CA Performance Management Data Aggregator

パワー ユーザ認定ガイド

2.4

ca technologies

CA Technologies 製品リファレンス

このマニュアルが参照している CA Technologies の製品は以下のとおりです。

- CA Performance Management Data Aggregator （Data Aggregator）
- Data Collector
- CA Performance Center

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: はじめに

このセクションには、以下のトピックが含まれています。

このガイドについて

Data Aggregator は、ベンダー自己認定インターフェースを提供することにより、CA Performance Management 監視サポートを拡張できるようにしています。 自己認定により、ベンダー MIB からの新規またはカスタムのコンポーネントおよびメトリックで、Data Aggregator 内の既存のメトリック ファミリを補足できます。

ベンダー認定を作成するか、または既存のデバイスまたはコンポーネントとなるサポートを変更できます。 このガイドでは、カスタム メトリック ファミリおよびカスタム ベンダー サポートを伴う新技術のサポートを追加する方法について説明します。

このガイドは、カスタム ベンダー認定およびメトリック ファミリを作成するための高度なシナリオを扱います。 XML およびスキーマのファイルの基礎知識が求められます。

注: Data Aggregator には、カスタムのベンダー認定とメトリック ファミリを作成するための基本的な方法と高度な方法があります。 基本的方法はより単純なプロセスであり、ユーザ インターフェースを使用して、サポートされた既存の技術（メトリック ファミリ）のベンダー サポートを追加することから構成されます。 この方法は、多くのユーザの要件を満たします。 一方、高度な方法は、ファクトリ認定の形式に基づいており、完全な機能セットを利用できます。 このガイドでは、高度な認定方法について説明します。 基本的な認定方法の詳細については、「Data Aggregator 基本自己認定ガイド」を参照してください。

このガイドは全体にわたって特定の例を使用しています。 管理者がフレーム リレー恒久仮想回線（PVC）を監視するとします。 Data Aggregator は、フレーム リレー PVC を監視するためのサポートを標準では備えていませんが、自己認定を使用してこのサポートを生成できます。 このガイドでは、コンポーネントを定義し、カスタム メトリック ファミリを作成し、フレーム リレー恒久仮想回線（PVC）を監視するカスタム ベンダー認定を定義する手順を提供します。

デバイス サポートの仕組み

Data Aggregator はメトリック ファミリおよびベンダー認定を使用して、ベンダー デバイスをサポートします。 これらのコンポーネントを組み合わせることで、Data Aggregator がデバイスの設定メトリックと運用メトリックを収集する方法が決まります。 Data Aggregator でのデバイス サポートの仕組みを理解すると、デバイスが Data Aggregator で正しくサポートされているかどうかを判断するのに役立ちます。 デバイスが正しくサポートされていない場合は、このプロセスを理解することによって、望ましい結果を得られるように設定を調整することができます。

注: 必要な場合は、メトリック ファミリ、ベンダー認定、または両方をカスタマイズして、ベンダー デバイスに対するサポートを追加できます。

Data Aggregator は以下の設定機能を使用して、デバイスをサポートします。

1. **ディスカバリ プロファイル** - 環境内で Data Aggregator がどのアイテムを検出するのかを制御します。通常は、IP アドレスの範囲をベースとします。ディスカバリ プロセスは、検索する各アイテムの「タイプ」を特定します。
2. **デバイス コレクション** - インベントリを関連アイテムのグループにまとめます。 アイテム タイプおよび IP アドレスに基づいて、アイテムは自動的にデバイス コレクションに追加されます。
3. **監視プロファイル** - デバイス コレクションのポーリング レートを制御し、ポーリングするメトリック ファミリを決定します。 監視プロファイルは 1 つ以上のメトリック ファミリをポーリングすることができます。

   注: システムがポーリング トラフィックで過負荷にならないことを保証するために、監視プロファイルを使用して、さまざまなメトリック セットのポーリング レートを調整します。

4. **メトリック ファミリ** - 監視プロファイルに対してどのメトリックが収集されるかを制御します。メトリック ファミリは 1 つ以上のベンダー認定と関連付けられ、優先順にリスト表示されます。

   注: 一貫性のあるデータのレポートを保証するために、監視プロファイルでメトリック ファミリを再利用します。

5. **ベンダー認定** - ベンダー MIB の属性を、メトリック ファミリ内のメトリックにマップします。 また、アイテムから収集されたメトリックが CA Performance Center UI およびレポートで使用するためにどのようにフォーマットされるかを決定します。 アイテムに対して提供されるメトリックは、アイテムのベンダーによって変わる場合があります。 これらの値をマッピングすることにより、ベンダーに関係なく、メトリック値を一様にレポートします。 複数のベンダー認定は単一のメトリック ファミリと関連付けることができます。 そのような場合、Data Aggregator はベンダー認定のランク リストを使用してメトリック値をマップします。 Data Aggregator は、ポーリングされたアイテムと一致する、最も高い優先度のベンダー認定を使用してメトリック値を計算します。

   注: SNMP MIB などの MIB はシステムにインポートして、ベンダー認定のビルドの一部としてコンパイルすることができます。

例: ルータ デバイスのサポート

ディスカバリ プロファイルを実行すると、Data Aggregator はアイテムをルータとして検索し識別します。 ルータ管理対象アイテムは、［すべてのルータ］デバイス コレクションに自動的に追加されます。 このデバイス コレクションは、ルータ監視プロファイルに関連付けられます。これは CPU およびメモリのメトリック ファミリを使用して、デバイス上の CPU およびメモリのコンポーネントを検出します。 これらのメトリック ファミリは、これらコンポーネントのメトリック値の計算に使用するベンダー認定も決定します。 この監視プロファイルに基づいて、Data Aggregator は 5 分間隔でルータをポーリングし、これらメトリック ファミリのメトリック データを取得します。 たとえば、CPU メトリック ファミリには、CPU アイドル使用率、CPU システム使用率、および CPU Nice 使用率が含まれます。最後に、メトリック ファミリと関連付けられるベンダー認定は、未加工のメトリック データを矛盾なく計算し、フォーマットする方法を確定します。Data Aggregator は、ルータについて収集されたメトリック データを格納します。CA Performance Center は、このデータを UI およびレポートで使用します。

自己認定によるデバイス サポート

Data Aggregator は事前定義済み認定を使用して、共通のベンダー デバイスをサポートしています。認定は、デバイスの設定および運用上のメトリックを収集する方法を指定します。Data Aggregator は認定に対して以下の方法を使用します。

- メトリック ファミリ
- ベンダー認定

ご使用のデバイスの事前定義済み認定が Data Aggregator によって提供されない場合、どのようにデータを収集しますか。 デバイスのサポートを*自己認定*できます。

注: メトリック ファミリおよびベンダー認定はグローバルです（つまり、テナントを認識しません）。テナントの詳細については、「CA Performance Center 管理者ガイド」を参照してください。

Data Aggregator 内の自己認定サポートでは、カスタム ベンダー認定、カスタム メトリック ファミリ、またはその両方を作成できます。 必要とする方法を、以下のように決定します。

