# CA Application Performance Management

## for SOA 実装ガイド

リリース 9.5

このドキュメント（組み込みヘルプ システムおよび電子的に配布される資料を含む、以下「本ドキュメント」）は、お客様への情報提供のみを目的としたもので、日本 CA 株式会社（以下「CA」）により随時、変更または撤回されることがあります。

CA の事前の書面による承諾を受けずに本ドキュメントの全部または一部を複写、譲渡、開示、変更、複本することはできません。 本ドキュメントは、CA が知的財産権を有する機密情報です。ユーザは本ドキュメントを開示したり、（i）本ドキュメントが関係する CA ソフトウェアの使用について CA とユーザとの間で別途締結される契約または (ii) CA とユーザとの間で別途締結される機密保持契約により許可された目的以外に、本ドキュメントを使用することはできません。

上記にかかわらず、本ドキュメントで言及されている CA ソフトウェア製品のライセンスを受けたユーザは、社内でユーザおよび従業員が使用する場合に限り、当該ソフトウェアに関連する本ドキュメントのコピーを妥当な部数だけ作成できます。ただし CA のすべての著作権表示およびその説明を当該複製に添付することを条件とします。

本ドキュメントを印刷するまたはコピーを作成する上記の権利は、当該ソフトウェアのライセンスが完全に有効となっている期間内に限定されます。 いかなる理由であれ、上記のライセンスが終了した場合には、お客様は本ドキュメントの全部または一部と、それらを複製したコピーのすべてを破棄したことを、CA に文書で証明する責任を負います。

準拠法により認められる限り、CA は本ドキュメントを現状有姿のまま提供し、商品性、特定の使用目的に対する適合性、他者の権利に対して侵害のないことについて、黙示の保証も含めいかなる保証もしません。 また、本ドキュメントの使用に起因して、逸失利益、投資損失、業務の中断、営業権の喪失、情報の喪失等、いかなる損害（直接損害か間接損害かを問いません）が発生しても、CA はお客様または第三者に対し責任を負いません。CA がかかる損害の発生の可能性について事前に明示に通告されていた場合も同様とします。

本ドキュメントで参照されているすべてのソフトウェア製品の使用には、該当するライセンス契約が適用され、当該ライセンス契約はこの通知の条件によっていかなる変更も行われません。

本ドキュメントの制作者は CA です。

「制限された権利」のもとでの提供: アメリカ合衆国政府が使用、複製、開示する場合は、FAR Sections 12.212、52.227-14 及び 52.227-19(c)(1)及び(2)、ならびに DFARS Section252.227-7014(b)(3) または、これらの後継の条項に規定される該当する制限に従うものとします。

Copyright © 2013 CA. All rights reserved. 本書に記載された全ての製品名、サービス名、商号およびロゴは各社のそれぞれの商標またはサービスマークです。

# CA Technologies 製品リファレンス

このドキュメントは、以下の CA Technologies 製品および機能に関するものです。

- CA Application Performance Management （CA APM）
- CA Application Performance Management ChangeDetector （CA APM ChangeDetector）
- CA Application Performance Management ErrorDetector （CA APM ErrorDetector）
- CA Application Performance Management for CA Database Performance （CA APM for CA Database Performance）
- CA Application Performance Management for CA SiteMinder® （CA APM for CA SiteMinder®）
- CA Application Performance Management for CA SiteMinder® Application Server Agents （CA APM for CA SiteMinder® ASA）
- CA Application Performance Management for IBM CICS Transaction Gateway （CA APM for IBM CICS Transaction Gateway）
- CA Application Performance Management for IBM WebSphere Application Server （CA APM for IBM WebSphere Application Server）
- CA Application Performance Management for IBM WebSphere Distributed Environments （CA APM for IBM WebSphere Distributed Environments）
- CA Application Performance Management for IBM WebSphere MQ （CA APM for IBM WebSphere MQ）
- CA Application Performance Management for IBM WebSphere Portal （CA APM for IBM WebSphere Portal）
- CA Application Performance Management for IBM WebSphere Process Server （CA APM for IBM WebSphere Process Server）
- CA Application Performance Management for IBM z/OS® （CA APM for IBM z/OS®）
- CA Application Performance Management for Microsoft SharePoint （CA APM for Microsoft SharePoint）
- CA Application Performance Management for Oracle Databases （CA APM for Oracle Databases）

- CA Application Performance Management for Oracle Service Bus （CA APM for Oracle Service Bus）
- CA Application Performance Management for Oracle WebLogic Portal （CA APM for Oracle WebLogic Portal）
- CA Application Performance Management for Oracle WebLogic Server （CA APM for Oracle WebLogic Server）
- CA Application Performance Management for SOA （CA APM for SOA）
- CA Application Performance Management for TIBCO BusinessWorks （CA APM for TIBCO BusinessWorks）
- CA Application Performance Management for TIBCO Enterprise Message Service （CA APM for TIBCO Enterprise Message Service）
- CA Application Performance Management for Web Servers （CA APM for Web Servers）
- CA Application Performance Management for webMethods Broker （CA APM for webMethods Broker）
- CA Application Performance Management for webMethods Integration Server （CA APM for webMethods Integration Server）
- CA Application Performance Management Integration for CA CMDB （CA APM Integration for CA CMDB）
- CA Application Performance Management Integration for CA NSM（CA APM Integration for CA NSM）
- CA Application Performance Management LeakHunter （CA APM LeakHunter）
- CA Application Performance Management Transaction Generator （CA APM TG）
- CA Cross-Enterprise Application Performance Management
- CA Customer Experience Manager （CA CEM）
- CA Embedded Entitlements Manager （CA EEM）
- CA eHealth® Performance Manager （CA eHealth）
- CA Insight™ Database Performance Monitor for DB2 for z/OS®
- CA Introscope®
- CA SiteMinder®
- CA Spectrum® Infrastructure Manager （CA Spectrum）

- CA SYSVIEW® Performance Management （CA SYSVIEW）

# CA への連絡先

テクニカル サポートの詳細については、弊社テクニカル サポートの Web サイト（http://www.ca.com/jp/support/）をご覧ください。

# 目次

# 第 1 章: CA Application Performance Management for SOA について

サービス指向アーキテクチャ (SOA) は、アプリケーション プラットフォームの 1 つであり、このアーキテクチャにより、組織では疎結合のサービスを共有および再使用して、ビジネス目標を達成できるようになります。

サービス指向アーキテクチャを使用したアプリケーションの展開には複数のメリットがあります。しかし、SOA 環境の監視に固有の課題もあります。 このセクションでは、CA Application Performance Management for SOA（CA APM for SOA）を使用して、それらの課題に対応するための重要な方法を主に取り上げます。

このセクションには、以下のトピックが含まれています。



## サービス指向アーキテクチャとは

*サービス指向アーキテクチャ（SOA）*は、標準化された通信プロトコルに基づくアプリケーション プラットフォームであり、このアーキテクチャにより、疎結合のサービスを使用して、ビジネス目標を達成できるようになります。

SOA を使用するメリットには、ビジネスの敏捷性と柔軟性の向上、カスタマ サービスと作業効率の向上、開発コストの削減などがあります。 ただし、複雑な SOA 環境の監視および管理は、従来のクライアント サーバ環境の管理に比べると、はるかに難しくなる場合があります。

従来のクライアント/サーバ環境では、クライアントと限られた数のサーバとの間で直接の通信が行われます。 問題が発生しても、少数のシステムのみが個々のビジネス トランザクションにかかわっているため、障害の原因を見つけることは通常は容易です。 トランザクションに直接かかわる特定のシステムを調査することにより、その問題の原因を隔離できます。

Web アプリケーション サーバを中心に、複数のクライアント サーバ システムにわたってアプリケーションへのアクセスが分散されていると、問題の原因の特定はより困難になります。 パフォーマンスの低下、エラー、または処理の失敗は、Web サーバが接続されたインフラストラクチャに属している、あらゆるコンポーネントやコンピュータが原因となる可能性があります。

サービス指向アーキテクチャでは、アプリケーション パフォーマンスおよび可用性の監視のために、新たな複雑な仕組みが導入されています。 SOA では、疎結合のサービスは標準化された通信に基づいて、さまざまなプラットフォーム上で実行されているアプリケーションを統合および拡張します。 そのようなサービスでは、基盤となるオペレーティング システムまたはプラットフォームをビジネス ロジックから切り離すため、組織では市場や製品の動向の変化に対して、対応性と俊敏性が向上します。 個々のサービスは、異機種環境間にまたがって結びつけられた依存関係を形成することで、複雑なビジネス プロセスや多段階のビジネス プロセスの特定の一部分を処理するように設計できます。

サービス指向アーキテクチャの使用により、組織ではアプリケーションをより短期間に、よりコスト パフォーマンスの高い方法で開発して展開できます。これは、独立したコンポーネントとしてサービスを再使用したり、修正したり、置き換えたりできるためです。 ただし、アプリケーション アーキテクチャへのこの効率的なモジュール方式のアプローチゆえに、アプリケーション管理に関する独自の課題が生じることになります。

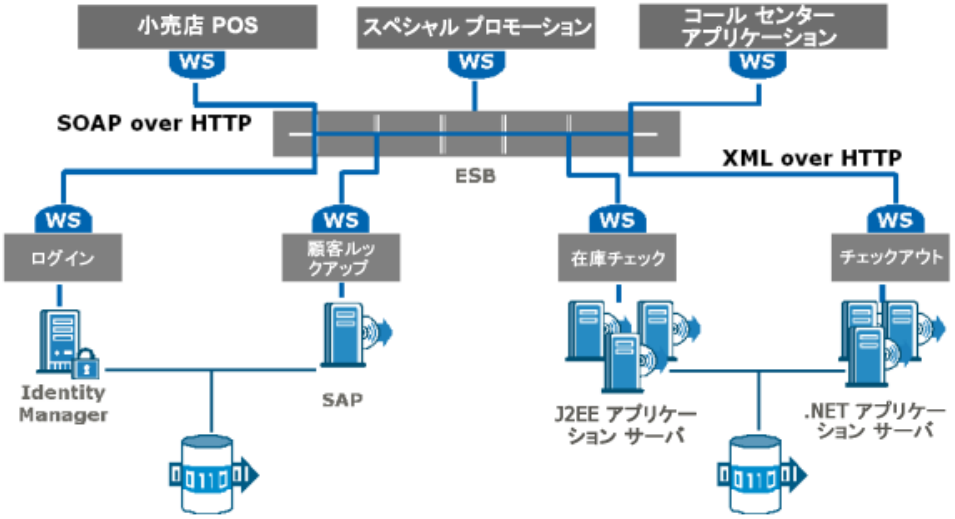
# SOA インフラストラクチャの一般的なコンポーネント

SOA を使用することで、ビジネス プロセスの導入および統合が効率化されますが、基盤となる SOA インフラストラクチャは通常、複数のコンポーネントの複雑なインタラクションに依存します。 たとえば、単一のビジネス トランザクションを完了するには、通常、複数のサービスが必要であり、それらのサービスは異なるプロトコルを使用して互いにメッセージ交換を行うさまざまなコンポーネント上で実行されているのが一般的です。トランザクションの重要なコンポーネントの監視には、サービスがどのように通信しているか、要求と応答がどのようにサービス間でルーティングされているか、どのサービスが他のサービスへの依存関係があるか、重大なボトルネックがどこで発生するかについて認識できる必要があります。

具体的な監視の要件は SOA の実装によって異なりますが、標準の SOA インフラストラクチャには次のものが含まれます。

- 複数のサービス。標準のインターフェースを使用して、クライアント側およびサーバ側コンポーネントが相互にメッセージをやり取りします。
- メッセージ処理システム。コンポーネント間で要求および応答を変換、ルーティング、配信します。
- アダプタ。外部またはレガシー システムが、展開されたサービスに接続して、それらのサービスを利用するために使用します。
- レジストリ。使用可能なサービスに関する情報が記録および発行されます。

以下の図で示している簡略化された SOA インフラストラクチャでは、簡易オブジェクト アクセス プロトコル（SOAP）および拡張可能マークアップ言語（XML）を使用して、Web サービス（WS）が *Enterprise Service Bus*（ESB）を通じて、相互に通信しています。ESB は、適切なターゲットへのメッセージの配信を制御するために使用されています。




SOA インフラストラクチャは、モジュラー方式で提供される再利用可能で柔軟なコンポーネントで構築されます。そのため、このような環境でのトランザクションでは通常、多くのコンポーネントが関与します。 トランザクションにかかわるコンポーネントの数が増えるにつれ、トランザクションのフローの追跡はますます難しくなります。 さまざまなトランスポート プロトコルを使用してプロセス間またはプラットフォーム間で渡されるメッセージが増えるにつれて、潜在的な障害点も増えます。 さらに、組織は一度 SOA 環境でのアプリケーションの展開を始めると、ますます多くのミッション クリティカルなサービスをその環境に移動させる傾向があります。その結果、ビジネスの稼働状況を保つために、インフラストラクチャの監視がますます重要になります。

CA APM for SOA では、これらの独自の課題に対処するために、SOA インフラストラクチャの稼働状況、および SOA トランザクションにかかわっているコンポーネントのリアルタイム パフォーマンスを可視化します。

# CA Introscope® を使用して SOA パフォーマンスを監視する方法

CA APM for SOA では、SOA 環境のアプリケーション パフォーマンスのリアル タイム ビューおよび履歴ビューが提供されています。- また、CA Introscope® のコア機能を拡張して、Web サービスのトランザクションの監視および追跡が可能になります。対象になるのは、ユーザがサービスを要求する*フロントエンド*から、要求への応答に使用される*バックエンド*システムへのトランザクションです。 CA Introscope® と共に CA APM for SOA を使用することにより、プロアクティブに SOA クライアントおよびサーバのパフォーマンスの監視や問題の切り分けなどを行うことができます。またアプリケーションのソース コードへのアクセス、SOA アーキテクトに問い合わせずにサービス関連の問題の詳細な分析などを行うことができます。

CA Introscope® に CA APM for SOA を追加すると、以下のことが可能になり、SOA 環境の監視がより容易になります。

- SOA クライアントおよびサーバの稼働状況およびパフォーマンスをダッシュボードで可視化。SOA 環境をプロアクティブに管理できるようになります。
- 概要および詳細メトリック。Web サービス、アプリケーション、またはバックエンドへの問題の切り分けに役立ちます。
- エージェント、サービス、オペレーション間の依存関係を視覚的に表示。あるコンポーネントに関する問題がどのように他のコンポーネントに影響を与える可能性があるかの評価に役立ちます。
- プラットフォーム、トランスポート プロトコル、およびアプリケーション サーバにわたる相関トランザクションの追跡。
- Web サービス コンポーネントにより発生したアプリケーション エラーまたは SOAP 障害を可視化およびコンテキスト表示。

これらのデフォルトの機能により、WSOA クライアントおよびサーバの監視およびアプリケーション管理が可能になりますが、さらに、独自のニーズに対応するために CA Introscope® および CA APM for SOA をカスタマイズできます。 たとえば、以下のことが可能になります。

- Web サービスの呼び出しの受信、完了を確認して、Web サービスの監査証跡を作成する
- Web サービスのアプリケーション グループを作成し、グループが、応答時間、トランザクション容量、またはエラーしきい値を超えているかどうかを確認する
- Web サービス クライアントおよびサーバに関して集約されたメトリックを収集するように、仮想エージェントをセットアップする

## Web サービス クライアントおよびサーバを監視する

ほとんどの場合、Web サービスのトランザクションは、Web サービスに要求を送信するクライアント、および要求に応答するサーバで構成されます。 たとえば、ユーザまたはプログラムが株価情報を要求できる Java クラスは、Web サービス クライアントと見なすことができます。 要求に応じて価格情報を提供する株価情報 Web サービスは、Web サービス サーバと見なされます。

エージェントは、クライアントとサーバのメトリックを別々に追跡して表示します。それにより、クライアント側およびサーバ側のパフォーマンスに関するしきい値を別々に監視および設定することが可能です。 また、クライアント側およびサーバ側が区別されるため、適切なコンテキストでの依存関係を参照することも可能です。

ほとんどの場合、Web サービスのクライアントとサーバは、個別の Java 仮想マシン（JVM）または Microsoft Common Language Runtime 環境（CLR）で実行されています。 したがって、クライアント側およびサーバ側オペレーションに関するメトリックは通常、2 つ以上のエージェントによってレポートされます。 クライアントとサーバが共に集約されたメトリックを確認したい場合、そのように仮想エージェント (P. 54)を構成します。

## SOA 固有のダッシュボードを使用してプロアクティブに管理する

CA APM for SOA により、実運用環境でクリティカルなサービスを 24 時間、リアルタイムで監視できます。SOA 固有のダッシュボードでは、重要なパフォーマンス インジケータが要約され、デフォルトのしきい値およびアラート インジケータが提供されます。その目的は、応答時間が長くなったときや、負荷の増加やスループットの低下のような異常なアクティビティがあったときに、そのような状況を早期に通知することです。 ダッシュボードおよびアラート インジケータにより、潜在的な問題をすばやくプロアクティブに特定できます。その後、問題を詳細に分析して、問題の性質や解決の担当者を決定できます。

CA Introscope® と共に CA APM for SOA のダッシュボードを使用すると、Web サービス クライアントおよびサーバのパフォーマンスを継続的に監視して、潜在的な問題をその進展に応じてひとめで検出できます。 多くの場合に、このプロアクティブなアプローチによって、顧客またはエンドユーザに影響を与える前に、障害のあるコンポーネントを識別して修正することができます。

## 依存関係の表示およびメトリックの使用により、問題を切り分けて診断する

こうした複雑な SOA 環境で問題が発生した場合、その発生場所をどこから探し始めるべきかが判断しにくいことがよくあります。 アプリケーション アーキテクチャはモジュール方式であるため、より多くのコンポーネントが各トランザクションにかかわっており、より多くの潜在的な障害点があります。

ダッシュボードおよびアラート インジケータは、潜在的な問題を早期に可視化します。SOA 依存マップは、モジュール方式のコンポーネントがどのように相関しているか、それらのリレーションシップがどのようにパフォーマンスに影響を与えているかを可視化します。 依存関係を使用することにより、潜在的なボトルネックまたはクリティカルなオペレーションがどこにあるかを分析し、相互に依存関係があるサービスについて、平均応答時間やそのほかのメトリックを比較できます。

俯瞰的なビューは、全体的なアプリケーション パフォーマンスを要約し、潜在的な問題を警告し、コンポーネント間のリレーションシップを示します。 CA APM for SOA を使用した CA Introscope® では、俯瞰的なサマリと詳細なオペレーション レベル両方のメトリックが提供されます。これらの情報を使用して、設計者や開発者は問題を確認し、切り分けて解決することができます。 メトリックを使用した Web サービス パフォーマンスおよびオペレーションの評価の詳細については、「Investigator を使用して SOA パフォーマンス メトリックを表示する (P. 64)」を参照してください。

## SOA コンポーネントが関わるトランザクションを追跡する

SOA 環境では一般的に、複数のプロトコルを使用して疎結合のコンポーネント間でメッセージが渡されます。そのため、トランザクション フローの追跡は一層難しくなります。 トランザクション追跡では、Web サービスがかかわる個々のトランザクションに関する詳細情報、Web サービスの障害の数と性質、コンポーネントのやり取りが提供されます。たとえば、トランザクション追跡は、トランザクションの各部分が実行された場所および実行状況をリアルタイムで特定します。その後、アプリケーション エキスパートは、それらの情報を使用して、エラーまたはパフォーマンス低下の根本原因を特定できます。

たとえば、サービスの応答時間が遅延するなどの問題が発生した場合、サービス レベル アグリーメント（SLA）違反が発生したり、ユーザに影響が及ぶ前に、アラートをしかるべき関係者に送ることができます。 その後、アプリケーション サポート担当者は、個々のトランザクションを監視して、最も時間のかかっているトランザクションの部分、および収集された各トランザクションを完了するための同期または非同期の呼び出しシーケンスを視覚的に特定することで、その問題の性質に関する追加の詳細情報を収集できます。

## 複数のプラットフォームおよびトランスポートにわたって監視する

CA APM for SOA を使用した CA Introscope® では、インフラストラクチャ全体を監視および管理できます。たとえば、その対象は複数のオペレーティング システム、アプリケーション プラットフォーム、またはトランスポート プロトコルにわたるトランザクションなどです。標準の SOA 環境では、トランザクションは一般的に SOAP、XML、HTTP、HTTPS および JMS トランスポート プロトコルの組み合わせを使用します。また、複数のアプリケーション サーバ環境にわたる処理が含まれます。

CA APM for SOA では、サポートされている J2EE または .NET サーバのあらゆる組み合わせにわたるトランザクションを追跡できます。 たとえば、WebLogic 上で実行されているサービスが SAP NetWeaver または .NET アプリケーション サーバ上で実行されているサービスに要求を送信している場合、そのようなトランザクションを最初から最後まで追跡できます。また、追加の SOA 拡張機能を有効にすることで、Oracle Service Bus、IBM WebSphere Process Server、IBM WebSphere Enterprise Service Bus、TIBCO BusinessWorks、webMethods Integration Server、IBM WebSphere MQ コンポーネントのあらゆる組み合わせにわたるトランザクションを監視することもできます。

さらに、SOA 環境では一般的に、アダプタを通じて SOA インフラストラクチャにリンクされる *複合アプリケーション*、および SOAP や XML のような標準化されたプロトコルを使用する新しいアプリケーションの両方が含まれます。 SOA 環境にはレガシー アプリケーションおよび新しいアプリケーションが含まれるため、さまざまなトランスポートおよびペイロード タイプを使用するアプリケーションを監視することが必要になる場合があります。たとえば、CA APM for SOA では、SOAP over HTTP、SOAP over JMS、XML over HTTP、またはそのほかの通信トランスポートおよびプロトコルを使用するアプリケーションを監視できます。

# SOA 監視用のアーキテクチャについて

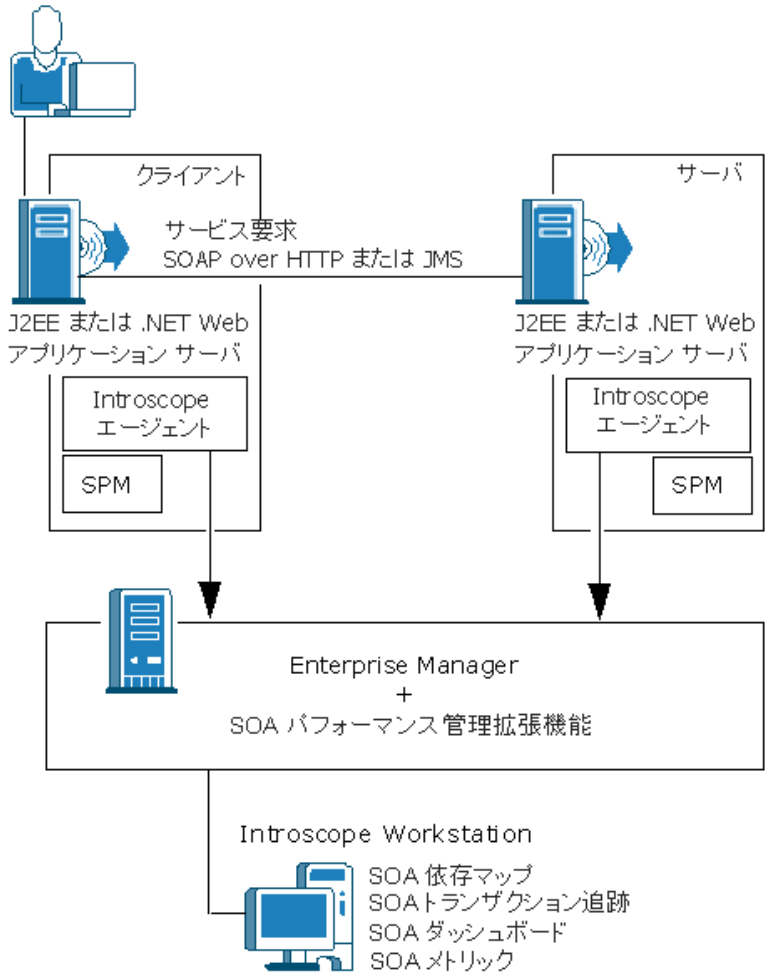
CA APM for SOA では、CA Introscope® 用の以下の軽量コンポーネントが用意されています。

- Java または .NET エージェント対応の SOA 固有の拡張機能
- Enterprise Manager 対応の SOA 固有の管理モジュール

エージェント拡張機能により、エージェントは、サポートされているアプリケーション サーバ上の Java と .NET ベースの Web サービスを監視し、Enterprise Manager にそれらの Web サービスのパフォーマンスに関する情報を送信できます。

Enterprise Manager 対応の SOA 固有の拡張機能により、エージェントによって収集された SOA 固有の情報は、CA Introscope® Workstation を使用して、SOA 固有の依存マップ、ダッシュボード、Investigator ノード、およびタブで表示可能になります。

以下の図は、2 つのアプリケーション サーバのためのエージェントおよび CA APM を備えた基本的なアーキテクチャを示しています。 アプリケーション サーバは、Web サービス要求の送信および受信を行い、Web サービス データを CA APM for SOA が有効な単一の Enterprise Manager にレポートします。

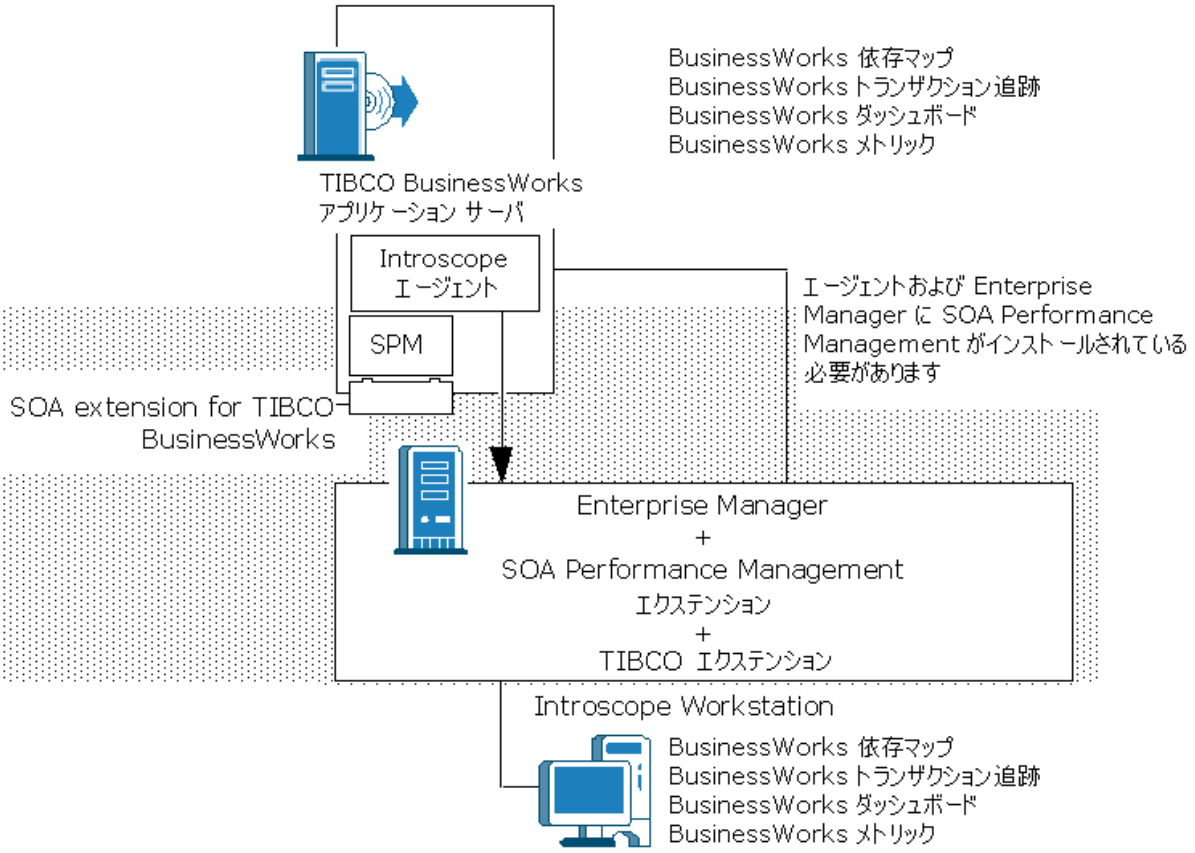


# そのほかの SOA プラットフォームのサポート

CA APM for SOA によって提供される機能に加えて、このほかにも監視できる SOA プラットフォームがいくつかあります。 組織の環境に応じて、以下の追加の SOA プラットフォームを監視することもできます。

- Oracle Service Bus （OSB）。Oracle WebLogic 用にメッセージ配信および変換を処理する Enterprise Service Bus です。
- TIBCO BusinessWorks。ビジネス サービスの開発および処理エンジンです。
- TIBCO Enterprise Message Service。キュー、トピック、拡張コンポーネント（ブリッジ、マルチキャスト チャネル、ルートなど）を使用して、エンタープライズ全体にわたり、システム間の同期および非同期通信が可能です。
- webMethods Broker。ドキュメントを公開および配信するための、非同期処理およびメッセージ通信処理サービスを提供します。
- webMethods Integration Server。組織でビジネス プロセスおよび Web サービスの作成、組み合わせ、統合が可能になる、インフラストラクチャ コンポーネントを提供します。
- IBM WebSphere Process Server （WPS） with WebSphere Enterprise Service Bus （WESB）。WebSphere Enterprise Service Bus を通じたメッセージ処理が含まれる、統合プラットフォームです。
- スタンドアロン製品としての IBM WebSphere Enterprise Service Bus （WESB）。 WebSphere Enterprise Service Bus は、メディエーション フローとプリミティブの支援により、サービスとクライアントとの間でメッセージ フロー、データ変換、ルーティングを管理します。

ほとんどの SOA プラットフォーム拡張機能では、固有のファイルがエージェントと Enterprise Manager に追加されます。たとえば、以下の図では、アプリケーション サーバおよび Enterprise Manager に SOA Extension for TIBCO BusinessWorks が追加されていることを示しています。




一部の SOA 拡張機能には、エージェントを使用できません。 たとえば、SOA Extension for TIBCO Enterprise Message Service は、スタンドアロン エージェントを使用します。 各 SOA 拡張機能の詳細については、その拡張機能に関する該当する章を参照してください。

# 第 2 章: CA APM for SOA をインストールして構成する

このセクションには、以下のトピックが含まれています。



## インストールおよび設定

CA APM for SOA は、以下の方法でエージェントと Enterprise Manager にインストールおよび設定することができます。

| エージェントがインストールされていますか? | CA APM for SOA が有効になっていますか? | 次の手順 |
|---|---|---|
| はい | はい | Enterprise Manager 上で拡張機能を有効にする (P. 44)。 |
| はい | いいえ | CA APM for SOA を手動でエージェントに追加する (P. 39)。 |
| いいえ | いいえ | CA APM for SOA の前提条件を確認する (P. 32)。<br>CA APM for SOA をエージェントに追加する (P. 36)。<br>**注**: インストールを完了するには、これらの情報を「CA APM *Java Agent 実装ガイド*」または「CA APM *.NET Agent 実装ガイド*」の情報と共に使用してください。 |

# CA Introscope® のコンポーネントおよびバージョン

CA APM for SOA では、サービス指向アーキテクチャおよび Web サービスの監視が可能になる、CA Introscope® 対応の拡張機能が提供されます。 CA APM for SOA は、CA Introscope® のコア コンポーネントと同時にインストールすることも、CA Introscope® のコア コンポーネントをインストールした後に個別にインストールすることもできます。 CA APM for SOA を CA Introscope® と一緒にインストールする場合も、CA APM for SOA を個別にインストールする場合も、CA Introscope® の以下のコンポーネントおよびバージョンをインストールします。

- Java エージェントまたは .NET エージェントは、CA Introscope® と同じバージョンである必要があります。
- Enterprise Manager は、エージェントと同じバージョンか、またはエージェントより高いバージョンである必要があります。

## エージェントの基本的なシステム要件

CA APM for SOA は、Enterprise Manager およびエージェントにファイルを追加します。 CA APM for SOA はエージェントの拡張機能であるため、処理、メモリ、使用可能なディスク空き容量の最小限のシステム要件は、エージェントの要件と同じです。

CA APM のシステム要件に関する固有の情報については、以下のガイドを参照してください。

- メモリおよびディスク空き容量の基本的な要件については、「CA APM インストールおよびアップグレード ガイド」を参照してください。
- 処理の負荷調整、および CPU のようなシステム リソースの管理に関するガイドラインについては、「CA APM サイジングおよびパフォーマンス ガイド」を参照してください。
- JVM と JRE の最小限のバージョン、および関連する JVM のメモリ要件については、「CA APM Java Agent 実装ガイド」を参照してください。

エージェントの基本的な要件に加えて、CA APM for SOA 対応のエージェントでは、サポートされている SOAP エンジンがローカルの運用環境で使用可能であることが必要です。 サポートされている SOAP エンジンについては、「*CA APM Compatibility Guide*」を参照してください。 使用する SOAP エンジンによって、SOAP スタックの実装、およびエージェントがサポートできる Web サービス メッセージ交換のタイプは異なります。

# エージェントに関する SOAP およびアプリケーション サーバの要件

使用するオペレーティング システムおよびアプリケーション サーバによって、Web サービス メッセージの処理に使用可能な SOAP エンジンおよび SOAP スタックの実装は異なります。 たとえば、Apache Axis 2.0 を使用するコンピュータ上のアプリケーションは、JAX-RPC または JAX-WS のいずれかを使用して、SOAP メッセージを処理できます。

## Java Agent によるネイティブ SOAP エンジンのサポート

Java Agent に CA APM for SOA を追加した場合、エージェントではさまざまな SOAP エンジンおよびアプリケーション プログラミング インターフェース（API）のサポートが可能になります。 アプリケーション サーバおよびそのバージョンに応じて、エージェントはネイティブ、JAX_RPC、または JAX -WS 標準に従う SOAP スタックの実装をサポートしています。

**注**: 現在のバージョンのエージェントでサポートされているアプリケーション サーバおよび SOAP スタック実装の完全なリストについては、「*Compatibility Guide*」の「*CA APM for SOA*」を参照してください。

## .NET Agent によるネイティブ SOAP エンジンのサポート

デフォルトでは、.NET エージェントは、ASP.NET Web サービスの監視をサポートしています。 .NET エージェントに CA APM for SOA を追加した場合、エージェントでは Web サービスの標準的な名前付けが可能であり、依存関係に関連する追加メトリックが提供され、使用中の SOAP 通信サービスに対応した依存関係ベースのマップも提供されます。.NET Framework を使用して、.NET エージェント上の Web サービスの監視をサポートするには、以下の Web サービスを使用します。

- 標準の ASP.NET Web サービス
- Windows Communication Foundation （WCF） Web サービス

.NET Framework に加えて、ASP.NET Web サービスを監視するには、以下の属性を使用します。

- サーバ側 Web サービスには WebMethod 属性を適用する必要があります。
- クライアント側の Web サービスには、SOAPDocumentMethod、HttpMethod、または SoapRpcMethod 属性を適用する必要があります。

**注**: SOAP のサポートの詳細については、「*Compatibility Guide*」の「SOA Performance Management」を参照してください。 これらの属性について、およびそれらの属性をメソッドやクラスに適用する方法については、Microsoft Developer Network（MSDN）ライブラリを参照してください。

## ディレクトリおよびファイルの命名規則について

本書では、ファイル名およびディレクトリ パスの表記で以下の規則を使用します。

| 表記 | 意味 |
|---|---|
| *<Agent_Home>* | エージェントがインストールされている最上位のディレクトリ。 通常、このディレクトリの名前は *wily* です。 |
| *<EM_Home>* | Enterprise Manager がインストールされている最上位のディレクトリ。 |
| *<バージョン>* | ファイル名に含まれているか、ユーザ インターフェースに表示される、バージョンに固有の識別子。<br>たとえば、ファイル名の形式は以下のようになります。<br>*com.wily.introscope.soa.dependencymap_<バージョン>.jar*<br>具体的なバージョンが付いたファイル名は以下のようになります。<br>*com.wily.introscope.soa.dependencymap_v9.1.0.0.jar* |

| 表記 | 意味 |
| --- | --- |
| スラッシュ（/）パス区切り記号 | 使用中のプラットフォームのディレクトリ名で使用されるパス区切り記号。<br>スラッシュ（/）は UNIX プラットフォーム、およびこのガイド全体にわたる例で使用されますが、使用中のプラットフォームで定められている区切り文字を使用する必要があります。 |
| ドル記号（$）環境変数 | プラットフォームで使用される環境変数の表記。<br>ドル記号（$）は UNIX プラットフォーム、およびこのガイド全体にわたる例で使用されますが、使用中のプラットフォームで定められている文字を使用する必要があります。 |

## 以前のバージョンからアップグレードする前のバックアップの作成

Web Services Manager または CA APM for SOA の以前のバージョンからアップグレードする場合は、CA APM for SOA エージェントまたは Enterprise Manager ファイルを追加する前に、現在の環境をバックアップし、バックアップ ディレクトリに既存のファイルを保存してください。

以前のバージョンからアップグレードする場合は、以下の手順に従います。

- バックアップ ディレクトリに既存の *<Agent_Home>*/IntroscopeAgent.profile ファイルをコピーします。
- バックアップ ディレクトリに既存の *<Agent_Home>*/webservices.pbd ファイルをコピーします。
- バックアップ ディレクトリに既存の *<EM_Home>*/modules/SPM_ManagementModule.jar ファイルをコピーします。

エージェントが .NET Agent である場合、以前のバージョンからアップグレードするには、以下の手順に従います。

- .NET Agent の新しいバージョンをインストールする前に、.NET Agent の以前のバージョンのアンインストールが完了していることを確認します。
- アップグレードする .NET エージェントと CA APM for SOA ファイルが同じバージョン レベルであることを確認します。バージョン情報に不一致があると、エラーが発生し、アプリケーションを監視できなくなる場合があります。

# CA APM for SOA をエージェントに追加する方法

エージェントを対話形式で、または応答ファイルを使用してサイレントモードでインストールするとき、エージェントに CA APM for SOA および SOA 関連の拡張機能を自動的に追加できます。エージェントのインストール後に、エージェントに CA APM for SOA およびそのほかの SOA 関連の拡張機能を手動で追加することもできます。

選択するインストールのタイプに応じて、適切なタスクを実行します。

- スタンドアロン エージェント インストーラで CA APM for SOA を選択する（P. 37）
- サイレント インストール中に CA APM for SOA を追加する（P. 38）
- 手動で CA APM for SOA をエージェントに追加する（P. 39）

エージェントをインストールまたは更新して CA APM for SOA を追加した後にアプリケーション環境を構成する方法については、「インストール後のエージェント プロパティ構成 (P. 41)」を参照してください。 Oracle Service Bus や TIBCO BusinessWorks のようなそのほかの SOA プラットフォームを監視する場合は、それらのファイルを対話形式で、または応答ファイルを使用してインストールできます。あるいは、エージェントをインストールし、CA APM for SOA を選択した後に手動でインストールすることもできます。詳細については、「SOA プラットフォーム拡張機能を有効にする方法 (P. 41)」を参照してください。

## スタンドアロン エージェント インストーラで CA APM for SOA を選択する

エージェントを対話形式でインストールする場合、インストール プロセスの手順内で、CA APM for SOA およびそのほかの監視拡張機能を追加することもできます。 ただし、スタンドアロン エージェント インストーラに表示される具体的なオプションは、選択するエージェントおよびアプリケーション サーバによって異なります。 例：

- CA APM for SOA を有効にする場合は、アプリケーション サーバとして［デフォルト］、［JBoss］、［Tomcat］、［WebLogic］、または［WebSphere］を選択します
- CA APM for Oracle Service Bus を有効にする場合は、アプリケーション サーバとして［WebLogic］を選択します
- CA APM for TIBCO BusinessWorks または CA APM for webMethods Integration Server を有効にする場合は、アプリケーション サーバとして［デフォルト］を選択します。
- CA APM for WebSphere Process Server with WESB またはスタンドアロン製品としての CA APM for WebSphere Enterprise Service Bus （WESB）を有効にする場合は、アプリケーション サーバとして［WebSphere］を選択します。

たとえば、Windows でスタンドアロン エージェント インストーラを対話形式で実行し、アプリケーション サーバとして［**デフォルト**］を選択する場合、監視オプションとして［CA APM for SOA］と、［CA APM for TIBCO BusinessWorks］または［CA APM for webMethods Integration Server］を選択できます。

スタンドアロン エージェント インストーラで［CA APM for SOA］オプションを選択する場合、エージェント用の関連するファイルが、エージェントのホーム ディレクトリ内の該当するディレクトリに自動的にコピーされます。また、SOA 環境を監視するための該当する ProbeBuilder ディレクティブ ファイルがエージェントのプロファイルに自動的に追加されます。 たとえば、*appmap-soa.pbd* および *webservices.pbd* は、*introscope.autoprobe.directivesFile* プロパティに自動的に追加されます。

スタンドアロン エージェント インストーラを使用して CA APM for SOA をインストールする場合は、Java Agent と .NET Agent を設定 (P. 41)するための以下の手順を実行します。

## サイレント モードを使用してエージェント用に CA APM for SOA を有効にする

サイレント モードでエージェントをインストールする場合、応答ファイルの設定を使用できます。 これらの設定では、エージェント用に CA APM for SOA を有効にするかどうかを指定します。 応答ファイルを使用することによって、リモートからユーザの操作なしにエージェント プロパティをインストールおよび設定できます。 この方法は、スクリプトやそのほかのソフトウェア配信オプションを使用してプロセスを自動化することを可能にします。

**注**: エージェントをサイレント モードでインストールする方法の詳細については、「CA APM Java Agent 実装ガイド」または「CA APM .NET Agent 実装ガイド」を参照してください。

**次の手順に従ってください**:

1. スタンドアロン エージェント インストーラと同じディレクトリにある SampleResponseFile.Agent.txt ファイルを開きます。
2. 指定したインストール設定を反映するように SampleResponseFile.Agent.txt ファイルを編集します。
3. shouldEnableSPM プロパティを true に設定して、CA APM for SOA をエージェントに追加します。 例：

   ```
   shouldEnableSPM=true
   ```

   このプロパティを false に設定した場合、SPM 固有のファイルが *<Agent_Home>*/examples ディレクトリに自動的にコピーされますが、エージェントのプロファイルは更新されません。

   応答ファイル内のプロパティを使用して、特定の SOA プラットフォームの監視を有効にすることもできます。 そのほかの SOA プラットフォームの監視を有効にする場合、shouldEnableSPM プロパティを true に設定します。
4. SampleResponseFile.Agent.txt ファイルを保存します。
5. コマンド ラインで適切なコマンドを入力して、インストーラを起動します。

   応答ファイルを使用して CA APM for SOA をサイレント モードでインストールする場合、インストール後にエージェント プロパティを構成します (P. 41)。

**詳細**:

CA Introscope® のコンポーネントおよびバージョン (P. 32)
SOA プラットフォーム拡張機能を有効にする方法 (P. 41)

## 手動で CA APM for SOA をエージェントに追加する

スタンドアロン エージェント インストーラで、またはサイレント インストール用の応答ファイルで、CA APM for SOA を有効にしなかった場合は、インストール後に手動でエージェントを更新できます。 手動でエージェントを更新して CA APM for SOA を追加する場合、エージェント プロファイルの修正が必要になります。 手動で更新する必要のあるエージェントのタイプに応じて、以下の該当するセクションを参照してください。

- Java Agent を手動で更新する（P. 39）
- .NET Agent を手動で更新する（P. 40）

### CA APM for SOA を使用するための Java Agent の手動更新

インストール後に、CA APM for SOA を使用するように、Java エージェントを手動で更新できます。*<Agent_Home>*/examples ディレクトリのファイルを *<Agent_Home>*/core/ext ディレクトリに移動します。 次に、CA APM for SOA 用の適切な ProbeBuilder ディレクティブ ファイルを含むようにエージェント プロファイルを編集します。

**次の手順に従ってください**:

1. *<Agent_Home>*/examples/SOAPerformanceManagement/ext ディレクトリに移動します。
2. このディレクトリからファイルをコピーし、*<Agent_Home>*/core/ext ディレクトリに貼り付けます。
3. アプリケーション サーバを停止します。
4. *<Agent_Home>*/core/config/IntroscopeAgent.profile ファイルをテキスト エディタで開きます。
5. spm.pbl の値を、IntroscopeAgent.profile ファイルの introscope.autoprobe.directivesFile プロパティに追加します。

   例：

   ```
   introscope.autoprobe.directivesFile=websphere-full.pbl,hotdeploy,spm.pbl
   ```

6. デフォルト設定を変更する場合は、IntroscopeAgent.profile ファイルでそのほかのエージェント プロパティを修正します。ファイルを保存し、テキスト エディタを終了します。
7. アプリケーション サーバを再起動します。

   アプリケーション サーバによってエージェントが再起動され、SOA パフォーマンス監視を有効にするようにアプリケーションがインスツルメントされます。

   手動で Java エージェント プロファイルを構成した後に、インストール後のエージェント プロパティ構成を行います (P. 41)。

## CA APM for SOA を使用するための .NET Agent の手動更新

インストール後に、CA APM for SOA を使用するように、.NET エージェントを手動で更新できます。 *<Agent_Home>*/examples ディレクトリから *<Agent_Home>*/ext ディレクトリにファイルを移動させ、CA APM for SOA ファイルを含めるようにデフォルト ProbeBuilder 一覧を編集します。エージェント プロファイルを設定した後に、インストール後のエージェント プロパティ構成を行います (P. 41)。

**次の手順に従ってください**:

1. *<Agent_Home>*¥examples¥SOAPerformanceManagement フォルダに移動します。
2. *<Agent_Home>*¥examples¥SOAPerformanceManagement¥ext フォルダのファイルを、*<Agent_Home>*¥ext フォルダにコピーします。
3. *<Agent_Home>*¥examples¥SOAPerformanceManagement¥wcf¥ext フォルダの wily.WCFServicesAgent.ext.dll ファイルを、*<Agent_Home>*¥ext フォルダにコピーします。
4. IIS アプリケーション サーバを停止します。
5. *<Agent_Home>*¥IntroscopeAgent.profile ファイルをテキスト エディタで開き、introscope.autoprobe.directivesFile プロパティ設定を確認します。

   例：

   ```
   introscope.autoprobe.directivesFile=default-full.pbl,hotdeploy
   ```

6. 変更を保存し、IntroscopeAgent.profile ファイルを閉じます。
7. introscope.autoprobe.directivesFile プロパティに指定した default-full.pbl または default-typical.pbl をテキスト エディタで開きます。

8. default*.pbl ファイルを保存し、閉じます。
9. 構成が完了したら、IIS サーバを再起動します。

   .NET エージェントは CA APM for SOA を使用するように更新されます。

## インストール後のエージェント プロパティ構成

インストール後に、必要に応じて、他のエージェント プロパティを設定し、アプリケーション サーバを更新できます。 たとえば、監視対象のコンポーネントまたは相関情報の処理を調整するために、エージェント プロファイルをカスタマイズできます。 アプリケーション環境に応じて、アプリケーション サーバの設定項目の設定、またはサーバ起動スクリプトの修正が必要になる場合もあります。

**注**: Java エージェントをインストールしている場合、アプリケーション環境を構成する方法の詳細については、「CA APM Java Agent 実装ガイド」を参照してください。.NET エージェントのインストールおよび構成、.NET 環境のプロパティの設定の詳細については、「CA APM .NET Agent 実装ガイド」を参照してください。 Java または .NET のエージェントが CA APM for SOA と連携するために必要なそのほかの特定の変更はありません。

## SOA プラットフォーム拡張機能を有効にする方法

CA APM for SOA 対応のエージェントを使用すると、以下の SOA プラットフォームのコンポーネントを監視できるようになります。

- AquaLogic Service Bus（ALSB）および Oracle Service Bus（OSB）
- IBM WebSphere Process Server（WPS） with WebSphere Enterprise Service Bus（WESB）、またはスタンドアロン製品として WebSphere Enterprise Service Bus（WESB）
- TIBCO BusinessWorks または TIBCO ActiveMatrix BusinessWorks（BW）
- Software AG webMethods Integration Server（IS）

アプリケーション サーバおよびエージェント環境に応じて、スタンドアロン エージェント インストーラを対話形式で実行するとき、応答ファイルを使用してエージェントをサイレント モードでインストールするとき、またはエージェントをインストールした後に手動で、これらの SOA プラットフォーム用の CA APM を有効にできます。

エージェントおよび CA APM for SOA に依存する SOA 関連の拡張機能に加えて、以下の SOA プラットフォームを監視するスタンドアロン拡張機能があります。

- TIBCO Enterprise Message Service（EMS）
- webMethods Broker

これらのスタンドアロン拡張機能はエージェントから独立してインストールおよび構成されます。

**注**: スタンドアロン拡張機能のインストールおよび構成については、該当するセクションを参照してください。

## SOA 関連の拡張機能を対話形式で追加する

スタンドアロン エージェント インストーラを対話形式で実行する場合、インストーラで有効にする SOA 関連の拡張機能を選択できます。 選択できる特定の監視オプションは、選択するアプリケーション サーバによって異なります。 たとえば、アプリケーション サーバとして［WebLogic］を選択する場合、［CA APM for SOA］と［CA APM for Oracle Service Bus］を選択できます。 アプリケーション サーバとして［デフォルト］を選択する場合、［CA APM for SOA］と、［CA APM for TIBCO BusinessWorks］、または［CA APM for webMethods Integration server］を選択できます。

インストーラでの選択に応じて、該当するファイルがエージェントのホーム ディレクトリの該当するディレクトリに自動的にコピーされ、該当する ProbeBuilder ディレクティブが、エージェントのプロファイルに自動的に追加されます。

## サイレント インストールの応答ファイルを使用して SOA 関連の拡張機能を追加する

エージェントをサイレント モードでインストールする場合、応答ファイルの設定を使用して、エージェント用の SOA 拡張機能を有効にするかどうかを指定できます。

**サイレント モードでプラットフォーム固有の SOA エージェント拡張機能を有効にする方法**

1. スタンドアロン エージェント インストーラと同じディレクトリにある *SampleResponseFile.Agent.txt* ファイルを開きます。
2. 指定したインストール設定を反映するように *SampleResponseFile.Agent.txt* ファイルを編集します。

3. *shouldEnableSPM* プロパティを *true* に設定して、CA APM for SOA をエージェントに追加します。 例：
   ```
   shouldEnableSPM=true
   ```
   CA APM for Oracle Service Bus（OSB）、CA APM for IBM WebSphere Process Server（WPS）、CA APM for WebSphere Enterprise Service Bus（WESB）、CA APM for TIBCO BusinessWorks（BW）、または CA APM for webMethods Integration Server を有効にする場合は、このプロパティを *true* に設定する必要があります。

4. エージェントに SOA 拡張機能を追加するには、該当する *shouldEnable** プロパティを *true* に設定します。 例：
   - *shouldEnableSOAExtForOSB=true* では、CA APM for Oracle Service Bus（OSB）が有効になります
   - *shouldEnableSOAExtForWPSandWESB=true* では、IBM WebSphere Process Server（WPS）と WebSphere Enterprise Service Bus（WESB）、またはスタンドアロン製品としての CA APM for WebSphere Enterprise Service Bus が有効になります
   - *shouldEnableSOAExtForTibcoBW=true* では、CA APM for TIBCO BusinessWorks が有効になります
   - *shouldEnableSOAExtForWebMethodsIS=true* では、CA APM for webMethods Integration Server が有効になります

   プロパティを *false* に設定する場合、拡張機能に固有のファイルは *<Agent_Home>/examples* ディレクトリにコピーされますが、エージェント プロファイルは更新されません。エージェントのインストール後に拡張機能を有効にする手順については、「エージェントのインストール後に SOA 拡張機能を手動で追加する (P. 44)」を参照してください。

5. *SampleResponseFile.Agent.txt* ファイルを保存します。
6. コマンド ラインで適切なコマンドを入力して、インストーラを起動します。

### エージェントのインストール後に SOA 関連の拡張機能を手動で追加する

スタンドアロン エージェント インストーラで、または応答ファイルで、SOA 関連の拡張機能を有効にしなかった場合は、インストール後に手動でエージェントを更新できます。 エージェントを手動で更新することには通常、以下の手順が含まれます。

- 該当する *<Agent_Home>/examples* ディレクトリからファイルを *<Agent_Home>/core/ext* ディレクトリにコピーします。
- プラットフォームに固有のエージェント拡張機能の該当する ProbeBuilder ディレクティブ（.pbd または .pbl）を含むようにエージェント プロファイルを構成します。
- エージェントを起動するようにサーバを構成するか、サーバを再起動します。

ただし、SOA プラットフォームに応じて、手動によるエージェントの更新には、プラットフォームに固有の追加の手順が必要になる場合があります。

**注**: プラットフォームに固有の手順については、有効にする SOA 拡張機能に関する該当するセクションを参照してください。

## Enterprise Manager 上で拡張機能を有効にする

Enterprise Manager をインストールすると、CA APM for SOA の管理モジュールとその他のファイル、および SOA 関連の拡張機能は、デフォルトでは *<EM_Home>*/examples ディレクトリにインストールされます。 たとえば、CA APM for SOA のファイルは、デフォルトでは *<EM_Home>*/examples/SOAPerformanceManagement ディレクトリにインストールされます。

任意の拡張機能を有効にするには、拡張機能の *<EM_Home>*/examples サブディレクトリにあるファイルを、使用している環境の *<EM_Home>* ディレクトリに移動し、拡張機能の管理モジュールをデプロイします。

**CA APM for SOA を Enterprise Manager に追加する方法**

1. Enterprise Manager を停止します。
2. *<EM_Home>*/examples/SOAPerformanceManagement ディレクトリに移動します。

3. *<EM_Home>*/config/modules ディレクトリに *<EM_Home>*/examples/SOAPerformanceManagement/config/modules ディレクトリの内容をコピーして貼り付けます。

   *<EM_Home>*/examples/SOAPerformanceManagement/config/modules ディレクトリには、CA APM for SOA の管理モジュール（SPM_ManagementModule.jar）があります。 クラスタ環境に複数の Enterprise Manager がある場合は、MOM コンピュータとして使用している Enterprise Manager の *<EM_Home>*/config/modules ディレクトリに、このファイルのみをコピーしてください。

4. *<EM_Home>/ext* ディレクトリに *<EM_Home>*/examples/SOAPerformanceManagement/ext ディレクトリの内容をコピーして貼り付けます。

   *<EM_Home>*/examples/SOAPerformanceManagement/ext/xmltv ディレクトリには、CA APM for SOA のタブ定義があります。これらのファイルを使用して、Workstation は SOA の概要、偏差、および依存関係のメトリックを表示します。

5. *<EM_Home>*/product/enterprisemanager/plugins ディレクトリに

   *<EM_Home>*/examples/SOAPerformanceManagement/product/enterprisemanager/plugins ディレクトリをコピーして貼り付けます。

   *<EM_Home>*/examples/SOAPerformanceManagement/product/enterprisemanager/plugins ディレクトリには、トランザクション追跡、偏差メトリック、および SOA 依存マップをサポートするファイルがあります。Enterprise Manager をクラスタとしてデプロイした場合は、*<EM_Home>*/examples/SOAPerformanceManagement/product/enterprisemanager/plugins のファイルを、クラスタ内のすべての Enterprise Manager 上の *<EM_Home>*/product/enterprisemanager/plugins にコピーします。 これらのファイルがコレクタ Enterprise Manager 上に見つからない場合、依存マップは適切に表示されません。

6. *<EM_Home>*/ws-plugins ディレクトリに *<EM_Home>*/examples/SOAPerformanceManagement/ws-plugins ディレクトリの内容をコピーして貼り付けます。

   *<EM_Home>*/examples/SOAPerformanceManagement/ws-plugins ディレクトリには、SOA 依存マップ、依存関係メトリック、およびトランザクション追跡をサポートする追加のファイルがあります。

7. Enterprise Manager を再起動します。

デフォルトでは、CA APM for SOA には、依存マップ内のリレーションシップをどれくらい頻繁にリフレッシュするか、どの偏差メトリックをレポートするかなど、オペレーションを制御する Enterprise Manager のプロパティ (P. 399)が含まれています。 ほとんどの組織にとってデフォルト設定は適切であり、変更する必要はありません。ただし、プロパティは変更できます。

そのほかの SOA プラットフォームの監視を有効にする場合は、最初に Enterprise Manager 上の CA APM for SOA を有効にします。

**Enterprise Manager にプラットフォームに固有の SOA 拡張機能を追加する方法**

1. Enterprise Manager を停止します。
2. 該当する *<EM_Home>*/examples ディレクトリに移動します。 たとえば、以下のディレクトリの 1 つに移動します。
   - SOAExtensionForOSB
   - SOAExtensionForTibcoBW
   - SOAExtensionForTibcoEMS
   - SOAExtensionForWebMethodsBroker
   - SOAExtensionForWebMethodsIS
   - SOAExtensionForWPSandWESB
3. *<EM_Home>*/config/modules ディレクトリに *<EM_Home>*/examples/*<extension>*/config/modules ディレクトリの内容をコピーして貼り付けます。

   *<EM_Home>*/examples/*<extension>*/config/modules ディレクトリには、SOA 拡張機能の管理モジュールがあります。 クラスタ環境に複数の Enterprise Manager が存在する場合は、MOM コンピュータとして使用している Enterprise Manager 上の modules ディレクトリにのみ、この管理モジュールをコピーしてください。

   あるいは、Enterprise Manager が実行されている場合は、*<EM_Home>/deploy* ディレクトリに管理モジュール ファイルをコピーできます。

4. *<EM_Home>*/ext/xmltv ディレクトリに *<EM_Home>*/examples/<extension>/ext/xmltv ディレクトリの内容をコピーして貼り付けます。

   *<EM_Home>*/examples/*<extension>*/ext/xmltv ディレクトリには、Workstation が拡張機能に固有のメトリックを表示できるようにするためのタブ定義が含まれています。

5. *<EM_Home>* ディレクトリの下で対応するディレクトリ内の追加のディレクトリの内容をコピーして貼り付けます。

   たとえば、*<EM_Home>*/product/enterprisemanager/plugins ディレクトリに *<EM_Home>*/examples/*<extension>*/product/enterprisemanager/plugins ディレクトリの内容をコピーして貼り付けます。

   **注**: Enterprise Manager をクラスタとしてデプロイした場合は、*<EM_Home>*/examples/SOAPerformanceManagement/product/enterprisemanager/plugins のファイルを、MOM およびクラスタ内のすべての Enterprise Manager 上の *<EM_Home>*/product/enterprisemanager/plugins にコピーします。

   追加のディレクトリは、有効にする SOA 拡張機能によって異なります。

6. Enterprise Manager を再起動します。

**注**: 特定の SOA プラットフォームの設定および操作の詳細については、そのプラットフォームに関する該当するセクションを参照してください。

# CA APM for SOA の展開を確認する

Web サービスまたはそのほかの WSOA コンポーネントを含むアプリケーションを起動し、WebServices ノードが Investigator ツリーに存在することを確認することにより、CA APM for SOA と SOA 関連の拡張機能の展開を確認できます。

**次の手順に従ってください**:

1. まだ実行されていない場合は、Enterprise Manager を起動します。
2. まだ実行されていない場合は、アプリケーション サーバを起動します。

   アプリケーション サーバを再起動することにより、エージェントが起動され、SOA パフォーマンス監視が有効になるようにアプリケーションがインスツルメントされます。

3. Web サービスを使用する任意のアプリケーションを起動し、1 つ以上のトランザクションを実行します。
4. Workstation を起動し、Investigator を開きます。
5. CA APM for SOA がインストールされている場所で、エージェントを展開します。
6. エージェントの下で［WebServices］ - ［Client］または［WebServices］ - ［Server］ノードにアクセスできることを確認します。

   同じエージェント上でトランザクションにクライアント側オペレーションおよびサーバ側オペレーションの両方が含まれる場合は、エージェント上で［WebServices］-［Client］および［WebServices］-［Server］ノードの両方を確認できる必要があります。

   検出された Web サービスを使用してトランザクションを実行していた場合は、サービスの［*ネームスペース*］ノードを展開して、トランザクションのクライアント側またはサーバ側に関連付けられたメトリックを表示できる必要があります。

   場合によっては、Web サービス オペレーションのデモンストレーション用にトランザクションをシミュレートするサンプル アプリケーションがメトリックを生成しない場合があります。 たとえば、WebLogic のテスト クライアントは WebLogic によってホストされたサービスに対して実行されていますが、それらのサービスは CA APM for SOA によって検出できません。
7. エージェント ノードを選択し、［概要］タブをクリックして、Enterprise Manager の *ws.overview.tv.xml* ファイルが適切にロードされていることを確認します。

# CA APM for SOA を削除する

お使いの環境から CA APM for SOA および SOA 関連の拡張機能を削除する場合は、以下の手順に従います。

- インストールして構成したエージェント拡張機能のすべてを削除します。
- 展開した Enterprise Manager 拡張機能を削除します。

### Java エージェントから CA APM for SOA を削除する方法

1. アプリケーション サーバを停止します。
2. <*Agent_Home*>/core/config ディレクトリに移動し、テキスト エディタで IntroscopeAgent.profile ファイルを開きます。
3. introscope.autoprobe.directivesFile プロパティから appmap-soa.pbd および webservices.pbd の値を削除します。 ファイルを保存して閉じます。
4. webservices.pbd を <*Agent_Home*> ディレクトリから削除します。
5. <*Agent_Home*>/core/ext ディレクトリに移動し、以下のファイルを削除します。
   ```
   WebServicesAgent.jar
   BoundaryOnlyTrace.jar
   ```
6. アプリケーション サーバを起動します。

   CA APM for SOA は Java エージェントから削除されます。

### .NET エージェントから CA APM for SOA を削除する方法

1. IIS アプリケーション サーバを停止します。
2. コマンド プロンプトを開き、*<Agent_Home>/ext* ディレクトリに移動します。 以下のファイルを削除します。
   - wily.WebServicesAgent.ext.dll
   - wilyHttpCorrelationTracers.ext.dll
   - wilyBoundaryOnlyTrace.ext.dll
3. IIS アプリケーション サーバを起動します。

   CA APM for SOA が .NET エージェントから削除されます。

### Enterprise Manager から CA APM for SOA を削除する方法

1. Enterprise Manager を停止します。
2. <*EM_Home*>/config/modules ディレクトリに移動し、以下のファイルを削除します。
   ```
   SPM_ManagementModule.jar.
   ```
3. <*EM_Home*>/ext/xmltv ディレクトリに移動し、以下のファイルを削除します。
   ```
   ws.overview.tv.xml
   ws.wsdependency.tv.xml
   ws.wsdeviation.tv.xml
   ```

4. <*EM_Home*>/product/enterprisemanager/plugins ディレクトリに移動し、以下のファイルを削除します。
   com.wily.introscope.soa.tracefilters.common_<*バージョン*>.jar
   com.wily.introscope.soa.tracefilters.em_<*バージョン*>.jar
   com.wily.introscope.soa.dependencymap.common_<*バージョン*>.jar
   com.wily.introscope.soa.dependencymap_<*バージョン*>.jar
   com.wily.introscope.soa.deviationmetrics_<*バージョン*>.jar
5. <*EM_Home*>/ws-plugins ディレクトリに移動し、以下のファイルを削除します。
   com.wily.introscope.soa.crossprocessviewer.workstation_<*バージョン*>.jar
   com.wily.introscope.soa.dependencymap.common_<*バージョン*>.jar
   *com.wily.introscope.soa.dependencymap.ui_<バージョン>.jar*
   com.wily.introscope.soa.dependencymetrics.typeviewer_<*バージョン*>.jar
   com.wily.introscope.soa.tracefilters.common_<*バージョン*>.jar
   com.wily.introscope.soa.tracefilters.workstation_<*バージョン*>.jar
   com.wily.introscope.ui.tomsawyer_<*バージョン*>.jar
6. Enterprise Manager を起動します。

   CA APM for SOA は Enterprise Manager から削除されます。

# 第 3 章: サービス指向アーキテクチャの監視

CA APM for SOA では、Web サービス クライアントとプロバイダに関する情報およびそれらの基盤となるオペレーションの表示により、SOA インフラストラクチャを監視できます。たとえば、プロセス依存関係、クリティカルなオペレーション、プロセス フロー、SOAP 障害などが表示されます。

このセクションでは、SOA インフラストラクチャを監視するためのデフォルトのダッシュボードおよびメトリックについて説明します。また、クライアントおよびサーバ オペレーションに関する SOA 固有の情報にアクセスする方法についても説明します。

このセクションには、以下のトピックが含まれています。



## CA Introscope® で SOA 固有の情報を表示する

CA APM for SOA のインストール後、Workstation および SOA 固有のダッシュボード、タブ、メトリックを使用して、Web サービス クライアントおよびプロバイダに関する情報を表示することにより、SOA インフラストラクチャを監視できます。

以下のリストでは、SOA 環境を監視するために Workstation に何が追加されるかを要約しています。

- SOA パフォーマンス ダッシュボードでは、展開したエージェントにわたって SOA 環境全体の稼働状況をひとめで監視できます。
- SOA 依存マップでは、SOA コンポーネント間の依存関係の視覚的な表示が可能であり、Web サービスおよびオペレーションが相互にどのように関連するかを理解するのに役立ちます。

- ［依存］タブでは、特定のサービスが直接または間接依存するクリティカルなオペレーションの数のような、依存関係メトリックを表示できます。
- ［偏差］タブでは、特定のネームスペースまたはオペレーションについて、平均応答時間からの偏差のような偏差メトリックを表示できます。
- ［クリティカル（上位）］タブでは、依存するオペレーションが最も多いオペレーションを表示できます。
- ［最も依存度が高い］タブでは、そのほかのオペレーションへの依存度が最も高いオペレーションを表示できます。
- ［エラー］タブでは、Web サービス オペレーションにより発生した SOAP 障害エラーに関する追加の情報が表示されます。

また、Investigator を使用して標準および SOA 固有のメトリックにドリルダウンして SOA 固有のコンポーネントに関する詳細情報を表示することもできます。トランザクション追跡ビューアを使用して、JVM または CLR の境界にまたがる SOA 関連のトランザクションを監視することもできます。

## クライアントおよびサーバの Web サービスとオペレーションについて

CA APM for SOA のインストール後にアプリケーション サーバを再起動すると、エージェントは、ディレクティブ ファイルの一覧で定義されたディレクティブに基づいて見つかるアプリケーションをインスツルメントします。 このプロセスの一部として、エージェントは、オペレーションを呼び出すために使用されたリモート プロシージャ コールまたは SOAP メッセージから、Web サービス *オペレーション*（メソッドの呼び出しに相当）を識別します。 各 Web サービス内の個々のオペレーションは、*クライアント*（アウトバウンド）要求または *サーバ*応答として識別され、エージェント アプリケーションの［WebServices］|［Client］または［WebServices］|［Server］ノードの下で表示されます。

ほとんどの場合、Web サービスのクライアントとサーバは、個別の Java 仮想マシン（JVM）または Microsoft Common Language Runtime 環境（CLR）で実行されています。 したがって、［Client］および［Server］ノードは多くの場合、個別のエージェントの下で一覧表示されます。 同じ JVM または CLR が要求と応答の両方を処理する場合、［Client］および［Server］ノードの両方はそのエージェントの［WebServices］ノードの下で一覧表示されます。 たとえば、クライアントとサーバが同じ JVM を使用する場合、Investigator ツリーでは、<*ホスト名*>|<*プロセス名*>|<*エージェント名*>|［WebServices］の下で［Client］および［Server］ノードの両方が表示されます。

## サービスのネームスペースおよびオペレーション名について

Investigator で、［Client］および［Server］ノードを展開すると、個々の Web サービスが Web サービスの*ネームスペース*に表示されます。 Web サービスのネームスペースは Web サービスの Uniform Resource Identifier（URI）の修正されたバージョンです。URI には、以下の 2 つのタイプがあります。

- Uniform Resource Locator（URL）
- Uniform Resource Name（URN）

Investigator に表示されるネームスペースは、URL 内のコロン(:)のような、予約された意味のある文字を置き換えるように修正されます。しかし、それ以外の点は Web サービスの URL または URN と同じです。 たとえば、標準のネームスペースは次のようになる場合があります。

```
http_//CreditManagement.demobank.ca.com
```

任意の Web サービスのネームスペースを展開して、そのオペレーションを表示できます。 Web サービスでのオペレーション名はメソッド名に相当します。そのため、アプリケーションで最低レベルにドリル ダウンして、低下したパフォーマンスを分析できます。たとえば、標準のオペレーション名が *getSmallBusinessAcctBalance* または *isAcctNumberValid* のように、実行される詳細なアクティビティを説明する場合があります。

## 未確認のサービスおよびオペレーションについて

場合によって、CA APM for SOA エージェントの拡張機能は Web サービスのネームスペースまたはオペレーションを識別できないことがあります。Web サービスの名前またはオペレーションを識別できないと、レポートされるメトリックおよびデータの集約に影響が出る場合があります。 たとえば、Investigator ツリーで、ネームスペースの代わりに UnknownService、またはオペレーション名の代わりに UnknownOperationName が表示される場合があります。

WebServices クライアントまたはサーバについて UnknownService または UnknownOperationName が表示される場合は、以下を確認します。

- 使用中のプロトコル、API、およびアプリケーション サーバ バージョンに対する Web サービスの実装
- 受信および送信メッセージの構成

## 仮想エージェントについて

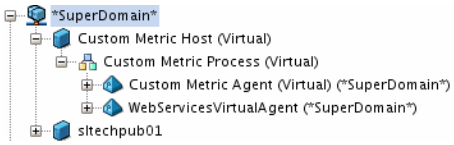
デフォルトでは、メトリックはエージェントごとに独立して表示されます。複数のエージェントにわたって、またはクライアントとサーバのネームスペースにわたって集約された Web サービスに関するメトリックを表示する場合は、*仮想エージェント*を定義する必要があります。 たとえば、Web サービスのクライアントとサーバの両方が同じ JVM を使用する場合、クライアントとサーバの両方が含まれる集約されたデータとして、Web サービスに関するメトリックが表示されるように、仮想エージェントを定義することをお勧めします。

エージェントのレポート先となる Enterprise Manager の *<EM_Home>/config/agentclusters.xml* ファイルを変更することにより、仮想エージェントを定義できます。 *agentclusters.xml* ファイル内では、集約されたデータとして表示するエージェントおよびメトリックを指定します。たとえば、すべてのサーバ側 Web サービスにわたって Average Response Time （平均応答時間） に関するデータを収集して集約するか、サーバ側とクライアント側の両方の Web サービスにわたって［Average Response Time （平均応答時間）］メトリックを組み合わせるように、仮想エージェントを設定できます。

以下の例では、すべての非カスタム エージェントにわたってサーバ メトリックを集約する、WebServicesVirtualAgent という名前の仮想エージェントを定義する方法を示しています。

```
<agent-cluster name="WebServicesVirtualAgent" domain="SuperDomain" >
    <agent-specifier>(.*)¥|[^Custom.*](.*)¥|(.*)</agent-specifier>
    <metric-specifier>WebServices¥|Server¥|(.*):Average Response Time
    ¥(ms¥)</metric-specifier>
    <metric-specifier>WebServices¥|Server¥|(.*):Concurrent
    Invocations</metric-specifier>
    <metric-specifier>WebServices¥|Server¥|(.*):Errors Per
    Interval</metric-specifier>
    <metric-specifier>WebServices¥|Server¥|(.*):Responses Per
    Interval</metric-specifier>
    <metric-specifier>WebServices¥|Server¥|(.*):Stall Count</metric-specifier>
    <metric-specifier>WebServices¥|Server¥|(.*):SOAP Faults Per
    Interval</metric-specifier>
</agent-cluster>
```

その後、WebServicesVirtualAgent ノードを展開して、集約されたサーバ メトリックを表示できます。 以下の図では、agentclusters.xml ファイルで定義された仮想エージェントが Investigator ツリーでノードとして表示されていることを示しています。




**注**: 仮想エージェントを構成する方法の詳細については、「CA APM 設定および管理ガイド」を参照してください。

# SOA パフォーマンス ダッシュボードの使用

CA APM for SOA には、SOA 環境全体の稼働状況を監視するために使用できる、事前構成されたダッシュボードが含まれています。 ダッシュボードは、エージェントにわたってデータを集約し、パフォーマンスおよび可用性の情報を要約して、潜在的な問題への高レベルの可視性を提供します。

通常、ダッシュボードは以下の機能を備えているため、環境を監視するための起点として使用されます。

- SOA インフラストラクチャの重要なコンポーネントの全般的な稼働状況、パフォーマンス、可用性、およびステータスをひとめで監視する。
- 低いレベルのメトリックによって警告しきい値や危険しきい値を超えたことが示されたときに、実運用アプリケーション環境に発生する可能性がある問題の通知を早めに取得する。
- パフォーマンス情報にドリルダウンして、どの Web サービス クライアント、プロバイダ、またはオペレーションが原因で遅延またはエラーが発生しているかを切り分けて特定する。

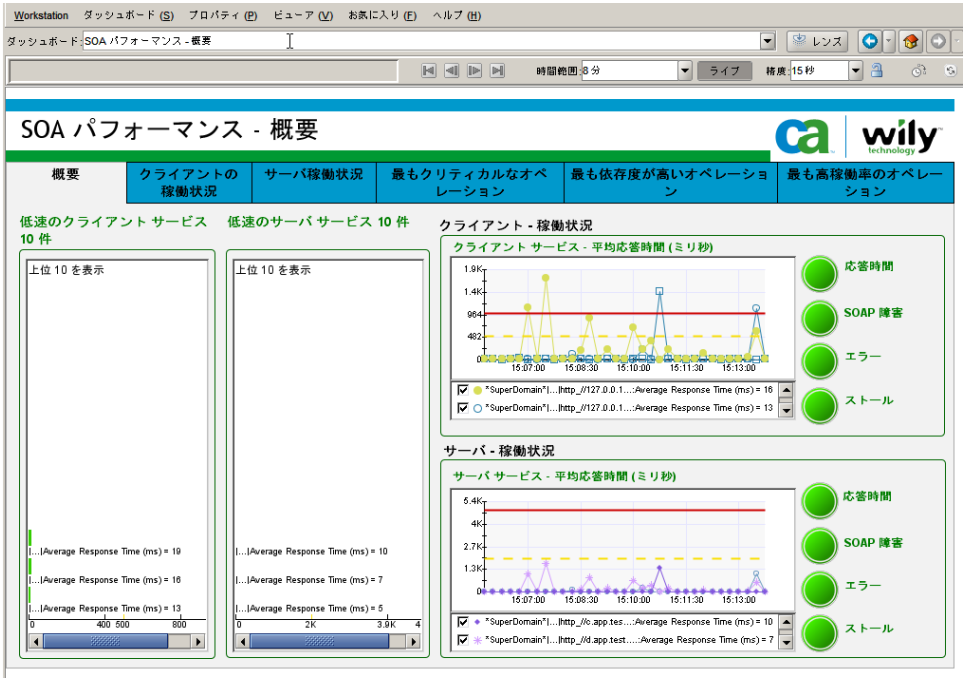
Enterprise Manager Extension for CA APM for SOA には、SPM_ManagementModule.jar ファイルにパッケージされた、事前設定された SOA 固有のダッシュボードが含まれています。

**次の手順に従ってください**:

1. Enterprise Manager を起動します（現在実行されていない場合）。
2. Workstation を起動し、SOA 拡張機能がインストールされている Enterprise Manager にログインします。
3. ［Workstation］-［新規コンソール］を選択します。

4. ［ダッシュボード］ドロップダウン リストから SOA パフォーマンスのダッシュボードの 1 つを選択します。

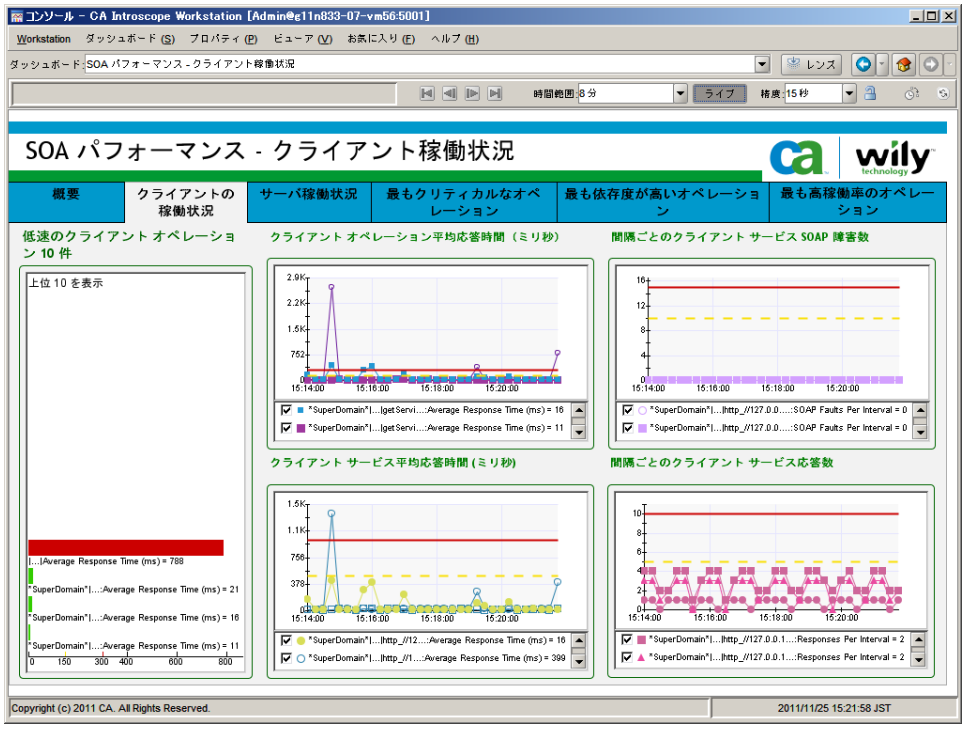
   たとえば、Web サービスを監視する開始点として、［SOA パフォーマンス - 概要］ダッシュボードを選択します。［SOA パフォーマンス - 概要］ダッシュボードには、展開したサービス全体の稼働状況に関する情報が表示されます。



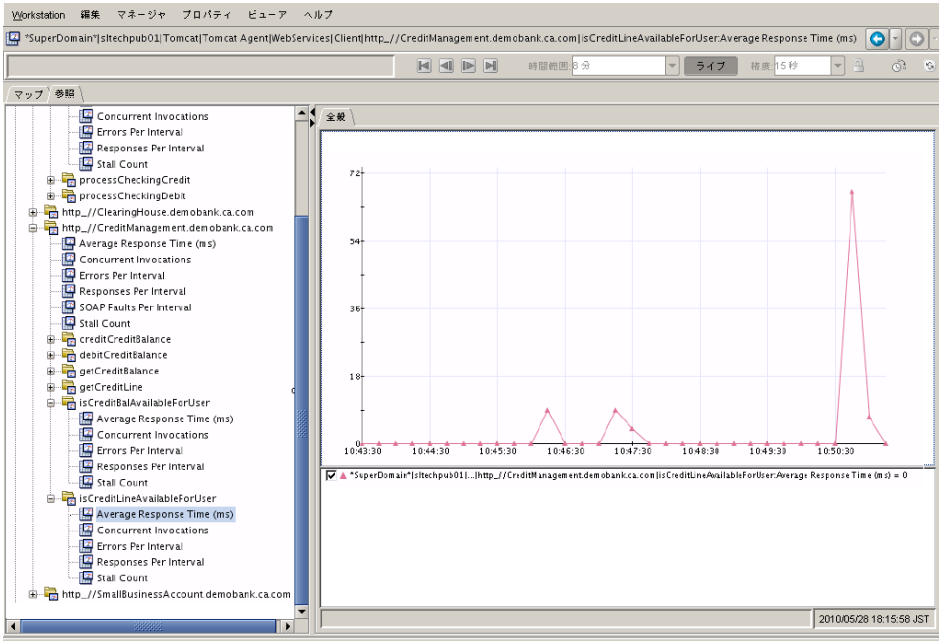

   ［SOA パフォーマンス - 概要］ダッシュボードから、以下のダッシュボードを操作して、クライアントまたはサーバ オペレーションに関するより詳細な情報を表示できます。

   - クライアント稼働状況
   - サーバ稼働状況
   - 最もクリティカルなオペレーション
   - 最も依存度が高いオペレーション
   - 最も稼働率が高いオペレーション

たとえば、その問題がクライアント側の平均応答時間であるように見える場合は、[クライアントの稼働状況]タブをダブルクリックして、最も遅いクライアント オペレーション、平均応答時間、1 間隔あたりの応答、および 1 間隔あたりの SOAP 障害に関する詳細をダッシュボードに表示できます。



5. 任意のダッシュボードから、サービスまたはオペレーション名をダブルクリックして、さらなる分析のために Workstation Investigator を開くことができます。たとえば、[クライアントの稼働状況] ダッシュボードで遅いサービスを選択する場合は、Investigator が開き、そのクライアント サービスに関するメトリックが表示されます。




**注**: Workstation の起動と使い方、ダッシュボードへのアクセス、または Investigator の起動と操作については、「CA APM Workstation ユーザ ガイド」を参照してください。

**詳細**:

Investigator を使用して SOA パフォーマンス メトリックを表示する (P. 64)

## [SOA パフォーマンス - 概要]ダッシュボード

[SOA パフォーマンス - 概要] ダッシュボードは、監視対象の Web サービスの概要を、クライアント側とサーバ側のサービス オペレーションに分けて表示します。

概要ダッシュボードには、以下の情報が含まれます。

- 最も遅いクライアントおよびサーバ サービスの一覧
- クライアントおよびサーバ サービスに関する平均応答時間のグラフ
- クライアントおよびサーバ サービスに関する応答時間、SOAP 障害、エラー、ストールを対象とするアラート インジケータ

このダッシュボードを使用して、監視中の Web サービスについて要約された情報を表示できます。

## [SOA パフォーマンス - クライアント稼働状況]ダッシュボード

[SOA パフォーマンス - クライアント稼働状況] ダッシュボードは、クライアント側の Web サービス オペレーションに関する詳細情報を表示します。

[クライアント稼働状況]ダッシュボードには、以下の情報が含まれます。

- 低速のクライアント オペレーション 10 件の一覧
- 各クライアント側オペレーションについて集約された平均応答時間
- 各クライアント側 Web サービスのすべてのオペレーションについて集約された平均応答時間
- 1 間隔あたりに各クライアント側 Web サービスについて集約された SOAP 障害の合計数
- 1 間隔あたりに各クライアント側 Web サービスについて集約された応答の合計数

監視中のクライアント側 Web サービスについて問題を警告された場合、このダッシュボードを使用して、クライアント側オペレーションに関する追加の詳細情報を表示できます。

## [SOA パフォーマンス - サーバ稼働状況]ダッシュボード

［SOA パフォーマンス - サーバ稼働状況］ダッシュボードは、サーバ側の Web サービス オペレーションに関する詳細情報を表示します。

［サーバ稼働状況］ダッシュボードには、以下の情報が含まれます。

- 低速の Web サービス オペレーション 10 件の一覧
- 各サーバ側オペレーションに関する平均応答時間
- 各サーバ側 Web サービスのすべてのメソッドまたはオペレーションについて集約された応答時間の平均
- 1 間隔あたりに各サーバ側 Web サービスについて集約された SOAP 障害の合計数
- 1 間隔あたりに各サーバ側 Web サービスについて集約された応答の合計数

監視中のサーバ側 Web サービスについて問題を警告された場合、このダッシュボードを使用して、サーバ側オペレーションに関する追加の詳細情報を表示できます。

## [SOA パフォーマンス - 最もクリティカルなオペレーション]ダッシュボード

［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボードには、依存度が高いオペレーションが最も多いオペレーションの偏差メトリックが示されます。 クリティカルなオペレーションはほかのオペレーションが成功するか失敗するかどうかに影響を与える可能性が最も高いため、これらのオペレーションを緊密に監視します。

［最もクリティカルなオペレーション］ダッシュボードには、依存度が高いオペレーションが最も多いオペレーションの一覧、および以下のグラフが含まれています。

- Average Response Time Deviation
- Errors Per Interval Deviation
- Responses Per Interval Deviation

高い偏差値は、クリティカルなオペレーションが確実に実行されていないか、ダウンストリーム オペレーションが正常に実行されていないことを示す場合があります。

CA APM for TIBCO BusinessWorks のような、そのほかの SOA プラットフォームの監視を有効にする場合、このダッシュボードには、拡張機能の最もクリティカルなプロセスに関する情報も含まれることがあります。

## [SOA パフォーマンス - 最も依存度が高いオペレーション]ダッシュボード

［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボードには、ほかのオペレーションへの依存度が最も高いオペレーションの偏差メトリックが示されます。

［最も依存度が高いオペレーション］ダッシュボードには、ほかのオペレーションへの最も依存度が高いオペレーションの一覧、および以下のグラフが含まれています。

- Average Response Time Deviation
- Errors Per Interval Deviation
- Responses Per Interval Deviation

高い偏差値は、アップストリーム オペレーションが確実に実行されていないことを示す場合があります。

CA APM for TIBCO BusinessWorks のような、そのほかの SOA プラットフォームの監視を有効にする場合、このダッシュボードには、拡張機能の最も依存度の高いプロセスに関する情報も含まれることがあります。

## [SOA パフォーマンス - 最も稼働率が高いオペレーション]ダッシュボード

［SOA パフォーマンス - 最も稼働率が高いオペレーション］ダッシュボードには、クライアント側とサーバ側に分類された、1 間隔あたりの最も稼働率の高い Web サービスに関する詳細情報が示されます。

［最も稼働率が高いオペレーション］ダッシュボードには、以下の情報が含まれます。

- 最も稼働率が高いクライアントおよびサーバ オペレーションの一覧
- すべてのクライアント側オペレーションに関する 1 間隔あたりの応答の合計
- すべてのサーバ側オペレーションに関する 1 間隔あたりの応答の合計

このダッシュボードを使用して、監視中のクライアントおよびサーバ Web サービスについて現在のスループットに関する追加の詳細情報を表示できます。

# Investigator を使用して SOA パフォーマンス メトリックを表示する

ダッシュボードで、クライアントとサーバの稼働状況の概要、および最もクリティカルまたは高稼働率のオペレーションのような潜在的な障害点を表示した場合は、Investigator により、問題の根本原因の特定に役立つ、SOA 環境に固有の詳細情報にドリル ダウンできます。

