記録されたイベントの Common Base Event コーディングを復号する方法について詳しくは、この資料のイベント・カタログのセクションを参照してください。このセクションを参照することにより、ログ・ファイルの各項目に含まれる他のフィールドや、ロガーの構成時に選択したログ・ファイル用の形式がどのように構造化されるかについて理解できます。

### 基本形式フィールドと拡張形式フィールド

ロギング出力は、ファイルまたはメモリー内循環バッファーに送信することができます。トレース出力をメモリー内の循環バッファーに送信する場合は、表示できるようにするためにまずトレース出力をファイルにダンプする必要があります。出力はプレーン・テキストとして生成されます。形式は基本、拡張、またはログ・アナライザーのうちからユーザーが指定した形式になります。出力の基本形式と拡張形式は、メッセージ・ログで使用可能な基本形式と拡張形式に似ています。基本形式と拡張形式で使用するフィールドおよびフォーマット手法の多くは同一のもので、これらの形式で使用可能なフィールドは次の通りです。

#### TimeStamp

タイム・スタンプは、フォーマット設定するプロセスのロケールを使用してフォーマット設定されます。タイム・スタンプには完全修飾日付 (YYMMDD)、ミリ秒単位までの 24 時間表示、および時間帯が含まれません。

#### ThreadId

トレース・イベントを発行したスレッドのハッシュ・コードから生成される 8 文字の 16 進値。

#### ThreadName

メッセージまたはトレース・イベントを発行した Java スレッドの名前。

**ShortName**

トレース・イベントを発行したロギング・コンポーネントの省略名。通常は、WebSphere Process Server 内部コンポーネントのクラス名ですが、ユーザー・アプリケーションで使用される他の ID になっている場合もあります。

**LongName**

トレース・イベントを発行したロギング・コンポーネントの絶対パス名。通常は、WebSphere Process Server 内部コンポーネントの完全修飾クラス名ですが、ユーザー・アプリケーションで使用される他の ID になっている場合もあります。

**EventType**

トレース・イベントのタイプを示す 1 文字フィールド。トレース・タイプは小文字です。次の値があります。

- 1 FINE または EVENT タイプのトレース・エントリ。
- 2 FINER タイプのトレース・エントリ。
- 3 FINEST、DEBUG、または DUMP タイプのトレース・エントリ。
- Z トレース・タイプが認識されなかったことを示すプレースホルダー。

**ClassName**

メッセージまたはトレース・イベントを発行したクラス。

**MethodName**

メッセージまたはトレース・イベントを発行したメソッド。

**Organization**

メッセージまたはトレース・イベントを発行したアプリケーションを所有する組織。

**Product**

メッセージまたはトレース・イベントを発行した製品。

**Component**

メッセージまたはトレース・イベントを発行した製品内のコンポーネント。

**基本形式**

基本形式で表示されるトレース・イベントでは、次の形式が使用されます。

```
<timestamp><threadId><shortName><eventType>[className] [methodName] <textmessage>
      [parameter 1]
      [parameter 2]
```

**拡張形式**

拡張形式で表示されるトレース・イベントでは、次の形式が使用されます。

```
<timestamp><threadId><eventType><UOW><source=longName>[className] [methodName]
<Organization><Product><Component> [thread=threadName]
<textMessage> [parameter 1=parameterValue] [parameter 2=parameterValue]
```

**ログ・アナライザー形式**

ログ・アナライザー形式を指定することにより、WebSphere Application Server に組み込まれたアプリケーションである Log Analyzer ツールを使用して、トレース出力を開くことができます。この形式では、Log Analyzer のマージ機能を使用できるため、2 つの異なるサーバー・プロセスのトレースを相関させる場合に便利です。

---

## 第 4 章 イベント・カタログ

イベント・カタログには、サービス・コンポーネント・タイプごとにモニター可能なすべてのイベントの仕様、および各イベントによって作成される、関連付けられた Common Base Event 拡張データ・エレメントが格納されています。

このセクションに記載されている情報は、個々のイベントの構造を理解するための参照資料としてご使用ください。この知識により、各イベントに含まれている情報の意味を理解し、各イベントによって生成される比較的大量のデータの中から必要な情報を素早く識別することができます。