- **ベンダー認定のみ** - デフォルトでは、望んでいるメトリック セットがポーリングされますが、Data Aggregator はこれらのメトリックに対してユーザのデバイス ベンダー MIB をサポートしていません。たとえば、Data Aggregator は、CPU 使用率などデータを収集するための CPU メトリック ファミリを提供しています。ただし、Bargain Server Company が製造するサーバについて CPU データを収集するとします。製造元が提供した MIB を使用すると、自身のサーバ CPU に対するカスタム ベンダー認定を作成できます。
- **メトリック ファミリのみ** - デフォルトでは、自身のデバイス ベンダー MIB のサポートのみが含まれています。ただし、MIB がサポートしているいくつかのメトリックはポーリングされません。 たとえば、自身のベンダー MIB はプロセスに対するメトリックをサポートしていますが、Data Aggregator は、そのメトリック データを収集するための「プロセス」メトリック ファミリを提供していません。
- **両方の方法** - Data Aggregator が、デバイス ベンダー MIB、またはそのメトリックに対するサポートを提供していない場合は、メトリック ファミリとベンダー認定の両方を作成することができます。

カスタム認定シナリオ

CA Performance Management は、ネットワーク インフラストラクチャ内の共通のベンダー、メトリック、コンポーネントを標準でサポートします。技術認定ポータルには、Data Aggregator バージョン、ベンダー認定、メトリック ファミリ別にすぐに使用できる認定が一覧表示されています。

http://serviceassurance.ca.com/im/ http://serviceassurance.ca.com/im/

ただし、カスタム CA Performance Management 認定の作成により、インフラストラクチャ管理能力を最大化できます。以下に対して CA Performance Management サポートを必要とする場合、カスタマイズします。

- 新しい技術：
  - 新規ファイルを定義します。
  - カスタム メトリック ファミリを作成します。
- 既存の技術用の新しいメトリック：
  - 事前定義済み（ファクトリ）メトリック ファミリからカスタム メトリック ファミリを作成します。
  - カスタム メトリック ファミリ用のカスタム ベンダー認定を定義します。
- 新しいベンダー：
  - ファクトリ メトリック ファミリ用のカスタム ベンダー認定を作成します。
  - カスタム メトリック ファミリ用のカスタム ベンダー認定を作成します。

前提条件

以下の情報は、ベンダー認定 XML を使用するための前提条件を説明しています。

- CA Performance Management Data Aggregator リリース 2.2.00 以降が必要です。
- 「認定」についての十分な知識、および［ベンダー認定］ウィザードを使用した経験が必要です（「CA Performance Management Data Aggregator 自己認定ガイド」参照）。

- XML エディタ。

  スキーマで XML を検証する XML エディタが推奨されます。XSD ファイルを使用するためには XML エディタのガイドラインに従ってください。

  注: このガイドの例は、Microsoft® Download Center から無料でダウンロード可能な Windows 用の XML Notepad 2007 で作成および検証されました。

- MIB ブラウザ。 たとえば、iReasoning の無料バージョンがサポートされています。
- CA Performance Management Data Aggregator サーバへの接続。
- このガイドで使用する例をサポートするには、FRAMERELAY-RFC1315-MIB および IF-MIB が必要です。

第 2 章: スキーマおよび例ファイルのダウンロード

カスタム メトリック ファミリおよびベンダー認定 XML ファイルを作成する前に、関連するスキーマ XSD ファイルおよび XML サンプル ファイルをダウンロードして確認します。 自身の XML ファイルを確認するにはスキーマが必要です。 自身の XML ファイルを作成する前にこれらの例ファイルを確認しておくと、XML コンテンツの正確性の保証に役立ちます。

Data Aggregator と共に提供されるスキーマ ファイルには、エレメント タイプ、発生および許可されている長さに関する詳細情報があります。ファイルには、許可されている文字および命名規則など、より多くの情報を提供する注釈も含まれます。

Web ブラウザの［アドレス］フィールドに以下の URL を入力して、スキーマおよび例 XML ファイルをダウンロードします。 *ホスト名*は Data Aggregator ホストで、デフォルト *ポート*は 8581 です。

- http://*hostname*:*port*/resource/xsd/IMDBCertificationFacet.xsd
- http://*hostname*:port/resource/xsd/ComponentFacet.xsd
- http://*hostname*:*port*/resource/xsd/ItemSyncDefinition.xsd
- http://*hostname*:*port*/resource/xsd/SNMPCertificationFacet.xsd
- http://*hostname*:*port*/resource/xsd/CertificationFacet.xsd
- http://*hostname*:*port*/resource/xsd/CertificationFacet.xsd
- http://*hostname*:*port*/resource/xsd/webservices.xsd
- http://*hostname*:*port*/resource/xsd/basewebservices.xsd
- http://*hostname*:*port*/resource/xsd/datamodel.xsd

第 3 章: カスタム コンポーネントの作成

このセクションには、以下のトピックが含まれています。

コンポーネント XML テンプレートの作成

カスタム コンポーネント作成用のテンプレートとして使用できる XML サンプル ファイルを作成するには、適切な REST クライアントを使用します。

まず最初に、既存のコンポーネント定義リストを取得します。 次に、必要とするコンポーネントがすでにサポートされているかどうかを確認します。

次の手順に従ってください:

1. Data Aggregator サーバに接続されている REST クライアントをセットアップします。
2. REST クライアントで、Data Aggregator Web サービス API に対して以下の URL を入力します。

   `http://da_hostname:8581/typecatalog/components`

   すべての使用可能なコンポーネントのリストが表示されます。
3. 必要とするコンポーネントがすでにサポートされているかどうかを確認します。

必要とするコンポーネントがまだサポートされていない場合は、作成するカスタム コンポーネントに類似している単一のコンポーネント定義を参照してエクスポートします。

次の手順に従ってください:

1. 必要とするコンポーネントに類似している特定のコンポーネントを取得するには、以下の URL を入力します。

   `http://da-hostname:8581/typecatalog/components/name`

   name

   NormalizedCPUInfo など、既存の特定のコンポーネントの名前です。

2. ［Method］タブで GET を選択します。
3. メソッドを実行します。

   返される XML にはコンポーネント定義が含まれます。

   新しいコンポーネントを作成するためにテンプレートとしてこの XML を使用できます。
4. コンポーネント XML をテキスト ファイルにコピーし、必要に応じて変更します。XML 構造の例については、「コンポーネント XML 構造について (P. 18)」を参照してください。

コンポーネント XML 構造について

デバイス コンポーネントは、デバイスと関連付けられているコンポーネント アイテムのクラスを定義します。 事前に定義された多くのコンポーネントが提供されますが、通常、カスタム コンポーネントはカスタム メトリック ファミリに対して定義されます。 コンポーネントは、CA Performance Center に対してコンポーネント アイテムを同期するオプションの ItemSyncDefinition を定義できます。 その後コンポーネントは、インベントリ リスト、グループ、およびコンテキスト ページ内で参照できます。

フレーム リレー PVC の例をサポートするコンポーネント定義 XML の例を以下に示します。 新しいコンポーネント「frPVC」が追加されました。

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Auto-generated by the type catalog local manager.-->
<DataModel namespace="http://im.ca.com/inventory"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="ComponentFacet.xsd">
```