Investigator により、個々のサービス内の個々のオペレーションについて、アプリケーションの概要レベルのメトリックから詳細レベルのメトリックまで集約されたメトリックを分析して、パフォーマンス低下やエラー アクティビティの原因を見つけることができます。

## 利用可能なメトリック

監視対象の各*オペレーション*またはサービスの*ネームスペース*に関する CA Introscope® の標準メトリックに加えて、CA APM for SOA では、複数の SOA 固有のメトリックが提供されます。個々のオペレーションに関するメトリックは、個々の Web サービスに関するメトリックに集約され、アプリケーションのクライアント側およびサーバ側パフォーマンスを評価するために使用できます。

通常は、以下の標準および SOA 固有のメトリックが、Web サービスとして実装されているアプリケーションで利用可能です。

- **Average Response Time (ms)** - 15 秒の期間内に呼び出されたメソッド、オペレーション、またはサービスが完了するのにかかった平均時間（ミリ秒単位）
- **Average Response Time Deviation** - 間隔あたりの平均応答時間の平均値からの偏差
- **Concurrent Invocations** - 15 秒の期間の終了時に進行中だが完了していない同時進行中の要求の数
- **Errors Per Interval** - 15 秒の期間内に発生した例外、HTTP エラー、または SOAP 障害の数
- **Errors Per Interval Deviation** - 1 間隔内に発生した例外、HTTP エラー、または SOAP 障害の数について平均値からの偏差。
- **Responses Per Interval** - 15 秒の期間内に完了したオペレーションまたはサービス要求の数。

- **Responses Per Interval Deviation** - 1 間隔内に終了した呼び出しの平均値からの偏差
- **Stall Count** - 30 秒の期間の終了時に完了していない要求の数
- **SOAP Faults Per Interval** - 15 秒の期間内に SOAP エンジンによって生成または消費された SOAP メッセージ エラーの数。
- **Critical Direct** - 正常に実行されている現在のオペレーションに直接依存するダウンストリーム オペレーションの数
- **Critical Indirect** - 正常に実行されている現在のオペレーションに直接または間接依存するダウンストリーム オペレーションの数
- **Dependent Direct** - 現在のオペレーションが直接依存するアップストリーム オペレーションの数。
- **Dependent Indirect** - 現在のオペレーションが直接または間接依存するアップストリーム オペレーションの数

## ［概要］タブで要約されたメトリックを表示する

一般的に、Investigator を使用して問題の根本原因を探す開始点となるのは［概要］タブです。［概要］タブに表示される情報は、選択した Investigator ノードによって異なります。 たとえば、エージェント ノードを選択すると、［概要］タブにはエージェント上でインスツルメントされたすべてのアプリケーションの高レベルの稼働状況インジケータが表示されます。この［概要］タブは、Workstation ではデフォルトで表示されます。CA APM for SOA には、以下のようなそのほかの［概要］タブが用意されています。

- ［WebServices］ノード - ［WebServices］ノードを選択すると、［概要］タブには、選択したエージェント アプリケーションに関連付けられたクライアントまたはサーバ Web サービスのネームスペースのすべてが表示されます。 エージェントにクライアント Web サービスとサーバ Web サービスの両方がある場合、［概要］タブには、クライアントとサーバのネームスペースが別々の一覧に表示されます。
- ［Client］ノード - ［Client］ノードを選択すると、［概要］タブには、選択したエージェント アプリケーションのクライアント側オペレーションに関連付けられた Web サービスのネームスペースが一覧表示されます。

- ［Server］ノード - ［Server］ノードを選択すると、［概要］タブには、選択したエージェント アプリケーションのサーバ側オペレーションに関連付けられた Web サービスのネームスペースが一覧表示されます。
- *<web_service_namespace>* ノード - 個々の Web サービスのネームスペースを選択すると、［概要］タブには、選択した *<web_service_namespace>* によって識別されるクライアントまたはサーバ Web サービスに属するオペレーションが一覧表示されます。
- *<operation_name>* ノード - 個々のオペレーションを選択すると、［概要］タブには、選択した *<operation_name>* によって識別されるオペレーションについて、CA Introscope® の標準メトリックと［間隔ごとの SOAP 障害数］を使用して、アクティビティの現在のレベルを示すグラフが表示されます。

**次の手順に従ってください**:

1. Investigator を開き、エージェントのノードを選択し、［ビューア］ペインで［概要］タブをクリックします。

   ［概要］には、選択したエージェント上でインスツルメントされた各 Web サービスのアラート インジケータが表示されます。

2. エージェント ノードを展開して、Investigator ツリーで［WebServices］を選択します。 ［概要］タブをクリックします。

   クライアントおよびサーバ Web サービスのメトリックが別々のリストに表示されます。 以下のオプションがあります。

   - クライアントまたはサーバの表のいずれかで特定のサービスのネームスペースを選択します。

     そのサービスの Average Response Time （平均応答時間）のグラフが表示されます。

   - 表示されるグラフを変更するには、表示したいメトリックに対応するメトリック列ヘッダをクリックします。 たとえば、Errors Per Interval（間隔ごとのエラー数）のグラフを表示するには、［Errors Per Interval（間隔ごとのエラー数）］メトリック列ヘッダをクリックします。

3. ［WebServices］ノードを展開し、Investigator ツリーで［Client］または［Server］ノードを選択します。 ［概要］タブをクリックします。

   クライアント ネームスペースのリストまたはサーバ ネームスペースのリストのみが表示されます。

4. ［Client］または［Server］ノードを展開し、Investigator ツリーで特定の <*web_service_namespace*> を選択します。 ［概要］タブをクリックします。

   選択した Web サービスのネームスペースのオペレーションおよび関連するメトリックが表示されます。

5. <*web_service_namespace*> ノードを展開し、Investigator ツリーで <*operation_name*> を選択します。 ［概要］タブをクリックします。

   そのオペレーションのメトリックのグラフが表示されます。

## ［依存］タブで依存関係メトリックを表示する

CA Introscope® の標準メトリックに加えて、CA APM for SOA では、Web サービスのトランザクション内の最もクリティカルなオペレーションおよび最も依存度が高いオペレーションを調べるためのメトリックが提供されます。 これらのメトリックは、各オペレーションの直接的および間接的な依存関係の数に基づきます。

［依存］タブでこれらの依存関係ベースのメトリックを表示するには、該当する Investigator ノードを選択します。-

［依存］タブに表示される情報は、選択する Investigator ノードによって異なります。 たとえば、以下のようになります。

- ［WebService s ］ノード。［依存］タブには、選択したエージェント アプリケーションに関連付けられたクライアントおよびサーバ Web サービスのネームスペース、オペレーション、依存関係メトリックが表示されます。
- ［Client］ノード。［依存］タブには、選択したエージェント アプリケーションのクライアント側オペレーションに関連付けられた Web サービスのネームスペース、オペレーション、依存関係メトリックが表示されます。
- ［Server］ノード。［依存］タブには、選択したエージェント アプリケーションのサーバ側オペレーションに関連付けられた Web サービスのネームスペース、オペレーション、依存関係メトリックが表示されます。
- 個々の <*web_service_namespace*> ノード。［依存］タブには、選択した <*web_service_namespace*> によって識別されるクライアントまたはサーバ Web サービスに関連付けられたオペレーションおよび依存関係メトリックが表示されます。

## 直接および間接オペレーションの依存関係について

Web サービスのトランザクションでは、別のオペレーションを呼び出すオペレーションも*依存関係の連鎖*を作成します。依存関係の連鎖では、呼び出されたオペレーションは呼び出したオペレーションに依存します。 呼び出されたオペレーションからさらに呼び出しや応答が行われると、追加の依存関係が作成されます。 オペレーション間の依存関係は、オペレーションが依存関係の連鎖内のどこにあるかに基づいて、*直接*依存関係または*間接*依存関係のいずれかになる場合があります。

たとえば、オペレーション A がオペレーション B を呼び出し、オペレーション B がオペレーション C を呼び出す場合、トランザクションは以下のようになります。

A --> B --> C

このトランザクションの流れでは、

- オペレーション A と B には*直接*依存関係があります。
- オペレーション B と C には*直接*依存関係があります。
- オペレーション A と C には*間接*依存関係があります。

あるオペレーションは、依存関係の連鎖内のどこに位置するかに応じて、別のオペレーションから*アップストリーム*または*ダウンストリーム*で発生しているとして識別することも可能です。 複数のオペレーションで構成されるトランザクションで、*直接依存関係*にあるオペレーションは、相互に直接呼び出されるアップストリームまたはダウンストリーム オペレーションです。 上記の例で、オペレーション A はオペレーション B と C の両方からアップストリームにありますが、オペレーション B. からのみ直接的なアップストリームにあります。

同様に、*間接依存関係*は、直接呼び出されないオペレーションに対するアップストリームまたはダウンストリームで生じる場合があります。 たとえば、オペレーション A がオペレーションを B と呼び出し、オペレーション B がオペレーション C と D を呼び出す場合、C と D の両方にはオペレーション A に対するダウンストリームの間接依存関係があります。

## クリティカルなオペレーションとそのメトリックについて

Web サービス内のオペレーションは、他のオペレーションとのそれらのインタラクションに基づいて、*クリティカル*であるか、*依存関係*があると考えられます。あるいは、クリティカルであり、かつ依存関係があると考えられます。

*クリティカルなオペレーション*とは、ダウンストリーム オペレーションが依存するオペレーションです。 ダウンストリーム オペレーションはクリティカルなオペレーションに対して直接依存関係を持つ場合と、間接依存関係を持つ場合がありますが、ダウンストリーム オペレーションを実行するには、クリティカルなオペレーションが正常に動作している必要があります。

依存度が高いオペレーションに関連付けられた以下の 2 つのメトリックがあります。

- ［Critical Direct］メトリックは、特定のオペレーションへの直接アップストリームにあるオペレーションの数のみを追跡します。 オペレーションがどのアップストリーム オペレーションにも依存しない場合、つまり、オペレーションが他のオペレーションからのダウンストリームにない場合、その［Critical Direct］メトリックはゼロです。
- ［Critical Indirect］メトリックは、特定のオペレーションへの直接および間接アップストリームにあるオペレーションの数を追跡します。

たとえば、*getAccountNum* オペレーションについて［Critical Indirect］メトリック値が 5 である場合、*getAccountNum* オペレーションの実行前に実行する必要のある直接または間接オペレーションが 5 個あることを示します。

## 依存度が高いオペレーションとそのメトリックについて

*依存度が高いオペレーション*とは、アップストリーム オペレーションの実行に依存しているオペレーションです。 依存度が高いオペレーションはアップストリーム オペレーションに対して直接依存関係を持つ場合と、間接依存関係を持つ場合がありますが、依存するオペレーションを実行するには、アップストリーム オペレーションが正常に動作している必要があります。

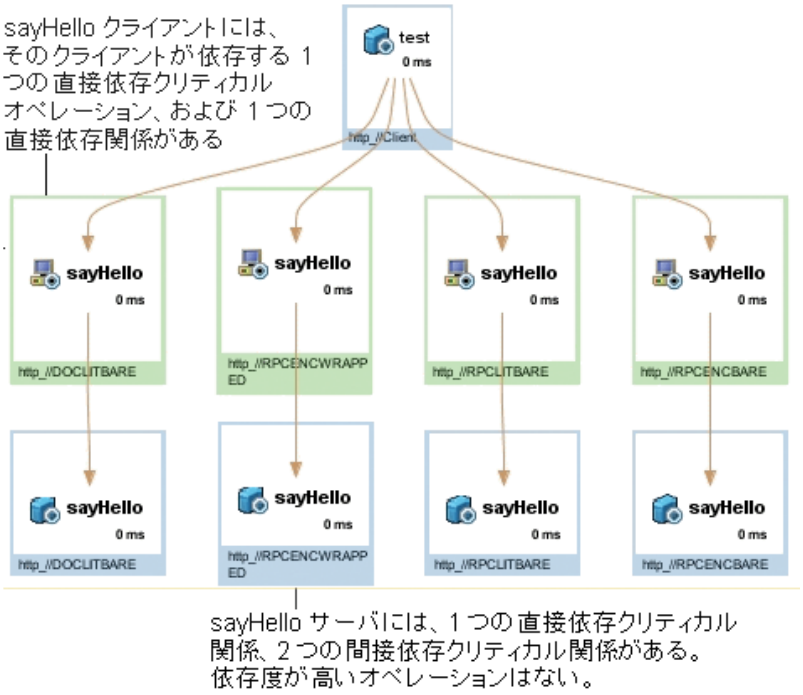
依存度が高いオペレーションに関連付けられた以下の 2 つのメトリックがあります。

- ［Dependency Direct］メトリックは、特定のオペレーションからの直接ダウンストリームにあるオペレーションの数を追跡します。 オペレーションに依存度が高いオペレーションがない場合、つまり、オペレーションが他のオペレーションからのアップストリームにない場合、その［Dependency Direct］メトリックはゼロです。
- ［Dependency Indirect］メトリックは、特定のオペレーションからの直接および間接ダウンストリームにあるオペレーションの数を追跡します。

たとえば、*sendConfirmation* オペレーションについて［Dependency Indirect］メトリック値が 8 である場合、*sendConfirmation* オペレーションの実行前に実行する必要のある直接または間接オペレーションが 8 個あることを示します。

## クリティカルなオペレーションおよび依存度の高いオペレーションに関する依存関係メトリックをより詳細に理解する

［Critical Direct］、［Critical Indirect］、［Dependency Direct］、および［Dependency Indirect］メトリックがどのように機能するかをより詳細に理解するために、4 つの *sayHello* クライアント オペレーションを呼び出す *Test* サーバ オペレーションの例を考えます。 その後、以下のサンプル依存マップで示しているように、各クライアント オペレーションは 1 つのダウンストリーム *sayHello* サーバ オペレーションを呼び出します。



この例では、クリティカルなオペレーションおよび依存度の高いオペレーションに関する依存関係メトリックの値は以下のように適用されます。

**Test サーバ**

アップストリーム オペレーションはありません。したがって、クリティカルなオペレーションに関する依存関係メトリックの値は、Critical Direct = 0 および Critical Indirect = 0 です。

4 つの直接ダウンストリーム オペレーションおよび 4 つの追加の間接ダウンストリーム オペレーションがあります。したがって、依存度が高いオペレーションの依存関係メトリックの値は、Dependency Direct = 4 および Dependency Indirect = 8 です。

**sayHello クライアント**

直接アップストリーム オペレーションがあります。したがって、クリティカルなオペレーションに関する依存関係メトリックの値は、Critical Direct = 1 および Critical Indirect = 1 です。

直接ダウンストリーム オペレーションがあります。したがって、依存度の高いオペレーションに関する依存関係メトリックの値は、Dependency Direct = 1 および Dependency Indirect = 1 です。

**sayHello サーバ**

Test サーバ上に sayHello へのアップストリームの直接および間接依存関係があります。したがって、クリティカルなオペレーションの依存関係メトリックの値は、Critical Direct = 1 および Critical Indirect = 2 です。

直接または間接ダウンストリーム オペレーションはありません。したがって、依存度が高いオペレーションの依存関係メトリックの値は、Dependency Direct = 0 および Dependency Indirect = 0 です。

**[依存]タブで依存関係メトリックを表示するには、以下の手順に従います。**

1. エージェント ノードを展開し、Investigator ツリーで［WebServices］を選択します。次に、［ビューア］ペインで［依存］タブをクリックして、クライアントとサーバの依存関係メトリックを別々の一覧に表示します。

   オペレーションごとに、［依存］タブには［Critical Direct］、［Critical Indirect］、［Dependency Direct］、および［Dependency Indirect］メトリックの値が表示されます。

2. ［WebServices］ノードを展開し、Investigator ツリーで［Client］または［Server］ノードを選択します。［ビューア］ペインで［依存］タブをクリックして、クライアントのネームスペースのみまたはサーバのネームスペースのみ、および関連する依存関係メトリックを表示します。

3. ［Client］または［Server］ノードを展開し、Investigator ツリーで特定の <*web_service_namespace*> を選択します。次に、［ビューア］ペインで[依存]タブをクリックして、選択した Web サービスのネームスペースについて、オペレーションおよび関連する依存関係メトリックの一覧を表示します。

## [偏差]タブで偏差メトリックを表示する

CA APM for SOA では、［Average Response Time］、［Errors Per Interval］、および［Responses Per Interval］について、平均値からの偏差を監視するためのメトリックが提供されます。 低い偏差値は、［Average Response Time （平均応答時間）］のようなメトリック データが平均値の周りに密集していることを示します。 高い偏差値は、メトリック データが平均値から上下に広範囲にわたって分散していることを示します。

**注**: オペレーションが間隔内に呼び出されなかった場合、Average Response Time メトリックの値は 0 （ゼロ）になります。 メトリック値が 0 の場合は、平均応答時間の計算に含まれていません。 そのため、［偏差］タブでは、監視が停止しているように見えることがあります。 この外見上の一時停止は正常です。

偏差メトリックを表示するには、Investigator のノードを選択し、［偏差］タブをクリックします。 例：

- ［WebServices］ノードには、選択したエージェント アプリケーションに関連付けられたクライアントおよびサーバ Web サービスのネームスペース、オペレーション、および偏差メトリックが表示されます。
- ［Client］ノードには、クライアントに関連付けられた Web サービス ネームスペース、オペレーション、および偏差メトリックが表示されます。
- ［Server］ノードには、サーバに関連付けられた Web サービス ネームスペース、オペレーション、および偏差メトリックが表示されます。
- 個々の <web_service_namespace> ノードには、選択したクライアントまたはサーバ Web サービスに関連付けられたオペレーションおよび偏差メトリックが表示されます。

**注**: デフォルトでは、偏差はネームスペースのレベルでのみ計算されます。［WebService］、［Client］、および［Server］ノードについて偏差を有効にするには、追加メトリックを計算するために com.wily.introscope.soa.deviation.metric.expressionlist (P. 432) プロパティを設定します。

## 偏差メトリックの計算

偏差メトリックは、CA APM for SOA が平均値を計算するために必要です。この製品は、ローリングする「n」日の期間にわたるオペレーション メトリックに基づいた平均値を決定します。 ローリングする平均が計算されるデフォルトの日数は 7 です。 日数を調整するには、com.wily.introscope.soa.deviation.mean.days プロパティ値を設定します。

Enterprise Manager のパフォーマンスへの影響を最小限に抑えるために、CA APM for SOA では、偏差メトリックを計算できる対象となるオペレーションの数が制限されています。 デフォルトでは、偏差メトリックは、各タイプの依存関係メトリックの上位 25 個のオペレーションについて計算されます。 偏差が計算されるオペレーションの最大数を変更するには、Enterprise Manager のプロパティ ファイルで com.wily.introscope.soa.deviation.count.per.metric プロパティを修正します。

偏差メトリックの生成に関するパフォーマンスのオーバーヘッドを削減するために、特定のタイプの偏差メトリックを選択的に有効または無効にすることもできます。たとえば、Average Response Time Deviation メトリックのみを生成するように構成プロパティを設定できます。 または、偏差メトリックを計算しないように、com.wily.introscope.soa.deviation.enable プロパティを false に設定します。

**詳細**:

Enterprise Manager プロパティの構成 (P. 411)

## [クリティカル（上位）]タブでクリティカルなオペレーションに関するメトリックを表示する

SOA 環境で最もクリティカルなオペレーションとは、そのオペレーションに依存するオペレーションが最も多いオペレーションです。 これらのオペレーションは最も重要な潜在的な障害点であり、Investigator の［最もクリティカルなオペレーション］ダッシュボードおよび［クリティカル（上位）］タブを通じて、緊密に監視する必要があります。 Investigator で［WebServices］ノードを選択すると、最もクリティカルなオペレーションの一覧、およびそれらの偏差メトリックを［クリティカル（上位）］タブで表示できます。

**次の手順に従ってください**:

1. エージェント ノードを展開し、Investigator ツリーで［WebServices］を選択して、［ビューア］ペインで［クリティカル（上位）］タブをクリックします。
2. クリティカルなオペレーションの一覧で個々のオペレーションを選択します。

## [最も依存度が高い]タブで依存度が高いオペレーションを表示する

SOA 環境で最も依存度が高いオペレーションとは、他のオペレーションへの依存関係が最も多いオペレーションです。 Investigator で［WebServices］ノードを選択すると、最も依存度が高いオペレーションの一覧、およびそれらの偏差メトリックを［最も依存度が高い］タブで表示できます。

**次の手順に従ってください:**

1. エージェント ノードを展開し、Investigator ツリーで［WebServices］を選択して、［ビューア］ペインで［最も依存度が高い］タブをクリックします。
2. 依存度が高いオペレーションの一覧で個々のオペレーションを選択して、そのオペレーションについて偏差メトリックのグラフを表示します。

## ［エラー］タブで SOAP 障害およびエラーに関するメトリックを表示する

Investigator では、Investigator ツリーで選択したノードについて、エラー メッセージおよび詳細が表示されるデフォルトの［エラー］タブが提供されます。 CA APM for SOA 拡張機能がインストールされている場合、［エラー］タブには、クライアント側およびサーバ側の SOAP 障害に関する情報も表示されます。

Errors Per Interval メトリックは、エージェントによって生成される CA Introscope® の標準メトリックです。［SOAP Faults Per Interval］メトリックは、CA APM for SOA エージェントの拡張機能によって生成される SOA 固有のメトリックです。 これらのメトリックの間に直接の相関はありません。

SOAP 障害は、Web サービスが正常応答ではなく障害応答を返すときに、SOAP エンジンによって生成または消費されるメッセージです。SOAP 障害は、条件に応じて自動的に生成できるか、アプリケーションのロジックで明示的にコード化できます。 障害がどのように収集されて処理されるかは、使用する SOAP エンジンによって異なります。

［Errors Per Interval］メトリックは、スローされた例外および HTTP エラーを追跡します。 Web サービスのビジネス ロジックおよびアプリケーション サーバの動作に応じて、SOAP 障害が発行される場合に、［Errors Per Interval］メトリックに対応するエラーが記録されるかどうかが決まります。

アプリケーション サーバで問題が発生し、SOAP 障害が発行される場合、クライアントとサーバの両方の［SOAP Faults Per Interval］メトリックにその障害が記録されます。アプリケーション サーバが SOAP 障害に関連する例外をスローする場合、その Errors Per Interval（間隔ごとのエラー数）の数も更新されます。 SOAP 障害を例外としてキャッチするコードがクライアント上にない場合、クライアントの Errors Per Interval メトリックは更新されません。

Errors Per Interval メトリックは、CA APM for SOA にインスツルメントされたアプリケーション サーバまたは SOAP スタック メソッドでキャッチされた例外およびエラー コードのみを追跡するため、メトリックに SOAP 障害が含まれない場合があります。

**次の手順に従ってください**:

1. エージェント ノードを展開し、Investigator ツリーで［WebServices］を選択するか、［WebServices］ノードを展開し、［Client］または［Server］ノードを選択します。
2. ［ビューア］ペインで［エラー］タブを右クリックします。

   ［エラー］タブの上部ペインに、各エラーの時間、説明、およびエラー メッセージが一覧表示されます。 SOAP エンジンによって生成されたエラー メッセージには、その先頭に *SOAP 障害*が表示されます。
3. 個々のエラーを選択して、そのエラーに関する詳細情報を表示します。

   SOAP 障害に関する詳細情報は、赤色で強調表示されたメッセージ テキストで、［スタック ビュー］タブに表示されます。

# Investigator で Boundary Blame を表示する

Boundary Blame は、Investigator ツリーにフロントエンドおよびバックエンド コンポーネントのみが含まれるように、メトリックを分離します。これは、応答時間の問題がアプリケーション サーバに対する内部の呼び出しの結果か、たとえば遅いサーブレットの結果か、データベース サーバのようなバックエンド システムに対する外部の呼び出しの結果かを判断するのに役立ちます。 CA APM for SOA は、バックエンド システムに対する Web サービスの呼び出しを自動的に検出することにより、SOA コンポーネントにまでこの機能を拡張します。

フロントエンドおよびバックエンド コンポーネントに関するメトリックを分離するためのこの拡張機能は、デフォルトで有効になっています。

**注**: アプリケーションに対して Boundary Blame を構成する、無効にする、またはカスタマイズする方法の詳細については、「CA APM Java Agent 実装ガイド」、「CA APM .NET Agent 実装ガイド」、および「CA APM Workstation ユーザ ガイド」を参照してください。

# デフォルトの SOA 固有のメトリック グループを表示する

CA APM for SOA には、デフォルトのダッシュボードおよびアラートの定義に使用されるデフォルト メトリック グループが含まれています。 また、カスタム ダッシュボードまたはアラートでデフォルト メトリック グループを使用することもできます。 デフォルト メトリック グループは、Enterprise Manager Extension for CA APM for SOA の一部として、SPM_ManagementModule.jar ファイルにパッケージされています。 デフォルトのメトリック グループは、Workstation の管理モジュール エディタを使用して表示できます。

**次の手順に従ってください**:

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］-［管理モジュール］-［SOA Performance Management (*Super Domain*)］を展開します。
3. ［メトリック グループ］ノードを展開して、SOA パフォーマンス管理の管理モジュール用に定義されたメトリック グループのすべてを表示します。
4. 特定のメトリック グループをクリックすると、［ビューア］ペインにその定義が表示されます。

任意のメトリック グループのデフォルト設定を修正するか、必要に応じて独自のカスタム メトリック グループを作成できます。

**注**: メトリック グループの作成および使い方の詳細については、「CA APM 設定および管理ガイド」を参照してください。

# デフォルトの CA APM for SOA アラートを表示する

CA APM for SOA には、事前設定されたダッシュボードで使用されるデフォルト アラート定義が含まれています。 デフォルト アラート定義は Enterprise Manager extension for CA APM for SOA の一部として、SPM_ManagementModule.jar ファイルにパッケージされています。

デフォルトのアラート定義は、Workstation の管理モジュール エディタを使用して表示できます。また、CA APM for SOA 管理モジュールを拡張して、カスタム アラート定義および通知タイプを含めるようにできます。カスタム ダッシュボードでデフォルト アラート定義を使用することもできます。

**次の手順に従ってください**:

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］-［管理モジュール］-［SOA Performance Management (*Super Domain*)］を展開します。
3. ［アラート］ノードを展開して、CA APM for SOA 用に定義されたアラートのすべてを表示します。
4. 特定のアラートをクリックすると、［ビューア］ペインにその定義が表示されます。

   特に、警告および危険しきい値のデフォルト設定を確認し、必要に応じて値を調整します。 また、通知または処置を追加することもお勧めします。

以下の表では、事前構成された CA APM for SOA のアラート定義に関連付けられたデフォルトの警告および危険しきい値を要約しています。

| アラート | 警告 | 危険 |
|---|---|---|
| クライアント オペレーションの平均応答時間<br>サーバ オペレーションの平均応答時間<br>サーバ サービスの平均応答時間 | 2000 ミリ秒 | 5000 ミリ秒 |
| クライアント サービスの平均応答時間 | 2500 ミリ秒 | 5500 ミリ秒 |
| クライアント オペレーションの Errors Per Interval（間隔ごとのエラー数）<br>クライアント サービスの Errors Per Interval（間隔ごとのエラー数）<br>サーバ オペレーションの Errors Per Interval（間隔ごとのエラー数）<br>サーバ サービスの Errors Per Interval（間隔ごとのエラー数） | 10 エラー | 15 エラー |

| アラート | 警告 | 危険 |
| --- | --- | --- |
| クライアント オペレーションの平均応答時間<br>サーバ オペレーションの平均応答時間<br>サーバ サービスの平均応答時間 | 2000 ミリ秒 | 5000 ミリ秒 |
| クライアント オペレーションの Responses Per Interval（間隔ごとの応答数）<br>クライアント サービスの Responses Per Interval（間隔ごとの応答数）<br>サーバ オペレーションの Responses Per Interval（間隔ごとの応答数）<br>サーバ サービスの Responses Per Interval（間隔ごとの応答数） | 5 応答 | 10 応答 |
| クライアント オペレーションの間隔ごとの SOAP 障害数<br>クライアント サービスの間隔ごとの SOAP 障害数<br>サーバ オペレーションの間隔ごとの SOAP 障害数<br>サーバ サービスの間隔ごとの SOAP 障害数 | 10 障害 | 15 障害 |
| クライアント オペレーションの Stall Count （ストール数）<br>クライアント サービスの Stall Count （ストール数）<br>サーバ オペレーションの Stall Count （ストール数）<br>サーバ サービスの Stall Count （ストール数） | 1 ストール | 2 ストール |

### CA APM for SOA と連携するエージェントのカスタム アラートを作成する方法

1. 管理モジュール エディタで、［エレメント］-［アラートを新規作成］をクリックします。
2. ［名前］フィールドに、アラート名を入力します。
3. ［管理モジュール］ドロップ ダウン リストから［SOA パフォーマンス管理］を選択し、［OK］をクリックします。

**注**: アラートの作成およびアラート通知のカスタマイズの詳細については、「CA APM 設定および管理ガイド」を参照してください。

# 第 4 章: SOA 依存マップの使い方

SOA 依存マップでは、SOA 環境で展開した Web サービスまたはビジネス プロセス ワークフローの視覚的な表示が可能です。 このマップを使用して、コンポーネント間のリアルタイムの依存関係を表示および操作したり、関連するコンポーネントにわたる重要なメトリックを監視したり、トランザクション追跡を開始したり、問題のサービスを探して依存関係にドリルダウンしたりできます。

このセクションには、以下のトピックが含まれています。



## SOA 依存マップの使い方について

標準の SOA 環境では、使用可能な Web サービスまたはビジネス プロセス間に複雑な関係があります。 ただし、それらのサービスが疎結合であるため、コンポーネントの関係は追跡または理解するのが通常は難しくなります。 SOA 依存マップでは、展開したサービスの視覚的な表示が可能であり、さまざまなコンポーネントが相互にどのように関連するかを監視および理解するのに役立ちます。

## SOA 環境の課題について

ほとんどの組織では、サービスの開発および展開には複数のグループがかかわり、通常は、サービスがどのように利用されているかを要約する単一のアーキテクチャ図はありません。 場合によって、展開したサービスを記録するレジストリがあることがありますが、その一覧が必ずしも最新であるとは限りません。 サービスのレジストリが適切に維持される環境でさえ、そのレジストリは場合によって、適切な担当者から容易に使用可能でなかったり、サービスの複雑な関係について正確に説明していなかったりします。

たとえば、アプリケーションにはさまざまなグループによって開発されている複数レイヤのサービスが埋め込まれていることがよくあります。 これらの追加のサービス レイヤは視覚的に表示するのが難しく、アプリケーション開発者が全容を把握していない場合があります。しかし、それらのサービスのパフォーマンスは、依存するアプリケーションに関するミッション クリティカルな問題になることがあります。 これらの「非表示の」サービスを視覚的に表示することにより、問題を特定し、切り分けて、解決するための追加のレベルの可視性が提供されます。

## SOA 依存マップで提供される情報について

SOA 環境の課題に取り組むために、自動および適時の方法で SOA 環境内のすべてのサービスを認識している必要があります。 また、コンポーネント間の関係を操作して、あるサービスが他のサービスによってどこでどのように使用されるか、潜在的なボトルネックがどこにあるかを把握することも必要です。 リアル タイムおよび実運用環境ですべてのサービスの稼働状況および可用性を監視する場合、この情報は重要です。

SOA 依存マップでは、この可視性が提供されます。依存マップを使用して、リアル タイムで以下のことをひとめで確認できます。

- どのようなサービスが実際にデプロイされているか
- いつ新しいサービスが展開されるか、依存関係が変更されるか
- どのようにサービスが相互に関連しているか
- どのようにクライアント側要求およびサーバ側応答が実行されているか

## 依存関係および SOA 用語について

SOA 依存マップのコンテキスト内では、*サービス*という用語は、特定の結果を得るために設計された一連のオペレーションを表すために、一般的に使用されます。 サービスのコンポーネントに関する詳細は環境によって異なり、さまざまな環境で個別に定義できます。 たとえば、依存マップの［サービス］ビューを選択すると、Web サービス、ビジネス プロセス、アダプタ、またはメッセージ サービスに関する情報が表示される場合があります。

1 つのオペレーションの実行により、後続のオペレーションが起動されるとき、依存関係が存在します。後続のオペレーションはまったく別のサービスに属していたり、異なるアプリケーション サーバ上で実行されたりする場合があります。 たとえば、旅行代理店 Web サイトにアクセスするユーザは、フライトを予約するトランザクションを開始できます。 旅行代理店 Web サイトの予約サービスは、座席を予約するために航空会社予約サービスを呼び出す場合があり、その後、チケットを購入するためにクレジット カード処理サービスを呼び出す場合があります。 この例では、トランザクションを開始した旅行 Web サイト上のサービスは、トランザクションを完了するために、航空会社予約サービスおよびクレジット カード処理サービスへの依存関係があります。

依存関係は多くの場合、低レベルのサービスを組み合わせて、より高レベルのサービスが形成されることにより生じます。 たとえば、旅行代理店 Web サイトの予約サービスは、正常に処理を完了するために、航空会社予約サービスおよびクレジット カード処理サービスへの依存関係があります。 それらのいずれかのサービスで障害またはパフォーマンスの問題が発生すると、旅行代理店 Web サイトの予約サービスの動作に影響が出ます。

選択するコンテキストに応じて、SOA 依存マップには、展開したトップレベルのサービスのすべてが表示されます。あるいは、個々のサービスの詳細な構造が、依存度が高いオペレーションの依存関係およびパフォーマンス情報と共に表示されます。 表示される範囲および詳細を変更できるため、SOA 依存マップでは、SOA 環境での問題を分析するための優れた柔軟性が提供されます。 プロセス フロー内でのサービスおよびオペレーション間の依存関係を追跡できます。それらの情報は、遅い応答時間、SOAP 障害などの問題の根本原因を特定するのに役立ちます。

## SOA 依存マップで可能なことについて

SOA 依存マップを使用して次の重要なタスクを達成できます。

- 設計対象ではなく展開対象のサービスに関する最新の正確な情報を表示する。
- サービスの依存関係を表示し、マップ ノードをクリックして、Investigator ツリーでコンポーネントに関する詳細にドリル ダウンすることにより、サービスの問題を調査および解決する。
- アプリケーションの依存関係の［物理モード］および［論理モード］ビューの両方を提供することにより、IT アーキテクトと連携して企業規模の SOA イニシアチブを計画し、それに基づいて行動する。
- マップのスナップショット画像を作成して、サービスの依存関係および問題に関する情報を組織内の他の関係者と共有する。
- アプリケーション エキスパートのサポートを必要とせずに、監視対象のエージェント、サービス、オペレーションの依存関係を理解し、それらの依存関係がサービスのパフォーマンスにどのように影響を与えているかを把握する。
- 問題のサービスについて、手動によるトランザクション追跡をセットアップせずに、SOA 依存マップから直接、トランザクション追跡を開始して、その問題の診断を開始する。

# SOA 依存マップがデータを取得する仕組みについて

最初に CA APM for SOA をインストールし、アプリケーション サーバを起動すると、エージェントは見つかった Web サービスまたはビジネス プロセスのすべてを自動的に検出し、その情報を Enterprise Manager に送信します。

これらのサービスが起動され実行されると、エージェントはプロセスにわたって呼び出しのシーケンスを追跡し、この情報を Enterprise Manager に渡します。 Enterprise Manager は、呼び出しのシーケンスの情報を使用して、サービス間の依存関係を識別します。 Enterprise Manager は、検出された依存関係をファイルに保存し、マップ データを表示用に Workstation に送信します。

最初の検出後、エージェントは新しいアプリケーションについて、および以前に検出された依存関係に対する変更について定期的に確認します。新しいサービスが検出された場合、エージェントはサービスの呼び出しに関する詳細を Enterprise Manager に送信します。Enterprise Manager は、Workstation に新しいマップ データを送信します。それにより、以前に不明だったサービスを使用して、または既知のサービスについては新たに検出された依存関係を使用して、SOA 依存マップを更新します。

新しい SOA 依存関係が検出された場合、Enterprise Manager は論理等価物ルールが有効になっているかどうかを確認します。有効になっている場合は、新しい依存関係が既存の依存関係と論理的に等価かどうかを判断します。 新しい依存関係が一意である場合、それらは Enterprise Manager 上で保存された依存関係の永続データに追加されます。Workstation は 15 秒に 1 回、データ ストアをチェックして、SOA 依存マップのデータが変更されたかどうかを確認します。その後、表示された SOA 依存マップを必要に応じて更新します。

## SOA 依存マップの永続データについて

Enterprise Manager は検出された SOA 依存関係をローカル データ ストアに保存することにより、再検出を必要とせずにそのデータが使用可能になるようにします。 SOA 依存関係のデータ ストアは、<*EM_Home*>/data/dependencymap ディレクトリ内の以下のファイルで構成されます。

- dependencymap.sav には、最新に保存された依存関係データが含まれます。
- dependencymap.bak は、以前に保存された SOA 依存関係のバックアップ コピーです。

Enterprise Manager は、検出された依存関係のすべてを dependencymap.sav ファイルに保存します。この処理を毎時に 1 回、および Enterprise Manager の手動シャットダウンのたびに行います。 Enterprise Manager は、dependencymap.sav ファイルの保存間で、SOA 依存マップ内の現在検出されている依存関係をメモリに保持します。

Enterprise Manager は起動時、dependencymap.sav ファイルから依存関係をロードすることにより、CA APM for SOA が以前に検出された依存関係を再検出する必要がないようにします。 dependencymap.sav ファイルの新しいコピーを保存せずに、Enterprise Manager が突然停止した場合、dependencymap.sav ファイル内のデータは 1 時間前までのデータです。 dependencymap.sav ファイルが見つからない場合、Enterprise Manager は、エージェントがサービス依存関係を再検出するまで待機し、その後、このファイルを再作成します。

## Enterprise Manager による依存関係の再検出を強制する

ほとんどの場合、*dependencymap.sav* ファイルからの永続データのロードによって再検出なしで最新の情報が提供されます。 ただし起動時に、Enterprise Manager が以前に保存された依存関係を無視し、すべての依存関係を再検出するようにする場合があります。 例：

- 論理等価物のヒューリスティックを有効または無効にする場合、Enterprise Manager がすべての依存関係を再検出して、依存関係に新しいルールを適用するようにすることもできます。論理等価物のヒューリスティックの詳細については、「論理等価物ルールについて (P. 90)」を参照してください。 論理等価物のヒューリスティックのプロパティの設定の詳細については、「Enterprise Manager プロパティの構成 (P. 399)」を参照してください。
- Enterprise Manager の停止中にエージェントを切断する場合、期限切れになるまで（デフォルトでは 60 日間）、そのエージェントに対応するマップ ノードおよび依存関係がマップに表示され続けることがあります。期限切れになるのを待たずに、Enterprise Manager が依存関係を再検出して、切断されたエージェントのデータを依存マップから削除するようにすることもできます。

## 保存された依存関係データを無視または削除する

起動時に Enterprise Manager が保存された依存関係データを無視し、依存関係を再検出するようにする場合は、*dependencymap.sav* と *dependencymap.bak* の両方のファイルを削除、移動、名前変更する必要があります。

**保存された SOA 依存マップ ファイルを削除、移動、名前変更するには、以下の手順に従います。**

1. Enterprise Manager を停止します。
2. *<EM_Home>/data/dependencymap* ディレクトリに移動します。
3. *dependencymap.sav* ファイルを削除、移動、または名前変更します。たとえば、ファイルの名前を *old_dependencymap.savedcopy* に変更して、Enterprise Manager がこのファイル内のデータを使用して依存マップを表示できないようにします。
4. *dependencymap.bak* ファイルを削除、移動、または名前変更します。たとえば、ファイルの名前を *old_dependencymap.backupcopy* に変更して、Enterprise Manager がこのファイルを使用して依存マップを表示できないようにします。
5. Enterprise Manager を起動します。これにより、現在の依存関係が再検出され、新しい依存マップおよび *dependencymap.sav* ファイルが作成されます。

### エージェントを切断および再マウントする

Enterprise Manager の動作中にエージェントが Enterprise Manager から切断される場合、そのエージェントに対応するマップ ノードおよび依存関係は非アクティブになります。 Workstation からエージェントをマウント解除する場合、マウント解除イベントでは、そのエージェントに対応するマップ ノードおよび依存関係のすべてがマップから削除されます。たとえば、削除されたサービスから開始する依存関係や、そのサービスで終了する依存関係も削除されます。

エージェントが再マウントされると、Enterprise Manager はその依存関係のすべてを再検出し、それらをマップに追加します。

## 新しいサービスおよびオペレーションを使用して依存マップを更新する

エージェントは新しいサービスまたはオペレーションを検出するたびに、更新した情報を Enterprise Manager に送信します。Enterprise Manager は依存マップ ストアを更新し、新しいデータが SOA 依存マップ上で表示可能になったことを Workstation に通知します。Workstation がこの通知を受け取ると、「利用可能なデータが変更されました」というメッセージが表示されます。 その後、Workstation で、依存マップを再ロードして、新しいサービスまたはオペレーションを表示できます。

**SOA 依存マップを再ロードするには、以下の手順に従います。**

1. ［SOA 依存マップ］ツール バーの下に「利用可能なデータが変更されました」というメッセージが表示されているかどうかを確認します。
2. ［ここをクリックします］を使用して依存関係を再ロードし、新しいサービスまたはオペレーションおよび関連するメトリックを表示します。 たとえば、再ロード後、「利用可能なデータが変更されました」というメッセージが消えて、新しく検出されたサービスおよび依存関係がマップに含まれています。

依存マップの再ロード時に表示されるビューは、最後にクリックした Investigator ツリー ノードに関連付けられたデフォルトのビューです。 たとえば、［WebServices］ノードを選択し、マップを再ロードする場合、マップはサービスの［物理モード］ビューに表示されます。

コンテンツ タイプを変更し、「利用可能なデータが変更されました」というメッセージを確認し、マップを再ロードした場合、マップはサービスの［物理モード］ビューで表示されます。そのビューが［WebServices］ツリー ノードのデフォルトのビューであるためです。 Investigator ツリーで別のノード、たとえばエージェント ノードをクリックし、マップを再ロードした場合、マップはエージェントの［物理モード］ビューに表示されます。

## 使用されなくなったマップ ノードのエイジングおよび削除を行う

Enterprise Manager は検出された各依存関係の経過期間を追跡します。そして定期的に依存関係を再検出して、依存関係がまだ存在するかどうかを確認します。 特定のエージェント、サービス、またはオペレーションのマップ ノードが、一定期間再検出されない場合、その項目は期限切れになり、依存マップから削除されます。 削除されたエージェント、サービス、またはオペレーションが後でもう一度呼び出される場合、Enterprise Manager は依存マップにもう一度その項目を追加します。

デフォルトでは、Enterprise Manager は 6 時間ごとに依存関係の最新の検出があるかどうかを確認し、60 日間検出されていない依存関係を削除します。

Enterprise Manager が検出された依存関係の有効期間をチェックする頻度、および再検出されないときに期限切れになるまで依存関係をマップに残す期間を設定できます。そのためには、*IntroscopeEnterpriseManager.properties* ファイル内のプロパティの値を設定します。 Enterprise Manager のプロパティの設定の詳細については、「Enterprise Manager プロパティの構成 (P. 399)」を参照してください。

## 依存マップに不完全なデータが表示される場合の処置

ほとんどの場合、サービスおよび依存関係の最初の検出は完了までに時間がかかります。エージェントが情報を収集し、Enterprise Manager に提供する必要があるためです。 最初の検出の後、サービスおよび依存関係に対する変更はすぐに表示されるとは限りません。 通常、スタンドアロンで Enterprise Manager を実行している場合は、変更が表示されるまでに数分程度かかります。 クラスタ上で Enterprise Manager を実行している場合は、それよりも若干長くかかることがあります。

ただし、エージェントがすべてのノードおよび依存関係に関するデータを収集するためには、サービスが実行されているか、アクティブに実行されている必要があります。 呼び出されたことのないサービスおよびオペレーションは呼び出されるまでマップに表示されません。

依存マップにデータが表示されないか、不完全なデータが表示される場合、以下のような理由が考えられます。

- 監視中のアプリケーション サーバまたはサービスがアクティブに実行されていないか、まだ呼び出されていない。 サーバが実行されており、サービスが呼び出し可能であることを確認します。
- 非クラスタ化（MOM のない）環境で、実行されているエージェントと Enterprise Manager との接続が失われた。 新しいまたは再起動された Enterprise Manager に情報を送信できるように、エージェント キャッシュをフラッシュして、以前に検出された依存関係をエージェントが再検出するようにします。
- エージェント キャッシュのフラッシュにより、サービスおよび依存関係のデータのすべてが削除された。 キャッシュ フラッシュ後にエージェントが情報を収集して Enterprise Manager に提供するまでの時間を考慮に入れてください。

- 依存マップに必要なマップ トレーサがエージェントのプロファイルで無効になっている。 *IntroscopeAgent.profile* ファイル内の com.wily.introscope.agent.transactiontrace.boundaryTracing.enable プロパティが false に設定されていないことを確認します。
- エージェントの名前を変更したため、結果として関連付けされているマップ ノードのデータが存在しない。エージェントが新しいエージェント マップ ノードについて表示するデータを収集する時間を見越します。
- 見つからないサービスが、エージェントによって監視されていないアプリケーション サーバ上で実行されている。
- 見つからない Web サービスが、サポートされていないプラットフォーム、またはサポートされていない SOAP エンジン上で実装されている。

## 論理等価物ルールについて

CA APM for SOA では、サービスおよびオペレーションに関する論理等価物を決定するための 3 つのヒューリスティックが提供されます。 これらのヒューリスティックは組織の環境に対して必要に応じて有効または無効にできます。デフォルトでは、CA APM for SOA は、以下のルールを使用して、クライアント側およびサーバ側の論理等価物を決定します。

- クライアント側サービスについて、サービスが複数のサーバ側サービスに依存している場合、すべてのサーバ側サービスは論理的に等価である。
- サーバ側のサービスについて、サービスが複数のクライアント側 Web サービスに依存している場合、すべてのクライアント側 Web サービスは論理的に等価である。
- 名前の一致について、2 つの物理サービス オペレーションのメトリック パスがエージェント指定子の削除後に同じになる場合、これらのオペレーションは論理的に等価であると見なす。

論理等価物のヒューリスティックを設定するために使用されるプロパティについては、「Enterprise Manager プロパティの構成 (P. 399)」を参照してください。

# SOA 依存マップでのコンテキストについて

SOA 依存マップで表示される情報は*以下の順で*2 つの要因によって決まります。

- 選択する SOA 依存マップの *コンテンツ タイプ*。
- 情報の表示対象となる *Investigator ツリー ノード*。

コンテンツ タイプおよび Investigator ツリー ノードによって SOA 依存マップの *コンテキスト*が決まります。 別のデータを選択すると、このコンテキストは変わります。

## コンテンツ タイプによる表示への影響について

SOA 依存マップに何が表示されるかを決定する最初の要因は、SOA 依存マップの *コンテンツ タイプ*です。 選択できるコンテンツ タイプは以下のとおりです。

- エージェント
- サービス
- オペレーション

［エージェント］の選択により、たとえば、エージェントにわたって高レベルの依存関係を表示できます。これに対して、［オペレーション］の選択では、オペレーション間の低レベルの依存関係を表示できます。 該当するコンテンツ タイプの選択により、SOA トポロジ全体でどこを表示しているかを見失わずに、依存関係の特定の部分にドリルダウンできます。

### 選択したノードを変更せずに、コンテンツ タイプを変更する

［サービス］を選択し、Investigator ツリーでエージェント ノードをクリックする場合、SOA 依存マップには、すべてのサービス、およびそのエージェントに関連付けられたサービス レベルの依存関係が表示されます。

同じ Investigator ツリー ノードで他のコンテンツ タイプ、たとえば［エージェント］を選択すると、SOA 依存マップには、選択したエージェント ノードへの依存関係のあるすべてのエージェントが表示されます。 たとえば、エージェント Tomcat01 上のサービスに、エージェント WebLogic02 上で実行されているサービスへの依存関係がある場合、SOA 依存マップには、WebLogic02 エージェント アイコンを指し示す依存関係の矢印と共に、Tomcat01 エージェント アイコンが表示され、エージェント レベルの依存関係が示されます。

### コンテンツ タイプの変更をマップ内のすべてのオブジェクトに適用する

コンテンツ タイプとして［エージェント］が表示されており、依存マップに 2 つのエージェント（たとえば Tomcat01 と WebLogic02）の間の依存関係が表示されている場合、［サービス］へのコンテンツ タイプの変更により、Tomcat01 と WebLogic02 の両方のエージェントへのサービス レベルの依存関係のすべてが表示されます。 コンテンツ タイプを変更するたびに、変更は SOA 依存マップに現在表示されているあらゆるものに適用されます。 この場合、［エージェント］から［サービス］へ変更することにより、マップに現在表示されているエージェント上のすべてのサービスが表示されるように、コンテキストが設定されます。

## Investigator ノードの表示への影響について

SOA 依存マップに何が表示されるかを決定する 2 番目の要因は、選択した Investigator ツリー ノードです。 コンテンツ タイプによって、表示される依存関係のタイプが決まりますが、Investigator ノードによって、マップに含まれる情報の範囲が決まります。 トップレベルのマップ ノードおよびその下の依存関係は *常に*、現在選択した Investigator ツリー ノードに基づきます。 別の Investigator ツリー ノードを選択すると、関連する SOA 依存マップの表示が大きく変わる場合があります。

ただし、現在選択している Investigator ツリー ノードに対して表示されたマップ ノードを使用して、新しい Investigator ツリー ノードに移動することもできます。 たとえば、SOA 依存マップを使用して、広範囲のサービスのネットワークを展開します。次に、マップ ノードをクリックして新しい Investigator ツリー ノードにジャンプし、そのノードを開始点としてマップを再表示できます。 これにより、ワンクリック操作で特定の Investigator ツリー ノードまたは SOA 依存マップ ノードに絞り込んで、関係のない不要なネットワーク ブランチを除外できます。

# SOA 依存マップを表示する

Investigator ツリー ノードを選択し、［SOA 依存マップ］タブをクリックすることにより、SOA 依存マップを表示できます。エージェントに対して CA APM for SOA が有効になっている場合、Investigator でエージェントまたは任意の［WebServices］ノードを選択して、SOA 依存マップを表示できます。CA APM for Oracle Service Bus や CA APM for TIBCO BusinessWorks などそのほかの SOA プラットフォームを監視する場合、拡張機能に関連付けられた Investigator ツリー ノードを選択しても、SOA 依存マップを表示できます。 たとえば、Oracle Service Bus を監視している場合、Investigator ツリーで［Proxy Services］ノードまたは特定のプロキシ サービス名を選択して、すべてのプロキシ サービスまたは特定のプロキシ サービスについて依存マップを表示できます。

**SOA 依存マップを表示するには、以下の手順に従います。**

1. Investigator ツリーで該当するノードを選択します。
2. ［ビューア］ペインで［SOA 依存マップ］タブをクリックします。

SOA 依存マップが、選択した Investigator ノードに基づいてデフォルト ビューで表示されます。 例：

- エージェント ノードの場合、SOA 依存マップには、エージェント レベルの依存関係の［物理モード］ビューが表示されます。
- ［WebServices］、［Client］、または［Server］ノードの場合、SOA 依存マップには、サービス レベルの依存関係の［物理モード］ビューが表示されます。

- <*web_service_namespace*> ノードの場合、SOA 依存マップには、サービス レベルの依存関係の［物理モード］ビューが表示されます。
- <operation_name> ノードの場合、SOA 依存マップには、オペレーション レベルの依存関係の［物理モード］ビューが表示されます。

たとえば、Investigator ツリーで［WebServices］-［Server］を選択し、［SOA 依存マップ］タブをクリックする場合、マップには、サーバ側サービスのすべてと共に、他のサービスへの 1 レベルのダウンストリームの依存関係が表示されます。

## Investigator ツリー ノードおよびマップ ノードについて

Investigator ツリーでノードを選択すると、そのノードが SOA 依存マップの開始点になります。選択したノードから開始される依存関係のみが SOA 依存マップに表示されます。

SOA 依存マップ上の各項目はマップ ノードと考えられます。 表示されるマップ ノードは選択したコンテンツ タイプによって異なります。 たとえば、エージェント、クライアント側またはサーバ側サービス、または個々のオペレーションを表すマップ ノードがあります。

## スタンドアロンのマップ ノードおよび依存関係のあるマップ ノードについて

エージェント、サービス、またはオペレーションに依存関係がない場合、それはスタンドアロンのマップ ノードとして表示されます。 設定したコンテキストに応じて、SOA 依存マップには、スタンドアロンのマップ ノードのみ、スタンドアロンのマップ ノードへの依存関係のあるマップ ノードの組み合わせ、または依存関係のあるマップ ノードのみが表示される場合があります。 たとえば、コンテンツ タイプとして［サービス］を選択し、Investigator ツリーで［Client］または［Server］ノードを選択する場合、SOA 依存マップには、選択したノードに関連付けられたスタンドアロンのサービスおよび依存関係のあるサービスが表示されることがあります。

Investigator ノードを変更せずに、コンテンツ タイプを［エージェント］に変更する場合、エージェント レベルの依存関係がないと、1 つのスタンドアロンのエージェント マップ ノードのみが表示されることがあります。

ただし、コンテンツ タイプを［オペレーション］に変更する場合、マップには、選択した Investigator ツリー ノードに関連付けられた依存度が高いオペレーションのすべてについて、1 レベルの依存関係が表示されます。依存度が高いオペレーションには、クライアント側オペレーションまたはサーバ側オペレーションが含まれる場合があります。

## オペレーションに対するマップ ノードについて

コンテンツ タイプを［オペレーション］に設定するか、サービスを右クリックして［すべてのオペレーションを表示］を選択する場合、SOA 依存マップには、次の規則を使用してオペレーション レベルの依存関係が表示されます。

- クライアント側サービスのネームスペースは緑色のボックスでマップ ノードとして表示される。
- サーバ側サービスのネームスペースは青色のボックスでマップ ノードとして表示される。
- エージェント名は灰色のボックスでマップ ノードとして表示される。

ボックスの色に関するのこれらの規則は、特定のオペレーションが属するサービスおよびエージェントの識別に役立つように使用されています。

# [物理モード]または[論理モード]ビューを選択する

SOA 依存マップでは、組織の環境内の SOA コンポーネントが 2 通りの方法で表示されます。

- *［物理モード］ビュー*は、エージェントで検出されるサービスおよびオペレーションの物理的な場所に基づいた表示形式です。
- *［論理モード］ビュー*は、物理的な場所に関係なく、類似の名前の付いたサービスおよびオペレーションのグループに基づいた表示形式です。

［物理モード］ビューと［論理モード］ビューとの違い、およびそれらの切り替えがどのように問題の解決に役立つかを示すために、以下の例を考えます。

サービスの問題を分析すると、根本原因としてあるサービスを特定しましたが、特定したサービスが分散サービスであることがわかりました。ロード バランサは 5 つの別個のアプリケーション サーバにサービスの要求を分散させています。

このシナリオでのその問題の原因を見つけるには、次のことを把握する必要があります。

- あるサービスが別のサービスをいつ呼び出すかを決定する論理インスタンス間の関係。
- どの物理インスタンスが実際に呼び出されるか（サービス A のどのインスタンスがサービス B のどのインスタンスを呼び出すか）を決定する物理インスタンス間の関係。
- パフォーマンスは物理インスタンス間でどの程度異なるか。
- サービスへのトラフィックはそのインスタンス間でどの程度分散されているか。

たとえば、論理サービスで 1 分間につき 100 回の呼び出しがあったが、5 つのインスタンスしか作成されなかった場合、この 100 回の呼び出しが個々のインスタンスにどのように対応しているのかを理解しておくことが重要です。 100 回の呼び出しの 90 回が単一のインスタンスに偏っていることがわかった場合、そのインスタンスに問題を切り分けて、問題を解決し、さらに、インスタンス上で増加した負荷の理由を分析できます。

SOA 依存マップでは、［表示］ドロップダウン リストを使用して、［物理モード］ビューと［論理モード］ビューを切り替えることができます。これにより、サービスの論理および物理インスタンスを区別して、特定の物理インスタンスの問題がどのように論理サービスのパフォーマンス全体に影響を与えている可能性があるかを判断できます。

**SOA 依存マップの[物理モード]または[論理モード]ビューを選択するには、以下の手順に従います。**

- ［SOA 依存マップ］ツール バーで［表示］ドロップダウン リストをクリックし、［物理モード］または［論理モード］のいずれかを選択します。

SOA 依存マップがリフレッシュされ、選択したビューに基づいて SOA 環境が表示されます。 たとえば、［物理モード］ビューでオペレーションが表示されていて、［論理モード］ビューに切り替えた場合、SOA 依存マップはリフレッシュされ、SOA 依存マップの論理表示に基づいてオペレーションが表示されます。

デフォルトでは、Investigator ツリーで仮想エージェントのノードを選択しない場合、SOA 依存マップには［物理モード］ビューが表示されます。［Virtual Agent］ノードおよびそのサブノードの場合、［論理モード］ビューがデフォルトで表示されます。

# コンテンツ タイプの選択

コンテンツ タイプは表示に影響を及ぼすため、オペレーション、サービス、またはエージェントに対する依存関係のマップを表示できます。 デフォルトのコンテンツ タイプは、Investigator ツリーで選択したノードによって異なります。 コンテンツ タイプを変更することにより、別の視点から依存関係にドリル ダウンできます。ビューを切り替えれば、問題サービスの診断に役立ちます。

**次の手順に従ってください**:

1. ［SOA 依存マップ］ツールバーで、コンテンツ タイプのドロップダウン リストをクリックします。
2. ［エージェント］、［サービス］、または［オペレーション］を選択します。

   **注**: ［論理モード］ビューを表示しているか、カスタム仮想エージェントを選択した場合は、［オペレーション］または［サービス］のみを選択できます。

   SOA 依存マップがリフレッシュされ、そのコンテンツ タイプの依存関係が表示されます。

**詳細**:

コンテンツ タイプによる表示への影響について (P. 91)

# 依存マップ ノードについてプライマリ メトリックを設定する

SOA 依存マップ上で CA Introscope® の標準メトリックおよび SOA 固有のメトリックの両方を表示できます。SOA 依存マップの各サービスおよびオペレーション ノードの下で、*プライマリ メトリック*には、最も注目度の高い CA Introscope® の標準メトリックが表示されます。 また、SOA 依存マップのエージェント、サービス、またはオペレーション ノードの上にマウス ポインタを置くと、ヒントには、表示するように選択したメトリックが表示されます。

SOA 依存マップでエージェントおよびサービスについて表示されるメトリックは、そのエージェントまたはサービスの下にあるすべてのオペレーションの集計値です。SOA 依存マップで現在選択または表示されているマップ ノードのみの集計値ではありません。 たとえば、サーバ側サービスのマップ ノードの上にマウス ポインタを置くと、ヒントには、表示するように選択したメトリックが表示されます。 ただし、マップ ノードに関するメトリックには、依存しているクライアント ノードおよび監視対象外のクライアントの呼び出しが含まれることがあります。または、SOA 依存マップに表示されていないクライアント（たとえば C++ クライアント）の呼び出しが含まれることがあります。 クライアントの起動はすべて、サーバ側のメトリック計算に影響を与えます。

**次の手順に従ってください**:

1. ［SOA 依存マップ］ツールバーで、［プライマリ メトリック］ドロップダウン リストをクリックします。
2. あらゆる依存マップ ノードに表示する CA Introscope® の標準メトリックを以下のいずれかから選択します。
   - Average Response Time
   - Responses Per Interval
   - Errors Per Interval
   - Concurrent Invocations
   - Stall count

   SOA 依存マップのデフォルトのプライマリ メトリックは［平均応答時間］です。

プライマリ メトリックを設定した後、依存マップ内のマップ ノードのすべてには、そのメトリックの現在の値が表示されます。

マップ ノードの上にマウス ポインタを置くと、ヒントには、追加の情報が表示されます。たとえば、収集されたデータポイントの数について最小、最大、現在の値が表示されます。

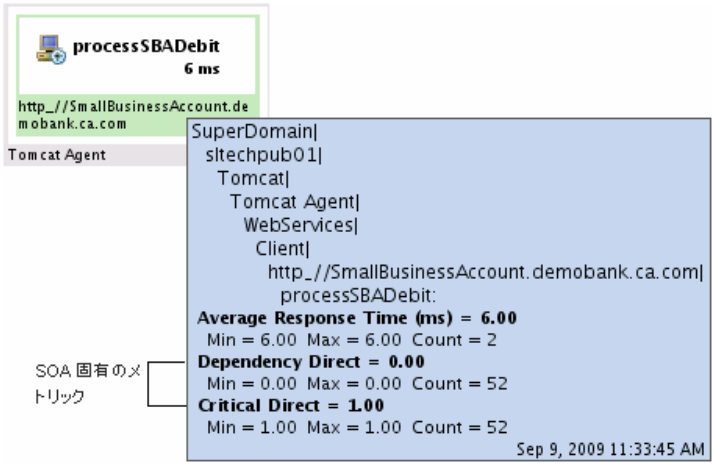
# 依存マップ ノードについてヒント メトリックを選択する

マップ ノードをポイントするたびに、SOA 依存マップにヒントが表示されます。 ヒントにはデフォルトではプライマリ メトリックのデータが表示されます。ヒントに SOA 固有のメトリックを表示することもできます。たとえば、ヒントに直接および間接依存関係メトリックまたは偏差メトリックが表示されるように選択できます。

**SOA 依存マップのヒント メトリックを設定するには、以下の手順に従います。**

1. ［SOA 依存マップ］ツール バーで［ヒント］ボタンをクリックして、［メトリックの選択］ダイアログ ボックスを表示します。
2. SOA 依存マップのヒントとして表示するメトリックを選択します。 たとえば、［Dependency Direct］および［Critical Direct］メトリックのような、SOA 固有のメトリックを選択できます。
3. ［OK］をクリックします。

選択した Investigator ノードに対応する SOA 依存マップ ノードにマウス ポインタを置くと、Enterprise Manager がその Investigator ツリー ノードにレポートするメトリックに応じて SOA 依存マップに、選択したメトリックのすべてまたは一部が表示されます。 たとえば、関連する［Dependency Direct ToolTips］メトリックが表示されるように SOA 依存マップを設定する場合、オペレーション ノードをポイントすると、ヒントにこのメトリックに関するデータが表示されます。



ただし、サービス ノードをポイントする場合、サービスが［Dependency Direct］メトリックを計算するためのデータを生成しないため、［Dependency Direct］メトリックは表示されません。

# 表示される依存関係のレベルを変更する

Investigator ツリーでノードを選択し、［SOA 依存マップ］タブをクリックすると、SOA 依存マップにはデフォルトで、選択したノードに関する依存関係と、追加の 1 レベルの依存関係が表示されます。依存関係が以下のいくつかの方法で表示されるように修正できます。

- マップで個々の項目について追加の依存関係を表示する。
- マップで個々の項目についてより低いレベルの依存関係を非表示にする。
- マップであらゆる項目についてすべてのレベルの依存関係を展開する。
- マップであらゆる項目について最低レベルの依存関係を非表示または*折りたたむ*。

## マップ ノードについて追加の依存関係を確認する

デフォルトでは SOA 依存マップに 1 レベルのみの依存関係が表示されます。 特定のマップ ノードに関心のある場合は、他のマップ ノードについてより低レベルの依存関係を展開せずに、そのエージェント、サービス、またはオペレーションについて追加のより低レベルの依存関係を確認できます。

マップで項目について追加の依存関係を確認するときは、一度に 1 レベルずつ依存関係を*展開*します。 追加の依存関係を確認するたびに、依存関係の次の最低レベルがマップに追加されます。 次のレベルを確認し、追加の依存関係が見つからない場合は、マップには「追加可能な依存関係はありません」というメッセージが表示されます。

**特定のマップ ノードについて追加の依存関係を確認するには、以下の手順に従います。**

- マップで個々の項目を右クリックし、メニューから［次の依存関係を表示］を選択します。

選択したマップ ノードについてのみ、次のレベルの依存関係が SOA 依存マップに展開されます。 たとえば、サーバ側のオペレーションが展開された場合、展開される次の依存関係のレベルは、関連するクライアント側のオペレーションです。

## マップ ノードについて依存関係を非表示にする

特定のエージェント、サービス、またはオペレーションについて依存関係の多くのレベルを展開している場合、その項目について依存関係の一部のレベルを非表示にすることもできます。 マップで項目について依存関係を非表示にするときは、一度に 1 レベルずつ依存関係を*折りたたみ*ます。依存関係を非表示にするたびに、依存関係の次の最低レベルがマップから削除されます。

**特定のマップ ノードの依存関係を削除するには、以下の手順に従います。**

- マップで個々の項目を右クリックし、メニューから［依存関係を非表示にする］を選択します。

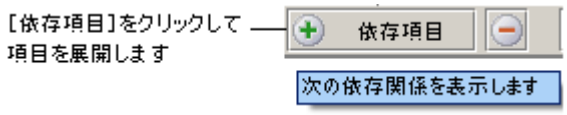
SOA 依存マップでは、選択したマップ ノードについて依存関係のすべてのレベルがロール アップされます。

## マップですべての項目について追加の依存関係を確認する

デフォルトでは SOA 依存マップに 1 レベルのみの依存関係が表示されます。 現在のマップですべての項目について追加のより低レベルの依存関係を確認するには、マップ全体について一度に 1 レベルずつ依存関係を*展開*できます。 追加の依存関係を確認するたびに、依存関係の次の最低レベルがマップに追加されます。 次のレベルを確認し、追加の依存関係が見つからない場合は、マップには「追加可能な依存関係はありません」というメッセージが表示されます。

**マップであらゆる項目について依存関係を展開するには、以下の手順に従います。**

- ［SOA 依存マップ］ツール バーで［次の依存関係を表示します］ボタンをクリックします。 例：



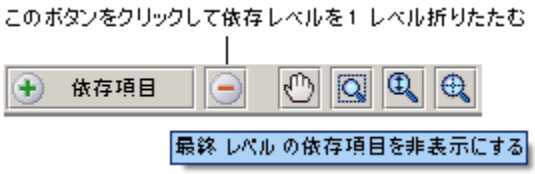
SOA 依存マップにあらゆる項目の最低レベルの依存関係が展開されます。最初のマップ ノードに 3 つの依存関係が表示されている場合、このボタンをクリックすると、3 つの各項目に依存関係の新しいレベルが追加されます。

## マップですべての項目について依存関係を非表示にする

マップで、すべてのエージェント、サービス、またはオペレーションについて多くのレベルの依存関係を展開している場合、マップ全体から一部の依存関係レベルを削除することもできます。 マップですべての項目について依存関係を非表示にするときは、一度に 1 レベルずつ依存関係を*折りたたみます*。 依存関係を非表示にするたびに、依存関係の次の最低レベルがマップから削除されます。

**マップであらゆる項目について依存関係を非表示にするには、以下の手順に従います。**

- ［SOA 依存マップ］ツール バーで［最終レベルの依存項目を非表示にする］ボタンをクリックします。 例：




SOA 依存マップでは、最低レベルの依存関係がロール アップされます。

# SOA 依存マップ内で移動する

SOA 依存マップを展開して複数のレベルの依存関係を表示する場合や、組織に特に複雑な SOA インフラストラクチャがある場合は、SOA 依存マップで移動することや、検索対象の情報を見つけることが難しくなっていることがあります。 SOA 依存マップには、マップ内での移動、マップと Investigator との間の移動、情報のすばやい表示/非表示に役立つ複数のツールが含まれています。

## SOA 依存マップをパン、ズーム、調整する

表示している依存マップが大きい場合、特定のセクションに移動したり、見やすくなるように部分のサイズを変更したりすることもできます。SOA 依存マップでは、タブ内で SOA 依存マップを移動、拡大、調整する次のツールが提供されます。

- パン
- 選択領域の拡大
- ズーム イン/ズーム アウト
- ウィンドウに合わせる

**タブ ビューで SOA 依存マップを移動するには、以下の手順に従います。**

1. ［SOA 依存マップ］ツール バーで［パン］ツール ボタンをクリックします。［パン］ツール ボタンをクリックすると、手のアイコンが表示されます。
2. SOA 依存マップ内でクリックします。
3. タブ内でマップを中央にドラッグして、目的の領域を確認します。

**SOA 依存マップの領域を拡大するには、以下の手順に従います。**

1. ［SOA 依存マップ］ツール バーで［選択領域の拡大］ツール ボタンをクリックします。［選択領域の拡大］ボタンをクリックすると、選択領域の内部に虫眼鏡が表示されます。
2. SOA 依存マップ内でクリックします。
3. ドラッグして、拡大する長方形の領域を選択します。

**SOA 依存マップをズーム イン/アウトするには、以下の手順に従います。**

1. ［SOA 依存マップ］ツール バーで［ズーム イン/ズーム アウト］ツール ボタンをクリックします。［Zoom in and out］ボタンをクリックすると、虫眼鏡と共に上向きおよび下向きの矢印が表示されます。
2. SOA 依存マップ内でマウス ボタンをクリックして押したままにします。
3. SOA 依存マップで、マウスのホイールを前方に回すと拡大し、後方に回すと縮小します。

**タブに収まるように SOA 依存マップのサイズを変更するには、以下の手順に従います。**

1. ［SOA 依存マップ］ツール バーで［タブに合わせる］ツール ボタンをクリックします。これにより、虫眼鏡とグリッド線が表示されます。

   SOA 依存マップはタブ内に収まるようにサイズが変更されます。

2. ［選択領域の拡大］または［ズーム イン/ズーム アウト］ツールを使用して、読みやすいようにマップをカスタマイズします。

## マップ ノードから関連する Investigator ツリー ノードにジャンプする

［物理モード］ビューで SOA 依存マップ ノードを表示している場合、任意のマップ ノードをクリックして、Investigator ツリー内の該当するコンポーネントに移動できます。 新しい Investigator ツリー ノードにジャンプすると、SOA 依存マップはその新しい Investigator ツリー ノードを開始点として再表示されます。 これにより、選択したマップ ノードの場所に基づいて現在の位置を再確認できます。

**マップ ノードから新しい Investigator ツリー ノードへジャンプし、マップを再表示するには、以下の手順に従います。**

1. SOA 依存マップが［物理モード］ビューで表示されていることを確認します。
2. SOA 依存マップでマップ ノードを選択します。
3. 右クリックし、［ツリー内のこの場所にジャンプ］を選択します。

   新しい Investigator ツリー ノードが選択され、SOA 依存マップの新しい開始点になります。

## サービスのすべてのオペレーションを表示する

［サービス］コンテンツ タイプを選択している場合、SOA 依存マップを展開して、特定のサービスのすべてのオペレーションを表示できます。

**サービスのすべてのオペレーションを表示するには、次の手順に従います。**

1. Investigator ツリーで、［Client］または［Server］ノードの下のサービスの個々のオペレーションを選択します。
2. ［SOA 依存マップ］タブをクリックします。

   SOA 依存マップで、対応するクライアントまたはサーバ オペレーションと、その第 1 レベルの依存関係が、対応するサービスおよびエージェントを表すボックス内に表示されます。
3. サービス マップ ノードを選択して右クリックし、［すべてのオペレーションを表示］を選択します。

   クライアント側またはサーバ側サービスのすべてのオペレーションが SOA 依存マップに表示されます。

## エージェントのすべてのサービスを表示する

［エージェント］コンテンツ タイプを表示するように選択している場合、SOA 依存マップを展開して、指定したエージェントのすべてのサービスを表示できます。

**エージェントのすべてのサービスを表示するには、次の手順に従います。**

1. Investigator ツリーで、［Client］または［Server］ノードの下の個々のサービスを選択します。
2. ［SOA 依存マップ］タブをクリックします。

   SOA 依存マップで、対応するクライアントまたはサーバ サービスと、その第 1 レベルの依存関係が、対応するエージェントを表す 1 つ以上のボックス内に表示されます。
3. SOA 依存マップで、関連するエージェントを選択して右クリックし、［全てのサービスを表示］を選択します。

   エージェントのすべてのサービスが SOA 依存マップに表示されます。

# SOA 依存マップ画像を保存する

さまざまな形式で SOA 依存マップのすべてまたは一部を保存して、以下のことが可能です。

- 問題の解決に役立つように同僚と情報を共有する。
- 組織の環境のスナップショットとして現在のマップを保存して後で確認できるようにする。

## SOA 依存マップの共有

サービスの問題の解決では、多くの場合、CA Introscope® または SOA 環境に慣れていない同僚と共同作業を行うことが必要になります。 たとえば、アラートに対応する場合、アプリケーションの問題を解決するために、別のサービスを担当する開発者と連携することが必要になることがあります。 画像として依存マップを保存することにより、サービスと現在のメトリックとの関係に関する価値のある情報をこれらの同僚と共有できます。

また、エージェント、サービス、またはオペレーション レベルの依存関係を表示する依存マップを、それらの依存関係の［物理モード］または［論理モード］ビューと共に保存することもできます。 サービス依存関係のグラフィカルなビューの提供により、コミュニケーションが容易になります。そのような情報は組織で問題をより効果的に解決するのに役立ちます。

## スナップショットとして SOA 依存マップを保存する

SOA 依存マップには常に、監視対象のサービス間の最も最近に検出された依存関係が反映されます。 データは定期的に確認されリフレッシュされますが、依存関係情報は履歴目的で保存されません。

ほとんどの場合、SOA 依存マップで依存関係は非常に静的です。これは、ほとんどの組織で運用環境内のアプリケーションの展開、修正、または削除は頻繁に行われないためです。 ただし場合によって、後日に確認できるように、依存関係データの「スナップショット」として実際の環境の画像を保存しておくことが、有用であるとわかります。

画像として SOA 依存マップのすべてまたは一部を保存するには、以下の手順に従います。

1. ［SOA 依存マップ］ツール バーで［画像として保存］ボタンをクリックすると、［画像のエクスポート］ダイアログ ボックスが表示されます。［画像として保存］ボタンをクリックすると、ディスクのアイコンが表示されます。
2. 出力ファイルの以下の設定を選択します。

**タイプ**

画像に使用するファイル形式を選択します。JPEG、GIF、PDF、PNG、または SVG ファイルとして依存マップを保存できます。

**ファイル名**

出力画像ファイルの出力先パスおよびファイル名を入力します。

**イメージ コンテンツ**

画像ファイルに SOA 依存マップの特定の部分を保存するオプションを選択します。

**可視ウィンドウのみ**

このオプションは、［現在のズーム レベル］サイズ オプションを選択する場合のみ使用できます。現在のウィンドウで表示可能なマップの一部のみを保存する場合は、このオプションを選択します。 たとえば、現在 SOA 依存マップに 100 のノードが表示されていて、拡大表示により、そのうち 10 のノードのみを表示している場合、このオプションを選択すると、その 10 のノードのみの画像が作成されます。

**グリッドを描画**

このオプションは有効ではありません。

**選択したオブジェクトのみ**

このオプションは、マップ ノードが選択されている場合のみ使用できます。マップで選択したオブジェクトのみを保存する場合は、このオプションを選択します。 たとえば、クライアントの Web サービス マップ ノードを選択した場合、このオプションを選択すると、クライアントの Web サービスの画像が作成されます。

**イメージ特性**

スライダ バーを動かすか、1 から 100 までの数字を選択して、画質とファイル サイズを指定します。100 は画像出力の最高画質を示します。 画像の品質が高くなるほど、保存される画像のファイル サイズが大きくなります。

**サイズ**

画像のサイズを制御する以下のオプションの 1 つを選択します。

**現在のズーム レベル**

マップの領域を拡大している場合は、このオプションを選択します。 拡大された領域のみを画像に表示する場合は、この［可視ウィンドウのみ］オプションを使用します。

**実際のサイズ**

その実際のサイズでマップ全体を保存するには、このオプションを選択します。 このオプションを選択すると、マップは縮小されません。

**画面に合わせる**

ウィンドウに収まるようにマップ全体を保存するには、このオプションを選択します。必要に応じてこのオプションを選択すると、ウィンドウに収まるようにマップが縮小されます。

**カスタム**

ピクセル単位で画像の幅および高さの設定を選択するには、このオプションを選択します。

3. ［OK］をクリックして、ファイルを作成します。

ファイルが指定した場所に保存されます。

# マップ ノードからトランザクション追跡を開始する

マップ ノードから直接トランザクション追跡セッションを開始することができます。 これにより、サービスの関係の高レベルのビューから、そのサービスを通過する実際のトランザクションの詳細ビューへすばやく移動できます。

たとえば、［SOA パフォーマンス - 概要］ダッシュボードで赤色のアラートが表示された後に特定のサービスの問題を解決しようとしているとします。 調査対象のサービスが重要なビジネス トランザクションによって使用されているため、そのサービスに関連付けられたトランザクションをできるだけすばやく表示することが重要です。

この場合、トランザクション追跡セッションを手動で開始し、フィルタ基準を入力する必要はありません。SOA 依存マップ上のサービスについて、マップ ノードから新しいトランザクション追跡セッションを直接開始することにより、時間を節約できます。

**マップ ノードからトランザクション追跡セッションを開始する方法**

1. 任意のマップ ノードを右クリックし、［トランザクション追跡の起動］を選択して、トランザクション追跡セッションを開始します。

   依存マップ ノードからトランザクション追跡を開始すると、自動的に選択されたマップ ノードに基づいて、デフォルト フィルタを使用して［新規トランザクション追跡セッション］ダイアログが開きます。

2. ［追加］をクリックしてフィルタの一覧にデフォルト フィルタを追加するか、必要に応じてフィルタ基準を修正して［追加］をクリックします。

3. ［フィルタを設定］をクリックし、［OK］をクリックして、トランザクション追跡ビューアを開きます。 トランザクション追跡ビューアでは、選択したマップ ノードにかかわるトランザクションに関連付けられた追跡のすべてが収集され、表示されます。

**注**: SOA 環境でのプロセスにまたがるトランザクション追跡の使い方、およびフィルタの操作の詳細については、「SOA 環境でトランザクション追跡を使用する (P. 111)」を参照してください。 新規トランザクション追跡セッションとトランザクション追跡ビューアの使い方の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# クラスタの SOA 依存マップ

クラスタ化 CA Introscope® 環境の場合、SOA 依存マップに、MOM の配下のコレクタの、サービスと検出された依存関係が表示されます。たとえば、3 つのコレクタ Enterprise Manager がある環境の場合は、エージェント間または個別のコレクタ上で実行されているサービス間の依存関係を表示できます。 依存関係は、アプリケーション サーバの場合と同様、コンテンツ タイプと Investigator ツリー ノードに基づいて表示されます。

SOA 依存マップ データはコレクタでのみ保存されます。MOM はコレクタからそのデータを受信します。MOM は SOA 依存マップ データ (P. 84)を保存しません。

SOA 依存マップの表示には、以下のファイルがクラスタ内の MOM Enterprise Manager および各コレクタ Enterprise Manager の両方にインストールされている必要があります。

`com.wily.introscope.soa.dependencymap.common_<バージョン>.jar`
`com.wily.introscope.soa.dependencymap_<バージョン>.jar`

これらのファイルは、クラスタ内のすべての Enterprise Manager 上の *<EM_Home>/product/enterprisemanager/plugins* ディレクトリにインストールされています。これらのファイルが見つからない場合、SOA 依存マップは適切に表示されません。

# 第 5 章: SOA 環境でトランザクション追跡を使用する

CA APM for SOA では、サービスに関連するコンポーネントのパフォーマンス、オペレーション、および全体の稼働状況に関するメトリックの収集により、サービスおよびビジネス プロセスを監視できます。 サービスに関する問題がある可能性があるとダッシュボードまたは固有のメトリックで提示されると、トランザクション追跡を使用して、実際のビジネス トランザクションに関する詳細を表示できます。これは、原因を見つけて問題を解決するのに役立ちます。

このセクションには、以下のトピックが含まれています。



## プロセスにまたがるトランザクション追跡について

SOA 環境では一般的に、複数の Java 仮想マシン（JVM）または共通言語ランタイム（CLR）インスタンスにわたってトランザクションが実行されます。あるサービスから別のサービスへ処理が渡されます。トランザクションの最初から最後までを参照するには、JVM と CLR の境界にまたがって同期および非同期の呼び出しを追跡できる必要があります。また、Java または .NET エージェントのいずれかを実行する複数のプラットフォームにわたって、トランザクションを追跡できることも必要になる場合があります。

CA APM for SOA を有効にすると、サポートされるプラットフォーム上で実行される、HTTP、HTTPS、または JMS トランスポート プロトコルを使用するビジネス トランザクションを、最初から最後まで追跡できます。 トランザクション全体には、多種多様なプラットフォームで処理されるセグメントを含めることができます。 たとえば、WebLogic 上で実行されているサービスを使用して、SAP NetWeaver または .NET サーバ上で実行されているサービスに渡されるトランザクションを追跡できます。 追加の SOA 拡張機能を有効にすることで、以下のコンポーネントの任意のセグメントをトランザクションに含めることができます。

- Oracle Service Bus
- WebSphere Process Server
- WebSphere Enterprise Service Bus
- TIBCO BusinessWorks
- webMethods Integration Server
- WebSphere MQ
- Spring Web サービス

エージェントで Spring Web サービスをサポートすると、クライアントとサーバ両方の Web サービス内部メトリックおよび Web サービス転送メトリックが提供されます。

追跡するすべてのノードに対して CA APM for SOA が有効であれば、ビジネス トランザクションには、あらゆる組み合わせのプラットフォームを含めることができます。 エージェントに対して CA APM for SOA を有効にすることにより、別のノード上で実行されている複数の JVM または CLR インスタンス上のサービスを呼び出すトランザクションの詳細にドリルダウンできます。

CA APM for SOA を有効にした後に、追加の構成は必要ありません。

**注**: サポートの詳細については、「Compatibility Guide」の「SOA Performance Management」セクションを参照してください。

## トランザクションのセグメントがリンクされる仕組みについて

通常、トランザクションは、あるプロセスから別のプロセスへ渡される一連の呼び出しおよび応答で構成されます。 多くの場合、トランザクションのさまざまなセグメントは別々の論理または物理サーバ上で実行されるか、別々のコンポーネントまたはバックエンド システムに分散される場合があります。

したがって、完全なトランザクションをアセンブルするには、どの処理セグメントが同じトランザクションの一部か、いつトランザクション内の 1 つのプロセスが別のプロセスを呼び出すかを特定することが必要です。 異なる JVM または CLR インスタンス上で実行できるプロセスを呼び出すトランザクションを最初から最後まで追跡するには、エージェントはトランザクションに*相関識別子*を追加します。 相関識別子は、同じトランザクションの一部であるセグメントを特定するために、あるプロセスから別のプロセスへ渡すことができます。

同じトランザクションの一部であるコンポーネントを特定することに加えて、相関識別子は、トランザクションのさまざまな部分が呼び出される順序を追跡するための、*シーケンス*の情報を提供します。 シーケンスの情報により、トランザクション セグメントが呼び出される順序を参照できます。 同期トランザクションの場合、順序は、トランザクション セグメント間の呼び出し元と呼び出し先の関係の特定に役立ちます。 非同期トランザクションの場合、順序は、複雑なクライアントおよびサーバ トランザクション セグメントの複数のプロセスにまたがる処理ワークフローの特定に役立ちます。

相関識別子データ セットは、エージェントによって自動的に管理され、Enterprise Manager に提供されます。この情報は、選択したトランザクションをトランザクション追跡ビューアでグラフィカル形式で表示するために使用されます。

## プロセスにまたがるトランザクション追跡のコンテキストについて

サービスのネームスペースまたはオペレーション名に基づいてトランザクション追跡セッションを開始できます。ほとんどの場合、ネームスペースは、問題のあるサービスの特定に役立つ適切な情報を提供します。 問題の原因である可能性があるオペレーションを特定した場合、そのオペレーションについてトランザクション追跡を開始することもできます。

ただし、オペレーション名を使用する場合、サービスのネームスペースがオペレーションのコンテキストを提供することに注意してください。 たとえば、航空会社で座席を予約したり、レストランでテーブルを予約したり、病院で診察を予約したりするアプリケーションで、オペレーション名 *makeReservation* を使用するとします。 オペレーション名 *makeReservation* についてトランザクション追跡を開始する場合、そのオペレーションが使用されるネームスペースに注目する必要があります。

特定の追跡を選択すると、オペレーションに関するネームスペースの情報が［追跡ビュー］タブの［コンポーネントの詳細］に表示されます。 たとえば、［追跡ビュー］タブで *makeReservation* オペレーションを選択する場合、［コンポーネントの詳細］をチェックして、そのオペレーションに関連付けられたネームスペースを確認することにより、このオペレーションが旅行代理店 Web サービスまたはホテル予約サービスの一部かどうかを調べることできます。

# トランザクション追跡を使用して問題を解決する

プロセスにまたがるトランザクション追跡がどのようにすばやく効果的に問題の評価に役立つかを確認するには、以下の例を考えます。 トランザクション追跡セッションを実行した後に、アプリケーション サポートは 6 秒（6000 ミリ秒）の実行期間のトランザクションを見つけます。

トランザクションの追跡ビューから明らかになるのは、トランザクションにクライアント側 Web サービス（*dataservice.yourcompany.net/invoke*）からサーバ側 Web サービス（*cics.mycompany.net/invoke*）への呼び出しが含まれること、サーバ側 Web サービスが CICS メインフレームへの多くの呼び出しを行っていることです。

CICS の処理時間は追跡ビューに明示的に表示されません。トランザクションの該当する部分がインスツルメントされないためです。しかし追跡ビューには、高速に連続で繰り返される要求に CICS のバックエンドが応答していることが示されます。 このトランザクション追跡から、アプリケーション サポート スペシャリストが確認できるのは、サーバ側サービスで、ネストされたループのようなロジックをプログラムすることにより、呼び出しが繰り返されている可能性が最も高いことです。また、サービスの*呼び出し*オペレーションが、トランザクションの実行時間の合計のほとんどの時間を占めていることです。この情報を使用して、アプリケーション サポート スペシャリストは、サーバ側 Web サービスの所有者または開発者に直接問い合わせて、CICS のバックエンドを呼び出すアプリケーション ロジックの詳細な調査を依頼できます。

# トランザクション追跡の開始と表示

ローカルまたはリモート コンピュータ上の別の JVM または CLR インスタンスへの呼び出しがビジネス トランザクションに含まれる場合、プロセスにまたがるトランザクション追跡を使用して Web サービスの問題を特定して解決できます。 Enterprise Manager がクラスタの一部でない場合、トランザクションに参加する JVM または CLR 上のエージェントは同じ Enterprise Manager に情報をレポートする必要があります。そうしないと、相関する追跡が表示されません。 エージェントが同じ Enterprise Manager に情報をレポートする場合、トランザクション追跡セッションを開始して、ビジネス プロセス全体に関する情報を収集できます。クラスタ環境では、Workstation が MOM に接続する限り、プロセスにまたがるトランザクションに参加するエージェントはコレクタのいずれかに接続でき、追跡が関連付けられます。