このセクションに記載される情報は、以下の項目を対象としています。

- Common Base Event の構造および標準エレメント
- Business Process Choreographer サービス・コンポーネントのイベントのリスト
- WebSphere Process Server 固有のサービス・コンポーネントのリスト
- 各イベント・タイプに固有の Common Base Event の拡張機能

また、サービス・コンポーネントによって処理可能なビジネス・オブジェクトを、サービス・コンポーネント・イベントに取り込む方法についても論じています。

指定されたタイプのイベントは、Common Event Infrastructure (CEI) バスを通して CEI サーバーまたはロガーに送出される場合、Common Base Event の形式を取ります。これは、基本的には、イベント・カタログの仕様に従って作成されたイベント・エレメントのカプセル化形式の XML です。Common Base Event には、標準エレメント、サーバー・コンポーネント識別エレメント、イベント相関領域 ID、および各イベント・タイプに固有の追加エレメントなどのセットが含まれています。これらのすべてのエレメントは、イベントがサービス・コンポーネント・モニターによって送出されたときにはいつでも、CEI サーバーまたはロガーに渡されます。ただし例外として、イベントに有効搭載量内のビジネス・オブジェクト・コードが組み込まれている場合は、イベントに組み込むビジネス・オブジェクト・データの量を指定します。

---

### Common Base Event の標準エレメント

ここでは、サービス・コンポーネント・モニターから送出されるすべてのイベントに含まれる Common Base Event のエレメントをリストします。

属性	説明
version	1.0.1 に設定します。
creationTime	イベントが作成された時刻 (UTC 形式)。
globalInstanceId	Common Base Event インスタンスの ID。この ID は自動的に生成されます。
localInstanceId	この ID は自動的に生成されます (空白の場合もあり)。

属性	説明
severity	ビジネス・プロセスまたはヒューマン・プロセスにイベントが及ぼす影響。この属性は 10 (情報) に設定されます。それ以外の場合、これは使用されません。
priority	使用しません。
reporterComponentId	使用しません。
locationType	Hostname に設定します。
location	実行中のサーバーのホスト名に設定します。
application	使用しません。
executionEnvironment	オペレーティング・システムを示すストリング。
component	プロセス・サーバーのバージョン。ビジネス・プロセスおよびヒューマン・タスクの場合は、順に、WPS#、SCA バージョン、現行プラットフォームの ID、および下位層のソフトウェア・スタックのバージョン ID を設定します。
componentType	Apache QName 形式を基にしたコンポーネント QName。  ビジネス・プロセスの場合、次のように設定します。  www.ibm.com/namespaces/autonomic/Workflow_Engine  ヒューマン・タスクの場合、次のように設定します。  www.ibm.com/xmlns/prod/websphere/scdl/human-task
subComponent	監視可能なエレメント名。  ビジネス・プロセスの場合、BFM に設定します。 ヒューマン・タスクの場合、HTM に設定します。
componentIdType	ProductName に設定します。
instanceId	サーバーの ID。この ID の形式は、 <i>cell_name/node_name/server_name</i> です。区切り文字はオペレーティング・システムによって異なります。
processId	オペレーティング・システムのプロセス ID。
threadId	Java 仮想マシン (JVM) のスレッド ID。
Situation Type	イベントが報告される原因となったシチュエーションのタイプ。固有のコンポーネントの場合は、ReportSituation に設定します。
Situation Category	イベントが報告される原因となったシチュエーション・タイプのカテゴリ。固有のコンポーネントの場合は、STATUS に設定します。
Situation Reasoning Scope	報告されたシチュエーションの影響の有効範囲。固有のコンポーネントの場合は、EXTERNAL に設定します。
ECSCurrentID	現在のイベント相関範囲 ID の値。
ECSParentID	親イベント相関範囲 ID の値。
WBISessionID	現在のセッション ID の値。
extensionName	イベント名に設定します。



---

## イベント内のビジネス・オブジェクト

ビジネス・オブジェクト・データは、バージョン 6.1 以降、イベントの中で XML 形式で搬送されます。Common Base Event 形式は `xs:any` スキーマを含んでおり、これは、ビジネス・オブジェクト・ペイロードを XML エlement内にカプセル化します。