```
  <FacetType name="frPVC">
    <Documentation>A Frame Relay PVC</Documentation>
    <FacetOf namespace="http://im.ca.com/core" name="Item" />
    <Component>true</Component>
    <ItemSyncDefinition itemTypeName="component"
itemSubtypeName="frpvc" itemTypeLabel="FrameRelayPVC"
itemTypeLabelPlural="FrameRelayPVCs" categorize="false"
groupBy="false" context="true">
    </ItemSyncDefinition>
  </FacetType>
</DataModel>
```

カスタム コンテキスト ページへのリンクを有効にするには、ItemSyncDefinition の context を “true” に設定します。 このページには、frPVC デバイス コンポーネントから移動できます（［インベントリ］-［デバイス コンポーネント］リストに表示）。 "true" に設定すると、メトリック ファミリを "context" として選択できるようになります。これにより、カスタム メトリック ファミリが動的トレンド タイプのチャートで使用可能になります。詳細については、ItemSyncDefinition (P. 20) を参照してください。

基本プロパティ

カスタム コンポーネントの基本プロパティは、ユーザが作成したその他のカスタム コンポーネントとの区別に役立ちます。

FacetType/name

コンポーネント名を指定します。 各コンポーネントには、システム内部で識別するための一意の名前が必要です。 将来的に、同様のコンポーネントとのネーミング競合の可能性が最も小さいと考えられる名前を慎重に選択してください。 たとえば、これらのコンポーネント名が一意であることの確認に役立つネーミング スキームを定義します。

注: この名前は外部には表示されません。 ユーザ インターフェースにコンポーネント名を表示するには、ItemSyncDefinition、itemTypeLabel、および itemTypeLabelPlural エレメントを使用します。

更新の可否： いいえ

可能な値： 英数字およびアンダースコア。 ドットおよびダッシュは使用できません。

Documentation

コンポーネント用の内部コメントを指定します。これらのコメントを役立てるために、私たちは、ユーザがコンポーネントを追加あるいは変更した理由と日時を記述することを推奨します

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： なし

更新が有効化されるタイミング： 即時

更新の有効化に必要なアクション： なし

Component

このアイテムがコンポーネントであることを示します。

更新の可否： いいえ

可能な値： true

FacetOf

このコンポーネントがアイテムであることを示します。

更新の可否： いいえ

可能な値： namespace="http://im.ca.com/core" name="Item"

ItemSyncDefinition

ItemSyncDefinition 属性はオプションです。この属性は、コンポーネント アイテムが CA Performance Center でどのように同期され表示されるかを指定します。コンポーネント アイテムが指定されない場合、CA Performance Center の［インベントリ］リストには表示されません（たとえばデバイス コンポーネント）。 しかし、今までどおりカスタム ビューでレポートできます。

ItemSyncDefinition/itemTypeName

アイテムのタイプを指定します。カスタム コンポーネントについては、この値は「component」である必要があります。

更新の可否： いいえ

可能な値： Component

ItemSyncDefinition/itemSubtypeName

CA Performance Center でコンポーネントの内部名を指定します。 この値はすべてのコンポーネントに対して一意である必要があります。 将来的に標準およびカスタムのコンポーネントとの競合を回避する命名規則を使用します。たとえば、ユーザの組織「acmeFan」を表す接頭辞を使用します。

更新の可否： いいえ

可能な値： 英数字（すべてのコンポーネントに対して一意）

ItemSyncDefinition/itemTypeLabel

このタイプの単一コンポーネントを表示するときに使用するユーザ インターフェースのラベルを指定します。 たとえば、この値は［インベントリ］-［デバイス コンポーネント］の UI ［タイプ］列）で使用されます。

更新の可否： はい

可能な値: プレーン テキスト (すべてのコンポーネントに対して一意)

更新による影響: ラベルは CA Performance Center ［インベントリ］ユーザ インターフェースに表示されます。

更新が有効化されるタイミング： 再同期が発生し、15 分以内に更新が完了する場合。

更新の有効化に必要なアクション： なし

ItemSyncDefinition/itemTypeLabelPlural

このタイプの複数のコンポーネントを表示するときに使用するユーザ インターフェースのラベルを指定します。 ［インベントリ］メニュー（**groupBy** を参照）および［グループ］名（**categorize** を参照）によって使用されます。

更新の可否： はい

可能な値: プレーン テキスト (すべてのコンポーネントに対して一意)

更新による影響： ラベルは CA Performance Center ［インベントリ］の UI に表示されます。

更新が有効化されるタイミング： 再同期が発生し、15 分以内に更新が完了する場合。

更新の有効化に必要なアクション： なし

ItemSyncDefinition/categorize

CA Performance Center で、［インベントリ］-［すべてのアイテム］の下にインベントリ グループが作成されるようにします。このグループには、このコンポーネント タイプのアイテムがすべて含まれます。グループは「*{itemTypeLabelPlural}*」と命名されます。

注: 作成されるインベントリ グループはダッシュボードのレポートには使用できません。 このグループはインベントリ目的専用です。 グループがレポート用に選択された場合、データは表示されません。 その他のデバイス ベースのインベントリ グループ（［インベントリ］-［すべてのアイテム］）は、ルータおよびサーバのようにレポートに使用できます。ただし、コンポーネント ベースのインベントリ グループ（デバイス コンポーネントなど）は使用できません。

更新の可否： はい

可能な値： true、false

更新による影響： CA Performance Center でインベントリ グループを作成または削除します。 CA Performance Center グループ管理者の場合、［グループの管理］ページで、［インベントリ］-［すべてのアイテム］-［*{itemTypeLabelPlural}*］の下にグループが作成されます。

更新が有効化されるタイミング： 再同期が発生し、30 分以内に更新が完了する場合。

更新の有効化に必要なアクション： 通常 30 分以内にアイテムがグループに表示されます。 表示されない場合は、手動で Data Aggregator データ ソースを再同期します。［再同期］の確認ボタンをクリックする前に、［完全な再同期を実行］を必ず選択してください。

ItemSyncDefinition/groupBy

CA Performance Center で、このコンポーネント アイテム タイプのすべてのアイテムを表示するためのインベントリ メニュー アイテム（［インベントリ］の下）が作成されるようにします。 メニューは「*{itemTypeLabelPlural}*」と命名されます。この属性はまた、ビュー コンテキストを設定する場合に［コンテキスト タイプ］ドロップダウン メニューにコンポーネント タイプが表示されるようにします。 False の場合、コンポーネントは、［インベントリ］の［デバイス コンポーネント］テーブルに、「*{itemTypeLabel}*」タイプで表示されます。groupBy プロパティはグループを作成しません（「**categorize**」を参照）。

更新の可否： はい

可能な値： true、false

更新による影響： true の場合、メニュー アイテムが作成され、false の場合は削除されます。

更新が有効化されるタイミング： 再同期が発生し、15 分以内に更新完了が可能な場合。

更新の有効化に必要なアクション： なし

ItemSyncDefinition/context

各コンポーネント アイテム名を［インベントリ］コンポーネント ビュー内でコンテキスト ハイパーリンクにします。このリンクをクリックすると、個別のコンポーネント コンテキスト ページに移動します。

更新の可否： はい/いいえ

可能な値： プレーン テキスト

更新による影響： 各コンポーネント アイテム名を［インベントリ］コンポーネント ビュー内でコンテキスト ハイパーリンクにします。

更新が有効化されるタイミング： 再同期が発生し、15 分以内に更新完了が可能な場合。

更新の有効化に必要なアクション： なし

ItemSyncDefinition の削除

ItemSyncDefinition を完全に削除するには特定の手順が必要です。

次の手順に従ってください:

1. <ItemSyncDefinition> の開始タグおよび終了タグを含む ItemSyncDefinition セクションを削除します。
2. Data Aggregator へのコンポーネント変更を適用した後、管理者として CA Performance Center にログインします。
3. ［データ ソース］ページで DataAggregator データソースを選択します。
4. ［再同期］ボタンをクリックします。
5. ［完全な再同期を実行］オプションを選択し、［再同期］の確認ボタンをクリックします。

   再同期処理が開始します。 変更が同期されるには 15～30 分必要です。

   このプロセスが完了したら、ItemSyncDefinition がコンポーネントに対して定義した CA Performance Center の動作がすべて削除されます。

サポートされていないカスタム コンポーネントのプロパティ

以下のプロパティは、カスタム コンポーネントに対してサポートされていません。

- 属性
- Web サービス
- ItemSyncDefinition/isDeviceComponent*
- ItemSyncDefinition/mapped*
- ItemSyncDefinition/ItemProperty

* これらのプロパティは XML 内に存在することができますが、true に設定できません。

カスタム コンポーネントのインポート

カスタム コンポーネントを Data Aggregator にインポートしたら、新しいコンポーネントはカスタム メトリック ファミリをサポートするために使用可能です。

Data Aggregator は、新しいコンポーネントのインポートをサポートするために Web サービスを提供します。 REST クライアントを使用します。 RESTClient を CA Performance Management と使用することをお勧めします。インストールしていない場合、http://code.google.com/p/rest-client/ から RESTClient GUI JAR ファイルを取得できます。

RESTClient を使用する場合、以下の情報を考慮してください。

- JAR ファイルをダブルクリックして起動します。
- POST XML を実行する場合、Charset が UTF-8 に設定されていることを確認します。

  この設定を表示して確認するには、［Edit Content-Type & Charset］ボタンをクリックします。
- また以下のように Response Body をオート インデントできます。
  1. ［Tools］メニューの［Options］をクリックします。
  2. ［Etc.］タブを選択します。
  3. ［Auto-indent Response Body］を選択し、［OK］をクリックします。

次の手順に従ってください:

1. URL として http://*da-hostname*:8581/typecatalog/components を入力します。
2. メソッドとして POST を選択します。
3. Body 設定内の「Body Content-type」として「application/xml」を選択します。

   重要: Content-type の設定に失敗すると 404 エラーとなります。
4. カスタム コンポーネント XML を［Body］フィールドにコピーして貼り付けます。
5. メソッドを実行します。

カスタム コンポーネントがインポートされます。エラーが発生しない場合、［HTTP Response］セクション内の［Status］フィールドには以下が表示されます。