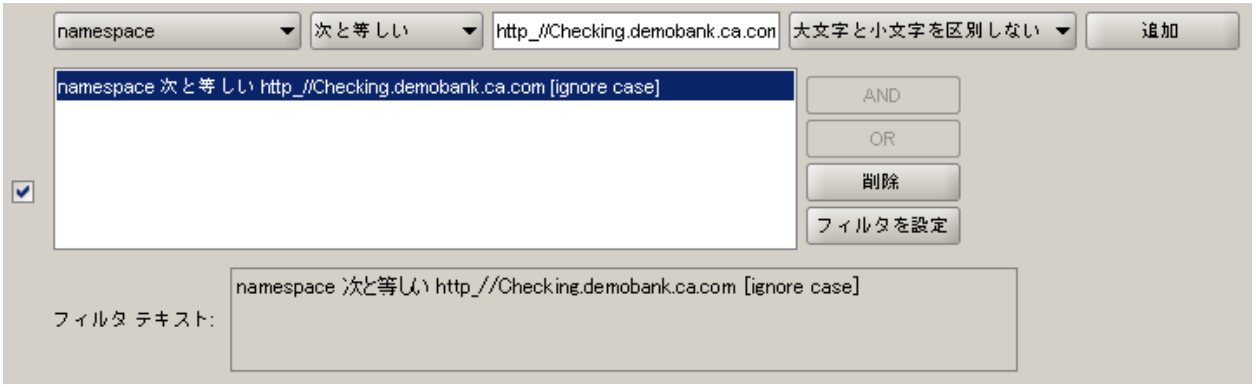
トランザクション追跡セッションは、以下のいずれかの方法で開始できます。

- SOA 依存マップ内のマップ ノードから直接開始する方法。
- ［Workstation］-［新規トランザクション追跡セッション］をクリックして Workstation から手動で開始する方法。

次の手順に従ってください:

1. SOA 依存マップでマップ ノードを右クリックするか、［Workstation］-［新規トランザクション追跡セッション］をクリックして、［新規トランザクション追跡セッション］ダイアログ ボックスを表示します。
2. SOA 関連のフィルタ オプションをアクティブにするために 4 番目のチェック ボックスをオンにします。

   以下の例では、SOA フィルタおよびネームスペース フィルタ定義をアクティブにするためにオンにするチェック ボックスを示しています。




3. SOA 関連のフィルタ オプションの一覧から使用するトランザクション追跡フィルタのタイプを選択し、［追加］をクリックします。

   CA APM for SOA には、サービスのネームスペースまたは特定のオペレーション名に基づいてトランザクションを選択できる、SOA 固有のトランザクション追跡フィルタが含まれています。新しいトランザクション追跡を開始する前に選択したノードに応じて、デフォルトのフィルタおよび値が自動的に設定されます。

   指定した基準に一致するすべての Web サービスまたは Web サービス オペレーションがかかわるトランザクションを追跡するためのネームスペースまたはオペレーション名フィルタを選択できます。この基準には、TIBCO BusinessWorks や webMethods Integration Server のような、SOA プラットフォームによって提供される Web サービスが含まれる可能性があります。
4. ［フィルタを設定］をクリックします。
5. ［追跡セッション期間（分）］フィールドで、トランザクション追跡セッションの実行にかかる分数を入力します。

6. ［追跡エージェント］セクションで、トランザクションを追跡するすべてまたは特定のエージェントを選択します。
7. ［OK］をクリックします。

   トランザクション追跡セッションが開始されます。

**注**: Workstation からトランザクション追跡を開始する方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

**詳細**:

マップ ノードからトランザクション追跡を開始する (P. 108)

## サマリ ビューの使い方

トランザクション追跡セッションを開始した後、トランザクション追跡の結果はトランザクション追跡ビューアの上部に表示されます。 この一覧には、追跡中のネームスペースまたはオペレーションについてトランザクションのすべてが表示されます。 ［継続時間］列見出しをクリックすると、継続時間に基づいてトランザクションを並べ替えることができます。

ビューアの上部でトランザクションを選択すると、［サマリ ビュー］タブに、他の JVM または CLR への呼び出しなど、トランザクションに含まれる呼び出しのすべてが表示されます。

トランザクションを選択した後、サマリ ビューには、呼び出しの数、呼び出しの長さ（ミリ秒単位）、最小、平均、最大の呼び出し数など、選択したトランザクションで呼び出されたサービスおよびオペレーションが一覧表示されます。

**注**: トランザクション追跡にサマリ ビューを使用する方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

## 追跡ビューの使用

グラフィカル形式でトランザクションに関する詳細を表示するには、［追跡ビュー］タブをクリックします。

トランザクション追跡ビューには、バーとしてトランザクション内の各コンポーネントが表示されます。また、各コンポーネントがトランザクション実行時間の合計のどのくらいの時間を占めているかが示されます。 同期トランザクションの場合、［追跡ビュー］タブには、表示しているトランザクションのセグメントにかかわるノードごとに、ノード上のコンポーネント間の呼び出し関係も示されます。

トランザクションに複数の JVM 処理が含まれる場合は、各 JVM 追跡は個別の行に表示されます。 SOA トランザクションでは、複数の JVM 上で処理が発生するのが一般的です。

グラフでトランザクションのコンポーネントを選択すると、コンポーネントに関する追加の詳細がグラフの下の［コンポーネントの詳細］に表示されます。

**注**: 追跡ビューの使い方の詳細、トランザクション追跡の［コンポーネントの詳細］に表示される情報の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

## ツリー ビューの使用

選択したトランザクションのコンポーネントの階層ビューを表示するには、［ツリー ビュー］タブをクリックします。

［ツリー ビュー］には、階層構造でトランザクションのコンポーネントが表示されます。さらに、コンポーネントがトランザクションの実行時間の合計のどれくらいの時間を占めているかに基づいて、赤色、黄色、または緑色のインジケータが表示されます。

コンポーネントを選択して展開すると、さらに詳細な情報を参照できます。コンポーネントを選択する場合、このタブの［コンポーネントの詳細］ペインに追加の情報が表示されます。

**注**: ツリー ビューの使い方の詳細、トランザクション追跡の［コンポーネントの詳細］に表示される情報の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

## シーケンス ビューの使い方

SOA 環境の分散アプリケーションは複雑であり、一般的に、1 つのユーザ トランザクションは個別のエージェント JVM または CLR 上で実行されている複数のスレッドにまたがり、同期および非同期の呼び出しを含んでいます。 これらの個々のトランザクション セグメントから、論理ユニットとしてトランザクションの最初から最後までを把握する必要があります。［シーケンス ビュー］タブには、トランザクションのセグメント間の呼び出し元と呼び出し先の関係が表示され、シーケンスの順序が視覚的に明らかになります。

シーケンス ビューは、トランザクションが以下のような場合に特に役立ちます。

- 非同期呼び出しを含んでいる。
- 相互に時刻同期されていないエージェント上で実行されているプロセスへの呼び出しを含んでいる。
- 複数の JVM または CLR にわたる複雑な同期呼び出しを含んでいる。
- 別のエージェント上のトランザクション追跡セッションから呼び出されたダウンストリーム エージェントに対して自動的に開始された追跡を含んでいる。

選択したトランザクション内のプロセスの呼び出し順序を表示するには、［シーケンス ビュー］タブをクリックします。 このタブでは、矢印は呼び出されたプロセスを示します。棒グラフでは、継続時間に基づいて並べ替えられたプロセスが一覧表示されます。 トランザクションの一部として特定されたが、正しい場所を決定できない追跡がある場合、それらは点線を使用してシーケンスに追加されます。 グラフまたは棒グラフでプロセスを選択する場合、そのプロセスの追加の詳細が右のパネルに表示されます。

最も遅いプロセスが、棒グラフおよび遅いプロセスのアイコンで、それらのランキングに基づいて示されます。 プロセスの実行中に何らかのエラーがある場合は、エラーのアイコンがそのプロセスに対して表示されます。

## プロセス ノードにラベルを付ける方法を選択する

シーケンス ビューで一覧表示された各プロセスは、オペレーションの完了かかわるプロセスのコンポーネントの追跡を表します。 追跡が同じサーバまたはエージェントから取得された場合、プロセスに表示されるラベルは識別しにくかったり、識別できなかったりすることがあります。

デフォルトでは、オペレーション名が含まれるコンポーネント名、EJB 文字列が含まれるコンポーネント名、または先頭エージェント名 + 一意コンポーネント リソース名を使用して、プロセス ラベルは自動的に選択されます。 デフォルトの選択を使用しない場合は、先頭コンポーネント名、末尾コンポーネント名、スレッド名、または説明を使用するように、プロセス ラベルを変更できます。

プロセス ラベルとして末尾コンポーネント名を選択する場合、使用される名前は呼び出しスタック内の末尾の明確な名前です。 呼び出しのシーケンスに応じて、および同期または非同期呼び出しがあるかどうかに応じて、名前がプロセスの末尾コンポーネントになるかどうかが決まります。

**異なるプロセス ラベル付けスキームを選択するには、以下の手順に従います。**

- ［シーケンス ビュー］タブでプロセス ラベルのオプションを選択します。

## 継続時間の計算方法を選択する

［シーケンス ビュー］タブで、追跡について継続時間を計算するには、2 通りの方法があります。

- **フル継続時間**は、プロセスの開始と終了の時間を使用して計算されます。 このアプローチは、継続時間が［追跡ビュー］タブにどのように表示されるかに似ています。 この方法では、ルート追跡が同期トランザクションの中で最も遅くなります。 これはデフォルトの計算です。
- **正味継続時間**は、フル継続時間から追跡の非同期子の継続時間を減算することにより計算されます。 このアプローチの主な利点は、同期トランザクションの最も遅いプロセスが容易に明らかになることです。 フル継続時間から正味継続時間への切り替えにより、ランク付けされた継続時間に基づいて一覧表示されたプロセスの順序が変更されます。 継続時間は実行時間の計算方法によって異なります。

**継続時間の計算方法を変更するには、以下の手順に従います。**

- ［シーケンス ビュー］タブの［継続時間のタイプ］で、［フル］または［正味］を選択します。

## シーケンス ビューと追跡ビューとの間で移動する

シーケンス ビューと追跡ビューとの間を行き来する操作は、多くの場面で有用です。 右クリック メニュー オプションにより、特定のプロセスまたはコンポーネントのコンテキストを失わずに、これらのビュー間ですばやく移動できます。

**追跡ビュー内のプロセスからシーケンス ビューにジャンプするには、以下の手順に従います。**

1. グラフでプロセスを選択します。
2. 右クリックし、［追跡ビューでスレッドを選択］をクリックします。

**追跡ビュー内のコンポーネントからシーケンス ビューにジャンプするには、以下の手順に従います。**

1. グラフでコンポーネントを選択します。
2. 右クリックし、シーケンス ビューで［Select Thread］をクリックします。

## トランザクション内のプロセスに関する詳細を表示する

グラフまたは棒グラフでプロセスを選択する場合、右のパネルでそのプロセスに関する詳細を表示できます。 選択したプロセスに応じて、以下の情報の一部またはすべてを使用できます。

**スレッド名**

呼び出しスタック内の選択したスレッドの名前。

**スレッド グループ名**

スレッド グループの名前。

**アプリケーション名**

アプリケーションに使用される表示名。

**クラス**

インスツルメントされたクラス。

**メソッド**

インスツルメントされたメソッド。

**メソッド記述子**

インスツルメントされたメソッド引数またはシグネチャ。

**リソース名**

アプリケーション リソースのタイプ。 たとえば、リソース タイプはサーブレットまたは EJB の識別に使用できます。

URL

アプリケーション フロントエンドの識別に使用されるフル Uniform Resource Locator（URL）。

**正規化された URL**

アプリケーション フロントエンドを識別する方法としてエージェント プロファイルで定義された Uniform Resource Locator（URL）の正規化された部分。

**参照 URL**

参照ページの Uniform Resource Locator（URL）。

**サーバ名**

プロセスが実行されているサーバの名前。

**サーバ ポート**

プロセスが実行されているポートの番号。

**スキーム**

使用される Uniform Resource Identifier（URI）スキームのタイプ。

URI スキームは、Uniform Resource Identifier の名前付け構造の最上位であり、スキーム名とその後のコロン文字（:）で構成されます。

たとえば、ハイパーテキスト転送プロトコルを使用してアクセスされる Web サービスの URI スキームとして http を指定できます。

HTTP メソッド

使用される HTTP メソッドのタイプ。 たとえば、最も一般的な HTTP メソッドは POST と GET です。

**プロセスにまたがるデータ**

同じトランザクションの一部であるプロセスを識別するために使用される一意の相関識別子。

**追跡タイプ**

プロセスに対するトランザクション追跡のタイプ。 追跡タイプは、プロセスがトランザクション追跡セッションの一部であるか、サンプリングされたトランザクション追跡で収集されたかどうかを示します。

# トランザクション追跡用のフィルタの設定

CA APM for SOA を有効にすると、Workstation に **namespace** および **operationname** フィルタが追加されます。これらは、関心のあるトランザクションをすばやく分離するのに役立ちます。 たとえば、特定のサービスが重要なビジネス トランザクションで使用される場合、namespace フィルタにより、該当するネームスペースを含むトランザクションについて、すべての受信トランザクションをフィルタできます。 問題が発生しているネームスペースを特定するときは、operationname フィルタを使用して、問題の切り分けに役立つ 1 つ以上の特定のオペレーションについて、トランザクションをフィルタできます。

また、namespace および operationname フィルタを他のフィルタと組み合わせることにより、返されるトランザクション結果をさらに詳細に制御できます。 たとえば、［ユーザ ID］または［エラー］フィルタと namespace フィルタとを組み合わせて、両方の基準を満たすトランザクションのみを収集できます。

**次の手順に従ってください**:

1. ［Workstation］ > ［新規トランザクション追跡セッション］をクリックします。

   使用可能なフィルタはインストールしているコンポーネントによって異なります。

2. 特定のフィルタをアクティブにするためのチェック ボックスをオンにします。

   たとえば、［最小トランザクション継続時間］チェック ボックスをオンにして、指定した時間より長く継続するトランザクションを検索します。 また、特定のパラメータ フィルタ（［ユーザ ID］や［セッション ID］など）を使用して、指定した条件に一致するトランザクションに絞り込みます。

3. CA APM for SOA およびそのほかの SOA プラットフォーム用のフィルタをアクティブにするためのチェック ボックスをオンにします。

4. フィルタ タイプとフィルタ条件を選択し、値を入力して、大文字/小文字の区別を選択します。 ［追加］をクリックして、使用可能なフィルタの一覧にこのフィルタを追加します。

   namespace と operationname のオプションは Web サービス用の標準のフィルタです。 これらのフィルタは、Oracle Service Bus、TIBCO BusinessWorks、または webMethods Integration Server のような SOA プラットフォームを通じて公開される Web サービスに適用できます。

   ただし、その他の SOA プラットフォームを監視している場合、トランザクションを収集するために、以下の追加のフィルタ タイプを使用できます。

   **アダプタ ノード**

   WebSphere Process Server アウトバウンド アダプタ

   **ビジネス プロセス**

   TIBCO BusinessWorks プロセス インスタンス

   webMethods Integration Server ビジネス プロセス

   WebSphere Process Server ビジネス プロセス

**ビジネス サービス**

Oracle Service Bus ビジネス プロセス

**ビジネス ステート マシン**

WebSphere Process Server ビジネス ステート マシン

**メディエーション フロー**

WebSphere Enterprise Service Bus メディエーション フロー

**メディエーション フロー オペレーション**

WebSphere Enterprise Service Bus メディエーション フロー オペレーション

**メッセージ サーバ**

IBM WebSphere MQ メッセージ サーバ

**プロキシ サービス**

Oracle Service Bus プロキシ サービス

手動でネームスペースまたはオペレーション名を入力する場合は、フィルタに名前の正しいスペルを使用して、それらの条件に一致するトランザクションが見つかるようにします。

5. フィルタ リストにフィルタを追加します。必要に応じて、フィルタを組み合わせてコンプレックス フィルタを作成 (P. 127)します。 たとえば、ネームスペース用のフィルタを追加し、次にオペレーション名用の別のフィルタを追加できます。 定義したフィルタ リストで両方のフィルタを選択し、［AND］をクリックして、指定したネームスペースおよびオペレーション名の両方に基づいてトランザクションをフィルタできます。

6. このセッションに使用するフィルタを選択し、［フィルタを設定］をクリックします。

   以前に設定したいずれかのフィルタが置き換えられます。 一度に有効になるのは、1 つのフィルタのみです。

7. ［追跡セッション期間（分）］フィールドで、トランザクション追跡セッションの実行にかかる分数を入力します。

8. ［追跡エージェント］セクションで、トランザクションを追跡するすべてまたは特定のエージェントを選択します。

9. ［OK］をクリックします。

   トランザクション追跡セッションが開始されます。

## フィルタを追加および保存する

フィルタを定義して［追加］をクリックすることにより、必要な数のさまざまなフィルタを追加できます。 ［追加］をクリックすると、使用可能なフィルタの一覧に新しいフィルタが追加されます。 フィルタの重複は許可されません。

任意のトランザクション追跡セッションで使用するフィルタを指定するには、使用可能なフィルタの一覧でフィルタ定義を選択し、［フィルタを設定］をクリックする必要があります。

［OK］をクリックしてトランザクション追跡セッションを開始すると、定義したフィルタが保存されます。したがって次回、新しいトランザクション追跡セッションを開始すると、それらのフィルタは使用可能です。［OK］をクリックすると、フィルタはユーザ別に自動的に保存されます。次回、Workstation にログオンすると、以前に定義した独自のフィルタのセットから追跡用のフィルタを選択できます。

## フィルタを削除する

使用可能なフィルタの一覧でフィルタを選択し、［削除］をクリックすることにより、そのフィルタを削除できます。［削除］をクリックすると、［新規トランザクション追跡セッション］ダイアログ ボックスで、使用可能なフィルタの一覧から、選択したフィルタが削除されます。 ［OK］をクリックして、選択したフィルタを永続的に削除する必要があります。［OK］をクリックしない場合、次回、新規トランザクション追跡セッションを開始すると、削除したフィルタが使用可能なフィルタの一覧に復元されます。

フィルタが別のフィルタのコンポーネントとして使用されている場合は、フィルタを削除することはできません。 たとえば、*namespace filter1* と *namespace filter2* を組み合わせて *namespace filter1 AND namespace filter2* フィルタを作成する場合、この AND フィルタを削除するまで、*namespace filter1* は削除できません。

## コンプレックス フィルタの使い方

AND または OR のいずれかの演算子を使用すると、フィルタのリストの 1 つ以上のフィルタを組み合わせたコンプレックス フィルタを作成できます。 組み合せるフィルタは、定義済みのフィルタのリストにあらかじめ含められている必要があります。AND または OR の演算子を使用して、2 つの簡単なフィルタのステートメントを組み合せると、新しいフィルタを作成できます。そのフィルタを 1 つのフィルタとして使用したり、追加のフィルタを作成するための構成単位として使用することもできます。

AND 演算子を使用することは、追跡の際にフィルタのすべてのコンポーネントを通過する必要があることを意味しています。 たとえば、*namespace contains demobank AND operationname starts with process* フィルタでは、受信追跡の際に両方のフィルタ条件を満たす必要があります。

一方、OR の演算子を使用する場合、その追跡は少なくとも 1 つのフィルタ コンポーネントを満たす必要があることを意味しています。たとえば、*namespace contains demobank OR namespace contains creditcheck* フィルタでは、受信追跡の際に最初のフィルタ条件、2 番目のフィルタ条件、または両方のフィルタ条件を満たす必要があります。

# SOA サービスについてイベント データベースにクエリする

トランザクション追跡ビューアは主に、トランザクション追跡セッションからライブまたは最新のデータを表示するために使用します。 トランザクション追跡セッションからの結果は、自動的にトランザクション イベント データベースに保存され、履歴クエリ ビューアを使用して表示できます。後で情報へのアクセスを必要とする場合、SOA 関連のサービスについて任意のビジネス トランザクションまたはエラーに関する情報を表示するために、トランザクション イベント データベースにクエリできます。

**Web サービスに関連するすべてのトランザクションについてデータベースにクエリするには、以下の手順に従います。**

1. ［Workstation］を開いた状態で、［Workstation］ - ［履歴イベントをクエリ］を選択します。
2. ［クエリ］フィールドで、該当する情報を取得するためのクエリ情報を入力します。 例：
   - Web サービスを含む任意のイベントのクエリを行う場合、*「webservices」と入力します。*
   - 特定の Web サービス エンドポイントに関連する任意のイベントについてクエリするには、必要な URL を入力します。
3. 必要に応じて、クエリについて時間範囲を選択します。
4. ［実行］をクリックして、Web サービスの情報が含まれるあらゆるビジネス トランザクションまたはエラーを表示します。
5. 一覧で任意の行をクリックして、下部のペインでその詳細を確認します。

   たとえば、選択した行がトランザクション追跡（T）である場合、［サマリ ビュー］、［追跡ビュー］、［ツリー ビュー］、および［シーケンス ビュー］タブをクリックして、追加の情報を参照できます。これは、トランザクション追跡ビューアでの操作と同様です。

## エラー イベントの情報を表示する

履歴イベントがエラー（E）である場合、下部のペインには［スタック ビュー］タブが表示されます。イベントがトランザクションである場合、下部のペインには［サマリ ビュー］、［追跡ビュー］、［ツリー ビュー］、および［シーケンス ビュー］タブが表示されます。 表示された情報を操作して、Web サービスのどのコンポーネントが問題の原因になっているかを判断できます。

**Web サービスに関連するエラーについてデータベースをクエリするには、以下の手順に従います。**

1. ［Workstation］ - ［履歴イベントをクエリ］を選択します。
2. ［クエリ］フィールドで、以下のように入力します。

   ```
   type:errorsnapshot AND webservices
   ```

3. 一覧で行をクリックして、［スタック ビュー］タブでエラーの詳細を確認します。
   - サーバ側のエラーは、赤のエラー メッセージで表示され、ここでエラーの根本原因を確認できます。
   - クライアント側とサーバ側のエラーについては、SOAP 障害の例外に沿った根本原因を確認できます。
   - SOAP 障害のエラー メッセージには、赤で表示されるエラー メッセージに［*SOAP 障害*］の接頭辞が付いています。

## 相関イベント情報を表示する

トランザクションに関する詳細を確認する場合、同じトランザクションの一部であるイベントのうち、トランザクション中に呼び出された他のプロセスまたは他のノード上で実行されているイベントがあるかどうかをチェックできます。

**トランザクション追跡について相関イベントを表示するには、以下の手順に従います。**

1. トランザクション追跡ビューアまたは履歴クエリ ビューアに表示された一覧からトランザクション追跡を選択し、［追跡ビュー］タブをクリックします。
2. ［追跡］ -［相関イベント］を選択して、選択したトランザクション追跡の一部である他のプロセスまたはノード上の追跡の一覧を表示します。

   その後、任意の相関追跡を選択して、その追跡についてサマリ ビュー、追跡ビュー、ツリー ビュー、またはシーケンス ビューを表示できます。

# 第 6 章: SOA 固有のプロパティの構成

このセクションでは、構成プロパティを使用して、CA APM for SOA をカスタマイズする方法について説明します。

このセクションには、以下のトピックが含まれています。



## Web サービスについて表示される名前を設定する

デフォルトでは、Web サービスのノードは Web サービスの*ネームスペース*を使用して名前を付けられます。 Web サービスのネームスペースはその URL に似ています。 たとえば、Web サービスが以下の URL を使用するとします。

```
http://ClearingHouse.demobank.ca.com
```

この場合、そのノードはデフォルトでは以下のように表示されます。

```
http_//ClearingHouse.demobank.ca.com
```

オプションで、ノード名として Web サービス *エンドポイント*を使用するようにエージェントを設定できます。Web サービス エンドポイントには、サービスについてサーバ名やポート番号のような追加の情報が含まれます。たとえば、*ClearingHouse.demobank.ca.com* Web サービスについて Web サービス エンドポイントが表示されるように選択した場合、Investigator で、そのノードは以下のように表示されることがあります。

```
http_localhost_8383_demobank_services_ClearingHouseService
```

エージェントの *webservices.pbd* ファイルを編集し、{namespace} または {servicename} のいずれを使用するかを指定することにより、表示される名前を変更できます。 ほとんどの場合、ネームスペースは、Investigator と SOA 依存マップに表示される最もわかりやすい名前です。 ただし、*webservices.pbd* ファイルと *IntroscopeAgent.profile* ファイルを編集する場合は、サービス エンドポイントを使用できます。

**Investigator および依存マップでノード名としてサービス エンドポイントを使用するには、以下の手順に従います。**

1. *<Agent_Home>* ディレクトリ内の *webservices.pbd* ファイルを開きます。
2. *{namespace}* のすべてのインスタンスを検索して、*{servicename}* に置き換えます。

   {namespace} が完全修飾メトリック名の一部としてトレーサに含まれるため、ファイルにはこの文字列が多数含まれています。
3. *webservices.pbd* ファイルを保存します。
4. *<_Agent_Home >* ディレクトリ内の *appmap-soa.pbd* ファイルを開きます。
5. *{namespace}* のすべてのインスタンスを検索して、*{servicename}* に置き換えます。
6. *appmap-soa.pbd* ファイルを保存します。
7. アプリケーション サーバを再起動します。

   アプリケーション サーバの再起動後、Investigator ツリーと SOA 依存マップには、ネームスペースではなく Web サービス エンドポイント名が表示されます。

   ネームスペースではなくサービス エンドポイント名を使用している場合は、クライアントとサーバの［概要］タブに表示されるネームスペースのラベルを変更することもお勧めします。

**クライアントとサーバの［概要］タブで使用されるラベルを変更するには、以下の手順に従います。**

1. *<EM_Home>/ext/xmltv/* ディレクトリ内の *ws.overview.tv.xml* ファイルを開きます。
2. *Namespaces* のすべてのインスタンスを検索して、*Services* に置き換えます。
3. *ws.overview.tv.xml* ファイルを保存します。
4. Enterprise Manager を再起動します。

   Enterprise Manager の再起動後、Investigator ツリーで［WebServices］、［Client］、または［Server］ノードを選択すると、［概要］タブに［サービス］ラベルが表示されます。

**Web Services Manager 7.0.x のメトリック命名規則を使用するには、以下の手順に従います。**

1. *<_Agent_Home >* ディレクトリ内の *IntroscopeAgent.profile* ファイルを開きます。
2. *com.wily.introscope.agent.soa.metricNameFormatting* プロパティをそのファイルに追加します。
3. *com.wily.introscope.agent.soa.metricNameFormatting* プロパティを以下のように設定します。
   ```
   com.wily.introscope.agent.soa.metricNameFormatting=/:
   ```
   この設定は、メトリック名内のスラッシュ（/）およびコロン（:）文字をアンダースコア（_）文字に置き換えます。この設定を使用して、http://CheckingAccount/demobank.ca.com は http_CheckingAccount_demobank.ca.com として表示されます。
4. *IntroscopeAgent.profile* ファイルを保存します。
5. アプリケーション サーバを再起動します。

   アプリケーション サーバの再起動後、Investigator ツリーと SOA 依存マップには、ネームスペースではなく Web サービス エンドポイント名が表示されます。

# 相関追跡を設定する

プロセスにまたがるトランザクション追跡では、エージェントはあるプロセスから別のプロセスへ渡すことができる相関識別子を挿入する必要があります。 エージェントはこの相関識別子を SOAP ヘッダまたは HTTP ヘッダに挿入できます。

ほとんどのサービスが SOAP メッセージングを使用するため、相関識別子は SOAP ヘッダにデフォルトで挿入され、そのヘッダから読み取られます。ただし、ある特定の状況では、HTTP ヘッダを使用して相関識別子を渡すことが優先される場合があります。 たとえば、まれな状況ですが、SOAP ヘッダに相関識別子を追加すると、メッセージが拒否される場合があります。これは、相関 ID によって SOAP ペイロードが変更されるためです。

アプリケーションがどのように SOAP メッセージを処理するか、どのようにセキュリティを実装したかに応じて、*IntroscopeAgent.profile* ファイルを修正して、HTTP ヘッダまたは SOAP ヘッダのいずれで相関 ID を渡すかを選択できます。

クライアント側およびサーバ側の相関識別子の処理を制御する 4 つのエージェント設定プロパティがあります。 これらのプロパティを使用して、相関識別子を SOAP ヘッダ、HTTP ヘッダ、または両方のヘッダ プロトコルに挿入するか、まったく挿入しないかを指定できます。 これらのプロパティは、以下の標準のエージェント プレフィックス *com.wily.introscope.agent* で始まります。

soapheaderinsertion.enabled

クライアントが SOAP ヘッダに相関識別子を挿入することを有効にします。

- クライアントが SOAP ヘッダを使用できるようにするには、このプロパティを true に設定します。
- クライアントが SOAP ヘッダに相関識別子を挿入しないようにするには、このプロパティを false に設定します。

デフォルトでは、このプロパティは true に設定されています。

httpheaderinsertion.enabled

クライアントが HTTP ヘッダに相関識別子を挿入することを有効にします。

- クライアントが HTTP ヘッダを使用できるようにするには、このプロパティを true に設定します。
- クライアントが HTTP ヘッダに相関識別子を挿入しないようにするには、このプロパティを false に設定します。

デフォルトでは、このプロパティは false に設定されています。

soapheaderread.enabled

サーバが SOAP ヘッダから相関識別子を読み取ることを有効にします。

- サーバが SOAP ヘッダを使用できるようにするには、このプロパティを true に設定します。
- サーバが SOAP ヘッダを読み取らないようにするには、このプロパティを false に設定します。

デフォルトでは、このプロパティは true に設定されています。

httpheaderread.enabled

サーバが HTTP ヘッダから相関識別子を読み取ることを有効にします。

- サーバが HTTP ヘッダを使用できるようにするには、このプロパティを true に設定します。
- サーバが HTTP ヘッダを読み取らないようにするには、このプロパティを false に設定します。

デフォルトでは、このプロパティは false に設定されています。

## クライアントおよびサーバ用のプロパティを設定する

これらのプロパティを設定して、エージェント単位で特定の動作を有効または無効にできます。 たとえば、サーバが SOAP と HTTP の両方に基づいたサービスからのメッセージを処理する場合、いずれかのタイプのヘッダ内の相関 ID を読み取るが、SOAP ヘッダに識別子を挿入しないように、そのサーバ上のエージェントを以下のように設定することもできます。

```
com.wily.introscope.agent.soapheaderread.enabled=true
com.wily.introscope.agent.httpheaderread.enabled=true
com.wily.introscope.agent.soapheaderinsertion.enabled=false
com.wily.introscope.agent.httpheaderinsertion.enabled=true
```

これらの設定を使用して、ローカル コンピュータは、他のエージェントによって送信された SOAP ヘッダから相関識別子を読み取ることができますが、HTTP ヘッダにのみ相関識別子を挿入できます。 ただし、追跡を有効にするには、Web サービスのクライアントとサーバが同じタイプのヘッダに対して相関識別子の挿入と読み取りを行う必要があることに注意してください。 たとえば、HTTP ヘッダに相関識別子を挿入するようにクライアントを設定する場合、HTTP ヘッダから識別子を読み取るようにサーバを設定する必要があります。

## 相関追跡を無効にする

SOAP と HTTP の両方に対して相関識別子の挿入を無効にし、プロセスにまたがるトランザクション追跡をオフにする場合、4 つのすべてのプロパティを false に設定できます。 例：

```
com.wily.introscope.agent.soapheaderread.enabled=false
com.wily.introscope.agent.httpheaderread.enabled=false
com.wily.introscope.agent.soapheaderinsertion.enabled=false
com.wily.introscope.agent.httpheaderinsertion.enabled=false
```

これらのプロパティを修正する前に、SOA 依存マップおよびプロセスにまたがるトランザクション追跡では、相関識別子をあるプロセスから別のプロセスへ渡す必要があることに注意してください。

これらのプロパティを false に設定することにより相関追跡を無効にする場合、SOA 依存マップに依存関係を適切に表示できず、依存関係メトリックを正確に収集できなくなり、トランザクション追跡が不完全になることがあります。

# SOAP ハンドラの挿入ポイントの設定

IntroscopeAgent.profile ファイル内の設定プロパティを使用して、SOAP ハンドラが SOAP ハンドラ チェーン内のどこに挿入されるかを制御できます。 これらのプロパティにより、暗号化や元の SOAP メッセージに対するシグネチャ確認が必要なアプリケーションに柔軟性が提供され、SOAP ヘッダの挿入がアプリケーションによる SOAP メッセージの検証の妨げにならないようにできます。

## SOAP ハンドラを挿入するためのクライアントおよびサーバのプロパティ

SOAP ハンドラを使用するアプリケーション サーバの場合は、以下の構成プロパティを使用して CA APM for SOA SOAP ヘッダを制御できます。

soa.client.prependhandler

SOAP ヘッダをクライアント上の SOAP ハンドラ チェーン内の先頭または末尾に挿入することを有効にします。

- SOAP ヘッダが SOAP ハンドラ チェーン内の先頭のハンドラによって挿入されるようにするには、このプロパティを true に設定します。
- SOAP ヘッダが SOAP ハンドラ チェーン内の末尾のハンドラによって挿入されるようにするには、このプロパティを false に設定します。

デフォルトでは、このプロパティは true に設定されています。

soa.server.appendhandler

SOAP ヘッダをサーバ上の SOAP ハンドラ チェーン内の先頭または末尾で読み取ることを有効にします。

- SOAP ヘッダが SOAP ハンドラ チェーン内の末尾のハンドラによって読み取られるようにするには、このプロパティを true に設定します。
- SOAP ヘッダが SOAP ハンドラ チェーン内の先頭のハンドラによって読み取られ、その後に削除されるようにするには、このプロパティを false に設定します。

デフォルトでは、このプロパティは true に設定されています。

## SOAP ハンドラのデフォルトの順序を変更する

デフォルトでは、クライアント上のハンドラ チェーン内の先頭のハンドラは SOAP ヘッダを挿入し、サーバ上のチェーン内の末尾のハンドラはヘッダを読み取ります。 一部のアプリケーションでは、CA APM for SOA SOAP ヘッダおよび元の SOAP メッセージが適切に渡されることを確認するため、デフォルト動作の修正が必要になる場合があります。たとえば、アプリケーションが SOAP メッセージのシグネチャを確認する必要がある場合、デフォルトの動作を修正することが必要になることがあります。

デフォルトのプロパティ設定を変更する理由を示すために、チェーン内の先頭のハンドラによって SOAP ヘッダが挿入されるアプリケーションを考えます。 SOAP メッセージはその後 2 番目のハンドラでシグネチャを付与されるとします。 メッセージがサーバ上で受信されると、先頭のハンドラはシグネチャを確認しようとします。 CA APM for SOA SOAP ヘッダは挿入されましたが、まだ読み取られて削除されていないため、シグネチャ確認は失敗し、メッセージが拒否されます。

構成プロパティにより、チェーン内の先頭のハンドラが CA APM for SOA SOAP ヘッダを読み取って削除するようにデフォルトの動作を変更することができます。 この設定により、チェーン内の次のハンドラが元の SOAP メッセージに対してシグネチャを確認できるようになります。 例：

```
com.wily.introscope.agent.soa.client.prependhandler=true
com.wily.introscope.agent.soa.server.appendhandler=false
```

プロパティのデフォルト設定はほとんどの環境で適切ですが、これらの構成プロパティにより、さまざまなアプリケーション シナリオに CA APM for SOA を適応させる柔軟性が得られます。

# SOA 依存マップに関する制限を設定する

通常、Enterprise Manager 上に保存された依存マップ データは、検出されたアプリケーション間のすべての依存関係を表しており、展開されているサービス指向アーキテクチャの完全なモデルを提供します。 しかし、非常に大規模または複雑な SOA 環境では、すべての SOA コンポーネントとそれらの依存関係を完全に表現すると、Enterprise Manager 自体のパフォーマンスやオペレーションに影響する場合があります。

依存マップが Enterprise Manager のパフォーマンスに影響を与えないようにするために、マップのサイズおよび複雑さを制限する設定プロパティがあります。デフォルトでは、これらの設定プロパティは、最大 5,000 のノードおよび最大 25,000 の依存関係（1 ノードあたり 5 つの依存関係という割合）に依存マップを制限します。

これらの制限を制御するために、*IntroscopeEnterpriseManager.properties* ファイルで以下の Enterprise Manager 設定プロパティを設定できます。

dependencymap.max.vertices

依存関係データがスタンドアロンまたはコレクタ Enterprise Manager で保存できるノードの最大数を設定します。

デフォルトでは、このプロパティは 5000 に設定されています。

dependencymap.max.edge.ratio

依存関係対ノードの最大比率を設定して、スタンドアロンまたはコレクタ Enterprise Manager 上で保存できる依存関係データの全体的な複雑さを制御します。

デフォルトでは、このプロパティは 5 に設定されています。

ほとんどの組織の場合、SOA ネットワークには 5000 未満のコンポーネントが必要です。また、標準の依存関係比率は 1 つのコンポーネントあたり 1 つまたは 2 つの依存関係です。 デフォルト設定により、したがって、ほとんどの組織のニーズを超えるサイズおよび複雑さが可能です。 ただし、まれな場合ですが、デフォルト値を修正することもできます。たとえば、以下のようにプロパティを修正することもできます。

- デフォルト値が SOA 環境に対応しない場合に、依存マップのサイズと複雑さを大きくする。 変更がどのように Enterprise Manager のパフォーマンスに影響を与える可能性があるかを考慮する必要があります。
- SOA 環境にデフォルト値が許可するほど多くのノードおよび依存関係が必要でない場合に、依存マップのサイズおよび複雑さを意図的に制限する。 何らかの変更により SOA 依存マップ モデルが余計に不完全になる可能性があるかどうかを考慮する必要があります。

たとえば、データの保存対象となるノードを減らすが、1 つのノードあたりの依存関係を増やすには、以下のようにプロパティを修正します。

```
com.wily.introscope.soa.dependencymap.max.vertices=2000
com.wily.introscope.soa.dependencymap.max.edge.ratio=8
```

# 第 7 章: OSB（Oracle Service Bus）を監視する

Oracle Service Bus（OSB）は、疎結合のサービスのコンシューマとサービス プロバイダとの間のメッセージ処理および通信に使用されます。 SOA Extension for Oracle Service Bus により、パイプライン、プロキシ サービス、およびトランスポート プロトコルのような重要なコンポーネントを通じて Oracle Service Bus のオペレーションを監視できます。

このセクションでは、Oracle Service Bus 環境のパフォーマンス、可用性、全体の稼働状況の監視および分析に使用できる、OSB 固有のダッシュボード、メトリック、アラートについて説明します。

このセクションには、以下のトピックが含まれています。



## Oracle Service Bus（OSB）について

サービス指向アーキテクチャでは、Enterprise Service Bus によって通常、サービス コンシューマとサービス プロバイダとの間のメッセージング レイヤが提供されます。 Enterprise Service Bus により、分散異機種環境にわたるデータとメッセージに対して検証、変換、ルーティング、セキュリティが有効になります。

Oracle Service Bus は、Enterprise Service Bus の一例であり、プロキシ インターフェースの軽量の中間レイヤでメッセージの仲介とメディエーション、およびサービス ライフサイクルの管理に使用されます。

Oracle Service Bus を使用してメッセージの処理ルールを定義している場合は、以下のものに関するメトリックを使用して Oracle Service Bus のオペレーションを監視できます。

### ビジネス サービス

ビジネス サービスは、Oracle Service Bus がクライアントである外部サービスの定義です。

外部 Web サービスは、外部システムに実装され、そのシステムでホストされます。それらのサービスを使用するために、Oracle Service Bus は、呼び出すインターフェース、その呼び出し方法、想定する呼び出しの結果を認識している必要があります。OSB 内のビジネス サービスでは、OSB が起動して外部システムと対話できるように、外部インターフェースがモデル化されています。 OSB 内では、ビジネス サービスの設定には、そのインターフェース、トランスポート設定、およびセキュリティ設定が含まれます。

SOA Extension for OSB を使用している場合、［OSB］-［BusinessServices］ノードの下で、ビジネス サービスがどのように外部システムと対話しているかを監視し、それらの全体の稼働状況に関するデータを収集できます。

### パイプライン

パイプラインは、要求、応答、またはエラー メッセージ フローについて特定の手順を処理する名前付きのシーケンスです。

プロキシ サービスの処理ロジックを実装するには、要求および応答パイプラインがパイプラインのペア ノード内で対になる必要があります。これらのパイプラインのペア ノードは、他のノードと組み合わせて 1 つのツリー構造を作成することができます。これによってフロー全体を管理することができます。エラー パイプラインは、メッセージ フローのステージおよびノードのエラーと、メッセージ フローまたはビジネス サービスのエラーを処理します。

SOA extension for OSB を使用して、［OSB］-［Pipelines］ノードの下で、要求と応答のパイプラインのパフォーマンスを監視し、それらの全体の稼働状況に関するデータを収集できます。

### プロキシ サービス

プロキシ サービスは、サービス バスがローカルで実装してホストする中間 Web サービスの定義です。

Oracle Service Bus はプロキシ サービスを使用して、ビジネス サービスとサービス クライアント（プレゼンテーション アプリケーション、その他のビジネス サービスなど）との間のメッセージをルーティングします。

プロキシ サービスの設定には、インターフェース設定、トランスポート設定、セキュリティ設定、メッセージ フローの定義が含まれます。メッセージ フローは、メッセージがプロキシ サービスを通過する際の処理方法を決定する論理を定義します。

SOA Extension for OSB を使用して、［OSB］ -［ProxyServices］ノードの下で、プロキシ サービスのパフォーマンスを監視し、それらの全体の稼働状況に関するデータを収集できます。

### トランスポート

Transports は、メッセージを送信および配信するためのメカニズムを定義し、Oracle Service Bus がサポートするあらゆるトランスポート プロトコルが含まれることがあります。

トランスポート プロバイダはライフ サイクルとトランスポート エンドポイントのランタイム動作を管理します。ターゲット エンドポイントは、メッセージの発信元または送信先のリソースです。 OSB のネイティブ プロバイダを使用して、これらのトランスポート プロトコルを必要とするプロキシおよびビジネス サービスを設定することができます。 また、カスタム トランスポート プロバイダを作成またはインストールすることもできます。

SOA Extension for OSB を使用して、［OSB］ -［Transports］ノードの下で、サポートされているトランスポート プロトコルのすべてを監視し、インバウンドおよびアウトバウンド エンドポイントに関するメトリックを収集できます。

### UDDI

UDDI（Universal Description, Discovery, and Integration）は、インターネットにビジネスを登録する、ワールドワイドなビジネス向けの、プラットフォーム非依存の、XML ベースのレジストリです。

UDDI はオープンな業界イニシアチブで、それぞれのビジネスでサービス リストを発行し、他のビジネスのサービス リストを検出でき、また、サービスまたはソフトウェア アプリケーションがインターネット上でどのように連携するのかを定義することができます。

Oracle Service Bus、および UDDI バージョン 3.0 に準拠している UDDI レジストリを使用して、以下のことが可能です。

- プロキシ サービスに関する情報をレジストリに発行する。 レジストリは、Web Services Description Language（WSDL）、SOAP、または XML ベースである場合があります。
- レジストリ内の特定のサービスを照会する、または利用可能なすべてのサービスの一覧を表示する。 ビジネス エントリ、サービス名のパターン、またはその両方で検索できます。
- レジストリからビジネス サービスをインポートする。

SOA Extension for OSB を使用して、［OSB］-［UDDI］ノードの下で、UDDI レジストリを監視し、発行、インポート、および照会オペレーションに関するメトリックを収集できます。

### XQuery

Oracle Service Bus は、XQuery エンジンの Oracle Data Services Platform の実装を使用したデータ変換のために XQuery をサポートしています。

Oracle XQuery エンジンは、変換マップを使用してデータ型間のマッピングを記述します。また、Oracle Service Bus は XQuery を使用してデータ マッピングをサポートします。 変換を作成、解析、および実行することができます。

SOA extension for OSB を使用して、［OSB］-［XQuery］ノードの下で、XQuery の作成、解析、実行について XQuery 変換を監視し、それらのオペレーションに関するメトリックを収集できます。

# Oracle Service Bus の監視を有効にする方法

Oracle Service Bus の監視を有効にする手順の概要は以下のとおりです。

1. Oracle Service Bus のサポートされているバージョンがインストールされていることを確認します。

   **注**: システム要件については、「*Compatibility Guide*」を参照してください。

2. エージェントと CA APM for SOA がインストールされて有効になっていることを確認します。

3. Oracle Service Bus の監視用の適切なディレクティブを含むようにエージェント プロファイルを設定することによって、Oracle Service Bus の監視用のエージェントを有効にします (P. 145)。スタンドアロン エージェント インストーラまたは応答ファイルを使用してエージェントを有効にしている場合は、この手順をスキップできます。

4. *<EM_Home>*/examples/SOAExtensionForOSB ディレクトリから適切な Enterprise Manager ディレクトリにファイルを移動して、Enterprise Manager 拡張機能を有効にします (P. 148)。

## エージェントに対して Oracle Service Bus の監視を有効にする

エージェントをインストールし、アプリケーション サーバとして［WebLogic］を選択していると、CA APM for Oracle Service Bus を追加して有効にできます。また、エージェントをインストールした後に CA APM for Oracle Service Bus を手動で有効にすることもできます。

エージェントをインストールするときに、CA APM for Oracle Service Bus を有効にした場合は、エージェント プロファイルがデフォルト設定で自動的に構成されます。エージェント拡張機能を有効にするためのそのほかの手順は不要です。

エージェントをインストールするときに、CA APM for for Oracle Service Bus を有効にしなかった場合は、監視を有効にするようにエージェント プロファイルを手動で構成する必要があります。

### CA APM for Oracle Service Bus を手動で有効にする方法

1. エージェントと CA APM for SOA がインストールされて有効になっていることを確認します。
2. CA APM for Oracle Service Bus のディレクトリ *SOAExtensionForOSB* が *<Agent_Home>/examples* ディレクトリ内にあることを確認して、*<Agent_Home>/examples/SOAExtensionForOSB/ext* ディレクトリからファイルを *<Agent_Home>/core/ext* ディレクトリにコピーします。
3. *IntroscopeAgent.profile* ファイルをテキスト エディタで開きます。
4. *IntroscopeAgent.profile* ファイル内の *introscope.autoprobe.directivesFile* プロパティに、該当する *OSB-full.pbl* または *OSB-typical.pbl* ファイルを追加します。

   該当する ProbeBuilder ディレクティブの選択の詳細については、「Oracle Service Bus 用のディレクティブ ファイルについて (P. 147)」を参照してください。
5. デフォルト設定を変更する場合は、*IntroscopeAgent.profile* ファイルで追加の SOA 固有のエージェント設定プロパティを修正します。

   どのようにセキュリティを実装したかに応じて、一部のアプリケーションにはデフォルト設定に対する変更が必要になる場合があることに注意してください。

   たとえば、Web サービスの通信を暗号化する場合、エージェントのデフォルト設定を修正して、エージェントに対して HTTP 相関を有効にし、SOAP 相関を無効にする必要があります。 デフォルトでは、SOAP 相関と HTTP 相関の両方は Oracle Service Bus に対して有効になっています。HTTP または SOAP 相関の設定プロパティの設定の詳細については、「相関追跡を設定する (P. 133)」を参照してください。
6. *IntroscopeAgent.profile* ファイルに対する変更を保存して、テキスト エディタを閉じます。

## Oracle Service Bus 用のディレクティブ ファイルについて

*IntroscopeAgent.profile* で *introscope.autoprobe.directivesFile* プロパティを設定するときに、標準（デフォルト）またはフル インスツルメンテーションを選択できます。これにより、エージェントを展開している環境に合わせて、監視するレベル、メトリックの可視性、およびパフォーマンスのオーバーヘッドを必要に応じて制御できます。その後、特定のトレーサ グループに対して追跡をオンまたはオフにすることにより、標準またはフル トグル ファイルを使用して、特定のコンポーネントの監視を微調整できます。

OSB-full.pbl

すべての Oracle Service Bus コンポーネントのオペレーションへの深い可視性を提供する、フル インスツルメンテーション。

フル インスツルメンテーションにより、詳細なレポートが提供されますが、追加のオーバーヘッドが必要です。 このインスツルメンテーションは通常、実運用環境ではなくテストまたは開発環境にお勧めします。

デフォルトでは、以下のディレクティブ ファイルが OSB-full.pbl ファイルの一覧に表示されます。

- OSB-toggles-full.pbd
- OSB.pbd

OSB-typical.pbl

オーバーヘッドが重要である実運用環境での、重要な Oracle Service Bus コンポーネントの標準の監視。

標準インスツルメンテーションは、簡潔なレポートが提供され、オーバーヘッドも少ないため、実運用環境にお勧めします。

デフォルトでは、以下のディレクティブ ファイルが OSB-typical.pbl ファイルの一覧に表示されます。

- OSB-toggles-typical.pbd
- OSB.pbd

OSB-toggles-full.pbd

フル インスツルメンテーションを使用するときに、Oracle Service Bus コンポーネントの監視のオンとオフを切り替えます。

このファイルは他のディレクティブ ファイルで有効になる追跡用のディレクティブを提供します。デフォルトでは、このファイルでほとんどのトレーサ グループはオンになっています。

OSB-toggles-typical.pbd

標準インスツルメンテーションを使用するときに、Oracle Service Bus コンポーネントの監視のオンとオフを切り替えます。 このファイルは他のディレクティブ ファイルで有効になる追跡用のディレクティブを提供します。デフォルトでは、ごく一部のトレーサ グループがオンになっています。

同じ Enterprise Manager にデータをレポートするすべてのエージェントが同じインスツルメンテーション レベルを使用する必要があることに注意してください。また、Enterprise Manager は標準またはフル インスツルメンテーション エージェントのいずれかで実行されるように設定する必要があることにも注意してください。 たとえば、標準インスツルメンテーションを使用するようにエージェントを設定する場合は、Enterprise Manager ファイルの標準バージョンを展開する必要があります。

## Oracle Service Bus の Enterprise Manager 拡張機能を有効にする

Enterprise Manager をインストールする際に、CA APM for Oracle Service Bus を追加して有効にできます。 また、Enterprise Manager をインストールした後に CA APM for Oracle Service Bus を手動で有効にすることもできます。

Enterprise Manager をインストールすると、CA APM for Oracle Service Bus のファイルはデフォルトでは *<EM_Home>*/examples ディレクトリにインストールされます。 CA APM for Oracle Service Bus を有効にするには、*<EM_Home>*/examples ディレクトリにある Enterprise Manager のファイルを Enterprise Manager ホーム ディレクトリ内の適切な場所にコピーまたは移動します。

**注**: CA APM for Oracle Service Bus を使用するには、CA APM for SOA を Enterprise Manager 上で有効にする必要があります。

次の手順に従ってください:

1. CA APM for Oracle Service Bus のディレクトリ（SOAExtensionForOSB）が *<EM_Home>*/examples ディレクトリ内にあることを確認します。
2. *<EM_Home>*/examples/SOAExtensionForOSB ディレクトリから標準またはフル インスツルメンテーション用の該当するファイルを Enterprise Manager ディレクトリ構造内の対応する場所にコピーします。 たとえば、標準インスツルメンテーションを使用する場合は、*<EM_Home>*/examples/SOAExtensionForOSB/ext ディレクトリから com.wily.powerpacks.osb.emext.calculator_typical.jar ファイルを *<EM_Home>*/ext ディレクトリにコピーします。

   **注**: 同じ Enterprise Manager にデータをレポートするすべてのエージェントに対して、同じインスツルメンテーション レベルを使用します。たとえば、Enterprise Manager ファイルの標準バージョンを展開する場合、Enterprise Manager と通信するすべてのエージェントは、標準インスツルメンテーションを使用するように設定します。フル インスツルメンテーションを使用するエージェントは、標準モード用に設定された Enterprise Manager にデータをレポートしません。
3. Enterprise Manager がクラスタ化環境内のコレクタである場合は、*<EM_Home>*/config/module ディレクトリから Oracle Service Bus Management Module、OSB_ManagementModule_typical.jar、または OSB_ManagementModule_full.jar を削除します。

   この管理モジュールは、MOM コンピュータとして使用している Enterprise Manager の *<EM_Home>*/config/modules ディレクトリにのみコピーします。 他のすべてのファイルとスクリプトは、コレクタ Enterprise Manager と MOM Enterprise Manager の両方にインストールする必要があります。

4. SOA Extension for Oracle Service Bus の以前のバージョンからアップグレードする場合は、Enterprise Manager ファイルの古いバージョンを削除します。

   - Enterprise Manager に CA APM for Oracle Service Bus の以前のリリースがインストールされている場合は、CA APM for Oracle Service Bus の新しいバージョンを使用し始める前に、Enterprise Manager ファイルの古いバージョンを手動で削除します。
   - CA APM for Oracle Service Bus の以前のリリースからアップグレードした場合は、Enterprise Manager のホーム ディレクトリから以下のファイルを削除します。

     ```
     <EM_home>/config/modules/OSB_ManagementModuleV<バージョン>.jar
     <EM_home>/ext/xmltv/OSB.overview.tv.xml
     <EM_home>/product/enterprisemanager/plugins/
     com.wily.powerpacks.osb.emext.calculator_<バージョン>.jar
     ```

     たとえば、バージョン 8.1.1 からアップグレードしている場合は、以下のファイルを削除します。

     ```
     OSB_ManagementModuleV8.1.1.0.jar
     OSB.overview.tv.xml
     com.wily.powerpacks.osb.emext.calculator_8.1.1.0.jar
     ```

5. Workstation を再起動します。

   CA APM for Oracle Service Bus に固有のダッシュボードと［概要］タブがロードされます。

**詳細**:

Enterprise Manager 上で拡張機能を有効にする (P. 44)

# ダッシュボードを使用して Oracle Service Bus を監視する

SOA Extension for Oracle Service Bus には、アプリケーション環境全体の稼働状況を監視するために使用できる、複数の事前設定されたダッシュボードが含まれています。 ダッシュボードは、ユーザが問題をすばやく診断して解決できるように、展開されたエージェントからデータを集計してパフォーマンス情報を要約します。

通常、ダッシュボードは以下の機能を備えているため、環境を監視するための起点として使用されます。

- Oracle Service Bus の重要なコンポーネントの全体の稼働状況、パフォーマンス、可用性、現在のステータスをひとめで監視する。
- 警告または危険のしきい値を超えたことがより低レベルのメトリックによって検出されると、実運用アプリケーション環境での潜在的な問題について早期の通知を受け取る。
- パフォーマンスの情報にドリルダウンして、どの Oracle Service Bus コンポーネント、トランスポート プロトコル、エンドポイント、またはオペレーションが原因で遅延またはエラーが発生しているかを切り分けて特定する。

事前設定された Oracle Service Bus ダッシュボードは、展開した OSB Management Module（*OSB_ManagementModule_typical.jar* または *OSB_ManagementModule_full.jar*）の一部として、Enterprise Manager Extension for Oracle Service Bus にパッケージされています。

OSB Management Module では、Oracle Service Bus に対応する以下の事前設定されたダッシュボードが提供されます。

**OSB ホーム**

Oracle Service Bus の重要なコンポーネントを使用したワークフローのトップレベルのビュー。たとえば、すべてのトランスポート、プロキシ サービス、パイプライン、ビジネス サービスの全体の稼働状況に関するアラート インジケータなど。

**OSB プロキシ サービス - 概要**

すべてのプロキシ サービスの要約されたステータス。たとえば、Average Response Time （平均応答時間）、Errors Per Interval（間隔ごとのエラー数）、Stall Count（ストール数）に関するアラートとグラフ、低速プロキシ サービス 10 件の一覧など。

**OSB プロキシ サービス - 詳細**

プロキシ サービスにわたる要求および応答オペレーションに関する個別の要約されたメトリック。たとえば、Average Response Time （平均応答時間）、Errors Per Interval（間隔ごとのエラー数）、Stall Count（ストール数） に関するアラートとグラフなど。

### OSB ビジネス サービス - 概要

すべてのビジネス サービスの要約されたステータス。たとえば、Average Response Time （平均応答時間） および Errors Per Interval （間隔ごとのエラー数）に関するアラートとグラフ、低速ビジネス サービス 10 件の一覧など。

### OSB パイプライン - 概要

すべてのパイプラインの要約されたステータス。たとえば、Average Response Time（平均応答時間）、Errors Per Interval（間隔ごとのエラー数）、Stall Count （ストール数） に関するアラートとグラフ、低速パイプライン 10 件の一覧など。

### OSB トランスポート I - 概要

一般的に使用されるトランスポート タイプ（HTTP/HTTPS、JMS、EJB、FTP、Email および SFTP）について要約された稼働状況。たとえば、すべてのトランスポート タイプにわたる Average Response Time（平均応答時間）、Errors Per Interval （間隔ごとのエラー数）、Stall Count （ストール数） に関するアラート、HTTP/HTTPS、JMS、EJB、FTP、Email、および SFTP トランスポート タイプに関する個別の Average Response Time （平均応答時間） のグラフなど。

### OSB トランスポート II - 概要

追加のトランスポート タイプ（Tuxedo、MQ、WS、JPD、SB、DSP、File、および Local）に関する個別の Average Response Time （平均応答時間）のグラフ。

### OSB トランスポート エンドポイント - 概要

すべてのトランスポート タイプにわたるインバウンドおよびアウトバウンド サービスについて要約されたステータス。たとえば、すべてのインバウンド サービスとアウトバウンド エンドポイントについて Errors Per Interval （間隔ごとのエラー数） と Stall Count （ストール数）に関する Average Response Time（平均応答時間）のグラフとアラート、低速インバウンド サービス 10 件の一覧、低速アウトバウンド エンドポイント 10 件の一覧など。

OSB XQuery - 概要

すべての XQuery 実行の試行について要約されたステータス。たとえば、Execute オペレーションについて Average Response Time （平均応答時間）、Errors Per Interval （間隔ごとのエラー数）、Stall Count （ストール数） に関するグラフとアラート、Average Response Time （平均応答時間） および解析オペレーションの Errors Per Interval （間隔ごとのエラー数） に関するグラフなど。

OSB UDDI - 概要

UDDI レジストリ オペレーションについて要約されたステータス。たとえば、UDDI レジストリに対するインポートおよび発行オペレーションについて、Average Response Time （平均応答時間） および Errors Per Interval （間隔ごとのエラー数） に関するアラートとグラフ。

事前構成済みのダッシュボードは、Workstation コンソールを使用して表示できます。 また、カスタム ダッシュボードを含めるように OSB Management Module を拡張するか、カスタム メトリックまたはアラートを含めるようにデフォルトのダッシュボード定義を修正することもできます。

**注**: ダッシュボードを作成および変更する方法の詳細については、「CA APM 設定および管理ガイド」を参照してください。

**次の手順に従ってください**:

1. Enterprise Manager を起動します（現在実行されていない場合）。
2. Workstation を起動し、SOA Extension for Oracle Service Bus がインストールされている Enterprise Manager にログインします。
3. ［Workstation］-［新規コンソール］を選択します。
4. ［ダッシュボード］ドロップダウン リストから OSB ダッシュボードの 1 つを選択します。

   たとえば、トランスポート、プロキシ、サービス、パイプライン、およびビジネス サービスなどのような OSB ワークフローの重要なコンポーネントのすべてに関するアラート インジケータを使用して、Oracle Service Bus のアーキテクチャの概要を確認するには、［OSB ホーム］ダッシュボードを選択します。

5. ダッシュボードでアラートをダブルクリックして、そのコンポーネントのダッシュボードを開きます。

   たとえば、プロキシ サービス応答時間のアラートをダブルクリックして、［OSB プロキシ サービス - 概要］ダッシュボードに移動して、低速プロキシ サービスの一覧を確認したり、すべてのプロキシ サービスについて応答時間、ストール数、エラーに関するトップレベルのアラート インジケータを確認したりできます。

6. ダッシュボードで特定のプロキシ サービス、パイプライン、またはトランスポートをダブルクリックして、さらなる分析のために Investigator を開きます。

   たとえば、［OSB プロキシ サービス - 概要］ダッシュボードから、［低速プロキシ サービス 10 件］リストで特定のプロキシ サービスをダブルクリックして、Investigator でその詳細を表示できます。集約されたメトリックを通じて、選択したパイプラインの個々のコンポーネントにドリル ダウンし続けて、ダッシュボード アラートで通知された遅延またはエラーしきい値の根本原因を特定できます。

**注**: Investigator で OSB 固有の情報を表示することの詳細については、「Oracle Service Bus に関するメトリックを理解および表示する (P. 154)」を参照してください。 ダッシュボード間、またはダッシュボードと Workstation Investigator との間で移動することの詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# Oracle Service Bus に関するメトリックを理解および表示する

Investigator ツリー内で移動すると、Oracle Service Bus のほとんどのコンポーネントおよびオペレーションについて、CA Introscope® の標準メトリックを表示できます。標準メトリックのデータは、OSB 固有のメトリック カテゴリ別に収集されて集約され、Investigator ツリーでノードおよびサブノードとして表示されます。 表示された特定のメトリック カテゴリおよびノード名は、アプリケーションが使用する Oracle Service Bus リソースによって異なります。

Investigator ツリーを通じて移動すると、個々のオペレーションに関する低レベルのメトリック、または選択するノードに応じた集約されたメトリックを表示することもできます。これにより、さまざまな OSB コンポーネントの全体の稼働状況を監視できます。

CA CPM for SOA によって提供される標準メトリックおよびそのほかのメトリックの概要については、「利用可能なメトリック (P. 64)」を参照してください。OSB コンポーネントに関するメトリックの追加の情報については、以下のセクションを参照してください。

- BusinessServices に関するメトリック（P. 156）
- Pipelines に関するメトリック（P. 156）
- ProxyServices に関するメトリック（P. 157）
- Transports に関するメトリック（P. 158）
- UDDI に関するメトリック（P. 159）
- XQuery に関するメトリック（P. 159）

**Investigator で Oracle Service Bus に関するメトリックを表示および操作するには、以下の手順に従います。**

1. エージェント ノードを展開し、［OSB］を選択して、Oracle Service Bus コンポーネントの［概要］タブを表示します。

   ［概要］タブには、Oracle Service Bus 用に定義したプロキシ サービスおよびビジネス サービスのすべてに関する概要情報が一覧表示されます。

2. ［概要］タブでプロキシ サービスまたはビジネス サービスをダブルクリックして、そのプロキシ サービスまたはビジネス サービスに関するすべての集約されたメトリックをグラフィカル形式で表示します。

3. ［OSB］ノードを展開して、Business Services、Pipelines、Proxy Services、Transports、UDDI、および XQuery といった、トップレベルの Oracle Service Bus メトリック カテゴリのサブノードを表示します。

4. サブノードをクリックすると、そのメトリック カテゴリに関するサマリ情報が［概要］タブに表示されます。 たとえば、［概要］タブで一覧表示されたビジネス サービスのすべてを表示するには、［OSB］ - ［BusinessServices］をクリックします。

5. 個々のサービスまたはオペレーション、およびそれらの個々に関連付けられたメトリックを参照するには、サブノードを展開します。

   たとえば、Oracle Service Bus で設定したビジネス サービスのすべてを参照するには、［BusinessServices］ノードを展開します。その後、個々のビジネス サービス名を展開して、その特定のサービスに関する標準メトリックを参照できます。

## BusinessServices に関するメトリック

Oracle Service Bus の場合、ビジネス サービスは外部システムによってホストされる外部エンタープライズ サービスとして定義されます。 以下の標準メトリックのみが、［OSB］ > ［BusinessServices］ノードの下の Oracle Service Bus ビジネス サービスに使用可能です。

- Average Response Time (ms)
- Errors Per Interval
- Responses Per Interval

Oracle Service Bus プラットフォームで、ビジネス サービスは、Oracle Service Bus がメッセージをやり取りできる外部サービスです。ビジネス サービスへのメッセージが外部サービスへの呼び出しを表すため、そのメッセージは要求と応答のトランザクションで構成されません。 したがって、ビジネス サービスに関する［Average Response Time（平均応答時間）］メトリックの計算は、ビジネス サービス実行時間の合計に基づきます。

CA Introscope® は Oracle Service Bus のトランスポート時間、実際のサービスの実行時間、およびネットワーク関連時間を合計することにより、ビジネス サービス実行時間を計算します。 たとえば、OSB トランスポート時間が 10 ミリ秒、実際のサービスの実行時間が 18 ミリ秒、ネットワーク関連時間が 5 ミリ秒である場合、ビジネス サービス実行時間は 33 ミリ秒です。

## Pipelines に関するメトリック

パイプラインは、要求、応答、またはエラー メッセージ フローについて特定の手順を処理する名前付きのシーケンスです。 OSB フル インスツルメンテーションを使用する場合、Oracle Service Bus パイプラインに使用可能な CA Introscope® の標準メトリックのすべては、［OSB］-［Pipelines］ノードの下にあります。

OSB の標準インスツルメンテーションを使用する場合、［Pipelines］ノードは Investigator に表示されません。 Investigator でパイプラインに関するメトリックを表示する必要がある場合は、*OSB-toggles-typical.pbd* ファイルを修正して、パイプラインの追跡を有効にします。

## ProxyServices に関するメトリック

プロキシ サービスは、Oracle Service Bus がビジネス サービス間でメッセージをどのようにルーティングするかを定義します。 外部の Web サービスやデータベースからのメッセージは、プレゼンテーション アプリケーションやその他のビジネス サービスなどのサービス クライアントにルーティングできます。 Oracle Service Bus プロキシ サービスに使用可能な CA Introscope® の標準メトリックのすべては、［OSB］-［ProxyServices］ノードの下にあります。

プロキシ サービスは、要求と応答のトランザクションで構成されます。個々のプロキシ サービス名の下のメトリックはすべて、［Request］および［Response］サブノードの下に一覧表示されたメトリックに基づいて集約されたメトリックです。

たとえば、mortgageBroker1 ～ loanGateway1 プロキシ サービスに対する Concurrent Invocations （同時進行中の呼び出し） の数を求めるには、loanGateway1 要求の数と同時実行中の loanGateway1 応答の数を加算します。

各トランザクションの応答時間は、往復時間の合計として定義されます。トランザクションの開始は、呼び出し元から要求を受け取った時点です。トランザクションは、応答が呼び出し元に返されると終了します。

Average Response Time （平均応答時間）は、トランザクション時間の合計をトランザクションの数で割ることにより計算されます。

## Transports に関するメトリック

トランスポートは、メッセージを送信および配信するためのメカニズムを定義します。 Oracle Service Bus の場合、Oracle Service Bus でデフォルトでサポートされる以下のトランスポート タイプなど、最も一般的なトランスポート プロトコルに関するメトリックを収集できます。

- HTTP/HTTPS
- JMS
- EJB
- FTP
- Email
- SFTP
- Tuxedo
- MQ
- WS
- JPD
- SB
- DSP
- File
- Local

プロキシ サービスが特定のトランスポート タイプを使用する場合、そのトランスポート タイプとそのメトリックは Investigator ツリーに表示されます。 トランスポート タイプがどのサービスによっても使用されない場合、そのトランスポート タイプは Investigator ツリーに表示されません。

プロキシ サービスがカスタム トランスポート タイプを使用する場合、そのトランスポート タイプに定義された Java のパッケージ名が Investigator に表示されます。 たとえば、OSB Console で *com.bea.wli.sb.test.service* というカスタム トランスポート タイプを使用すると、Investigator にはトランスポート タイプとしてこの名前が表示されます。

使用している特定のトランスポート プロトコルに使用可能な CA Introscope® の標準メトリックのすべては、［OSB］ - ［トランスポート］ ノードの下にあります。 トランスポート プロトコルごとに表示されるノードは、標準またはフル インスツルメンテーションのいずれを設定したかによって異なります。

たとえば、標準インスツルメンテーションを使用する場合、メトリックは［送信］および［受信］ノードの下で要約されます。 フル インスツルメンテーションを使用しており、トランスポート タイプを展開する場合、インバウンドおよびアウトバウンド サブノードが表示され、トランスポート タイプについて、*getRequestPayload*、*send*、または *sendMessageAsync* のような個々のオペレーションに関するメトリックが一覧表示されます。

## UDDI に関するメトリック

AquaLogic または Oracle Service Bus Registry を設定して、UDDI 3.0 準拠のレジストリと対話するようにする必要があります。その後、UDDI メトリックを収集して表示できるようになります。 UDDI を使用するように Oracle Service Bus を設定している場合、インポート、照会、発行オペレーションに関する集約されたメトリックを収集して表示できます。

インポート、照会、および発行オペレーションに使用可能な CA Introscope® の標準メトリックのすべては、［OSB］ - ［UDDI］サブ-ノードの下にあります。

## XQuery に関するメトリック

XQuery を使用するように Oracle Service Bus を設定している場合、作成、解析、実行オペレーションに関する集約されたメトリックを収集して表示できます。

作成、解析、および実行オペレーションに使用可能な CA Introscope® の標準メトリックのすべては、［OSB］ - ［XQuery］サブ-ノードの下にあります。

作成および解析メソッドのメトリックは、それぞれの XQuery に 1 度だけ入力されます。 実行メソッドのメトリックは、XQuery の式または条件が実行されるたびに入力されます。

# デフォルトの Oracle Service Bus メトリック グループを表示する

SOA Extension for Oracle Service Bus には、デフォルトのダッシュボードおよびアラートの定義に使用されるデフォルト メトリック グループが含まれています。これらのデフォルト メトリック グループをカスタム ダッシュボードやカスタム アラートで使用することもできます。

デフォルト メトリック グループは、展開した OSB Management Module（*OSB_ManagementModule_typical.jar* または *OSB_ManagementModule_full.jar*）の一部として、Enterprise Manager Extension for Oracle Service Bus にパッケージされています。

デフォルトのメトリック グループは、Workstation の管理モジュール エディタを使用して表示できます。 また、カスタム メトリック グループを含めるか、カスタム ダッシュボードまたはアラートでデフォルト メトリック グループを使用するように、OSB Management Module を拡張できます。

**次の手順に従ってください**:

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］ > ［管理モジュール］ ノードで、デプロイ済みの OSB_ManagementModule (*Super Domain*) を展開します。

   たとえば、標準の設定を展開している場合は、［OSB_ManagementModule Typical (*Super Domain*)］を展開します。
3. Oracle Service Bus Management Module 用に定義されたメトリック グループのすべてを表示するには、［Metric Groupings］ノードを展開します。
4. 特定のメトリック グループをクリックすると、［ビューア］ペインにその定義が表示されます。

任意のメトリック グループのデフォルト設定を変更したり、ユーザ独自のカスタム メトリック グループを作成したりできます。

**注**: メトリック グループを作成および変更する方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# デフォルトの Oracle Service Bus アラートを表示する

SOA Extension for Oracle Service Bus には、事前設定されたダッシュボードで使用されるデフォルト アラート定義が含まれています。 これらのデフォルト アラートをカスタム ダッシュボードで使用することもできます。 ほとんどのデフォルト アラートは、デフォルトの警告しきい値と危険しきい値を使用して、しきい値を超えるか重要度が高くなった場合にコンソールに通知を送信するように事前設定されています。

デフォルト メトリック定義は、展開した OSB Management Module（*OSB_ManagementModule_typical.jar* または *OSB_ManagementModule_full.jar*）の一部として、Enterprise Manager Extension for Oracle Service Bus にパッケージされています。

デフォルトのアラート定義は、Workstation の管理モジュール エディタを使用して表示できます。また、OSB Management Module を拡張して、カスタム アラート定義および通知タイプを含めるようにできます。カスタム ダッシュボードでデフォルト アラート定義を使用することもできます。

**次の手順に従ってください**:

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］ -［管理モジュール］ > 展開した［OSB_ManagementModule (*SuperDomain*)］を展開します。

   たとえば、標準の設定を展開している場合は、［OSB_ManagementModule Typical (*SuperDomain*)］を展開します。
3. ［Alerts］ノードを展開して、Oracle Service Bus Management Module 用に定義されたアラートのすべてを表示します。
4. 特定のアラートをクリックすると、［ビューア］ペインにその定義が表示されます。

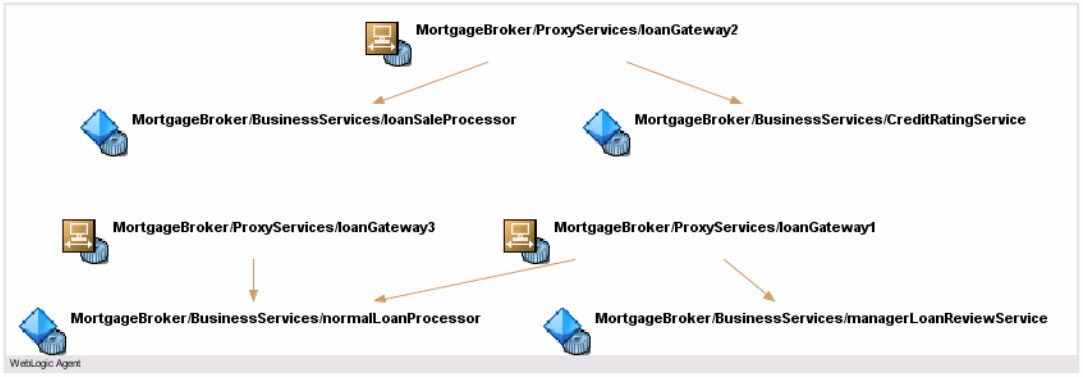
特に、警告しきい値と危険しきい値のデフォルト設定を確認し、必要な場合は値を調整し、通知や修正処置を追加してください。

任意のアラートのデフォルト設定を変更したり、ユーザ独自のカスタム アラートを作成したりできます。

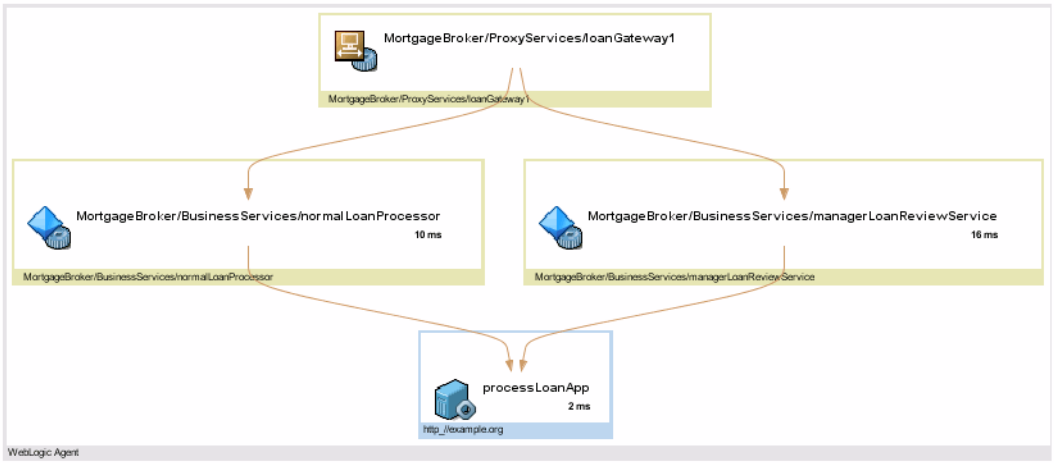
**注**: アラートを作成および変更する方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# Oracle Service Bus の依存関係を表示する

Oracle Service Bus プロキシ サービスおよびビジネス サービスについて依存関係を表示できます。そのためには、Investigator ツリーで［ProxyServices］または［BusinessService］ノードを選択するか、個々のプロキシ サービスまたはビジネス サービスを選択し、［SOA 依存マップ］タブをクリックします。 たとえば、エージェント上のすべてのプロキシ サービスの依存関係の高レベル ビューを表示するには、Investigator で［ProxyServices］ノードを選択し、［SOA 依存マップ］タブをクリックできます。




選択するノードによって、依存マップに表示されるコンテキストが決まります。さらに、表示される詳細情報のコンテキストとレベルをロールアップして折りたたんだり、ロールダウンして展開したりできます。 たとえば、特定のプロキシ サービスの依存関係の高レベル ビューを表示するには、Investigator でプロキシ サービス名を選択し、［SOA 依存マップ］タブをクリックできます。以下の例では、Investigator ツリーで選択したローン要求を処理するためのプロキシ サービスを示しています。依存マップが展開されて、追加の依存関係が表示されています。



必要に応じて、ビジネス プロセスのワークフロー全体が表示されるまでマップに依存関係のレベルを追加したり、マップの特定のノードを拡大表示したりできます。依存マップの操作方法の詳細については、「SOA 依存マップの使用 (P. 81)」を参照してください。

# Oracle Service Bus に関するトランザクションを追跡する

トランザクション追跡は、トランザクションの完了にかかわる特定の手順を収集します。たとえば、それらのトランザクションには、SOAP メッセージで、HTTP または HTTPS ヘッダで、または Java メッセージ サービスの呼び出しで渡されたオペレーションや、メッセージ、データ、またはプロトコル変換を実行する Oracle Service Bus コンポーネントを通じて渡されたオペレーションなどが含まれます。

トランザクションに Oracle Service Bus プロキシ サービスまたはその他の Oracle Service Bus コンポーネントを通じてルーティングされたメッセージが含まれる場合、プロセスにまたがるトランザクション追跡では、実行されたオペレーションに関する情報、および各オペレーションの完了にかかった時間に関する情報を表示できます。 CA APM for SOA および CA APM for Oracle Service Bus が、追跡中のあらゆるノードで有効になっている限り、プラットフォームのあらゆる組み合わせにわたって BusinessTransaction を追跡できます。

## プロセスにまたがるトランザクション追跡の値について

プロセスにまたがるトランザクション追跡は、サービス指向アーキテクチャ内の疎結合サービスによって実行されているオペレーションに関して貴重な情報を提供します。 プロセスにまたがるトランザクション追跡を使用して、以下のことを特定できます。

- メッセージは Oracle Service Bus コンポーネントを通じてどのように変換されルーティングされるか。
- トランザクション中にどのビジネス サービスおよびプロキシ サービスが呼び出されるか。
- トランザクション中に行われる呼び出しのシーケンス。
- 要求または返答の処理が最も遅い箇所。

## サンプル トランザクション追跡を開始して表示する

トランザクション追跡セッションは、以下のいずれかの方法で開始できます。

- SOA 依存マップ内のマップ ノードから直接開始する方法。
- ［Workstation］-［新規トランザクション追跡セッション］をクリックして Workstation から手動で開始する方法。

依存マップからトランザクション追跡を開始する場合は、マップ ノードのタイプに応じてデフォルト フィルタが自動的に設定されます。 新しいトランザクション追跡セッションを手動で開始する場合は、Oracle Service Bus 用に以下のフィルタ タイプの 1 つを選択できます。

- ビジネス サービス
- プロキシ サービス
- ネームスペース
- オペレーション

たとえば、特定の Oracle Service Bus プロキシ サービスについてトランザクションをフィルタするには、proxyservice フィルタを選択し、プロキシ サービス名のすべてまたは一部を入力できます。

トランザクション追跡セッションを開始した後、トランザクション追跡ビューアを使用して、選択したトランザクションの各セグメントに関するサマリおよび詳細情報を表示できます。Oracle Service Bus が含まれるトランザクションに非同期呼び出しが含まれることが多いため、場合によって有用なのは、［シーケンス ビュー］をクリックして、トランザクションの一部として非同期に実行されたプロセスについて、トランザクション ワークフローを表示することです。 シーケンス ビューには、シーケンスを識別できる範囲でプロセスの実行順序が表示されます。 Oracle Service Bus トランザクションについては、シーケンスは必ずしも従来の呼び出し元と呼び出し先の関係を表しません。しかし、あるプロセスが別のプロセスの実行をいつトリガするかを示します。

**注**: トランザクション追跡の詳細については、「SOA 環境でトランザクション追跡を使用する (P. 111)」を参照してください。 エージェントに対応した追跡の構成方法の詳細については、「CA APM Java Agent 実装ガイド」および「CA APM .NET Agent 実装ガイド」を参照してください。 トランザクション追跡ビューと履歴データの使用方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# 第 8 章: TIBCO BusinessWorks を監視する

TIBCO BusinessWorks は複数のインフラストラクチャ コンポーネントおよび機能で構成される SOA プラットフォームです。 CA APM for TIBCO BusinessWorks を使用することによって、プロセス開始元、ジョブ インスタンス、トランスポート プロトコルなど、TIBCO BusinessWorks の処理エンジンの多くの主要なエレメントを監視できます。

このセクションでは、TIBCO BusinessWorks 環境のパフォーマンス、可用性、全体の稼働状況の監視および分析に使用できる、TIBCO BusinessWorks 固有のダッシュボード、メトリック、アラートについて説明します。

このセクションには、以下のトピックが含まれています。



## TIBCO BusinessWorks について

TIBCO BusinessWorks により、企業は、既存のサービスおよびレガシー システムを公開および統合し、新しいサービスを設計およびテストし、疎結合のサービスを組み合わせてアプリケーションにアセンブルできます。

統合プラットフォームとして、BusinessWorks は、複数の分散サブシステムで構成されます。 たとえば、TIBCO BusinessWorks は複数のメッセージ サービスおよび転送プロトコルをサポートしています。

以下のトップレベルのコンポーネントに関するメトリックを使用して、TIBCO BusinessWorks のオペレーションを監視できます。

Activities

アクティビティは、TIBCO ビジネス プロセス定義内のオペレーションを実行する処理の 1 単位です。

TIBCO BusinessWorks 拡張機能を使用して、［Tibco］-［Activities］ノードの下で、ビジネス プロセス内のアクティビティのパフォーマンスおよび全体の稼働状況を監視できます。

Group Actions

グループ アクションは、関連する一連のアクティビティが参加するアクションのタイプを識別します。

TIBCO BusinessWorks 拡張機能を使用して、［Tibco］-［Group Actions］ノードの下で、ビジネス プロセス内のアクティビティのパフォーマンスおよび全体の稼働状況を監視できます。

Hawk Metrics

TIBCO Hawk マイクロ エージェントは、ホスト オペレーティング システムを監視する方法をローカル エージェントに提供します。

TIBCO BusinessWorks 拡張機能を使用して、［Tibco］-［Hawk Metrics］ノードの下で、TIBCO Hawk マイクロ エージェントによって収集されたプロセス エンジンの統計情報を監視できます。

Jobs

ジョブは、メモリで作成され、ジョブ プール内で TIBCO BusinessWorks エンジンによって実行されるプロセス インスタンスの実行を表します。

TIBCO BusinessWorks 拡張機能を使用して、［Tibco］-［Jobs］ノードの下で、プロセス開始元およびジョブ プールのパフォーマンスおよび全体の稼働状況を監視できます。

Processes

プロセスは、特定のタスクを完了するために設計されるビジネス ワークフローです。 TIBCO Designer で定義したプロセスには、親プロセスの一部として実行される、または親プロセスと並列で実行されるサブプロセスが含まれることがあります。 個々のビジネス プロセスは TIBCO BusinessWorks プロセス定義のランタイム インスタンスです。

TIBCO BusinessWorks 拡張機能を使用して、［Tibco］-［Processes］ノードの下で、ビジネス プロセスのパフォーマンスおよび全体の稼働状況を監視できます。

Transports

トランスポートは、メッセージを送信および配信するためのメカニズムを定義します。

TIBCO BusinessWorks 拡張機能を使用して、［Tibco］-［Transports］ノードの下で、HTTP、SOAP、および FTP トランスポート プロトコルのパフォーマンスおよび全体の稼働状況、さらに［Tibco］ -［Transports］-［RV］ノードの下のメトリックを使用して、Rendezvous のパフォーマンスおよび全体の稼働状況を監視できます。

WebServices

WebServices メトリックは、クライアントとサーバのビジネス サービス エンドポイント、および各サービス内の関連するオペレーションを表します。

TIBCO BusinessWorks 拡張機能を使用して、［WebServices］ノードの下で、クライアントとサーバの Web サービス エンドポイントのパフォーマンスおよび全体の稼働状況を監視できます。

# TIBCO BusinessWorks の監視を有効にする方法

TIBCO BusinessWorks の監視を有効にするには、以下の概略手順を実行します。-

1. TIBCO BusinessWorks のサポートされているバージョンがインストールされていることを確認します。

   **注**: TIBCO BusinessWorks の要件の全リストについては、「*Compatibility Guide*」の「SOA Performance Management」を参照してください。

2. エージェントと CA APM for SOA がインストールされて有効になっていることを確認します。

3. TIBCO BusinessWorks を監視するようにエージェント プロファイルを設定 (P. 170) して、エージェントが CA APM for TIBCO BusinessWorks を使用できるようにします。 スタンドアロン エージェント インストーラまたは応答ファイルを使用して、エージェントで CA APM for TIBCO BusinessWorks を有効にした場合は、この手順をスキップできます。
4. エージェントを使用するように TIBCO BusinessWorks を設定します (P. 174)。
5. TIBCO BusinessWorks 用の Enterprise Manager 拡張機能を有効にします (P. 177)。

## エージェントに対して TIBCO BusinessWorks の監視を有効にする

Introscope エージェントのインストール時にアプリケーション サーバとして［**デフォルト**］を選択しているか、エージェントをインストールした後に手動で選択している場合には、TIBCO BusinessWorks の監視を有効にできます。

エージェントをインストールするときに CA APM for TIBCO BusinessWorks を選択した場合は、エージェント プロファイルがデフォルト設定で自動的に構成されます。そのほかの手順は不要です。

エージェントをインストールするときに、CA APM for TIBCO BusinessWorks を選択しなかった場合は、監視を有効にするようにエージェント プロファイルを手動で構成する必要があります。

**SOA Extension for TIBCO BusinessWorks を手動で有効にするには、以下の手順に従います。**

1. エージェントと CA APM for SOA がインストールされて有効になっていることを確認します。
2. CA APM for TIBCO BusinessWorks のディレクトリ（*SOAExtensionForTibcoBW*）が *<Agent_Home>/examples* ディレクトリ内にあることを確認します。
3. *<Agent_Home>/examples/SOAExtensionForTibcoBW/ext* ディレクトリからファイルを *<Agent_Home>/core/ext* ディレクトリにコピーします。
4. *IntroscopeAgent.profile* ファイルをテキスト エディタで開きます。

5. *IntroscopeAgent.profile* ファイル内の *introscope.autoprobe.directivesFile* プロパティに標準またはフル インスツルメンテーションの該当するディレクティブ ファイルを追加します。