サービス・コンポーネント・イベントに収集するビジネス・オブジェクトの詳細レベルを指定します。この詳細レベルは、イベントに渡されるビジネス・オブジェクト・コードの量にのみ影響します。その他のすべての Common Base Event Element (標準とイベント固有の両方) は、イベントにパブリッシュされます。サービス・コンポーネント・イベントに適用可能な詳細レベルの名前は、WebSphere Integration Developer を使用して静的モニターを作成したか、または管理コンソールで動的モニターを作成したかに応じて異なりますが、次の表に示すように対応しています。

管理コンソールの詳細レベル	Common Base Event/WebSphere Integration Developer の詳細レベル	パブリッシュされる有効搭載量情報
FINE	EMPTY	なし。
FINER	DIGEST	有効搭載量の説明のみです。
FINEST	FULL	有効搭載量のすべてです。

詳細レベルは、イベント・インスタンス・データに含まれている `PayloadType` Elementによって指定されます。実際のビジネス・オブジェクト・データは、モニターが FULL/FINEST の詳細を記録する設定になっていれば、イベントのみに組み込まれます。ビジネス・オブジェクト・データ自体は、`xsd:any` スキーマの下で Common Base Event に組み込まれています。プロセス・サーバーのビジネス・オブジェクト・ペイロードは `wbi:event` という名前のルート・Elementによって表示できます。イベント出力をロガーにパブリッシュすると、ログ・ファイルの参照時に出力が表示されます。イベントが CEI サーバーに対してパブリッシュされている場合は、Common Base Event ブラウザーを使用してイベントを表示できます。その場合、`wbi:event` リンクをクリックすると、ビジネス・オブジェクト・データが表示されます。

---

## Business Process Choreographer イベント

WebSphere Process Server には、ビジネス・プロセスとヒューマン・タスクで使用する Business Process Choreographer サービス・コンポーネントが組み込まれています。このセクションでは、これらのコンポーネントでモニター可能なイベント・ポイントについて説明します。

---

## WebSphere Process Server イベント

WebSphere Process Server の特徴は独自のサービス・コンポーネントを持つことであり、これらのコンポーネントにはそれぞれ、モニターできる独自のイベント・ポイントのセットがあります。

サービス・コンポーネントにはエレメントが 1 つ以上あります。それらは各サービス・コンポーネントが処理する異なるステップの集合です。同様に、各エレメントには独自のイベント性質セットがあります。イベント性質は、サービス・コンポーネント・エレメントの処理時に到達するキーポイントです。すべてのサービス・コンポーネント、そのエレメントと関連するイベント性質、および各イベントに固有の拡張データ・エレメントがリストされています。

## リソース・アダプター・イベント

リソース・アダプター・コンポーネントで使用可能なイベント・タイプをリストします。

ここでは、モニター可能なリソース・アダプター・コンポーネント (ベース名 eis:WBI.JCAAdapter) のエレメントを、関連するイベント性質、イベント名、および各イベントに固有の拡張データ・エレメントとともにリストします。

イベント名	イベント性質	イベント内容	タイプ
<b>InboundEventRetrieval エレメント</b>			
eis:WBI.JCAAdapter. InboundEventRetrieval. ENTRY	ENTRY	pollQuantity	int
		status	int
		eventTypeFilters	string
eis:WBI.JCAAdapter. InboundEventRetrieval. EXIT	EXIT	なし	
eis:WBI.JCAAdapter. InboundEventRetrieval. FAILURE	FAILURE	FailureReason	例外
<b>InboundEventDelivery エレメント</b>			
eis:WBI.JCAAdapter. InboundEventDelivery.ENTRY	ENTRY	なし	
eis:WBI.JCAAdapter. InboundEventDelivery.EXIT	EXIT	なし	
eis:WBI.JCAAdapter. InboundEventDelivery.FAILURE	FAILURE	FailureReason	例外
<b>Outbound エレメント</b>			
eis:WBI.JCAAdapter. Outbound.ENTRY	ENTRY	なし	
eis:WBI.JCAAdapter. Outbound.EXIT	EXIT	なし	
eis:WBI.JCAAdapter. Outbound.FAILURE	FAILURE	FailureReason	例外
<b>InboundCallbackAsyncDeliverEvent エレメント</b>			
eis:WBI.JCAAdapter. InboundCallbackAsyncDeliverEvent. ENTRY	ENTRY	なし	
eis:WBI.JCAAdapter. InboundCallbackAsyncDeliverEvent. EXIT	EXIT	なし	
eis:WBI.JCAAdapter. InboundCallbackAsyncDeliverEvent. FAILURE	FAILURE	FailureReason	例外