```
HTTP/1.1 200 OK
```

注: その他のリターン コードは、カスタム コンポーネントを更新する間にエラーが発生したことを示します。エラーを修正し、別の POST を実行することによりコンポーネントの更新を再試行します。

重要: データの損失を回避するには、ベンダー認定、メトリック ファミリ、またはコンポーネントを作成または更新するたびに、展開ディレクトリを必ずバックアップします。

カスタム コンポーネント結果の確認

カスタム コンポーネント XML をインポートした後、結果を確認します。この例では、frPVC コンポーネントが正常にインポートされたことを確認します。

次の手順に従ってください:

1. REST クライアントに以下の URL を入力します。

   `http://da_hostname:8581/typecatalog/components/frPVC`

   カスタムのフレーム リレー コンポーネント XML がページに表示されます。

新しいコンポーネントを検出することもできます。 同期後、［インベントリ］-［デバイス コンポーネント］リストを使用してコンポーネントを表示できます。

カスタム コンポーネントの更新

既存のカスタム コンポーネントは更新できます。

注: コンポーネントを更新する場合のタグまたは属性を更新する影響の詳細については、「カスタム コンポーネント XML の詳細」を確認してください。 具体的には、特定の属性説明を確認します。

次の手順に従ってください:

1. URL フィールドに以下のアドレスを入力します。

   `http://da_hostname:8581/typecatalog/components/name`

   name

   更新するユーザのカスタム コンポーネントの名前です。

2. ［Method］タブで PUT を選択します。
3. 更新したコンポーネント XML を［Body］タブの［編集］フィールドにコピーして貼り付け、［Content-type］を application/xml に設定します。

   重要: Content-TYpe を設定しないと、404 エラーとなります。

4. ［URL］フィールドの隣の［Go］ボタンをクリックします。

   カスタム コンポーネントが更新されます。 エラーが発生しない場合、［HTTP Response］セクション内の［Status］フィールドには以下が表示されます。

   `HTTP/1.1 200 OK`

   注: その他のリターン コードは、カスタム コンポーネントを更新する間にエラーが発生したことを示します。 エラーを修正し、別の PUT を実行することによりコンポーネントを更新することを再試行します。

   重要: データの損失を回避するには、ベンダー認定、メトリック ファミリ、またはコンポーネントを作成または更新するたびに、展開ディレクトリを必ずバックアップします。

詳細:

コンポーネント XML 構造について (P. 18)

第 4 章: カスタム メトリック ファミリの作成

インポートが成功すると、新しいメトリック ファミリがカスタム ベンダー認定をサポートするために使用可能になります。また、ディスカバリおよびポーリング用の監視プロファイルと関連付けるために使用できます。

このセクションには、以下のトピックが含まれています。

ガイドの構造

このガイドでは、必要な XXX について説明します。 TO COME

VC：

- vcs の作成
- vcs の編集
- vc をインポートする
- vc をエクスポートする

ファクトリ メトリック ファミリは、監視する最も一般的なメトリック属性を定義します。 ただし、新しいメトリック属性のデータを収集する場合には、カスタム メトリック ファミリを作成できます。 たとえば、プロセス データの収集用のメトリック ファミリがない場合は、それを作成します。次に、新しいメトリック ファミリを使用して、プロセスのメトリック属性を監視するためのベンダー認定を作成します。

メトリック ファミリ XML テンプレートの作成

優先の REST クライアントを使用して、ユーザのカスタム メトリック ファミリの作成に際してテンプレートとして使用できる XML 例ファイルを作成できます。

まず最初に、既存のメトリック ファミリのリストを取得します。その後、必要とするメトリック ファミリがすでにサポートされているかどうかを確認できます。

次の手順に従ってください:

1. Data Aggregator サーバに接続されている REST クライアントをセットアップします。
2. REST クライアントの Data Aggregator Web サービス API について、以下の URL を入力します。

   `http://da_hostname:8581/typecatalog/metricfamilies`

   すべての使用可能なメトリック ファミリのリストが表示されます。
3. 必要とするメトリック ファミリがすでにサポートされているかどうかを確認します。

必要とするメトリック ファミリがまだサポートされていない場合、作成するカスタム メトリック ファミリに類似しているメトリック ファミリを参照してエクスポートします。

次の手順に従ってください:

1. 必要とするメトリック ファミリに類似している特定のメトリック ファミリを取得するには、以下の URL を入力します。

   `http://da_hostname:8581/typecatalog/metricfamilies/name`

   name

   NormalizedCPUInfo など、特定の既存のメトリック ファミリの名前です。
2. ［Method］タブで GET を選択します。
3. メソッドを実行します。

   返される XML にはメトリック ファミリ情報が含まれます。

カスタム メトリック ファミリを作成するためのテンプレートとしてこの XML を使用できます。

4. メトリック ファミリ XML をテキスト ファイルにコピーし、必要に応じて変更します。 XML 構造の例については、「メトリック ファミリ XML 構造について (P. 31)」を参照してください。

メトリック ファミリ XML 構造について

メトリック ファミリは、指定されたテクノロジーに対して収集およびレポートするメトリック一式を定義します。これらのメトリックは、レポートがベンダー（データ ソース）にかかわらず均一になるように正規化されます。 すべてのベンダーがメトリック ファミリ内の全メトリックに対する値を提供するとは限りません。また、すべてのメトリックが必要だとは限りません。 ベンダーが値を提供しない場合、メトリックは「null」になります。また、null メトリックに基づいたレポート ビューはすべて空になります。

メトリック ファミリは、アイテム名およびインデックスなど、ディスカバリ中にキャプチャされる属性も定義します。また、コンポーネントの一致を照合するディスカバリ ルールが定義されている場合があります。 メトリック ファミリは監視プロファイルに含めます。 監視プロファイル内のメトリック ファミリのセットは、プロファイルと関連付けられている各デバイス コレクション内のデバイスに対してどのメトリックを収集するかを決めます。

フレーム リレー PVC の例をサポートするメトリック ファミリの例を以下に示します。カスタム コンポーネントの例 (frPVC) が ComponentFacets セクションにどのように含まれているか注目してください (サンプルを示す目的で太字になっています) 。

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Auto-generated by the type catalog local manager.  -->
<DataModel namespace="http://im.ca.com/normalizer"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="IMDBCertificationFacet.xsd">
  <FacetType name="frPVCInfo"
descriptorClass="com.ca.im.core.datamodel.certs.NormalizedFacetDes
criptorImpl">
    <Documentation>Frame Relay Permanent Virtual
Circuit</Documentation>
    <FacetOf namespace="http://im.ca.com/core" name="Item" />
    <AttributeGroup name="AttributeGroup" external="true"
list="true">
      <Documentation />
      <Attribute name="Indexes" type="ObjectID[]">
         <Documentation />
         <Polled>false</Polled>
         <Baseline>false</Baseline>
         <IsDbColumn>false</IsDbColumn>
         <Variance>false</Variance>
         <StandardDeviation>false</StandardDeviation>
         <Minimum>false</Minimum>
         <Maximum>false</Maximum>
         <WriteOnPoll>false</WriteOnPoll>