   標準またはフル インスツルメンテーションの選択の詳細については、「標準およびフル インスツルメンテーションについて (P. 171)」を参照してください。 ProbeBuilder ディレクティブ ファイルを使用して追跡をカスタマイズすることの詳細については、「TIBCO BusinessWorks 用のディレクティブ ファイルについて (P. 173)」を参照してください。

6. *IntroscopeAgent.profile* ファイルに対する変更を保存して、テキスト エディタを閉じます。

## 標準およびフル インスツルメンテーションについて

*IntroscopeAgent.profile* で *introscope.autoprobe.directivesFile* プロパティを設定するときに、標準（デフォルト）またはフル インスツルメンテーションを選択できます。これにより、エージェントを展開している環境に合わせて、監視するレベル、メトリックの可視性、およびパフォーマンスのオーバーヘッドを必要に応じて制御できます。その後、特定のトレーサ グループに対して追跡をオンまたはオフにすることにより、標準またはフル トグル ファイルを使用して、特定のコンポーネントの監視を微調整できます。

tibcobw-full.pbl

すべての TIBCO BusinessWorks コンポーネントのオペレーションへの深い可視性を提供する、フル インスツルメンテーションを可能にします。

フル インスツルメンテーションにより、詳細なレポートが提供されますが、追加のオーバーヘッドが必要です。 このインスツルメンテーションは通常、実運用環境ではなくテストまたは開発環境にお勧めします。

デフォルトでは、以下のディレクティブ ファイルが tibcobw-full.pbl ファイルの一覧に表示されます。

- tibcobw-toggles-full.pbd
- tibcobw-webservices.pbd
- tibcobw-processes.pbd
- tibcobw-tasks.pbd
- tibcobw-RV.pbd
- tibcobw-activities.pbd
- tibcobw-jobs.pbd
- tibcobw-correlation.pbd
- tibcobw-transports.pbd
- tibcobw-hawk.pbd

tibcobw-typical.pbl

オーバーヘッドが重要である実運用環境での、重要な TIBCO BusinessWorks コンポーネントの標準の監視を定義します。

標準インスツルメンテーションは、簡潔なレポートが提供され、オーバーヘッドも少ないため、実運用環境にお勧めします。

デフォルトでは、以下のディレクティブ ファイルが tibcobw-typical.pbl ファイル リストに表示されます。

- tibcobw-toggles-typical.pbd
- tibcobw-webservices.pbd
- tibcobw-processes.pbd
- tibcobw-tasks.pbd
- tibcobw-RV.pbd
- tibcobw-activities.pbd
- tibcobw-correlation.pbd

tibcobw-toggles-full.pbd

フル インスツルメンテーションを使用するときに、TIBCO BusinessWorks コンポーネントの監視のオンとオフを切り替えます。

このファイルは他のディレクティブ ファイルで有効になる追跡用のディレクティブを提供します。デフォルトでは、このファイルでほとんどのトレーサ グループはオンになっています。

tibcobw-toggles-typical.pbd

標準インスツルメンテーションを使用するときに、TIBCO BusinessWorks コンポーネントの監視のオンとオフを切り替えます。このファイルは他のディレクティブ ファイルで有効になる追跡用のディレクティブを提供します。デフォルトでは、ごく一部のトレーサグループがオンになっています。

ディレクティブ ファイルの詳細については、「TIBCO BusinessWorks 用のディレクティブ ファイルについて (P. 173)」を参照してください。

## TIBCO BusinessWorks 用のディレクティブ ファイルについて

*tibcobw-toggles-full.pbd* および *tibcobw-toggles-typical.pbd* ファイルにより、ProbeBuilder ディレクティブ (*.pbd*) ファイルのセットで提供されるデフォルトの追跡を制御できます。 また、以下のファイル内の設定の修正により提供される追跡を必要に応じて手動でカスタマイズできます。

tibcobw-activities.pbd

ビジネス プロセス内のアクティビティおよびグループ アクティビティ アクションに関する監視ルールを提供します。

tibcobw-correlation.pbd

TIBCO BusinessWorks コンポーネントに対してプロセスにまたがるトランザクション追跡を有効にします。

tibcobw-hawk.pbd

TIBCO BusinessWorks 内の Hawk メトリックに関する監視ルールを提供します。

tibcobw-jobs.pbd

TIBCO BusinessWorks 上で実行されているジョブおよびジョブ プールに関する監視ルールを提供します。

tibcobw-processes.pbd

TIBCO BusinessWorks ビジネス プロセス定義に関する監視ルールを提供します。たとえば、その対象は、プロセス開始元、子サブプロセス、子でないサブプロセスなどです。

tibcobw-RV.pbd

BusinessWorks 内の Rendezvous メッセージ サービスに関する監視ルールを提供します。

tibcobw-tasks.pbd

TIBCO BusinessWorks ビジネス プロセス内のタスクに関する監視ルールを提供します。

tibcobw-transports.pbd

FTP、HTTP、SOAP のような個々のトランスポート プロトコルに関する監視ルールを提供します。

tibcobw-webservices.pbd

BusinessWorks Web サービス エンドポイントに関する監視ルールを提供します。

**注**: 追跡を構成する方法の詳細については、「CA APM Java Agent 実装ガイド」または「CA APM .NET Agent 実装ガイド」を参照してください。

## エージェントを使用するための TIBCO BusinessWorks の設定

インストーラを使用してエージェントを有効にした後、またはインストール後に手動でエージェントを有効にした後に、アプリケーションをインスツルメントするエージェントが実行されるように TIBCO BusinessWorks を設定する必要があります。 設定手順は、組織の環境でどのようにアーカイブを展開するかによって異なります。

TIBCO Designer でプロセス定義を作成するとき、それらを*プロジェクト アーカイブ*（EAR）としてエクスポートします。 その後、各プロジェクト アーカイブは 1 つ以上の*プロセス アーカイブ*（PAR）で構成できます。

標準の TIBCO BusinessWorks の展開で、各プロセス アーカイブはそれぞれ独自の JVM 内で実行されています。 各プロセス アーカイブが組織の環境内のそれぞれ独自の JVM で実行されている場合、以下の手順に従って、エージェントを使用するように TIBCO BusinessWorks を設定できます。

### サービス コンテナなしで TIBCO BusinessWorks を設定する方法

1. TIBCO BusinessWorks の *bwengine.tra* ファイルまたは展開した *<アプリケーション>.tra* ファイルを開いて、*java.extended.properties* プロパティに以下のエントリを追加します。

   ```
   java.extended.properties=-javaagent:<Agent_Home>/Agent.jar -Dcom.wily.introscope.agentProfile=<Agent_Home>/core/config/IntroscopeAgent.profile
   ```

   **注**: UNIX 上の TIBCO BusinessWorks では、java.extended.properties でコロン（:）および等号（=）をエスケープします。

アプリケーションをデプロイしている場合は、デプロイを解除してから、*bwengine.tra* ファイルに変更を加えた後に、それらのアプリケーションを再デプロイします。あるいは、個々のアプリケーションの *.tra* ファイルにこのプロパティを追加できます。 たとえば、<*TIBCO_DOMAIN_HOME*>/<*DOMAIN_NAME*>/application/<*application_name*> ディレクトリ内の <*application_name*>.tra ファイルにこのプロパティを追加できます。

2. SOA Extension for TIBCO BusinessWorks をインストールしたコンピュータ上で、アプリケーションのあらゆる実行中のサービス インスタンスを停止して再起動します。

各プロセス アーカイブをそれぞれ独自の JVM で実行する代わりに、TIBCO BusinessWorks の *サービス コンテナ*をデプロイすることもできます。 サービス コンテナにより、1 つの JVM で複数のプロセス アーカイブを実行できます。 サービス コンテナを使用してプロセス アーカイブを展開する場合は、以下の手順に従って、エージェントを使用するように TIBCO BusinessWorks を設定できます。

**サービス コンテナを使用するときに TIBCO BusinessWorks を設定する方法**

1. <*TIBCO_BW_HOME*>/*bin* ディレクトリ内の *bwcontainer.tra* ファイルを開き、*java.extended.properties* プロパティに以下のエントリを追加します。

```
java.extended.properties=-javaagent:<Agent_Home>/Agent.jar
-Dcom.wily.introscope.agentProfile=<Agent_Home>/core/config/IntroscopeAgent.profile -Dcom.wily.introscope.agent.agentName=<Agent_Name>
```

   **注**: UNIX 上の TIBCO BusinessWorks では、java.extended.properties でコロン（:）および等号（=）をエスケープします。

2. 展開したアプリケーションの *bwengine.tra* または <*アプリケーション*>*.tra* でエージェントが設定されていないことを確認します。
3. サービス コンテナ インスタンスを停止して再起動します。

## エージェント自動名前付け機能を有効にする

ほとんどの TIBCO BusinessWorks ドメインには、複数のビジネス プロセスが複数のプロセス アーカイブ（PAR）で展開されます。 各プロセス アーカイブがそれぞれ独自の JVM インスタンスで実行されるため、各プロセス アーカイブはそれぞれ独自のエージェント インスタンスも起動します。

各アプリケーションの *.tra* ファイル内の *com.wily.introscope.agent.agentName* プロパティを使用して手動で各エージェント インスタンスに名前を付けずに済むように、SOA Extension for TIBCO BusinessWorks では、エージェントへの動的な名前付けを有効にする構成ファイルおよびプロパティが提供されます。この名前付けでは、プレフィックスとして TIBCO BusinessWorks ドメイン名、その後ろにアプリケーション名とアンダースコア（_）が使用されます。たとえば、TIBCO ドメイン名が *caDomain* で、アプリケーション名が *soap_over_http-Process_Archive* であり、エージェントの自動名前付けを有効にしている場合、エージェント名は以下のように動的に生成されます。

```
caDomain_soap_over_http-Process_Archive
```

プロセス アーカイブの名前プロパティが見つからない場合、エージェントは展開名を使用して自動的に名前を付けられます。 例：

```
caDomain_soap_over_http
```

**エージェント自動名前付け機能を設定にするには、以下の手順に従います。**

1. あらゆる実行中の TIBCO BusinessWorks アプリケーションを停止します。
2. *TibcoBWNaming.jar* ファイルが *<Agent_Home>/core/ext* ディレクトリ内にあるかどうかを確認します。

   そのファイルがない場合は、*<Agent_Home>/examples/SOAExtensionForTibcoBW/ext* ディレクトリからそれを *<Agent_Home>/core/ext* ディレクトリにコピーします。
3. *<Agent_Home>/core/config/IntroscopeAgent.profile* ファイルをテキスト エディタで開きます。
4. *introscope.agent.agentAutoNamingEnabled* プロパティを見つけ、true に設定します。 例：

   ```
   introscope.agent.agentAutoNamingEnabled=true
   ```
5. 接続遅延のプロパティを設定して、自動名前付けが試みられている間にエージェントが Enterprise Manager への接続を遅らせる時間を定義します。 たとえば、30 秒間、接続を遅らせるには、以下のようにプロパティを設定します。

   ```
   introscope.agent.agentAutoNamingMaximumConnectionDelayInSeconds=30
   ```

6. 名前変更間隔のプロパティを設定して、自動名前付けが有効になっているときに、エージェントが動的に付けた名前をどれくらい頻繁に確認する必要があるかを定義します。 たとえば、エージェントの自動的に生成された名前を 1 分ごとに確認するには、以下のようにプロパティを設定します。
   ```
   introscope.agent.agentAutoRenamingIntervalInMinutes=1
   ```
7. *IntroscopeAgent.profile* ファイルへの変更を保存します。
8. TIBCO BusinessWorks アプリケーションを再起動します。

個別のプロセス アーカイブで展開する TIBCO BusinessWorks アプリケーションに対してのみ、エージェントの自動名前付けを使用する必要があります。BusinessWorks のサービス コンテナを使用してプロセス アーカイブを展開する場合は、*<Agent_Home>/ext* ディレクトリから *TibcoBWNaming.jar* ファイルを削除する必要があります。

## Enterprise Manager 拡張機能を有効にする

Enterprise Manager をインストールすると、CA APM for TIBCO BusinessWorks のファイルはデフォルトでは *<EM_Home>/examples* ディレクトリにインストールされます。 CA APM for TIBCO BusinessWorks を有効にするには、*<EM_Home>/examples* ディレクトリから Enterprise Manager のファイルを Enterprise Manager のホーム ディレクトリ内の適切な場所にコピーまたは移動する必要があります。

**注**: WSOA Extension for TIBCO BusinessWorks を使用する前に、CA APM for SOA を Enterprise Manager 上で有効にする必要があります。 CA APM for SOA Enterprise Manager 拡張機能を有効にする方法については、「Enterprise Manager 上で拡張機能を有効にする (P. 44)」を参照してください。

**次の手順に従ってください**:

1. CA APM for TIBCO BusinessWorks のディレクトリ（*SOAExtensionForTibcoBW*）が *<EM_Home>/examples* ディレクトリ内にあることを確認して、*<EM_Home>/examples/SOAExtensionForTibcoBW* ディレクトリからファイルを Enterprise Manager ディレクトリ構造内の対応する場所にコピーします。 たとえば、*<EM_Home>/examples/SOAExtensionForTibcoBW/ext* ディレクトリのファイルを *<EM_Home>/ext* ディレクトリにコピーします。
2. Enterprise Manager がクラスタ化環境内のコレクタである場合は、*<EM_Home>/config/modules* ディレクトリから、CA APM for TIBCO BusinessWorks 管理モジュール（*TibcoBWManagementModule.jar*）を削除します。

   この管理モジュールは、MOM コンピュータとして使用している Enterprise Manager の *<EM_Home>/config/modules* ディレクトリにのみコピーする必要があります。 他のすべてのファイルとスクリプトは、コレクタ Enterprise Manager と MOM Enterprise Manager の両方にインストールする必要があります。
3. Workstation を再起動すると、CA APM for TIBCO BusinessWorks 固有のダッシュボードおよび［概要］タブがロードされます。

# ダッシュボードを使用して TIBCO BusinessWorks を監視する

SOA Extension for TIBCO BusinessWorks には、アプリケーション環境全体の稼働状況を監視するために使用できる、複数の事前設定されたダッシュボードが含まれています。 ダッシュボードは、ユーザが問題をすばやく診断して解決できるように、展開されたエージェントからデータを集計してパフォーマンス情報を要約します。

通常、ダッシュボードは以下の機能を備えているため、環境を監視するための起点として使用されます。

- TIBCO BusinessWorks の重要なコンポーネント全体の稼働状況、パフォーマンス、可用性、現在のステータスをひとめで監視する。
- 低いレベルのメトリックによって警告しきい値や危険しきい値を超えたことが示されたときに、実運用アプリケーション環境に発生する可能性がある問題の通知を早めに取得する。
- パフォーマンスの情報にドリルダウンして、どの TIBCO BusinessWorks プロセス、トランスポート プロトコル、または Web サービスが原因で遅延またはエラーが発生しているかを切り分けて特定する。

事前設定された TIBCO BusinessWorks ダッシュボードは、SOA Extension for TIBCO BusinessWorks Management Module（*TibcoBWManagementModule.jar*）の一部として、Enterprise Manager Extension for TIBCO BusinessWorks にパッケージされています。

TIBCO BusinessWorks Management Module では、TIBCO BusinessWorks に対応する以下の事前設定されたダッシュボードが提供されます。

Tibco BW - **概要**

TIBCO BusinessWorks に関する重要なアクティビティのトップレベルの概要。たとえば、すべての Web サービスについて、全体の応答時間、エラー、ストール、SOAP 障害などです。さらに、ビジネス プロセスについて、全体の応答時間、エラー、ストールに関するアラート インジケータなどです。

Tibco BW - **ビジネス プロセス**

すべてのビジネス プロセスの要約されたステータス。たとえば、ビジネス プロセスについて、応答時間、エラー数、ストール数に関するアラート インジケータとグラフなどです。さらに、低速ビジネス プロセスの一覧などです。

Tibco BW - **アクティビティ**

すべてのアクティビティの要約されたステータス。たとえば、アクティビティについて、応答時間、エラー数、ストール数に関するアラート インジケータとグラフなどです。さらに、低速アクティビティの一覧などです。

Tibco BW - ジョブ

ジョブおよびジョブ プールに関する要約されたステータス。さらに、作成されたジョブ、実行中のジョブ、毎時に作成されたジョブ、完了したジョブの数のグラフ。

ダッシュボードには、監視対象のプロセス開始元の数、および監視対象のジョブのフロー制限設定を示すグラフも含まれます。

Tibco BW - トランスポート

すべてのトランスポート タイプに関する要約されたステータス。さらに、各トランスポート タイプのすべてのオペレーションについて平均応答時間のグラフ。たとえば、SOAP を使用する TIBCO BusinessWorks プロセスを監視する場合、ダッシュボードには writeSoapEnvelope や sendMessage のような SOAP オペレーションについて平均応答時間が表示されます。

事前構成済みのダッシュボードは、Workstation コンソールを使用して表示できます。 また、カスタム ダッシュボードを含めるように TIBCO BusinessWorks Management Module を拡張するか、カスタム メトリックまたはアラートを含めるようにデフォルトのダッシュボード定義を修正することもできます。

**注**: ダッシュボードを作成および変更する方法の詳細については、「CA APM 設定および管理ガイド」を参照してください。

**次の手順に従ってください**:

1. Enterprise Manager を起動します（現在実行されていない場合）。
2. Workstation を起動し、SOA Extension for TIBCO がインストールされている Enterprise Manager にログインします。
3. ［Workstation］-［新規コンソール］を選択します。
4. ［ダッシュボード］ドロップダウン リストから TIBCO BusinessWorks ダッシュボードの 1 つを選択します。

   たとえば、［Tibco BW - 概要］を選択して、TIBCO BusinessWorks のフロントエンド、バックエンド、Web サービス、およびビジネス プロセスについて、トップレベルのアラート インジケータを確認します。

5. 別のタブまたはアラートをダブルクリックすると、関連するダッシュボードが開き、詳細情報が表示されます。

   たとえば、［ビジネス プロセス］タブまたはビジネス プロセス応答時間のアラートをダブルクリックして、低速ビジネス プロセスの一覧を確認したり、すべてのビジネス プロセスについて平均応答時間、間隔ごとのエラー数、ストール数に関する追加の詳細情報を確認したりします。

6. ダッシュボードで特定のビジネス プロセス、アクティビティ、またはジョブ名をダブルクリックして、さらなる分析のために Investigator を開きます。たとえば、［Tibco - ビジネス プロセス］ダッシュボードで低速プロセスをダブルクリックして、Investigator を開いて、そのプロセスを表示します。

**注**: Investigator で TIBCO 固有の情報を表示することの詳細については、「TIBCO BusinessWorks に関するメトリックを理解および表示する (P. 181)」を参照してください。 Workstation の起動と使い方、ダッシュボードへのアクセス、または Investigator の起動と操作については、「CA APM Workstation ユーザ ガイド」を参照してください。

# TIBCO BusinessWorks に関するメトリックを理解および表示する

Investigator ツリー内で移動すると、TIBCO BusinessWorks インフラストラクチャのほとんどのコンポーネントについて、CA Introscope® の標準メトリックを表示できます。 標準メトリックのデータは、TIBCO 固有のメトリック カテゴリ別に収集されて集約され、Investigator ツリーでノードおよびサブノードとして表示されます。 表示される具体的なメトリック カテゴリとノード名は、環境内に展開したプロセス、サービス、およびリソースによって異なります。

Investigator ツリーを通じて移動すると、個々のオペレーションに関する低レベルのメトリック、または選択するノードに応じた集約されたメトリックを表示できます。これにより、さまざまな TIBCO BusinessWorks コンポーネントの全体の稼働状況を監視できます。

CA APM for SOA によって提供される標準メトリックおよびそのほかのメトリックの概要については、「使用可能なメトリックについて (P. 64)」を参照してください。 TIBCO BusinessWorks の監視専用のメトリック、または特定の TIBCO BusinessWorks コンポーネントに関するメトリックの追加の情報については、以下のメトリック カテゴリを参照してください。

- Activities に関するメトリック（P. 184）
- Group Actions に関するメトリック（P. 185）
- Hawk に関するメトリック（P. 185）
- Jobs および Job Pools に関するメトリック（P. 194）
- Processes に関するメトリック（P. 196）
- Transports に関するメトリック（P. 200）
- Web サービスに関するメトリック（P. 204）

**Investigator で TIBCO BusinessWorks に関するメトリックを表示および操作するには、以下の手順に従います。**

1. エージェント ノードを展開し、［Tibco］ノードをクリックして、［概要］タブを表示します。このタブに、TIBCO BusinessWorks プロセスのすべてに関するサマリ情報が表示されます。
2. ビジネス プロセスをダブルクリックして、そのプロセスに関するメトリックをグラフィカルな形式で表示します。

   また、インスツルメンテーションのレベルに応じて、選択したプロセスに関連付けられたタスクの一覧を確認することもできます。
3. ［Tibco］ノードを展開して、トップレベルの TIBCO BusinessWorks メトリック カテゴリのサブノードを表示します。

4. サブノードをクリックまたは展開すると、そのメトリック カテゴリに関するサマリ情報を含む［概要］タブが表示されます。 たとえば、［Jobs］ノードをクリックすると、［概要］タブにジョブ プールに関するメトリックが表示されます。

   トランスポートに関する概要情報を参照するには、［Transports］ノードを展開し、［FTP］、［HTTP］、または［SOAP］ノードを選択して、［概要］タブを表示します。 Rendezvous に関するメトリックについては、［Transports］-［RV］を展開し、サブノードを選択して、［概要］タブを表示します。

5. 任意のサブノードを展開するトランスポートに関する概要情報を参照するには、［Transports］ノードを展開し、［FTP］、［HTTP］、または［SOAP］ノードを選択して、［概要］タブを表示します。と、サブプロセス、アクティビティ、メモリ内のジョブのような、個々のコンポーネントに関する追加の詳細情報、およびそれぞれに関連付けられたメトリックが表示されます。

## Activities に関するメトリック

アクティビティは、TIBCO ビジネス プロセス定義内の処理の 1 単位です。アクティビティは多くの場合、外部システムに接続するオペレーションです。しかし、アクティビティは内部処理にも使用できます。TIBCO Designer では、アクティビティはタイプに基づいてグループ化され、パレットのオプションから使用可能です。 たとえば、［General Activities］パレットを使用して、一般的なオペレーション（プロセスの呼び出し、タイマの設定、ログ ファイルや［ファイル］パレットへの書き込みなど）について、ファイル処理アクティビティを定義できます。

Investigator ツリーで、アクティビティは、実行されるアクティビティのタイプに基づいてグループ化されます。ノード名は、アクティビティ タイプに対応する特定のアクションを反映します。 たとえば、Investigator 内の［File］ノードには、FileCreateActivity、FileReadActivity、FileWriteActivity に関するメトリックが含まれることがあります。 ノード名は、一部のアクティビティについては、TIBCO Designer で使用されるパレット名に似ている場合がありますが、必ずしもそうとは限りません。 たとえば、Investigator ツリーで、［CallProcess］、［Error Handling］、または［Flow］ノードの下で、［General Activities］パレットからのアクティビティが一覧表示される場合があります。

TIBCO BusinessWorks アクティビティに使用可能な CA Introscope® の標準メトリックのすべては、［Tibco］-［アクティビティ］ノードの下にあります。

## Group Actions に関するメトリック

グループは、特定のタイプのアクションを連携して達成する一連のアクティビティを作成するために使用します。 たとえば、グループは、条件が true になるまで繰り返される一連のアクティビティや、条件付きで実行される一連のアクティビティ、同じトランザクションに参加し、一括でコミットされてロールバックされる一連のアクティビティを定義するために使用できます。 グループ アクションはアクションのタイプの識別します。たとえば、If Group の一連のアクティビティは if-then 条件を実行します。 Iterative Loop Group の一連のアクティビティはシーケンス内のあらゆる項目を繰り返します。

Group Actions に関するメトリックは、トランザクションをコミットまたはロールバックする、一連の手順を反復するようなアクションを実行するグループのパフォーマンスを監視します。 このメトリックは、グループ内の個々のアクティビティから集約されたデータに基づきません。

TIBCO BusinessWorks グループに使用可能な CA Introscope® の標準メトリックのすべては、［Tibco］-［Group Actions］ノードの下にあります。

## Hawk に関するメトリック

TIBCO Hawk マイクロ エージェントにより、BusinessWorks 処理エンジン、プロセス定義とアクティビティに関するメトリック、メモリ使用率の監視が可能です。

［Tibco］ > ［Hawk Metrics］ノードの下で、以下のメトリック カテゴリを使用して、Hawk マイクロ エージェントに関するメトリックを表示できます。

- ［ExecInfo (P. 186)］メトリックは、システム稼働時間や実行されているスレッドの数のような、プロセス エンジンの実行オペレーションに関する情報を提供します。
- ［MemoryUsage (P. 187)］メトリックは、使用可能なメモリおよび使用済みのメモリに関する情報を提供します。
- ［ProcessDefs (P. 187)］メトリックは、個々のプロセス定義、およびプロセス定義ごとに実行されたアクティビティに関する詳細情報を提供します。
- ［Processes (P. 191)］メトリックは、アクティブなプロセス インスタンスに関する概要情報を提供します。 （これらのメトリックは Hawk バージョン 4.8.1 にのみ有効です。）
- ［ServicesInfo (P. 192)］メトリックは、展開したサービスのすべてに関する情報を提供します。
- ［Status (P. 193)］メトリックは、プロセス エンジンのステータスに関する一般情報を提供します。

## 実行に関するメトリック

以下のメトリックが［Tibco］ - ［Hawk Metrics］ - ［ExecInfo］ノードの下の TIBCO BusinessWorks に使用可能です。

Status

実行エンジンに関する現在のステータス。 メトリック値は、監視対象のサーバ上のエンジンのステータスが ACTIVE、SUSPENDED、in STANDBY、STOPPING のいずれであるかを示します。

Threads

エンジン内のワーカ スレッドの数。

Uptime

プロセス エンジンが起動されてからの経過時間の合計。

Version

監視対象のサーバに関する設定のバージョン情報。

## メモリ使用率に関するメトリック

以下のメトリックが［Tibco］-［Hawk Metrics］-［MemoryUsage］ノードの下の TIBCO BusinessWorks に使用可能です。

FreeBytes

使用可能なバイトの合計。

PercentUsed

現在使用されているバイトの合計パーセンテージ。

TotalBytes

エンジン プロセスに割り当てられたバイトの合計数。

UsedBytes

現在使用されているバイトの合計数。

## プロセス定義およびアクティビティに関するメトリック

以下のメトリックが［Tibco］-［Hawk Metrics］> 個々のプロセス サブノードの［ProcessDefs］ノードの下の TIBCO BusinessWorks に使用可能です。

Aborted

選択したプロセス定義が異常終了した回数。

AverageElapsed

選択したプロセス定義を使用して完了されたすべてのプロセスについて平均経過時間（ミリ秒単位）。

AverageExecution

選択したプロセス定義を使用して完了されたすべてのプロセスについて平均実行時間（ミリ秒単位）。

Checkpointed

選択したプロセス定義がチェックポイントされた回数。

Completed

選択したプロセス定義が完了した回数。

CountSinceReset

前回のリセット以降に完了したプロセスの数。

Created

選択したプロセス定義について作成されたプロセスの数。

MaxElapsed

選択したプロセス定義を使用して完了されたすべてのプロセスについて最大経過時間（ミリ秒単位）。

MaxExecution

選択したプロセス定義を使用して完了されたすべてのプロセスについて最大実行時間（ミリ秒単位）。

MinElapsed

選択したプロセス定義を使用して完了されたすべてのプロセスについて最小経過時間（ミリ秒単位）。

MinExecution

選択したプロセス定義を使用して完了されたすべてのプロセスについて最小実行時間（ミリ秒単位）。

MostRecentElapsedTime

最新の経過時間（ミリ秒単位）。

MostRecentExecutionTime

最新の実行時間（ミリ秒単位）。

Name

プロセス定義の名前。

Queued

選択したプロセス定義がキューに格納された回数。

Starter

選択したプロセス定義のプロセス開始元アクティビティの名前。

Suspended

選択したプロセス定義を使用してプロセスが一時停止された回数。

Swapped

選択したプロセス定義がスワップされた回数。

TimeSinceLastUpdate

最新の値が更新されてからのミリ秒数。

TotalElapsed

選択したプロセス定義を使用して完了されたすべてのプロセスについて合計経過時間（ミリ秒単位）。

TotalExecution

選択したプロセス定義を使用して完了されたすべてのプロセスについて合計実行時間（ミリ秒単位）。

TIBCO BusinessWorks に使用可能な以下のメトリックは［Tibco］ - ［Hawk Metrics］ - ［ProcessDefs］ > 個々のアクティビティ名の［*process_name*］ノードの下にあります。

ActivityClass

選択したアクティビティを実装するクラスの名前。

CalledProcessDefs

選択したアクティビティによって呼び出されたプロセス定義の名前。

ElapsedTime

選択したアクティビティのすべての呼び出しに使用される経過時間の合計（ミリ秒単位）。たとえば、スリープ、呼び出しプロセス、待機アクティビティの待機時間の合計などです。

ErrorCount

選択したアクティビティがエラーを返した回数。

ExecutionCount

選択したアクティビティが TIBCO BusinessWorks エンジン インスタンスによって実行された回数。

ExecutionCountSinceReset

選択したアクティビティが前回のリセット以降に実行された回数。

ExecutionTime

選択したアクティビティのすべての呼び出しに使用される実行時間の合計（ミリ秒単位）。

このメトリックには、スリープ、呼び出しプロセス、および待機アクティビティの待機時間は含まれません。

LastReturnCode

選択したアクティビティの最新の実行によって返されたステータス。このメトリックの有効なリターン コードは OK、ERROR、DEAD、または DEBUG です。

MaxElapsedTime

選択したアクティビティについて最大経過時間（ミリ秒単位）。

MaxExecutionTime

選択したアクティビティについて最大実行時間（ミリ秒単位）。

MinElapsedTime

選択したアクティビティについて最小経過時間（ミリ秒単位）。

MinExecutionTime

選択したアクティビティについて最小実行時間（ミリ秒単位）。

MostRecentElapsedTime

最新の経過時間（ミリ秒単位）。

MostRecentExecutionTime

最新の実行時間（ミリ秒単位）。

Name

アクティビティの名前。

TimeSinceLastUpdate

最新の値が更新されてからのミリ秒数。

Tracing

選択したアクティビティに対して追跡が有効になっているかどうかを示します。

## プロセスに関するメトリック

**Hawk バージョン 4.8.1 で有効**

Hawk バージョン 4.8.1 を使用している場合、以下のメトリックが［Tibco］-［Hawk Metrics］-［Processes］の下の TIBCO BusinessWorks に使用可能です。

CurrentActivityName

選択したプロセスによって現在実行されているアクティビティの名前。

CustomId

存在する場合は、選択したプロセスのカスタム識別子。

Duration

選択したプロセスが開始してからの実際の経過時間（ミリ秒単位）。

MainProcessName

メイン プロセス定義の名前。

Name

選択したプロセスの名前。

StartTime

プロセスが開始した時間（ミリ秒単位）。

StarterName

選択したプロセス インスタンスを開始したプロセス開始元の名前。

Status

選択したプロセスについて現在のステータス。 たとえば、このメトリックは、ジョブが現在アクティブかどうかを示します。

SubProcessName

該当する場合、選択したプロセスについてサブプロセス定義の名前。

TrackingId

存在する場合は、選択したプロセスの追跡識別子。

## サービスに関するメトリック

以下のメトリックが［Tibco］-［Hawk Metrics］-［ServicesInfo］ノードの下の TIBCO BusinessWorks に使用可能です。

Description

サービスの説明。

EndpointURL

サービスのエンドポイント URL アドレス。

Name

サービスの名前。

PortName

サービスについてポートの名前。

PortTypeName

サービスについてポート タイプの説明。

TransportType

サービスについてトランスポート タイプの説明。

WSDL_Namespace

サービスについて具体的な Web Services Description Language（WSDL）ネームスペース。

WSDL_URL

サービスについて Web Services Description Language（WSDL）URL アドレス。

bindingName

サービスについてバインド名。

## Status メトリック

以下のメトリックが［Tibco］ - ［Hawk Metrics］ - ［Status］ノードの下の TIBCO BusinessWorks に使用可能です。

Adapter Name

アプリケーションの名前。

Host

TIBCO BusinessWorks エンジン プロセスが実行されているホスト マシンの名前。

Instance ID

エンジン インスタンスのインスタンス識別子。

New Errors

このメトリックに対する前回の呼び出しからの新しいエラーの数。

Process ID

TIBCO BusinessWorks エンジン プロセスのプロセス識別子。

Total Errors

起動後のエラーの合計数。

Uptime

起動してからの秒数。

## Hawk に関するメトリックの有効化

Hawk に関するメトリックの収集は、パフォーマンス集約型のオプションであり、デフォルトでは有効になっていません。 このオプションを有効にする前にパフォーマンスのオーバーヘッドを検討します。 Hawk マイクロ エージェントを使用して、Hawk に関するメトリックの収集を有効にします。

**次の手順に従ってください**:

1. *IntroscopeAgent.profile* ファイルをテキスト エディタで開きます。
2. フル インスツルメンテーション監視レベルを使用するように *introscope.autoprobe.directivesFile* プロパティを設定します。 例：
   `introscope.autoprobe.directivesFile=tibcobw-full.pbl`

3. コメントを外し、以下の Hawk プロパティを true に設定します。

   ```
   com.wily.soaextension.tibcobw.hawkmonitor.enabled=true
   ```

   このプロパティを true に設定すると、TIBCO Hawk の監視が有効になります。

4. 以下のプロパティのコメント化を解除することにより、Hawk マイクロエージェントからメトリックを収集するためのポーリング間隔を制御します。 例：

   ```
   com.wily.soaextension.tibcobw.hawkmonitor.frequency
   ```

   **注**:デフォルトでは、メトリックを収集するポーリング間隔は 30 秒（30,000 ミリ秒）です。

5. アプリケーション サーバを再起動します。

   変更が反映されます。

## Jobs および Job Pools に関するメトリック

プロセス定義に対応するプロセス開始元は実行を開始するためのデータを受信すると、ジョブを作成します。 そのジョブは、プロセス インスタンスでのタスクの実行を表します。 ほとんどの場合、ジョブはメモリ内に作成され、ジョブ プール内で TIBCO BusinessWorks エンジンによって実行されます。 次のジョブに処理を渡す前に、ジョブ プール内の各ジョブは、決められた数のタスクを実行できます。 ジョブ プール内の実行できるタスクの最大数は StepCount プロパティによって制御されます。 BusinessWorks エンジンによって同時に実行できるジョブの最大数は、ThreadCount プロパティによって定義されます。

Jobs および Job Pools カテゴリのメトリックは、作成され、メモリ内に保持され、各ジョブ プールで実行されたジョブの数を監視するのに役立ちます。ジョブおよびジョブ プールに関するメトリックは、30 秒ごとにエージェントによってリフレッシュされます。

以下のメトリックが［Tibco］ - ［Jobs］ノードの下の TIBCO BusinessWorks のジョブおよびジョブ プールに使用可能です。

Active Jobs in Job Pool

選択したジョブ プール内のアクティブな非ページ プロセスの合計数。

Job Error Count in Job Pool

選択したジョブ プール内のジョブのすべてについて起動からのエラーの合計数。

Running Jobs in Job Pool

選択したジョブ プール内の実行中のプロセスおよびキュー内のプロセスの合計数。

Thread Count

BusinessWorks エンジンについて設定されたワーカ スレッドの最大数。

Completed Jobs

起動から実行が完了されたプロセスの数。

Created Jobs

起動から BusinessWorks エンジン内で作成されたプロセスの数。

Flow Limit

TIBCO BusinessWorks の Flow Limit 設定の値。

Jobs Created Per Hour

TIBCO BusinessWorks エンジン内で 1 時間あたりに作成されたプロセスの数。

Process Starter State

ジョブについてプロセス開始元の現在のステータス。

Running Jobs

TIBCO BusinessWorks エンジン上で実行されているビジネス プロセスの数。

任意の個々のジョブ プール名を展開して、そのプロセス インスタンス用に作成されたジョブのサブノードを表示できます。

## ジョブおよびジョブ プールに関するメトリックを有効または無効にする

フル インスツルメンテーションと 30 秒のデフォルトのポーリング間隔を使用するとき、ジョブおよびジョブ プールの監視はデフォルトでオンになっています。 以下のプロパティを使用して、ジョブおよびジョブ プールの監視を有効または無効にして、ポーリング間隔の頻度を制御できます。

```
com.wily.soaextension.tibcobw.jobmonitor.enabled
com.wily.soaextension.tibcobw.jobmonitor.frequency
```

たとえば、標準インスツルメンテーションを使用する場合は、com.wily.soaextension.tibcobw.jobmonitor.enabled プロパティを *true* に設定して監視を有効にします。フル インスツルメンテーションを使用する場合は、*false* に設定して監視を無効にします。

```
com.wily.soaextension.tibcobw.jobmonitor.enabled=true
```

その後、*com.wily.soaextension.tibcobw.jobmonitor.frequency* プロパティを使用して、メトリックを収集するポーリング間隔を制御できます。 例：

```
com.wily.soaextension.tibcobw.jobmonitor.frequency=30000
```

標準インスツルメンテーションを使用しており、com.wily.soaextension.tibcobw.jobmonitor.enabled を *true* に設定している場合は、ジョブおよびジョブ プールの監視を有効にするために、*tibcobw-toggles-typical.pbd* ファイル内の以下の行のコメント アウトを解除することも必要です。

```
#TurnOn: JobTracing
#TurnOn: JobPoolTracing
```

たとえば、標準インスツルメンテーションを使用するときは、com.wily.soaextension.tibcobw.jobmonitor.enabled プロパティを *true* に設定し、ジョブおよびジョブ プールの監視トレーサを有効にするように *tibcobw-toggles-typical.pbd* ファイルを以下のように修正します。

```
TurnOn: JobTracing
TurnOn: JobPoolTracing
```

ジョブ監視のプロパティのいずれかを変更した場合は、変更を有効にするためにアプリケーション サーバを再起動する必要があります。

## Processes に関するメトリック

TIBCO BusinessWorks の主なコンポーネントはそのプロセス エンジンです。プロセス エンジンには、ビジネス プロセスを設計、展開、管理するためのツールが含まれています。 TIBCO BusinessWorks で、*ビジネス プロセス* は TIBCO プロセス定義のランタイム インスタンスです。 プロセス定義にはアクティビティ、サブプロセス、タスクを含めることができ、ビジネス プロセスのこれらのエレメントの監視により、プロセス エンジンを監視できます。

ビジネス プロセス、子サブプロセス、タスクに使用可能な CA Introscope® の標準メトリックのすべては、［Tibco］-［プロセス］ノードの下にあります。子でないサブプロセスには、［Average Response Time （平均応答時間）］、［Concurrent Invocations （同時進行中の呼び出し）］、［Errors Per Interval （間隔ごとのエラー数）］、［Responses Per Interval （間隔ごとの応答数）］のみが使用可能です。

標準メトリックに加えて、以下のメトリックがプロセスおよび子でないサブプロセスに使用可能です。

Running Average Response Time

ビジネス プロセスが TIBCO BusinessWorks エンジン上でのアクティブな実行に費やす平均時間（ミリ秒単位）。入力待ちに費やされる時間は含みません。

Running Concurrent Invocations

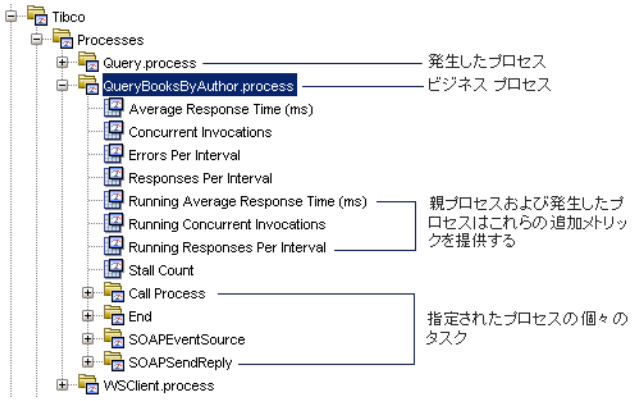
TIBCO BusinessWorks エンジン上でアクティブに実行されているビジネス プロセスの数。

Running Responses Per Interval

間隔ごとに TIBCO BusinessWorks エンジン上で正常に実行されているビジネス プロセスの数。

このメトリックは、ビジネス プロセスが間隔内に完了されることを示しません。 たとえば、このメトリックには、実行を一時停止されたビジネス プロセスが含まれることがあります。

たとえば、［Tibco］ - ［Processes］を展開すると、TIBCO BusinessWorks について TIBCO Designer で定義したビジネス プロセス名が表示されます。［Processes］ノードには、以下のように、親ビジネス プロセスと同じレベルにある子サブプロセスと子でないサブプロセス、および個々のタスクのノードが含まれる場合があります。



また、TIBCO BusinessWorks プロセスについて、依存関係および偏差メトリックを収集することもできます。 標準の依存関係および偏差のメトリックについては、「Investigator を使用して SOA パフォーマンス メトリックを表示する (P. 64)」を参照してください。

## ビジネス プロセスの完全およびアクティブな実行について

ビジネス プロセス定義には、ユーザ入力を待つためのタイマや実行の一時停止が含まれることがあるため、[Processes]メトリック カテゴリでは、次のことを測定対象とするメトリックが提供されます。

- Business process completion： これらのメトリックは、ビジネス プロセスが一時停止されているかアクティブに実行されているかどうかにかかわらず、ビジネス プロセスのエンド ツー エンドの処理状況を提供します。
- Business process active execution： これらのメトリックは、ビジネス プロセスが TIBCO BusinessWorks エンジン上でアクティブに実行されているときのみ、ビジネス プロセスの処理状況を提供します。

ビジネス プロセスに関する［**Average Response Time （平均応答時間）**］メトリックは、プロセスの開始から終了までの平均応答時間を測定します。これには、ユーザからの入力または応答を待ってプロセスが一時停止されている時間も含まれます。 たとえば、ビジネス プロセスが開始され、BusinessWorks エンジン上でのアクティブな実行に 15 ミリ秒かかり、ユーザ入力を 30 ミリ秒待ち、そのオペレーションの完了までの実行に 10 ミリ秒かかったとします。このプロセスの開始から終了までの合計応答時間は、15 ミリ秒、30 ミリ秒、10 ミリ秒の合計で、55 ミリ秒となります。 例：

Average Response Time （平均応答時間（ミリ秒）） = 実行時間 + 一時停止時間

［Average Response Time （平均応答時間）］メトリックには、処理が一時停止される時間が含まれるため、その値は BusinessWorks エンジン自体のパフォーマンスを反映しません。 別の［**平均応答時間の実行**］メトリックは、ビジネス プロセスが BusinessWorks エンジンでのアクティブな実行に費やす時間を追跡します。

［平均応答時間の実行］メトリックは、ビジネス プロセスについてアクティブな実行時間のみを測定します。 これには、入力待ちに費やされた時間は含まれません。 たとえば、ビジネス プロセスが開始され、BusinessWorks エンジン上での実行に 15 ミリ秒かかり、ユーザ入力を待って実行が一時停止され、その後、BusinessWorks エンジン上での実行の完了までに 10 ミリ秒かかったとします。この場合、平均応答時間の実行は、最初の実行の 15 ミリ秒、2 番目の実行の 10 ミリ秒の合計になります。 これには、ユーザ入力を待って一時停止された時間は含まれません。 例：

平均応答時間の実行（ms） = 実行時間の合計 - 一時停止時間

同様に、標準の［**Concurrent Invocations （同時進行中の呼び出し）**］および［**Responses Per Interval （間隔ごとの応答数）**］メトリックには、アクティブに実行されているか一時停止されているかどうかにかかわらず、すべてのプロセスが対象になります。［**Concurrent Invocations （同時進行中の呼び出し） 実行**］および［**Responses Per Interval （間隔ごとの応答数）**］メトリックは、BusinessWorks エンジン上で発生する処理アクティビティのみを反映します。

［平均応答時間の実行］、［Concurrent Invocations （同時進行中の呼び出し） の実行］、および［Responses Per Interval （間隔ごとの応答数）の実行］メトリックはすべて、ビジネス プロセスおよび子プロセスにのみ適用可能であり、子でないサブプロセスまたは個々のタスクについては提供されません。

## ビジネス プロセスに関するストール数について

ビジネス プロセスについて、［**Stall Count（ストール数）**］メトリックは、BusinessWorks エンジン上でアクティブに実行されているプロセスのみが対象になります。 ユーザ入力を待って一時停止されているプロセスはストール数に含まれません。

### サブプロセスでの 1 間隔あたりのエラー数について

［**Errors Per Interval （間隔ごとのエラー数）**］メトリックは、子でないサブプロセスのみを対象とし、それらの特定のサブプロセスの実行中にエラーが発生した場合に表示されます。 子でないサブプロセスの実行中にエラーが発生した場合、サブプロセスおよびその親プロセスの両方に関する Errors Per Interval （間隔ごとのエラー数） の数字は 1 ずつ増えます。

親プロセスおよび子サブプロセスに関する［Errors Per Interval （間隔ごとのエラー数）］メトリックはデフォルトで常に表示されています。 子サブプロセスの実行中にエラーが発生した場合、エラーはそのサブプロセスに関する［Errors Per Interval （間隔ごとのエラー数）］メトリックに記録されます。ただし、子サブプロセスのエラーは、親プロセスに関する Errors Per Interval （間隔ごとのエラー数） の数字に加算されません。

## Transports に関するメトリック

トランスポートは、メッセージを送信および配信するためのメカニズムを定義します。 TIBCO BusinessWorks について、一連のトランスポート プロトコルに関するメトリックを収集できます。たとえば、その対象は、TIBCO Rendezvous（TIBCO BusinessWorks でのメッセージのルーティングと処理に使用できるエンタープライズ メッセージング システム）、および HTTP、SOAP、JMS、FTP、XML のような標準のトランスポート プロトコルです。

TIBCO BusinessWorks 拡張機能を使用して、［Tibco］ - ［Transports］ノードの下で、HTTP、SOAP、および FTP トランスポート プロトコルのパフォーマンスおよび全体の稼働状況、さらに［Tibco］-［Transports］-［RV］ノードの下のメトリックを使用して、Rendezvous のパフォーマンスおよび全体の稼働状況を監視できます。

また、標準の *j2ee.pbd* ファイルで JMS と XML の追跡を有効にすることにより、JMS と XML のパフォーマンスおよび全体の稼働状況を監視できます。*j2ee.pbd* ファイルは、エージェントと共にデフォルトで提供されます。しかし、TIBCO BusinessWorks 拡張機能と共に使用することにより、追加の監視が可能になります。

オペレーションに使用可能な CA Introscope® の標準メトリックのすべては、SOAP や HTTP のような、使用している特定のトランスポート プロトコルの［Transports］ノードの下にあります。 その後、トランスポート プロトコル ノードを展開して、ドライバ、ハンドラ、リクエスタ クラスのような、トランスポート クラスを表示し、次に、サブノードを展開して、*sendMessage* や *writeSoapEnvelope* のような個々のオペレーションに関するメトリックにドリル ダウンできます。

エージェントの *j2ee.pbd* ファイルを使用して JMS または XML を監視する場合、［Tibco］ - ［Transports］ノードではなくエージェント ノードの下にある、JMS および XML トランスポート プロトコル ノードを表示できます。

## Rendezvous（RV）に関するメトリック

Rendezvous は、分散アプリケーションがネットワークを介してデータを交換できるエンタープライズ メッセージング システムです。 統合プラットフォームとして、TIBCO BusinessWorks は、サポートされているメッセージ サービスの 1 つとして Rendezvous を使用して、送信および受信された大容量のメッセージの処理を信頼性の高い方法で行います。

Rendezvous に関するメトリックは、Rendezvous のメッセージ配信、キュー、トランスポートに関する詳細情報を提供します。 TIBCO BusinessWorks 内で、［Tibco］ - ［Transports］ - ［RV］ノードの下の以下のメトリック カテゴリのメトリックを使用して、Rendezvous を監視できます。

Message Listener

Message Listener は、Inbound Rendezvous メッセージをリッスンするオブジェクトです。新しいインバウンド メッセージが届くと、リスナはメッセージの処理のために Message Processor オブジェクトを呼び出します。

Message Processor

Message Processor は、Message Listener によって検出されたメッセージを処理するコールバック メソッドです。

Queue Groups

Queue Groups を使用して、イベントがディスパッチされる順序に対して詳細な制御が可能です。 グループ内の各キューの相対的な優先度によって、ディスパッチ プロセスがそのイベントをディスパッチするためにキューを呼び出す順序が決まります。

このカテゴリのメトリックは、ディスパッチ方法について応答時間、スループット、エラーを監視します。

Queues

Queues は、インバウンド メッセージやタイマ プロセスのような非同期イベントを、メッセージ プロセッサによって処理できるまで、受信された順に格納するために使用されます。 イベントが発生すると、イベントはイベント キューに格納され、ディスパッチ ルーチンによって処理のために渡されるのを待ちます。

このカテゴリのメトリックは、ディスパッチ方法について応答時間、スループット、エラーを監視します。

Transport

トランスポートは、メッセージを送信および配信するためのメカニズムを定義します。 Rendezvous with TIBCO BusinessWorks については、［Reliable Transport］（標準の Rendezvous Transport）および［Certified Transport］（RVCM）トランスポート タイプのメトリックを収集できます。

## Message Listeners、Message Processors、および Transport メトリック

Rendezvous に使用可能な CA Introscope® の標準メトリックのすべては［Tibco］-［Transports］の下の［Message Listeners］、［Message Processors］、［Reliable］、および［Certified］トランスポート プロトコルの［RV］サブノードの下にあります。

## Queue Groups メトリック

以下のディスパッチ関連のメトリックは［Tibco］ -［Transports］ -［RV］の下の、個々のキュー グループに関連付けられたディスパッチ方法の［Queue Groups］ノードの下にあります。

- Average Response Time (ms)
- Errors Per Interval
- Responses Per Interval

## Queues メトリック

以下のディスパッチ関連のメトリックは、個々のキューに関連付けられたディスパッチ方法の［Queues］ノードの下にあります。

- Average Response Time (ms)
- Errors Per Interval
- Responses Per Interval

## Rendezvous メトリックを有効または無効にする

標準またはフル インスツルメンテーションと 30 秒のデフォルトのポーリング間隔を使用するとき、Rendezvous の監視はデフォルトでオンになっています。Rendezvous の監視を無効にする場合は、*tibcobw-toggles-typical.pbd* または *tibcobw-toggles-full.pbd* ファイルのいずれかで以下の行をコメントアウトします。

```
TurnOn: TibrvTransport
TurnOn: TibrvDispatchable
TurnOn: TibcoRVMessageProcessorTracing
TurnOn: TibcoRVMessageListenerTracing
```

たとえば、標準インスツルメンテーションを使用する場合は、テキスト エディタで *tibcobw-toggles-typical.pbd* ファイルを開き、以下のように行を修正することにより、追跡を無効にします。

```
#TurnOn: TibrvTransport
#TurnOn: TibrvDispatchable
#TurnOn: TibcoRVMessageProcessorTracing
#TurnOn: TibcoRVMessageListenerTracing
```

## Web サービスに関するメトリック

クライアント Web サービスとサーバ Web サービスのネームスペースのエンドポイントに使用可能な CA Introscope® の標準メトリックのすべては、［WebServices］ノードの下にあります。 さらに、SOA プラットフォーム上の Web サービスを監視するすべての拡張用の標準メトリックとして、間隔ごとの SOAP 障害数があります。Errors Per Interval（間隔ごとのエラー数） と［SOAP Faults Per Interval］との違いについては、「［エラー］タブで SOAP 障害およびエラーに関するメトリックを表示する」を参照してください。

また、TIBCO BusinessWorks の Web サービス オペレーションについて、依存関係および偏差メトリックを収集できます。 標準の依存関係および偏差のメトリックについては、「Investigator を使用して SOA パフォーマンス メトリックを表示する (P. 64)」を参照してください。

**Investigator で WebServices に関するメトリックを表示および操作するには、以下の手順に従います。**

1. エージェント ノードを展開し、［WebServices］を展開して、Web サービス エンドポイントの［Client］および［Server］ノードを表示します。
2. ［Client］または［Server］ノードを展開して、メトリックを表示する個々の Web サービス エンドポイントを表示します。
3. 特定の Web サービス エンドポイントの <*ネームスペース*> または <*サービス名*> を展開して、オペレーションを表示するか、その Web サービスに関するメトリックを選択します。
4. Web サービス内の特定のオペレーションを展開して、そのオペレーションに関するメトリックを表示します。

## サーバ エンドポイントに関する Average Response Time（平均応答時間）について

サーバ側オペレーションについて、［Average Response Time （平均応答時間）］メトリックは、TIBCO BusinessWorks が受信 SOAP メッセージの受信および解析に要する時間を表します。 ほとんどの場合、その処理はより複雑なトランザクションの一部にすぎません。たとえば、BusinessWorks の標準のトランザクションには、Web サービスのサーバ側で以下の段階があります。

- 受信 SOAP メッセージを受信および解析することに費やす時間
- BusinessWorks エンジンが、Web サービスによって公開されるプロセス定義を実行すること、および返信を送信する準備をすることに費やす時間
- 返信を送信することに費やす時間

トランザクションの各段階が別のスレッド上で実行できるため、［Average Response Time（平均応答時間）］メトリックは、各オペレーションのトランザクションの最初の部分のみを測定して、Web サービス エンドポイントに関するデータを集約できます。 そのメトリックは、その他のスレッドで発生する処理時間を提供できないため、Web サービスのトランザクション全体の応答時間の合計を表しません。

## クライアントとサーバの SOAP 障害に関するメトリックについて

サーバ側オペレーション（［WebServices］ - ［Server］ - ［*ネームスペース*］）について、［SOAP Faults Per Interval］メトリックは、要求に応じて Web サービス クライアントに BusinessWorks によって送信された、SOAP 障害の数を表します。

クライアント側オペレーション（［WebServices］ - ［Client］ - ［*ネームスペース*］）について、［SOAP Faults Per IntervalStall Count］メトリックは、クライアント上で SOAPSendReceiveActivity によって受信された、SOAP 障害の数を表します。

# デフォルト TIBCO BusinessWorks メトリック グループを表示する

SOA Extension for TIBCO BusinessWorks には、デフォルトのダッシュボードおよびアラートの定義に使用されるデフォルト メトリック グループが含まれています。これらのデフォルト メトリック グループをカスタム ダッシュボードやカスタム アラートで使用することもできます。

デフォルト メトリック グループは、SOA Extension for TIBCO BusinessWorks Management Module（*TibcoBWManagementModule.jar*）の一部として、Enterprise Manager Extension for TIBCO BusinessWorks にパッケージされています。

**次の手順に従ってください**:

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］-［Management Modules］-［Introscope SOA Extension for TibcoBW <*version*> (*Super Domain*)］の順に展開します。
3. ［Metric Groupings］ノードを展開して、TIBCO Management Module 用に定義されたメトリック グループのすべてを表示します。
4. 特定のメトリック グループをクリックすると、［ビューア］ペインにその定義が表示されます。

   任意のメトリック グループのデフォルト設定を変更したり、ユーザ独自のカスタム メトリック グループを作成したりできます。

   **注**: メトリック グループを作成および変更する方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# TIBCO BusinessWorks のデフォルト アラートの表示

SOA Extension for TIBCO BusinessWorks には、事前設定されたダッシュボードで使用されるデフォルト アラート定義が含まれています。 これらのデフォルト アラートをカスタム ダッシュボードで使用することもできます。ほとんどのデフォルト アラートは、デフォルトの警告しきい値と危険しきい値を使用して、しきい値を超えるか重要度が高くなった場合にコンソールに通知を送信するように事前設定されています。

デフォルト アラート定義は、SOA Extension for TIBCO BusinessWorks Management Module（*TibcoBWManagementModule.jar*）の一部として、Enterprise Manager Extension for TIBCO BusinessWorks にパッケージされています。

**TIBCO BusinessWorks エージェントに関するデフォルト アラート定義を表示するには、以下の手順に従います。**

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］ - ［Management Modules］ - ［Introscope SOA Extension for TibcoBW <*バージョン*> (*Super Domain*)］を展開します。
3. ［Alerts］ノードを展開して、TIBCO BusinessWorks Management Module 用に定義されたアラートのすべてを表示します。
4. 特定のアラートをクリックすると、［ビューア］ペインにその定義が表示されます。

特に、警告しきい値と危険しきい値のデフォルト設定を確認し、必要な場合は値を調整し、通知や修正処置を追加してください。

任意のアラートのデフォルト設定を変更したり、ユーザ独自のカスタム アラートを作成したりできます。

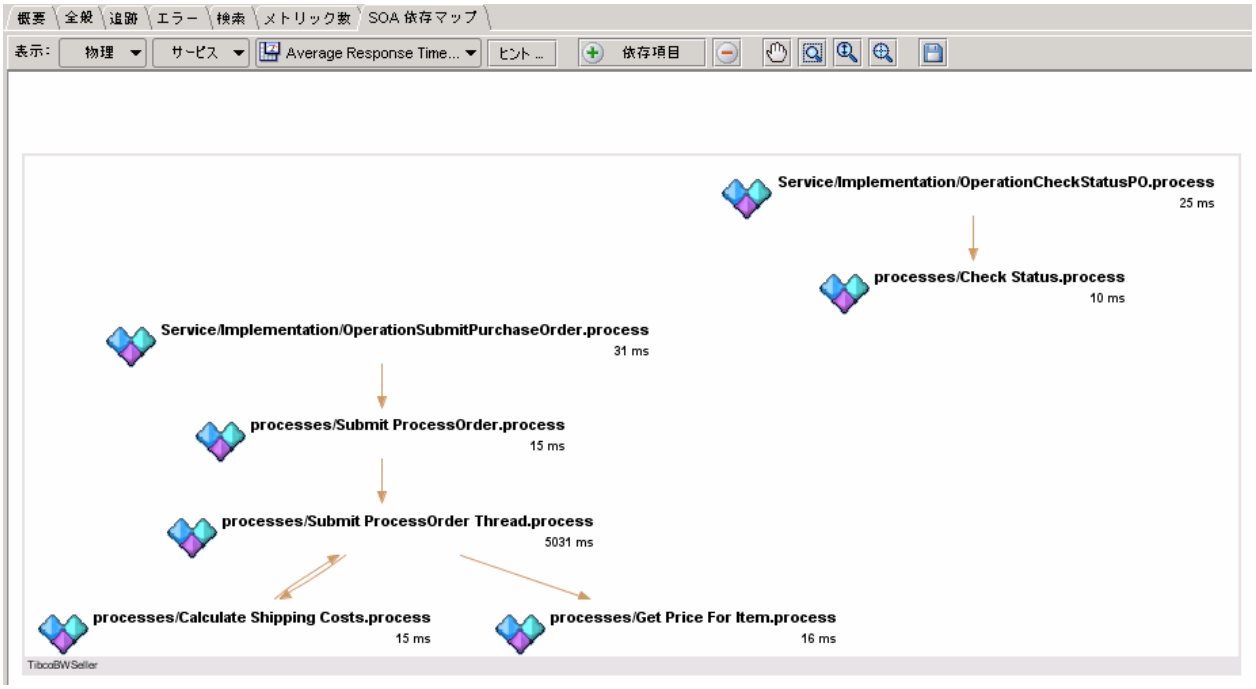
**注**: アラートを作成および変更する方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# TIBCO BusinessWorks 依存関係を表示する

TIBCO BusinessWorks Web サービス、サービス オペレーション、およびプロセス定義について依存関係を表示できます。そのためには、Investigator ツリーで［TIBCO BusinessWorks Processes］または［WebServices］ノードを選択し、［SOA 依存マップ］タブをクリックします。

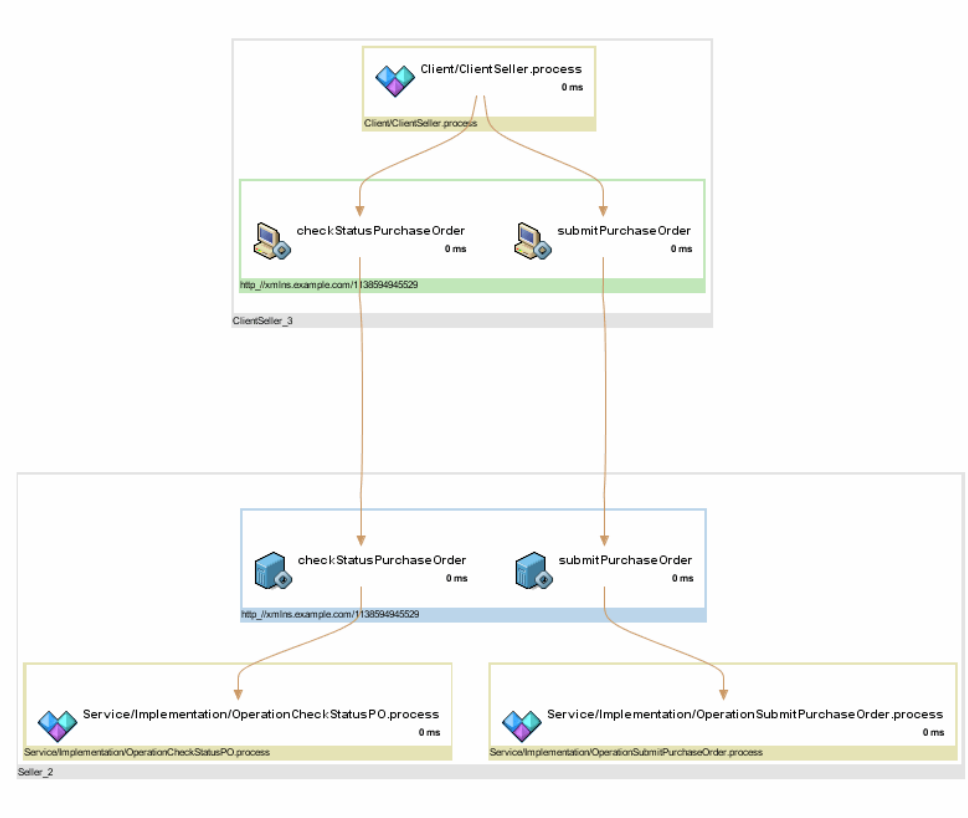
選択するノードによって、依存マップに表示されるコンテキストが決まります。さらに、表示される詳細情報のコンテキストとレベルをロールアップして折りたたんだり、ロールダウンして展開したりできます。

たとえば、エージェント上のすべてのビジネス プロセスの依存関係の高レベル ビューを表示するには、Investigator で［Processes］ノードを選択し、［SOA 依存マップ］タブをクリックできます。




エージェント上で特定のビジネス プロセスの依存関係の高レベル ビューを表示するには、Investigator でビジネス プロセス名選択し、［SOA 依存マップ］タブをクリックできます。 コンテンツ タイプまたは Investigator ツリー ノードを変更すると、依存マップのコンテキストが変更されることに注意してください。 たとえば、［サービス］から［オペレーション］へコンテンツ タイプを変更する場合、表示されるマップは変更されます。しかし、マップに含まれる詳細が同じままになるか変わるかは、Investigator ツリーおよびプロセス定義内に選択されたノードによって決まります。

以下の例では、発注書要求を送信するビジネス プロセスが Investigator ツリーで選択されて展開され、追加の依存関係が表示されています。




必要に応じて、ビジネス プロセスのワークフロー全体が表示されるまでマップに依存関係のレベルを追加したり、マップの特定のノードを拡大表示したりできます。 依存関係レベルの表示および非表示など、依存マップの操作の詳細については、「SOA 依存マップの使い方 (P. 81)」を参照してください。

# TIBCO BusinessWorks に関するトランザクションを追跡する

トランザクションの追跡では、ビジネス トランザクションの完了に関係する具体的な手順の詳細ビューまたはサマリ ビューが提供されます。TIBCO BusinessWorks ビジネス プロセスまたは Web サービスについて、以下のプロトコルを使用してルーティングされたオペレーションが含まれるトランザクションを追跡できます。

- SOAP（Simple Object Access Protocol）
- HTTP（HyperText Transport Protocol）
- HTTPS（Hypertext Transport Protocol Secure）
- JMS（Java Message Service）

TIBCO BusinessWorks ビジネス プロセスまたは Web サービスにかかわるトランザクションには、同期と非同期の呼び出し、および別のスレッドを使用して並列で実行されるアクティビティが含まれることがあります。 複数のスレッドまたはプロセスにまたがるトランザクションのトランザクション追跡を有効にするため、トランザクションが個々のコンポーネントやオペレーションを順番に実行するときに、相関識別子が挿入され、使用されます。その結果、実行された特定のオペレーションと各オペレーションの完了までにかかった時間に関する詳細情報を表示できます。

プラットフォームのあらゆる組み合わせにわたって BusinessTransaction を追跡するには、CA APM for SOA および CA APM for TIBCO BusinessWorks が、追跡対象のあらゆるノードで有効になっている必要があります。 このため、トランザクションが複数の JVM や CLR にまたがっている場合でも、トランザクションに関する詳細情報を表示できます。

TIBCO BusinessWorks からの呼び出しが、監視対象外のサーバ上のプロセスによって実行される場合、トランザクション追跡は、監視対象外のプロセスを呼び出すビジネス プロセスの実行時間を提供します。そのため、たとえ監視対象外のプロセス自体の実行時間が記録されなくても、返信の受信にかかる時間を特定できます。 トランザクション追跡は、エラーが発生したかどうかも追跡します。

## Tibco 内でトランザクション追跡データを使用する方法

Tibco ビジネス プロセスで非常に高い平均応答時間 (たとえば x とします) が発生する場合、x を超えるトランザクションをフィルタするのに時間フィルタを使用することはできません。 一般にビジネス プロセスは複数のトランザクションから構成されています。したがって、問題の領域を診断するには、以下の手順に従います。

1. TT ウィンドウに移動し、その特定のビジネス プロセス用のフィルタを有効にします。
2. アプリケーションを起動します。

   そのビジネス プロセスについて多数のトランザクション追跡が行われます。 追跡はビジネス プロセスの実行単位です。
3. 実行単位の長い追跡がないかどうか調べます。 特定の追跡を掘り下げて調査し、詳細情報を入手します。

   実行中の追跡間の時差が表示されます。この時間は、ビジネス プロセスがいつ一時停止されたかを示します。 この情報を使用して、ビジネス プロセスが一時停止された頻度と時間の長さを割り出します。

   HTTP 要求でタイムアウトが発生すると、対応するアクティビティでエラーが表示されます。

## プロセスにまたがるトランザクション追跡の値について

プロセスにまたがるトランザクション追跡は、サービス指向アーキテクチャ内の疎結合サービスによって実行されているオペレーションに関して貴重な情報を提供します。 プロセスにまたがるトランザクション追跡を使用して、以下のことを特定できます。

- ビジネス プロセスはコンポーネントを通じてどのようにルーティングされるか。
- トランザクション中にどのビジネス プロセスが呼び出され実行されるか。
- トランザクション中に呼び出され、実行される呼び出しのシーケンス。
- 要求または返答の処理が最も遅い箇所。

## サンプル トランザクション追跡の開始と表示

ビジネス プロセス トランザクションに、ローカルまたはリモート コンピュータ上の個別の JVM または CLR への呼び出しが含まれ、呼び出された JVM または CLR 上のエージェントが同じ Enterprise Manager への呼び出しをレポートする場合、トランザクション追跡を実行して、ビジネス プロセスの一部として実行された呼び出しのすべてを表示できます。

トランザクション追跡セッションは、以下のいずれかの方法で開始できます。

- SOA 依存マップ内のマップ ノードから直接開始する方法。
- ［Workstation］-［新規トランザクション追跡セッション］をクリックして Workstation から手動で開始する方法。

フィルタを設定してトランザクション追跡セッションでどのトランザクションを収集するかを制御したり、セッションが継続する必要のある期間を指定したりできます。 依存マップからトランザクション追跡を開始する場合は、マップ ノードのタイプに応じてデフォルト フィルタが自動的に設定されます。 新しいトランザクション追跡セッションを手動で開始する場合は、TIBCO BusinessWorks 用に以下のフィルタ タイプの 1 つを選択できます。

- ビジネス プロセス
- ネームスペース
- オペレーション

たとえば、特定の TIBCO BusinessWorks プロセス定義についてトランザクションをフィルタするには、businessprocess フィルタを選択し、プロセス定義名のすべてまたは一部を入力できます。

フィルタを設定してからトランザクション追跡セッションを開始すると、トランザクション追跡ビューアが表示されます。 トランザクション追跡ビューアから、追跡を選択して、トランザクションで行われた呼び出しに関する追加の詳細情報を表示できます。 たとえば、追跡を選択し、以下のことが可能です。

- ［サマリ ビュー］タブをクリックして、トランザクションにかかわるコンポーネントを表示する
- ［追跡ビュー］タブをクリックして、トランザクションで行われた呼び出し、およびトランザクションにかかわる各呼び出しに費やした時間を表示する
- ［シーケンス ビュー］をクリックして、トランザクションについて呼び出しのシーケンスやプロセス ワークフロー、およびトランザクション追跡で最も時間のかかったスレッドを表示する

**注**: トランザクション追跡の詳細については、「SOA 環境でトランザクション追跡を使用する (P. 111)」を参照してください。 追跡を構成する方法の詳細については、「CA APM Java Agent 実装ガイド」または「CA APM .NET Agent 実装ガイド」を参照してください。 トランザクション追跡ビューと履歴データの使用方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# BusinessWorks のフロントエンドおよびバックエンドについて

SOA でのトランザクションには、多くの場合、複数のシステムに同期および非同期呼び出しの複雑な処理が含まれます。 たとえば、1 つのトランザクションには、外部の Web サービス、TIBCO Enterprise Message Service サーバ インスタンス、およびデータベースへの呼び出しが含まれる場合があります。 TIBCO BusinessWorks の場合、バックエンド呼び出しはすべて、別々のスレッドを使用して処理される場合があり、それにより、同じトランザクションの一部としてそれらを特定することがより困難になります。

トランザクションの同じ開始点を複数のバックエンド呼び出しに関連付けることができるように、SOA Extension for TIBCO BusinessWorks には、以下の設定プロパティが含まれています。

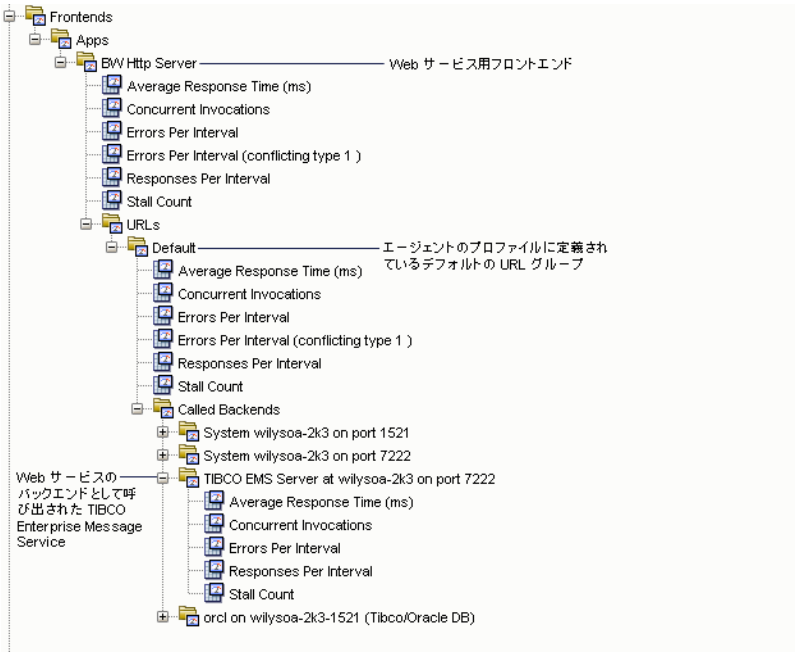
```
com.wily.soaextension.tibcobw.mbbs.enabled
```

デフォルトでは、このプロパティは true に設定されます。これにより、同じトランザクションに参加するバックエンド呼び出しのスレッドを、［バックエンド］または［呼び出されたバックエンド］ノードの下に追加し、適切な*フロントエンド*に関連付けることが可能になります。

フロントエンドは、トランザクションの出発点を識別します。 TIBCO BusinessWorks の場合、デフォルトのフロントエンドは、ビジネス プロセスに対してはプロセス開始元であり、Web サービスに対しては TIBCO BusinessWorks HTTP サーブレットです。

［Frontend］メトリックは、TIBCO BusinessWorks アプリケーション名別にフロントエンド ノードの下で収集されポストされます。 トランザクションについて同じフロントエンドによって開始されたバックエンド呼び出しのすべては、そのフロントエンドの［Called Backends］ノードの下で収集されます。たとえば、プロセス開始元が Web サービス、データベース、および TIBCO Enterprise メッセージ サービスに呼び出しを発行する場合、そのプロセス開始元に関連付けられた［Frontend］ノードの下で、その呼び出しに関するメトリックを表示できます。 ビジネス プロセスのバックエンドによって生成されたあらゆるエラーは、関連するフロントエンド ノードに伝達されます。

たとえば、TIBCO Enterprise Message Service サーバ インスタンスを呼び出した Web サービスのトランザクションに関するメトリックを表示するには、Investigator で BusinessWorks HTTP Server フロントエンドを展開し、その［Called Backends］ノードを展開できます。




## 子プロセスに関する制限について

子サブプロセスを生成するビジネス プロセスがある場合、親プロセスは通常、子プロセスの実行が完了するまで待ちません。 サブプロセスを生成したビジネス プロセスが、子プロセスの完了を待たないため、子プロセスによって生成されたあらゆるメトリックまたはエラーは、呼び出されたバックエンドに関するメトリック、またはビジネス プロセスのフロントエンドに関するメトリックに含まれない場合があります。

メインのビジネス プロセスの実行中に、子プロセスが開始され完了まで実行される場合、子プロセスに関するメトリックおよびエラーは、ビジネス プロセスのフロントエンドに含まれることがあります。

## フロントエンドとの複数のバックエンドの関連付けを無効にする

デフォルトでは、*com.wily.soaextension.tibcobw.mbbs.enabled* プロパティは true に設定されています。それにより、別々のスレッド上で実行されている複数のバックエンド システムへの呼び出しが、トランザクションを開始したフロントエンドに関連付けられるようになります。 この機能を無効にする場合は、*com.wily.soaextension.tibcobw.mbbs.enabled* プロパティを false に設定できます。

このプロパティを false に設定した場合、フロントエンドと同じスレッドを使用して実行されるバックエンド呼び出しのみが、［Called Backends］ノードの下で収集されます。 TIBCO BusinessWorks が処理のために複数のスレッドを使用することが一般的であるため、それらの対応するフロントエンドにパフォーマンス メトリックおよびエラーを関連付けることに関心がない場合は、このプロパティを false に設定するだけです。

## カスタム TIBCO BusinessWorks バックエンドを追加する

カスタム バックエンドを追加し、関連するフロントエンドにカスタム バックエンドを関連付ける場合は、カスタム バックエンド トレーサを定義することにより、カスタム バックエンドをマークし、エージェント用のカスタムの calledbackend トレーサおよび calledbackend エラー トレーサを定義する必要があります。 たとえば、以下のバックエンド トレーサを定義するとします。

```
SetTracerClassMapping: <CustomBackendTracer>
com.wily.introscope.agent.trace.BackendTracer
com.wily.introscope.probebuilder.validate.ResourceNameValidator
SetTracerParameter: <CustomBackendTracer> nameformatter <CustomNameFormatter>

TraceOneMethodWithParametersIfFlagged: <CustomTracinFlag> <method>
<CustomBackendTracer> "Metric Path"
```

この場合、以下のように CalledBackendTracer と CalledBackendErrorTracer を定義できます。

```
SetTracerClassMapping: <CustomCalledBackendTracer>
com.wily.soaextension.tibcobw.multithread.blame.CalledBackendTracer
com.wily.introscope.probebuilder.validate.ResourceNameValidator
SetTracerParameter: <CustomCalledBackendTracer> nameformatter
<CustomNameFormatter>

SetTracerClassMapping: <CustomCalledBackendErrorTracer>
com.wily.soaextension.tibcobw.multithread.blame.CalledBackendErrorTracer
com.wily.introscope.probebuilder.validate.MetricNameValidator
SetTracerParameter: <CustomCalledBackendErrorTracer> nameformatter
<CustomNameFormatter>

TraceOneMethodWithParametersIfFlagged: <CustomTracinFlag> <method>
CustomCalledBackendTracer "Metric Path"
TraceOneMethodWithParametersIfFlagged: <CustomTracinFlag> <method>
CustomCalledBackendErrorTracer "Metric Path:Errors Per Interval"
```

**注**: カスタム トレーサを構成し、名前フォーマッタを定義し、カスタム フロントエンドおよびバックエンドをマークすることの詳細については、「CA APM Java Agent 実装ガイド」および「CA APM .NET Agent 実装ガイド」を参照してください。

# メトリック エイジングおよび削除をカスタマイズする

エージェント メトリック エイジングは、エージェントのメモリ キャッシュから定期的にデッド メトリックを削除します。 デッド メトリックとは、指定した時間内に新しいデータをレポートしていないメトリックです。これは、エージェントのパフォーマンスを向上させること、システムの処理能力を超えるメトリックを収集しないようにすることに役立ちます。

ただし、実行時間の長いトランザクションまたはビジネス プロセスがある場合は、一部のメトリックに対してメトリック エイジングを回避することもできます。 たとえば、12 時間を超えて実行されているビジネス プロセスがある場合、以下のような構成プロパティを使用して、ビジネス プロセスの Concurrent Invocations（同時進行中の呼び出し）に対して、メトリック エイジングを回避できます。

```
introscope.agent.metricAging.metricExclude.ignore.2=Tibco¥|Processes¥|*.*:Concurr
ent Invocations
```

また、その他のエージェント プロパティを使用して、メトリック エイジングをカスタマイズすることもできます。たとえば、それらのプロパティを設定して、メトリックが削除候補になるまでの期間、または各間隔で確認するメトリックの数をカスタマイズできます。

**注**: メトリック エイジングおよびエージェント プロパティの構成方法の詳細については、「CA APM Java Agent 実装ガイド」または「CA APM .NET Agent 実装ガイド」を参照してください。

**メトリック エイジングから TIBCO BusinessWorks メトリックを除外する方法**

1. *<Agent_Home>/core/config* ディレクトリに移動します
2. *IntroscopeAgent.profile* ファイルをテキスト エディタで開きます。
3. *Agent Metric Aging* セクションを見つけ、以下のプロパティのコメントアウトを解除します。

```
introscope.agent.metricAging.metricExclude.ignore.2=Tibco¥|Processes¥|*.*:Concurrent Invocations
```

12 時間を超えて実行されているビジネス プロセスがある場合、メトリック エイジングから Concurrent Invocations （同時進行中の呼び出し） を除外することをお勧めします。

# TIBCO BusinessWorks の相関追跡を設定する

プロセスにまたがるトランザクション追跡では、エージェントはあるプロセスから別のプロセスへ渡すことができる相関識別子を挿入する必要があります。 エージェントはこの相関識別子を SOAP ヘッダまたは HTTP ヘッダに挿入できます。 デフォルトでは、SOA Extension for TIBCO BusinessWorks は、SOAP および HTTP ヘッダの両方で相関情報を渡すことをサポートするように設定されます。 相関情報を使用してプロセスにまたがるトランザクション追跡を有効にする方法の詳細については、「トランザクションのセグメントがリンクされる仕組みについて (P. 113)」を参照してください。 相関追跡のプロパティの設定の詳細については、「相関追跡を設定する (P. 133)」を参照してください。

# 第 9 章: TIBCO Enterprise Message Service を監視する

TIBCO Enterprise Message Service では、企業全体にわたるシステム間の同期および非同期通信を可能にすることにより、SOA インフラストラクチャの重要なコンポーネントが提供されています。 SOA Extension for TIBCO Enterprise Message Service（EMS）により、TIBCO EMS サーバの多くの重要な要素を監視できます。たとえば、その対象は、キュー、トピック、拡張コンポーネント（ブリッジ、マルチキャスト チャネル、ルートなど）です。

このセクションでは、TIBCO EMS 環境の稼働状況とオペレーションの監視および分析に使用できる、ダッシュボードおよびメトリックについて説明します。

このセクションには、以下のトピックが含まれています。

# TIBCO Enterprise Message Service について

TIBCO Enterprise Message Service は、広範囲のプラットフォームおよびアプリケーション テクノロジにわたって JMS 準拠の通信を可能にすることにより、サービス指向アーキテクチャのバックボーンとして機能できる、標準ベースのメッセージング レイヤです。

メッセージの信頼性の高い安全な配信がビジネス サービスの提供とビジネス トランザクションの完了の重要な部分であるため、SOA Extension for TIBCO EMS では、TIBCO Enterprise Message Service の重要なコンポーネントおよびパフォーマンス メトリックを監視するための、個別のエージェントが提供されます。

TIBCO Enterprise Message Service の監視に使用されるエージェントは、コア Java または .NET Agent の拡張機能ではなく、スタンドアロンのエージェントです。 Java クラスをインスツルメントする代わりに TIBCO EMS Admin API を使用して、エージェント（*EMSMonitor*）は、TIBCO EMS コンポーネントに関するパフォーマンス情報を収集し、その情報を Enterprise Manager にレポートします。

*EMSMonitor* エージェントはスタンドアロン エージェントであるため、個別のソフトウェア パッケージとして配布され、独自の *IntroscopeAgent.profile* および独自の設定手順が必要です。 たとえば、監視対象の EMS サーバに関する接続情報を設定し、フィルタを追加して、メトリックの収集対象となるコンポーネントをカスタマイズする必要があります。

該当するエージェント プロパティを設定した後、*EMSMonitor* エージェントを使用して、ローカルおよびリモートの TIBCO EMS コンポーネントを監視できます。

# SOA Extension for TIBCO EMS をインストールする方法

EMSMonitor エージェントはコア エージェントに依存しません。 TIBCO Enterprise Message Service を実行している 1 台以上のコンピュータ上に、その他の CA Introscope® コンポーネントから独立して動作する EMSMonitor エージェントをインストールして設定することができます。

SOA Extension for TIBCO EMS を追加する手順の概要は以下のとおりです。

1. TIBCO Enterprise Message Service のサポートされているバージョンがインストールされていることを確認します。

   **注**: システム要件については、「*Compatibility Guide*」を参照してください。

2. 環境に Enterprise Manager と Workstation がインストールされていることを確認します。 EMSMonitor エージェントがデータを送信する Enterprise Manager の接続情報があることを確認します。

3. スタンドアロン エージェント インストーラを実行する (P. 221)か、または応答ファイルを使用して、必要なエージェント ファイルをお使いの環境に追加します。

4. EMS サーバで監視の準備をします (P. 223)。

5. TIBCO EMSMonitor エージェント プロファイルを設定 (P. 225)して、接続および監視プロパティを定義します。

6. TIBCO BusinessWorks 用の Enterprise Manager 拡張機能を有効にします (P. 242)。

# スタンドアロン エージェント インストーラを実行する

スタンドアロン エージェント インストーラを使用すると、コア Java または .NET エージェントに依存しないスタンドアロン エージェントをインストールできます。 TIBCO Enterprise Message Service の監視を有効にする場合、スタンドアロン エージェント インストーラを使用して組織の環境にエージェント関連のファイルを追加し、Enterprise Manager へのエージェントの接続を構成できます。 スタンドアロン エージェント インストーラは、必要なファイルを抽出して適切な場所にそれらを配置します。これらのファイルは、エージェントの構成を完了するために必要に応じて変更できます。

### TIBCO EMS を監視するためのスタンドアロン エージェント インストーラを実行する方法

1. 運用環境に対応する適切なスタンドアロン エージェント インストーラを起動します。
2. ［開始画面］ページで、［次へ］をクリックします。
3. インストールする監視パッケージを選択して、［次へ］をクリックします。たとえば、TIBCO EMS の監視を有効にするには、［SOA Extension For TIBCO Enterprise Message Service］オプションを選択します。
4. インストール ディレクトリについては、［次へ］をクリックしてデフォルトの場所をそのまま使用するか、［参照］をクリックして別の場所を指定します。
5. Enterprise Manager の接続設定については、エージェントがデータを送信する宛先となる Enterprise Manager のホスト名とポート番号を指定して、［次へ］をクリックします。
6. 設定のサマリを確認し、［インストール］をクリックしてインストールを開始します。
7. インストールが完了したら、［完了］をクリックしてスタンドアロン エージェント インストーラを終了します。

## サイレント インストール用の応答ファイルを使用する

スタンドアロン エージェント インストーラを対話形式で実行したくない場合は、サンプルのスタンドアロン応答ファイルを編集して、エージェント ファイルをインストールできます。 この方法を使用すると、サイレント モードで SOA Extension for TIBCO EMS を有効にできます。

### 次の手順に従ってください:

1. スタンドアロン エージェント インストーラと同じディレクトリにある SampleResponseFile.StandaloneAgentPP.txt ファイルを開きます。
2. SampleResponseFile.StandaloneAgentPP.txt ファイルを編集して、shouldInstallTibcoEMS プロパティを true に設定します。これにより、エージェントに SOA Extension for TIBCO Enterprise Message Service が追加されます。 例：

```
shouldInstallTibcoEMS=true
```

3. SampleResponseFile.StandaloneAgentPP.txt ファイルを保存します。
4. コマンド ラインで適切なコマンドを入力して、インストーラを起動します。

**注**: エージェントをサイレント モードでインストールする方法の詳細については、「CA APM Java Agent 実装ガイド」または「CA APM .NET Agent 実装ガイド」を参照してください。

## インストール アーカイブを手動で抽出する

スタンドアロン エージェント インストーラまたはスタンドアロン応答ファイルにアクセスできない場合は、運用環境に対応するスタンドアロン インストール アーカイブをダウンロードできます。CA APM 製品は、CA サポート の CA APM ソフトウェア ダウンロード セクションからダウンロードできます。ダウンロードした後は、運用環境の適切なコマンドを使用して、手動でアーカイブからファイルを抽出できます。 たとえば、UNIX コンピュータ上では以下のように *tar* コマンドを使用します。

```
tar -xvf IntroscopeStandaloneAgentPPInstaller<バージョン>unix.tar
r
```

# TIBCO EMS サーバでの監視の準備

TIBCO Enterprise Message Service を監視できる前に、EMSMonitor エージェントが使用するユーザ アカウントを準備し、監視に必要なライブラリがローカルで使用可能であることを確認する必要があります。