イベント名	イベント性質	イベント内容	タイプ
<b>InboundCallbackSyncDeliverEvent エレメント</b>			
eis:WBIJCAAdapter. InboundCallbackSyncDeliverEvent. ENTRY	ENTRY	なし	
eis:WBIJCAAdapter. InboundCallbackSyncDeliverEvent. EXIT	EXIT	なし	
eis:WBIJCAAdapter. InboundCallbackSyncDeliverEvent. FAILURE	FAILURE	FailureReason	例外
<b>Polling エレメント</b>			
eis:WBIJCAAdapter. Polling.STARTED	STARTED	PollFrequency	int
		PollQuantity	int
eis:WBIJCAAdapter. Polling.STOPPED	STOPPED	なし	
<b>Delivery エレメント</b>			
eis:WBIJCAAdapter. Delivery.EXIT	EXIT	なし	
eis:WBIJCAAdapter. Delivery.FAILURE	FAILURE	EventID	string
		FailureReason	例外
<b>Retrieval エレメント</b>			
eis:WBIJCAAdapter. Retrieval.FAILURE	FAILURE	EventID	string
		FailureReason	例外
<b>Endpoint エレメント</b>			
eis:WBIJCAAdapter. Endpoint.FAILURE	FAILURE	FailureReason	例外
<b>Recovery エレメント</b>			
eis:WBIJCAAdapter. Recovery.EXIT	EXIT	なし	
eis:WBIJCAAdapter. Recovery.FAILURE	FAILURE	FailureReason	例外
<b>EventFailure エレメント</b>			
eis:WBIJCAAdapter. EventFailure.FAILURE	FAILURE	FailureReason	例外
<b>Connection エレメント</b>			
eis:WBIJCAAdapter. Connection.FAILURE	FAILURE	FailureReason	例外

## ビジネス・ルール・イベント

ビジネス・ルール・コンポーネントで使用可能なイベント・タイプをリストします。

ビジネス・ルール・コンポーネント (ベース名 br:WBI.BR) には、モニター可能な単一エレメントが含まれています。ここでは、このエレメントのすべてのイベント・タイプを、関連するイベント性質、イベント名、および各イベントに固有の拡張データ・エレメントとともにリストします。

イベント名	イベント性質	イベント内容	タイプ
br:WBI.BR.ENTRY	ENTRY	operationName	string
br:WBI.BR.EXIT	EXIT	operationName	string
br:WBI.BR.FAILURE	FAILURE	ErrorReport	例外
		operationName	string
WBI.BR. br:SelectionKeyExtracted	SelectionKeyExtracted	operationName	string
br:WBI.BR.TargetFound	TargetFound	operationName	string
		target	string

## ビジネス・ステート・マシン・イベント

ビジネス・ステート・マシン・コンポーネントで使用可能なイベント・タイプをリストします。

ここでは、モニター可能なビジネス・ステート・マシン・コンポーネント (ベース名 bsm:WBI.BSM) のエレメントを、関連するイベント性質、イベント名、および各イベントに固有のすべての拡張データ・エレメントとともにリストします。

イベント名	イベント性質	イベント内容	タイプ
<b>StateMachineDefinition エレメント</b>			
bsm:WBI.BSM. StateMachineDefinition. ALLOCATED	ALLOCATED	instanceID	string
bsm:WBI.BSM. StateMachineDefinition. RELEASED	RELEASED	instanceID	string
<b>Transition エレメント</b>			
bsm:WBI.BSM.Transition.ENTRY	ENTRY	instanceID	string
		name	string
bsm:WBI.BSM.Transition.EXIT	EXIT	instanceID	string
		name	string
bsm:WBI.BSM.Transition.FAILURE	FAILURE	ErrorReport	例外
		instanceID	string
		name	string
<b>State エレメント</b>			
bsm:WBI.BSM.State.ENTRY	ENTRY	instanceID	string
		name	string
bsm:WBI.BSM.State.EXIT	EXIT	instanceID	string
		name	string
bsm:WBI.BSM.State.FAILURE	FAILURE	ErrorReport	例外
		instanceID	string
		name	string
<b>Guard エレメント</b>			
bsm:WBI.BSM.Guard.ENTRY	ENTRY	instanceID	string
		name	string