         <RollupStrategy />
         <AttributeDisplayName />
         <Percentile>0</Percentile>
       </Attribute>
       <Attribute name="Names" type="String">
         <Documentation>The name of the frame relay
circuit</Documentation>
         <Polled>false</Polled>
         <Baseline>false</Baseline>
         <IsDbColumn>false</IsDbColumn>
         <Variance>false</Variance>
         <StandardDeviation>false</StandardDeviation>
         <Minimum>false</Minimum>
         <Maximum>false</Maximum>
         <WriteOnPoll>false</WriteOnPoll>
         <RollupStrategy />
         <AttributeDisplayName />
         <Percentile>0</Percentile>
        </Attribute>
        <Attribute name="Description" type="String">
```

```
          <Documentation>A description for the frame relay
circuit</Documentation>
          <Polled>false</Polled>
          <Baseline>false</Baseline>
          <IsDbColumn>false</IsDbColumn>
          <Variance>false</Variance>
          <StandardDeviation>false</StandardDeviation>
          <Minimum>false</Minimum>
          <Maximum>false</Maximum>
          <WriteOnPoll>false</WriteOnPoll>
          <RollupStrategy />
          <AttributeDisplayName />
          <Percentile>0</Percentile>
        </Attribute>
        <Attribute name="BECNIn" type="Double">
          <Documentation>Backward congestion since the virtual
circuit was created</Documentation>
          <Polled>true</Polled>
          <Baseline>false</Baseline>
          <IsDbColumn>true</IsDbColumn>
          <Variance>false</Variance>
          <StandardDeviation>false</StandardDeviation>
          <Minimum>false</Minimum>
          <Maximum>false</Maximum>
          <WriteOnPoll>false</WriteOnPoll>
          <RollupStrategy>Sum</RollupStrategy>
          <AttributeDisplayName />
          <Percentile>0</Percentile>
        </Attribute>
        <Attribute name="FECNIn" type="Double">
          <Documentation>Forward congestion since the virtual
circuit was created</Documentation>
          <Polled>true</Polled>
          <Baseline>false</Baseline>
          <IsDbColumn>true</IsDbColumn>
          <Variance>false</Variance>
          <StandardDeviation>false</StandardDeviation>
          <Minimum>false</Minimum>
          <Maximum>false</Maximum>
          <WriteOnPoll>false</WriteOnPoll>
          <RollupStrategy>Sum</RollupStrategy>
          <AttributeDisplayName />
          <Percentile>0</Percentile>
        </Attribute>
        <Attribute name="FramesIn" type="Double">
          <Documentation>Frames received since the virtual circuit
was created</Documentation>
          <Polled>true</Polled>
          <Baseline>false</Baseline>
```

```
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
          <Attribute name="FramesOut" type="Double">
            <Documentation>Frames sent since the virtual circuit was
created</Documentation>
            <Polled>true</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
          <Attribute name="BytesIn" type="Double">
            <Documentation>Bytes received since the virtual circuit was
created</Documentation>
            <Polled>true</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
          <Attribute name="BytesOut" type="Double">
            <Documentation>Bytes sent since the virtual circuit was
created</Documentation>
            <Polled>true</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
```

```
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
        </AttributeGroup>
        <Attribute name="SourceFacetTypes" cached="true" list="true"
persistent="true" type="QName">
          <Documentation />
        </Attribute>
        <DisplayName>Frame Relay PVC</DisplayName>
        <Expressions>
          <ExpressionGroup destCert="{http://im.ca.com/core}Item">
            <Expression destAttr="Name">Names</Expression>
          </ExpressionGroup>
          <ExpressionGroup
destCert="{http://im.ca.com/inventory}DeviceComponent">
            <Expression destAttr="IndexList">Indexes</Expression>
          </ExpressionGroup>
          </Expressions>
          <TableName>FR_PVC_INFO</TableName>
          <ComponentFacets>
           <Facet>{http://im.ca.com/inventory}frPVC</Facet>
          </ComponentFacets>
          <Protocol>IMDB</Protocol>
          <Normalized>true</Normalized>
         </FacetType>
       </DataModel>
```