**次の手順に従ってください**:

1. change-admin-acl および change-user 権限のあるアカウントを使用して、EMS サーバにログインします。 たとえば、admin アカウントを使用してログオンします。
2. EMSMonitor エージェントに対する view-all 権限のあるユーザ アカウントを見つけるか作成します。 たとえば、EMSMonitor エージェントを実行する新規のユーザを作成するには、EMS Administration Tool で以下のコマンドを実行します。

   ```
   create user wilyemsuser
   grant admin wilyemsuser view-all
   ```

3. 以下の TIBCO EMS ライブラリが *<TIBCO_EMS_HOME>/lib* ディレクトリ内で使用可能であることを確認します。

   ```
   jms.jar
   tibjms.jar
   tibjmsadmin.jar
   ```

   リモートのサーバ インスタンスを監視する場合、EMSMonitor エージェントが実行されているコンピュータを EMS クライアントとして設定する必要があります。 つまり、通常は jms.jar、tibjms.jar、および tibjmsadmin.jar ファイルが使用可能であり、これらのファイルを EMSMonitor エージェントと同じコンピュータ上の classpath に追加することが必要です。

   **注**: EMS クライアント アプリケーションのセットアップおよび classpath へのこれらのファイルの追加については、「*TIBCO Enterprise Message Service User's Guide*」を参照してください。

   **EMS バージョン 5.x 以降で有効：** Secure Socket Layer （SSL）接続を使用して、エージェントが EMS サーバに接続する必要がある場合、EMS バージョン 5.x 以降については、以下の追加のライブラリがローカル コンピュータで使用可能であることが必要です。

   ```
   tibcrypt.jar
   slf4j-api-1.4.2.jar
   slf4j-simple-1.4.2.jar
   ```

   **EMS サーバ バージョン 4.4.x で有効：** *tibcrypt.jar* ライブラリのみが必要です。

   **注**: EMS への SSL 通信の設定の詳細については、「*TIBCO Enterprise Message Service User's Guide*」を参照してください。

   **EMS サーバ バージョン 4.4.x および 5.x で有効:** EMS 4.4.x および 5.x が含まれる環境を監視する場合、ローカル コンピュータ上の必須の .jar ファイルのすべてが、使用可能な最新の .jar ファイルであることを確認します。 たとえば、環境に EMS 4.4.x および 5.x が含まれている場合は、EMSMonitor エージェントが 5.x .jar ファイルを使用していることを確認します。

# TIBCO EMSMonitor エージェントを設定する

スタンドアロン エージェント インストーラを実行した後に、*TibcoEMSMonitor* ディレクトリを確認する必要があります。*TibcoEMSMonitor* ディレクトリには、TIBCO Enterprise メッセージ サービスの監視に必要な、以下のファイルおよびディレクトリがあります。

**ext ディレクトリ**

Supportability-Agent.jar ファイルが保存されています。Supportability-Agent.jar ファイルは、システムのプロパティ、構成ファイル、その他の情報をログ ファイルに記録する拡張機能です。ログ ファイルは、EMSMonitor エージェントのサポートが必要になった場合に、CA サポートに提供することができます。

**lib ディレクトリ**

.jar ファイル（Agent.jar および jline-0.9.9.jar）としてパッケージされた必須の EMSMonitor エージェント ライブラリが保存されています。

**properties ディレクトリ**

IntroscopeAgent.profile、TibcoEMSMonitor.properties、および MonitoringLevel.xml ファイルが保存されています。

- IntroscopeAgent.profile を使用して Enterprise Manager への接続を設定できます。
- TibcoEMSMonitor.properties ファイルを使用して、ローカルおよびリモートの TIBCO EMS サーバ インスタンスに接続するためのパラメータを設定し、監視レベル、およびメトリックの収集対象となるコンポーネントをカスタマイズします。
- MonitoringLevel.xml ファイルを使用して、最小、推奨、フルの監視レベルのセットから成るメトリックをカスタマイズできます。

**EMSMonitor.jar ファイル**

EMS サーバ インスタンスを監視するために、エージェントが使用するクラスを提供します。

**emsPwdEncryptor ファイル**

パスワード暗号化スクリプト（emsPwdEncryptor.bat または emsPwdEncryptor.sh）を提供します。それにより、EMS サーバに接続するための、EMS ユーザ パスワードおよび SSL クライアント証明書パスワードを暗号化できます。

EMSMonitor ファイル

EMS サーバインスタンスの監視の開始準備ができたときに、EMSMonitor エージェントを実行する、起動スクリプト（EMSMonitor.bat または EMSMonitor.sh）を提供します。

UNIX プラットフォームで、EMSMonitor.sh スクリプトを使用して、EMSMonitor エージェントの起動、停止、再起動、ステータス確認を行うことができます。Windows で、EMSMonitor.bat または Windows サービスを使用して、エージェントを起動し、EMSMonitor エージェントを停止できます。

*TibcoEMSMonitor* ディレクトリ内のファイルを使用して、*EMSMonitor* エージェント接続プロパティ、監視プロパティ、セキュリティで保護された通信プロパティを設定できます。

## 基本的な接続プロパティの設定

*EMSMonitor* エージェントは TIBCO EMS サーバから Enterprise Manager にデータをレポートします。 メトリックの収集およびレポートを有効にするには、*EMSMonitor* エージェントのレポート先となる Enterprise Manager に関する接続情報を指定し、エージェントのデータ収集先となるサーバのリストを指定する必要があります。

EMSMonitor エージェントの接続プロパティを設定するには、以下の手順に従います。

1. テキスト エディタで *IntroscopeAgent.profile* ファイルを開いて、Enterprise Manager の接続パラメータを指定します。 例：
   ```
   introscope.agent.enterprisemanager.transport.tcp.host.DEFAULT=mercury
   introscope.agent.enterprisemanager.transport.tcp.port.DEFAULT=5001
   ```
2. テキスト エディタで *TibcoEMSMonitor.properties* ファイルを開き、*ems.server.list* プロパティに監視対象の EMS サーバ インスタンスの名前を設定します。 例：
   ```
   ems.server.list = tibco_ems_srv01, tibco_ems_srv02
   ```

このリストでは任意の名前を使用できます。監視対象のサーバインスタンスの実際の名前と一致する必要はありません。 ただし、リスト内の名前は、追加のプロパティの定義に使用するサーバインスタンス名に一致する必要があります。 たとえば、*NewYork* という名前の 2 つのサーバインスタンスがある場合、*ems.server.list* プロパティでエイリアス *newyork1* および *newyork2* を使用できます。その後、エイリアスを使用して、これらのサーバインスタンスの残りのプロパティを定義できます。 例：

```
ems.server.list = newyork1, newyork2
newyork1.host = catsmcn01
newyork1.port = 6001
```

各インスタンスの URL は、その後、Investigator ツリーに表示されるサーバインスタンス名に追加され、それにより、両方のインスタンスの監視すること、一意に識別することが可能になります。

3. *ems.server.list* プロパティ内で指定した各サーバインスタンスに接続するためのホスト名およびポートを設定します。 例：

```
tibco_ems_srv01.host=localhost
tibco_ems_srv01.port=7222

tibco_ems_srv02.host=vespa
tibco_ems_srv02.port=7222
```

4. *emsPwdEncryptor* ユーティリティを実行して、EMS サーバへの接続に使用するユーザアカウントの暗号化されたパスワードを生成します。

EMS サーバに接続するためのユーザアカウントの作成または選択については、「EMS サーバで監視の準備をする (P. 223)」を参照してください。 *emsPwdEncryptor* ユーティリティの使用方法については、「暗号化パスワードの作成 (P. 236)」を参照してください。

## サーバに対するポーリング間隔を設定する

*EMSMonitor* エージェントは、最新のステータスおよび設定情報を取得するために定期的に各サーバインスタンスをポーリングします。 これらのクエリの頻度は、構成プロパティを使用して制御できます。

**EMSMonitor エージェントがサーバをクエリする頻度を設定するには、以下の手順に従います。**

1. ステータス関連のメトリックを収集するために各サーバインスタンスをポーリングする間隔を設定します。

指定した間隔により、EMS サーバ インスタンスがステータス情報についてどれくらい頻繁にクエリされるかが制御されます。 たとえば、60 秒ごとに *tibco_ems_srv01* インスタンスをクエリし、120 秒ごとに *tibco_ems_srv02* インスタンスをクエリするには、以下のプロパティを設定します。

```
tibco_ems_srv01.delaytime=60
tibco_ems_srv02.delaytime=120
```

間隔を指定しない場合、デフォルトではメトリックが 60 秒ごとにリフレッシュされます。

2. 各サーバ インスタンスの静的な設定関連のメトリックをリフレッシュする間隔を設定します。

   設定プロパティが頻繁に変更されないため、EMSMonitor エージェントは起動時に値を一回、収集します。その後、*<ServerInstance>.report.static.freq* プロパティの値に *<ServerInstance>.delaytime* プロパティの値を掛けて得られた秒数の後に、設定メトリックをポーリングします。*<ServerInstance>.delaytime* プロパティが 60 秒に設定され、*<ServerInstance>.report.static.freq* プロパティが 20 に設定されている場合、静的メトリックは 1,200 秒ごと。つまり 20 分ごとに確認されます。

   例：

   ```
   tibco_ems_srv01.report.static.freq=20
   tibco_ems_srv02.report.static.freq=20
   ```

   このプロパティを 0 に設定した場合、*EMSMonitor* エージェントは起動時にすべての設定メトリックをレポートしますが、以降にそれらに対して行われたあらゆる変更をレポートしません。

# 選択的な監視用のフィルタを設定する

デフォルトでは、EMSMonitor エージェントは、*ems.server.list* プロパティで指定したサーバ インスタンス上のすべてのキューおよびトピックに関するメトリックを収集します。 キューおよびトピックを選択的に監視する場合、関心のあるキューおよびトピックを指定するようにフィルタを設定できます。また、フィルタを設定して、ブリッジ、マルチキャスト チャネル、ルートのような拡張コンポーネントの監視を有効にし、正規表現を使用して、メトリックの収集対象となる拡張コンポーネントのサブセットを指定できます。

## キューおよびトピック用のフィルタを設定する

EMS Server は静的/動的キューおよびトピックの両方をサポートしています。静的キューおよびトピックは EMS 管理者によって作成され管理され、それらが使用されていなくても、サーバに残ります。 動的キューおよびトピック（一時的な動的キューなど）は、EMS クライアントによって必要に応じて作成、破棄され、短期間のみサーバに残ります。

デフォルトでは、EMSMonitor エージェントは、不要なオーバーヘッドを防ぐために、静的キューおよびトピックのみを監視します。 ただし、動的キューまたは動的トピックを監視する場合は、それらを含めるようにフィルタを定義できます。 また、正規表現を使用してフィルタを設定して、メトリックの収集対象となる特定のキューおよびトピックを指定できます。

**動的キューまたはトピックの監視を有効にするには、以下の手順に従います。**

1. *TibcoEMSMonitor.Properties* ファイルをテキスト エディタで開きます。
2. 監視する EMS サーバ インスタンスの名前を確認します。
3. 動的オブジェクトを監視するサーバ インスタンスごとに、動的キューまたは動的トピックの包含フィルタを設定します。 たとえば、EMS サーバ インスタンス *tibco_ems_srv01* の動的キューと動的トピックの監視を有効にするには、以下のように指定します。

```
tibco_ems_srv01.queue.filter.include.dynamic=true
tibco_ems_srv01.topic.filter.include.dynamic=true
```

**正規表現を使用して特定のキューまたはトピックを監視する方法**

1. *TibcoEMSMonitor.Properties* ファイルをテキスト エディタで開きます。
2. 監視する EMS サーバ インスタンスの名前を確認します。
3. 必要に応じて、正規表現を使用して各サーバ インスタンスにキューとトピックのフィルタを設定します。 例：

```
tibco_ems_srv01.queue.filter.includeonly.regex=sample.*
tibco_ems_srv01.topic.filter.includeonly.regex=sample.*

tibco_ems_srv02.queue.filter.includeonly.regex=test.*
tibco_ems_srv02.topic.filter.includeonly.regex=test.*
```

フィルタを指定しなかった場合は、すべてのキューとトピックに関するメトリックが収集されます。動的および一時的なキューやトピックを含めるようにプロパティを設定した場合は、それらも収集されます。 含めるキューの正規表現を指定すると、その正規表現と一致するキューのみが監視されます。 含めるトピックの正規表現を指定すると、その正規表現と一致するトピックのみが監視されます。

## 含める拡張コンポーネントの定義

デフォルトでは、EMSMonitor エージェントはブリッジ、チャネル、またはルートに関するメトリックをレポートしません。 これらの拡張コンポーネントの監視を有効にするには、*TibcoEMSMonitor.properties* ファイルに包含フィルタを指定する必要があります。 含める拡張コンポーネントの監視を有効にした後で、正規表現フィルタを使用して、メトリックを収集する拡張コンポーネントのリストをさらに細かく設定できます。

**ブリッジ、チャネル、およびルートの監視を有効にする方法**

1. *TibcoEMSMonitor.Properties* ファイルをテキスト エディタで開きます。
2. 監視する EMS サーバ インスタンスの名前を確認します。
3. 監視する各サーバ インスタンス上の拡張コンポーネントごとに包含フィルタを設定します。たとえば、サーバ インスタンス *tibco_ems_srv1* のブリッジ、チャネル、およびルートを監視する場合は、以下の包含フィルタをファイルに追加します。

```
tibco_ems_srv1.monitor.bridges=true
tibco_ems_srv1.monitor.channels=true
tibco_ems_srv1.monitor.routes=true
```

## 拡張コンポーネントの正規表現フィルタの定義

ブリッジ、チャネル、またはルートの監視を有効にするための包含フィルタを定義した場合は、メトリックを収集するブリッジ、チャネル、またはルートを特定するための正規表現フィルタを設定できます。

コンポーネントの包含フィルタを定義して、正規表現フィルタを定義しないと、そのタイプのコンポーネントがすべて監視されます。 たとえば、ブリッジの監視を有効にして、ブリッジの正規表現フィルタを追加しないと、すべてのブリッジに関するメトリックが収集されます。 コンポーネントの包含フィルタを定義していない場合、そのコンポーネントの正規表現フィルタはすべて無視されます。

**ブリッジ、チャネル、またはルートの正規表現フィルタを定義する方法**

1. *TibcoEMSMonitor.Properties* ファイルをテキスト エディタで開きます。
2. 監視する EMS サーバ インスタンスの名前を確認します。
3. 監視する各サーバ インスタンス上の拡張コンポーネントごとに正規表現フィルタを設定します。 たとえば、サーバ インスタンス *tibco_ems_srv1* のブリッジ、チャネル、およびルートの監視を有効にした場合は、以下の包含フィルタをファイルに追加します。

   ```
   <ServerInstance>.bridge.filter.includeonly.regex
   <ServerInstance>.channel.filter.includeonly.regex
   <ServerInstance>.route.filter.includeonly.regex
   ```

   フィルタごとに、正規表現を使用して、監視するコンポーネントのサブセットを特定します。 例：

   ```
   tibco_ems_srv1.bridge.filter.includeonly.regex=new.*
   tibco_ems_srv1.channel.filter.includeonly.regex=.*bulletin
   tibco_ems_srv1.route.filter.includeonly.regex=.*
   ```

   正規表現フィルタは、包含フィルタを *true* に設定した拡張コンポーネントにのみ適用されます。

## EMS 名の特殊文字の置換

TIBCO Enterprise Message Service では、すべてのコンポーネント名にコロン（:）やパイプ（|）などの特殊文字を含めることができます。 しかし、コロン（:）とパイプ（|）は CA Introscope® の予約文字であるため、コンポーネント名に含まれる場合はアンダースコア（_）に自動的に置換されます。 たとえば、*Queue:WorkInProgress* という名前のキューがある場合、EMS エージェントはコロンをアンダースコアに置換し、*Queue_WorkInProgress* という名前のノードの下にこのキューに関するメトリックをレポートします。

*Queue:WorkInProgress* や *Queue|WorkInProgress* など、同じような名前のキューがある場合は、特殊文字の置換によって名前が重複します。 コロンとパイプを置換するとキューの名前が同じになるため、Investigator では *Queue_WorkInProgress* という名前のキューに対するメトリックが 1 セットしか表示されません。

特殊文字に関するこの制限は、すべての EMS コンポーネントに適用されます。 監視を構成するためにフィルタを定義する場合は、コロンやパイプの置換がコンポーネント名にどのように影響するかを考慮してください。

# 監視レベルの定義

EMSMonitor エージェントは、各 TIBCO EMS サーバ インスタンスに関する多数のメトリックを収集できます。 収集して Enterprise Manager にレポートするメトリックの数を制御するには、TIBCO EMS サーバ インスタンスごとに*監視レベル*を定義します。 各サーバ インスタンスの Server、Queue、および Topic メトリックの監視レベルを個別に定義できます。

## デフォルトの監視レベル定義の使用

各 EMS サーバとそのコンポーネントに関するメトリックの数を制御するため、*EMSMonitor* エージェントは Server メトリック、Queue メトリック、および Topic メトリックに関して以下の監視レベルをサポートします。

minimum

サーバ、キュー、またはトピックを監視するのに不可欠なメトリックのみを収集します。 デフォルトでは、事前設定済みのダッシュボードと［概要］タブの情報を表示するには、最小の監視レベルで定義されたメトリックが必要です。

この監視レベルは、EMS サーバ インスタンス上の Server、Queue、または Topic メトリックのオーバーヘッドを最小限に抑えたい場合に使用します。

recommended

最小レベルのすべてのメトリックに加えて、サーバの Fault Tolerance メトリック、キューのインバウンドおよびアウトバウンド メッセージの統計、トピックのマルチキャスト関連メトリックなどの追加メトリックを収集します。

この監視レベルは、オーバーヘッドのわずかな増加と引き換えに、サーバ、キュー、またはトピックのパフォーマンスとオペレーションの状況をより詳しく表示したい場合に使用します。

full

サーバ、キュー、またはトピックの（最小メトリックと推奨メトリックを含む）利用可能なすべてのメトリックを収集してレポートします。

この監視レベルは、サーバ、キュー、またはトピックのパフォーマンスとオペレーションの状況を最も詳しく表示したい場合や、パフォーマンスのオーバーヘッドを考慮する必要がない場合に使用します。

監視レベルは、各サーバ インスタンス上のサーバ、キュー、またはトピックに関して収集するメトリックのセットを規定し、各サーバ インスタンス上のコンポーネントごとに個別に設定できます。

たとえば、サーバ インスタンス *tibco_ems_server01* のすべての Server メトリック、最小の Queue メトリック、および推奨される Topic メトリックを収集するには、以下のプロパティを使用して、これらのコンポーネントの監視レベルを設定します。

```
tibco_ems_server01.monitoring.level = full
tibco_ems_server01.queue.monitoring.level = minimum
tibco_ems_server01.topic.monitoring.level = recommended
```

また、他のサーバ インスタンスのサーバ、キュー、およびトピックの監視レベルについても、これらのプロパティを設定できます。 例：

```
tibco_ems_server02.monitoring.level = minimum
tibco_ems_server02.queue.monitoring.level = recommended
tibco_ems_server02.topic.monitoring.level = recommended
```

### TIBCO EMS サーバ インスタンスの監視レベルを設定する方法

1. *TibcoEMSMonitor.Properties* ファイルをテキスト エディタで開きます。
2. 監視する EMS サーバ インスタンスの名前を確認します。
3. 各サーバ インスタンスの監視レベル プロパティである *<ServerInstance>.monitoring.level* を設定します。 たとえば、EMS サーバ インスタンス *tibco_ems_srv01* の監視レベルを *minimum* に設定し、EMS サーバ インスタンス *tibco_ems_srv02* の監視レベルを *full* に設定するには、以下のように指定します。

   ```
   tibco_ems_srv01.monitoring.level=minimum
   tibco_ems_srv02.monitoring.level=full
   ```

TIBCO EMS キューの監視レベルを設定する方法

1. *TibcoEMSMonitor.Properties* ファイルをテキスト エディタで開きます。
2. 監視する EMS サーバ インスタンスの名前を確認します。
3. 各サーバ インスタンスのキューの監視レベル プロパティである *<ServerInstance>.queue.monitoring.level* を設定します。 たとえば、EMS サーバ インスタンス *tibco_ems_srv01* のキューの監視レベルを *minimum* に設定し、EMS サーバ インスタンス *tibco_ems_srv02* のキューの監視レベルを *recommended* に設定するには、以下のように指定します。

```
tibco_ems_srv01.queue.monitoring.level=minimum
tibco_ems_srv02.queue.monitoring.level=recommended
```

設定した監視レベルは、指定したサーバインスタンスのすべての静的キューと動的キューに適用されます。

TIBCO EMS トピックの監視レベルを設定する方法

1. *TibcoEMSMonitor.Properties* ファイルをテキスト エディタで開きます。
2. 監視する EMS サーバ インスタンスの名前を確認します。
3. 各サーバ インスタンスのトピックの監視レベル プロパティである *<ServerInstance>.topic.monitoring.level* を設定します。 たとえば、EMS サーバ インスタンス *tibco_ems_srv01* のトピックの監視レベルを *full* に設定し、EMS サーバ インスタンス *tibco_ems_srv02* のトピックの監視レベルを *recommended* に設定するには、以下のように指定します。

```
tibco_ems_srv01.topic.monitoring.level=full
tibco_ems_srv02.topic.monitoring.level=recommended
```

設定した監視レベルは、指定したサーバインスタンスのすべての静的トピックと動的トピックに適用されます。

## デフォルトの監視レベル定義の変更

最小、推奨、またはフルの監視レベルに関連付けられたメトリックを変更するには、*MonitoringLevel.xml* ファイルを編集します。

*MonitoringLevel.xml* ファイルには、EMS サーバインスタンス、キュー、およびトピックに関して使用できるすべてのメトリックがリストされます。各メトリックの監視レベルもこのファイルで定義されます。*TibcoEMSMonitor.properties* ファイルでサーバ、キュー、およびトピックの監視レベルを最小、推奨、またはフルに設定したときに収集されるメトリックを変更するには、このファイルの監視レベルを変更します。

このファイルを使用して、ブリッジ、チャネル、またはルートに関連付けられたメトリックの監視レベルを設定することはできません。 ブリッジ、チャネル、またはルートの監視をカスタマイズするには、フィルタを使用します。 フィルタを使用してブリッジ、チャネル、およびルートの監視をカスタマイズする方法の詳細については、「選択的な監視用のフィルタを設定する (P. 228)」を参照してください。

### 個々のメトリックの監視レベルを変更する方法

1. *MonitoringLevel.xml* ファイルをテキストエディタで開きます。
2. メトリック設定を変更するコンポーネントの *<MetricGroup>* セクションを見つけます。 たとえば、Server メトリックの監視レベルを変更する場合は、*<MetricGroup name="Server">* セクションを見つけます。

   EMS サーバインスタンスの監視に使用できるすべてのメトリックが、各メトリック名の属性として定義されたメトリックの監視レベルと共にリストされます。たとえば、監視レベルをデフォルトで *minimum* に設定すると、Queue Count メトリックと Topic Count メトリックが収集されます。

   ```
   <Metric level="Minimum">Queue Count</Metric>
   <Metric level="Minimum">Topic Count</Metric>
   ```

3. 監視レベルを変更するメトリックを見つけ、そのメトリックの *level* 属性を新しい監視レベルに変更します。

   たとえば、キュー数とトピックス数の監視レベルを *minimum* から *recommended* に変更するには、以下のようにします。

   ```
   <Metric level="Recommended">Queue Count</Metric>
   <Metric level="Recommended">Topic Count</Metric>
   ```

   メトリックの監視レベルを *minimum* から *recommended* または *full* に変更すると、*EMSMonitor* エージェントはそのログ ファイルに警告を記録し、警告メッセージを表示します。 エージェントは、他のタイプの変更については警告を発行しません。

また、メトリックの最小セットを変更すると、一部のデータがグラフやダッシュボードに表示されなくなったり、含まれなくなったりする可能性があることにも注意する必要があります。 メトリックを *minimum* から *recommended* または *full* に変更する前に、事前設定済みまたはカスタムのビューやダッシュボードにそのメトリックが使用されているかどうかを確認してください。

4. 起動スクリプトを使用して *EMSMonitor* エージェントを再起動し、監視レベルへの変更を有効にします。

変更を行ってエージェントを再起動すると、監視レベルが *recommended* に設定されたサーバインスタンスのみで Queue Count メトリックと Topic Count メトリックが収集されるようになります。

*<MetricGroup name>* 属性と *<Metric level>* 属性は、大文字と小文字を区別しません。- ただし、*MonitoringLevel.xml* ファイルではメトリック名の大文字と小文字を区別します。

エージェントは、*MonitoringLevel.xml* ファイルを使用して各監視レベルのメトリックを決定します。 メトリックがこのファイルにリストされていないか、このファイルから削除されている場合、エージェントはそのメトリックの情報を収集したりレポートしたりできません。

*MonitoringLevel.xml* ファイルが破損しているか、*EMSMonitor* エージェントがこのファイルを読み取ることができない場合、エージェントはエラーメッセージをログに記録し、起動時に検出したメトリックをレポートします。

# 暗号化パスワードの作成

*EMSMonitor* エージェントが指定した EMS サーバに接続するには、エージェントが認証用の有効なユーザ クレデンシャルを提示できる必要があります。クライアントと EMS サーバ間で SSL（Secure Socket Layer）接続を使用しており、クライアントのセキュリティ ライセンスを確認するように設定された EMS サーバにエージェントが接続する場合、エージェントはさらに検証用の署名証明を提示できる必要があります。

### ユーザ パスワードまたはクライアント認証パスとパスワードを暗号化して格納する方法

1. *emsPwdEncryptor* スクリプトをテキスト エディタで開き、*JAVA_HOME* 環境変数を適切なディレクトリに設定します。 例：
   ```
   set JAVA_HOME=C:¥Java¥jdk1.5.0_10
   ```
2. *emsPwdEncryptor* スクリプトを実行します。

   *TibcoEMSMonitor.properties* ファイルの *ems.server.list* プロパティでリストした EMS サーバ インスタンスごとに、ユーザ名とパスワードを指定するように求めるプロンプトが表示されます。

   - サーバ インスタンスの暗号化されたパスワードを作成するには、*y* を入力し、EMS のユーザ名とパスワードを入力します。
   - サーバ インスタンスの暗号化されたパスワードの作成をスキップするには、*n* を入力します。

   各インスタンスに接続するために使用するユーザ名とパスワードを入力すると、ユーザ名と暗号化されたパスワードが *TibcoEMSMonitor.properties* ファイルに書き込まれます。 接続を成功させるためには、EMS サーバが指定されたユーザ名とパスワードを認証できる必要があります。
3. SSL（Secure Socket Layer）のプロパティを設定するように求めるプロンプトが表示されたら、*y* または *n* を入力します。

   SSL 接続を使用しており、いずれかの EMS サーバがクライアントのセキュリティ ライセンスを確認するように設定されている場合は、*y* を入力します。 その後、クライアント証明書を読み取るためのクライアント認証パスとパスワードを指定できます。

   情報を入力すると、以下の *client.identity* プロパティと暗号化された *ssl.password* プロパティが *EMSMonitor* エージェントの *TibcoEMSMonitor.properties* ファイルに書き込まれます。

   - *client.identity* プロパティには、EMS サーバが EMSMonitor エージェントの ID を確認するために使用できる証明書へのパスが指定されます。 例：
     ```
     client.identity=C:/Tibco/TibcoEMSMonitor/certs/client_identity.p12
     ```
   - *ssl.password* プロパティには、クライアントのセキュリティ ライセンス用の暗号化されたパスワードが指定されます。

# SSL を使用した TIBCO EMS サーバ インスタンスへの接続

TIBCO EMS が SSL（Secure Socket Layer）プロトコルを使用してネットワークに接続するように設定されている場合は、SSL を使用して TIBCO EMS サーバに接続するように *EMSMonitor* エージェントを設定する必要があります。 SSL を使用するように *EMSMonitor* エージェントを設定するには、*TibcoEMSMonitor.properties* ファイルでいくつかのプロパティを設定します。 一部のプロパティは、EMS サーバ インスタンスごとに個別に設定する必要があります。 その他プロパティは、すべての EMS サーバ インスタンスに対して 1 回で設定できます。

### TIBCO EMS サーバ インスタンスの SSL 接続情報を定義する方法

1. *TibcoEMSMonitor.Properties* ファイルをテキスト エディタで開きます。
2. 監視する EMS サーバ インスタンスの名前を確認します。
3. *<ServerInstance>.ssl.connection* プロパティを設定して、各サーバ インスタンスの暗号化通信を有効（*enable*）または無効（*disable*）にします。このプロパティには、サーバ インスタンス名を指定する必要があります。

   たとえば、SSL を使用して EMS サーバ インスタンス *tibco_ems_srv01* に接続するようにエージェントを設定するには、以下のように指定します。

   ```
   tibco_ems_srv01.ssl.connection=enable
   ```

4. *<ServerInstance>.verify.host* プロパティを設定して、EMSMonitor エージェントが EMS サーバの証明書を確認する必要があるかどうかを指定します。

   このプロパティには、サーバ インスタンス名を指定する必要があります。 たとえば、エージェントが EMS サーバ インスタンス *tibco_ems_srv01* に接続するときに確認を求めるには、以下のように指定します。

   ```
   tibco_ems_srv01.verify.host=true
   ```

   このプロパティを *true* に設定すると、エージェントは EMS サーバのセキュリティ ライセンスを *trusted.certificates* プロパティで定義されたリストと照らし合わせて確認します。

5. *trusted.certificates* プロパティを設定して、EMSMonitor エージェントがサーバの証明書を確認するために使用する信頼された証明書のカンマ区切りリストを指定します。 例：

   ```
   trusted.certificates=C:/Tibco/wily/TibcoEMSMonitor/certs/tbx_root.cert.pem
   ```

   このプロパティは、*verify.host* プロパティを *true* に設定したときに必要であり、SSL を使用するすべての EMS サーバ インスタンスに適用されます。

6. *<ServerInstance>.verify.hostname* プロパティを設定して、エージェントがサーバの証明書の CN（Common Name、共通名）を確認する必要があるかどうかを指定します。

   ```
   tibco_ems_srv01.verify.hostname=true
   ```

   このプロパティを *true* に設定すると、エージェントは接続されたホストの名前または *<ServerInstance>.expected.name* プロパティに指定された名前を、サーバの証明書内の CN（Common Name、共通名）フィールドの値と比較します。 名前が一致しない場合、エージェントは接続を拒否します。

   このプロパティを *false* に設定すると、エージェントはサーバとの SSL 接続を確立しますが、サーバの名前を確認しません。

7. *<ServerInstance>.expected.hostname* プロパティを設定して、*EMSMonitor* エージェントがサーバの証明書内の CN（Common Name、共通名）フィールドの値と比較する名前を指定します。

   ```
   tibco_ems_srv01.expected.hostname=tbxserver
   ```

8. *cipher.suites* プロパティを設定して、EMSMonitor エージェントが SSL 対応の EMS サーバとの通信を暗号化するために使用できる暗号スイートのカンマ区切りリストを指定します。-

*cipher.suites* プロパティはオプションです。 *EMSMonitor* エージェントは、監視対象の EMS サーバがサポートする任意の暗号化パッケージを使用できます。このプロパティを設定すると、すべての SSL 対応の EMS サーバ インスタンスに適用されます。

暗号スイートの標準アルゴリズム名または OpenSSL 名を使用できます。たとえば、*cipher.suites* プロパティを *RC4-MD5*（OpenSSL 名を使用した場合）と *SSL_RSA_WITH_RC4_128_MD*（標準の名前を使用した場合）のどちらに設定しても、同じ暗号スイートを参照できます。

例：

```
cipher.suites=RC4-MD5,RC4-SHA
```

9. クライアントのセキュリティ ライセンスを確認するように EMS サーバを設定した場合は、クライアント証明書のパスと暗号化されたパスワードを設定するために *emsPwdEncryptor* ユーティリティを実行します。

   *emsPwdEncryptor* ユーティリティの使用方法については、「暗号化パスワードの作成 (P. 236)」を参照してください。

   EMS サーバでクライアント証明書を確認する必要がない場合は、この手順をスキップできます。

# EMSMonitor エージェントの起動

接続プロパティを設定した後は、追加のプロパティを設定するか、またはエージェントを起動して EMS サーバ インスタンスの監視を開始することができます。 デフォルトの監視プロパティを使用する場合は、以下の方法を使用できます。

- 起動スクリプトからエージェントを起動する方法
- エージェントを Windows サービスとして実行するように設定する方法

## 起動スクリプトによる EMSMonitor の起動

UNIX コンピュータでは、EMSMonitor.sh スクリプトを使用して EMSMonitor エージェントの起動と停止を行います。Windows では、EMSMonitor.bat スクリプトを使用するか、または Windows サービスとして実行するように EMSMonitor エージェントを設定することができます。

**次の手順に従ってください**:

1. EMSMonitor 起動スクリプトをテキスト エディタで開きます。
2. TIBCO_EMS_HOME 環境変数を TIBCO Enterprise Message Service のインストール ディレクトリに設定します。 例：
   ```
   set TIBCO_EMS_HOME=C:¥tibco¥ems¥5.1
   ```
3. JAVA_HOME 環境変数を適切なディレクトリに設定します。 例：
   ```
   set JAVA_HOME=C:¥Java¥jdk1.5.0_10
   ```
   **JRE バージョン 1.5 以降で有効：** TIBCO Enterprise Message Service を監視するには、JRE バージョン 1.5 以降が必要です。
4. EMSMonitor スクリプトを実行します。 UNIX コンピュータ上で EMSMonitor スクリプトを実行する場合は、以下のコマンド ライン パラメータがサポートされます。
   ```
   EMSMonitor.sh [start|stop|restart|status]
   ```
   エージェントの新しいインスタンスを起動するには、start オプションを使用します。たとえば、UNIX コンピュータ上で以下のように入力します。
   ```
   ./EMSMonitor.sh start
   ```

## Windows サービスとしての EMSMonitor エージェントの実行

Windows に *EMSMonitor* エージェントをインストールする場合は、エージェントを Windows サービスとして実行するオプションがあります。エージェントを Windows サービスとして実行することには、以下のようなメリットがあります。

- ホスト コンピュータの起動時またはシャットダウン時に、エージェント サービスを自動的に起動または停止できます。
- エージェントをコンソール内ではなくバックグラウンド プロセスとして実行できるため、改ざんや不正アクセスに対する脆弱性が軽減されます。
- ユーザが現在のセッションからログオフしても、エージェントの実行を継続できます。

EMSMonitor エージェントを Windows サービスとして実行する方法

1. 前のセクションの説明に従って、EMS Server、*IntroscopeAgent.profile*、および *TibcoEMSMonitor.properties* ファイルをインストールして設定します。
2. *JAVA_HOME* 環境変数が、適切な JVM に設定され、ユーザ変数ではなくシステム変数として設定されていることを確認します。
3. *TibcoEMSMonitor¥Windows Service¥jsw-3.2.3¥conf¥ wrapper.conf* ファイルをテキスト エディタで開きます。
4. *TIBCO_EMS_HOME* 環境変数を TIBCO Enterprise Message Service のインストール ディレクトリに設定します。 例：
   `set TIBCO_EMS_HOME=C:¥tibco¥ems¥5.1`
5. *wrapper.conf* ファイルを保存します。
6. コマンド ウィンドウを開き、*TibcoEMSMonitor¥Windows Service¥ RegisterEMSMonitorAgentService.bat* ファイルを実行して、EMSMonitor エージェントを Windows サービスとして登録します。

   後で登録済みのサービスから EMSMonitor エージェントを削除するには、*DeRegisterEMSMonitorAgentService.bat* ファイルを実行します。

# Enterprise Manager 拡張機能を有効にする

Enterprise Manager をインストールすると、CA APM for TIBCO Enterprise Message Service のファイルが *<EM_Home>/examples* ディレクトリにデフォルトでインストールされます。 CA APM for TIBCO Enterprise Message Service を有効にするには、Enterprise Manager のファイルを *<EM_Home>/examples* ディレクトリから Enterprise Manager のホーム ディレクトリ内の適切な場所にコピーまたは移動する必要があります。

**注**: CA APM for TIBCO Enterprise Message Service を使用する前に、CA APM for SOA を Enterprise Manager 上で有効にする必要があります。 CA APM for SOA Enterprise Manager 拡張機能を有効にする方法については、「Enterprise Manager 上で拡張機能を有効にする (P. 44)」を参照してください。

**次の手順に従ってください**:

1. CA APM for TIBCO Enterprise Message Service のディレクトリ（*SOAExtensionForTibcoEMS*）が *<EM_Home>/examples* ディレクトリ内にあることを確認し、*<EM_Home>/examples/SOAExtensionForTibcoEMS* ディレクトリのファイルを Enterprise Manager ディレクトリ構造の対応する場所にコピーします。 たとえば、*<EM_Home>/examples/SOAExtensionForTibcoEMS/ext* ディレクトリのファイルを *<EM_Home>/ext* ディレクトリにコピーします。
2. Enterprise Manager がクラスタ化環境内のコレクタである場合は、*<EM_Home>/config/modules* ディレクトリから CA APM for TIBCO Enterprise Message Service 管理モジュール（*TibcoEMSManagementModule.jar*）を削除します。

   この管理モジュールは、MOM コンピュータとして使用している Enterprise Manager の *<EM_Home>/config/modules* ディレクトリにのみコピーする必要があります。 他のすべてのファイルとスクリプトは、コレクタ Enterprise Manager と MOM Enterprise Manager の両方にインストールする必要があります。
3. Workstation を再起動すると、SOA extension for TIBCO Enterprise Message Service 専用のダッシュボードと［概要］タブがロードされます。

# ダッシュボードを使用して TIBCO EMS を監視する

SOA extension for TIBCO Enterprise Message Service には、EMS 環境の全般的な稼働状況を監視するために使用できる事前設定済みのダッシュボードがいくつか含まれています。 ダッシュボードは、ユーザが問題をすばやく診断して解決できるように、展開されたエージェントからデータを集計してパフォーマンス情報を要約します。

通常、ダッシュボードは以下の機能を備えているため、環境を監視するための起点として使用されます。

- Enterprise Message Service の主要コンポーネントの全般的な稼働状況、パフォーマンス、可用性、および現在のステータスをひとめで監視する。
- 警告または危険のしきい値を超えたことがより低レベルのメトリックによって検出されると、実運用アプリケーション環境での潜在的な問題について早期の通知を受け取る。
- パフォーマンス情報にドリルダウンして、Enterprise Message Service のどのコンポーネントに遅延やエラーが発生しているかを特定する。

事前設定済みの TIBCO Enterprise Message Service ダッシュボードは、TIBCO Enterprise Message Service 管理モジュール（*TibcoEMSManagementModule.jar*）の一部として TIBCO Enterprise Message Service 用の Enterprise Manager 拡張にパッケージ化されています。

TIBCO Enterprise Message Service 管理モジュールには、TIBCO Enterprise Message Service のために以下の事前設定済みのダッシュボードが用意されています。

Tibco EMS - **概要**

Enterprise Message Service の主要アクティビティに関するトップレベルの概要。これには、EMS サーバとエージェント間の接続ステータス、EMS サーバの全般的な稼働状況、キューの最大の深さ、トピックの最大の深さ、ルートとチャネルの全般的な稼働状況、および使用中の接続の割合に関するアラート インジケータが含まれます。

Tibco EMS - **サーバ**

すべての EMS サーバに関する要約されたステータス。これには、EMS サーバとバックアップ サーバのステータスや使用中の接続の割合に関するアラート インジケータが含まれます。

このダッシュボードには、接続数、コンシューマ数、プロデューサ数、全般的なインバウンドとアウトバウンドのメッセージ レート、および保留中メッセージ数を要約したグラフも表示されます。

Tibco EMS - キュー

すべての EMS キューに関する要約されたステータス。これには、キューのインバウンドとアウトバウンドのメッセージ レートに関するアラート インジケータとグラフ、キューの最大の深さに関するアラート インジケータ、およびキューの保留中メッセージ数とコンシューマ数に関するグラフが含まれます。

Tibco EMS - トピック

すべての EMS トピックに関する要約されたステータス。これには、トピックのインバウンドとアウトバウンドのメッセージ レートに関するアラートとグラフ、トピックの最大の深さに関するアラート インジケータ、および保留中メッセージ数に関するグラフが含まれます。

このダッシュボードには、サブスクライバの総数や、アクティブなサブスクライバと恒久サブスクライバの数も表示されます。

Tibco EMS - ルート

すべての EMS ルートに関する要約されたステータス。これには、ルートのインバウンドとアウトバウンドのメッセージ レートに関するアラートとグラフ、ルートのステータスに関するアラート インジケータ、およびバックログ数とバックログ サイズに関するグラフ（該当する場合）が含まれます。

Tibco EMS - チャネル

すべての EMS チャネルに関する要約されたステータス。これには、全体的なチャネルのステータス、チャネル メッセージ レート、バックログ数に関するアラート インジケータ、およびメッセージ レート、バイト レート、バックログ数、バックログ サイズに関するグラフが含まれます。

次の手順に従ってください:

1. Enterprise Manager を起動します（現在実行されていない場合）。
2. Workstation を起動し、SOA extension for TIBCO EMS がインストールされている Enterprise Manager にログインします。
3. ［Workstation］-［新規コンソール］を選択します。

4. ダッシュボードのドロップダウン リストから TIBCO Enterprise Message Service のいずれかのダッシュボードを選択します。

   たとえば、TIBCO Enterprise Message Service 環境の全般的な稼働状況に関する情報（接続ステータス、使用済み接続の割合、キューおよびトピックの最大の深さを含む）を表示するには、［Tibco EMS - 概要］ダッシュボードを選択します。

5. サーバ、キュー、トピック、ルート、またはチャネルに関する詳細情報を表示するには、該当するコンポーネントのタブをダブルクリックして、そのコンポーネントのダッシュボードを表示します。たとえば、［Tibco EMS - サーバ］ダッシュボードを表示するには、［サーバ］タブをダブルクリックします。-

# TIBCO EMS に関するメトリックの概要と表示

*EMSMonitor* エージェントは、ローカルとリモートの EMS サーバ インスタンスを監視し、それらの全般的な稼働状況に関するデータを［Tibco EMS サーバ］-［*<host_name>*］-［*<EMS_server_instance_name>*］ノードの下に表示します。

以下のメトリック カテゴリのメトリックを使用して、TIBCO EMS コンポーネントのパフォーマンスと稼働状況を監視できます。

Bridges

ブリッジを使用すると、1 つの宛先に送信したメッセージがブリッジされたすべての宛先にも配信されるように、宛先間でメッセージをルーティングできます。 1 つの宛先から同じまたは異なるタイプの 1 つ以上の宛先に対してブリッジを作成できます。 たとえば、トピックからキューへのブリッジ、キューからトピックへのブリッジ、1 つの宛先と複数の宛先間のブリッジなどを作成できます。

［Bridges］ノードの下には、定義した各ブリッジに関連付けられた個々のターゲットの宛先に関するメトリックが表示されます。

Channels

マルチキャスト メッセージングでは、購読しているコンシューマにメッセージのコピーを個別に送信する代わりに、多くのコンシューマに同時にメッセージをブロードキャストします。マルチキャスト メッセージングでは、サーバはメッセージをマルチキャスト チャネル経由でマルチキャスト対応のトピックに送信します。 チャネルは、サーバがメッセージを送信するときの宛先となるマルチキャスト ポートとマルチキャスト グループ アドレスを決定します。

［Channels］ノードの下には、個々のマルチキャスト チャネルに関するメトリックが表示されます。

Last Check

［Last Check］ノードの下には、EMS サーバ インスタンスとエージェント間の接続に関するメトリックが表示されます。

Queues

キューは、Enterprise Message Service ネットワーク内のクライアントや他のキューに転送されるのを待っているメッセージが格納される一時的なストレージ オブジェクトです。

［Queues］ノードの下には、配信済みまたは転送中のメッセージの数、各キューのインバウンドまたはアウトバウンドのレートなど、個々のキューに関するメトリックが表示されます。

Routes

ルートは、2 つの TIBCO Enterprise Message Service サーバをサーバ ペアとして接続するために使用されます。 ペア内の各サーバは、もう一方のサーバ上の対応する宛先へのルートに沿ってメッセージをルーティングします。ルートは、両方のサーバで同じ名前を持つグローバル トピックに関するメッセージ、またはキューの所有者が同じであるルーティング済みキューに関するメッセージのみを転送します。

［Routes］ノードの下には、ルート バックログ内のメッセージ数や、各ルートのインバウンドおよびアウトバウンドのレートなど、サーバ間で定義された個々のルートに関するメトリックが表示されます。

Server

Server カテゴリは、Enterprise Message Service のメイン ランタイム プロセスに関するメトリックを提供します。 EMS サーバ プロセスは、メッセージング トランザクションを構成する他のコンポーネントを作成して管理します。

［Server］ノードの下には、接続数、使用可能なメモリ、サーバが管理しているキューとトピックの数など、サーバ プロセスに関するメトリックが表示されます。

Topics

トピックは、パブリッシャがメッセージを書き込むことができ、サブスクライバが発行されたメッセージを受信できる、論理的なサブジェクトを表します。 メッセージのコピーが 1 つしか格納されず、それを受信できるレシーバが 1 つしか存在しないキューとは異なり、トピックは 1 つのメッセージを関心のある複数のサブスクライバに発行するように、EMS サーバによって管理されます。

［Topics］ノードの下には、各トピックのインバウンド メッセージ数、アウトバウンド メッセージ数、保留中メッセージ数など、個々のトピックに関するメトリックが表示されます。

### Investigator で TIBCO EMS メトリックのサマリを表示および操作する方法

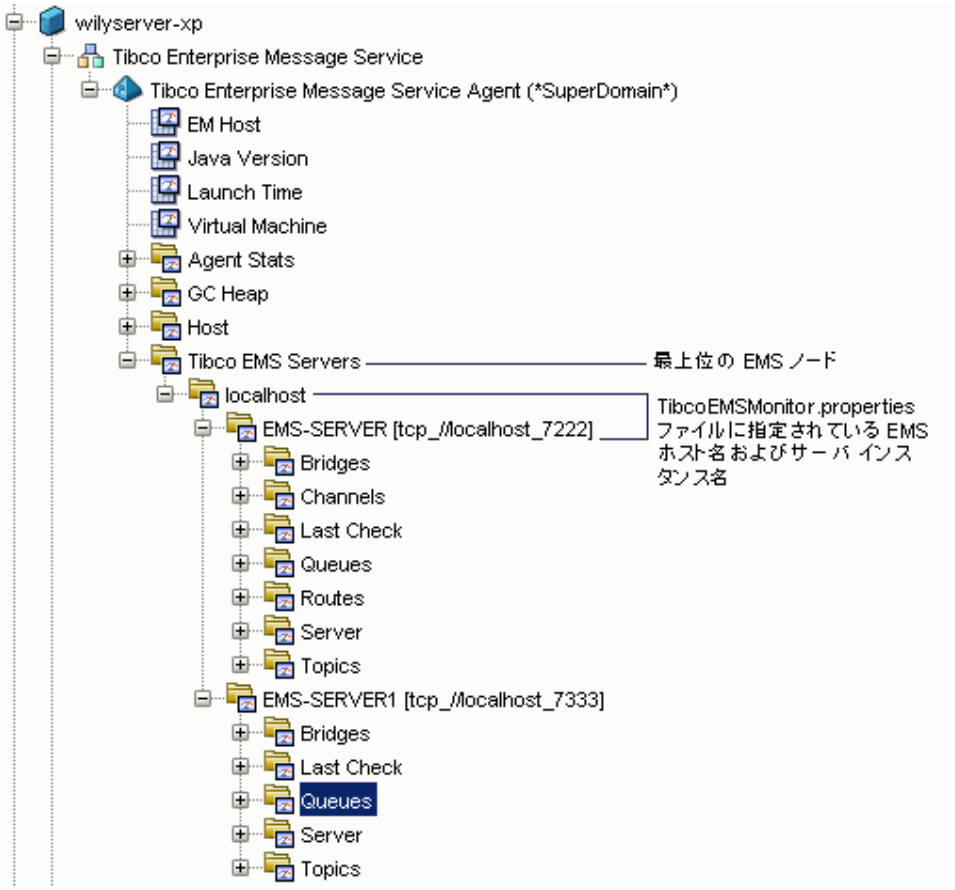
1. エージェント ノードおよび［Tibco Enterprise Message Service］ノードを展開し、［Tibco EMS サーバ］をクリックして［概要］タブを表示します。このタブには、監視対象となるすべてのサーバ インスタンスが現在のステータスと共に一覧表示されます。
2. サーバ インスタンスを選択すると、そのサーバ インスタンスの最もクリティカルな Status メトリックが［概要］タブにグラフィカルな形式で表示されます。

3. ［構成］タブをクリックすると、選択したサーバインスタンスの Configuration メトリックが表示されます。
4. サーバインスタンスを展開し、いずれかのサブノードをクリックすると、［ビューア］ペイン内のタブにそのメトリック カテゴリに関するサマリ情報または構成情報が表示されます。たとえば、［Queues］ノードを選択してから［一時キュー］タブをクリックすると、すべての一時キューに関するメトリックのサマリが表示され、［動的キュー］タブをクリックすると、すべての動的キューに関するメトリックのサマリが表示されます。
5. いずれかのサブノードを展開するか、または特定のキューやトピックなどの個々のコンポーネントを選択すると、選択したコンポーネントとそれに関連付けられたメトリックに関する詳細情報が表示されます。

### Investigator で TIBCO EMS メトリックのノードを表示および操作する方法

1. エージェントを展開し、［Tibco EMS サーバ］ノードを展開すると、監視している TIBCO EMS サーバインスタンスのホスト名が表示されます。
2. *TibcoEMSMonitor.properties* ファイルで定義したホスト名に対応する個々の *server_name* ノードを展開します。

3. *TibcoEMSMonitor.properties* ファイルで指定したサーバ インスタンスに対応する個々の TIBCO EMS サーバ *instance_name* ノードを展開すると、トップレベルの TIBCO EMS メトリック カテゴリのサブノードが表示されます。-- 例 :




4. サブノードを展開すると、そのメトリック カテゴリに関する情報が表示されます。 たとえば、選択したキューの Configuration メトリックまたは Status メトリックを表示するには、［Queues］-［Static Queues］-［*<queue_name>*］をクリックします。

5. サブノードをさらに展開すると、選択したキューの Configuration メトリックまたは Status メトリックが表示されます。 たとえば、選択したキューの Status メトリックを表示するには、［Undelivered Message Queue］-［*<queue_name>*］-［Status］ノードをクリックします。

## 最終チェックに関するメトリック

*EMSMonitor* エージェントは、TIBCO EMS サーバとそのコンポーネントのパフォーマンスとオペレーションに関するメトリックに加えて、各 EMS サーバとエージェント間の接続に関するメトリックを収集します。［Last Check］ノードのメトリックは、30 秒ごとに収集されます。これらのメトリックの間隔は、遅延時間や静的な間隔の構成プロパティでは制御できません。また、これらのメトリックはすべての監視レベルで収集されます。

各 EMS サーバと *EMSMonitor* エージェント間の接続ステータスを評価するために、以下のメトリックが用意されています。

Agent - EMS Connection Status

EMSMonitor エージェントが TIBCO EMS サーバに現在接続されているかどうかを示すテキスト文字列。

メトリック値が Connected の場合は、接続が存在することを示します。値が Not Connected の場合は、エージェントと EMS サーバが切断されていることを示します。

Agent - EMS Connection Status Value

EMSMonitor エージェントが TIBCO EMS サーバに現在接続されているかどうかを示す数値。

メトリック値が 0（ゼロ）の場合は、エージェントが EMS サーバに接続されていることを示します。メトリック値が 1 の場合は、エージェントが EMS サーバに接続されていないことを示します。

EMS Server Name

監視している EMS サーバの名前。

Timestamp

Last Check メトリックが最後に収集された日時。このメトリックの形式は以下のとおりです。

yyyy-MM-dd HH:mm:ss

## キューに関するメトリック

以下の設定メトリックを TIBCO EMS の動的および静的キューについて使用できます。

Bridged Queue

キューにブリッジ ターゲットがあるかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、キューにブリッジ ターゲットがあることを示します。
- メトリック値が false の場合は、キューにブリッジ ターゲットがないことを示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Exclusive

キューに排他プロパティが設定されているかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、キューが排他キューとして設定されていることを示します。
- メトリック値が false の場合は、キューが排他キューとして設定されていないことを示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Expiration (Sec)

選択したキュー内のメッセージが期限切れになるまでの最大時間を示します。 このメトリックの値がゼロ（0）である場合、メッセージは期限切れになりません。

キューに対してサーバの有効期限のプロパティが設定されている場合、このプロパティの値はメッセージ プロデューサが設定した JMSExpiration 値より優先されます。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Failsafe

キューがフェールセーフ ターゲットとして設定されているかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、キューがフェールセーフ ターゲットとして設定されていることを示します。
- 値が false の場合は、キューがフェールセーフ ターゲットとして設定されていないことを示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

**注**: このメトリックは EMS バージョン 4.4 にのみ有効です。

Global

あるサーバから別のサーバへメッセージをルーティングするためのグローバル キューとしてキューが設定されているかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、キューがグローバル キューであることを示します。
- メトリック値が false の場合は、キューがグローバル キューではないことを示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Is Route Connected

選択したキューについてルートが現在接続されているかどうかを示します。選択したキューがルーティング キューである場合のみ、このメトリックは表示されます。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Maximum Flow Control Bytes

このキューについてフロー制御を有効にするために使用するバイトの最大数を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Maximum Messages

このキューについてサーバが保留中メッセージとして保存できるメッセージの最大数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているときのみ、このメトリックは収集されます。

Maximum Redelivery

選択したキューからキューの受信者への指定したメッセージの配信をサーバが試行できる最大回数を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Overflow Policy

メッセージの最大サイズまたは最大数を超えた場合にキューに使用されるオーバーフロー ポリシーを示します。 以下のオーバーフロー ポリシー値が有効です。

- 0 — デフォルトのオーバーフロー ポリシーを示します。 メッセージの最大サイズまたは最大数を超えた場合、サーバは新しいメッセージを拒否し、メッセージ プロデューサにエラーを返します。
- 1 — discardOld オーバーフロー ポリシーを示します。 キュー内のメッセージが最大バイトまたは最大数を超えた場合、最も古いメッセージがキューから破棄され、エラーがメッセージ プロデューサに返されます。
- 2 — rejectIncoming オーバーフロー ポリシーを示します。メッセージの最大サイズまたは最大数を超えた場合、サーバは新しいメッセージを拒否し、メッセージ プロデューサにエラーを返します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Prefetch Count

メッセージ コンシューマが EMS サーバから取得できるメッセージの最大数を示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Route Name

キューと関連付けられるルート名を示します。選択したキューがルーティング キューである場合のみ、このメトリックは表示されます。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Routed Queue

選択したキューがルーティング キューかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、キューがルーティング キューであることを示します。
- 値が false の場合は、キューがルーティング キューでないことを示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Secure

受信接続を認証するようにキューが設定されているかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、受信接続が認証されることを示します。
- メトリック値が false の場合は、受信接続が認証されないことを示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Sender Name

キューに対して sender_name プロパティが設定されているかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、sender_name プロパティが設定されていることを示します。
- メトリック値が false の場合は、sender_name プロパティが設定されていないことを示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Sender Name Enforced

キューに対して sender_name_enforced プロパティが設定されているかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、sender_name プロパティが適用されていることを示します。
- メトリック値が false の場合は、sender_name プロパティが適用されていないことを示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Store Name

永続的メッセージが保存されるストアの名前を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

**注**: このメトリックは、EMS バージョン 5.x 以降で有効です。

以下のステータス メトリックを TIBCO EMS の動的および静的キューについて使用できます。

Delivered Message Count

配信されて認識されたメッセージの合計数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Inbound Byte Rate

EMS クライアントおよびルーティングされたサーバから選択したキューに送信される 1 秒あたりのインバウンド バイト数を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

In Transit Message Count

キューの所有者に配信されたが、まだ認識されていないメッセージの合計数を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Inbound Message Rate

EMS クライアントまたはルーティングされたサーバから選択したキューに 1 秒あたりに送信されたインバウンド メッセージ数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Outbound Byte Rate

選択したキューのコンシューマに送信されたか、ほかのサーバにルーティングされた 1 秒あたりのアウトバウンド バイト数を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Outbound Message Rate

選択したキューのコンシューマに送信されたか、ほかのサーバにルーティングされた 1 秒あたりのアウトバウンド メッセージ数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Pending Message Count

選択したキュー内に現在ある保留中メッセージの合計数を示します。このメトリックは監視キューの深さと同じです。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Pending Message Size

選択したキューのすべての保留中メッセージの合計サイズ（KB 単位）を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Receiver Count

選択したキューの受信者の数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

## サーバに関するメトリック

TIBCO EMS サーバ プロセスについては、以下の Configuration メトリックが用意されています。

Authorization Enabled

サーバで許可機能が有効になっているかどうかを示します。

このメトリックが True である場合、このサーバにアクティブに接続する他のサーバは、名前とパスワードによって自身を認証する必要があります。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Backup Server Name

選択したサーバ インスタンスのバックアップ サーバとして設定されたサーバ インスタンスの名前とコンピュータのホスト名を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Client Heartbeat Server Interval (sec)

クライアントからサーバに送信されるハートビート メッセージの間隔（秒単位）を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Client Timeout Server Connection (sec)

クライアントがサーバへの接続を終了する前にサーバからのハートビートを待つ時間（秒単位）を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Fault Tolerant Activation Time (sec)

バックアップ サーバがアクティブなサーバに障害が発生したと判定する前にハートビート メッセージを待つ時間（秒単位）を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Fault Tolerant Reread

フェールオーバの後で、バックアップ サーバがメイン ファイルを除く構成ファイルの再読み取りを行うかどうかを示します。

メトリック値が true の場合は、ファイルの再読み取りが行われることを示します。値が false の場合は、ファイルの再読み取りが行われないことを示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Fault Tolerant Heartbeat Interval (sec)

アクティブ サーバからバックアップ サーバに送信されるハートビート メッセージの間隔（秒単位）を示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Fault Tolerant Reconnect Timeout (sec)

新しくアクティブになったサーバがフェールオーバの後でクライアントが再接続するのを待つ時間（秒単位）を示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Fault Tolerant Server URL

バックアップ サーバの URL を示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Flow Control Enabled

サーバでメッセージ プロデューサと宛先のフロー制御が有効になっているかどうかを示します。

メトリック値が true の場合は、フロー制御が有効になっていることを示します。値が false の場合は、フロー制御が有効になっていないことを示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Maximum Log File Size

ログ ファイルの最大サイズ（KB 単位）を示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Maximum Connections

EMS サーバへの接続の最大数を示します。

このメトリックが 0（ゼロ）の場合、許可される接続の数に制限はありません。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Maximum Message Memory

メッセージを格納するために使用できるメモリの最大量（バイト単位）を示します。

このメトリックが 0（ゼロ）の場合、メッセージに使用できるメモリのサイズに制限はありません。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Maximum Statistics Memory

詳細な統計情報を収集するために割り当てることができるメモリの最大量（バイト単位）を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Minimum Size Asynchronous Store file

サーバの非同期ストア ファイルの最小サイズを示します。

このメトリックの値は、ストア ファイルの構成によって MB 単位の場合と GB 単位の場合があります。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

**注**: このメトリックは EMS バージョン 4.4 にのみ有効です。

Minimum Size Store file

サーバのストア ファイルの最小サイズを示します。

このメトリックの値は、ストア ファイルの構成によって MB 単位の場合と GB 単位の場合があります。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

**注**: このメトリックは EMS バージョン 4.4 にのみ有効です。

Minimum Size Synchronous Store file

サーバの同期ストア ファイルの最小サイズを示します。

このメトリックの値は、ストア ファイルの構成によって MB 単位の場合と GB 単位の場合があります。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

**注**: このメトリックは EMS バージョン 4.4 にのみ有効です。

Multicast Enabled

サーバがマルチキャスト メッセージングに対応しているかどうかを示します。

メトリック値が true の場合は、マルチキャストが有効になっていることを示します。値が false の場合は、マルチキャストが有効になっていないことを示します。

EMS バージョン 4.4.x ではマルチキャスト メッセージングがサポートされないため、EMS 4.4.x のサーバ インスタンスではこのメトリックは常に false になります。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Reserve Memory

予約メモリのサイズ（KB 単位）を示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Routing Enabled

サーバでルーティングが有効になっているかどうかを示します。

メトリック値が true の場合は、ルーティングが有効になっていることを示します。値が false の場合は、ルーティングが有効になっていないことを示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Server Heartbeat Client Interval (sec)

接続を確認するためにサーバからクライアントに送信されるハートビートの間隔（秒単位）を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Server Heartbeat Server Interval (sec)

このサーバから別のサーバに送信されるハートビートの間隔（秒単位）を示します。 2 つのサーバは、ルートによって接続されるか、またはフォールト トレランスのペアとして接続されます。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Server Timeout for Client Connection (sec)

サーバがクライアントへの接続を終了する前にクライアントからのハートビートを待つ時間（秒単位）を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Server Timeout for Server Connection (sec)

サーバが別のサーバへの接続を終了する前にそのサーバからのハートビートを待つ時間（秒単位）を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Server Start Time

サーバが起動した時刻（yyyy MM-dd HH:mm:ss 形式）を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Statistics Cleanup Interval (sec)

統計のクリーンアップ間隔（秒単位）を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Statistics Enabled

統計の収集が有効になっているかどうかを示します。

メトリック値が true の場合は、統計が収集されることを示します。値が false の場合は、統計が収集されないことを示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Store Truncation Enabled

サーバが必要に応じてストア ファイルを切り捨てるかどうかを示します。

メトリック値が true の場合は、ストアの切り捨てが有効になっていることを示します。値が false の場合は、ストアの切り捨てが有効になっていないことを示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

**注**: このメトリックは EMS バージョン 4.4 にのみ有効です。

Swapping Enabled

サーバがメッセージをプロセス メモリからディスクにスワップできるメッセージ スワッピングが有効になっているかどうかを示します。

メトリック値が true の場合は、スワッピングが有効になっていることを示します。値が false の場合は、スワッピングが有効になっていないことを示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Tibco RV Transport Enabled

サーバで TIBCO Rendezvous メッセージングが有効になっているかどうかを示します。

メトリック値が true の場合は、tibrv トランスポートと tibrvcm トランスポートのブリッジが有効になっていることを示します。値が false の場合は、Rendezvous トランスポートが有効になっていないことを示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Tibco SmartSockets Transport Enabled

サーバで TIBCO SmartSockets トランスポート プロトコルが有効になっているかどうかを示します。

メトリック値が true の場合は、SmartSockets トランスポート間のブリッジが有効になっていることを示します。 値が false の場合は、SmartSockets トランスポートが有効になっていないことを示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

URL

サーバ インスタンスの URL を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Version

EMS サーバ インスタンスのバージョン番号を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

TIBCO EMS サーバ プロセスについては、以下の Status メトリックが用意されています。

Asynchronous DB Size

非同期ストア ファイルの現在のサイズ（KB 単位）を示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Backup Server State

サーバの現在の状態を実行中または停止中として示すテキスト文字列を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Backup Server State Value

サーバの現在の状態を示す数値を示します。

値が 0（ゼロ）の場合は、バックアップ サーバが実行中であることを示します。値が 1 の場合は、バックアップ サーバが停止していることを示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Connection Count

サーバとのアクティブな接続の数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Consumers Count

サーバ上のメッセージ コンシューマの総数を示します。このメトリックには、すべてのサブスクライバとキュー レシーバが含まれます。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Durable Subscriber Count

サーバ上の恒久サブスクライバの数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Inbound Byte Rate

サーバでインバウンド メッセージが受信されるレート（1 秒あたりのバイト数（bps））を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Inbound Message Rate

サーバで受信された 1 秒あたりのインバウンド メッセージの数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Log File Size

ログ ファイルの現在のサイズ（KB 単位）を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Message Memory Used

サーバ上でメッセージを格納するために現在使用されているメモリ（バイト単位）を示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Outbound Byte Rate

サーバからアウトバウンド メッセージが送信されるレート（1 秒あたりのバイト数（bps））を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Outbound Message Rate

サーバの 1 秒あたりのアウトバウンド メッセージの数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Pending Message Count

サーバの保留中メッセージの総数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Pending Message Size

サーバの保留中メッセージの合計サイズ（KB 単位）を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Producers Count

サーバ上のメッセージ プロデューサの総数を示します。

このメトリックには、すべてのトピック パブリッシャとキュー センダが含まれます。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Queue Count

サーバ上の静的キュー、動的キュー、一時キューを含むキューの総数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Route Recover Count

サーバ上に格納できるルート リカバリ メッセージの総数を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Route Recover Interval (sec)

サーバ上のルート リカバリ メッセージをチェックする間隔（秒単位）を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Sessions Count

クライアント アプリケーションによって作成されたサーバ上のセッションの総数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Server State

サーバの現在の状態をアクティブまたはスタンバイとして示すテキスト文字列を示します。エージェントが EMS サーバに接続できない場合、サーバ状態は不明として表示されます。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Server State Value

サーバの現在の状態を示す数値を示します。

値が 0（ゼロ）の場合は、サーバがアクティブであることを示します。値が 1 の場合は、サーバがスタンバイ状態であることを示します。 値が 2 の場合は、状態が不明であることを示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Synchronous DB Size

同期ストア ファイルの現在のサイズ（KB 単位）を示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Topic Count

サーバ上の静的トピック、動的トピック、一時トピックを含むトピックの総数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

**詳細**:

プライマリおよびバックアップ サーバの監視について (P. 268)

## プライマリおよびバックアップ サーバの監視について

プライマリ サーバとバックアップ サーバのペアを監視する場合は、同じユーザ名とパスワードの設定を使用するようにプライマリ サーバとバックアップ サーバの両方を設定する必要があります。 プライマリ サーバがオフラインのときに *EMSMonitor* エージェントがバックアップ サーバに接続できるようにするには、同じユーザ名とパスワードが必要です。

バックアップ サーバと共に設定されているプライマリ サーバがオフラインになっても、Server State メトリックと Backup Server State メトリックは、引き続きプライマリ サーバとバックアップ サーバの両方のステータスをレポートします。 しかし、Backup Server Name メトリックはプライマリ サーバについてのみレポートします。 Server Status メトリックはペアになっているサーバの状態によってアクティブからスタンバイに切り替わることができますが、Backup Server Name は一度に 1 つのサーバについてのみレポートします。

## トピックに関するメトリック

TIBCO EMS の動的トピックと静的トピックについては、以下の Configuration メトリックが用意されています。

Bridged Topic

トピックにブリッジ ターゲットがあるかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、ブリッジ ターゲットがトピックに存在することを示します。
- 値が false の場合は、ブリッジ ターゲットがトピックに存在しないことを示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Expiration (Sec)

選択したトピックでメッセージが失効するまでの最長時間を示します。このメトリックの値がゼロ（0）である場合、メッセージは期限切れになりません。

キューに対してサーバの有効期限のプロパティが設定されている場合、このプロパティの値はメッセージ プロデューサが設定した JMSExpiration 値より優先されます。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Failsafe

トピックがフェールセーフ ターゲットとして設定されているかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、トピックがフェールセーフ ターゲットとして設定されていることを示します。
- メトリック値が false の場合は、トピックがフェールセーフ ターゲットとして設定されていないことを示します。

**注**: このメトリックは EMS バージョン 4.4 にのみ有効です。

Global

トピックがグローバル トピックとして設定され、サーバ間でメッセージをルーティングするために使用できるかどうかを示すテキスト文字列を示します。

- メトリック値が true の場合は、トピックがグローバル トピックであることを示します。
- メトリック値が false の場合は、トピックがグローバル トピックでないことを示します。

このメトリックは、デフォルトでは監視レベルを推奨に設定したときにのみ収集されます。

Maximum Flow Control Bytes

このトピックのフロー制御を有効にするために使用される最大バイト数を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Maximum Messages

サーバが選択したトピックの保留中メッセージとして格納できる最大メッセージ数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているときのみ、このメトリックは収集されます。

Multicast Channel Name

トピックがマルチキャストに対応している場合に、トピックに関連付けられたマルチキャスト チャネルの名前を示します。

このメトリックは、デフォルトでは監視レベルを推奨に設定したときにのみ収集されます。

Multicast Enabled

トピックがマルチキャストに対応しているかどうかを示します。

- メトリック値が true の場合は、トピックのマルチキャストが有効になっていることを示します。
- メトリック値が false の場合は、マルチキャストが有効になっていないことを示します。

  注: EMS バージョン 4.4.x ではマルチキャスト メッセージングがサポートされないため、EMS 4.4.x のサーバ インスタンスのトピックではこのメトリックは常に false になります。

このメトリックは、デフォルトでは監視レベルを推奨に設定したときにのみ収集されます。

Overflow Policy

メッセージの最大数または最大サイズ（バイト単位）を超えたトピックに使用されるオーバーフロー ポリシーを示します。以下のオーバーフロー ポリシー値が有効です。

- 0 — デフォルトのオーバーフロー ポリシーを示します。 サブスクライバがメッセージの最大バイト数または最大数を超えた場合、そのサブスクライバはメッセージを受け取りません。 メッセージ プロデューサにエラーは返されません。
- 1 — discardOld オーバーフロー ポリシーを示します。サブスクライバがメッセージの最大バイト数または最大数を超えた場合は、新しいメッセージがサブスクライバに配信される前に、最も古いメッセージが破棄されます。
- 2 — rejectIncoming オーバーフロー ポリシーを示します。 サブスクライバがメッセージの最大バイト数または最大数を超えた場合は、新しいメッセージがすべて拒否され、プロデューサにエラーが返されます。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Prefetch Count

メッセージ コンシューマがトピックから取得できるメッセージの最大数を示します。

デフォルトでは、監視レベルを「recommended」に設定しているとき、このメトリックは収集されます。

Secure

トピックが受信接続を認証するように設定されているかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、受信接続が認証されることを示します。
- メトリック値が false の場合は、受信接続が認証されないことを示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Sender Name

トピックに対して sender_name プロパティが設定されているかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、sender_name プロパティが設定されていることを示します。
- メトリック値が false の場合は、sender_name プロパティが設定されていないことを示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Sender Name Enforced

トピックに対して sender_name_enforced プロパティが設定されているかどうかをテキスト文字列で示します。

- メトリック値が true の場合は、sender_name プロパティが適用されていることを示します。
- メトリック値が false の場合は、sender_name プロパティが適用されていないことを示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Store Name

永続的メッセージが保存されるストアの名前を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

**注**: このメトリックは、EMS バージョン 5.0 以降にのみ有効です。

TIBCO EMS の動的トピックと静的トピックについては、以下の Status メトリックが用意されています。

Active Durable Subscriber Count

選択したトピックの現在アクティブな恒久サブスクライバの数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Durable Subscriber Count

選択したトピックの恒久サブスクライバの総数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Inbound Byte Rate

EMS クライアントおよびルーティングされたサーバから選択したトピックに送信される 1 秒あたりのインバウンド バイト数を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Inbound Message Rate

EMS クライアントまたはルーティングされたサーバから選択したトピックに 1 秒あたりに送信されたインバウンド メッセージ数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Outbound Byte Rate

選択したトピックのコンシューマに送信されたか、ほかのサーバにルーティングされた 1 秒あたりのアウトバウンド バイト数を示します。

デフォルトでは、監視レベルを「full」に設定しているときのみ、このメトリックは収集されます。

Outbound Message Rate

選択したトピックのコンシューマに送信されたか、ほかのサーバにルーティングされた 1 秒あたりのアウトバウンド メッセージ数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Pending Message Count

選択したトピックの保留中メッセージの数を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Pending Message Size

選択したトピックのすべての保留中メッセージの合計サイズ（KB 単位）を示します。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

Subscriber Count

選択したトピックのサブスクライバの総数を示します。

このメトリックには、恒久サブスクライバと非恒久サブスクライバが含まれます。

デフォルトでは、監視レベルを「minimum」に設定しているとき、このメトリックは収集されます。

## ルートに関するメトリック

TIBCO EMS のルートはデフォルトでは監視されませんが、*TibcoEMSMonitor.properties* ファイルに *<ServerInstance>.monitor.routes* プロパティを追加することにより、ルートの監視を有効にすることができます。ルートの監視を有効にすると、TIBCO EMS のルートについて以下の **Configuration** メトリックを使用できます。

Remote Server URL

ルートが接続しているリモート サーバの URL。

Route Type

ルートがアクティブ ルートとパッシブ ルートのどちらであるかを示します。

このメトリック値は、ルートがアクティブとパッシブのどちらであるかを示します。

Zone Name

選択したルートが含まれるルーティング ゾーンの名前。

Zone Type

ルートのルーティング ゾーンのタイプが 1 ホップ（1hop）とマルチホップ（mhop）のどちらであるかを示します。

ゾーン タイプを特定できない場合、このメトリック値は不明になります。

ルートの監視を有効にすると、TIBCO EMS のルートについて以下の **Status** メトリックが表示されます。

Backlog Count

ルートのバックログのメッセージ数。

このメトリックは、EMS バージョン 5.0 以降にのみ有効です。

Backlog Size

ルートのバックログに含まれるすべてのメッセージの合計サイズ（KB 単位）。

このメトリックは、EMS バージョン 5.0 以降にのみ有効です。

Inbound Byte Rate

インバウンド メッセージがルーティングされるレート（1 秒あたりのバイト数（bps））。

Inbound Message Rate

1 秒あたりに受信されたインバウンド メッセージの数。

Is Connected

ルートが現在接続されているかどうかを示すテキスト文字列。

メトリック値が true の場合は、ルートが接続されていることを示します。 値が false の場合は、ルートが切断されていることを示します。

Is Connected Value

ルートが接続されているかどうかを示す数値。

メトリック値が 0（ゼロ）の場合は、ルートが接続されていることを示します。 値がゼロ以外の場合は、ルートが切断されていることを示します。

Outbound Byte Rate

アウトバウンド メッセージがルーティングされるレート（1 秒あたりのバイト数（bps））。

Outbound Message Rate

1 秒あたりに送信されたアウトバウンド メッセージの数。

Stalled Destinations

ルートにストールした宛先があるかどうかを示します。

メトリック値が true の場合は、ストールした宛先があることを示します。 値が false の場合は、ストールした宛先がないことを示します。

## チャネルに関するメトリック

TIBCO EMS のマルチキャスト チャネルはデフォルトでは監視されませんが、*TibcoEMSMonitor.properties* ファイルに *<ServerInstance>.monitor.channels* プロパティを追加することにより、チャネルの監視を有効にすることができます。 チャネルの監視を有効にすると、TIBCO EMS のマルチキャスト チャネル オペレーションについて以下の **Configuration** メトリックを使用できます。

Channel Interface

マルチキャスト データの送信に使用されるマルチキャスト対応の IP アドレス。

Maximum Time

サーバが送信済みのメッセージを再送信用に保持する最長時間（秒単位）。

Maximum Rate

マルチキャスト データを転送するための最大伝送レート（1 秒あたりのバイト数（bps））。

Multicast Group Address

メッセージの送信先となるマルチキャスト グループのアドレスとポート。

Multicast Time to Live

メッセージがサーバとマルチキャスト デーモンの間で通過できるネットワーク ホップの最大数。

Priority

帯域幅の割り当て時にこのチャネルに与えられた優先度。

最も高い優先度は -5 で、最も低い優先度は 5 です。

チャネルの監視を有効にすると、TIBCO EMS のマルチキャスト チャネル オペレーションについて以下の **Status** メトリックが表示されます。

Backlog Count

チャネルで送信されるのを待っているバッファ内のメッセージの数。

Backlog Size

チャネルで送信されるのを待っているバッファ内のメッセージのサイズ（バイト単位）。

Byte Rate

インバウンドおよびアウトバウンド メッセージがチャネルによって処理されるレート（1 秒あたりのバイト数（bps））。

Channel State

チャネルがアクティブか、サーバ構成で定義されているのにアクティブでないかを示すテキスト文字列。

チャネルがアクティブな場合は、［アクティブ］と表示されます。チャネルがサーバ構成で定義されているのにアクティブでない場合は、［非アクティブ］と表示されます。

Channel State Value

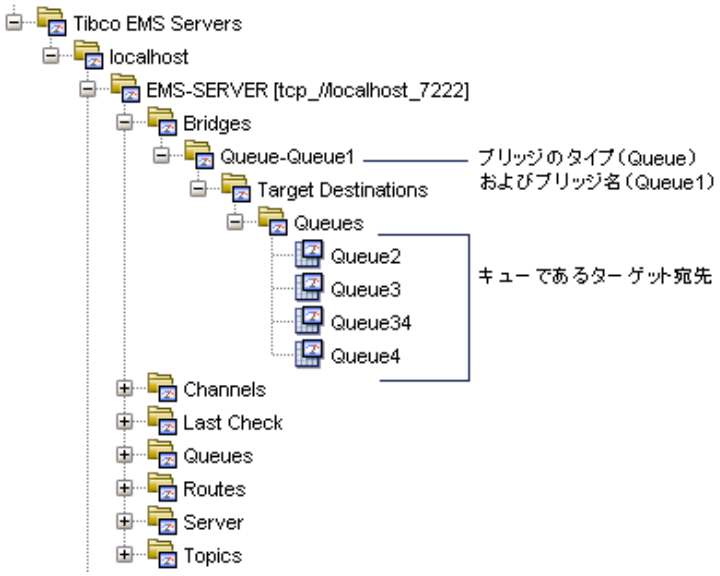
チャネルがアクティブ（ゼロ）か、サーバ構成で定義されているのにアクティブでない（ゼロ以外の値）かを示す数値。

Message Rate

チャネルによって処理される 1 秒あたりのメッセージの数。 このメトリックには、インバウンド メッセージとアウトバウンド メッセージの両方が含まれます。

## ブリッジに関するメトリック

TIBCO EMS のブリッジはデフォルトでは監視されませんが、*TibcoEMSMonitor.properties* ファイルに *<ServerInstance>.monitor.bridges* プロパティを追加することにより、ブリッジの監視を有効にすることができます。 ブリッジの監視を有効にすると、ノード名によってブリッジの種類を識別できます。また、ブリッジのソース名とそのサブノードによってターゲットの宛先を宛先の種類ごとに識別できます。- Investigator ツリーに表示されるメトリックは、ターゲットの宛先と、メッセージをフィルタするために使用されるメッセージ セレクタ（セレクタが定義されている場合）のみです。 例：




# TIBCO EMS のデフォルト メトリック グループの表示

SOA extension for TIBCO Enterprise Message Service には、デフォルトのダッシュボードとアラートを定義するために使用されるデフォルト メトリック グループが含まれています。これらのデフォルト メトリック グループをカスタム ダッシュボードやカスタム アラートで使用することもできます。

デフォルト メトリック グループは、TIBCO Enterprise Message Service 管理モジュール（*TibcoEMSManagementModule.jar*）の一部として TIBCO Enterprise Message Service 用の Enterprise Manager 拡張にパッケージ化されています。

### TIBCO Enterprise Message Service エージェントのデフォルト メトリック グループを表示する方法

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］-［Management Modules］-［Introscope SOA Extension for TibcoEMS <*version*> (*Super Domain*)］の順に展開します。
3. ［Metric Groupings］ノードを展開して、TIBCO Enterprise Message Service 管理モジュールに定義されているすべてのメトリック グループを表示します。
4. 特定のメトリック グループをクリックすると、［ビューア］ペインにその定義が表示されます。

任意のメトリック グループのデフォルト設定を変更したり、ユーザ独自のカスタム メトリック グループを作成したりできます。

**注**: メトリック グループを作成および変更する方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

## TIBCO EMS のデフォルト アラートの表示

SOA extension for TIBCO Enterprise Message Service には、事前設定済みのダッシュボードで使用されるデフォルト アラート定義が含まれています。これらのデフォルト アラートをカスタム ダッシュボードで使用することもできます。 ほとんどのデフォルト アラートは、デフォルトの警告しきい値と危険しきい値を使用して、しきい値を超えるか重要度が高くなった場合にコンソールに通知を送信するように事前設定されています。

デフォルト アラート定義は、TIBCO Enterprise Message Service 管理モジュール（*TibcoEMSManagementModule.jar*）の一部として TIBCO Enterprise Message Service 用の Enterprise Manager 拡張にパッケージ化されています。

### TIBCO Enterprise Message Service エージェントのデフォルト アラート定義を表示する方法

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］-［Management Modules］-［Introscope SOA Extension for TibcoEMS <*version*> (*Super Domain*)］の順に展開します。

3. ［Alerts］ノードを展開して、TIBCO Enterprise Message Service 管理モジュールに定義されているすべてのアラートを表示します。
4. 特定のアラートをクリックすると、［ビューア］ペインにその定義が表示されます。

特に、最も重要と思われるアラートの警告しきい値と危険しきい値のデフォルト設定を確認し、必要な場合は値を調整してください。 通知を追加したり、一部のアラートに対する処置を指定したりする必要がある場合もあります。

任意のアラートのデフォルト設定を変更したり、ユーザ独自のカスタムアラートを作成したりできます。

**注**: アラートを作成および変更する方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# エージェント構成プロパティのサマリ

*EMSMonitor* エージェントは、接続先の EMS サーバ インスタンスと監視対象のコンポーネントを定義するために *TibcoEMSMonitor.properties* ファイルの多数の構成プロパティを利用します。 以下のリストに、これらのプロパティの概要と各プロパティに設定できる値のタイプを示します。このリストは、有効な値を設定するためのクイック レファレンスとして利用できます。

一部の構成プロパティには、パターン マッチング用の正規表現を指定できます。正規表現の構文および使用法の概要については、Java ドキュメントでパターン マッチングに関する情報を参照してください。

ems.server.list

監視する EMS サーバ インスタンス名のカンマ区切りリスト。

指定する名前が監視するサーバ インスタンスの実際の名前と一致する必要はありません。 たとえば、名前が同じでポートが異なる 2 つのサーバ インスタンスがある場合は、それらを区別するために ems.server.list プロパティにエイリアスを使用できます。

例：

```
ems.server.list=mercury01,mercury02,jupiter03
```

**<*ServerInstance*>.host**

指定したサーバインスタンスをホストする EMS サーバのホスト名または IP アドレス。 デフォルト値は *localhost* です。

例：

```
mercury01.host=winsrvT400
```

<*ServerInstance*>.port

指定したサーバインスタンスをホストする EMS サーバのポート番号。デフォルト値は 7222 です。

例：

```
mercury01.port=7200
```

<*ServerInstance*>.username

EMS サーバへの接続を確立するのに必要な管理者権限を持つユーザ名。このプロパティは、通常、ユーザが *emsPwdEncryptor* ユーティリティを使用してアカウントのパスワードを暗号化したときに自動的に設定されます。

デフォルト値は *admin* です。

例：

```
mercury01.username=jgarcia
```

<*ServerInstance*>.password

*<ServerInstance>.username* プロパティに指定したユーザ アカウントのパスワード。このプロパティは、通常、ユーザが *emsPwdEncryptor* ユーティリティを使用してアカウントのパスワードを暗号化したときに自動的に設定されます。

デフォルト値は空の文字列（パスワードなし）です。

例：

```
mercury01.password=YCLhqcwQfpc=
```

<*ServerInstance*>.delaytime

監視している EMS サーバ コンポーネントに関するメトリックを更新するために EMS サーバに対してクエリを実行する間隔（秒単位）。

*EMSMonitor* エージェントは、サーバ インスタンスとそのコンポーネントのステータスに関するメトリックをこの間隔で収集します。

デフォルト値は 60 秒です。

例：

```
mercury01.delaytime=90
```

<*ServerInstance*>.report.static.freq

静的なメトリックに関するクエリの間で実行される EMS サーバ ステータス クエリの数。たとえば、このプロパティを 20 に設定すると、*EMSMonitor* エージェントがメトリックを 20 回収集したときにのみ、静的な構成に関するメトリックが更新されます。

デフォルト値は 20 です。

例：

```
mercury01.report.static.freq=50
```

<*ServerInstance*>.queue.filter.includeonly.regex

名前が式と一致するキューを監視対象にするための正規表現を指定します。任意の有効な正規表現を使用できます。一部の特殊文字にはエスケープ シーケンスが必要です。

このプロパティでフィルタを指定しなかった場合、*EMSMonitor* エージェントはデフォルトですべてのキューに関するメトリックを収集します。

例：

```
mercury01.queue.filter.includeonly.regex=[A-H]
```

<*ServerInstance*>.topic.filter.includeonly.regex

名前が式と一致するトピックを監視対象にするための正規表現を指定します。任意の有効な正規表現を使用できます。一部の特殊文字にはエスケープ シーケンスが必要です。

このプロパティでフィルタを指定しなかった場合、*EMSMonitor* エージェントはデフォルトですべてのトピックに関するメトリックを収集します。

例：

```
mercury01.topic.filter.includeonly.regex=[a-hA-H]
```

<*ServerInstance*>.queue.filter.include.dynamic

この EMS サーバ インスタンスの動的キューを監視するかどうかを指定します。 動的キューに関するメトリックを含める場合は、このプロパティを *true* に設定します。

デフォルトでは、静的キューのみが *EMSMonitor* エージェントによって監視されます。

例：

```
mercury01.queue.filter.include.dynamic=true
```

<*ServerInstance*>.topic.filter.include.dynamic

この EMS サーバ インスタンスの動的トピックを監視するかどうかを指定します。 動的トピックに関するメトリックを含める場合は、このプロパティを *true* に設定します。

デフォルトでは、静的トピックのみが *EMSMonitor* エージェントによって監視されます。

例：

```
mercury01.topic.filter.include.dynamic=true
```

<*ServerInstance*>.monitor.bridges

この EMS サーバ インスタンスのブリッジを監視するかどうかを指定します。 ブリッジに関するメトリックを含める場合は、このプロパティを *true* に設定します。

デフォルトでは、*EMSMonitor* エージェントはブリッジを監視しません。

例：

```
mercury01.monitor.bridges=true
```

<*ServerInstance*>.monitor.channels

この EMS サーバ インスタンスのマルチキャスト チャネルを監視するかどうかを指定します。チャネルに関するメトリックを含める場合は、このプロパティを *true* に設定します。