イベント名	イベント性質	イベント内容	タイプ
bsm:WBI.BSM.Guard.EXIT	EXIT	instanceID	string
		name	string
		result	boolean
bsm:WBI.BSM.Guard.FAILURE	FAILURE	ErrorReport	例外
		instanceID	string
		name	string
<b>Action エlement</b>			
bsm:WBI.BSM.Action.ENTRY	ENTRY	instanceID	string
		name	string
bsm:WBI.BSM.Action.EXIT	EXIT	instanceID	string
		name	string
bsm:WBI.BSM.Action.FAILURE	FAILURE	ErrorReport	例外
		instanceID	string
		name	string
<b>EntryAction Element</b>			
bsm:WBI.BSM.EntryAction. ENTRY	ENTRY	instanceID	string
		name	string
bsm:WBI.BSM.EntryAction. EXIT	EXIT	instanceID	string
		name	string
bsm:WBI.BSM.EntryAction. FAILURE	FAILURE	ErrorReport	例外
		instanceID	string
		name	string
<b>ExitAction Element</b>			
bsm:WBI.BSM.ExitAction.ENTRY	ENTRY	instanceID	string
		name	string
bsm:WBI.BSM.ExitAction.EXIT	EXIT	instanceID	string
		name	string
bsm:WBI.BSM.ExitAction. FAILURE	FAILURE	ErrorReport	例外
		instanceID	string
		name	string
<b>Timer Element</b>			
bsm:WBI.BSM.Timer.START	START	instanceID	string
		name	string
		duration	string
bsm:WBI.BSM.Timer.STOPPED	STOPPED	instanceID	string
		name	string
		duration	string

## マップ・イベント

マップ・コンポーネントで使用可能なイベント・タイプをリストします。

ここでは、モニター可能なマップ・コンポーネント (ベース名 map:WBI.MAP) のエレメントを、そのイベント性質、イベント名、および各イベントに固有のすべての拡張データ・エレメントとともにリストします。

表 21. 基本エレメント

イベント名	イベント性質	イベント内容	タイプ
map:WBI.MAP.ENTRY	ENTRY	なし	なし
map:WBI.MAP.EXIT	EXIT	なし	なし
map:WBI.MAP.FAILURE	FAILURE	FailureReason	例外
Transformation エレメント			
map:WBI.MAP.Transformation. ENTRY	ENTRY	なし	なし
map:WBI.MAP.Transformation. EXIT	EXIT	なし	なし
map:WBI.MAP.Transformation. FAILURE	FAILURE	FailureReason	例外

## メディエーション・イベント

メディエーション・コンポーネントで使用可能なイベント・タイプをリストします。

ここでは、モニター可能なメディエーション・コンポーネント (ベース名 ifm:WBI.MEDIATION) のエレメントを、関連するイベント性質、名前、および各イベントに固有のすべての拡張データ・エレメントとともにリストします。

イベント名	イベント性質	イベント内容	タイプ
<b>OperationBinding エレメント</b>			
ifm:WBI.MEDIATION. OperationBinding.ENTRY	ENTRY	InteractionType	string
		TicketID	string
		Source	string
		Target	string
ifm:WBI.MEDIATION. OperationBinding.EXIT	EXIT	InteractionType	string
		TicketID	string
		Source	string
		Target	string
ifm:WBI.MEDIATION. OperationBinding.FAILURE	FAILURE	InteractionType	string
		TicketID	string
		Source	string
		Target	string
		ErrorReport	例外
<b>ParameterMediation エレメント</b>			
ifm:WBI.MEDIATION. ParameterMediation. ENTRY	ENTRY	タイプ	string
		TransformName	string
WBI.MEDIATION. ParameterMediation. EXIT	EXIT	タイプ	string
		TransformName	string