基本プロパティ

カスタム メトリック ファミリの基本プロパティは、ユーザが作成したその他のカスタム メトリック ファミリとの区別に役立ちます。

基本のプロパティを確定する場合、以下の制限を考慮します。

- FacetType/name、DisplayName、TableName プロパティは各メトリック ファミリに対して一意である必要があります。
- Protocol タグは常に IMDB です。
- Normalized タグは常に true です。
- FacetType/descriptorClass プロパティおよびすべての DataModel と FacetOf プロパティを設定します。

FacetType/name

メトリック ファミリ名を指定します。 各メトリック ファミリには、システム内部で識別するための一意の名前が必要です。 今後同様のメトリック ファミリとのネーミング競合の可能性が最も小さいと考えられる名前を慎重に選択します。 たとえば、これらのメトリック ファミリ名が一意であることの確認に役立つ命名スキームを定義します。

注: この名前は外部には表示されません。 ユーザ インターフェースにメトリック ファミリ名を表示するには、DisplayName エレメントを使用します。

更新の可否： いいえ

可能な値： 英数字およびアンダースコア。 ドットおよびダッシュは使用できません。すべてのメトリック ファミリにわたって一意である必要があります。

DisplayName

ユーザ インターフェースに表示するメトリック ファミリ名を指定します。

更新の可否： はい

可能な値: プレーン テキスト。すべてのメトリック ファミリにわたって一意である必要があります。

更新による影響： 管理者ユーザ インターフェース内の名前の変更。

更新が有効化されるタイミング： 即時

更新を有効にするのに必要なアクション: ユーザ インターフェースをリフレッシュします。

Documentation

メトリック ファミリの外部的な説明を指定します。これらのコメントを役立てるために、私たちは、ユーザがメトリック ファミリを追加あるいは変更した理由と日時を記述することを推奨します。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： なし

TableName

メトリック ファミリが収集するメトリックを格納するために使用されるデータベース テーブル名を指定します。

更新の可否： はい

可能な値： 大文字英数字およびアンダースコア。文字から始まる必要があります。すべてのメトリック ファミリにわたって一意である必要があります。

例： PROCESS_STATS

更新による影響： ポーリング データはデータベース テーブルの新しいセットに保存されます。

重要: TableName を更新すると、古いポーリング データは失われます。古いレポート ビューは壊れています。

更新が有効化されるタイミング： 即時。新しいビューが作成されるまでに、CA Performance Center は新しい MIB ファイルをロードするため、5 分程度の遅延が生じます。

更新の有効化に必要なアクション： ビューを再作成する必要があります。

ComponentFacets

ComponentsFacets セクションは、ディスカバリ中に作成されたファセットのリストを示します。 ディスカバリは、アイテムをデバイス コンポーネントとして識別するか、またはアイテム間の階層関係を作成します。

Facet

コンポーネント ディスカバリ中にコンポーネント アイテムに付属しているファセットを指定します。

更新の可否： はい

可能な値： ファセットの QName

更新による影響： コンポーネント ファセットが CA Performance Center に同期された場合、そのコンポーネントは CA Performance Center で表示されるようになります。

更新が有効化されるタイミング： 再検出時

更新の有効化に必要なアクション： デバイスを削除して再検出します。

詳細:

階層 (P. 51)
ComponentReconciliation (P. 52)

ItemFacets

重要: ItemFacets は新しいセクションで、将来の複雑なメトリック ファミリ構造をサポートするために変更が考えられます。 使用は推奨されません。

ItemFacets は、ディスカバリ中に作成されたファセットで、アイテムをデバイスとして識別するファセットをリスト表示します。

Facet

ディスカバリ中にアイテムに付属しているファセットを指定します。

更新の可否： はい

可能な値： ファセットの QName

更新による影響： コンポーネントは指定されたファセットに対する REST サービス上で表示されるようになります。 コンポーネント ファセットが CA Performance Center に同期された場合、そのコンポーネントは CA Performance Center に表示されるようになります。

更新が有効化されるタイミング： 再検出時

更新の有効化に必要なアクション： デバイスを削除して再検出します。

例:

```
<ItemFacets>
    <Facet>{http://im.ca.com/inventory}Host</Facet>
    <Facet>{http://im.ca.com/inventory}Device</Facet>
    <Facet>{http://im.ca.com/inventory}ConsolidatedAndDiscoveredMetricFamilyHisto
    ry</Facet>
    <Facet>{http://im.ca.com/core}Syncable</Facet>
    <!-- The IPDomainID attribute will be filled in by discovery  -->
    <Facet>{http://im.ca.com/core}IPDomainMember</Facet>
</ItemFacets>
```

詳細:

ItemReconciliation (P. 52)

SourceFacetTypes 属性

SourceFacetTypes 属性はディスカバリに必要で、定義する必要があります。

以下の必要な値を使用します。

- 名前： SourceFacetTypes
- タイプ： QName
- cached： true
- persistent： true
- list： true

例： <Attribute name="SourceFacetTypes" type="QName" cached="true" persistent="true" list="true" />

更新の可否： いいえ

AttributeGroup（メトリック ファミリ）

AttributeGroup はアイテム ディスカバリ属性およびメトリック属性のコレクションです。アイテム ディスカバリ属性は、アイテム説明のように、ディスカバリ中に設定されます。 メトリック属性はポーリング中に収集されます。 以下の情報は、AttributeGroup セクションで使用されるエレメントの説明です。

AttributeGroup/list および AttributeGroup/external プロパティを true に設定します。 これらのプロパティは、各属性が外部ソースから取得される値のリストを表すことを指定します。 以下の XML エレメントをカスタマイズします。

AttributeGroup/name

属性グループ名を指定します。 「<FacetType/name>>Group」ネーミング スキームに準拠します。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： なし

Documentation

（オプション）属性グループの説明を指定します。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： なし

一般的な属性（メトリック ファミリ）

すべてのメトリック ファミリ用の一般的な属性を以下に示します。

Attribute/name

一意の内部名を指定します。 メトリックの場合、この名前がデータベース列の命名に対しても使用されます。

注: この名前は外部には表示されません。 ユーザ インターフェースに属性名を表示するには、AttributeDisplayName エレメントを使用します。

更新の可否： はい

可能な値： 英数字およびアンダースコア。

更新による影響： メトリックの場合、この属性に対する値は、更新された名前に対応する新しいデータベース列に格納されます。 このメトリックに対して（古い名前で）収集された履歴データは失われます。このメトリックに対するカスタム レポートは失敗します。

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

Attribute/type

この属性のデータ型を示します。最も使用頻度の高いデータ型は、Int、Long、Double、String、または ObjectID です。 データベースはメトリック属性を浮動小数型として格納します。 そのため、これらの属性は数値型を使用する必要があります（Double が推奨されます）。 その他のタイプはアイテム属性に使用されます。

更新の可否： はい

可能な値： Boolean、Int、Long、Double（浮動小数点）、BigInteger、String、DateTime、IPAddress、MACaddress、IPSubnet、OctetString（16 進表現）、ObjectID、ItemID、QName （修飾名）

注: タイプ名は大文字と小文字を区別しません、たとえば、「boolean」と「Boolean」は同じです。

更新による影響： メトリックの場合は、なし。すべてのメトリックは浮動小数型としてデータベースに格納されます。アイテム属性の場合は、デバイスを削除して再検出する必要があります。

更新が有効化されるタイミング： メトリックの場合は次のポーリング時。 アイテム属性の場合は再検出時。

更新の有効化に必要なアクション： メトリックの場合は、なし。 アイテム属性の場合は、デバイスを削除して再検出する。

AttributeDisplayName

オペレータと管理者のインターフェースに表示される値を指定します。

更新の可否： はい

可能な値： 英数字、スペース、アンダースコア。

更新による影響： メトリックは、［メトリック ファミリ］ UI およびカスタム レポートで AttributeDisplayName の更新を反映します。

更新が有効化されるタイミング： 即時

更新の有効化に必要なアクション： なし

Documentation

ユーザ インターフェースに属性の説明を表示します。 属性名の上にカーソルを置いた場合も、ツールヒントとして表示されます。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： 属性名の上にカーソルを置くと、更新されたドキュメントを表示します。