デフォルトでは、*EMSMonitor* エージェントはチャネルを監視しません。

例：

```
mercury01.monitor.channels=true
```

<*ServerInstance*>.monitor.routes

この EMS サーバ インスタンスのルートを監視するかどうかを指定します。ルートに関するメトリックを含める場合は、このプロパティを *true* に設定します。

デフォルトでは、*EMSMonitor* エージェントはルートを監視しません。

例：

```
mercury01.monitor.routes=true
```

<*ServerInstance*>.bridge.filter.includeonly.regex

名前が式と一致するブリッジを監視対象にするための正規表現を指定します。任意の有効な正規表現を使用できます。一部の特殊文字にはエスケープ シーケンスが必要です。

*<ServerInstance>.monitor.bridges* プロパティを *true* に設定し、このプロパティでフィルタを指定しなかった場合、*EMSMonitor* エージェントはデフォルトですべてのブリッジに関するメトリックを収集します。

例：

```
mercury01.bridge.filter.includeonly.regex=test.*
```

<*ServerInstance*>.channel.filter.includeonly.regex

名前が式と一致するチャネルを監視対象にするための正規表現を指定します。任意の有効な正規表現を使用できます。一部の特殊文字にはエスケープ シーケンスが必要です。

*<ServerInstance>.monitor.channels* プロパティを *true* に設定し、このプロパティでフィルタを指定しなかった場合、*EMSMonitor* エージェントはデフォルトですべてのチャネルに関するメトリックを収集します。

例：

```
mercury01.channel.filter.includeonly.regex=test.*
```

<*ServerInstance*>.route.filter.includeonly.regex

名前が式と一致するルートを監視対象にするための正規表現を指定します。任意の有効な正規表現を使用できます。一部の特殊文字にはエスケープ シーケンスが必要です。

*<ServerInstance>.monitor.routes* プロパティを *true* に設定し、このプロパティでフィルタを指定しなかった場合、*EMSMonitor* エージェントはデフォルトですべてのルートに関するメトリックを収集します。

例：

```
mercury01.route.filter.includeonly.regex=test.*
```

<*ServerInstance*>.monitoring.level

EMS サーバ インスタンスの監視レベルを定義します。このプロパティの有効な設定値は *minimum*、*recommended*、および *full* です。

サーバのデフォルト監視レベルは *recommended* です。

例：

```
mercury01.monitoring.level=minimum
```

<*ServerInstance*>.queue.monitoring.level

EMS サーバ インスタンス上のキューの監視レベルを定義します。このプロパティの有効な設定値は *minimum*、*recommended*、および *full* です。

キューのデフォルト監視レベルは *recommended* です。

例：

```
mercury01.queue.monitoring.level=recommended
```

<*ServerInstance*>.topic.monitoring.level

EMS サーバ インスタンス上のトピックの監視レベルを定義します。このプロパティの有効な設定値は *minimum*、*recommended*、および *full* です。

トピックのデフォルト監視レベルは *recommended* です。

例：

```
mercury01.topic.monitoring.level=full
```

client.identity

EMS サーバが *EMSMonitor* エージェントの ID を確認するために使用できる証明書へのパスを指定します。 ほとんどの場合、このプロパティは *emsPwdEncryptor* プログラムを実行したときに自動的に設定されます。

例：

```
client.identity=C:/TibcoEMSMonitor/certs/client.p12
```

ssl.password

クライアントのセキュリティ ライセンス用の暗号化されたパスワードを指定します。

ほとんどの場合、このプロパティは *emsPwdEncryptor* プログラムを実行したときに自動的に設定されます。

<*ServerInstance*>.ssl.connection

SSL（Secure Socket Layer）プロトコルを使用してサーバ インスタンスに接続するかどうかを指定します。

SSL を使用してサーバ インスタンスに接続する場合は、このプロパティを *enable* に設定します。セキュリティ保護されない通信を許可する場合は、このプロパティを *disable* に設定します。

セキュリティ関連のプロパティにはデフォルト値はありません。

<*ServerInstance*>.verify.host

*EMSMonitor* エージェントが EMS サーバの証明書を確認する必要があるかどうかを指定します。 EMS サーバのセキュリティ ライセンスを *trusted.certificates* プロパティで定義されたリストと照らし合わせて確認するようにエージェントに求めるには、このプロパティを *true* に設定します。

セキュリティ関連のプロパティにはデフォルト値はありません。

たとえば、エージェントが EMS サーバ インスタンス *mercury01* に接続するときに確認を求めるには、以下のように指定します。

```
mercury01.verify.host=true
```

trusted.certificates

*EMSMonitor* エージェントがサーバの証明書を確認するために使用する信頼された証明書のカンマ区切りリストを指定します。 このプロパティは、*verify.host* プロパティを *true* に設定したときに必要であり、SSL を使用するすべての EMS サーバ インスタンスに適用されます。

セキュリティ関連のプロパティにはデフォルト値はありません。

<*ServerInstance*>.verify.hostname

エージェントがサーバの証明書の CN（Common Name、共通名）フィールドを確認するかどうかを指定します。

接続されたホストの名前または *<ServerInstance>.expected.name* プロパティに指定された名前を、サーバの証明書内の CN（Common Name、共通名）フィールドと比較するようにエージェントに求めるには、このプロパティを *true* に設定します。 true に設定し、名前が一致しない場合、エージェントは接続を拒否します。

セキュリティ関連のプロパティにはデフォルト値はありません。

<*ServerInstance*>.expected.hostname

*EMSMonitor* エージェントがサーバの証明書の CN（Common Name、共通名）フィールドに存在すると想定している名前を指定します。

セキュリティ関連のプロパティにはデフォルト値はありません。

cipher.suites

*EMSMonitor* エージェントが SSL 対応の EMS サーバとの通信を暗号化するために使用できる暗号スイートのカンマ区切りリストを指定します。- *EMSMonitor* エージェントは、監視対象の EMS サーバがサポートする任意の暗号化パッケージを使用できます。 このプロパティを設定すると、すべての SSL 対応の EMS サーバ インスタンスに適用されます。

セキュリティ関連のプロパティにはデフォルト値はありません。

# 第 10 章: webMethods Broker の監視

Software AG の webMethods Broker は、webMethods Integration Server からドキュメントを個別に発行するための非同期処理とメッセージ処理のサービスを提供します。 webMethods Broker を使用すると、保証配信によってドキュメントが発行されるため、ドキュメントが宛先に届いたことが確認できます。 保証配信ドキュメントを宛先に配信できない場合、それらのドキュメントは決して発行されません。 SOA extension for webMethods Broker を使用すると、主要な webMethods Broker コンポーネントを監視できます。これには、Broker サーバ、個々の Broker インスタンス、Broker クライアント、およびドキュメント処理イベントが含まれます。

このセクションでは、webMethods Broker 環境のパフォーマンス、可用性、および全般的な稼働状況の監視と分析に使用できる Broker 専用のダッシュボード、メトリック、およびアラートについて説明します。

このセクションには、以下のトピックが含まれています。



## webMethods Broker について

webMethods Broker は、アプリケーション間のドキュメントのルーティングを管理し、組織内および組織間に存在するビジネス プロセス、エンタープライズ システムとレガシー システム、データベースとその他のバックエンド システム、内部と外部のワークフローや Web サービスを結ぶリンクとして機能します。 webMethods Broker は、多種多様なプラットフォームとプロトコルにまたがる大量のメッセージングに対応するため、サービス指向アーキテクチャの内部でメッセージのルーティング、キューイング、保存、およびフィルタリングを処理します。-

webMethods Broker は、クライアント プログラム間のドキュメントの交換を調整します。 以下のサービスが提供されます。

- Broker クライアントによって発行されるドキュメントに関するすべてのドキュメント イベントをキューに配置します。
- ドキュメントを購読し、そのイベントを受信する準備ができているクライアントにドキュメントを送信します。
- クライアントが受信するドキュメントの配信の信頼性を確認します。
- Broker クライアントが受信するドキュメントを内容に基づいて選択的にフィルタできるフィルタ サービスを提供します。
- 各グループに含まれるクライアントのドキュメント タイプ、クライアント グループ、発行許可、サブスクリプション許可、ネットワーク アクセス制御、キュー特性に関する情報、および作成された各 Broker クライアントの状態情報を保持します。

信頼性と安全性の高いメッセージ配信はビジネス サービスの提供とビジネス トランザクションの完了にとって重要な部分であるため、SOA extension for webMethods Broker には webMethods Broker のオペレーションとパフォーマンスのメトリックを監視するためのスタンドアロン エージェントが用意されています。

webMethods Broker の監視に使用されるスタンドアロン エージェントは、コア Java または .NET エージェントに対する拡張機能ではありません。 このエージェント（*WilyWMBrokerMonitor*）は、Java クラスをインスツルメントする代わりに、WmBrokerClient API を使用して webMethods Broker のコンポーネントとオペレーションのパフォーマンス情報を収集し、その情報を Enterprise Manager にレポートします。

*WilyWMBrokerMonitor* エージェントは、スタンドアロンのエージェントであるため、別個のソフトウェア パッケージとして配布され、独自の構成手順を必要とします。 たとえば、監視する Broker サーバの接続情報を設定したり、メトリックを収集するコンポーネントをカスタマイズするためにフィルタを追加したりする必要があります。

適切なエージェント プロパティを設定した後は、*WilyWMBrokerMonitor* エージェントを使用して webMethods Broker のサーバとクライアントのオペレーションを監視し始めることができます。

# SOA extension for webMethods Broker をインストールする方法

WilyWMBrokerMonitor エージェントはコア エージェントに依存しないため、そのほかの CA Introscope® コンポーネントとは無関係に WilyWMBrokerMonitor エージェントのインストールと構成を行うことができます。

SOA extension for webMethods Broker を追加するには、以下の概略手順を実行します。-

1. SOA extension for webMethods Broker の追加に対して、実装が前提条件を満たしていることを確認します (P. 291)。
2. スタンドアロン エージェント インストーラを実行 (P. 292)するか、または応答ファイルを使用 (P. 293) して、必要なエージェント ファイルをお使いの環境に追加します。
3. webMethods Broker サーバを監視するための準備を行います (P. 294)。
4. 接続と監視のプロパティを定義するために WilyWMBrokerMonitor エージェント プロファイルを設定します (P. 294)。
5. Enterprise Manager Extension for webMethods Broker を有効にします (P. 301)。

## 前提条件の確認

WebMethods Broker の SOA Extension を追加する前に、実装環境で一定の要件が満たされている必要があります。

**次の手順に従ってください**:

1. webMethods Broker がインストールされていることを確認します。

   **注**: webMethods Broker の要件については、「*Compatibility Guide*」を参照してください。

2. 環境に Enterprise Manager と Workstation がインストールされていることを確認します。
3. WilyWMBrokerMonitor エージェントがデータを送信する Enterprise Manager の接続情報があることを確認します。

## スタンドアロン エージェント インストーラを実行する

スタンドアロン エージェント インストーラを使用すると、コア Java または .NET エージェントに依存しないスタンドアロン エージェントをインストールできます。 スタンドアロン エージェント インストーラでは、webMethods Broker の監視を有効にできます。 このインストーラは、エージェント関連のファイルを環境に追加し、エージェントと Enterprise Manager の接続を設定します。 スタンドアロン エージェント インストーラは、必要なファイルを抽出して適切な場所にそれらを配置します。これらのファイルは、エージェントの構成を完了するために変更できます。

**次の手順に従ってください**:

1. 運用環境に対応する適切なスタンドアロン エージェント インストーラを起動します。
2. ［開始画面］ページで、［次へ］をクリックします。
3. インストールする監視パッケージを選択して、［次へ］をクリックします。 たとえば、［SOA Extension for webMethods Broker］を選択します。

   このオプションにより、webMethods Broker の監視が有効になります。
4. インストール ディレクトリについては、［次へ］をクリックしてデフォルトの場所をそのまま使用するか、［参照］をクリックして別の場所を指定します。
5. Enterprise Manager の接続設定については、エージェントがデータを送信する宛先となる Enterprise Manager のホスト名とポート番号を指定して、［次へ］をクリックします。
6. 設定のサマリを確認し、［インストール］をクリックします。

   インストールが開始されます。
7. インストールが完了したら［完了］をクリックします。

   スタンドアロン エージェント インストーラが終了します。

## サイレント インストール用の応答ファイルを使用する

スタンドアロン エージェント インストーラを対話形式で実行したくない場合は、サンプルのスタンドアロン応答ファイルを編集して、エージェント ファイルをインストールできます。 この方法を使用すると、サイレント モードで SOA Extension for webMethods Broker を有効にできます。

**次の手順に従ってください**:

1. スタンドアロン エージェント インストーラと同じディレクトリにある SampleResponseFile.StandaloneAgentPP.txt ファイルを開きます。
2. SampleResponseFile.StandaloneAgentPP.txt ファイルを編集して、SOA extension for webMethods Broker をエージェントに追加するための shouldInstallWMBroker プロパティを true に設定します。 例：
   `shouldInstallWMBroker=true`
3. SampleResponseFile.StandaloneAgentPP.txt ファイルを保存します。
4. コマンド ラインで適切なコマンドを入力して、インストーラを起動します。

**注**: エージェントをサイレント モードでインストールする方法の詳細については、「CA APM Java Agent 実装ガイド」または「CA APM .NET Agent 実装ガイド」を参照してください。

## インストール アーカイブを手動で抽出する

スタンドアロン エージェント インストーラまたはスタンドアロン応答ファイルにアクセスできない場合は、運用環境に対応するスタンドアロン インストール アーカイブをダウンロードできます

**次の手順に従ってください**:

1. CA サポート の CA APM ソフトウェア ダウンロード セクションに移動します。
2. 運用環境にスタンドアロン インストール アーカイブをダウンロードします。
3. 運用環境の適切なコマンドを使用して、手動でアーカイブからファイルを抽出します。たとえば、UNIX コンピュータ上では以下のように tar コマンドを使用します。

   `tar -xvf IntroscopeStandaloneAgentPPInstaller<バージョン>unix.tar`

## webMethods Broker を監視するための準備

webMethods Broker を監視する前に、管理者権限を持つクライアント グループを特定し、監視に必要なライブラリがローカルで利用可能であることを確認します。

**次の手順に従ってください**:

1. WmBrokerAgent が使用する Broker クライアントの作成に使用できる admin クライアント グループがあることを確認します。
2. 以下の Broker ライブラリを使用できることを確認します。

   webMethods 7.x の場合

   ```
   wmbrokerclient.jar
   g11nutils.jar
   ```

   webMethods 8.x の場合

   ```
   wm-brokerclient.jar
   wm-g11nutils.jar
   ```

   これらのファイルは通常、以下の場所にあります。

   - *<Broker_Home>*/common/lib -- webMethods Broker 6.5 で有効。
   - *<Broker_Home>*/lib -- その他のサポートされている webMethods Broker バージョンで有効。

**注**: webMethods Broker の要件については、「*Compatibility Guide*」を参照してください。

## webMethods Broker のエージェントの設定

インストーラを実行してエージェント ファイルを環境に追加すると、WilyWMBrokerMonitor ディレクトリが作成されます。このディレクトリ内のファイルを使用して、webMethods Broker のエージェントを設定します。

**次の手順に従ってください**:

1. *<Broker_Home>*/lib ディレクトリまたは *<Broker_Home>*/common/lib ディレクトリから WilyWMBrokerMonitor/lib ディレクトリに、以下の Broker ライブラリをコピーします。
   - wm-brokerclient.jar
   - wm-g11nutils.jar

2. WilyWMBrokerMonitor ディレクトリおよびサブディレクトリのファイルを使用して、WmBrokerAgent の接続と監視のプロパティを設定します。

   lib ディレクトリ

   .jar ファイル（jline-0.9.9.jar）としてパッケージ化された必須のライブラリが格納されています。

   config ディレクトリ

   webMethods Broker のクライアントとサーバへの接続に関するパラメータを設定するために使用する WilyWMBrokerMonitor.properties ファイルが格納されています。

   Windows Service ディレクトリ

   wmBrokerAgent プロセスを Windows サービスとして登録および登録解除するために使用するファイルが格納されています。

   Agent.jar ファイル

   エージェントがメトリックを収集して Enterprise Manager に転送するために使用するクラスを提供します。

   IntroscopeAgent.profile ファイル

   Enterprise Manager への接続を設定できるようにするプロパティを提供します。

   WmBrokerAgent.jar ファイル

   エージェントが webMethods Broker のサーバとクライアントを監視するために使用するクラスを提供します。

   wmBrokerPwdEncryptor ファイル

   パスワード暗号化スクリプト（wmBrokerPwdEncryptor.bat または wmBrokerPwdEncryptor.sh）を提供します。 このスクリプトを使用すると、webMethods Broker サーバに接続するためのユーザ パスワードおよび SSL クライアント証明書パスワードを暗号化できます。

   WmBrokerAgent ファイル

   webMethods Broker の監視を開始する準備ができたときに WmBrokerAgent を実行するための起動スクリプト（WmBrokerAgent.bat または WmBrokerAgent.sh）を提供します。

## 基本的な接続プロパティの構成

WmBrokerAgent は、webMethods Broker サーバから Enterprise Manager にデータをレポートします。 以下の手順に従って、メトリックの収集およびレポートを有効にできます。

1. WmBrokerAgent がレポートを行う Enterprise Manager の接続情報を指定します。
2. エージェントがデータを収集するサーバのリストを識別します。

**次の手順に従ってください**:

1. IntroscopeAgent.profile ファイルをテキスト エディタで開き、Enterprise Manager の接続パラメータを確認します。 例：

   ```
   introscope.agent.enterprisemanager.transport.tcp.host.DEFAULT=mercury
   introscope.agent.enterprisemanager.transport.tcp.port.DEFAULT=5001
   ```

   スタンドアロン エージェント インストーラを使用してエージェント ファイルをインストールした場合、インストール時に入力した値を使用します。 エージェント ファイルを手動で抽出した場合は、正しい Enterprise Manager に接続するようにデフォルト設定を変更する必要がある可能性があります。
2. WmBrokerAgent 起動スクリプトをテキスト エディタで開きます。
3. JAVA_HOME 環境変数を適切なディレクトリに設定します。 例：

   ```
   set JAVA_HOME=C:¥webMethods7¥jvm¥win150¥jre
   ```

   **JRE バージョン 1.5 以降で有効：** webMethods Broker を監視するには、JRE バージョン 1.5 以降が必要です。
4. WilyWMBrokerMonitor.properties ファイルをテキスト エディタで開き、host プロパティを設定して、監視する webMethods Broker サーバを指定します。 例：

   ```
   wily.webmethods.broker.server.host=vepsa09
   ```
5. port プロパティを設定して、監視する webMethods Broker サーバへの接続に使用するポート番号を指定します。 例：

   ```
   wily.webmethods.broker.server.port=6890
   ```
6. clientgroup プロパティを設定して、BrokerAdminClient を作成する必要があるクライアント グループを指定します。 例：

   ```
   wily.webmethods.broker.server.clientgroup=admin
   ```

## メトリック収集のポーリング間隔の設定

WmBrokerAgent は、各 Broker サーバを一定間隔でポーリングし、最新のステータスおよび構成情報を取得します。 これらのクエリの頻度は、構成プロパティを使用して制御できます。

WmBrokerAgent がサーバに対してクエリを実行する頻度を設定するには、interval プロパティを設定して、Broker のメトリックを収集するためのポーリング間隔を設定します。たとえば 14000 ミリ秒ごとにメトリックを収集してレポートするには、以下のように指定します。

```
wily.webmethods.broker.interval=14000
```

## 監視する Broker クライアント グループの設定

wily.webmethods.broker.clientstat.clientGroups 設定プロパティを使用して、Broker 上で監視する特定のクライアント グループを制御できます。

監視する Broker クライアント グループを設定するには、監視する Broker クライアント グループのリストを指定します。 デフォルトでは、指定したクライアント グループに含まれるすべてのクライアントに関するメトリックがレポートされます。 たとえば、IntegrationServer および IS-Backup クライアント グループに含まれるすべての Broker クライアントに関するメトリックを収集してレポートするには、以下のようにプロパティを設定します。

```
wily.webmethods.broker.clientstat.clientGroups=IntegrationServer,IS-Backup
```

## webMethods Broker 用の SSL 接続の設定

クライアント グループと webMethods Broker サーバ間で SSL （Secure Socket Layer）接続を使用する場合、エージェントは検証用の署名証明を提示する必要があります。 SSL 接続のサポートを有効にするには、SSL 証明書用の暗号化されたパスワードを作成し、WilyWMBrokerMonitor.properties ファイルの追加プロパティを設定します。

**注**: webMethods Broker サーバのサポートされているバージョン（*6.5 以外*）では、この手順を使用します。 サポートされているバージョンの詳細については、「*Compatibility Guide*」を参照してください。

wmBrokerPwdEncryptor スクリプトを使用して、使用する適切な証明書ファイルへのパスを指定し、SSL 接続用の暗号化されたパスワードを作成できます。

次の手順に従ってください:

1. wmBrokerPwdEncryptor スクリプトをテキスト エディタで開き、JAVA_HOME 環境変数を適切なディレクトリに設定します。 例：
   ```
   set JAVA_HOME=C:¥webMethods7¥jvm¥win150¥jre
   ```
2. wmBrokerPwdEncryptor スクリプトを実行し、SSL 接続プロパティを設定するために **y** を入力します。
   a. キーストア ファイルへのパスの入力を求めるプロンプトが表示されたら、Broker の PKCS12 キーストア証明書ファイルへのパスを入力します。
   b. トラストストア ファイルへのパスの入力を求めるプロンプトが表示されたら、Broker の JKS （Java Key Store）トラストストア証明書ファイルへのパスを入力します。
   c. キーストアまたは証明書ファイルのパスワードの入力を求めるプロンプトが表示されたら、そのファイルの認証に使用するパスワードを入力します。
   d. Enter キーを押して構成を完了します。

   wmBrokerPwdEncryptor スクリプトによって以下のアクションが実行されます。
   - SSL キーストアまたは証明書パスワードの暗号化
   - WilyWMBrokerMonitor.properties ファイル内の Broker の適切なプロパティを更新します。
3. WilyWMBrokerMonitor.Properties ファイルをテキスト エディタで開きます。
   a. wily.webmethods.broker.connection.ssl_encrypted プロパティを true に設定して、SSL 接続の暗号化を有効にします。 例：
      ```
      wily.webmethods.broker.connection.ssl_encrypted=true
      ```
   b. サーバへの接続に使用する完全識別名を wily.webmethods.broker.ssl.distinguished_name プロパティに設定します。 例：
      ```
      wily.webmethods.broker.ssl.distinguished_name=cn=vespa09,dc=test,dc=org
      ```
   c. ファイルを保存して閉じます。

**詳細**:

webMethods Broker 6.5 用の SSL 接続の設定 (P. 299)

## webMethods Broker 6.5 用の SSL 接続の設定

クライアント グループと webMethods Broker サーバ間で SSL （Secure Socket Layer）接続を使用する場合、エージェントは検証用の署名証明を提示する必要があります。 SSL 接続のサポートを有効にするには、SSL 証明書用の暗号化されたパスワードを作成し、WilyWMBrokerMonitor.properties ファイルの追加プロパティを設定します。

**注**: 設定する特定のプロパティは、使用している webMethods Broker のバージョンによって異なります。 webMethods Broker 6.5 の場合は、以下の手順に従います。 webMethods Broker のサポートされているバージョンについては、「*Compatibility Guide*」を参照してください。

wmBrokerPwdEncryptor スクリプトを使用して、使用する適切な証明書ファイルへのパスを指定し、SSL 接続用の暗号化されたパスワードを作成できます。

**次の手順に従ってください**:

1. wmBrokerPwdEncryptor スクリプトをテキスト エディタで開き、JAVA_HOME 環境変数を適切なディレクトリに設定します。 例：
   `set JAVA_HOME=C:¥webMethods<version_number>¥jvm¥win150¥jre`
2. wmBrokerPwdEncryptor スクリプトを実行し、SSL 接続プロパティを設定するために **y** を入力します。
   a. キーストア ファイルへのパスの入力を求めるプロンプトが表示されたら、Enter キーを押してこのプロパティを空にします。
   b. トラストストア ファイルへのパスの入力を求めるプロンプトが表示されたら、Enter キーを押してこのプロパティを空にします。
   c. 証明書ファイルへのパスの入力を求めるプロンプトが表示されたら、ブローカの SSL 証明書ファイルへのパスを入力します。

      ブローカの証明書は、通常、PKCS10 形式です。 このファイルは、webMethods Broker Certificate Manager（awcert）を使用して生成できます。

d. キーストアまたは証明書ファイルのパスワードの入力を求めるプロンプトが表示されたら、そのファイルの認証に使用するパスワードを入力します。

e. Enter キーを押して構成を完了します。

wmBrokerPwdEncryptor スクリプトによって以下のアクションが実行されます。

- SSL キーストアまたは証明書パスワードの暗号化
- WilyWMBrokerMonitor.properties ファイル内の Broker の適切なプロパティを更新します。

3. WilyWMBrokerMonitor.Properties ファイルをテキスト エディタで開きます。

a. wily.webmethods.broker.connection.ssl_encrypted プロパティを true に設定して、SSL 接続の暗号化を有効にします。 例：

```
wily.webmethods.broker.connection.ssl_encrypted=true
```

b. サーバへの接続に使用する完全識別名を wily.webmethods.broker.ssl.distinguished_name プロパティに設定します。 例：

```
wily.webmethods.broker.ssl.distinguished_name=cn=vespa09,dc=test,dc=org
```

c. ファイルを保存して閉じます。

4. 運用環境に応じて、操作を実行します。

- Windows では、<*webMethods_Home*>¥common¥lib ディレクトリの awssl65jn.dll ファイルを WilyWMBrokerMonitor¥lib ディレクトリにコピーします。
- UNIX では、<*webMethods_Home*>/common/bin ディレクトリの libawssl65jn.so ファイルを WilyWMBrokerMonitor/lib ディレクトリにコピーします。

**詳細**:

webMethods Broker 用の SSL 接続の設定 (P. 297)

### エージェントの起動

エージェントの接続と監視のプロパティを設定した後で、エージェントを起動できます。 エージェント起動スクリプトを使用してエージェントを起動し、Broker のサーバとクライアントの監視を開始します。

webMethods Broker を監視するためにエージェントを起動するには、WmBrokerAgent 起動スクリプトを実行します。

webMethods Broker サーバの監視が開始されます。

## Enterprise Manager Extension for webMethods Broker を有効にする

Enterprise Manager をインストールすると、CA APM for webMethods Broker のファイルが *<EM_Home>*/examples ディレクトリにデフォルトでインストールされます。 CA APM for webMethods Broker を有効にするには、ファイルを examples ディレクトリから Enterprise Manager のホーム ディレクトリにコピーまたは移動します。

**次の手順に従ってください**:

1. SOAExtensionForWebMethodsBroker ディレクトリが *<EM_Home>*/examples ディレクトリに存在することを確認します。
2. ファイルを *<EM_Home>*/examples/SOAExtensionForWebMethodsBroker ディレクトリから Enterprise Manager ディレクトリ構造内の対応する場所にコピーします。 たとえば、*<EM_Home>*/examples/SOAExtensionForWebMethodsBroker/ext ディレクトリからファイルを *<EM_Home>*/ext ディレクトリにコピーします。
3. Enterprise Manager がクラスタ環境内のコレクタである場合は、*<EM_Home>*/config/modules ディレクトリから webMethods Broker 管理モジュール（WebMethodsBrokerManagementModule.jar）を削除します。

   **注**: この管理モジュールは、MOM コンピュータとして使用している Enterprise Manager の *<EM_Home>*/config/modules ディレクトリにのみコピーする必要があります。 他のすべてのファイルとスクリプトは、コレクタ Enterprise Manager と MOM Enterprise Manager の両方にインストールする必要があります。
4. Workstation を再起動すると、SOA extension for webMethods Broker 専用のダッシュボードと［概要］タブがロードされます。

# ダッシュボードを使用して webMethods Broker を監視する

SOA extension for webMethods Broker には、アプリケーション環境の全般的な稼働状況を監視するために使用できる事前設定済みのダッシュボードがいくつか含まれています。 ダッシュボードは、ユーザが問題をすばやく診断して解決できるように、展開されたエージェントからデータを集計してパフォーマンス情報を要約します。

通常、ダッシュボードは以下の機能を備えているため、環境を監視するための起点として使用されます。

- webMethods Broker サーバの主要コンポーネントの全般的な稼働状況、パフォーマンス、可用性、および現在のステータスをひとめで監視する。
- 警告または危険のしきい値を超えたことがより低レベルのメトリックによって検出されると、実運用アプリケーション環境での潜在的な問題について早期の通知を受け取る。
- パフォーマンス情報にドリルダウンして、どの Broker、クライアントグループ、クライアント、またはドキュメント タイプに配信の遅延、発行の遅延、大量のトラフィックなどが発生しているかを特定する。

事前設定済みの webMethods Broker ダッシュボードは、webMethods Broker 管理モジュール（*WebMethodsBrokerManagementModule.jar*）の一部として webMethods 用の Enterprise Manager 拡張にパッケージ化されています。

webMethods Broker 管理モジュールには、webMethods Broker のために以下の事前設定済みのダッシュボードが用意されています。

WebMethods Broker - **概要**

webMethods Broker サーバの主要なアクティビティとストレージの統計に関するトップレベルの概要。これには以下の情報が含まれます。

- 発行ドキュメントおよびキュー内ドキュメントに関するグラフとアラート インジケータ
- クライアントのキューの長さおよび取得されたドキュメントに関するグラフ
- キューの長さ、キュー サイズ、および未確認ドキュメントの数に関するアラート インジケータ

WebMethods Broker - **ブローカ**

すべての Broker インスタンスに関する要約されたステータス。これには、以下の情報を示す Broker のリストが含まれます。

- 最多クライアント
- 最多ドキュメント タイプ
- 最多発行ドキュメント
- 最多キュー内ドキュメント
- 再試行キュー内の最新ドキュメント
- 再試行キュー内の配信試行イベントの最大数
- 再試行キュー内の最多発行ドキュメント

WebMethods Broker - **クライアント**

すべての Broker クライアント グループおよびクライアントに関する要約されたステータス。これには、以下の情報のリストが含まれます。

- 最多配信ドキュメントを持つクライアント グループ
- 最多発行ドキュメントを持つクライアント グループ
- 最多配信ドキュメントを持つクライアント
- 最多発行ドキュメントを持つクライアント
- 最多キュー内ドキュメントを持つクライアント
- 最多抽出イベントを持つクライアント

このダッシュボードで［クライアント詳細］リンクをダブルクリックすると、［**WebMethodsBroker - クライアント詳細**］ダッシュボードが表示されます。このダッシュボードには、最多キュー内ドキュメントを持つクライアントのキュー サイズ、キューの長さ、スキャンの数、キューの最大長、および最多未確認ドキュメントが一覧表示されます。

WebMethods Broker - **ドキュメント**

すべての Broker ドキュメント タイプの要約されたステータス。これには、最も頻繁に送受信されたドキュメント タイプのリスト、最も頻繁に配信、発行、受信されたドキュメント タイプのリスト、およびクライアント サブスクリプションが最多のドキュメント タイプのリストが含まれます。

WebMethods Broker - Territories

すべての Broker 領域に関する要約されたステータス。これには、以下の情報のリストが含まれます。

- キュー内のドキュメントが最多の領域
- キューに配置され、転送され、受信された最多のドキュメント タイプを含む領域
- キュー サイズが最大、現在のキューの長さが最大、およびキューの最大長が最大の領域

**次の手順に従ってください**:

1. Enterprise Manager を起動します（現在実行されていない場合）。
2. Workstation を起動し、SOA extension for webMethods がインストールされている Enterprise Manager にログインします。
3. ［Workstation］-［新規コンソール］を選択します。
4. ［ダッシュボード］ドロップダウン リストから webMethods Broker のいずれかのダッシュボードを選択します。

   たとえば、webMethods Broker のメトリックの概要を表示するには、［webMethods Broker - 概要］ダッシュボードを選択します。
5. 別のタブまたはアラートをダブルクリックすると、関連するダッシュボードが開き、詳細情報が表示されます。

   たとえば、［クライアント］タブをダブルクリックすると、webMethods Broker のクライアント グループとクライアントに関する詳細情報が表示されます。
6. ダッシュボードに表示された特定の Broker、クライアント、ドキュメント タイプ、または領域のメトリックをダブルクリックすると、さらなる分析のために Investigator が表示されます。

   ［WebMethodsBroker - クライアント詳細］ダッシュボードでは、たとえば、最も高い確率でキューに配置されるクライアントをダブルクリックすると、そのクライアントの［Events］-［Queued］メトリックが選択された状態で Investigator が表示されます。

# Broker に関するメトリックの概要と表示

WmBrokerAgent は、Broker のサーバとクライアントを監視し、［Broker Server on *<host_name>* at *<port_number>*］ノードの下にそれらの全般的な稼働状況に関するデータを表示します。

以下のメトリック カテゴリのメトリックを使用して、webMethods Broker のサーバおよびクライアントのパフォーマンスと稼働状況を監視できます。

Brokers

［Brokers］ノードの下には、クライアント数、サブスクリプション数、発行ドキュメント数、配信ドキュメント数など、個々の Broker インスタンスに関するメトリックが表示されます。

クライアント グループ

［Client Groups］ノードの下には、発行ドキュメント数や配信ドキュメント数など、WilyWMBrokerMonitor.properties ファイルで監視対象として指定したクライアント グループと各グループに関連付けられたクライアントに関するメトリックが表示されます。

Document Types

［Document Types］ノードの下には、定義した個々のドキュメント タイプに関するメトリックが表示されます。 ドキュメント タイプには、パートナー間またはプログラム間で情報を交換するために使用されるドキュメントの構造と内容を定義する一連のフィールドが含まれます。

Retry Queue

［Retry Queue］ノードの下には、内部の再試行キューに関するメトリックが表示されます。

Territory Stats

［Territory Stats］ノードの下には、個々の領域に関するメトリックが表示されます。

Trace

［Trace］ノードの下には、内部の追跡キューに関するメトリックが表示されます。

Utilization

［Utilization］ノードの下には、Broker の構成とデータのストレージに関する統計を示すメトリックが表示されます。

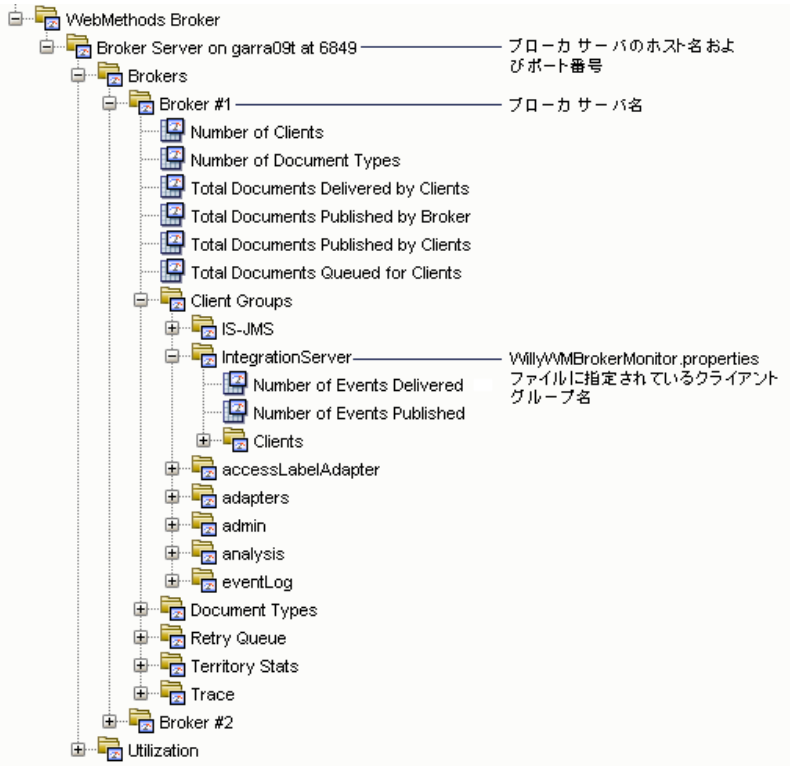
**注**: 各メトリック カテゴリに含まれるメトリックについては、そのカテゴリに対応するセクションを参照してください。

### Investigator で webMethods Broker メトリックのサマリを表示および操作する方法

1. エージェント ノードおよび［Broker Server on *<host_name>* at *<port_number>*］ノードを展開し、［Brokers］をクリックして［概要］タブを表示します。このタブには、監視対象となるすべての Broker サーバ インスタンスが、配信ドキュメント数、発行ドキュメント数、およびキュー内ドキュメント数に関するサマリ情報と共に一覧表示されます。
2. 特定の Broker 名を選択すると、そのインスタンスの配信ドキュメントと発行ドキュメントに関するサマリ情報が［概要］タブにグラフィカルな形式で表示されます。 たとえば、個々の Broker 名を選択すると、［概要］タブに以下の情報のグラフが表示されます。
   - Total Documents Delivered by Clients
   - Total Documents Published by Clients
   - Total Documents Published by Broker
   - Total Documents Queued for Clients
   - Number of Clients
   - Number of Document Types
3. 個々の Broker 名を展開し、いずれかのサブノードをクリックすると、そのメトリック カテゴリに関するサマリ情報が表示されます。たとえば、［Client Groups］ノードを選択すると、その Broker のクライアント グループのリストと、各クライアント グループの配信ドキュメントおよび発行ドキュメントのサマリが［概要］タブに表示されます。

### Investigator で webMethods Broker メトリックのノードを表示および操作する方法

1. エージェントと［Broker Server on *<host_name>* at *<port_number>*］ノードを展開し、さらに［Brokers］ノードを展開すると、監視している Broker サーバ名が表示されます。
2. 個々の［*Broker_server_name*］ノードを展開すると、その Broker サーバ インスタンスに関するメトリックが表示されます。
3. ［Client Groups］ノードを展開すると、*WilyWMBrokerMonitor.properties* ファイルで監視対象として指定したクライアント グループが表示されます。さらに、特定の［*<client_group_name>*］を展開すると、そのクライアント グループに関するメトリックが表示されます。 例：




4. ［Clients］ノードを展開し、さらに個々の［*<client_name>*］サブノードを展開すると、そのクライアントに関するメトリックが表示されます。-
5. ［Document Types］ノードを展開し、さらに個々の *<document_type_name>* サブノードを展開すると、選択した Broker のそのドキュメント タイプに関するメトリックが表示されます。-

6. ［Retry Queue］、［Territory Stats］、または［Trace］サブノードを展開すると、選択した Broker の再試行キュー、領域、または追跡の統計が表示されます。
7. ［Utilization］-［Storage Statics］を展開し、さらに［CONFIG］または［DATA］サブノードを展開すると、選択した Broker サーバの構成またはデータの保存に関する情報が表示されます。-

## Broker に関するメトリック

webMethods Broker サーバは、クライアント、Broker、および各種アプリケーションにまたがってドキュメントのフローを管理するホスト コンピュータです。 クライアント プログラムは情報をドキュメントの形式で発行および購読し、Broker サーバはドキュメントのルーティング、キューイング、およびフィルタリングを自動的に行います。 各 Broker サーバには 1 つ以上の Broker が配置されています。

個々の Broker は、クライアント接続を受信し、固有のドキュメント タイプ、クライアント キュー、およびサブスクリプションに関する情報を格納することができます。 クライアントがドキュメントを発行すると、Broker はどの Broker クライアントがそのタイプのドキュメントを受信するために購読したかを特定し、対応する Broker クライアント キューにドキュメントを配置します。

監視している Broker インスタンスについては、［Brokers］サブノードの各 Broker インスタンス名の下に以下のメトリックが表示されます。

Number of Clients

選択した Broker サーバに対応するクライアントの総数。

Number of Document Types

選択した Broker サーバのために定義されたドキュメント タイプの総数。

Total documents delivered by clients

クライアントから選択した Broker サーバに配信されたドキュメントの総数。

Total documents published by broker

選択した Broker サーバからそのクライアントに発行されたドキュメントの総数。

Total documents published by clients

クライアントから選択した Broker サーバに発行されたドキュメントの総数。

Total documents queued for clients

選択した Broker サーバからそのクライアントに配信するためにキューに配置されたドキュメントの総数。

## クライアント グループに関するメトリック

webMethods Broker では、クライアント グループを使用して、複数の Broker クライアントのプロパティをまとめて設定することができます。監視しているクライアント グループについては、［Client Groups］サブノードの各クライアント グループ名の下に以下のメトリックが表示されます。

Number of Events Delivered

選択したクライアント グループに含まれるすべてのクライアントのドキュメント配信イベントの総数。

Number of Events Published

選択したクライアント グループに含まれるすべてのクライアントのドキュメント発行イベントの総数。

## クライアントに関するメトリック

Broker クライアントは、クライアント プログラムが特定の Broker に接続するために作成して使用する webMethods オブジェクトです。クライアント プログラムは必要に応じて 1 つ以上の Broker クライアントを作成し、1 つ以上の Broker に接続します。webMethods Broker クライアントについては、ドキュメント処理タスクがイベントとして記録されます。たとえば、ドキュメントがクライアントに正常に配信されると、その処理がドキュメント配信イベントとして記録されます。

Broker クライアントは、［Clients］-［<*client_name*>］ノードの下に表示される以下のメトリック カテゴリのメトリックを使用して監視できます。

Events

イベントは、ドキュメント処理タスクを記録します。たとえば、ドキュメントがクライアントに正常に配信されると、その処理がドキュメント配信イベントとして記録されます。

Queue

キューは、非同期イベントを処理できるようになるまで受信した順に格納するために使用されます。

Session

セッションは、Broker へのクライアント接続を表します。

## イベントのメトリック

選択した Broker クライアントの各クライアントの［Events］サブノードの下には、以下のメトリックが表示されます。-

Delivered

選択したクライアントによって記録されたドキュメント配信イベントの総数。

Published

選択したクライアントによって記録されたドキュメント発行イベントの総数。

Queued

選択したクライアントのキュー内ドキュメントの総数。

Retrieved

選択したクライアントがキューから取得したドキュメントの総数。

Unacknowledged

クライアント プログラムがクライアント キューから取得したが、Broker がまだ確認していないドキュメントの総数。

Broker は、確認応答を受信した後、クライアント キューからドキュメントを削除します。

### キューのメトリック

選択した Broker クライアントの各クライアントの［Queue］サブノードの下には、以下のメトリックが表示されます。-

Byte Size

キューのサイズ（バイト単位）。

Highest Length

記録されたキューの長さの最大値。

Length

クライアント キューに現在含まれているドキュメントの総数。これには、クライアント プログラムがまだ取得していないドキュメントと、クライアント プログラムが取得してまだ確認していないドキュメントが含まれます。

### セッションのメトリック

選択した Broker クライアントの各クライアントの［Session］サブノードの下には、以下のメトリックが表示されます。-

Last Connect

最後に確立されたクライアント セッションのセッション識別子。

Last Disconnect

最後に Broker から切断されたクライアント セッションのセッション識別子。

Last Retrieved

このクライアント キューから最後にドキュメントを取得したクライアント セッションのセッション識別子。

## ドキュメント タイプに関するメトリック

ドキュメントは、ネットワーク上で Broker を介してパブリッシャからサブスクライバに伝送されるメッセージです。 個々のドキュメントは、何らかのドキュメント タイプのインスタンスです。 各ドキュメント タイプには、一意の名前と関連するプロパティがあります。プロパティには、ドキュメント フォルダ名、作成日時、Broker クライアントによる発行回数と取得回数、サブスクリプション数などがあります。

監視している Broker インスタンスについては、［Document Types］サブノードの各ドキュメント タイプの下に以下のメトリックが表示されます。

Number of Client Subscriptions

選択した Broker のクライアントによるこのドキュメント タイプへのサブスクリプションの総数。

Number of Events Delivered

選択した Broker サーバから配信された、選択したドキュメント タイプのドキュメントの総数。

Number of Events Published

選択した Broker サーバに発行された、選択したドキュメント タイプのドキュメントの総数。

Number of Forwards Received

同じ領域の別の Broker から転送された、選択したドキュメント タイプのドキュメントの総数。

Number of Groups Can Publish

選択したドキュメント タイプを発行できるクライアント グループの総数。

Number of Groups Can Subscribe

選択したドキュメント タイプを購読できるクライアント グループの総数。

## 再試行キューに関するメトリック

［webMethods Broker］の［Retry Queue］には、Broker が再試行する必要がある要求を格納するために使用する内部キューに関する統計が表示されます。 これらの統計は、通常、サポート担当者がトラブルシューティングのために使用します。

監視している Broker インスタンスについては、［Retry Queue］サブノードの下に以下のメトリックが表示されます。

Current Events

現在再試行キューにある配信予定のドキュメントの数。

Current Publishes

現在再試行キューにある発行予定のドキュメントの数。

Maximum Events

再試行キューに保持できる配信予定のドキュメントの最大数。

Maximum Publishes

再試行キューに保持できる発行予定のドキュメントの最大数。

Next Operation Sequence Number

次に再試行する要求を識別するシーケンス番号。

Number of Attempts

Broker が再試行キューからドキュメントの配信を試行する回数。

Reserved Guaranteed Events

保証配信の対象としてマークされ、再試行キューの領域が予約されたドキュメントの数。

Reserved Total Publishes

再試行キューの領域が予約された発行予定のドキュメントの総数。

Reserved Volatile Events

再試行キューの領域が予約された配信予定の一時ドキュメントの数。

Reserved Volatile Publishes

再試行キューの領域が予約された発行予定の一時ドキュメントの数。

## 領域統計に関するメトリック

webMethods Broker の領域統計は、領域ごとの情報のイベントとキューを提供します。 これらの統計は、各領域のキューに配置され、転送され、受信されたイベントの数、キュー サイズ、およびキューの長さを示します。

監視している Broker インスタンスについては、［Territory Stats］-［<*territory_name*>］-［Events］サブノードの下に以下のメトリックが表示されます。

Number Enqueued

この領域でキューに配置されたドキュメント配信イベントの数。

Number Forwarded

この領域で転送されたドキュメント配信イベントの数。

Number Received

この領域で受信されたドキュメント配信イベントの数。

監視している Broker インスタンスについては、［Territory Stats］-［<*territory_name*>］-［Queue］サブノードの下に以下のメトリックが表示されます。

Byte Size

選択した領域のキューの合計サイズ（バイト単位）。

Highest Length

この領域で記録されたキューの長さの最大値。

Length

この領域のキューに現在含まれているドキュメントの総数。これには、クライアント プログラムがまだ取得していないドキュメントと、クライアント プログラムが取得してまだ確認していないドキュメントが含まれます。

## 追跡キューに関するメトリック

webMethods Broker の追跡キューは、追跡イベントを格納する一時的なストレージ領域です。 追跡イベントは、Integration Server 上の発行トリガまたは確認応答トリガに対応して生成されます。 追跡キューの統計は、内部キューのアクティビティ イベントおよび追跡イベントであるため、通常、サポート担当者がトラブルシューティングのために使用します。 監視している Broker インスタンスについては、［Trace］サブノードの下に以下のメトリックが表示されます。

Number of Events Queued

選択した Broker の追跡キューに含まれるドキュメントの総数。

Queue Byte Size

選択した Broker の追跡キューの合計サイズ（バイト単位）。

Queue Length

選択した Broker の追跡キューに含まれる項目の数。

## 使用率に関するメトリック

Broker サーバの構成（CONFIG）データと実行時（DATA）データのストレージについては、［Utilization］-［Storage Statistics］サブノードの下に以下のメトリックが表示されます。

Current KB Reserved

選択したストレージ ファイルのために予約されたサイズ（KB 単位）。

Current KB in Use

選択したストレージ ファイルの現在のサイズ（KB 単位）。

Maximum KB Available

選択したストレージ ファイルを拡張できる最大サイズ（KB 単位）。

Maximum Transaction (KB)

Broker サーバの最大トランザクション サイズ（KB 単位）。

Session URL

ストレージ機能の構成の場所とタイプ。

［Utilization］-［Storage Statistics］の下のメトリックは、CONFIG ストレージと DATA ストレージに分かれています。 CONFIG メトリックは、Broker サーバ、領域、Broker、ドキュメント タイプ、および統計を定義するのに必要なデータ ストレージを示します。 DATA メトリックは、クライアント キュー、ドキュメント、およびログに必要なデータ ストレージを示します。 Broker サーバの個々の構成およびデータ ストレージ ファイル名についても、［File Statistics］-［<*file_name*>］サブノードの下に同様のメトリックが表示されます。-

# Broker のデフォルト メトリック グループを表示する

SOA extension for webMethods には、デフォルトのダッシュボードとアラートを定義するために使用されるデフォルト メトリック グループが含まれています。これらのデフォルト メトリック グループをカスタム ダッシュボードやカスタム アラートで使用することもできます。

デフォルト メトリック グループは、webMethods Broker 管理モジュール（*WebMethodsBrokerManagementModule.jar*）の一部として webMethods 用の Enterprise Manager 拡張にパッケージ化されています。

デフォルト メトリック グループは、CA Introscope® Workstation の管理モジュール エディタを使用して表示できます。また、webMethods Broker 管理モジュールを拡張してカスタム メトリック グループを含めたり、カスタム ダッシュボードやカスタム アラートでデフォルト メトリック グループを使用したりできます。

**webMethods Broker エージェントのデフォルト メトリック グループを表示する方法**

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］-［Management Modules］-［WebMethodsBrokerManagementModule (*Super Domain*)］の順に展開します。
3. ［メトリック グループ］ノードを展開すると、webMethods 管理モジュールに定義されたすべてのメトリック グループが表示されます。
4. 特定のメトリック グループをクリックすると、［ビューア］ペインにその定義が表示されます。

# Broker のデフォルト アラートの表示

SOA extension for webMethods には、事前設定済みのダッシュボードで使用されるデフォルト アラート定義が含まれています。 これらのデフォルト アラートをカスタム ダッシュボードで使用することもできます。 ほとんどのデフォルト アラートは、デフォルトの警告しきい値と危険しきい値を使用して、しきい値を超えるか重要度が高くなった場合にコンソールに通知を送信するように事前設定されています。

デフォルト アラート定義は、webMethods Broker 管理モジュール（*WebMethodsBrokerManagementModule.jar*）の一部として webMethods Broker 用の Enterprise Manager 拡張にパッケージ化されています。

デフォルト アラート定義は、CA Introscope® Workstation の管理モジュール エディタを使用して表示できます。 また、webMethods Broker 管理モジュールを拡張してカスタム アラート定義やカスタム通知タイプを含めたり、カスタム ダッシュボードでデフォルト アラート定義を使用したりできます。

**webMethods Broker エージェントのデフォルト アラート定義を表示する方法**

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］-［Management Modules］-［WebMethodsBrokerManagementModule (*Super Domain*)］の順に展開します。
3. ［Alerts］ノードを展開すると、webMethods Broker 管理モジュールに定義されているすべてのアラートが表示されます。
4. 特定のアラートをクリックすると、［ビューア］ペインにその定義が表示されます。

   特に、警告しきい値と危険しきい値のデフォルト設定を確認し、必要な場合は値を調整し、通知や修正処置を追加してください。

# 第 11 章: webMethods Integration Server の監視

Software AG の webMethods 製品スイートは、各組織がビジネス プロセスと Web サービスを作成、調整、統合できるようにするために、複数のインフラストラクチャ コンポーネントと機能で構成された SOA プラットフォームです。 SOA extension for webMethods を使用すると、webMethods インフラストラクチャの多くの主要な要素を監視できます。

このセクションでは、webMethods サービスについて説明し、webMethods Integration Server の稼働状況とオペレーションを監視して分析するために用意された関連するダッシュボードとメトリックについて説明します。

このセクションには、以下のトピックが含まれています。



## webMethods Integration Server について

webMethods Integration Server は、企業が新しいビジネス サービスと既存のビジネス サービスを公開および統合できるようにします。 この製品には、新しいサービスの設計、テスト、展開を行い、疎結合サービスとレガシー システムを自動化し、調整し、組み合わせることによってビジネス プロセスを改善するためのツールが含まれています。

webMethods Integration Server は、サービスを実行および配布するための一元化されたプラットフォームを提供します。 クライアントの要求を受信して解釈し、要求されたサービスを識別して呼び出し、データを予期された形式で実行中のサービスに渡し、サービスによって生成された出力を受信し、その出力をクライアントに返します。

統合プラットフォームとしての webMethods は、主にアプリケーション サーバ、データベース、およびカスタム アプリケーション間でオペレーションを調整し、企業または取引相手が電子ドキュメントを交換できるようにするために使用されます。

webMethods Integration Server のオペレーションを監視するには、以下のトップレベル コンポーネントに関するメトリックを使用します。

**アダプタ**

アダプタを使用すると、サービス インターフェースの共通のアダプタ フレームワークによって外部アプリケーションと webMethods を統合できます。 アダプタは以下の要素で構成されます。

Integration Server が実行時に外部リソースまたはシステムに接続できるようにする**アダプタ接続**。-

Integration Server 上で実行され、外部リソースに対するオペレーションを開始する**アダプタ サービス**。

外部リソースを監視し、Integration Server によって開始されないイベントが発生したことを Integration Server に通知する**アダプタ通知**。

SOA extension for webMethods では、［Adapter Connection Pools］、［Adapter Services］、および［Adapter Notifications］ノードのメトリックを使用して、webMethods Integration Server のために展開したすべてのアダプタのパフォーマンスと全般的な稼働状況を監視できます。

**ビジネス プロセス**

ビジネス プロセスは、特定のビジネス ルールのセットを使用して特定の順序で実行される、一連の相互に関連するビジネス タスクです。ほとんどのビジネス プロセスでは、役割が異なる複数のシステムおよび複数のユーザどうしのやり取りが必要です。

たとえば、新入社員の受け入れ準備、発注書の処理、請求書の送付などに対応するビジネス プロセスがあるとします。その場合、各ビジネス プロセスには、新入社員に対する職場スペースの割り当て、人事（HR）システムへの従業員の追加、事務機器と備品の発注などのビジネス タスクが含まれます。

SOA extension for webMethods では、［WebMethods］-［Business Processes］ノードのメトリックを使用して、定義されているビジネス プロセスのパフォーマンスと全般的な稼働状況を監視できます。

### フロー サービス

フロー サービスは、webMethods のフロー言語で記述され、webMethods Integration Server 上に展開されるサービスです。 フロー サービスは、他のフロー サービス、ユーザ定義サービス、組み込みサービス、他のプロバイダ（webMethods アダプタや .NET プラグインなど）のサービスを含め、webMethods サーバ上で実行されているどんなサービスも呼び出すことができます。-

SOA extension for webMethods では、すべてのフロー サービスのパフォーマンスと全般的な稼働状況を監視することも、監視対象にしないフロー サービスをフィルタして除外することもできます。 フロー サービスに含まれる個々のフロー手順に関するメトリックは、［WebMethods］-［Flow Services］ノードの下に表示されます。

### Java サービス

Java サービスは、Java で記述され（または、他の言語で記述されたサービスが Java クラスを使用してラップされ）、webMethods Integration Server 上のサービスとして公開された組み込みサービスおよびユーザ定義サービスです。

SOA extension for webMethods では、すべての Java サービスのパフォーマンスと全般的な稼働状況を監視することも、監視対象にしない Java サービスをフィルタして除外することもできます。各 Java クラスに含まれる Java メソッドに関するメトリックは、［WebMethods］-［Java services］ノードの下に表示されます。

### JDBC 接続プール

webMethods Integration Server は、ネットワーク経由の通信と情報転送を行うために Java データベース接続を使用します。

SOA extension for webMethods では、［WebMethods］-［JDBC Connection Pools］ノードのメトリックを使用して JDBC 接続の可用性を監視できます。

### スレッド プール

webMethods Integration Server は、サービスの実行、webMethods Broker からのドキュメントの取得、およびトリガの実行にスレッドを使用します。

SOA extension for webMethods では、［WebMethods］-［Thread Pools］ノードのメトリックを使用してスレッドの可用性を監視できます。

### トレーディング ネットワーク

各組織は、トレーディング ネットワークを使用してドキュメントを交換することにより、企業間の関係を確立し、充実させることができます。

SOA extension for webMethods では、［WebMethods］-［Trading Networks］ノードのメトリックを使用してドキュメントの認識と処理を監視できます。

### トリガ

トリガは、発行可能なドキュメント タイプへのサブスクリプションを確立し、それらのドキュメント インスタンスの処理方法を指定します。

Broker（ローカル）トリガは、Integration Server 上でローカルに発行されたドキュメント、または Broker に配信されたドキュメントを購読して処理するトリガです。 Broker トリガは、多くの場合、非同期のアダプタ通知に関連付けられています。

JMS トリガは、JMS プロバイダ上の宛先（キューまたはトピック）からメッセージを受信し、それらのメッセージを処理するトリガです。

SOA extension for webMethods では、［WebMethods］-［Triggers］ノードのメトリックを使用してトリガを監視できます。

### Web サービス

WebServices メトリックは、クライアントとサーバのビジネス サービス エンドポイント、および各サービス内の関連するオペレーションを表します。

SOA extension for webMethods では、［WebMethods］-［WebServices］ノードを使用して、クライアントとサーバの Web サービス エンドポイントのパフォーマンスと全般的な稼働状況を監視できます。

### XSLT サービス

webMethods の内部では、XSLT スタイル シートを使用して XML データを別の形式に変換し、その変換結果を他のサービスに含めることができます。

SOA extension for webMethods では、［WebMethods］-［XSLT Services］ノードを使用して、XSLT 変換のパフォーマンスと全般的な稼働状況を監視できます。

# webMethods Integration Server の監視を有効にする方法

管理者は、以下の概略手順を実行することによって、webMethods Integration Server の監視を有効にします。

1. webMethods Integration Server がインストールされていることを確認します。

   **注**: webMethods Integration Server の要件については、「*Compatibility Guide*」を参照してください。

2. webMethods WmPRT.jar ファイルのサポートされているバージョンがあることを確認します。Web ブラウザで以下の URL を開きます。バージョン レベルが表示されることを確認します。

   `http://<Integration_Server_Hostname>:<port_number>/WmRoot/Updates.dsp`

3. エージェントと WSOA パフォーマンス管理がインストールされて有効になっていることを確認します。

4. エージェントが CA APM for webMethods Integration Server を使用できるようにエージェント プロファイルを設定します。

   **重要**: スタンドアロン エージェント インストーラまたは応答ファイルを使用して、エージェントで CA APM for webMethods Integration Server を有効にした場合は、この手順をスキップします。

5. Enterprise Manager 拡張機能を有効にします (P. 326)。

**詳細**:

webMethods Integration Server を監視するエージェントの手動での有効化 (P. 323)

## webMethods Integration Server を監視するエージェントの手動での有効化

webMethods Integration の監視を有効にするには、以下のいずれかの方法を使用します。

- エージェントをインストールする際に、CA APM for webMethods Integration Server を選択します。

  エージェント プロファイルは、デフォルト設定で自動的に設定されます。 それ以上の手順は必要ありません。

- エージェントをインストールする際に、CA APM for webMethods Integration Server を選択しないでください。以下の手順に従って、エージェント プロファイルを手動で設定します。

  **注**: webMethods Integration Server の要件については、「*Compatibility Guide*」を参照してください。

**次の手順に従ってください**:

1. デフォルトのエージェントと CA APM for SOA がインストールされて有効になっていることを確認します。
2. CA APM for webMethods Integration Server のディレクトリが <*Agent_Home*>/examples ディレクトリ内にあることを確認します。
3. <*Agent_Home*>/examples/SOAExtensionForWebMethodsIS ディレクトリのファイルを、対応する <*Agent_Home*> ディレクトリにコピーします。

   たとえば、<*Agent_Home*>/examples/SOAExtensionForWebMethodsIS/ext のファイルを、<*Agent_Home*>/core/ext ディレクトリにコピーします。
4. <*Agent_Home*>/core/config/IntroscopeAgent.profile ファイルをテキスト エディタで開きます。

   a. webmethods.pbl、デフォルトのエージェントの pbl、および spm pbl を IntroscopeAgent.profile ファイルの introscope.autoprobe.directivesFile プロパティに追加します。 例：

   ```
   introscope.autoprobe.directivesFile=default-typical.pbl,hotdeploy,spm.pbl
   ,webmethods.pbl
   ```

   **注**: webMethods Integration Server の ProbeBuilder ディレクティブ ファイル (P. 325)を使用して追跡をカスタマイズできます。

   b. introscope.agent.agentName プロパティ値を webMethods Agent に設定します。 例：

   ```
   introscope.agent.agentName=webMethods Agent
   ```

   この手順によって、webMethods データのみをダッシュボードに表示することができます。

   c. デフォルト設定を変更する場合は、IntroscopeAgent.profile ファイルでそのほかのプロパティを修正します。

d. IntroscopeAgent.profile ファイルを保存して閉じます。

5. <*Agent_Home*>/core/config/webmethods-toggles.pbd ファイルで、以下の編集を実行します（webMethods Integration Server 7.x にのみ有効）。

   a. Business Processes セクションの BProcAndStepTracing71 フラグのコメント化を解除します。

      BProcAndStepTracing71 フラグをオンにします。
      BProcAndStepTracing80 フラグをオフにします。
      BProcAndStepTracing82 フラグをオフにします。

   b. Web Services セクションの以下のオプションのコメント化を解除します。

      WM8xWebServicesTracing フラグをオフにします。
      WM7xWebServicesTracing フラグをオンにします。
      WebMethods8xWSClientTracing フラグをオフにします。
      WebMethods7xWSClientTracing フラグをオンにします。
      WebMethods65WSClientTracing フラグをオフにします。

   c. webmethods-toggles.pbd ファイルを保存して閉じます。

6. webMethods Integration Server プロセスを再起動します。

## webMethods Integration Server のディレクティブ ファイルについて

IntroscopeAgent.profile の introscope.autoprobe.directivesFile プロパティを設定するときは、webMethods Integration Server のデフォルトのインスツルメンテーションを有効にします。 その後、デフォルトの監視を変更する場合は、webmethods.pbd または webmethods-toggles.pbd の ProbeBuilder ディレクティブをカスタマイズできます。 たとえば、このトグル ファイルを使用して特定のトレーサ グループの追跡をオンまたはオフにすることで、特定のコンポーネントの監視を細かく調整できます。

webmethods.pbd

フロー サービス、Java サービス、トレーディング ネットワーク、Web サービス、XSLT サービス、およびビジネス プロセスを含む、webMethods Integration Server 上の主要なサービスを監視します。

webmethods-toggles.pbd

webMethods Integration Server コンポーネントの監視のオンとオフを切り替えます。

webmethods.pbl

webMethods Integration Server を監視するための以下の ProbeBuilder ディレクティブ ファイルのデフォルト リストを提供します。

- webmethods.pbd
- webmethods-toggles.pbd

## Enterprise Manager 拡張機能を有効にする

Enterprise Manager をインストールすると、CA APM for webMethods Integration Server のファイルが <*EM_Home*>/examples/SOAExtensionForWebMethodsIS ディレクトリにデフォルトでインストールされます。 CA APM for webMethods Integration Server を有効にするには、webMethods Integration Server 用の Enterprise Manager ファイルを <EM_Home>/examples ディレクトリから Enterprise Manager のホーム ディレクトリ内の適切な場所にコピーまたは移動する必要があります。

**注**: CA APM for webMethods Integration Server を使用するには、CA APM for SOA を Enterprise Manager 上で有効にする (P. 44)必要があります。

**次の手順に従ってください**:

1. CA APM for webMethods Integration Server のディレクトリ（SOAExtensionForWebMethodsIS）が <*EM_Home*>/examples ディレクトリ内にあることを確認し、<*EM_Home*>/examples/SOAExtensionForWebMethodsIS ディレクトリのファイルを Enterprise Manager ディレクトリ構造の対応する場所にコピーします。 たとえば、<*EM_Home*>/examples/SOAExtensionForWebMethodsIS/ext ディレクトリのファイルを <EM_Home>/ext ディレクトリにコピーします。
2. Enterprise Manager がクラスタ化環境内のコレクタである場合は、<*EM_Home*>/config/modules ディレクトリから webMethods Integration Server 管理モジュール（WebMethodsISManagementModule.jar）を削除します。

この管理モジュールは、MOM コンピュータとして使用している Enterprise Manager の <*EM_Home*>/config/modules ディレクトリにのみコピーする必要があります。 他のすべてのファイルとスクリプトは、コレクタ Enterprise Manager と MOM Enterprise Manager の両方にインストールする必要があります。

3. Workstation を再起動すると、CA APM for webMethods Integration Server 固有のダッシュボードおよび［概要］タブがロードされます。

# ダッシュボードを使用して webMethods を監視する

SOA extension for webMethods Integration Server には、アプリケーション環境の全般的な稼働状況を監視するために使用できる事前設定済みのダッシュボードがいくつか含まれています。 ダッシュボードは、ユーザが問題をすばやく診断して解決できるように、展開されたエージェントからデータを集計してパフォーマンス情報を要約します。

通常、ダッシュボードは以下の機能を備えているため、環境を監視するための起点として使用されます。

- webMethods Integration Server の主要コンポーネントの全般的な稼働状況、パフォーマンス、可用性、および現在のステータスをひとめで監視する。
- 警告または危険のしきい値を超えたことがより低レベルのメトリックによって検出されると、実運用アプリケーション環境での潜在的な問題について早期の通知を受け取る。
- パフォーマンス情報にドリルダウンして、webMethods Integration Server のどのビジネス プロセス、サービス、または接続プールに遅延やエラーが発生しているかを特定する。

事前設定済みの webMethods ダッシュボードは、webMethods Integration Server 管理モジュール（*WebMethodsISManagementModule.jar*）の一部として webMethods 用の Enterprise Manager 拡張にパッケージ化されています。

webMethods 管理モジュールには、webMethods Integration Server のために以下の事前設定済みのダッシュボードが用意されています。

WebMethods - **ホーム**

WebMethods Integration Server とその主要コンポーネントに関するトップレベルのアーキテクチャの概要。これには、すべてのサービス、ビジネス プロセス、トレーディング ネットワーク コンポーネント、および外部バックエンド システムの全般的な稼働状況に関するアラート インジケータが含まれます。

WebMethods - IS **サービス概要**

フロー、Java、XSLT サービスおよびトリガに関する要約されたステータス。これには、平均応答時間に関するグラフと、エラー数およびストール数に関するアラート インジケータが含まれます。

このダッシュボードで［IS の低速のサービス］リンクまたはいずれかの［全般的な稼働状況］ラベルをダブルクリックすると、［WebMethods - IS の低速のサービス］ダッシュボードが表示されます。このダッシュボードには、最も遅いフロー、Java、XSLT サービスのリストと、最も遅いトリガのリストが表示されます。

WebMethods - **ビジネス プロセス**

すべてのビジネス プロセスに関する要約されたステータス。これには、プロセス レベルと手順レベルの両方の平均応答時間と Errors Per Interval （間隔ごとのエラー数） のグラフが含まれます。

このダッシュボードには、プロセス レベルで再起動、一時停止、再開されたオペレーションに関するアラート インジケータ、手順レベルでの応答時間、ストール数、同時進行中の呼び出し数、エラー数に関するアラート インジケータ、および最も遅いビジネス プロセスのリストが含まれます。

### WebMethods - Web サービス概要

すべてのクライアント側およびサーバ側の Web サービスに関する要約されたステータス。これには、クライアント側とサーバ側のパフォーマンスをハイライト表示するための平均応答時間のグラフ、最も遅いクライアント側とサーバ側のサービスのリスト、およびクライアントとサーバのサービスの応答時間、SOAP 障害数、エラー数、ストール数に関するアラート インジケータが含まれます。

このダッシュボードで［Web サービス オペレーション］リンクまたはいずれかの［全般的な稼働状況］ラベルまたはグラフをダブルクリックすると、［WebMethods - Web サービス オペレーション］が表示されます。このダッシュボードには、クライアント側とサーバ側のオペレーションに関する平均応答時間のグラフ、SOAP 障害数とストール数に関するアラート インジケータ、および最も遅いクライアント側とサーバ側の Web サービス オペレーションのリストが含まれます。

### WebMethods - アダプタ

すべての webMethods アダプタに関する要約されたステータス。これには、すべてのアダプタ サービス、アダプタ通知、および外部バックエンド システムに関する平均応答時間のグラフが含まれます。

アダプタ サービスについては、ダッシュボードに以下の情報も表示されます。

- Errors Per Interval（間隔ごとのエラー数）と Stall Count（ストール数）に関するグラフ
- 同時進行中の呼び出し数、エラー数、およびストール数に関するアラート インジケータ
- 最も遅いアダプタ サービスのリスト

アダプタ通知と外部バックエンドについては、ダッシュボードにエラー数とストール数に関するアラート インジケータも表示されます。

### WebMethods - トレーディング ネットワーク

トレーディング ネットワークによって処理されるすべてのドキュメント処理に関する要約されたステータス。これには、すべてのドキュメント タイプと処理ルールに関する平均応答時間と Errors Per Interval（間隔ごとのエラー数）のグラフ、エラー数とストール数に関するアラート インジケータ、および処理が最も遅いドキュメントのリストが含まれます。

WebMethods - **接続 & スレッド プール**

アダプタ接続プール、JDBC 接続プール、およびスレッド プールに関する要約されたステータス。これには、アダプタ接続と JDBC 接続の使用可能な接続数と現在のプール サイズに関するグラフ、および使用中のスレッド数とスレッド プール内の最大スレッド数に関するグラフが含まれます。

事前構成済みのダッシュボードは、Workstation コンソールを使用して表示できます。 また、webMethods Integration Server 管理モジュールを拡張してカスタム ダッシュボードを含めたり、デフォルト ダッシュボード定義を変更してカスタム メトリックやカスタム アラートを含めたりすることもできます。

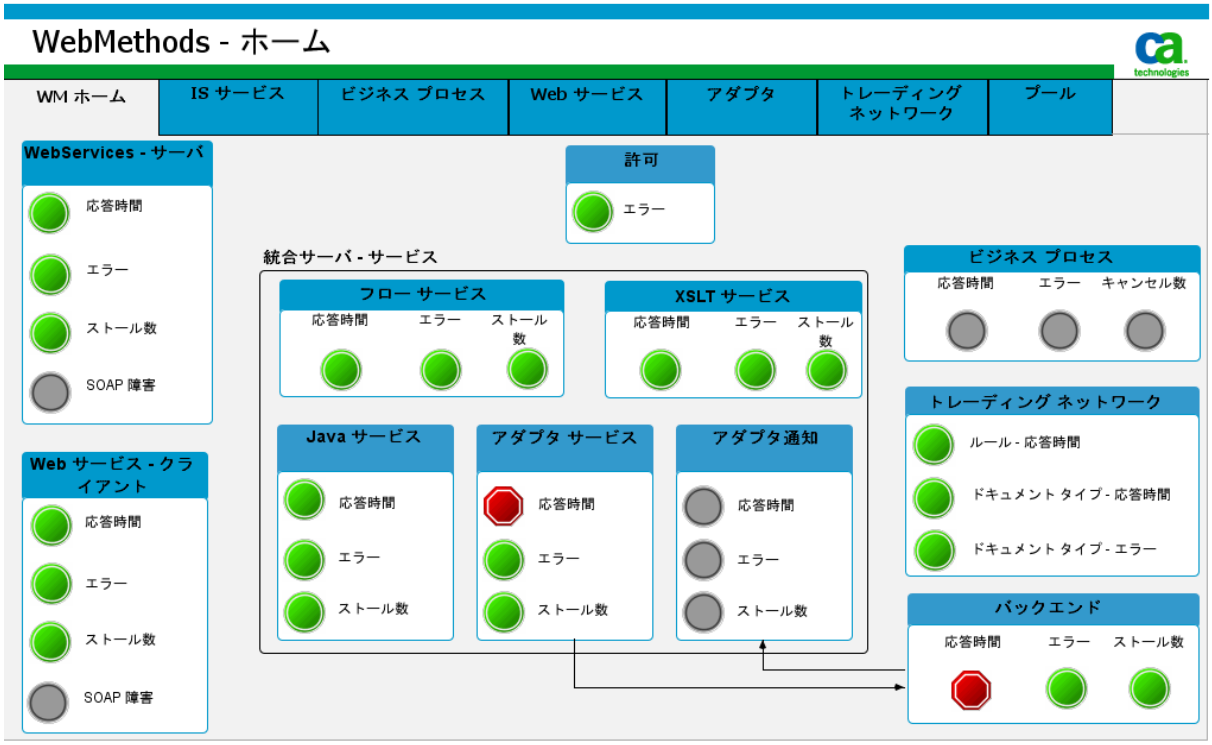
**注**: ダッシュボードを作成および変更する方法の詳細については、「CA APM 設定および管理ガイド」を参照してください。

**次の手順に従ってください**:

1. Enterprise Manager を起動します（現在実行されていない場合）。
2. Workstation を起動し、SOA extension for webMethods がインストールされている Enterprise Manager にログインします。
3. ［Workstation］-［新規コンソール］を選択します。

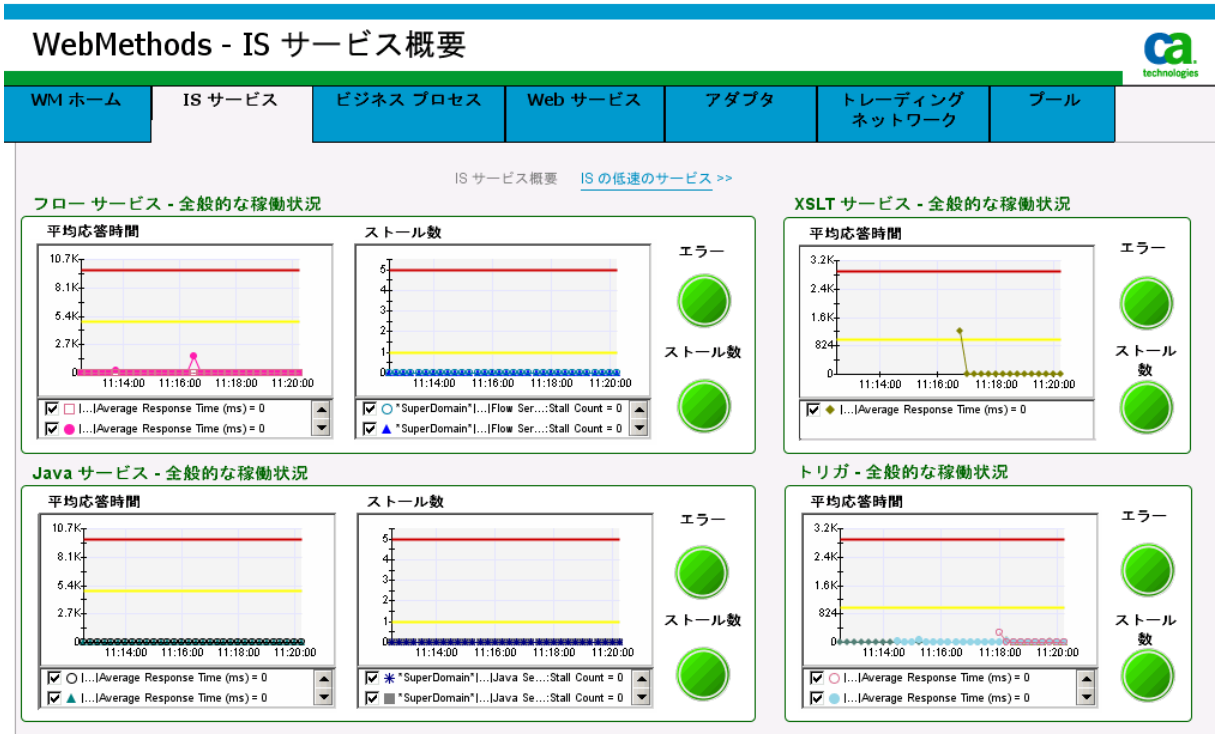
4. ［ダッシュボード］ドロップダウン リストから webMethods のいずれかのダッシュボードを選択します。

   たとえば、webMethods の主要コンポーネントと内部ワークフローの概要を表示するには、［WebMethods ホーム］ダッシュボードを選択します。

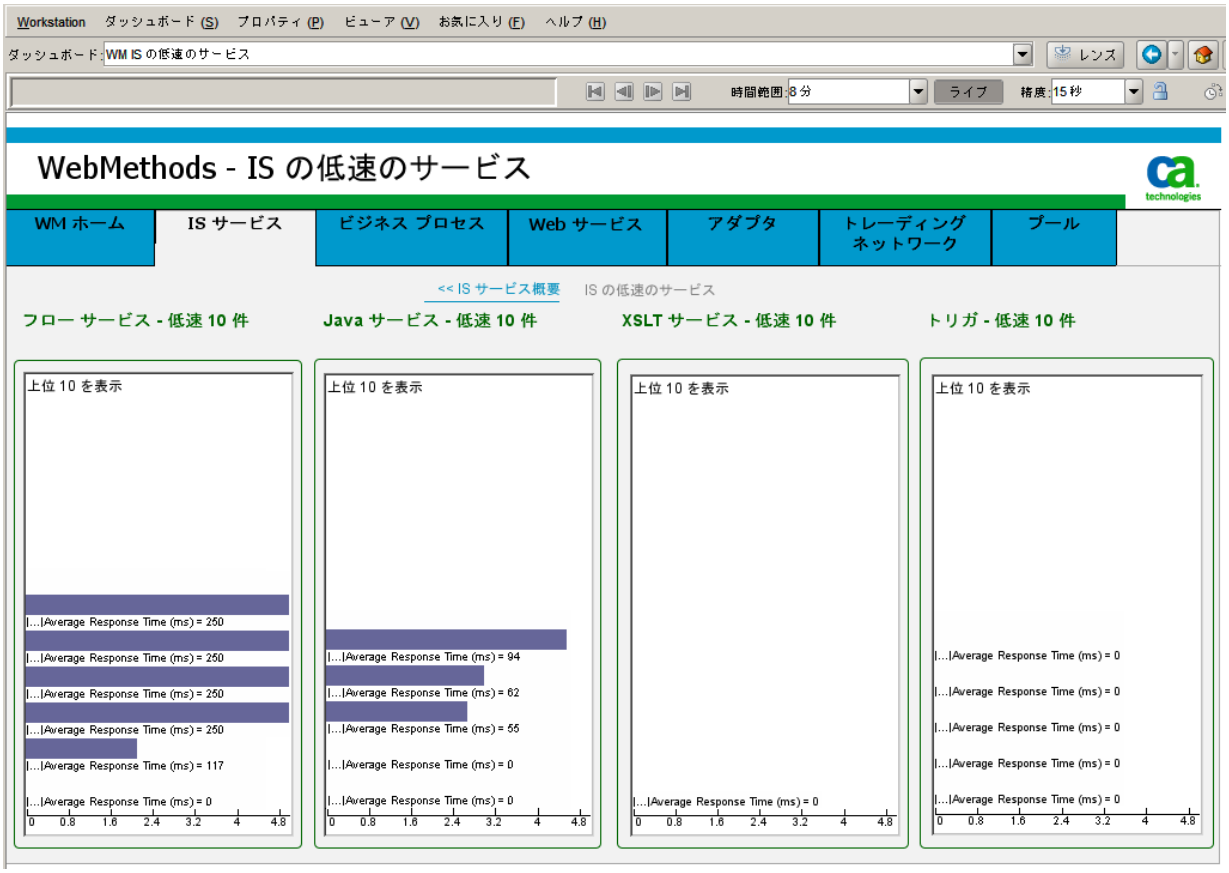


5. 別のタブまたはダッシュボード内のアラートをダブルクリックすると、関連するダッシュボードが開き、詳細情報が表示されます。

   たとえば、［Java サービス］アラートをダブルクリックすると、［WebMethods - IS サービス概要］ダッシュボードに webMethods Integration Server のフロー、Java、および XSLT サービスの全般的な稼働状況に関する詳細情報が表示されます。



6. ［WebMethods - IS サービス概要］ダッシュボードで［IS の低速のサービス］をダブルクリックすると、最も遅い個々のフロー、Java、XSLT、およびトリガ サービスのリストが表示されます。 例：




7. ダッシュボードに表示された特定のサービス、ビジネス プロセス、またはドキュメントのメトリックをダブルクリックすると、さらなる分析のために Investigator が表示されます。

［WebMethods - IS の低速のサービス］ダッシュボードでは、たとえば、最も遅いフロー サービスをダブルクリックすると、そのサービスの［Average Response Time （平均応答時間）］が選択された状態で Investigator が表示されます。

# 監視と表示の対象となるサービスのフィルタリング

webMethods Integration Server については、構成ファイルでフィルタを指定することにより、監視して Investigator ツリーに含めるフロー サービスまたは Java サービスを制御できます。 デフォルトの構成ファイルは *wmExtension.config* です。 このファイルを編集することにより、監視に含めるフロー サービス、または監視から除外するフロー サービスを識別する正規表現を使用してフィルタを作成できます。

## デフォルトの構成ファイルについて

デフォルトの構成ファイル（wmExtension.config）は、*<Agent_Home>*/common ディレクトリにあり、デフォルトの包含および除外フィルタを使用して設定されています。 たとえば、webMethods の組み込みサービスを除外するために以下のデフォルト フィルタが定義されています。

```
com.wily.wm.service.filter.exclude=wm.*,pub.*
```

この構成ファイルには、一致する webMethods サービスを含めるために以下のデフォルト フィルタも定義されています。

```
com.wily.wm.service.include=wm.tn:receive,wm.tn.route:routeBizdoc,pub.prt.tn:handleBiZDoc
```

デフォルトの構成ファイルを変更することにより、必要に応じて追加のサービスを包含または除外できます。また、組み込みサービスを含む webMethods のすべてのサービスを監視する場合は、デフォルト フィルタを削除できます。 wmExtension.config ファイルにフィルタが定義されていない場合は、組み込みサービスを含むすべてのサービスが Investigator ツリーに表示されます。

## 正規表現を使用したサービスの包含と除外

*wmExtension.config* 構成ファイルでは、*com.wily.wm.service.filter.include* プロパティと *com.wily.wm.service.filter.exclude* プロパティを使用して、包含または除外するサービスを指定できます。 たとえば、監視から除外するサービスを定義する正規表現を使用して *com.wily.wm.service.filter.exclude* プロパティを設定できます。 include および exclude プロパティには、サービスの完全修飾名に適用される任意の式を使用できます。 たとえば、名前が「*webservice*」という文字列で終わるすべてのフロー サービスを除外するには、exclude プロパティを以下のように設定します。

```
com.wily.wm.service.filter.exclude=.*webservice
```

複数のフィルタを定義する場合は、カンマを使用して正規表現を区切ります。 たとえば、名前が「*wm.server*」および「*wm.tomcat*」で始まるすべてのフロー サービスと Java サービスを除外するには、*com.wily.wm.service.filter.exclude* プロパティを以下のように設定します。

```
com.wily.wm.service.filter.exclude=wm.server.*,wm.tomcat.*
```

この正規表現と一致するすべてのフロー サービスと Java サービスが除外され、残りのサービスのみが Investigator に表示されます。 ただし、有効でない正規表現を指定すると、どのサービスも監視から除外されず、組み込みサービスを含むすべてのフロー サービスと Java サービスが表示されます。

## 構成ファイルの別の場所の指定

デフォルトの wmExtension.config ファイルは <*Agent_Home*>/common ディレクトリにありますが、サーバの起動スクリプトで別の場所を指定することにより、別のディレクトリまたは別の構成ファイルを指定できます。たとえば、代替の wmExtension.config ファイルが C:¥CA-Introscope ディレクトリにある場合は、サーバの起動スクリプト内の Java 引数に以下のように com.wily.wm.service.filter.fileloc プロパティを追加します。

```
set JAVA_OPTS=%JAVA_OPTS% -Dcom.wily.wm.service.filter.fileloc=C:¥CA-Introscope
```

指定したパス内に wmExtension.config という名前のファイルが存在する場合、エージェントは構成ファイル内の include プロパティと exclude プロパティをチェックして、監視するフロー サービスと Java サービスを決定します。

サーバの Java 引数に構成ファイルへのパスを指定すると、エージェントはそのファイルに指定されたフィルタを使用して、除外するフロー サービスと Java サービスを決定します。構成ファイルへのパスを指定しないと、エージェントは <*Agent_Home*>/common/wmExtension.config ファイルに定義されたフィルタを使用します。
<*Agent_Home*>/common/wmExtension.config も代替の wmExtension.config ファイルも見つからない場合は、フィルタリングが行われず、組み込みサービスを含むすべてのフロー サービスと Java サービスが表示されます。

# webMethods に関するメトリックを表示および操作する

Investigator ツリーを操作すると、webMethods Integration Server インフラストラクチャのほとんどのコンポーネントに関する CA Introscope® の標準メトリックが表示されます。 収集された標準メトリックのデータは、Investigator ツリーのノードおよびサブノードとして表示される webMethods 専用のメトリック カテゴリに集計されます。- 表示される具体的なメトリック カテゴリとノード名は、環境内に展開したプロセス、サービス、およびリソースによって異なります。

Investigator ツリーを操作しながら、選択するノードに応じて個々のオペレーションに関する低レベルのメトリックまたは集計メトリックを表示できます。これにより、webMethods Integration Server によって展開されたさまざまなサービスの全般的な稼働状況を監視できます。

### Investigator で webMethods のメトリックを表示および操作する方法

1. エージェント ノードを展開して［WebMethods］ノードをクリックし、［概要］タブを表示します。このタブには、監視している webMethods のすべてのフロー サービスとビジネス プロセスに関する CA Introscope® の標準メトリックを含むサマリ情報が一覧表示されます。
2. リスト内のフロー サービスまたはビジネス プロセスを選択すると、そのサービスまたはプロセスに関するすべての標準メトリックがグラフィカルな形式で表示されます。
3. ［WebMethods］ノードを展開すると、webMethods Integration Server に関するトップレベルのメトリック カテゴリのサブノードが表示されます。

4. サブノードをクリックまたは展開すると、そのメトリック カテゴリに関するサマリ情報を含む[概要]タブが表示されます。たとえば、[Java Services]ノードをクリックすると、[概要]タブに Java サービスに関する要約されたメトリックが表示されます。
5. いずれかのサブノードを展開すると、個々のビジネス プロセス、フロー サービス、Java サービス、または接続プールに関する詳細情報と、それぞれに関連付けられたメトリックが表示されます。

   たとえば、[Flow Services]ノード、特定のフロー サービス名、サブフォルダの順に展開すると、個々のフロー サービスに関するメトリックが表示されます。

## アダプタに関するメトリック

デフォルトの webMethods アダプタは、各組織がビジネス プロセスのリアルタイム実行を実現するのに必要な情報リソースとエンタープライズ アプリケーションへのシームレスな接続を提供します。 webMethods アダプタを使用することで、各組織はカスタム プログラミングや時間のかかる統合開発を最小限に抑えながら、パッケージ化されたアプリケーション（SAP や Oracle のアプリケーション スイートなど）やデータベース（Microsoft SQL Server や Oracle RDBMS など）に接続できるようになります。展開できる具体的なアダプタは、使用している webMethods Integration Server のバージョンによって異なります。