イベント名	イベント性質	イベント内容	タイプ
ifm:WBI.MEDIATION. ParameterMediation. FAILURE	FAILURE	タイプ	string
		TransformName	string
		ErrorReport	例外

## リカバリー・イベント

リカバリー・コンポーネントで使用可能なイベント・タイプをリストします。

リカバリー・コンポーネント (ベース名 `recovery:WBI.Recovery`) には、モニター可能な単一エレメントが含まれています。ここでは、このエレメントのすべてのイベント・タイプを、関連するイベント性質、イベント名、および各イベントに固有の拡張データ・エレメントとともにリストします。

イベント名	イベント性質	イベント内容	タイプ
recovery:WBI.Recovery. FAILURE	FAILURE	MsgId	string
		DestModuleName	string
		DestComponentName	string
		DestMethodName	string
		SourceModuleName	string
		SourceComponentName	string
		ResubmitDestination	string
		ExceptionDetails	string
		SessionId	string
		FailureTime	dateTime
		ExpirationTime	dateTime
		Status	int
		MessageBody	byteArray
		Deliverable	boolean
recovery:WBI.Recovery. DEADLOOP	DEADLOOP	DeadloopMsgId	string
		SIBusName	string
		QueueName	string
		Reason	string
recovery:WBI.Recovery. RESUBMIT	RESUBMIT	MsgId	string
		OriginalMesId	string
		ResubmitCount	int
		Description	string
recovery:WBI.Recovery. DELETE	DELETE	MsgId	string
		deleteTime	dateTime
		Description	string

## Service Component Architecture イベント

Service Component Architecture で使用可能なイベント・タイプをリストします。

Service Component Architecture (SCA) には、sca:WBI.SCA.MethodInvocation というベース名を持つ単一エレメントが含まれています。ここでは、このエレメントのすべてのイベントと関連する性質を、各イベントに固有のすべての拡張データ・エレメントとともにリストします。

注: これらのイベントを SCA 固有のアプリケーション応答測定 (ARM) パフォーマンス統計と混同しないでください。

イベント名	イベント性質	イベント内容	タイプ
WBI.SCA. MethodInvocation. ENTRY	ENTRY	SOURCE COMPONENT	string
		SOURCE INTERFACE	string
		SOURCE METHOD	string
		SOURCE MODULE	string
		SOURCE REFERENCE	string
		TARGET COMPONENT	string
		TARGET INTERFACE	string
		TARGET METHOD	string
		TARGET MODULE	string
WBI.SCA. MethodInvocation. EXIT	EXIT	SOURCE COMPONENT	string
		SOURCE INTERFACE	string
		SOURCE METHOD	string
		SOURCE MODULE	string
		SOURCE REFERENCE	string
		TARGET COMPONENT	string
		TARGET INTERFACE	string
		TARGET METHOD	string
		TARGET MODULE	string
WBI.SCA. MethodInvocation. FAILURE	FAILURE	SOURCE COMPONENT	string
		SOURCE INTERFACE	string
		SOURCE METHOD	string
		SOURCE MODULE	string
		SOURCE REFERENCE	string
		TARGET COMPONENT	string
		TARGET INTERFACE	string
		TARGET METHOD	string
		TARGET MODULE	string
		例外	string

## セレクター・イベント

セレクター・コンポーネントで使用可能なイベント・タイプをリストします。

セレクター・コンポーネントには、モニター可能な単一エレメントが含まれています。ここでは、このエレメントのすべてのイベント・タイプを、関連するイベント



性質、イベント名、および各イベントに固有の拡張データ・エレメントとともにリストします。すべてのセレクター・イベントのベース名は sel:WBI.SEL です。

イベント名	イベント性質	イベント内容	タイプ
sel:WBI.SEL.ENTRY	ENTRY	operationName	string
sel:WBI.SEL.EXIT	EXIT	operationName	string
sel:WBI.SEL.FAILURE	FAILURE	ErrorReport	例外
		operationName	string
sel:WBI.SEL. SelectionKeyExtracted	SelectionKeyExtracted	operationName	string
sel:WBI.SEL.TargetFound	TargetFound	operationName	string
		target	string



Printed in Japan