更新が有効化されるタイミング： 即時

更新の有効化に必要なアクション： なし

Polled

属性がポーリングされるかどうかを示します。 false に設定すると、ディスカバリ中にのみアクセスされます。

更新の可否： はい

可能な値： true、false

更新による影響： false に設定した場合、他のポーリングされる属性/メトリックがその OID を式で使用していなければ、この属性/メトリックに対応する OID はポーリングされません。 true に設定した場合、この属性/メトリックに対応する OID はポーリングされます。

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

IsDbColumn

データベース テーブルに値を格納します。 IsDbColumn はメトリック属性に使用されます。 Polled を true に設定した場合、IsDbColumn 値を true に設定してください。

更新の可否： はい

可能な値： true、false

更新による影響： false に設定した場合、この属性/メトリックのデータはデータベースに格納されません。 true に設定した場合、この属性/メトリックのデータはデータベースに格納されます。

更新が有効化されるタイミング： 即時

更新の有効化に必要なアクション： なし

ディスカバリ属性

多くの属性については、ディスカバリ中に取得された値のみがデータベースに格納されます。 ベースラインの評価など、それ以上のポーリングや処理は実行されません。

Indexes および Names 属性はすべてのメトリック ファミリに対して存在する必要があります。 Descriptions 属性はオプションです。

```
<Attribute name="Indexes" type="ObjectID[]"  />
<Attribute name="Names" type="String"  />
<Attribute name="Descriptions" type="String"  />
```

階層をサポートするメトリック ファミリにはこれらの属性を含む必要があります。

```
<Attribute name="ItemUniqueIDs" type="String" />
<Attribute name="ParentUniqueIDs" type="String" />
```

ポーリング属性およびベースライン属性

以下の情報は、ポーリングされた属性エレメントとベースライン属性エレメントを記述します。

Baseline

この属性の平均を計算するかどうかを示します。True に設定した場合、対応する BaselineList 定義を定義する必要があります。

注: ベースライン属性では、StandardDeviation 属性が true に設定されている必要があります。

更新の可否： はい

可能な値： true、false

更新による影響： true の場合、ベースライン値が計算されます。

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

Maximum

ロールアップ中にこの属性の最大を計算するかどうかを示します。データベース テーブルに「max_」列を作成します。 RollupStrategy を定義する場合、この属性も定義する必要があります。

更新の可否： はい

可能な値： true、false

更新効果: True は「最大」の計算および「最大」に対するレポートフィールドを提供します。

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

Minimum

ロールアップ中にこの属性の最小を計算するかどうかを示します。データベース テーブルに「min_」列を作成します。 RollupStrategy を定義する場合、この属性も定義する必要があります。

更新の可否： はい

可能な値： true、false

更新による影響： True は「最小」の計算および「最小」に対するレポート フィールドを提供します。

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

StandardDeviation

ロールアップ中にこの属性の標準偏差を計算するかどうかを示します。データベース テーブルに「std_」列を作成します。 RollupStrategy を定義する場合、この属性も定義する必要があります。

更新の可否： はい

可能な値： true、false

更新による影響： True は「標準偏差」の計算および「標準偏差」に対するレポートフィールドを提供します。

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

DeviationFromBaseline

ベースライン属性を true に設定することが必要です。 ベースライン データを使用して計算される 2 つの追加レポート フィールド「平均ベースライン」および「パーセント偏差」を提供します。 これらのフィールドはカスタム ビューの構築に使用できません。データベース テーブルに対する変更はありません。

更新の可否： はい

可能な値： true、false

更新による影響： True の場合、内部レポート開発に対して「平均ベースライン」および「パーセント偏差」フィールドを提供します。

更新が有効化されるタイミング： 即時

更新の有効化に必要なアクション： なし

Percentile

ロールアップ中にこの属性の 95 パーセンタイルを計算するかどうかを示します。 データベース テーブルに「pct_」列を作成します。 RollupStrategy を定義する場合、この属性も定義する必要があります。

更新の可否： はい

可能な値： 0 （ゼロ）、95

更新による影響： 95 の場合、「95 パーセンタイル」の計算およびレポート フィールドを提供します。ゼロの場合、計算は実行されず、レポート フィールドは使用可能ではありません。

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

RollupStrategy

個別にポーリングされた値のロールアップ中にすべてのサイクルで実行する操作を指定します。Polled および IsDbColumn が true に設定されている場合、このエレメントは必要です。

更新の可否： はい

可能な値： Sum（カウンタの合計）、Avg （ゲージの平均）

更新による影響： 指定された戦略はロールアップ計算の実行のために使用されます。

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

Rate

AVG （メトリック値 / 時間）として計算される追加のレポート フィールド「平均レート」を提供します。データベース テーブルに対する変更はありません。

注: Rate はレポートに使用可能ですが、プロファイル イベント ルールを監視する場合には使用できません。

更新の可否： はい

可能な値： true、false

更新による影響： レポートに対して「平均レート」フィールドを提供します。

更新が有効化されるタイミング： 即時

更新の有効化に必要なアクション： なし

Units

レポートで使用される単位ラベルの名前を指定します。表示される実際のラベルは、レポートの言語設定に従って翻訳されます。

更新の可否： はい

可能な値： Percent、Packets、PacketsPerSecond、DiscardedPackets、DiscardedPacketsPerSecond、ErroredPackets、ErroredPacketsPerSecond、Bits、BitsPerSecond、Bytes、BytesPerSecond、Microseconds、Milliseconds、UnixTime

更新による影響： 指定された単位ラベルがレポートで表示されます。

更新が有効化されるタイミング： 即時

更新の有効化に必要なアクション： なし

例: ポーリングされた属性エレメントおよびベースライン属性エレメントの使用

```
<Attribute name="Utilization" type="double">
    <AttributeDisplayName>Utilization</AttributeDisplayName>
    <AttributeAbbreviation>Utilization</AttributeAbbreviation>
    <IsDbColumn>true</IsDbColumn>
    <Baseline>true</Baseline>
    <Minimum>true</Minimum>
    <Maximum>true</Maximum>
    <RollupStrategy>Avg</RollupStrategy>
    <StandardDeviation>true</StandardDeviation>
    <DeviationFromBaseline>true</DeviationFromBaseline>
    <Percentile>95</Percentile>
    <Polled>true</Polled>
    <Units>Percent</Units>
</Attribute>
```

BaselineDefinitions

BaselineDefinitions セクションには、このメトリック ファミリで計算するベースライン定義の説明が含まれます。 ベースライン定義は、Baseline プロパティが true に設定されている AttributeGroup セクションの各メトリックに対して指定されている必要があります。

2 種類のベースライン［時間単位］（必須）および［日単位］（オプション）を定義できます。 時間単位のベースラインはイベント処理およびレポートでのベースラインの表示の両方に使用されます。 日単位のベースラインは、1 か月以上のタイムフレームを持ったレポートでのベースラインの表示に対して使用されます。

以下の情報は、使用されたベースライン属性エレメントの説明です。

Name

メトリック用のベースライン定義のタイプを指定します。 タイプは時間単位または日単位です。

更新の可否： いいえ

可能な値： HourlyBaseline、DailyBaseline

ID

使用されなくなった値を指定します。 ただし、フィールドを正の整数として指定する必要があり、このメトリック ファミリ内のすべての時間単位および日単位ベースライン定義にわたって一意である必要があります。