アダプタに関するメトリックは、展開された特定のアダプタに関する詳細情報を提供し、それらのアダプタに関連付けられた接続、オペレーション、イベントの監視を可能にします。

アダプタを監視するには、［WebMethods］ノードの下に表示される以下のメトリック カテゴリのメトリックを使用します。

Adapter Connection Pools

アダプタ接続は、Integration Server が実行時に外部アプリケーションまたは情報ストアに接続できるようにします。

Adapter Notifications

アダプタ通知は、外部リソースを監視し、そのリソースによって開始されたイベントが発生したことを Integration Server に通知できるようにします。 アダプタ通知は、イベント発生時に webMethods Broker にドキュメントを発行します。

Adapter Services

アダプタ サービスは、アダプタと外部リソースの接続を使用して、Integration Server からリソース上のオペレーションを開始できるようにします。

たとえば、webMethods には、JDBC ドライバを使用してデータベースとの統合を作成できる一連のユーザ インターフェース、サービス、およびテンプレートを提供するための JDBC アダプタが用意されています。 データベースへの接続を監視するには、［Adapter Connection Pools］カテゴリのメトリックを使用します。

JDBC アダプタ サービスは、Integration Server がデータベース上のデータの挿入、更新、削除などのデータベース操作を開始して実行できるようにします。アダプタ通知は、データベースを監視し、特定のデータベース テーブル上で更新、挿入、または削除が発生したことを Integration Server に通知できるようにします。

たとえば、アダプタ サービスを使用することで、取引相手がインベントリ データベースに対してクエリを実行し、特定の商品の在庫が現在あるかどうかを確認できるようになります。 また、アダプタ通知を使用することで、インベントリ データベースのテーブルに対して更新が実行されたことを Integration Server に通知できるようになります。

### Investigator でアダプタ関連のメトリックを表示および操作する方法

1. エージェント ノードを展開し、さらに［webMethods］を展開すると、アダプタのメトリック カテゴリが表示されます。
2. ［WebMethods］-［Adapter Connection Pools］を展開し、さらに特定のアダプタ名を展開すると、そのアダプタの接続情報が表示されます。

3. ［WebMethods］-［Adapter Notifications］を展開し、さらに特定のアダプタ名を展開すると、そのアダプタの通知情報が表示されます。 アダプタの通知を設定していない場合、このメトリック カテゴリは表示されません。
4. ［WebMethods］-［Adapter Services］を展開し、さらに特定のアダプタ タイプを展開すると、アクティブな接続のリストが表示されます。
5. アクティブなアダプタ接続を展開すると、Integration Server 上で実行されているアダプタ サービスのリストが表示されます。
6. 個々のサービスを展開すると、そのサービスに関するメトリックが表示されます。

## アダプタ接続プールに関するメトリック

設定済みのアダプタについては、［WebMethods］-［Adapter Connection Pools］サブノードの各アダプタ名の下に以下のメトリックが表示されます。

Available Connections

アダプタで使用可能な解放された接続の数。

利用可能な接続の数は、現在アクティブな接続の数からビジーな接続の数を引くことで計算されます。 サーバがリソースへの接続を試行するたびに、利用可能な接続の数は減少します。

Current Size

現在アクティブな接続の総数。

Maximum Size

選択したアダプタで許可される接続の最大数。

Minimum Size

選択したアダプタに設定された接続の最小数。

## アダプタ通知に関するメトリック

アダプタ通知を設定すると、ポーリング プロセスまたはリスナ プロセスが外部リソースの変更（データベース テーブルに対する挿入、更新、削除オペレーションなど）を監視します。これにより、適切なフロー サービスまたは Java サービスが外部リソースの変更に対応できるようになります。

たとえば、アダプタ サービスを展開して、取引相手がインベントリ データベースに対してクエリを実行できるようにした場合は、インベントリ データベースのテーブルに対して更新が実行されたことを常に Integration Server に通知するようにアダプタ通知を設定できます。 たとえば、アダプタ通知に基づいて請求書を送付するために、通知に関連付けられたドキュメントを処理するには、Integration Server のトリガを設定します。

アダプタ通知については、［WebMethods］-［アダプタ通知］サブノードの各アダプタ サービス名の下に CA Introscope® のすべての標準メトリックが表示されます。

### アダプタ サービスに関するメトリック

アダプタ サービスについては、［WebMethods］-［アダプタ サービス］サブノードの各アダプタ接続サービス名の下に CA Introscope® のすべての標準メトリックが表示されます。

Errors Per Interval （間隔ごとのエラー数） メトリックには、アダプタ サービスの実行時に発生したエラーのみが含まれます。 サービスが実行される前に発生したアクセス例外やその他のタイプのエラーは含まれません。

## 許可に関するメトリック

webMethods Integration Server には、通常、異なるユーザに付与される権限を定義した専用の ACL（Access Control List、アクセス制御リスト）を持つ複数のユーザ グループがあります。たとえば、Administrators、Developers、Monitor Users、Replicators などのグループには、デフォルトの権限が割り当てられています。 また、必要に応じて独自のユーザ グループや権限を作成することもできます。 ユーザがアクセス権限を持たない Integration Server サービスを実行しようとすると、アクセスが拒否され、エラーが記録されます。

［WebMethods］-［Authorization］の下にある Errors Per Interval （間隔ごとのエラー数） メトリックを選択すると、許可の失敗に関する情報が表示されます。

ユーザのアクセスが拒否された場合は、サービスが起動されないため、そのサービスの Errors Per Interval （間隔ごとのエラー数） メトリックにエラーは記録されません。 ただし、Investigator の［エラー］タブをクリックして、アクセス例外エラーを表示することもできます。 リスト内のエラーを選択すると、［エラー スナップショット］にそのエラーに関する詳細情報が表示されます。

## ビジネス プロセスに関するメトリック

ビジネス プロセスは、発注や新入社員の追加などのビジネス イベントを完了するための一連の手順で構成されます。

webMethods のビジネス プロセスおよびビジネス プロセス手順については、［WebMethods］-［ビジネス プロセス］-［*<business_process_name>*］ノードの下に CA Introscope® の以下の標準メトリックのみが表示されます。

- Average Response Time (ms)
- Errors Per Interval
- Responses Per Interval

これらのメトリックは、ビジネス プロセスのレベルで、プロセスが正常に完了するまでの時間、正常に完了したプロセスの数、およびエラーを生成して失敗したプロセスの数を追跡します。 分散型プロセスについては、ビジネス プロセスの最後の手順が実行されるエージェントに関してのみ、プロセス レベルの Average Response Time（平均応答時間）と Responses Per Interval （間隔ごとの応答数） が表示されます。

ビジネス プロセスについては、標準メトリックに加えて以下のメトリックが表示されます。

Cancels Per Interval

間隔内でキャンセルされたプロセスの数。

Suspends Per Interval

間隔内で一時停止されたプロセスの数。

Restarts Per Interval

間隔内で再起動されたプロセスの数。

Resumes Per Interval

間隔内で再開されたプロセスの数。

また、webMethods Integration Server のビジネス プロセスに関する依存関係および偏差のメトリックを収集することもできます。 標準の依存関係および偏差のメトリックについては、「Investigator を使用して SOA パフォーマンス メトリックを表示する (P. 64)」を参照してください。

## ビジネス プロセスの手順レベルのメトリックを表示する

デフォルトでは、webMethods Integration Server を監視している場合、CA Introscope® のすべての標準メトリックが、以下のようにビジネス プロセス手順にも使用できます。

［webMethods］-［BusinessProcesses］-［<*business_process_name*>］-［<*step_identifier*>］ノード

**webMethods Integration Server 6.5.2 ～ 6.5.3 で有効：** 単一レベルの手順についてのみ、手順レベルのメトリックが表示されます。 その他の手順レベルのメトリック（内部で複数の手順として実行されるヒューマン タスクの手順、参照されたプロセス、サブプロセス、フロー サービスの手順などに関するメトリック）は、デフォルトでは集計されません。

webMethods Integration Server 6.5.2 ～ 6.5.3 を監視するときに、ビジネス プロセスに関する手順レベルのメトリックをすべて使用できるようにするには、webmethods-toggles.pbd ファイルに含まれる以下の行のコメント化を解除します。-

```
TurnOn: BProc<version_number>FlowStepMarkTracing
TurnOn: BProc<version_number>StepFlowTracing
```

**注**: webMethods Integration Server の要件については、「*Compatibility Guide*」を参照してください。

**次の手順に従ってください**:

1. エージェント ノードを展開し、次に、［webMethods］-［ビジネス プロセス］を展開します。

   webMethods Designer で定義し、パッケージとして環境内に展開したビジネス プロセスが表示されます。

   実行されているビジネス プロセスのバージョンを区別するため、ビジネス プロセス名の末尾にアンダースコア（_）とバージョン番号が追加されます。

2. 任意のプロセス名を展開します。

   プロセスに対して定義した手順が表示されます。

3. 任意の手順 ID （StepID）を展開します。

   Integration Server 上の手順を表す手順レベルのメトリックとフロー サービス メトリックが表示されます。

## 完了したプロセスの平均応答時間について

webMethods のビジネス プロセスは、実行中に My WebMethods Server からキャンセルまたは一時停止できます。 My WebMethods Server を使用してキャンセルまたは一時停止したプロセスは、多くの場合、My WebMethods Server から手動で再開することもできます。 これらのプロセスは My WebMethods Server から再サブミットして完了するまで実行できるため、プロセス レベルの Average Response Time （平均応答時間）メトリックは正常に完了したプロセスのみを表します。

プロセスが失敗するか、または一時停止され、My WebMethods Server の内部から再サブミットされた場合、Average Response Time （平均応答時間）メトリックには再サブミット後のプロセスの初期呼び出しから正常完了までの時間が反映されます。プロセスが失敗し、My WebMethods Server を使用して再サブミットされなかった場合、そのプロセスは Average Response Time （平均応答時間）メトリックに含まれません。 しかし、プロセスがそのプロセス フローの一部として（たとえば、Terminate 手順を使用して）キャンセルされた場合、そのプロセスは Average Response Time （平均応答時間）メトリックに含まれます。

ビジネス プロセスに複数の Integration Server が関係している場合は、ビジネス プロセスが完了した Integration Server についてのみ Average Response Time （平均応答時間）メトリックがレポートされます。

### 完了したプロセスの Responses Per Interval（間隔ごとの応答数）について

Responses Per Interval （間隔ごとの応答数） メトリックには、正常に完了したプロセスの数が反映されます。ビジネス プロセスに複数の Integration Server が関係している場合は、ビジネス プロセスが完了した Integration Server についてのみ Responses Per Interval （間隔ごとの応答数） メトリックがレポートされます。

### 完了したプロセスの Errors Per Interval（間隔ごとのエラー数）について

Errors Per Interval （間隔ごとのエラー数） メトリックは、プロセスが完了するまで実行されなかった場合にのみ表示されます。

### プロセス内の手順の Concurrent Invocations（同時進行中の呼び出し）について

サブプロセス、参照されたプロセス、ヒューマン タスクの手順など、一部の手順は、webMethods Designer で単一の手順として表示されても、内部では複数の手順として実行されます。 たとえば、ヒューマン タスクの手順は、内部では *PRE_StepID* と *POST_StepID* の 2 つの手順として実行されます。 Concurrent Invocations （同時進行中の呼び出し） メトリックでは、これらの内部手順は単一の手順に集約されず、個別の呼び出しとしてカウントされます。

## フロー サービスに関するメトリック

フロー サービスは、webMethods のフロー言語で記述されたサービスです。フロー サービスは一連のフロー手順で構成され、各手順には入力と出力が明確に定義されます。 個々のフロー手順は、webMethods が実行時に解釈して実行する作業の基本単位です。 特定のサービスのフロー手順は、ある手順の出力がフロー内の次の手順の入力として使用できるように、すべて同じスレッドを使用して実行されます。

webMethods の個々のフロー サービスについては、［WebMethods］-［フロー サービス］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。 データは個々のフロー手順ごとに収集され、Integration Server の上のすべてのフロー サービスの全般的な稼働状況を監視するため、1 つのフロー サービスに関するメトリック、および複数のフロー サービスにまたがるメトリックに集計されます。

表示されるノード名は、監視対象として選択し、フィルタを使用して除外していないサービスの完全修飾名です。 監視対象のフロー サービスのフィルタリングについては、「監視と表示の対象となるサービスのフィルタリング (P. 334)」を参照してください。

### ほかのフロー サービスを呼び出すフロー サービス

ほとんどの場合、webMethods のフロー サービスには、ほかのアプリケーション コンポーネントと同じように CA Introscope® のすべての標準メトリックとそれらに対応する集計されたメトリック値が適用されます。

あるフロー サービスが別のフロー サービスを呼び出すと、両方のフロー サービスに関するメトリックが Investigator ツリーの［WebMethods］-［Flow Services］ノードの下に個別のノードとして表示されます。 一方のノードがもう一方のノードの下にネストされることはありません。

呼び出しシーケンスの中で同じ webMethods Integration Server 上で実行されたすべてのフロー手順を表示するには、トランザクション追跡セッションを開始します。

## Java サービスに関するメトリック

Java サービスは、Java 言語を使用して実装され（または他の言語で記述されたサービスが Java クラスを使用してラップされ）、webMethods Integration Server 上でサービスとして公開されるユーザ定義のサービスまたは webMethods に組み込まれた内部サービスです。 たとえば、webMethods Integration Server が提供する組み込みサービスはすべて Java サービスです。

展開されている個々の webMethods Java サービスについては、［WebMethods］-［Java services］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

webMethods では、同じフォルダ内に存在する Java サービスは同じクラスのメソッドです。 たとえば、完全修飾名が *recording.user.accounts:createAccount* である Java サービスは、Java パッケージ（*recording.user*）、Java クラス（*accounts*）、および Java メソッド（*createAccount*）で構成されます。 Investigator に表示されるノード名は、監視対象として選択し、フィルタを使用して除外していないサービスの完全修飾名を表します。

監視対象の Java サービスのフィルタリングについては、「監視と表示の対象となるサービスのフィルタリング (P. 334)」を参照してください。

## JDBC プールに関するメトリック

webMethods Integration Server は、JDBC 接続ポイントを介してデータベースに接続することにより、ユーザ、ドキュメント、サーバ内部の機能、監査ログとエラー ログ、およびその他の情報を収集して格納します。 JDBC プールで許可される JDBC 接続の最大数は、Integration Server Administrator を使用して設定できます。

JDBC の使用状況の監視については、［WebMethods］-［JDBC Pools］サブノードの各スレッド プール名の下に以下のメトリックが表示されます。

Available Connections

JDBC プールの利用可能な接続の数。

利用可能な接続の数は、現在アクティブな接続の数からビジーな接続の数を引くことで計算されます。

Current Size

現在アクティブな JDBC 接続の総数。

Maximum Size

選択したプールに設定された JDBC 接続の最大数。

Minimum Size

選択したプールに設定された JDBC 接続の最小数。

## スレッド プールに関するメトリック

webMethods Integration Server は、サービスの実行、webMethods Broker からのドキュメントの取得、およびトリガの実行にスレッドを使用します。サーバ起動時の最初のスレッド プールには、最小数のスレッドが含まれています。サーバは、許可された最大数に達するまで、必要に応じてプールにスレッドを追加します。 最大数のスレッドが使用されている場合、サーバはプロセスが完了するまで待機した後、プールにスレッドを返してから新しいプロセスを開始します。

スレッドの使用状況の監視については、［WebMethods］-［Thread Pools］サブノードの各スレッド プール名の下に以下のメトリックが表示されます。

Maximum Size

選択したスレッド プールに設定されたスレッドの最大数。

Minimum Size

選択したスレッド プールに設定されたスレッドの最小数。

Used Threads

選択したスレッド プールで現在使用中のスレッドの数。

## トレーディング ネットワークに関するメトリック

各組織は、トレーディング ネットワークを使用してドキュメントを交換することにより、企業間の関係を確立し、充実させることができます。たとえば、購入者、供給元、戦略的パートナー、またはその他の組織を、ドキュメントを交換する取引相手として識別できます。 ドキュメントの交換によって、組織の境界にまたがるビジネス プロセスを合理化できます。webMethods Integration Server では、トレーディング ネットワークが取引相手間のゲートウェイとして機能します。

webMethods のトレーディング ネットワークについては、［webMethods］-［Trading Networks］ノードの下に以下の標準メトリックが表示されます。

- Errors Per Interval

さらに、［WebMethods］-［Trading Networks］ノードの下にある以下のメトリック カテゴリを使用して、トレーディング ネットワークにおける XML ドキュメントの認識と処理ルールを監視できます。

Document Types

ドキュメント タイプは、異なるパートナー関係で交換されるデータの構造とタイプを識別します。

Processing Rules

処理ルールは、ドキュメントを Integration Server 経由でその宛先にルーティングする方法を示します。

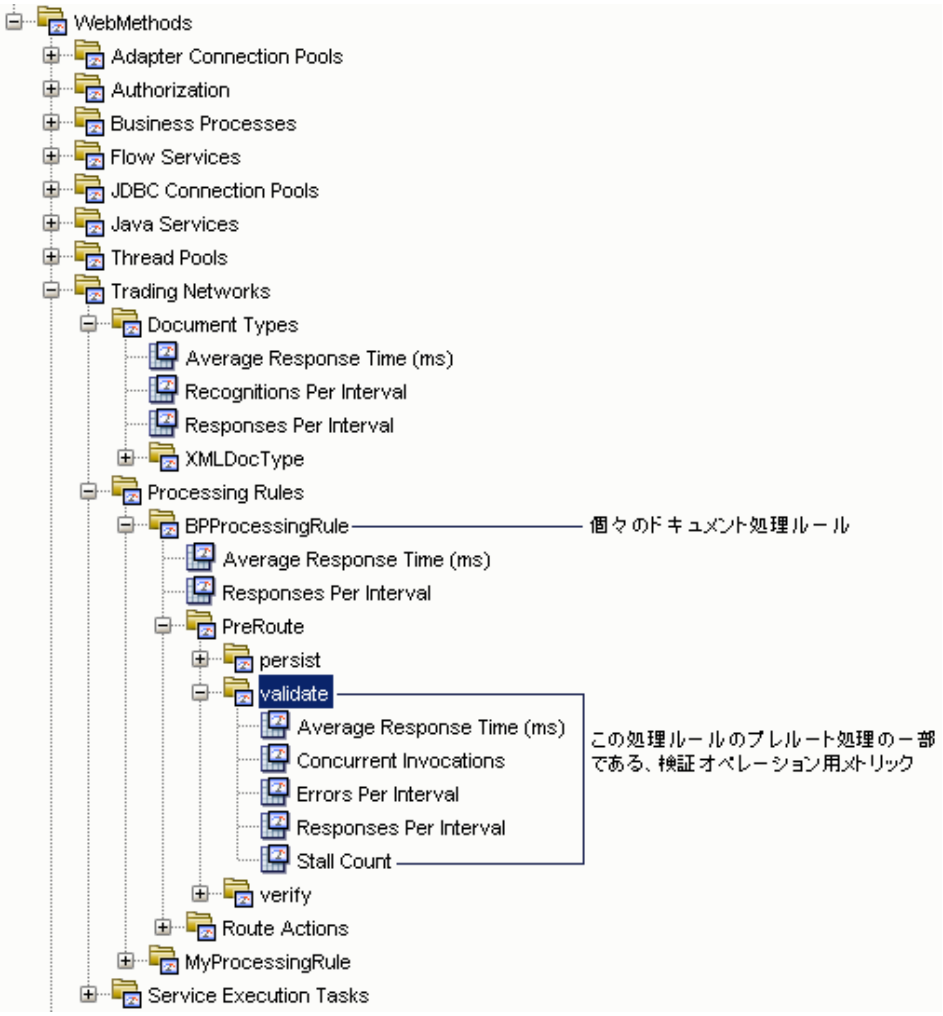
Service Execution Tasks

サービス実行タスクは、処理ルールによって再試行回数を制限しながらサービスを非同期に実行するときに作成されます。

Service Threads

サービス スレッドは、再試行回数を制限せずにルーティングされるドキュメントの非同期処理に対応するために作成されます。

たとえば、［WebMethods］-［Trading Networks］-［Document Types］サブノードを展開すると、特定のドキュメント タイプに関するメトリックが表示されます。これには、各ドキュメント タイプのドキュメントの認識と承認に関するメトリックが含まれます。 同様に、［WebMethods］-［Trading Networks］-［Processing Rules］サブノードを展開すると、特定の処理ルールに関するメトリックが表示されます。これには、個々のプレルーティングおよびルーティング オペレーションに関するメトリックが含まれます。-



## ドキュメント タイプに関するメトリック

webMethods のトレーディング ネットワークについては、［webMethods］-［トレーディング ネットワーク］-［ドキュメント タイプ］ノードの下に CA Introscope® の以下の標準メトリックのみが表示されます。

- Average Response Time (ms)
- Responses Per Interval

ドキュメント タイプの Responses Per Interval メトリックと 間隔ごとの応答数 メトリックは、ドキュメントが処理されるときに集計されます。 これらのメトリックは、処理ルールが実行されたときにのみレポートされます。 ドキュメントがサブミットされても、処理ルールによって処理されなければ、Average Response Time（平均応答時間）メトリックと Responses Per Interval （間隔ごとの応答数） メトリックはレポートされません。

有効なドキュメントがサブミットされた場合は、［webMethods］-［トレーディング ネットワーク］-［ドキュメント タイプ］ノードの下に、標準メトリックに加えて以下のメトリックが表示されます。

Recognitions Per Interval

15 秒の間隔が終了するまでに有効な取引相手のドキュメントとして認識されたドキュメントの数。-

認識された各ドキュメント タイプについては、［webMethods］-［トレーディング ネットワーク］-［ドキュメント タイプ］-［*<document_type_name>*］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

## 処理ルールに関するメトリック

webMethods のトレーディング ネットワークについては、［webMethods］-［トレーディング ネットワーク］-［処理ルール］-［*<processing_rule_name>*］サブノードの各処理ルールの下に CA Introscope® の以下の標準メトリックのみが表示されます。

- Average Response Time (ms)
- Responses Per Interval

webMethods の個々の処理ルール オペレーションについては、［WebMethods］-［トレーディング ネットワーク］-［処理ルール］-［*<processing_rule_name>*］-［プレルート］または［ルート アクション］サブノードに含まれる処理ルールの特定のオペレーションの下に、CA Introscope® のすべての標準メトリックが表示されます。

### 処理ルールの平均応答時間について

処理ルールの平均応答時間メトリックは、ドキュメントを同期的にルーティングするのにかかった平均応答時間を集計したものです。 処理ルール内の同期呼び出しについては、このメトリックはドキュメント タイプに関して集計された平均応答時間と同じです。

### サービス実行タスクに関するメトリック

webMethods のトレーディング ネットワークについては、［webMethods］-［Trading Networks］-［Service Execution Tasks］ノードの下に以下のメトリックが表示されます。

- New Tasks Per Interval
- Task Failures Per Interval
- Tasks Completed Per Interval

さらに、個々の呼び出しオペレーションについては、［webMethods］-［トレーディング ネットワーク］-［Service Execution Tasks］-［invoke］サブノードの下に CA Introscope® の以下の標準メトリックが表示されます。-

- Average Response Time (ms)
- Responses Per Interval

### サービス スレッドに関するメトリック

webMethods のトレーディング ネットワークについては、［webMethods］-［トレーディング ネットワーク］-［Service Threads］サブノードの下に CA Introscope® の以下の標準メトリックが表示されます。-

- Average Response Time (ms)
- Responses Per Interval

## トリガに関するメトリック

Integration Server トリガは、webMethods Broker または JMS（Java Message Service）を使用してドキュメントを処理するように設定できます。 webMethods Broker トリガは、ローカルに発行されたドキュメント、または webMethods Broker に配信されたドキュメントを購読して処理するトリガです。 JMS トリガは、JMS プロバイダ上の宛先（キューやトピックなど）からメッセージを受信し、それらのメッセージを処理するトリガです。

トリガについては、［WebMethods］-［トリガ］サブノードの各トリガ名の下に CA Introscope® のすべての標準メトリックが表示されます。 Broker トリガ、ローカル トリガ、または JMS トリガを設定していない場合、このメトリック カテゴリは表示されません。

## Web サービスに関するメトリック

Web サービスは、ユーザまたはソフトウェア プログラムが使用できるように 1 つの単位としてパッケージ化され、ネットワークに発行される構成要素です。

- webMethods Integration Server のサポートされているバージョンで有効 — Web サービス コネクタは、Integration Server が Web サービス *コンシューマ*（クライアント）と Web サービス *プロバイダ*（サーバ）のどちらとしてとして機能するかを定義します。 たとえば、展開している任意のフロー サービスまたは Java ベースのサービスを Web サービスとして外部に公開できます。 サービスに関する情報を UDDI レジストリに発行するプロバイダ Web サービス記述子を使用します。 webMethods Integration Server は、コンシューマ Web サービス記述子を使用して外部アプリケーション サーバ上の Web サービスを呼び出し、クライアントとしてサービスを要求することもできます。
- webMethods Integration Server 6.5.2 ～ 6.5.3 で有効 — WSDL ドキュメントから Web サービス *コネクタ*を生成して、リモートの Web サービスを呼び出すことができます。 Web サービス コネクタは入出力シグネチャを持つフロー サービスです。 そのシグネチャは、作成元の WSDL ドキュメントの入出力メッセージに対応します。 webMethods Integration Server では、Integration Server と Developer を使用して、Integration Server パッケージ内の既存のサービスを Web サービスに変換することもできます。

**注**: webMethods Integration Server の要件については、「*CA APM Compatibility Guide*」を参照してください。

webMethods Integration Server 上のクライアントおよびサーバ Web サービスとそのオペレーションについては、CA Introscope® のすべての標準メトリックが用意されています。さらに、SOA プラットフォーム上の Web サービスを監視するすべての拡張用の標準メトリックとして、間隔ごとの SOAP 障害数があります。

**詳細**:

［エラー］タブで SOAP 障害およびエラーに関するメトリックを表示する (P. 76)

## サーバ側のメトリックについて

webMethods は、Web サービス プロバイダとして Web サービス要求を処理するために、以下のように複数の SOAP プロセッサを備えています。

- デフォルトの Web サービス SOAP プロセッサ
- SOAP リモート プロシージャ コールを受信して処理する SOAP RPC プロセッサ
- 処理ディレクティブが未定義または省略されている場合にメッセージを処理するデフォルトの SOAP メッセージ ハンドラ

サーバ側の WebServices メトリックには、これら 3 つの SOAP プロセッサの情報がすべて含まれます。 ただし、カスタム SOAP プロセッサはメトリックに含まれません。

## クライアント側のメトリックについて

クライアント側のメトリックは、外部アプリケーション サーバ上で実行される Web サービス要求の実行状況を表します。

webMethods Integration Server は、Web サービス コンシューマとして機能する場合、各オペレーションの Web サービス *コネクタ*を自動的に生成します。 次に、クライアントはコネクタ経由で Web サービスのエンドポイントに直接バインドします。Web サービス コネクタが実行されると、Web サービスの呼び出し要求が Web サービスの実装に直接送信されます。 Web サービス コネクタは、内部的にはフロー サービスであり、Flow Services メトリックを使用して監視できます。

［WebServices］-［Client］メトリックは、クライアントが直接要求しているオペレーションではなく、外部の Web サービスを呼び出すコネクタの実行状況を表します。

サーバ側のメトリックは、webMethods Integration Server 上で実行される Web サービスの実行状況を表します。

## XSLT サービスに関するメトリック

XSL（Extensible Stylesheet Language）と XSLT（XSL Transformations）は、ソース XML ドキュメントを他のドキュメントに変換するための XML ベースの言語を提供します。 たとえば、元の XML ドキュメントを、新しい XML ドキュメントを作成するために使用したり、Web ページとして表示するための HTML に変換したり、プレーン テキストとして発行したりできます。

XSLT サービスを呼び出すと、Integration Server は関連するドキュメント（スタイル シート）の指示を取得し、それらの指示を適用して、ソース XML ドキュメントをスタイル シートで定義された形式の新しいドキュメントに変換します。

個々の webMethods XSLT サービスの監視については、［WebMethods］-［XSLT サービス］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

# webMethods のデフォルト メトリック グループの表示

SOA extension for webMethods Integration Server には、デフォルトのダッシュボードとアラートを定義するために使用されるデフォルト メトリック グループが含まれています。これらのデフォルト メトリック グループをカスタム ダッシュボードやカスタム アラートで使用することもできます。

デフォルト メトリック グループは、webMethods Integration Server 管理モジュール（*WebMethodsISManagementModule.jar*）の一部として webMethods Integration Server 用の Enterprise Manager 拡張にパッケージ化されています。

### webMethods エージェントのデフォルト メトリック グループを表示する方法

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］-［管理モジュール］-［WebMethods IS (*Super Domain*)］の順に展開します。

3. ［メトリック グループ］ノードを展開すると、webMethods 管理モジュールに定義されたすべてのメトリック グループが表示されます。
4. 特定のメトリック グループをクリックすると、［ビューア］ペインにその定義が表示されます。

   任意のメトリック グループのデフォルト設定を変更したり、ユーザ独自のカスタム メトリック グループを作成したりできます。

   **注**: メトリック グループを作成および変更する方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# webMethods のデフォルト アラートの表示

SOA extension for webMethods には、事前設定済みのダッシュボードで使用されるデフォルト アラート定義が含まれています。 これらのデフォルト アラートをカスタム ダッシュボードで使用することもできます。 ほとんどのデフォルト アラートは、デフォルトの警告しきい値と危険しきい値を使用して、しきい値を超えるか重要度が高くなった場合にコンソールに通知を送信するように事前設定されています。

デフォルト アラート定義は、webMethods Integration Server 管理モジュール（*WebMethodsISManagementModule.jar*）の一部として webMethods Integration Server 用の Enterprise Manager 拡張にパッケージ化されています。

**webMethods エージェントのデフォルト アラート定義を表示する方法**

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*SuperDomain*］-［Management Modules］-［WebMethods IS (*Super Domain*)］の順に展開します。
3. ［Alerts］ノードを展開すると、webMethods Integration Server 管理モジュールに定義されているすべてのアラートが表示されます。
4. 特定のアラートをクリックすると、［ビューア］ペインにその定義が表示されます。

特に、デフォルトの警告しきい値と危険しきい値、およびクリティカル アラートに対する事前定義済みのアクションを確認し、環境に合わせて調整してください。 たとえば、必要に応じてしきい値を調整したり、通知を追加したり、修正処置を定義したりできます。

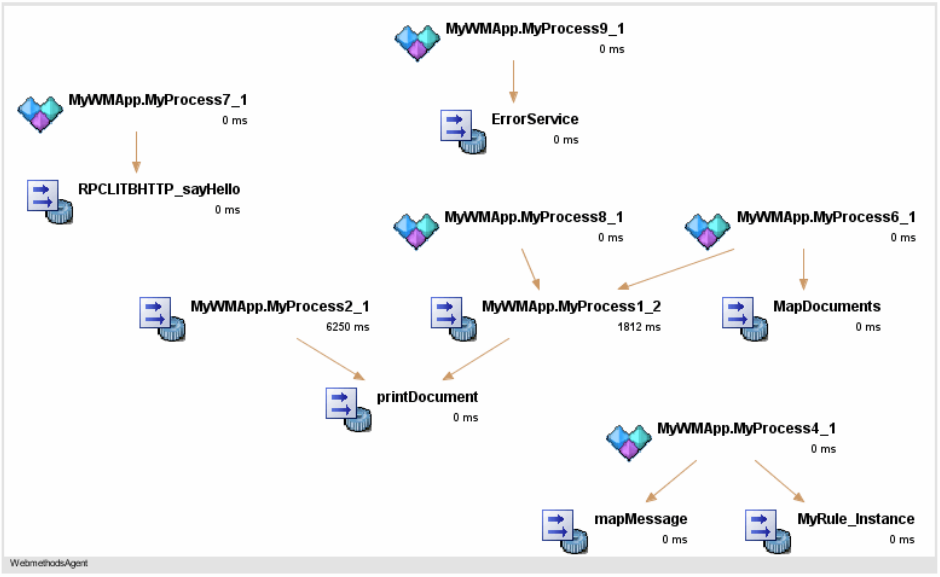
任意のアラートのデフォルト設定を変更したり、ユーザ独自のカスタム アラートを作成したりできます。

**注**: アラートを作成および変更する方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# webMethods の依存関係の表示

Investigator ツリーで適切な webMethods ノードを選択し、［SOA 依存マップ］タブをクリックすると、webMethods のフロー、Java、アダプタ、Web サービス、および webMethods のビジネス プロセスの依存関係が表示されます。

選択するノードによって、依存マップに表示されるコンテキストが決まります。さらに、表示される詳細情報のコンテキストとレベルをロールアップして折りたたんだり、ロールダウンして展開したりできます。 たとえば、すべてのビジネス プロセスの依存関係を以下のようにトップレベルで表示するには、Investigator で［Business Process］ノードを選択し、［SOA 依存マップ］タブをクリックします。



特定のビジネス プロセスの依存関係を高レベルで表示するには、Investigator でビジネス プロセス名を選択し、必要に応じて、そのビジネス プロセスのワークフロー全体が表示されるまでマップに依存関係のレベルを追加し続けるか、またはマップの特定のノードを拡大表示します。依存マップの操作方法の詳細については、「SOA 依存マップの使用 (P. 81)」を参照してください。

# webMethods のトランザクションの追跡

トランザクションの追跡では、ビジネス トランザクションの完了に関係する具体的な手順の詳細ビューまたはサマリ ビューが提供されます。webMethods Integration Server のビジネス プロセスまたはアプリケーション サービスについては、以下のプロトコルによってルーティングされたオペレーションを含むトランザクションを追跡できます。

- SOAP（Simple Object Access Protocol）
- HTTP（HyperText Transport Protocol）
- HTTPS（Hypertext Transport Protocol Secure）
- JMS（Java Message Service）
- webMethods Broker メッセージ サービス

webMethods のサービスが関係するトランザクションには、異なるスレッドを使用して同時に実行される同期および非同期の呼び出しやアクティビティが含まれる可能性があります。 複数のスレッドまたはプロセスにまたがるトランザクションのトランザクション追跡を有効にするため、トランザクションが個々のコンポーネントやオペレーションを順番に実行するときに、相関識別子が挿入され、使用されます。 その結果、実行された特定のオペレーションと各オペレーションの完了までにかかった時間に関する詳細情報を表示できます。

また、CA APM for SOA と CA APM for webMethods Integration Server が追跡対象のすべてのノードで有効になっている限り、あらゆるプラットフォームにまたがって BusinessTransaction を追跡できます。 このため、トランザクションが複数の JVM や CLR にまたがっている場合でも、トランザクションに関する詳細情報を表示できます。

## プロセスにまたがるトランザクション追跡の設定について

通常、プロセスにまたがるトランザクション追跡機能を設定するには、相関関係の情報をプロセス間で受け渡せるようにします。 情報は、SOAP または HTTP ヘッダのいずれかを使用して渡されます。 webMethods Integration Server バージョン 6.5.2 ～ 6.5.3 で有効 -- RPC スタイルの Web サービス クライアント（コネクタ）を使用しているときに、SOAP ヘッダを使用して相関識別子を渡すことができません。

webMethods Integration Server の RPC Web サービス コネクタが関係するビジネス トランザクションのトランザクション追跡を表示するには、HTTP ヘッダを使用して相関識別子を渡すようにエージェントの相関追跡を設定します (P. 133)。

## プロセスにまたがるトランザクション追跡の値について

プロセスにまたがるトランザクション追跡は、サービス指向アーキテクチャ内の疎結合サービスによって実行されているオペレーションに関して貴重な情報を提供します。 プロセスにまたがるトランザクション追跡を使用して、以下のことを特定できます。

- フロー サービスまたは Java サービスによってビジネス プロセスがどのようにルーティングされるか。
- トランザクション中にどのサービスが呼び出され、実行されるか。
- トランザクション中に呼び出され、実行される呼び出しのシーケンス。
- 要求または返答の処理が最も遅い箇所。

## サンプル トランザクション追跡の開始と表示

トランザクション追跡セッションは、以下の方法で開始できます。

- SOA 依存マップ内のマップ ノードから直接開始する方法。
- ［Workstation］-［新規トランザクション追跡セッション］をクリックして Workstation から手動で開始する方法。

依存マップからトランザクション追跡を開始する場合は、マップ ノードのタイプに応じてデフォルト フィルタが自動的に設定されます。 手動で新しいトランザクション追跡セッションを開始する場合は、webMethods 用の以下のフィルタ タイプから 1 つを選択できます。

- ビジネス プロセス
- ネームスペース
- オペレーション名

フィルタを設定してからトランザクション追跡セッションを開始すると、トランザクション追跡ビューアが表示されます。 追跡を選択して、トランザクションで行われた呼び出しに関する追加の詳細情報を表示できます。 これらの詳細には、webMethods Integration Server 上で実行されたトリガ、フロー サービス オペレーション、またはビジネス プロセス手順が含まれます。

## webMethods プロセスのシーケンス ビューの使用

webMethods Integration Server が関係するトランザクションには非同期の呼び出しが含まれることが多いため、［シーケンス ビュー］をクリックして、トランザクションの一部として非同期で実行されるプロセスのトランザクション ワークフローを表示すると便利です。 シーケンス ビューには、シーケンスを識別できる範囲でプロセスの実行順序が表示されます。webMethods Integration Server のトランザクションでは、このシーケンスは必ずしも従来の呼び出し元と呼び出し先の関係を表すものではなく、1 つのプロセスが別のプロセスの実行のトリガになることを示します。-

ただし、webMethods Integration Server のプロセスの処理時間は、呼び出されたプロセスに関連する処理時間を含む、プロセスの開始から完了までの全体の継続時間を使用して計算されることに注意してください。webMethods Integration Server のプロセスについては、正味の継続時間（呼び出し元プロセスの継続時間からブロック化されていない同期および非同期プロセスの処理時間を引いたもの）はサポートされていません。

トランザクション追跡の詳細については、「SOA 環境でのトランザクション追跡の使用 (P. 111)」を参照してください。

**注**: 追跡を構成する方法の詳細については、「CA APM Java Agent 実装ガイド」または「CA APM .NET Agent 実装ガイド」を参照してください。 トランザクション追跡ビューと履歴データの使用方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# 第 12 章: WebSphere Process Server と WESB の監視

WPS（WebSphere Process Server）は、SCA（Service Component Architecture）を使用して既存アプリケーションの統合や新規アプリケーションの開発と展開を行うのに必要なメッセージング サービスとデータ管理サービスを提供する、包括的な SOA（service-oriented architecture、サービス指向アーキテクチャ）統合プラットフォームです。 SOA extension for WPS（WebSphere Process Server）を使用すると、WebSphere Process Server の主要コンポーネントを監視できます。これには、WESB（WebSphere Enterprise Service Bus）、ビジネス サービス コンポーネント、およびサポート サービスが含まれます。

このセクションでは、WebSphere Process Server または WebSphere Enterprise Service Bus 環境のパフォーマンス、可用性、および全般的な稼働状況の監視と分析に使用できる WPS および WESB 専用のダッシュボード、メトリック、およびアラートについて説明します。

このセクションには、以下のトピックが含まれています。



## WebSphere Process Server と WESB について

WPS（WebSphere Process Server）は、新しいアプリケーションとレガシー アプリケーションを統合し、疎結合サービスを展開することによってビジネス目標を実現する、包括的な SOA（service-oriented architecture、サービス指向アーキテクチャ）統合プラットフォームです。

WebSphere Process Server アーキテクチャは以下の要素で構成されます。

- サービス コンポーネント
- サポート サービス
- WebSphere Enterprise Service Bus メッセージング

WebSphere Process Server のサービス コンポーネントには、ビジネス プロセス、ビジネス ステート マシン、ビジネス ルール、ヒューマン タスクなどがあります。WebSphere Process Server のサポート サービスには、メディエーション フロー、インターフェース マップ、ビジネス オブジェクト マップ、Java コンポーネント、サービス統合バス（SIB）通信、ビジネス プロセス、ビジネス ステート マシン、リレーションシップ、セレクタ、およびアダプタがあります。 WESB （WebSphere Enterprise Service Bus）は、WebSphere Process Server を通過するメッセージの流れを管理し、サービスとクライアント間の必要なデータ変換やルーティングを処理します。

SOA extension for WebSphere Process Server を使用すると、WebSphere Enterprise Service Bus を含む WebSphere Process Server アーキテクチャのすべてを監視できます。 あるいは、WebSphere Enterprise Service Bus のみを監視することもできます。 WebSphere Enterprise Service Bus は、スタンドアロン製品としてインストールできます。

WebSphere Process Server を監視する場合は、ダッシュボードとメトリックに WPS と WESB の両方の情報が表示されます。 スタンドアロン製品としての WESB を監視する場合は、ダッシュボードとメトリックに WESB の情報のみが表示されます。

## WebSphere Process Server コンポーネントの監視

WebSphere Process Server を使用している場合は、以下のメトリックを使用して WebSphere Process Server 環境のオペレーションを監視できます。

Business Object Maps

ビジネス オブジェクト マップは、ソースのビジネス オブジェクトのサービス コンポーネントの値に応じて、ターゲットのビジネス オブジェクトのサービス コンポーネントに値を割り当てます。どれか 1 つのビジネス オブジェクトがソースとなり、別のビジネス オブジェクトがターゲットになります。 ビジネス オブジェクト マップはソースおよびターゲットをマップします。

SOA extension for WPS では、［WProcServer］-［BOMaps］ノードでビジネス オブジェクト マップのパフォーマンスと全般的な稼働状況を監視できます。

Business Processes

ビジネス プロセスは、特定の目的を達成するために特定の順序で実行される個々のタスクで構成されます。 ビジネス プロセスは、要求に対する応答のメッセージの受信によって生じる、1 つ以上のタスクで構成されます。

SOA extension for WPS では、［WProcServer］-［BusinessProcesses］ノードでビジネス プロセス内のアクティビティのパフォーマンスと全般的な稼働状況を監視できます。

Business State Machines

ビジネス ステート マシンはイベント駆動型のアプリケーションです。このアプリケーションは、イベントの発生を待機して、発生したら適切なアクションを選択して実行します。 アクションを実行したら、マシンは別のステートに移るか、別のイベントが発生するのを単に待機します。

SOA extension for WPS では、［WProcServer］-［BusinessStateMachines］ノードでイベント駆動型プロセスのパフォーマンスと全般的な稼働状況を監視できます。

Business Rules

ビジネス ルールは、ビジネス ポリシーとビジネス手法を取り込んで実装したものです。 ビジネス ルールにより、ビジネス ポリシーを適用し、判断を下し、また、既存のデータから新しい事態でのデータを推測することが可能になります。

SOA extension for WPS では、［WProcServer］-［BusinessRules］ノードで定義済みのビジネス ルールのパフォーマンスと全般的な稼働状況を監視できます。

Data Binding

データ バインディングは、インバウンドまたはアウトバウンドの処理中にビジネス オブジェクトをデータのストリームに変換するために使用されます。

SOA extension for WPS では、［WProcServer］-［DataBinding］ノードでデータ バインディングのパフォーマンスと全般的な稼働状況を監視できます。

Human Tasks

ヒューマン タスクは、人によって行われるが、何らかの形で WPS 内のプロセスやサービスとのやり取りも行われるタスクを表します。

SOA extension for WPS では、［WProcServer］-［HumansTasks］ノードで人とのやり取りを必要とするタスクのパフォーマンスと全般的な稼働状況を監視できます。

Interface Maps

インターフェース マップは、異なるインターフェースを持つコンポーネント間の通信を実現するため、変換やその他の基本的なオペレーションを使用して各コンポーネント間の違いを調整します。

SOA extension for WPS では、［WProcServer］-［InterfaceMaps］ノードでインターフェース マップのパフォーマンスと全般的な稼働状況を監視できます。

J2CA（Adapters）

アダプタは、WebSphere Process Server が外部の EIS（Enterprise Information System）と通信できるようにするアプリケーションです。アウトバウンド通信は、WebSphere Process Server が EIS 専用のオペレーションを呼び出すときに発生します。インバウンド通信は、アプリケーションが特定の EIS イベントを待機するときに発生します。

SOA extension for WPS では、［WProcServer］-［J2CA］ノードでアダプタのパフォーマンスと全般的な稼働状況を監視できます。

Java Components

Java コンポーネントは、WPS または WESB で Java の実装をカスタマイズできるようにします。

SOA extension for WPS では、［WProcServer］-［JavaComponents］ノードで Java コンポーネントのパフォーマンスと全般的な稼働状況を監視できます。

Mediation Flows

メディエーション フローは、要求を処理する要求フロー、応答を処理する応答フロー、イベントを処理するイベント フロー、および障害を処理する障害フローで構成されます。 要求フローは各ソースのオペレーションで開始され、応答フローは各ターゲットのオペレーションで開始されます。

SOA extension for WPS では、［WProcServer］-［WESB］-［MediationFlows］ノードでメディエーション フローのパフォーマンスと全般的な稼働状況を監視できます。

Mediation Primitives

メディエーション プリミティブは、メディエーション フロー コンポーネントの機能を実装する構成要素です。 各メディエーション フローには、データを変換できるメディエーション プリミティブが含まれています。

SOA extension for WPS では、［WProcServer］-［WESB］-［MediationPrimitives］ノードでメディエーション プリミティブのパフォーマンスと全般的な稼働状況を監視できます。

Relationships

リレーションシップは、複数のデータ タイプのデータ間の関連付けを確立します。 たとえば、リレーションシップを使用してカスタマとカスタマ ID の相関関係を定義できます。

SOA extension for WPS では、［WProcServer］-［Relationships］ノードでリレーションシップ定義のパフォーマンスと全般的な稼働状況を監視できます。

Selectors

セレクタは、実行中のサービス コンポーネントの処理に柔軟性を提供します。セレクタにより、クライアント アプリケーションの呼び出しに対して、実行時に選択基準に応じて異なったターゲットを呼び出せるようになります。

SOA extension for WPS では、［WProcServer］-［Selectors］ノードで定義済みのセレクタのパフォーマンスと全般的な稼働状況を監視できます。

SIB-Communication

ビジネス プロセスとビジネス ステート マシンの実行を制御する Process Server Controller コンポーネントからのサービス統合バス（SIB）メトリックを表示します。 これらのメトリックは、［WProcServer］-［SIB-Communication］ノードの下で監視できます。

## WebSphere Enterprise Service Bus スタンドアロンの監視

WebSphere Enterprise Service Bus をスタンドアロン製品として使用している場合は、以下のメトリックを使用して WebSphere Enterprise Service Bus 環境のオペレーションを監視できます。

Business Object Maps

スタンドアロン WESB 環境では、［WESB］-［BOMaps］ノードでビジネス オブジェクト マップのパフォーマンスと全般的な稼働状況を監視できます。

Data Binding

スタンドアロン WESB 環境では、［WESB］-［DataBinding］ノードでデータ バインディングのパフォーマンスと全般的な稼働状況を監視できます。

J2CA（Adapters）

スタンドアロン WESB 環境では、［WESB］-［J2CA］ノードでアダプタのパフォーマンスと全般的な稼働状況を監視できます。

Java Components

スタンドアロン WESB 環境では、［WESB］-［JavaComponents］ノードで Java コンポーネントのパフォーマンスと全般的な稼働状況を監視できます。

Mediation Flows

スタンドアロン WESB 環境では、［WESB］-［MediationFlows］ノードでメディエーション フローのパフォーマンスと全般的な稼働状況を監視できます。

Mediation Primitives

スタンドアロン WESB 環境では、［WESB］-［MediationPrimitives］ノードでメディエーション プリミティブのパフォーマンスと全般的な稼働状況を監視できます。

Relationships

スタンドアロン WESB 環境では、［WESB］-［Relationships］ノードでリレーションシップ定義のパフォーマンスと全般的な稼働状況を監視できます。

# WebSphere Process Server または WESB の監視を有効にする方法

WebSphere Process Server または WebSphere Enterprise Service Bus の監視を有効にするには、以下の手順を実行します。

1. 以下のサポートされているバージョンの WebSphere Process Server または WebSphere Enterprise Service Bus がインストールされていることを確認します。

   **注**: システム要件については、「*Compatibility Guide*」を参照してください。

2. エージェントと WSOA パフォーマンス管理がインストールされて有効になっていることを確認します。

3. エージェント プロファイルを構成して (P. 386)、エージェントで CA APM for WebSphere Process Server または CA APM for WebSphere Enterprise Service Bus を使用できるようにします。 WebSphere Process Server または WebSphere Enterprise Service Bus を監視するための適切な ProbeBuilder ディレクティブ ファイルを追加します。
4. Enterprise Manager extension for WebSphere Process Server または Enterprise Manager extension for WebSphere Enterprise Service Bus を有効にします (P. 372)。
   *<EM_Home>*/examples/SOAExtensionForWPSandWESB ディレクトリから Enterprise Manager の適切なディレクトリにファイルを移動します。

## エージェントによる WPS または WESB の監視の有効化

WebSphere Process Server の監視（WebSphere Enterprise Service Bus の監視機能を含みます）を有効にするには、エージェントのインストール時にアプリケーション サーバとして [WebSphere] を選択するか、またはエージェントのインストール後に手動で監視を有効にします。

エージェントのインストール時に CA APM for WebSphere Process Server および CA APM for WESB を選択した場合は、WebSphere Process Server と WebSphere Enterprise Service Bus の両方を監視するためのファイルが *<Agent_Home>* ディレクトリに格納されます。

エージェントのインストール後は、エージェント プロファイルを構成して CA APM for WebSphere Process Server または CA APM for WebSphere Enterprise Service Bus を手動で有効にする必要があります。

### CA APM for WPS または CA APM for WESB を手動で有効にする方法

1. エージェントと CA APM for SOA がインストールされて有効になっていることを確認します。
2. CA APM for WebSphere Process Server のディレクトリ（*SOAExtensionForWPSandWESB*）が *<Agent_Home>/examples* ディレクトリ内にあることを確認し、*<Agent_Home>/examples/SOAExtensionForWPSandWESB/ext* ディレクトリから *<Agent_Home>/core/ext* ディレクトリにファイルをコピーします。
3. *<Agent_Home>/core/config/IntroscopeAgent.profile* ファイルをテキスト エディタで開きます。

4. 適切なディレクティブ ファイルを *IntroscopeAgent.profile* ファイルの *introscope.autoprobe.directivesFile* プロパティに追加します。

   - WebSphere Enterprise Service Bus を含むすべての WebSphere Process Server コンポーネントを監視するには、*wps.pbd* を追加します。
   - スタンドアロン WESB 環境で WebSphere Enterprise Service Bus コンポーネントのみを監視するには、*wesb.pbd* を追加します。

   デフォルトの監視を変更する場合は、*wps.pbd* または *wesb.pbd* ファイルの ProbeBuilder ディレクティブをカスタマイズできます。たとえば、特定のトレーサ グループの追跡をオンまたはオフにすることで、特定のコンポーネントの監視を細かく調整できます。

5. *IntroscopeAgent.profile* ファイルを保存して閉じます。

## Business Process Manager 8.0 および 8.1 に必要な設定

Business Process Manager 8.0 および 8.1 に対して、wps.pbd を設定して追跡を有効にします。

次の手順に従ってください：

1. Business Process Manager が実行されている場合は停止します。
2. <Agent_Home>/core/config ディレクトリに移動し、wps.pbd ファイルを開きます。

   **注**: レガシー モードを使用している場合は、同じ手順で wps-legacy.pbd ファイルを変更します。

3. pbd ファイルの以下のトレーサ マッピング ステートメントおよびトレーサ ディレクティブ ステートメントをコメント化します。

```
#SetTracerClassMapping:  ThreadCreationTracer
com.wily.powerpack.websphereprocserver.tracer.hc2.ThreadCreationTracer
com.wily.introscope.probebuilder.validate.ResourceNameValidator

#TraceOneMethodWithParametersOfClass:
com.ibm.bpe.framework.navigation.NavigationWorker doWork ThreadCreationTracer
"Navigation-Work"
```

4. pbd ファイルの以下のトレーサ マッピングおよびトレーサ ディレクティブのコメント化を解除します。

```
SetTracerClassMapping: ThreadCreationTracer80
com.wily.powerpack.websphereprocserver.tracer.hc2.ThreadCreationTracer80
com.wily.introscope.probebuilder.validate.ResourceNameValidator

TraceOneMethodWithParametersOfClass:
com.ibm.bpe.framework.navigation.NavigationWorker doWork
ThreadCreationTracer80 "Navigation-Work"
```

5. wps.pbd を保存して閉じます。

デフォルト設定の変更が完了したら、Business Process Manager を再起動します。

## 複数バージョンのビジネス プロセスおよびビジネス ステート マシンに関するメトリックの収集

wps.pbd を設定して、異なるバージョンのビジネス プロセスおよびビジネス ステート マシンに関するメトリックを個別に提供できます。 各ビジネス プロセスおよびビジネス ステート マシンのバージョンを 1 つ管理する場合は、以下の設定手順をスキップしてもかまいません。

**次の手順に従ってください**:

1. *<Agent_Home>*/core/config ディレクトリに移動し、wps.pbd ファイルを開きます。
2. ProcessCustomTracer パラメータと ProcessFaultTracer パラメータをコメント化します。 例：

```
##SetTracerParameter: ProcessCustomTracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.ProcessContextFormatter
#SetTracerParameter: ProcessFaultTracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.ProcessContextFormatter
#SetTracerParameter: BPMapTracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.ProcessContextFormatter
#SetTracerParameter: BusinessProcessTracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.BusinessProcessNameAndTi
meFormatter
#SetTracerParameter: BusinessProcessFaultTracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.BusinessProcessNameAndTi
meFormatter
#SetTracerParameter: BPMap7Tracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.BusinessProcessNameAndTi
meFormatter
```

3. 時間属性を取得するため、ProcessCustomTracer パラメータと ProcessFaultTracer パラメータのコメント化を解除します。 例：

```
SetTracerParameter: ProcessCustomTracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.ProcessAndTimeNameFormat
ter
SetTracerParameter: ProcessFaultTracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.ProcessAndTimeNameFormat
ter
SetTracerParameter: BPMapTracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.ProcessAndTimeNameFormat
ter
SetTracerParameter: BusinessProcessTracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.BusinessProcessNameForma
tter
SetTracerParameter: BusinessProcessFaultTracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.BusinessProcessNameForma
tter
SetTracerParameter: BPMap7Tracer nameformatter
com.wily.powerpack.websphereprocserver.nameformatter.BusinessProcessNameForma
tter
```

   **注**: この手順を省略した場合は、ビジネス プロセスおよびビジネス ステート マシン ノードが、ファイルの開発時に使用された時間形式で追加されます。

4. wps.pbd を保存して閉じます。

   デフォルト構成の変更が完了したら、WebSphere Process Server または WESB サーバを再起動できます。

## PMI（Performance Monitoring Infrastructure）メトリックの収集

分散環境の WebSphere Application Server を監視する場合は、WebSphere PMI（Performance Monitoring Infrastructure）メトリックを収集し、Investigator に WebSphere Process Server コンポーネントに関する PMI メトリックを表示することもできます。

そのためには、まず「CA APM for IBM WebSphere Distributed Environments ガイド」の説明に従って監視を構成する必要があります。 次に、アプリケーション サーバ上の追加のエージェント プロパティを変更して、WebSphere Process Server コンポーネントの監視を有効にします。

**次の手順に従ってください**:

1. *<Agent_Home>*/core/config の *IntroscopeAgent.profile* ファイルを開きます。
2. 以下のプロパティを検索します。
   *introscope.agent.pmi.enable.WBIStats.RootGroup*
   *introscope.agent.pmi.enable.SCAStats.RootGroup*
3. PMI メトリックのレポートを有効にするには、これらのプロパティを両方とも true に設定します。 例 :
   introscope.agent.pmi.enable.WBIStats.RootGroup=true
   introscope.agent.pmi.enable.SCAStats.RootGroup=true

   これらのプロパティが現在定義されていない場合は、これらのプロパティを *IntroscopeAgent.profile* に手動で追加し、true に設定してください。
4. *IntroscopeAgent.profile* ファイルを保存して閉じます。

   デフォルト構成の変更が完了したら、WebSphere Process Server または WESB サーバを再起動できます。
5. WebSphere 管理コンソールを使用して、WebSphere Process Server および WebSphere Enterprise Service Bus モジュールの Performance Monitoring Infrastructure を有効にします。

   **注**: WebSphere 管理コンソールを使用してモジュールを有効にする方法については、IBM WebSphere のマニュアルを参照してください。

## Enterprise Manager Extension for WPS または Enterprise Manager Extension for WESB を有効にする

Enterprise Manager をインストールすると、CA APM for WebSphere Process Server および CA APM for WebSphere Enterprise Service Bus のファイルが *<EM_Home>/examples* ディレクトリにデフォルトでインストールされます。CA APM for WebSphere Process Server または CA APM for WebSphere Enterprise Service Bus を有効にするには、Enterprise Manager のファイルを *<EM_Home>/examples* ディレクトリから Enterprise Manager のホーム ディレクトリ内の適切な場所にコピーまたは移動します。

**注**: CA APM for WebSphere Process Server または CA APM for WebSphere Enterprise Service Bus を使用するには、あらかじめ CA APM for SOA を Enterprise Manager で有効 (P. 44)にしておく必要があります。

### Enterprise Manager extension for WebSphere Process Server を有効にする方法

1. CA APM for WebSphere Process Server のディレクトリ（SOAExtensionForWPSandWESB）が *<EM_Home>*/examples ディレクトリに存在することを確認します。 ファイルを *<EM_Home>*/examples/SOAExtensionForWPSandWESB ディレクトリから Enterprise Manager ディレクトリ構造内の対応する場所にコピーします。たとえば、*<EM_Home>*/examples/SOAExtensionForWPSandWESB/ext ディレクトリからファイルを *<EM_Home>/ext* ディレクトリにコピーします。
2. *<EM_HOME>*/examples/SOAExtensionForWPSandWESB/config/modules ディレクトリの *WPS_Management_Module.jar* ファイルをコピーし、それを *<EM_Home>*/config//modules ディレクトリに貼り付けます。

   *<EM_HOME>*/examples/SOAExtensionForWPSandWESB/config/modules ディレクトリには、WebSphere Process Server 用と WebSphere Enterprise Service Bus スタンドアロン用の両方の管理モジュールが格納されています。 WebSphere Enterprise Service Bus と共に WebSphere Process Server を監視する場合のみ、*WPS_Management_Module.jar* ファイルをコピーします。

   クラスタ環境に複数の Enterprise Manager がある場合は、MOM コンピュータとして使用している Enterprise Manager の *<EM_Home>*/config/modules ディレクトリに、このファイルのみをコピーしてください。 他のすべてのファイルとスクリプトは、コレクタ Enterprise Manager と MOM Enterprise Manager の両方にインストールする必要があります。
3. 以前のバージョンの CA APM for WebSphere Process Server からアップグレードする場合は、古いバージョンの Enterprise Manager ファイルを削除します。

   Enterprise Manager 上に以前のリリースがインストールされている場合は、新しいバージョンの CA APM for WebSphere Process Server を使用し始める前に、古いバージョンの Enterprise Manager ファイルを手動で削除する必要があります。

   以前のリリースからアップグレードした場合は、Enterprise Manager のホーム ディレクトリから以下のファイルを削除します。

   - *<EM_home>*/config/modules/WPS_Management_ModuleV*<version>*.jar
   - *<EM_home>*/product/enterprisemanager/plugins/com.wily.powerpack.websphereprocserver.em.ext_*<version>*.jar

たとえば、バージョン 8.1 からアップグレードしている場合は、以下のファイルを削除します。

- WPS_Management_ModuleV8.1.0.0.jar
- com.wily.powerpack.websphereprocserver.em.ext_8.1.0.jar

4. Workstation を再起動します。

   SOA extension for WebSphere Process Server または SOA extension for WebSphere Enterprise Service Bus 専用のダッシュボードと［概要］タブがロードされます。

### WESB スタンドアロン用の Enterprise Manager 拡張を有効にする方法

1. CA APM for WebSphere Process Server のディレクトリ（SOAExtensionForWPSandWESB）が *<EM_Home>/examples* ディレクトリに存在することを確認します。 ファイルを *<EM_Home>*/examples/SOAExtensionForWPSandWESB ディレクトリから Enterprise Manager ディレクトリ構造内の対応する場所にコピーします。たとえば、*<EM_Home>*/examples/SOAExtensionForWPSandWESB/ext ディレクトリからファイルを *<EM_Home>*/ext ディレクトリにコピーします。

2. *<EM_HOME>*/examples/SOAExtensionForWPSandWESB/config/modules ディレクトリの *WESB_Management_Module.jar* ファイルをコピーし、それを *<EM_Home>*/config//modules ディレクトリに貼り付けます。

   *<EM_HOME>*/examples/SOAExtensionForWPSandWESB/config/modules ディレクトリには、WebSphere Process Server 用と WebSphere Enterprise Service Bus スタンドアロン用の両方の管理モジュールが格納されています。WebSphere Enterprise Service Bus スタンドアロンを監視する場合、*WESB_Management_Module.jar* ファイルのみをコピーします。

   クラスタ環境に複数の Enterprise Manager がある場合は、MOM コンピュータとして使用している Enterprise Manager の *<EM_Home>*/config/modules ディレクトリに、このファイルのみをコピーしてください。 他のすべてのファイルとスクリプトは、コレクタ Enterprise Manager と MOM Enterprise Manager の両方にインストールする必要があります。

3. 以前のバージョンの CA APM for WebSphere Enterprise Service Bus からアップグレードする場合は、古いバージョンの Enterprise Manager ファイルを削除します。

Enterprise Manager 上に以前のリリースがインストールされている場合は、新しいバージョンの CA APM for WESB を使用し始める前に、古いバージョンの Enterprise Manager ファイルを手動で削除する必要があります。以前のリリースからアップグレードした場合は、Enterprise Manager のホーム ディレクトリから以下のファイルを削除します。

- <*EM_home*>/config/modules/WESB_Management_ModuleV<*version*>.jar
- <*EM_home*>/product/enterprisemanager/plugins/com.wily.powerpack.websphereprocserver.em.ext_<*version*>.jar

たとえば、バージョン 8.1 からアップグレードしている場合は、以下のファイルを削除します。

- WESB_Management_ModuleV8.1.0.0.jar
- com.wily.powerpack.websphereprocserver.em.ext_8.1.0.jar

4. Workstation を再起動します。

   CA APM for WebSphere Process Server または CA APM for WebSphere Enterprise Service Bus 専用のダッシュボードと［概要］タブがロードされます。

# ダッシュボードを使用した WPS または WESB の監視

SOA extension for WebSphere Process Server には、アプリケーション環境の全般的な稼働状況を監視するために使用できる事前設定済みのダッシュボードがいくつか含まれています。 ダッシュボードは、ユーザが問題をすばやく診断して解決できるように、展開されたエージェントからデータを集計してパフォーマンス情報を要約します。

通常、ダッシュボードは以下の機能を備えているため、環境を監視するための起点として使用されます。

- WebSphere Process Server の主要コンポーネントの全般的な稼働状況、パフォーマンス、可用性、および現在のステータスを一覧形式で監視する機能
- 低いレベルのメトリックによって警告しきい値や危険しきい値を超えたことが示されたときに、実運用アプリケーション環境に発生する可能性がある問題の通知を早めに取得する。
- パフォーマンス情報にドリルダウンして、WebSphere のどのビジネス プロセス、ビジネス ルール、メディエーション フロー、またはその他のコンポーネントに遅延やエラーが発生しているかを特定する機能

事前設定済みの WebSphere Process Server および WebSphere Enterprise Service Bus ダッシュボードは、SOA Extension for WebSphere Process Server 管理モジュール（*WPS_Management_Module.jar*）または SOA Extension for WebSphere Enterprise Service Bus 管理モジュール（*WESB_Management_Module.jar*）の一部として WebSphere 用の Enterprise Manager 拡張にパッケージ化されています。

## WebSphere Process Server のダッシュボードについて

WebSphere Process Server 管理モジュールには、WebSphere Process Server のために以下の事前設定済みのダッシュボードが用意されています。

**WPS - 概要**

WebSphere Process Server の稼働状況に関するトップレベルの概要。これには、WebSphere Process Server アーキテクチャに含まれるすべての主要コンポーネントの全体的な応答時間、エラー数、およびストール数に関するアラート インジケータが含まれます。

**WPS - ビジネス プロセス & ステート マシン**

すべてのビジネス プロセスとビジネス ステート マシンに関する要約されたステータス。これには、すべてのビジネス プロセスとビジネス ステート マシンに関する平均応答時間のグラフ、エラー数とストール数に関するアラート インジケータ、および最も遅いビジネス プロセスとビジネス ステート マシンのリストが含まれます。

### WPS - ビジネス ルール & ヒューマン タスク

すべてのビジネス ルールとヒューマン タスクに関する要約されたステータス。これには、すべてのビジネス ルールとヒューマン タスクに関する平均応答時間のグラフ、エラー数とストール数に関するアラート インジケータ、および最も遅いビジネス ルールとヒューマン タスクのリストが含まれます。

### WPS - インターフェース マップ、BO マップ、リレーションシップ

すべてのインターフェース マップ、ビジネス オブジェクト マップ、およびリレーションシップに関する要約されたステータス。これには、インターフェース マップ、ビジネス オブジェクト マップ、およびリレーションシップに関する平均応答時間のグラフ、エラー数とストール数に関するアラート インジケータ、および最も遅いインターフェース マップ、ビジネス オブジェクト マップ、リレーションシップのリストが含まれます。

### WPS - セレクタ & Java コンポーネント

すべてのセレクタと Java コンポーネントに関する要約されたステータス。これには、すべてのセレクタと Java コンポーネントに関する平均応答時間のグラフ、エラー数とストール数に関するアラート インジケータ、および最も遅いセレクタと Java コンポーネントのリストが含まれます。

### WPS - データ バインディング

すべてのデータ バインディングに関する要約されたステータス。これには、平均応答時間、Errors Per Interval （間隔ごとのエラー数）、Stall Count （ストール数） に関するグラフとアラート インジケータ、および最も遅いデータ バインディングのリストが含まれます。

### WPS - アダプタ

すべてのインバウンド アダプタとアウトバウンド アダプタに関する要約されたステータス。これには、すべてのインバウンド アダプタとアウトバウンド アダプタに関する平均応答時間のグラフ、エラー数とストール数に関するアラート インジケータ、および最も遅いインバウンド アダプタとアウトバウンド アダプタのリストが含まれます。

WESB - メディエーション フロー

メディエーション フローの全体的な稼働状況に関する要約されたステータス、および要求フロー、応答フロー、障害フローに関する個別のグラフとアラート インジケータ。

メディエーション フローは要求フロー、応答フロー、および障害フローで構成されるため、このダッシュボードにはメディエーション フロー、要求フロー、応答フロー、障害フローに関する平均応答時間のグラフ、メディエーション フロー、要求フロー、応答フロー、障害フローのエラー数とストール数に関するアラート インジケータ、および最も遅いメディエーション フローのリストが表示されます。

WESB - メディエーション プリミティブ

メディエーション プリミティブの全体的な稼働状況に関する要約されたステータス。これには、すべてのメディエーション プリミティブの応答時間とエラー数に関するアラート インジケータ、および要求フロー、応答フロー、障害フローの応答時間、エラー数、ストール数に関する個別のアラート インジケータが含まれます。

## WESB ダッシュボードについて

スタンドアロン製品としての WebSphere Enterprise Service Bus を監視する場合は、以下のダッシュボードのみが使用可能です。

WESB - 概要

WebSphere Enterprise Service Bus の稼働状況に関するトップレベルの概要。これには、WebSphere アーキテクチャの WebSphere Enterprise Service Bus 層に含まれるすべての主要コンポーネントの全体的な応答時間、エラー数、およびストール数に関するアラート インジケータが含まれます。

WESB - メディエーション フロー

WebSphere Enterprise Service Bus のメディエーション フローの全体的な稼働状況に関する要約されたステータス、および要求フロー、応答フロー、障害フローに関する個別のグラフとアラート インジケータ。

### WESB - メディエーション プリミティブ

メディエーション プリミティブの全体的な稼働状況に関する要約されたステータス。これには、WebSphere Enterprise Service Bus のすべてのメディエーション プリミティブの応答時間とエラー数に関するアラート インジケータ、および要求フロー、応答フロー、障害フローの応答時間、エラー数、ストール数に関する個別のアラート インジケータが含まれます。

### WESB - アダプタ

すべてのインバウンド アダプタとアウトバウンド アダプタに関する要約されたステータス。これには、WebSphere Enterprise Service Bus のすべてのインバウンド アダプタとアウトバウンド アダプタに関する平均応答時間のグラフ、エラー数とストール数に関するアラート インジケータ、および最も遅いインバウンド アダプタとアウトバウンド アダプタのリストが含まれます。

### WESB - BO マップ & リレーションシップ

すべてのビジネス オブジェクト マップとリレーションシップに関する要約されたステータス。これには、WebSphere Enterprise Service Bus のすべてのビジネス オブジェクト マップとリレーションシップに関する平均応答時間のグラフ、エラー数とストール数に関するアラート インジケータ、および最も遅いビジネス オブジェクト マップとリレーションシップのリストが含まれます。

WESB - データ バインディング

すべてのデータ バインディングに関する要約されたステータス。これには、WebSphere Enterprise Service Bus の平均応答時間、Errors Per Interval （間隔ごとのエラー数）、Stall Count （ストール数） に関するグラフとアラート インジケータ、および最も遅いデータ バインディングのリストが含まれます。

WESB - Java コンポーネント

すべての Java コンポーネントに関する要約されたステータス。これには、WebSphere Enterprise Service Bus のすべての Java コンポーネントの平均応答時間、Errors Per Interval（間隔ごとのエラー数）、Stall Count（ストール数）に関するグラフとアラート インジケータ、および最も遅い Java コンポーネントのリストが含まれます。

WebSphere Enterprise Service Bus のダッシュボードは、WebSphere Process Server のダッシュボードに似ていますが、WebSphere Enterprise Service Bus コンポーネント専用のアラート インジケータとメトリックのみを提供します。

## WebSphere Process Server または WESB のダッシュボードを表示する

WebSphere Process Server または WebSphere Enterprise Service Bus のダッシュボードは、システムの稼働状況を評価するのに役立つアラート インジケータとメトリックを一覧形式で提供します。

**次の手順に従ってください**:

1. Enterprise Manager を起動します（現在実行されていない場合）。
2. Workstation を起動し、SOA extension for WebSphere Process Server または SOA extension for WESB がインストールされている Enterprise Manager にログインします。
3. ［Workstation］-［新規コンソール］を選択します。

4. ダッシュボードのドロップダウン リストから WPS または WESB のいずれかのダッシュボードを選択します。

   たとえば、すべてのサービス コンポーネントとサポート サービスに関するアラート インジケータを含む WebSphere Process Server の稼働状況の概要を表示するには、［**WPS - 概要**］ダッシュボードを選択します。

   ［WPS - 概要］または［WESB - 概要］ダッシュボードから他のダッシュボードに移動するには、表示するダッシュボードの名前を持つタブをダブルクリックします。

5. 別のタブまたはダッシュボード内のアラートをダブルクリックすると、関連するダッシュボードが開き、詳細情報が表示されます。

   たとえば、［ビジネス プロセス］の［応答時間］アラートをダブルクリックすると、［**WPS - ビジネス プロセス & ステート マシン**］ダッシュボードに最も時間のかかるビジネス プロセスに関する詳細情報が表示されます。

   ［WPS - ビジネス プロセス & ステート マシン］ダッシュボードには、最も遅いビジネス プロセスとステート マシンのリスト、平均応答時間のグラフ、およびエラー数とストール数に関するアラート インジケータが表示されます。

6. ［WPS - ビジネス プロセス & ステート マシン］ダッシュボードで、最も遅いビジネス プロセスの一覧に表示された特定のビジネス プロセス名をダブルクリックすると、さらに詳しく分析できるように、そのビジネス プロセスの応答時間が選択された状態で Investigator が表示されます。

**注**: Workstation の起動と使い方、ダッシュボードへのアクセス、または Investigator の起動と操作の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# WPS/WESB のメトリックを表示および操作する

Investigator ツリーを操作すると、WebSphere Process Server または WebSphere Enterprise Service Bus インフラストラクチャのほとんどのコンポーネントに関する CA Introscope® の標準メトリックが表示されます。収集された標準メトリックのデータは、Investigator ツリーのノードおよびサブノードとして表示される WebSphere Process Server または WESB 専用のメトリック カテゴリに集計されます。-- 表示される具体的なメトリック カテゴリとノード名は、環境内に展開してアクセスしたコンポーネント、サービス、およびリソースによって異なります。

Investigator ツリーを操作しながら、選択するノードに応じて個々のオペレーションに関する低レベルのメトリックまたは集計メトリックを表示できます。これにより、さまざまな Process Server または WESB コンポーネントの全般的な稼働状況を監視できます。 コンポーネントがまったく使用されていない場合（たとえば、どのプロセスにもヒューマン タスクが定義されていない場合）、そのコンポーネントに関するメトリック カテゴリは Investigator ツリーに表示されません。

### Investigator でメトリックのサマリを表示および操作する方法

1. エージェント ノードを展開して［WProcServer］または［WESB］ノードをクリックすると、サマリ情報を示す［概要］タブが表示されます。

   たとえば、［WProcServer］ノードを選択すると、WebSphere Process Server が使用しているすべてのビジネス プロセス、ビジネス ステート マシン、およびメディエーション フローのサマリ情報が［概要］に一覧表示されます。

2. ビジネス プロセス、ビジネス ステート マシン、またはメディエーション フローをダブルクリックすると、関連するすべてのメトリックがグラフィカルな形式で表示されます。

3. ［WProcServer］または［WESB］ノードを展開すると、WebSphere Process Server または WebSphere Enterprise Service Bus に関するトップレベルのメトリック カテゴリのサブノードが表示されます。

4. サブノードをクリックまたは展開すると、そのメトリック カテゴリに関するサマリ情報を含む［概要］タブが表示されます。 たとえば、［BusinessRules］ノードをクリックすると、［概要］タブにビジネス ルールのリストが表示されます。

5. いずれかのサブノードを展開すると、個々のコンポーネントに関する詳細情報が表示されます。たとえば、特定のビジネス プロセス名を選択すると、そのビジネス プロセスに関するメトリックがグラフィカルな形式で表示されます。
6. 個々のオブジェクトを展開すると、そのオブジェクトに関するメトリックが表示されます。たとえば、個々の Java コンポーネント名を展開し、さらにオペレーションを展開すると、そのオペレーションに関するメトリックが表示されます。

## ビジネス オブジェクト マップに関するメトリック

ビジネス オブジェクト マップは、ソースのビジネス オブジェクトのサービス コンポーネントの値に応じて、ターゲットのビジネス オブジェクトのサービス コンポーネントに値を割り当てます。

ターゲットのネームスペースと個々のビジネス オブジェクト マップについては、［WProcServer］-［BO マップ］または［WESB］ -［BO マップ］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

## ビジネス プロセスに関するメトリック

ビジネス プロセスは、特定のビジネス目標を達成するために特定の順序で実行される個々のタスクで構成されます。 このメトリック カテゴリは WebSphere Process Server にのみ適用されます。 スタンドアロン製品としての WebSphere Enterprise Service Bus を監視する場合には適用されません。

WebSphere の個々のビジネス プロセスおよびビジネス プロセス手順については、［WProcServer］-［ビジネス プロセス］-［*<business_process_name>*］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

「複数バージョンのビジネス プロセスに関するメトリックの収集 (P. 370)」で説明したように、同じビジネス プロセスの異なるバージョンに関して別個のメトリックを表示するように追跡を設定した場合は、同じプロセスの異なるバージョンを識別するため、［BusinessProcesses］ノードの下に表示されるビジネス プロセス名の末尾にアンダースコア（_）とタイム スタンプが追加されます。

## ビジネス ルールに関するメトリック

ビジネス ルールは、ビジネス ポリシーとビジネス手法を取り込んで実装したものです。 このメトリック カテゴリは WebSphere Process Server にのみ適用されます。 スタンドアロン製品としての WebSphere Enterprise Service Bus を監視する場合には適用されません。

展開されている個々のビジネス ルールについては、［WProcServer］-［ビジネス ルール］-［*<rule_name>*］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

## ビジネス ステート マシンに関するメトリック

ビジネス ステート マシンは、イベントに対してアクションを実行するイベント駆動型アプリケーションです。 このメトリック カテゴリは WebSphere Process Server にのみ適用されます。 このカテゴリは、スタンドアロン製品としての WebSphere Enterprise Service Bus を監視する場合には適用されません。

個々のビジネス ステート マシンについては、［WProcServer］-［ビジネス ステート マシン］-［*<business_state_machine_name>*］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

同じビジネス ステート マシンの異なるバージョン用の個別のメトリックを表示するために追跡を構成する (P. 370)場合、［ビジネス ステート マシン］ノードの下に名前が以下のように表示されます。

- 名前の末尾にアンダースコア［_］が追加されます。
- タイム スタンプは、同じプロセスの別のバージョンを識別します。

## データ バインディングに関するメトリック

データ バインディングは、インバウンドまたはアウトバウンドの処理中にビジネス オブジェクトをデータのストリームに変換するために使用されます。

個々のデータ バインディングについては、［WProcServer］-［データ バインディング］ -［*<data_binding_name>*］または［WESB］-［データ バインディング］ -［*<data_binding_name>*］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

## ヒューマン タスクに関するメトリック

ヒューマン タスクは、人によって行われるが、WebSphere Process Server 内のプロセスやサービスとのやり取りも行われるタスクを表します。 このメトリック カテゴリは WebSphere Process Server にのみ適用されます。 スタンドアロン製品としての WebSphere Enterprise Service Bus を監視する場合には適用されません。

個々のヒューマン タスクについては、［WProcServer］-［ヒューマン タスク］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

## インターフェース マップに関するメトリック

インターフェース マップは、異なるインターフェースを持つコンポーネント間の通信を実現するため、各コンポーネント間の違いを調整します。 このメトリック カテゴリは WebSphere Process Server にのみ適用されます。 スタンドアロン製品としての WebSphere Enterprise Service Bus を監視する場合には適用されません。

定義済みのインターフェース マップについては、［WProcServer］-［インターフェース マップ］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

同期呼び出しの場合、個々のインターフェース マップに関するメトリックは、そのインターフェース マップ コンポーネントに費やされた合計時間として表示されます。 非同期呼び出しの場合、これらのメトリックは入力変換と出力変換で別々に表示されます。

## J2CA アダプタに関するメトリック

アダプタは、WebSphere Process Server が外部の Enterprise Information System と通信できるようにするアプリケーションです。 アウトバウンド通信は、WebSphere Process Server が EIS 専用のオペレーションを呼び出すときに発生します。 インバウンド通信は、アプリケーションが特定の EIS イベントを待機するときに発生します。

WebSphere server と外部システム間のインバウンドおよびアウトバウンドのインタラクションについては、［WProcServer］-［J2CA］または［WESB］-［J2CA］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

## Java コンポーネントに関するメトリック

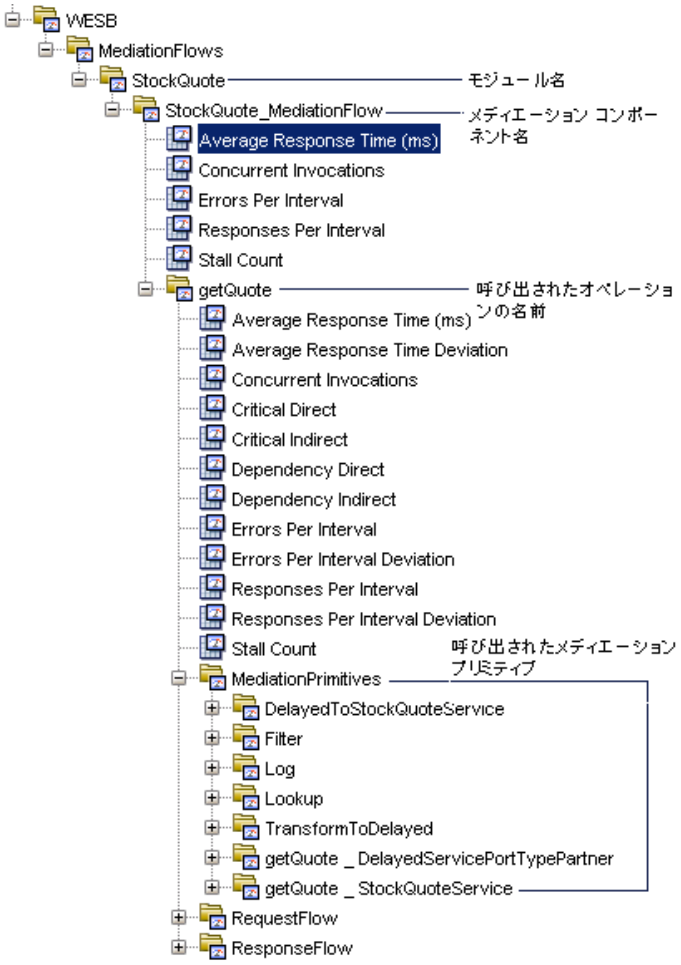
Java コンポーネントは、WPS または WESB で Java の実装をカスタマイズできるようにします。

個々の Java コンポーネントとオペレーションについては、［WProcServer］-［Java コンポーネント］または［WESB］-［Java コンポーネント］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

## メディエーション フローおよびメディエーション プリミティブに関するメトリック

メディエーション フローは、要求を処理する要求フロー、応答を処理する応答フロー、イベントを処理するイベント フロー、および障害を処理する障害フローで構成されます。 要求フローは各ソースのオペレーションで開始され、応答フローは各ターゲットのオペレーションで開始されます。 メディエーション プリミティブは、メディエーション フローの機能を実装する構成要素です。 各メディエーション フローには、必要に応じてデータを変換するメディエーション プリミティブが含まれています。

メディエーション フローおよびメディエーション プリミティブについては、［WProcServer］-［WESB］-［メディエーション フロー］または［WESB］-［メディエーション フロー］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。 ［MediationFlows］ノードには、メディエーション フロー コンポーネントを持つモジュールが一覧表示されます。 メディエーション フロー コンポーネントを展開すると、そのメディエーション フローと、そのオペレーション、フロー、およびメディエーション プリミティブに関する集計メトリックが表示されます。 例：



応答フロー、イベントフロー、または障害フローが非同期呼び出しとして呼び出された場合は、Investigator に表示されるノード名が *<flow_type>*_Asynch になります。*<flow_type>* は、フローを ResponseFlow（応答フロー）、EventFlow（イベント フロー）、または FaultFlow（障害フロー）として識別します。 たとえば、応答フローがコールバックで非同期に呼び出された場合、Investigator に表示されるノード名は ResponseFlow_Asynch になります。

Mediation Flow メトリックのコンテキストでは、非同期呼び出しはコールバック機能を使用して非同期で呼び出されたオペレーションです。

## 同期および非同期の応答時間に関するメトリックについて

メディエーション フロー コンポーネントに関するメトリックは、フロー名の下にあるオペレーションに関するすべての Average Response Time（平均応答時間）メトリックから集計されます。 メディエーション フロー オペレーションに関するメトリックは、オペレーションが同期呼び出しと非同期呼び出しのどちらを使用するかによって、異なる方法で集計されます。

- オペレーションが同期呼び出しを行う場合は、オペレーションの要求フローからオペレーション レベルのメトリックが集計されます。これは、要求フローに同期呼び出しの応答フローの継続時間が間接的に含まれているためです。
- オペレーションが非同期呼び出しを行う場合は、オペレーション名の下にある既存の要求フロー（RequestFlow）と非同期の応答フロー（ResponseFlow_Asynch）、障害フロー（FaultFlow_Asynch）、およびイベント フロー（EventFlow_Asynch）からオペレーション レベルのメトリックが集計されます。

たとえば、ある間隔の Average Response Time （平均応答時間） が要求フローで 10 ミリ秒、非同期応答フローで 12 ミリ秒、非同期障害フローで 8 ミリ秒、非同期イベント フローで 10 ミリ秒であり、カウント値が要求フローで 1、応答フローで 1、障害フローで 2、イベント フローで 3 だった場合、このオペレーションの Average Response Time （平均応答時間） は以下のように計算されます。

((10 x 1) + (12 x 1) + (8 x 2) + (10 x 3))/3

## 同期および非同期の同時進行中の呼び出しについて

メディエーション フロー コンポーネントに関するメトリックは、フロー名の下にあるオペレーションに関するすべての同時進行中の呼び出しメトリックから集計されます。 メディエーション フロー オペレーションに関するメトリックは、オペレーションが同期呼び出しと非同期呼び出しのどちらを使用するかによって、異なる方法で集計されます。

- オペレーションが同期呼び出しを行う場合は、オペレーション レベルの Concurrent Invocations （同時進行中の呼び出し） メトリックがオペレーションの要求フローの Concurrent Invocations （同時進行中の呼び出し） を表します。これは、要求フローに同期呼び出しの応答フローの継続時間が間接的に含まれているためです。
- オペレーションが非同期呼び出しを行う場合は、オペレーション名の下にある既存の要求フロー（RequestFlow）と非同期の応答フロー（ResponseFlow_Asynch）、障害フロー（FaultFlow_Asynch）、およびイベント フロー（EventFlow_Asynch）の Concurrent Invocations （同時進行中の呼び出し） の呼び出し からオペレーション レベルのメトリックが集計されます。

## 同期および非同期のエラーに関するメトリックについて

メディエーション フロー コンポーネントに関するメトリックは、フロー名の下にあるオペレーションに関するすべての Errors Per Interval（間隔ごとのエラー数） メトリックから集計されます。 メディエーション フロー オペレーションに関する Errors Per Interval （間隔ごとのエラー数） メトリックは、オペレーションが同期呼び出しと非同期呼び出しのどちらを使用するかによって、異なる方法で計算されます。

- オペレーションが同期呼び出しを行う場合は、要求フローの Errors Per Interval（間隔ごとのエラー数）メトリックを加算することによってオペレーション レベルのメトリックが集計されます。
- オペレーションが非同期呼び出しを行う場合は、オペレーション名の下にある既存の要求フロー（RequestFlow）と非同期の応答フロー（ResponseFlow_Asynch）およびイベント フロー（EventFlow_Asynch）の既存の Errors Per Interval （間隔ごとのエラー数） メトリックを非同期の障害フロー（FaultFlow_Asynch）の Responses Per Interval （間隔ごとの応答数）メトリックに加算することによってオペレーション レベルのメトリックが集計されます。

障害フローは障害発生時に発生すると想定されるため、非同期の障害フローの Responses Per Interval （間隔ごとの応答数） メトリック値をオペレーション レベルの Errors Per Interval （間隔ごとのエラー数） メトリック値に加算することにより、そのオペレーションの呼び出し中にエラーが生じたことが表されます。

## 同期および非同期の応答に関するメトリックについて

メディエーション フロー コンポーネントに関するメトリックは、フロー名の下にあるオペレーションに関するすべての Responses Per Interval （間隔ごとの応答数）メトリックから集計されます。メディエーション フロー オペレーションに関する Responses Per Interval （間隔ごとの応答数） メトリックは、オペレーションが同期呼び出しと非同期呼び出しのどちらを使用するかによって、異なる方法で計算されます。

- オペレーションが同期呼び出しを行う場合は、要求フローの Responses Per Interval （間隔ごとの応答数） を加算することによってオペレーション レベルのメトリックが集計されます。
- オペレーションが非同期呼び出しを行う場合は、オペレーション名の下にある要求フロー（RequestFlow）と非同期の応答フロー（ResponseFlow_Asynch）、障害フロー（FaultFlow_Asynch）、およびイベント フロー（EventFlow_Asynch）の Responses Per Interval （間隔ごとの応答数） の最大値によって 間隔ごとの応答数 メトリックが決定されます。

## 同期および非同期のストール数メトリックについて

メディエーション フロー コンポーネントに関するメトリックは、フロー名の下にあるオペレーションに関するすべての Stall Count （ストール数） メトリックから集計されます。 メディエーション フロー オペレーションに関する Stall Count （ストール数） メトリックは、オペレーションが同期呼び出しと非同期呼び出しのどちらを使用するかによって、異なる方法で計算されます。

- オペレーションが同期呼び出しを行う場合は、要求フローで記録されたストールの数を合計することによって Stall Count （ストール数） が計算されます。
- オペレーションが非同期呼び出しを行う場合は、オペレーション名の下にある要求フローと非同期の応答フロー（ResponseFlow_Asynch）、障害フロー（FaultFlow_Asynch）、およびイベント フロー（EventFlow_Asynch）で記録された Stall Count （ストール数） を合計することによってストール数が計算されます。

## リレーションシップに関するメトリック

リレーションシップは、意味的に同等のビジネス オブジェクトを識別します。

個々のリレーションシップについては、［WProcServer］-［リレーションシップ］または［WESB］-［リレーションシップ］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

## セレクタに関するメトリック

セレクタは、実行中のサービス コンポーネントの処理に柔軟性を提供します。- このメトリック カテゴリは WebSphere Process Server にのみ適用されます。 スタンドアロン製品としての WebSphere Enterprise Service Bus を監視する場合には適用されません。

［WProcServer］-［セレクタ］ノードの下に CA Introscope® のすべての標準メトリックが表示されます。

## サービス統合バス通信に関するメトリック

サービス統合バス（SIB）は、メッセージ ベースのアーキテクチャおよびサービス指向アーキテクチャを使用してアプリケーションをサポートします。 SIB 通信メトリックは、［WProcServer］-［SIB-Communication］ノードの下に表示できます。

## WebSphere Process Server の障害に関するメトリック

WPS 層で障害が発生し、その障害が処理されないときは例外がスローされます。 WPS Faults Per Interval メトリックは、15 秒のタイム スライスで発生した例外の数を示します。 このメトリックは WebSphere Process Server にのみ適用されます。 スタンドアロン製品としての WebSphere Enterprise Service Bus を監視する場合には適用されません。

# WPS および WESB のデフォルト メトリック グループの表示

SOA extension for WebSphere Process Server および WESB には、デフォルトのダッシュボードとアラートを定義するために使用されるデフォルト メトリック グループが含まれています。これらのデフォルト メトリック グループをカスタム ダッシュボードやカスタム アラートで使用することもできます。

デフォルト メトリック グループは、SOA Extension for WebSphere Process Server 管理モジュール（*WPS_Management_Module.jar*）または SOA Extension for WebSphere Enterprise Service Bus 管理モジュール（*WESB_Management_Module.jar*）の一部として WebSphere 用の Enterprise Manager 拡張にパッケージ化されています。

**WebSphere Process Server または WESB のデフォルト メトリック グループを表示する方法**

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*Super Domain*］-［管理モジュール］を展開し、［WPS_ManagementModule (*Super Domain*)］（すべての WebSphere Process Server コンポーネントを監視する場合）または［WESB_ManagementModule (*Super Domain*)］（スタンドアロンの WESB サーバを監視する場合）を展開します。
3. ［メトリック グループ］ノードを展開すると、WebSphere Process Server に関するすべてのデフォルト メトリック グループ（すべての WebSphere Process Server コンポーネントを監視する場合）、または WebSphere Enterprise Service Bus に関するすべてのデフォルト メトリック グループ（スタンドアロンの WESB サーバを監視する場合）が表示されます。
4. 特定のメトリック グループをクリックすると、［ビューア］ペインにその定義が表示されます。

   任意のメトリック グループのデフォルト設定を変更したり、ユーザ独自のカスタム メトリック グループを作成したりできます。

**注**: メトリック グループを作成および変更する方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

# WPS および WESB のデフォルト アラートの表示

SOA extension for WebSphere Process Server および WESB には、事前設定済みのダッシュボードで使用されるデフォルト アラート定義が含まれています。 これらのデフォルト アラートをカスタム ダッシュボードで使用することもできます。 ほとんどのデフォルト アラートは、デフォルトの警告しきい値と危険しきい値を使用して、しきい値を超えるか重要度が高くなった場合にコンソールに通知を送信するように事前設定されています。

デフォルト アラート定義は、SOA Extension for WebSphere Process Server 管理モジュール（*WPS_Management_Module.jar*）または SOA Extension for WebSphere Enterprise Service Bus 管理モジュール（*WESB_Management_Module.jar*）の一部として WebSphere 用の Enterprise Manager 拡張にパッケージ化されています。

### WebSphere Process Server エージェントまたは WESB エージェントのデフォルト アラート定義を表示する方法

1. Investigator で、［Workstation］-［新規管理モジュール エディタ］の順にクリックします。
2. ［*Super Domain*］-［Management Modules］を展開し、［WPS_ManagementModule (*Super Domain*)］（すべての WebSphere Process Server コンポーネントを監視する場合）または［WESB_ManagementModule (*Super Domain*)］（スタンドアロンの WESB サーバを監視する場合）を展開します。
3. ［Alerts］ノードを展開すると、WebSphere Process Server 管理モジュールに定義されているすべてのアラートが表示されます。
4. 特定のアラートをクリックすると、［ビューア］ペインにその定義が表示されます。

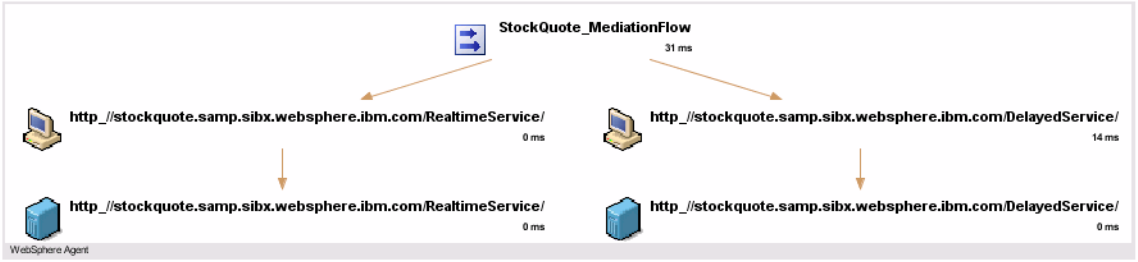
特に、デフォルトの警告しきい値と危険しきい値、およびクリティカル アラートに対する事前定義済みのアクションを確認し、環境に合わせて調整してください。 たとえば、必要に応じてしきい値を調整したり、通知を追加したり、修正処置を定義したりできます。

# WPS または WESB の依存関係の表示

Investigator ツリーのノードを使用して WebSphere Process Server の［BusinessProcesses］、［BusinessStateMachines］、［MediationFlows］、または WebSphere Enterprise Service Bus の［MediationFlows］を選択し、［SOA 依存マップ］タブをクリックすると、ビジネス プロセス、ビジネス ステート マシン、メディエーション フロー、およびアダプタのアウトバウンド コンポーネントの依存関係が表示されます。 EIS のオペレーションを呼び出すコンポーネントを選択すると、依存マップにアダプタのアウトバウンドの依存関係も表示されます。たとえば、メディエーション フローがアダプタを使用して JDBC テーブルから行を取得する場合は、メディエーション フローを選択し、その依存関係を表示することにより、依存マップにそのメディエーション フローとアウトバウンド アダプタ間の依存関係を表示できます。

選択するノードによって、依存マップに表示されるコンテキストが決まります。さらに、表示される詳細情報のコンテキストとレベルをロールアップして折りたたんだり、ロールダウンして展開したりできます。 たとえば、あるビジネス プロセスの依存関係を高レベルで表示するには、Investigator でそのビジネス プロセス名を選択し、［SOA 依存マップ］タブをクリックします。

以下の例は、Investigator ツリーで選択された株価情報ビジネス サービスのメディエーション フローと、詳細な依存関係を表示するために展開された依存マップを示しています。




必要に応じて、ビジネス プロセスのワークフロー全体が表示されるまでマップに依存関係のレベルを追加したり、マップの特定のノードを拡大表示したりできます。依存マップの操作方法の詳細については、「SOA 依存マップの使用 (P. 81)」を参照してください。

# WPS または WESB のトランザクションの追跡

トランザクションの追跡では、ビジネス トランザクションの完了に関係する具体的な手順の詳細ビューまたはサマリ ビューが提供されます。WebSphere Process Server または WebSphere Enterprise Service Bus のコンポーネントまたは Web サービスを含むトランザクションについては、以下のプロトコルでルーティングされるオペレーションを含むトランザクションを追跡できます。

- SOAP（Simple Object Access Protocol）
- HTTP（HyperText Transport Protocol）
- HTTPS（Hypertext Transport Protocol Secure）
- JMS（Java Message Service）

WebSphere Process Server または WebSphere Enterprise Service Bus コンポーネントの外部で開始されたトランザクションに、WebSphere Process Server または WebSphere Enterprise Service Bus 環境で開始された非同期呼び出しが含まれる可能性があります。 トランザクションに参加するすべてのコンポーネントにまたがってこれらのトランザクションを追跡できるようにするため、トランザクションが個々のコンポーネントやオペレーションを順番に実行するときに、相関識別子が挿入され、使用されます。 その結果、トランザクションの詳細にドリルダウンして、トランザクションに関係する WebSphere Process Server または WebSphere Enterprise Service Bus のコンポーネントを把握できます。これには、実行されたオペレーションや各オペレーションの完了までにかかった時間が含まれます。

また、CA APM for SOA と CA APM for WebSphere Process Server が追跡対象のすべてのノードで有効になっている限り、あらゆるプラットフォームにまたがって BusinessTransaction を追跡できます。 このため、トランザクションが複数の JVM や CLR にまたがっている場合でも、トランザクションに関する詳細情報を表示できます。

## プロセスにまたがるトランザクション追跡の値について

プロセスにまたがるトランザクション追跡は、サービス指向アーキテクチャ内の疎結合サービスによって実行されているオペレーションに関して貴重な情報を提供します。 プロセスにまたがるトランザクション追跡を使用して、以下のことを特定できます。

- WebSphere Enterprise Service Bus コンポーネントによってメッセージがどのようにルーティングされるか。
- トランザクション中にどのコンポーネントとアクティビティが呼び出され、実行されるか。
- トランザクション中に呼び出され、実行される呼び出しのシーケンス。
- 要求または返答の処理が最も遅い箇所。

## サンプル トランザクション追跡の開始と表示

トランザクション追跡セッションは、以下のいずれかの方法で開始できます。

- SOA 依存マップ内のマップ ノードから直接開始する方法。
- ［Workstation］-［新規トランザクション追跡セッション］をクリックして Workstation から手動で開始する方法。

依存マップからトランザクション追跡を開始する場合は、マップ ノードのタイプに応じてデフォルト フィルタが自動的に設定されます。 手動で新しいトランザクション追跡セッションを開始する場合は、WebSphere Process Server 用の以下のフィルタ タイプから 1 つを選択できます。

- アダプタ ノード
- ビジネス プロセス
- ビジネス ステート マシン
- メディエーション フロー
- メディエーション フロー オペレーション

たとえば、特定のメディエーション フローのトランザクションをフィルタするには、mediationflow フィルタ選択し、メディエーション フロー名の全部または一部を入力します。

フィルタを設定してからトランザクション追跡セッションを開始すると、トランザクション追跡ビューアが表示されます。 追跡を選択すると、WebSphere Process Server を使用して実行されたビジネス プロセス、ステータス変更、Java コンポーネントなど、トランザクション中に行われた呼び出しに関する詳細情報が表示されます。

## WebSphere プロセスのシーケンス ビューの使用

WebSphere Process Server が関係するトランザクションには非同期の呼び出しが含まれることが多いため、[シーケンス ビュー] をクリックして、トランザクションの一部として非同期で実行されるプロセスのトランザクション ワークフローを表示すると便利です。 シーケンス ビューには、シーケンスを識別できる範囲でプロセスの実行順序が表示されます。WebSphere Process Server のトランザクションでは、このシーケンスは必ずしも従来の呼び出し元と呼び出し先の関係を表すものではなく、1 つのプロセスが別のプロセスの実行のトリガになることを示します。-

ただし、WebSphere Process Server および WESB のプロセスの処理時間は、呼び出されたプロセスに関連する処理時間を含む、プロセスの開始から完了までの全体の継続時間を使用して計算されることに注意してください。WebSphere Process Server および WESB のプロセスについては、正味の継続時間 (呼び出し元プロセスの継続時間からブロック化されていない同期および非同期プロセスの処理時間を引いたもの) はサポートされていません。

トランザクション追跡の詳細については、「SOA 環境でのトランザクション追跡の使用 (P. 111)」を参照してください。

**注**: 追跡を構成する方法の詳細については、「CA APM Java Agent 実装ガイド」または「CA APM .NET Agent 実装ガイド」を参照してください。 トランザクション追跡ビューと履歴データの使用方法の詳細については、「CA APM Workstation ユーザ ガイド」を参照してください。

## 追跡にビジネス プロセス/ビジネス ステート マシン内のアクティビティを含めるための設定

デフォルトでは、WebSphere Process Server ビジネス プロセス/ビジネス ステート マシン用のトランザクション追跡には、アクティビティ レベルのパフォーマンス情報が含まれます。 これらの追跡には、アクティビティ自体の実行に関するそのほかの詳細情報は含まれません。 アクティビティ名の取得には大きなパフォーマンス上のオーバーヘッドが伴うため、詳細なアクティビティ レベルの追跡はデフォルトではオフになっています。

ビジネス プロセス/ビジネス ステート マシンで実行されるアクティビティに関する詳細情報を参照するために、アクティビティ追跡を手動でオンにできます。 ただし、限定された時間でのみアクティビティ追跡をオンにしてください。

**次の手順に従ってください**:

1. WebSphere Process Server または WESB アプリケーション サーバを停止します。
2. *<Agent_Home>* ディレクトリに移動し、wps.pbd ファイルをテキスト エディタで開きます。
3. ActivityTracing と ActivityResponseTracing のフラグのコメント化を解除します。

   ```
   TurnOn: ActivityTracing
   TurnOn: ActivityResponseTracing
   ```

   追加のアクティビティ レベルの追跡が有効になります。

   **注**: 時間属性を取得するためのパラメータを有効にしている (P. 370)場合は、以下のように ActivityTracer のコメント化を解除してください。 これにより、Activities ノードが、別のビジネス プロセス/ビジネス ステート マシン ノードではなく、同じビジネス プロセス/ビジネス ステート マシン ノードの下に表示されます。

   ```
   #SetTracerParameter: ActivityTracer nameformatter
   com.wily.powerpack.websphereprocserver.nameformatter.ActivityTimeNameformatte
   r
   ```

4. wps.pbd を保存して閉じます。
5. WebSphere Process Server または WESB サーバを起動します。

   エージェントが、新しい構成で再起動します。

# 付録 A: SOA 専用のエージェント構成プロパティ

このセクションでは、特に CA APM for SOA を構成するために用意されたエージェント プロパティについて説明します。

このセクションには、以下のトピックが含まれています。



## エージェント プロパティの格納場所について

CA APM for SOA エージェントのプロパティは、エージェント プロファイル（*IntroscopeAgent.profile*）に格納されます。 このプロファイルは、プロパティ名と値で構成されるテキスト ファイルです。 エージェントは、エージェント プロファイルを以下のように検索します。

- *com.wily.introscope.agentProfile* システム プロパティで指定された場所。
- *com.wily.introscope.agentProfile* プロパティが設定されていない場合、エージェントは *IntroscopeAgent.profile* ファイルを使用します。

ほとんどの場合、エージェント プロパティの変更を有効にするには、エージェントが監視するアプリケーション サーバを再起動する必要があります。

## エージェント プロパティの構成

エージェント プロパティは、エージェントの動作やオペレーションの制御や、ユーザの環境に合わせた設定のカスタマイズなどを行うために使用します。 CA APM for SOA には、デフォルトが設定された、SOA 専用のエージェント プロパティがいくつかあります。 これらのプロパティを使用する前に、プロパティ名を *IntroscopeAgent.profile* ファイルに追加し、プロパティに有効な値を設定する必要があります。

**次の手順に従ってください**:

1. IntroscopeAgent.profile ファイルをテキスト エディタで開きます。
2. SOA 専用のエージェント プロパティ (P. 400) の一覧から、*IntroscopeAgent.profile* ファイルに追加するプロパティを特定します。
3. *IntroscopeAgent.profile* ファイルの SOA 専用セクションに、*com.wily.introscope* プレフィックスを含むプロパティの完全な名前を入力します。 例：

```
########################################
# SOA Performance Management Agent Settings
# ====================================
com.wily.introscope.agent.httpheaderread.enabled
```

4. 必要に応じてプロパティ値を設定します。 例：

```
com.wily.introscope.agent.httpheaderread.enabled=true
```

5. *IntroscopeAgent.profile* ファイルを保存して閉じます。

   *IntroscopeAgent.profile* ファイルにプロパティを追加した後は、必要に応じてプロパティ値を変更できます。

# SOA 専用のエージェント プロパティについて

IntroscopeAgent.profile には、以下の SOA 専用プロパティを設定できます。すべてのプロパティは com.wily.introscope というプレフィックスで始まりますが、読みやすくするため、リストではこのプレフィックスを付けずにプロパティを表示します。

**注**: そのほかのエージェント プロパティの詳細については、「CA APM Java Agent 実装ガイド」または「CA APM .NET Agent 実装ガイド」を参照してください。

agent.httpheaderinsertion.enabled (P. 402)

HTTP ヘッダへのクライアント側の相関関係情報の挿入を有効または無効にします。

agent.httpheaderread.enabled (P. 403)

HTTP ヘッダからのサーバ側の相関関係情報の取得を有効または無効にします。

agent.soapheaderinsertion.enabled (P. 404)

SOAP ヘッダへのクライアント側の相関関係情報の挿入を有効または無効にします。

agent.soapheaderread.enabled (P. 405)

SOAP ヘッダからのサーバ側の相関関係情報の取得を有効または無効にします。

agent.soa.metricNameFormatting (P. 405)

指定された文字（1 文字または複数文字）をアンダースコア（_）に置換して、CA APM for SOA のメトリック名を変更します。

agent.transactiontrace.boundaryTracing.cacheFlushFrequency (P. 407)

エージェントのメモリに保持された依存関係データがキャッシュのフラッシュにより削除されるまでの日数を指定します。

agent.transactiontrace.boundaryTracing.enable (P. 408)

トランザクションの境界トレースを有効または無効にします。

soa.client.prependhandler (P. 409)

クライアント側の SOAP ヘッダの挿入位置を制御します。

soa.server.appendhandler (P. 410)

サーバ側の SOAP ヘッダの取得と削除を制御します。

ほとんどの場合、エージェント プロパティを変更したときは、プロパティの変更を有効にするためにアプリケーション サーバを再起動する必要があります。

## agent.httpheaderinsertion.enabled

HTTP ヘッダへのクライアント側の相関識別子の挿入を有効または無効にします。

このプロパティは、SAP NetWeaver ではサポートされません。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - クライアント側の相関関係と依存関係の情報を HTTP ヘッダに追加します。
- false - クライアント側の相関関係と依存関係の情報に関する HTTP ヘッダの変更を禁止します。

### デフォルト

CA APM for SOA、CA APM for WebSphere Process Server、および CA APM for WebSphere Enterprise Service Bus の場合は、false です。

CA APM for Oracle Service Bus、CA APM for TIBCO BusinessWorks、および CA APM for webMethods Integration Server の場合は、true です。

### 例

```
com.wily.introscope.agent.httpheaderinsertion.enabled=true
```

### 注

このプロパティの変更を有効にするには、マネージド アプリケーションを再起動する必要があります。

## agent.httpheaderread.enabled

HTTP ヘッダからのサーバ側の相関識別子の取得を有効または無効にします。

このプロパティは、SAP NetWeaver ではサポートされません。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - サーバ側の相関関係と依存関係の情報を HTTP ヘッダから読み取ります。
- false - HTTP ヘッダの相関関係と依存関係の情報をチェックしません。

### デフォルト

CA APM for SOA、CA APM for WebSphere Process Server、および CA APM for WebSphere Enterprise Service Bus の場合は、false です。

CA APM for Oracle Service Bus、CA APM for TIBCO BusinessWorks、および CA APM for webMethods Integration Server の場合は、true です。

### 例

```
com.wily.introscope.agent.httpheaderread.enabled=true
```

### 注

このプロパティの変更を有効にするには、マネージド アプリケーションを再起動する必要があります。

## agent.soapheaderinsertion.enabled

SOAP ヘッダへのクライアント側の相関識別子の挿入を有効または無効にします。

SOA 依存マップやプロセスにまたがるトランザクション追跡を使用するには、相関識別子をプロセス間で受け渡す必要があります。 エージェントはこの相関識別子を SOAP ヘッダまたは HTTP ヘッダに挿入できます。

予期しないヘッダ エントリのために SOAP ベースのアプリケーションが正常に機能しない場合は、このプロパティを false に設定してください。このプロパティを false に設定し、HTTP ヘッダへの相関識別子の挿入を有効にしないと、SOA 依存マップやプロセスにまたがるトランザクション追跡で関連する情報を表示できなくなります。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - クライアント側の相関関係と依存関係の情報を SOAP ヘッダに追加します。
- false - クライアント側の相関関係と依存関係の情報に関する SOAP ヘッダの変更を禁止します。

### デフォルト

true

### 例

```
com.wily.introscope.agent.soapheaderinsertion.enabled=true
```

### 注

このプロパティの変更を有効にするには、マネージド アプリケーションを再起動する必要があります。

## agent.soapheaderread.enabled

SOAP ヘッダからのサーバ側の相関識別子の取得を有効または無効にします。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - サーバ側の相関関係と依存関係の情報を SOAP ヘッダから読み取ります。
- false - SOAP ヘッダの相関関係と依存関係の情報をチェックしません。

### デフォルト

true

### 例

```
com.wily.introscope.agent.soapheaderread.enabled=true
```

### 注

このプロパティの変更を有効にするには、マネージド アプリケーションを再起動する必要があります。

## agent.soa.metricNameFormatting

CA APM for SOA のメトリック名の表示されている文字（1 文字または複数文字）を、アンダースコア［_］に置き換えます。

たとえば、スラッシュ（/）を置換するように com.wily.introscope.agent.soa.metricNameFormatting プロパティ設定すると、http://CheckingAccount/demobank.ca.com という URL は以下のように表示されます。

```
http_CheckingAccount_demobank.ca.com
```

このプロパティは、メトリック名に含まれる指定文字をアンダースコア（_）に置換する場合に設定します。

この置換を行うことにより、メトリック名を一部の Web Services Manager で使用される形式で表示できます。 この場合、webservices.pbd ファイル内の namespace を servicename に置換する必要があります。 これらの変更によって、CA APM for SOA のメトリック名がネームスペースではなくサービス名を使用して Investigator ツリーと SOA 依存マップに表示されるようになります。

### プロパティ設定

このプロパティは、アンダースコア（_）に置換するメトリック名内の 1 つ以上の文字に設定できます。

プロパティが定義されていない場合や、プロパティに値がない場合は、文字の置換は行われません。

### デフォルト

なし

### 例

```
com.wily.introscope.agent.soa.metricNameFormatting=/:
```

### 注

このプロパティの変更を有効にするには、マネージド アプリケーションを再起動する必要があります。

## agent.transactiontrace.boundaryTracing.cacheFlushFrequency

エージェントのメモリに保持された依存関係データがキャッシュのフラッシュにより削除されるまでの日数を指定します。 このプロパティを設定すると、定期的にアイドル スレッドがフラッシュされ、エージェントのキャッシュがクリアされます。 指定した期間の終了時にキャッシュがフラッシュされる結果、SOA 依存マップの依存関係データがエージェント内で削除されます。 その後、エージェントは依存関係データを再検出し、そのデータを Enterprise Manager に転送することにより、SOA 依存マップに表示された情報をリフレッシュします。

デフォルトでは、既知の依存関係に関する情報がエージェントから Enterprise Manager に不必要に送信されるのを防ぐため、以前に検出された依存関係はエージェント キャッシュに 30 日間保持されます。 ほとんどの場合、この期間は 30 日で十分です。 これは、依存関係データがほとんど変化せず、以前に検出された依存関係をエージェント キャッシュに格納しておけば、エージェントが一定の間隔で Enterprise Manager に転送する必要があるデータを削減できるためです。

実行中のエージェントが、スタンドアロン Enterprise Manager との接続を失った場合や、非クラスタ（MOM がない）環境下で新しい Enterprise Manager にフェールオーバした場合は、このプロパティを 1 に設定することにより、エージェント キャッシュを 1 日でフラッシュできます。 このプロパティを 1 に設定すると、キャッシュから既知の依存関係が削除されるため、エージェントは新しい Enterprise Manager にデータを送信できるように、削除された依存関係を強制的に再検出します。

### プロパティ設定

このプロパティは、0 より大きい任意の自然整数に設定できます。

### デフォルト

30 日

### 例

```
com.wily.introscope.agent.transactiontrace.boundaryTracing.cacheFlushFrequency=1
```

### 注

このプロパティへの変更はただちに有効になります。管理対象アプリケーション サーバを再起動する必要はありません。

## agent.transactiontrace.boundaryTracing.enable

トランザクション追跡境界トレースを有効または無効にします。SOA 依存マップに情報を表示する場合は、境界トレースを有効にする必要があります。SOA 依存マップを使用していない場合は、このプロパティを無効にしてオーバーヘッドを軽減できます。

オーバーヘッドの問題を回避するため、境界トレースは特定のサイズになったときにエージェント キャッシュから送信されます。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - SOA 依存マップに必要なトランザクション追跡境界トレースの情報を収集し、それを Enterprise Manager に送信します。
- false - SOA 依存マップに必要なトランザクション追跡境界トレースの情報を収集しません。

### デフォルト

true

### 例

```
com.wily.introscope.agent.transactiontrace.boundaryTracing.enable=true
```

### 注

このプロパティの変更を有効にするには、マネージド アプリケーションを再起動する必要があります。

## soa.client.prependhandler

クライアント側の CA APM for SOA SOAP ヘッダの挿入位置を制御します。デフォルトでは、SOAP ハンドラ チェーン内の最初のハンドラは CA APM for SOA SOAP ヘッダを挿入します。 このプロパティを使用して、アプリケーションが SOAP メッセージを処理する方法に応じて SOAP ハンドラ チェーンの最初または最後に SOAP ヘッダを挿入できます。

このプロパティは、SOAP ハンドラを使用し、使用中の SOAP エンジンおよび API に依存するアプリケーション サーバにのみ適用されます。 たとえば、アプリケーション サーバが WebLogic や WebSphere の古いバージョン、または SAP NetWeaver を実行する場合に、このプロパティを使用できます。Apache Axis、Apache CXF、ネイティブ JBoss、WebLogic 10.x（JAX-WS）、WebSphere 7.0（JAX-WS）、.NET、または Spring Web Service を使用している場合は、このプロパティは適用されません。

### プロパティ設定

このプロパティは、true または false に設定できます。

- True - SOAP ハンドラ チェーンの最初のハンドラを使用して SOAP ヘッダを挿入します。
- False - SOAP ハンドラ チェーンの最後のハンドラを使用して SOAP ヘッダを追加します。

### デフォルト

true

### 例

```
com.wily.introscope.soa.client.prependhandler=true
```

### 注

このプロパティの変更を有効にするには、マネージド アプリケーションを再起動する必要があります。

## soa.server.appendhandler

サーバ側の CA APM for SOA SOAP ヘッダの取得と削除を制御します。 デフォルトでは、SOAP ハンドラ チェーン内の最後のハンドラは CA APM for SOA SOAP ヘッダを読み取ります。このプロパティを使用して、アプリケーションが SOAP メッセージを処理する方法に応じて SOAP ハンドラ チェーンの最初または最後に SOAP ヘッダを読み取って削除できます。

このプロパティは、SOAP ハンドラを使用し、使用中の SOAP エンジンおよび API に依存するアプリケーション サーバにのみ適用されます。 たとえば、アプリケーション サーバが WebLogic や WebSphere の古いバージョン、または SAP NetWeaver を実行する場合に、このプロパティを使用できます。Apache Axis、Apache CXF、ネイティブ JBoss、WebLogic 10.x（JAX-WS）、WebSphere 7.0（JAX-WS）、.NET、または Spring Web Service を使用している場合は、このプロパティは適用されません。

### プロパティ設定

このプロパティは、true または false に設定できます。

- True - SOAP ハンドラ チェーンの最後のハンドラを使用して CA APM for SOA SOAP ヘッダを読み取り、削除します。
- False - SOAP ハンドラ チェーンの最初のハンドラを使用して CA APM for SOA SOAP ヘッダを読み取り、削除します。

### デフォルト

true

### 例

```
com.wily.introscope.soa.server.appendhandler=false
```

### 注

このプロパティの変更を有効にするには、マネージド アプリケーションを再起動する必要があります。

# 付録 B: SOA 専用の Enterprise Manager 構成プロパティ

このセクションでは、特に CA APM for SOA を構成するために用意された Enterprise Manager プロパティについて説明します。

このセクションには、以下のトピックが含まれています。



## Enterprise Manager プロパティの構成

Enterprise Manager 構成プロパティを使用すると、Enterprise Manager の動作やオペレーションを制御したり、ユーザの環境に合わせて設定をカスタマイズしたりできます。 CA APM for SOA には、デフォルトが設定された、SOA 専用の多くの Enterprise Manager プロパティがあります。 ただし、これらのプロパティを使用するには、プロパティ名を *IntroscopeEnterpriseManager.properties* ファイルに追加し、その値を更新する必要があります。

**IntroscopeEnterpriseManager.properties に構成プロパティを追加する方法**

1. *<EM_Home>/IntroscopeEnterpriseManager.properties* ファイルをテキスト エディタで開きます。
2. 「SOA 専用の Enterprise Manager プロパティについて」のプロパティ リストを使用して、*IntroscopeEnterpriseManager.properties* ファイルに追加するプロパティを特定します。

3. *IntroscopeEnterpriseManager.properties* ファイルの SOA 専用セクションに、*com.wily.introscope* プレフィックスを含むプロパティの完全な名前を入力します。 例：

```
########################################
# SOA Performance Management EM Settings
# ============================
com.wily.introscope.soa.dashboard.typeviewer.mostcritical.count
```

4. 必要に応じてプロパティ値を設定します。 たとえば、このプロパティのデフォルト設定を使用するには、以下のようにします。

```
com.wily.introscope.soa.dashboard.typeviewer.mostcritical.count=10
```

5. *IntroscopeEnterpriseManager.properties* ファイルを保存して閉じます。

*IntroscopeEnterpriseManager.properties* ファイルに Enterprise Manager 拡張プロパティを追加した後は、必要に応じてプロパティ値を変更できます。

### Enterprise Manager 構成プロパティの値を変更する方法

1. *<EM_Home>/IntroscopeEnterpriseManager.properties* ファイルをテキスト エディタで開きます。
2. 変更するプロパティを見つけ、ユーザ環境に合わせて新しい値を設定します。 例：

```
com.wily.introscope.soa.dashboard.typeviewer.mostcritical.count=15
```

   上記は単に例として挙げたもので、CA APM for SOA の推奨設定ではありません。

3. *IntroscopeEnterpriseManager.properties* ファイルを保存して閉じます。
4. Enterprise Manager を再起動します。

# SOA 専用の Enterprise Manager プロパティについて

*IntroscopeEnterpriseManager.properties* ファイルには、以下の SOA 専用プロパティを設定できます。 すべてのプロパティは *com.wily.introscope* というプレフィックスで始まりますが、読みやすくするため、リストではこのプレフィックスを付けずにプロパティを表示します。

**注**: そのほかの Enterprise Manager プロパティの設定方法については、「CA APM 設定および管理ガイド」を参照してください。

soa.dependencymap.aging.refresh.interval (P. 416)

以前に検出された依存関係の経過期間をチェックする間隔を指定します。

soa.dependencymap.aging.expire.interval (P. 417)

SOA 依存マップに保持されている依存関係が期限切れになるまでの最大日数を指定します。

soa.dependencymap.heuristics.clientside.enable (P. 418)

クライアント側の論理等価物ヒューリスティック ルールを有効または無効にします。

soa.dependencymap.heuristics.namematch.enable (P. 419)

論理等価物ヒューリスティック名前照合ルールを有効または無効にします。

soa.dependencymap.heuristics.serverside.enable (P. 420)

サーバ側の論理等価物ヒューリスティック ルールを有効または無効にします。

soa.dependencymap.log.suppress (P. 422)

ログ ファイルに同じエラーまたは警告メッセージを書き込むことができる最大回数を指定します。

soa.dependencymap.max.edge.ratio (P. 423)

マップ内のノードに対する依存関係の最大比率を指定します。

soa.dependencymap.max.vertices (P. 425)

スタンドアロンまたはコレクタ Enterprise Manager 上の依存マップのために格納できるノードの最大数を指定します。

soa.dashboard.typeviewer.errors.enable (P. 436)

選択したオペレーションの［間隔偏差ごとのエラー］グラフの表示を有効または無効にします。

soa.dashboard.typeviewer.mostcritical.enable (P. 439)

［最もクリティカルなオペレーション］ダッシュボードと［クリティカル（上位）］タブの表示を有効または無効にします。

soa.dashboard.typeviewer.mostcritical.count (P. 440)

レポートするクリティカルなオペレーションの最大数を指定します。

soa.dashboard.typeviewer.mostdependent.enable (P. 441)

［最も依存度が高いオペレーション］ダッシュボードと［最も依存度が高い］タブの表示を有効または無効にします。

soa.dashboard.typeviewer.mostdependent.count (P. 442)

レポートする依存オペレーションの最大数を指定します。

ほとんどの場合、プロパティの変更を有効にするには Enterprise Manager を再起動する必要があります。

## soa.dependencymap.aging.refresh.interval

Enterprise Manager が次に SOA 依存マップの依存関係の経過期間チェックを実行するまでの時間を指定します。

Enterprise Manager は検出された各依存関係の経過期間を追跡します。そして定期的に依存関係を再検出して、依存関係がまだ存在するかどうかを確認します。 依存関係の経過期間は、最新の検出が基準になります。

SOA 依存マップの依存関係の経過期間チェックでは、Enterprise Manager が SOA 依存マップ全体の各依存関係の経過期間を特定します。

SOA 依存マップのリフレッシュ間隔はデフォルトでは 1 時間に定義されています。したがって、SOA 依存マップのリフレッシュ間隔 6 個は 6 時間に相当します。

### プロパティ設定

このプロパティは、0 より大きい任意の自然整数に設定できます。

### デフォルト

6 時間

### 例

```
com.wily.introscope.soa.dependencymap.aging.refresh.interval=6
```

### 注

- 依存関係が削除されるかどうかは、このプロパティと *com.wily.introscope.soa.dependencymap.aging.expire.interval* プロパティの組み合わせによって決まります。
- このプロパティへの変更はただちに有効となり、Enterprise Manager を再起動する必要はありません。

## soa.dependencymap.aging.expire.interval

SOA 依存マップ内の依存関係の最大経過期間（日数）。 依存関係の経過期間チェックでは、Enterprise Manager が指定された値よりも古い依存関係を SOA 依存マップからすべて削除します。

たとえば、*com.wily.introscope.soa.dependencymap.aging.expire.interval* の値が 60 日だった場合は、過去 60 日間にわたって再検出されていない依存関係が期限切れの依存関係になります。- この期間中に再検出されていない依存関係は、依存マップから削除されます。

### プロパティ設定

このプロパティは、0 より大きい任意の自然整数に設定できます。

### デフォルト

60 日

### 例

```
com.wily.introscope.soa.dependencymap.aging.expire.interval=90
```

### 注

- このプロパティは、*com.wily.introscope.soa.dependencymap.aging.refresh.interval* プロパティと連動して機能します。
- このプロパティによって依存関係を削除する条件が決定され、SOA 依存マップの依存関係の経過期間をチェックするときにこの条件が適用されます。
- このプロパティへの変更はただちに有効となり、Enterprise Manager を再起動する必要はありません。

## soa.dependencymap.heuristics.clientside.enable

CA APM for SOA のクライアント側の論理等価物ヒューリスティック ルールを有効または無効にします。

クライアント側の論理等価物ヒューリスティック ルールには、以下のことが規定されています。

*クライアント*タイプの 2 つの物理的サービス オペレーションがどちらも*サーバ*タイプの同じ物理的サービス オペレーションに依存している場合、それら 2 つのサービス オペレーションは論理的に等価であるとみなされる。

この設定は、2 つの異なるアプリケーションが同じクライアント側のサービス オペレーションを呼び出した場合を想定しています。 このルールでは、2 つのオペレーションが同じメトリック パス（エージェント指定子を除く）を持っているかどうかに関係なく、この論理等価物が検出されます。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - CA APM for SOA のクライアント側の論理等価物ヒューリスティック ルールを適用します。
- False - CA APM for SOA のクライアント側の論理等価物ヒューリスティック ルールを無効にします。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dependencymap.heuristics.clientside.enable=true
```

### 注

このルールを変更しても、その変更が以前に検出された依存関係に適用されることはありません。 そのため、Enterprise Manager を停止し、以前に保存された 2 種類の SOA 依存マップ ファイルを削除しない限り、ルール値の変更が予測不可能な結果をもたらす可能性があります。

このプロパティへの変更はただちに有効となり、Enterprise Manager を再起動する必要はありません。

**詳細**:

保存された依存関係データを無視または削除する (P. 86)

## soa.dependencymap.heuristics.namematch.enable

CA APM for SOA の論理等価物ヒューリスティック名前照合ルールを有効または無効にします。

論理等価物ヒューリスティック名前照合ルールは、2 つのオペレーションが同じメトリック パス（エージェント指定子を除く）を持っている場合に論理等価物を検出します。

CA APM for SOA の論理等価物ヒューリスティック名前照合ルールでは、エージェント指定子を削除するとメトリック パスが同じになる 2 つの物理的 Web サービス オペレーションが論理的に等価であるとみなされます。

### プロパティ設定

このプロパティは、true または false に設定できます。

- True - 論理等価物ヒューリスティック名前照合ルールを有効にします。
- False - 論理等価物ヒューリスティック名前照合ルールを無効にします。

### デフォルト

false

### 例

```
com.wily.introscope.soa.dependencymap.heuristics.namematch.enable=true
```

**注**

このルールを変更しても、その変更が以前に検出された依存関係に適用されることはありません。 そのため、Enterprise Manager を停止し、以前に保存された 2 種類の SOA 依存マップ ファイルを削除しない限り、ルール値の変更が予測不可能な結果をもたらす可能性があります。

CA APM for SOA に仮想エージェントを作成するときは、このプロパティを有効にします。

このプロパティへの変更はただちに有効となり、Enterprise Manager を再起動する必要はありません。

**詳細**:

保存された依存関係データを無視または削除する (P. 86)

## soa.dependencymap.heuristics.serverside.enable

CA APM for SOA のサーバ側の論理等価物ヒューリスティック ルールを有効または無効にします。

サーバ側の論理等価物ヒューリスティック ルールでは、サーバ タイプの 2 つの物理的サービス オペレーションに依存しているクライアント タイプの何らかの物理的サービス オペレーションが存在している場合にのみ、それら 2 つのサービス オペレーションが論理的に等価であるとみなされます。

これは通常、クライアント側の Web サービス オペレーションの呼び出しが、サーバ側の利用可能な Web サービス オペレーション実装のいずれかにディスパッチされるなどの、何らかの負荷分散メカニズムが SOA に組み込まれている場合にのみ起こります。-

**プロパティ設定**

このプロパティは、true または false に設定できます。

- True - CA APM for SOA のサーバ側の論理等価物ヒューリスティック ルールを有効にします。
- false - CA APM for SOA のサーバ側の論理等価物ヒューリスティック ルールを無効にします。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dependencymap.heuristics.serverside.enable=true
```

### 注

このルールを変更しても、その変更が以前に検出された依存関係に適用されることはありません。 そのため、Enterprise Manager を停止し、以前に保存された 2 種類の SOA 依存マップ ファイルを削除しない限り、ルール値の変更が予測不可能な結果をもたらす可能性があります。このプロパティへの変更はただちに有効となり、Enterprise Manager を再起動する必要はありません。

**詳細**:

保存された依存関係データを無視または削除する (P. 86)

## soa.dependencymap.log.suppress

重複メッセージが抑制される前に、ログ ファイルに同じエラーまたは警告メッセージを書き込むことができる最大回数を指定します。

このプロパティを使用して、繰り返し発生するエラーまたは警告メッセージがその都度 Enterprise Manager のログ ファイルに送信されるのを防ぐことができます。 たとえば、許可された最大数を超えるノードや依存関係を含む依存マップがある場合は、最大数に達するたびにログ ファイルに警告メッセージが記録されます。このプロパティを設定すると、エラーまたは警告が指定された回数だけログ ファイルにポストされた後、メッセージが現在抑制されていることを示すテキスト文字列が最後のメッセージに追加され、Enterprise Manager を再起動するか、またはこのプロパティを変更するまで、同じエラーまたは警告がログ ファイルに記録されなくなります。

### プロパティ設定

このプロパティは、0 より大きい任意の自然整数に設定できます。

### デフォルト

5 個のログ エントリ

### 例

```
com.wily.introscope.soa.dependencymap.log.suppress=5
```

### 注

このプロパティのために Enterprise Manager を再起動する必要はありません。

ただし、プロパティ値を変更すると、エラーおよび警告メッセージを抑制するためのカウントが最初から開始されます。 たとえば、デフォルトでは特定のエラー メッセージが 5 回発生すると、そのエラー メッセージは抑制されます。 プロパティ値を 6 に変更すると、そのエラー メッセージは抑制されなくなります。そのメッセージが抑制されるまでには、エラーがさらに 6 回発生する必要があります。

## soa.dependencymap.max.edge.ratio

マップ内のノードに対する依存関係の最大比率を指定します。 このプロパティを使用して、大規模または複雑な SOA 環境の Enterprise Manager 上に格納された依存マップの全体的な複雑さを制御できます。

通常、Enterprise Manager 上に保存された依存マップ データは、検出されたアプリケーション間のすべての依存関係を表しており、展開されているサービス指向アーキテクチャの完全なモデルを提供します。 しかし、非常に大規模または複雑な SOA 環境では、すべての SOA コンポーネントとそれらの依存関係を完全に表現すると、Enterprise Manager 自体のパフォーマンスやオペレーションに影響する場合があります。 このプロパティを使用してノードと依存関係の比率を指定することにより、許容される複雑さのレベルを制御できます。 この制限に達すると、それ以上の依存関係が保存されず、Enterprise Manager のログ ファイルに警告が書き込まれます。

また、このプロパティと com.wily.introscope.soa.dependencymap.max.vertices プロパティを組み合わせて使用することにより、依存マップに格納されるノードの総数を制限できます。

クラスタ環境では、このプロパティはコレクタ Enterprise Manager にのみ適用されます。 MOM を使用すると、複数のコレクタにまたがって結合された SOA 環境の完全な表現を格納できます。

### プロパティ設定

このプロパティは、0 より大きい任意の自然整数に設定できます。

### デフォルト

1 ノードあたり 5 個の依存関係

### 例

```
com.wily.introscope.soa.dependencymap.max.edge.ratio=5
```

### 注

com.wily.introscope.soa.dependencymap.max.vertices プロパティと com.wily.introscope.soa.dependencymap.max.edge.ratio プロパティのデフォルト値では、依存マップが 5000 個のノードと最大 25000 個の依存関係（5 x 5000）までに制限されます。

ほとんどの SOA ネットワークに含まれるコンポーネントの数は 5000 個未満であり、依存関係の比率は 1 コンポーネントあたり 1 ～ 2 個です。 したがって、デフォルト設定はほとんどの組織で必要とされる以上の複雑さに対応します。 これらのプロパティを使用して依存マップのサイズと複雑さを意図的に制限できますが、それによって SOA 依存マップのモデルが必要以上に不完全になる可能性があります。

## soa.dependencymap.max.vertices

スタンドアロンまたはコレクタ Enterprise Manager 上の依存マップのために格納できるノードの最大数を指定します。 このプロパティを使用して、大規模または複雑な SOA 環境の Enterprise Manager 上に格納された依存マップの最大サイズを制御できます。

通常、Enterprise Manager 上に保存された依存マップ データは、検出されたアプリケーション間のすべての依存関係を表しており、展開されているサービス指向アーキテクチャの完全なモデルを提供します。 しかし、非常に大規模または複雑な SOA 環境では、すべての SOA コンポーネントとそれらの依存関係を完全に表現すると、Enterprise Manager 自体のパフォーマンスやオペレーションに影響する場合があります。 このプロパティを使用して含めるノードの最大数を指定することにより、SOA モデルのサイズを制限できます。 この制限に達すると、それ以上の情報が保存されず、Enterprise Manager のログ ファイルに警告が書き込まれます。

また、このプロパティと com.wily.introscope.soa.dependencymap.max.edge.ratio プロパティを組み合わせて使用することにより、依存マップに格納される依存関係の総数を制限できます。

クラスタ環境では、このプロパティはコレクタ Enterprise Manager にのみ適用されます。 MOM を使用すると、複数のコレクタにまたがって結合された SOA 環境の完全な表現を格納できます。

### プロパティ設定

このプロパティは、0 より大きい任意の自然整数に設定できます。

### デフォルト

5000 個のノード

### 例

```
com.wily.introscope.soa.dependencymap.max.vertices=5000
```

## soa.deviation.enable

以下の Deviation メトリックを生成するのに必要な計算を有効または無効にします。

- Average Response Time Deviation
- Errors Per Interval Deviation
- Responses Per Interval Deviation

このプロパティを true に設定すると、Deviation メトリックが収集され、レポートできるようになります。 さらに、個々の Deviation メトリックを選択的に有効または無効にすることができます。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - Deviation メトリック データを提供するための計算を実行します。
- false - Deviation メトリックをレポートしません。

### デフォルト

true

### 例

```
com.wily.introscope.soa.deviation.enable=true
```

### 注

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.deviation.art.enable

Average Response Time （平均応答時間）偏差メトリックを有効または無効にします。 このプロパティは、Deviation メトリックを計算する場合に Average Response Time （平均応答時間）偏差メトリックをレポートするかどうかを決定します。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - 計算を実行し、Average Response Time （平均応答時間）偏差メトリック データをレポートします。
- false - Average Response Time（平均応答時間）偏差メトリックをレポートしません。

### デフォルト

true

### 例

```
com.wily.introscope.soa.deviation.art.enable=true
```

### 注

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.deviation.dependencymetric.enable

このプロパティは、以下の依存関係メトリックを生成するのに必要な計算を有効または無効にします。

- Critical Direct
- 間接依存クリティカル オペレーション
- 直接依存項目
- 間接依存項目

このプロパティは、CA APM for SOA の依存関係メトリックをレポートするかどうかを決定します。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - 依存関係メトリック データを提供するために計算を実行します。
- false - 依存関係メトリックの計算を実行しません。

### デフォルト

true

### 例

```
com.wily.introscope.soa.deviation.dependencymetric.enable=true
```

### 注

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.deviation.count.per.metric

このプロパティは、偏差メトリックが計算される［WebServices］ノード下のオペレーションの最大数を指定します。 レポートされるオペレーションの数がこの数を超える場合、すべてのサーバおよびクライアント ノード ネームスペース下の最もクリティカルなオペレーションが使用されます。

### プロパティ設定

このプロパティは、0 より大きい任意の自然整数に設定できます。

### デフォルト

依存関係メトリックあたり 25 回のオペレーション

### 例

```
com.wily.introscope.soa.deviation.count.per.metric=25
```

### 注

このプロパティを有効にするには、Enterprise Manager を再起動します。

## soa.deviation.dependency.refreshrate

キャッシュされた SOA 依存マップの依存関係データのリフレッシュ間隔（時間単位）を指定します。

SOA 依存マップ データは、Deviation メトリック サービスによってキャッシュされます。Critical メトリック、Dependent メトリック、および Deviation メトリックは、これらのサービスおよびオペレーションごとに 15 秒おきにレポートされます。

キャッシュされた SOA 依存マップ データは、このプロパティに基づいて定期的にリフレッシュされます。

このマップはめったに変更されず、たいていはデプロイが変更されたときにのみ変更されます。

### プロパティ設定

このプロパティは、0 より大きい任意の自然整数に設定できます。

### デフォルト

1 時間

### 例

```
com.wily.introscope.soa.deviation.dependency.refreshrate=1
```

### 注

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.deviation.errors.enable

エラー偏差のメトリックを有効または無効にします。 このプロパティは、Deviation メトリックを計算する場合にエラー数偏差のメトリックをレポートするかどうかを決定します。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - 計算を実行し、Errors Per Interval （間隔ごとのエラー数）偏差メトリック データをレポートします。
- false - Errors Per Interval （間隔ごとのエラー数）偏差メトリックをレポートしません。

### デフォルト

true

### 例

```
com.wily.introscope.soa.deviation.errors.enable=true
```

### 注

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.deviation.max.metric.count

レポートする偏差メトリックの最大数を指定します。

このプロパティのデフォルト値を変更することは推奨しません。 デフォルト値に従えば、十分なメトリックをレポートし、Enterprise Manager の実行に影響を与えずに有益な情報を提供できます。 デフォルト値を変更すると、Enterprise Manager にパフォーマンスの問題が発生する可能性があります。

**プロパティ設定**

このプロパティは、0 より大きい任意の自然整数に設定できます。

**デフォルト**

合計 1000 個の Deviation メトリック

**例**

```
com.wily.introscope.soa.deviation.metric.count=1000
```

**注**

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.deviation.mean.days

このプロパティは、平均値を計算するためにオペレーションを累積する日数を指定します。 指定された日数が経過すると、平均値が自動的に計算されます。

**プロパティ設定**

このプロパティは、0 より大きい任意の自然整数に設定できます。

**デフォルト**

7 日

**例**

```
com.wily.introscope.soa.deviation.mean.days=3
```

### 注

このプロパティを有効にするには、Enterprise Manager を再起動します。

## soa.deviation.metric.expressionlist

このプロパティは、偏差メトリックの記述子の名前のリストを定義します。偏差メトリックを作成するために使用する名前のカンマ区切りリストに設定してください。 それぞれの名前は、別々のメトリックの記述子として使用されます。

*deviation.metric.calledbackends* プロパティは、ユーザ定義の偏差メトリックを作成するために、このプロパティを参照します。

### プロパティ設定

このプロパティは、ユーザ定義の名前のリストに設定できます。

### デフォルト

calledbackends

### 例

```
deviation.metric.expressionlist=alpha1, beta2, gama3
```

### 追加の偏差の計算

デフォルトでは、偏差メトリックは個々のオペレーションについてのみ計算されます。 Web サービス、クライアント、またはサーバ ネームスペース レベルで使用できる偏差については、以下のステートメントを *IntroscopeEnterpriseManager.properties* ファイルに追加します。

```
com.wily.introscope.soa.deviation.metric.expressionlist=test,test1
com.wily.introscope.soa.deviation.metric.test=WebServices¥¥|Client¥¥|.*
com.wily.introscope.soa.deviation.metric.test1=WebServices¥¥|Server¥¥|.*
```

重要: このように偏差が複数のレベルで計算される場合、余分な処理オーバーヘッドが必要です。

### 注

このプロパティの変更を有効にするには、Enterprise Manager を再起動します。

## soa.deviation.metric.calledbackends

*deviation.metric.expressionlist* プロパティで定義された名前を使用して、Deviation メトリックを作成するための新しいプロパティを作成します。

新しいプロパティ名は以下の形式です。

`com.wily.introscope.soa.deviation.metric.<user-defined_name> = <user-defined regular expression>`

*deviation.metric.calledbackends* プロパティに割り当てられる値は、メトリック記述子です。 例：

*com.wily.introscope.soa.deviation.metric.alpha1=Frontends|Called Backends*
*com.wily.introscope.soa.deviation.metric.beta2=Frontends|Called Backends*

### デフォルト

calledbackends

### 例

*com.wily.introscope.soa.deviation.metric.calledbackends=Frontends|Frontends|FBApp $Frontend|Called Backends|FBApp$Backend*

### 注

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.deviation.usage.enable

Responses Per Interval Deviation メトリックを有効または無効にします。 このプロパティは、Deviation メトリックを計算する場合に応答数偏差のメトリックをレポートするかどうかを決定します。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - 計算を実行し、Responses Per Interval （間隔ごとの応答数）偏差メトリック データをレポートします。
- false - Responses Per Interval （間隔ごとの応答数）偏差メトリックをレポートしません。

### デフォルト

true

### 例

```
com.wily.introscope.soa.deviation.usage.enable=true
```

### 注

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.dashboard.typeviewer.average.enable

選択したオペレーションに関して収集された偏差データを有効にし、以下の場所の［平均応答時間偏差］グラフに表示します。

- ［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボード
- ［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボード
- ［クリティカル（上位）］タブ
- ［最も依存度が高い］タブ

偏差データがダッシュボードやタブに表示されるのは、偏差データがエージェントによって Enterprise Manager にレポートされ、Workstation に表示できる場合のみです。 偏差データのレポートの詳細については、「［偏差］タブで Deviation メトリックを表示する (P. 73)」を参照してください。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - 選択したオペレーションのデータを有効にし、［平均応答時間偏差］グラフに表示します。
- false - 選択したオペレーションのデータを［平均応答時間偏差］グラフに表示しません。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.average.enable=true
```

### 注

このプロパティは *IntroscopeEnterpriseManager.properties* ファイルまたは *IntroscopeWorkstation.properties* ファイルで定義できます。このプロパティが *IntroscopeWorkstation.properties* ファイルと *IntroscopeEnterpriseManager.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.dashboard.typeviewer.errors.enable

選択したオペレーションに関して収集された偏差データを有効にし、以下の場所の［間隔偏差ごとのエラー］グラフに表示します。

- ［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボード
- ［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボード
- ［クリティカル（上位）］タブ
- ［最も依存度が高い］タブ

偏差データがダッシュボードやタブに表示されるのは、偏差データがエージェントによって Enterprise Manager にレポートされ、Workstation に表示できる場合のみです。 偏差データのレポートの詳細については、「［偏差］タブで Deviation メトリックを表示する (P. 73)」を参照してください。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - 選択したオペレーションのデータを有効にし、［Errors Per Interval （間隔ごとのエラー数）］グラフに表示します。
- false - 選択したオペレーションのデータを［Errors Per Interval （間隔ごとのエラー数）］グラフに表示しません。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.response.enable=true
```

### 注

このプロパティは *IntroscopeEnterpriseManager.properties* ファイルまたは *IntroscopeWorkstation.properties* ファイルで定義できます。このプロパティが *IntroscopeWorkstation.properties* ファイルと *IntroscopeEnterpriseManager.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.dashboard.typeviewer.response.enable

選択したオペレーションに関して収集された偏差データを有効にし、以下の場所の［Responses Per Interval （間隔ごとの応答数）偏差］グラフに表示します。

- ［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボード
- ［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボード
- ［クリティカル（上位）］タブ
- ［最も依存度が高い］タブ

偏差データがダッシュボードやタブに表示されるのは、偏差データがエージェントによって Enterprise Manager にレポートされ、Workstation に表示できる場合のみです。 偏差データのレポートの詳細については、「［偏差］タブで Deviation メトリックを表示する (P. 73)」を参照してください。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - 選択したオペレーションのデータを有効にし、［Responses Per Interval （間隔ごとの応答数）偏差］グラフに表示します。
- false - 選択したオペレーションのデータを［間隔ごとの応答数偏差］グラフに表示しません。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.response.enable=true
```

### 注

このプロパティは *IntroscopeEnterpriseManager.properties* ファイルまたは *IntroscopeWorkstation.properties* ファイルで定義できます。このプロパティが *IntroscopeWorkstation.properties* ファイルと *IntroscopeEnterpriseManager.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.dashboard.typeviewer.mostcritical.enable

コンソールの［最もクリティカルなオペレーション］ダッシュボードと Investigator の［クリティカル（上位）］タブの表示を有効にします。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - コンソールのダッシュボード メニューに［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボード オプションを表示し、Investigator に［クリティカル（上位）］タブを表示します。
- false - ［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボードおよび［クリティカル（上位）］タブへのアクセスを無効にします。

このプロパティを false に設定すると、Investigator 内のどのノードについても［クリティカル（上位）］タブが表示されなくなります。 コンソールのダッシュボード ドロップダウン リストには［最もクリティカルなオペレーション］ダッシュボードが表示され続けますが、選択してもダッシュボードは表示されません。 ［最もクリティカルなオペレーション］ダッシュボードの代わりに、このダッシュボードを使用できないことを示すエラー メッセージが表示されます。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.mostcritical.enable=true
```

### 注

このプロパティは IntroscopeEnterpriseManager.properties ファイルまたは IntroscopeWorkstation.properties ファイルで定義できます。 このプロパティが *IntroscopeWorkstation.properties* ファイルと *IntroscopeEnterpriseManager.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager と Workstation を再起動する必要があります。

## soa.dashboard.typeviewer.mostcritical.count

すべてのエージェントによって［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボードと［クリティカル（上位）］タブにレポートされるクリティカルなオペレーションの総数を決定します。

### プロパティ設定

最小値は 0 です。

最大値は *com.wily.introscope.soa.deviation.count.per.metric* プロパティの値（デフォルト値は 25）以下である必要があります。

### デフォルト

10 個のオペレーション

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.mostcritical.count=5
```

### 注

このプロパティは *IntroscopeEnterpriseManager.properties* ファイルまたは *IntroscopeWorkstation.properties* ファイルで定義できます。このプロパティが *IntroscopeEnterpriseManager.properties* ファイルと *IntroscopeWorkstation.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager と Workstation を再起動する必要があります。

## soa.dashboard.typeviewer.mostdependent.enable

コンソールの［最も依存度が高いオペレーション］ダッシュボードと Investigator の［最も依存度が高い］タブの表示を有効にします。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - コンソールのダッシュボード メニューに［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボード オプションを表示し、Investigator に［最も依存度が高い］タブを表示します。
- false - ［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボードおよび［最も依存度が高い］タブへのアクセスを無効にします。

このプロパティを false に設定すると、Investigator に［最も依存度が高い］タブが表示されなくなります。 コンソールのダッシュボード ドロップダウン リストには［最も依存度が高いオペレーション］ダッシュボードが表示され続けますが、選択してもダッシュボードは表示されません。［最も依存度が高いオペレーション］ダッシュボードの代わりに、このダッシュボードを使用できないことを示すエラー メッセージが表示されます。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.mostdependent.enable=true
```

### 注

このプロパティは *IntroscopeEnterpriseManager.properties* ファイルまたは *IntroscopeWorkstation.properties* ファイルで定義できます。このプロパティが *IntroscopeEnterpriseManager.properties* ファイルと *IntroscopeWorkstation.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager と Workstation を再起動する必要があります。

## soa.dashboard.typeviewer.mostdependent.count

すべてのエージェントによって［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボードと［最も依存度が高い］タブにレポートされる最も依存度が高いオペレーションの総数を決定します。

### プロパティ設定

最小値は 0 です。

最大値は com.wily.introscope.soa.deviation.count.per.metric プロパティの値（デフォルト値は 25）以下である必要があります。

### デフォルト

10 個のオペレーション

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.mostdependent.count=5
```

### 注

このプロパティは IntroscopeEnterpriseManager.properties ファイルまたは IntroscopeWorkstation.properties ファイルで定義できます。 このプロパティが IntroscopeEnterpriseManager.properties ファイルと IntroscopeWorkstation.properties ファイルの両方で定義されている場合は、IntroscopeWorkstation.properties のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager と Workstation を再起動する必要があります。

# 付録 C: SOA 専用の Workstation 構成プロパティ

Workstation 構成プロパティを使用すると、Workstation の動作やオペレーションを制御できます。 これらのプロパティを使用して、ユーザの環境に最もよく適合するように Workstation をカスタマイズすることもできます。 Workstation プロパティを使用すると、CA APM for SOA を特別に構成できます。

このセクションには、以下のトピックが含まれています。

# Workstation プロパティの設定

CA APM for SOA には、デフォルトが設定された、SOA 専用の Workstation プロパティがいくつかあります。 ただし、これらのプロパティを設定するには、まずプロパティ名を *IntroscopeWorkstation.properties* ファイルに追加する必要があります。

### Workstation プロパティを IntroscopeWorkstation.properties に追加する方法

1. *<EM_Home>/config* ディレクトリの *IntroscopeWorkstation.properties* ファイルを開きます。
2. 「SOA 専用の Workstation プロパティについて (P. 445)」に示されたプロパティの中から、*IntroscopeWorkstation.properties* ファイルに追加するプロパティを見つけます。
3. *IntroscopeWorkstation.properties* ファイルに、*com.wily.introscope* プレフィックスを含むプロパティの完全な名前を追加します。 例：
   `com.wily.introscope.soa.dashboard.typeviewer.mostcritical.count`
4. 必要に応じてプロパティ値を設定します。 たとえば、このプロパティのデフォルト設定を使用するには、以下のようにします。
   `com.wily.introscope.soa.dashboard.typeviewer.mostcritical.count=10`
5. *IntroscopeEnterpriseManager.properties* ファイルを保存して閉じます。

*IntroscopeWorkstation.properties* ファイルに Workstation プロパティを追加した後は、必要に応じてプロパティ値を設定できます。

### Workstation 構成プロパティの値を変更する方法

1. *<EM_Home>/config* ディレクトリの *IntroscopeWorkstation.properties* ファイルを開きます。
2. 変更するプロパティを見つけ、ユーザ環境に合わせて新しい値を設定します。 例：
   `com.wily.introscope.soa.dashboard.typeviewer.mostcritical.count=15`
   上記の設定は単に例として挙げたもので、CA APM for SOA の推奨設定ではありません。
3. *IntroscopeWorkstation.properties* ファイルを保存して閉じます。
4. プロパティの変更を有効にするため、Workstation を再起動します。

# SOA 専用の Workstation プロパティについて

*IntroscopeWorkstation.properties* ファイルには、以下の SOA 専用プロパティを設定できます。 すべてのプロパティは *com.wily.introscope* で始まりますが、読みやすくするため、リストではこのプレフィックスを付けずにプロパティを表示します。

**注**: そのほかの Workstation プロパティの設定方法については、「CA APM 設定および管理ガイド」を参照してください。

soa.dependencymap.ui.view.nodecount (P. 446)

SOA 依存マップに表示されるマップ ノードの最大数を指定します。

com.wily.introscope.soa.dependencymap.ui.view.edgecount (P. 448)

SOA 依存マップに表示されるマップの辺の最大数を指定します。

soa.dashboard.typeviewer.average.enable (P. 435)

選択したオペレーションの［平均応答時間偏差］グラフの表示を有効または無効にします。

soa.dashboard.typeviewer.response.enable (P. 437)

選択したオペレーションの［間隔ごとの応答数偏差］グラフの表示を有効または無効にします。

soa.dashboard.typeviewer.errors.enable (P. 436)

選択したオペレーションの［間隔偏差ごとのエラー］グラフの表示を有効または無効にします。

soa.dashboard.typeviewer.mostcritical.enable (P. 439)

［最もクリティカルなオペレーション］ダッシュボードと［クリティカル（上位）］タブの表示を有効または無効にします。

soa.dashboard.typeviewer.mostcritical.count (P. 440)

レポートするクリティカルなオペレーションの最大数を指定します。

soa.dashboard.typeviewer.mostdependent.enable (P. 441)

［最も依存度が高いオペレーション］ダッシュボードと［最も依存度が高い］タブの表示を有効または無効にします。

soa.dashboard.typeviewer.mostdependent.count (P. 442)

レポートする依存オペレーションの最大数を指定します。

workstation.soa.dependencymap.fetchmetrics (P. 458)

SOA 依存マップ内のメトリックの表示を制御します。

workstation.traceview.crossprocess.duration.full (P. 459)

トランザクション追跡に関して全体の継続時間を使用するかどうかを決定します。

workstation.traceview.crossprocess.duration.net (P. 461)

トランザクション追跡に関して正味の継続時間を使用するかどうかを決定します。

ほとんどの場合、プロパティの変更を有効にするには Workstation を再起動する必要があります。 プロパティが *IntroscopeWorkstation.properties* ファイルと *IntroscopeEnterpriseManager.properties* ファイルで定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

## soa.dependencymap.ui.view.nodecount

このプロパティは、Workstation の SOA 依存マップに表示されるマップ ノードの最大数を指定します。

Investigator のノードを選択したときに、SOA 依存マップのノード数が com.wily.introscope.soa.dependencymap.ui.view.nodecount の値を超えている場合は、エラー メッセージが表示されます。 SOA 依存マップは表示されません。

新しいコンテキストを選択したときに、SOA 依存マップのノード数が com.wily.introscope.soa.dependencymap.ui.view.nodecount の値を超えている場合は、エラー メッセージが表示され、SOA 依存マップが前のビューに戻ります。 たとえば、エージェントの［物理モード］ビューからサーバの［物理モード］ビューに変更したときに、ノード数が制限を超えている場合は、エラー メッセージが表示され、SOA 依存マップがエージェントの［物理モード］ビューに戻ります。

［すべてのオペレーションを表示］または［全てのサービスを表示］を使用して表示情報を展開したときに、ノード数が制限を超えている場合は、エラー メッセージが表示され、SOA 依存マップに最新の追加された SOA 依存マップ ノードが表示されます。

### プロパティ設定

このプロパティは、0 より大きい任意の自然整数に設定できます。

### デフォルト

200 個のマップ ノード

### 例

```
com.wily.introscope.soa.dependencymap.ui.view.nodecount=200
```

### 注

このプロパティの変更を有効にするには、Workstation を再起動する必要があります。

## com.wily.introscope.soa.dependencymap.ui.view.edgecount

このプロパティは、Workstation の SOA 依存マップに表示されるマップの辺の最大数を指定します。

Investigator のノードを選択したときに、SOA 依存マップの辺数が com.wily.introscope.soa.dependencymap.ui.view.edgecount プロパティの値を超えている場合は、エラー メッセージが表示されます。 SOA 依存マップは表示されません。

新しいコンテキストを選択したときに、SOA 依存マップの辺数が com.wily.introscope.soa.dependencymap.ui.view.edgecount プロパティの値を超えている場合は、エラー メッセージが表示され、SOA 依存マップが前のビューに戻ります (SOA 依存マップは表示されません)。たとえば、エージェントの［物理モード］ビューからサービスの［物理モード］ビューに変更したときに、ノードの辺数が制限を超えている場合は、エラー メッセージが表示され、SOA 依存マップがエージェントの［物理モード］ビューに戻り、マップは表示されません。

［すべてのオペレーションを表示］または［全てのサービスを表示］を使用して表示情報を展開したときに、辺数が制限を超えている場合は、エラー メッセージが表示され、SOA 依存マップに最新の追加された SOA 依存マップの辺が表示されます。

SOA 依存マップがアプリケーション パフォーマンスに影響を及ぼすことを防止するために、デフォルト値を 1000 から 400 に減らすことによってマップのサイズおよび複雑さを制限することができます。

**プロパティ設定**

このプロパティは、0 より大きい任意の自然整数に設定できます。

**デフォルト**

1000

**推奨値**

400

**例**

```
com.wily.introscope.soa.dependencymap.ui.view.edgecount=250
```

**注**

このプロパティの変更を有効にするには、Workstation アプリケーションを再起動する必要があります。

## soa.dashboard.typeviewer.average.enable

選択したオペレーションに関して収集された偏差データを有効にし、以下の場所の［平均応答時間偏差］グラフに表示します。

- ［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボード
- ［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボード
- ［クリティカル（上位）］タブ
- ［最も依存度が高い］タブ

偏差データがダッシュボードやタブに表示されるのは、偏差データがエージェントによって Enterprise Manager にレポートされ、Workstation に表示できる場合のみです。 偏差データのレポートの詳細については、「［偏差］タブで Deviation メトリックを表示する (P. 73)」を参照してください。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - 選択したオペレーションのデータを有効にし、［平均応答時間偏差］グラフに表示します。
- false - 選択したオペレーションのデータを［平均応答時間偏差］グラフに表示しません。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.average.enable=true
```

### 注

このプロパティは *IntroscopeEnterpriseManager.properties* ファイルまたは *IntroscopeWorkstation.properties* ファイルで定義できます。このプロパティが *IntroscopeWorkstation.properties* ファイルと *IntroscopeEnterpriseManager.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.dashboard.typeviewer.errors.enable

選択したオペレーションに関して収集された偏差データを有効にし、以下の場所の［間隔偏差ごとのエラー］グラフに表示します。

- ［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボード
- ［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボード
- ［クリティカル（上位）］タブ
- ［最も依存度が高い］タブ

偏差データがダッシュボードやタブに表示されるのは、偏差データがエージェントによって Enterprise Manager にレポートされ、Workstation に表示できる場合のみです。 偏差データのレポートの詳細については、「［偏差］タブで Deviation メトリックを表示する (P. 73)」を参照してください。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - 選択したオペレーションのデータを有効にし、［Errors Per Interval （間隔ごとのエラー数）］グラフに表示します。
- false - 選択したオペレーションのデータを［Errors Per Interval （間隔ごとのエラー数）］グラフに表示しません。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.response.enable=true
```

### 注

このプロパティは *IntroscopeEnterpriseManager.properties* ファイルまたは *IntroscopeWorkstation.properties* ファイルで定義できます。このプロパティが *IntroscopeWorkstation.properties* ファイルと *IntroscopeEnterpriseManager.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.dashboard.typeviewer.response.enable

選択したオペレーションに関して収集された偏差データを有効にし、以下の場所の［Responses Per Interval （間隔ごとの応答数）偏差］グラフに表示します。

- ［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボード
- ［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボード
- ［クリティカル（上位）］タブ
- ［最も依存度が高い］タブ

偏差データがダッシュボードやタブに表示されるのは、偏差データがエージェントによって Enterprise Manager にレポートされ、Workstation に表示できる場合のみです。 偏差データのレポートの詳細については、「［偏差］タブで Deviation メトリックを表示する (P. 73)」を参照してください。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - 選択したオペレーションのデータを有効にし、［Responses Per Interval （間隔ごとの応答数）偏差］グラフに表示します。
- false - 選択したオペレーションのデータを［間隔ごとの応答数偏差］グラフに表示しません。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.response.enable=true
```

### 注

このプロパティは *IntroscopeEnterpriseManager.properties* ファイルまたは *IntroscopeWorkstation.properties* ファイルで定義できます。このプロパティが *IntroscopeWorkstation.properties* ファイルと *IntroscopeEnterpriseManager.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager を再起動する必要があります。

## soa.dashboard.typeviewer.mostcritical.enable

コンソールの［最もクリティカルなオペレーション］ダッシュボードと Investigator の［クリティカル（上位）］タブの表示を有効にします。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - コンソールのダッシュボード メニューに［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボード オプションを表示し、Investigator に［クリティカル（上位）］タブを表示します。
- false - ［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボードおよび［クリティカル（上位）］タブへのアクセスを無効にします。

このプロパティを false に設定すると、Investigator 内のどのノードについても［クリティカル（上位）］タブが表示されなくなります。 コンソールのダッシュボード ドロップダウン リストには［最もクリティカルなオペレーション］ダッシュボードが表示され続けますが、選択してもダッシュボードは表示されません。 ［最もクリティカルなオペレーション］ダッシュボードの代わりに、このダッシュボードを使用できないことを示すエラー メッセージが表示されます。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.mostcritical.enable=true
```

### 注

このプロパティは IntroscopeEnterpriseManager.properties ファイルまたは IntroscopeWorkstation.properties ファイルで定義できます。 このプロパティが *IntroscopeWorkstation.properties* ファイルと *IntroscopeEnterpriseManager.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager と Workstation を再起動する必要があります。

## soa.dashboard.typeviewer.mostcritical.count

すべてのエージェントによって［SOA パフォーマンス - 最もクリティカルなオペレーション］ダッシュボードと［クリティカル（上位）］タブにレポートされるクリティカルなオペレーションの総数を決定します。

### プロパティ設定

最小値は 0 です。

最大値は *com.wily.introscope.soa.deviation.count.per.metric* プロパティの値（デフォルト値は 25）以下である必要があります。

### デフォルト

10 個のオペレーション

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.mostcritical.count=5
```

### 注

このプロパティは *IntroscopeEnterpriseManager.properties* ファイルまたは *IntroscopeWorkstation.properties* ファイルで定義できます。このプロパティが *IntroscopeEnterpriseManager.properties* ファイルと *IntroscopeWorkstation.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager と Workstation を再起動する必要があります。

## soa.dashboard.typeviewer.mostdependent.enable

コンソールの［最も依存度が高いオペレーション］ダッシュボードと Investigator の［最も依存度が高い］タブの表示を有効にします。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - コンソールのダッシュボード メニューに［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボード オプションを表示し、Investigator に［最も依存度が高い］タブを表示します。
- false - ［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボードおよび［最も依存度が高い］タブへのアクセスを無効にします。

このプロパティを false に設定すると、Investigator に［最も依存度が高い］タブが表示されなくなります。 コンソールのダッシュボード ドロップダウン リストには［最も依存度が高いオペレーション］ダッシュボードが表示され続けますが、選択してもダッシュボードは表示されません。［最も依存度が高いオペレーション］ダッシュボードの代わりに、このダッシュボードを使用できないことを示すエラー メッセージが表示されます。

### デフォルト

true

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.mostdependent.enable=true
```

### 注

このプロパティは *IntroscopeEnterpriseManager.properties* ファイルまたは *IntroscopeWorkstation.properties* ファイルで定義できます。このプロパティが *IntroscopeEnterpriseManager.properties* ファイルと *IntroscopeWorkstation.properties* ファイルの両方で定義されている場合は、*IntroscopeWorkstation.properties* のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager と Workstation を再起動する必要があります。

## soa.dashboard.typeviewer.mostdependent.count

すべてのエージェントによって［SOA パフォーマンス - 最も依存度が高いオペレーション］ダッシュボードと［最も依存度が高い］タブにレポートされる最も依存度が高いオペレーションの総数を決定します。

### プロパティ設定

最小値は 0 です。

最大値は com.wily.introscope.soa.deviation.count.per.metric プロパティの値（デフォルト値は 25）以下である必要があります。

### デフォルト

10 個のオペレーション

### 例

```
com.wily.introscope.soa.dashboard.typeviewer.mostdependent.count=5
```

### 注

このプロパティは IntroscopeEnterpriseManager.properties ファイルまたは IntroscopeWorkstation.properties ファイルで定義できます。 このプロパティが IntroscopeEnterpriseManager.properties ファイルと IntroscopeWorkstation.properties ファイルの両方で定義されている場合は、IntroscopeWorkstation.properties のプロパティ設定が使用されます。

このプロパティの変更を有効にするには、Enterprise Manager と Workstation を再起動する必要があります。

## workstation.soa.dependencymap.fetchmetrics

SOA 依存マップ内のメトリックの表示を制御します。

### プロパティ設定

このプロパティは、true または false に設定できます。

- true - Enterprise Manager から選択したプライマリ メトリックとヒント メトリックを要求します。 このプロパティを true に設定すると、SOA 依存マップ上のすべてのマップ ノードにプライマリ メトリックが表示されます。 その他のメトリックは、マップ ノードの上にマウス ポインタを置いたときにヒントとして表示されます。
- false - SOA 依存マップ上のノードにプライマリ メトリックを表示せず、ヒント ウィンドウにもその他のメトリックを表示しません。

### デフォルト

true

### 例

```
com.wily.introscope.workstation.soa.dependencymap.fetchmetrics=true
```

### 注

このプロパティの変更を有効にするには、Workstation を再起動する必要があります。

## workstation.traceview.crossprocess.duration.full

トランザクション追跡ビューアの［シーケンス ビュー］タブで、トランザクション追跡の全体の継続時間を追跡のデフォルトとして表示し、使用するかどうかを決定します。

全体の継続時間は、プロセスの開始から終了までの時間を使用して計算されます。これには、プロセスが実行した別のプロセスを待機する時間も含まれます。 この方法では、ルート追跡が同期トランザクションの中で最も遅くなります。 正味の継続時間は、全体の継続時間から追跡の非同期の子の継続時間を引くことで計算されます。

Workstation プロパティのファイルに変更を加えなかった場合は、［シーケンス ビュー］タブで、全体の継続時間を表示するか、またはデフォルトの全継続時間と共に正味の継続時間を表示するかを選択できます。

**プロパティ設定**

Workstation プロパティのファイルに *introscope.workstation.traceview.crossprocess.duration.full* プロパティのみを追加した場合は、［シーケンス ビュー］タブに全体の継続時間のみが表示されます。

Workstation プロパティのファイルに *introscope.workstation.traceview.crossprocess.duration.full* と *introscope.workstation.traceview.crossprocess.duration.net* を追加した場合は、［シーケンス ビュー］タブに全体の継続時間と正味の継続時間の両方が表示されます。また、デフォルトは値の小さいプロパティによって決定されます。

たとえば、全体の継続時間と正味の継続時間を選択できるようにしながら、正味の継続時間をデフォルトで表示する場合は、Workstation プロパティファイルに以下のようにプロパティを設定します。

```
com.wily.introscope.workstation.traceview.crossprocess.duration.full=2
com.wily.introscope.workstation.traceview.crossprocess.duration.net=1
```

**デフォルト**

1（全体の継続時間が表示される）

**例**

```
com.wily.introscope.workstation.traceview.crossprocess.duration.full=1
```

**注**

このプロパティの変更を有効にするには、Workstation を再起動する必要があります。

## workstation.traceview.crossprocess.duration.net

トランザクション追跡ビューアの［シーケンス ビュー］タブで、トランザクション追跡の正味の継続時間を追跡のデフォルトとして表示し、使用するかどうかを決定します。

正味の継続時間は、全体の継続時間から追跡の非同期の子の継続時間を引くことで計算されます。 この方法の主な利点は、同期トランザクションの最も遅いスレッドがすぐにわかることです。

Workstation プロパティのファイルに変更を加えなかった場合は、［シーケンス ビュー］タブで、全体の継続時間を表示するか、またはデフォルトの全継続時間と共に正味の継続時間を表示するかを選択できます。

### プロパティ設定

Workstation プロパティのファイルに *introscope.workstation.traceview.crossprocess.duration.net* プロパティのみを追加した場合は、［シーケンス ビュー］タブに正味の継続時間のみが表示されます。

Workstation プロパティのファイルに *introscope.workstation.traceview.crossprocess.duration.full* と *introscope.workstation.traceview.crossprocess.duration.net* を追加した場合は、［シーケンス ビュー］タブに全体の継続時間と正味の継続時間の両方が表示されます。また、デフォルトは値の小さいプロパティによって決定されます。

たとえば、全体の継続時間と正味の継続時間を選択できるようにしながら、正味の継続時間をデフォルトで表示する場合は、Workstation プロパティファイルに以下のようにプロパティを設定します。

```
com.wily.introscope.workstation.traceview.crossprocess.duration.full=2
com.wily.introscope.workstation.traceview.crossprocess.duration.net=1
```

### デフォルト

2（正味の継続時間がデフォルトとして表示されない）

### 例

```
com.wily.introscope.workstation.traceview.crossprocess.duration.net=1
```

### 注

このプロパティの変更を有効にするには、Workstation を再起動する必要があります。

# 付録 D: SOA 専用の WebView 構成プロパティ

WebView 構成プロパティを使用すると、WebView アプリケーションの動作やオペレーションを制御できます。 これらのプロパティを使用して、ユーザの環境に最もよく適合するように WebView をカスタマイズすることもできます。 WebView プロパティを使用すると、CA APM for SOA を特別に構成できます。

このセクションには、以下のトピックが含まれています。



## WebView プロパティの設定

CA APM for SOA には、デフォルトが設定された、SOA 専用の WebView プロパティがあります。 設定を行う前に、プロパティ名を *IntroscopeWebView.properties* ファイルに追加します。

**WebView プロパティを IntroscopeWebView.properties に追加する方法**

1. *<EM_Home>/config* ディレクトリの *IntroscopeWebView.properties* ファイルを開きます。
2. *IntroscopeWebView.properties* ファイルに追加する SOA 専用の WebView プロパティ (P. 464) を探します。
3. *IntroscopeWebView.properties* ファイルに、*com.wily.introscope* プレフィックスを含むプロパティの完全な名前を追加します。 例：
   `com.wily.introscope.soa.dependencymap.ui.view.edgecount`
4. 必要に応じてプロパティ値を設定します。 たとえば、このプロパティのデフォルト設定を使用するには、以下のようにします。
   `com.wily.introscope.soa.dependencymap.ui.view.edgecount=1000`
5. *IntroscopeWebView.properties* ファイルを保存して閉じます。

*IntroscopeWebView.properties* ファイルに WebView プロパティを追加した後は、必要に応じてプロパティ値を設定できます。

**WebView 構成プロパティの値を変更する方法**

1. *<EM_Home>/config* ディレクトリの *IntroscopeWebView.properties* ファイルを開きます。
2. 変更するプロパティを見つけ、ユーザ環境に合わせて新しい値を設定します。 例：
   `com.wily.introscope.soa.dependencymap.ui.view.edgecount=250`

   上記の設定は単に例として挙げたもので、CA APM for SOA の推奨設定ではありません。
3. *IntroscopeWebView.properties* ファイルを保存して閉じます。
4. プロパティの変更を有効にするため、WebView アプリケーションを再起動します。

# SOA 専用の WebView プロパティについて

IntroscopeWebView.properties ファイルには、以下の SOA 専用プロパティを設定できます。 すべてのプロパティは *com.wily.introscope* で始まりますが、読みやすくするため、リストではこのプレフィックスを付けずにプロパティを表示します。

**注**: その他の WebView プロパティの設定方法については、「*CA APM 設定および管理ガイド*」を参照してください。

soa.dependencymap.ui.view.nodecount (P. 465)

WebView の SOA 依存マップに表示されるマップ ノードの最大数を指定します。

com.wily.introscope.soa.dependencymap.ui.view.edgecount (P. 466)

WebView の SOA 依存マップに表示されるマップの辺の最大数を指定します。

## soa.dependencymap.ui.view.nodecount

このプロパティは、WebView の SOA 依存マップに表示されるマップ ノードの最大数を指定します。

Investigator のノードを選択したときに、SOA 依存マップのノード数が com.wily.introscope.soa.dependencymap.ui.view.nodecount プロパティの値を超えている場合は、エラー メッセージが表示されます。 SOA 依存マップは表示されません。

新しいコンテキストを選択したときに、SOA 依存マップのノード数が com.wily.introscope.soa.dependencymap.ui.view.nodecount プロパティの値を超えている場合は、エラー メッセージが表示され、SOA 依存マップは表示されません。 たとえば、エージェントの［物理モード］ビューからサービスの［物理モード］ビューに変更したときに、ノード数が制限を超えている場合は、エラー メッセージが表示され、マップは表示されません。

［すべてのオペレーションを表示］または［全てのサービスを表示］を使用して表示情報を展開したときに、ノード数が制限を超えている場合は、エラー メッセージが表示され、SOA 依存マップに最新の追加された SOA 依存マップ ノードが表示されます。

SOA 依存マップがアプリケーション パフォーマンスに影響を及ぼすことを防止するために、プロパティのデフォルト値を 200 から、より低い値に減らすことによってマップのサイズおよび複雑さを制限することができます。

**プロパティ設定**

このプロパティは、0 より大きい任意の自然整数に設定できます。

**デフォルト**

200 個のマップ ノード

**推奨値**

200 個のマップ ノード

**例**

```
com.wily.introscope.soa.dependencymap.ui.view.nodecount=200
```

**注**

このプロパティの変更を有効にするには、WebView アプリケーションを再起動する必要があります。

## com.wily.introscope.soa.dependencymap.ui.view.edgecount

このプロパティは、WebView の SOA 依存マップに表示されるマップの辺の最大数を指定します。

Investigator のノードを選択したときに、SOA 依存マップの辺数が com.wily.introscope.soa.dependencymap.ui.view.edgecount の値を超えている場合は、エラー メッセージが表示されます。 SOA 依存マップは表示されません。

新しいコンテキストを選択したときに、SOA 依存マップの辺数が com.wily.introscope.soa.dependencymap.ui.view.edgecount の値を超えている場合は、エラー メッセージが表示され、SOA 依存マップは表示されません たとえば、エージェントの［物理モード］ビューからサービスの［物理モード］ビューに変更したときに、辺数が制限を超えている場合は、エラー メッセージが表示され、マップは表示されません。

［すべてのオペレーションを表示］または［全てのサービスを表示］を使用して表示情報を展開したときに、辺数が制限を超えている場合は、エラー メッセージが表示され、SOA 依存マップに最新の追加された SOA 依存マップの辺が表示されます。

SOA 依存マップがアプリケーション パフォーマンスに影響を及ぼすことを防止するために、デフォルト値を 1000 から 250 に減らすことによってマップのサイズおよび複雑さを制限することができます。

### プロパティ設定

このプロパティは、0 より大きい任意の自然整数に設定できます。

### デフォルト

1000 個のマップの辺

### 推奨値

250 個のマップの辺

### 例

```
com.wily.introscope.soa.dependencymap.ui.view.edgecount=250
```

### 注

このプロパティの変更を有効にするには、WebView アプリケーションを再起動する必要があります。