更新の可否： はい

可能な値：任意の一意および正の整数。

更新による影響： なし

PerformanceMetric

ベースラインが計算されるメトリックの名前（大文字と小文字を区別する）を指定します。メトリック属性に対して Polled および Baseline プロパティを true に設定します。

更新の可否： はい

可能な値： 有効なメトリック名（大文字と小文字を区別する）。

更新による影響：ベースライン計算がメトリックに対して実行されます。

更新が有効化されるタイミング：次のベースライン計算（時間単位か日単位のいずれか）。

更新の有効化に必要なアクション： なし

Period

ベースライン計算のタイプを指定します（時間単位または日単位のベースライン）。 HourlyBaseline 名に対して「1 Day」、DailyBaseline 名に対して「1 Hour」の値を指定します。

更新の可否： はい

可能な値： 1 Hour、1 Day

更新による影響：ベースライン計算が時間単位または日単位でされます。

更新が有効化されるタイミング：次のベースライン計算（時間単位か日単位のいずれか）

更新の有効化に必要なアクション： なし

Window

使用されなくなった値を指定します。 ただし、フィールドは、時間単位のベースラインについては「30 日」、日単位ベースラインについては「90 日」として指定する必要があります。

更新の可否： いいえ

可能な値： 30 Days、90 Days

StartDate、EndDate、DaysOfWeek

他の使用されない値を指定しますが、それらを 0 （ゼロ）として指定する必要があります。

更新の可否： いいえ

可能な値： 0

Expressions

Expressions セクションは、コンポーネント ディスカバリに使用される［ExpressionGroup］タグから構成されます。 コンポーネント ディスカバリ中に、コンポーネント アイテム プロパティ（IndexList、Name、Description など）に対する値が計算されます。 メトリック ファミリ式をサポートするベンダー認定式は、この計算に使用されます。

注: メトリック ファミリとベンダー認定の［ExpressionGroup］タグを混同しないよう注意してください。

以下の DestCert URI に対して ExpressionGroup タグが存在する必要があります。

DestCert	DestAttr
{http://im.ca.com/core}Item	Name
{http://im.ca.com/core}Item	Description
{http://im.ca.com/inventory} DeviceComponent	IndexList

ExpressionGroup/name

（オプション）式グループ名を指定します。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： なし

ExpressionGroup/destCert

入力する destAttrs が含まれるコンポーネント ファセットを指定します。 ファセット名は通常、Item および DeviceComponent 以外は ComponentFacets セクションから取得されます。

更新の可否： はい

可能な値: ComponentFacets で定義されているファセット、または Item、DeviceComponent ファセット。

更新による影響： 許容可能な式 destAttr を変更します

更新が有効化されるタイミング： コンポーネント再検出時

更新の有効化に必要なアクション： なし

ExpressionGroup/Expression

コンポーネント ファセット属性に対する式を指定します。

更新の可否： はい

可能な値： 任意の有効なメトリック

更新による影響： 許容可能な式 destAttr を変更します

更新が有効化されるタイミング： コンポーネント再検出時

更新の有効化に必要なアクション： なし

ExpressionGroup/Expression/destAttr

コンポーネント ファセットの属性名を指定します。

更新の可否： はい

可能な値： そのコンポーネント ファセットからの任意の有効な属性。

更新による影響： 属性名を変更します

更新が有効化されるタイミング： コンポーネント再検出時

更新の有効化に必要なアクション： なし

階層

階層（親子関係）は、異なるメトリック ファミリのアイテム間で定義できます（たとえば Interface、CBQoS Classmap）。 メトリック ファミリ定義で、以下の要素持つ子メトリック ファミリに階層を指定する必要があります。

- ComponentFacets 内の階層 QName
- Hierarchy ExpressionGroup 内の ItemUniqueID および ParentUniqueID destAttr の値
- AttributeGroup 内の ItemUniqueIDs および ParentUniqueIDs の属性

サポートする式は、ベンダー認定で定義されています (P. 84)。

以下の DestCert URI 用の Hierarchy ExpressionGroup タグが存在する必要があります。

DestCert	DestAttr
{http://im.ca.com/inventory}Hierarchy	ItemUniqueID
{http://im.ca.com/inventory}Hierarchy	ParentUniqueID

例:

```
<ComponentFacets>
    <Facet>{http://im.ca.com/inventory}QoSClassMap</Facet>
    <Facet>{http://im.ca.com/inventory}Hierarchy</Facet>
</ComponentFacets>

<ExpressionGroup name="Hierarchy"
destCert="{http://im.ca.com/inventory}Hierarchy">
    <Expression destAttr="ItemUniqueID">ItemUniqueIDs</Expression>
    <Expression destAttr="ParentUniqueID">ParentUniqueIDs</Expression>
</ExpressionGroup>

<AttributeGroup name="QosCosGroup" list="true" external="true">
      <Attribute name="ItemUniqueIDs" type="String" />
      <Attribute name="ParentUniqueIDs" type="String" />
      ...
</AttributeGroup>
```

詳細:

ComponentFacets (P. 37)

ComponentReconciliation

以下の情報は、コンポーネント ディスカバリで使用されるコンポーネント照合ロジックを定義します。 まず、この情報は、システムがすでにこのコンポーネントを検出したかどうかを判断します。 次に、既存のコンポーネントを更新するべきか、新しいものを作成するべきかを判断します。

```
<ComponentReconciliation>
  <MatchAlgorithms>
    <ExactMatch>
      <MatchAttribute name="{http://im.ca.com/inventory}Port.Type"/>
      <MatchAttribute name="{http://im.ca.com/core}Item.Description"/>
    </ExactMatch>
    <BestOfMatch leastMatchCount="3">
      <MatchAttribute name="{http://im.ca.com/inventory}Port.Type" required="true"/>
      <MatchAttribute name="{http://im.ca.com/inventory}Port.Alias"/>
      <MatchAttribute name="{http://im.ca.com/core}Item.Description"/>
      <MatchAttribute name="{http://im.ca.com/inventory}Port.MACAddress"/>
      <MatchAttribute name="{http://im.ca.com/inventory}DeviceComponent.IndexList"/>
    </BestOfMatch>
  </MatchAlgorithms>
</ComponentReconciliation>
```

1 つのメトリック ファミリに対して順序付けされた複数のアルゴリズムが存在する可能性があります。 メトリック ファミリが照合アルゴリズムを定義しない場合、一致属性 Item.Name を持ったデフォルトが適用されます。

詳細:

ComponentFacets (P. 37)
MatchAlgorithms (P. 54)

ItemReconciliation

重要: ItemReconciliation は新しいセクションで、今後複雑なメトリック ファミリ構造をサポートするために変更が考えられます。 使用は推奨されません。

以下の情報は、アイテム ディスカバリで使用されるアイテム照合ロジックを定義します。 このロジックは、システムがすでにアイテムを検出したかどうかを判断します。 この判断に基づいて、既存のアイテムが更新されるか、または新しいアイテムが作成されます。 アイテムの照合はコンポーネントの照合に類似しています。 ただし、アイテムの照合は、仮想ホストなど、コンポーネントでないアイテムに使用されます。ItemFacets は、すべての新しいアイテムまたは一致するアイテムに追加されます（ファセットが存在しない場合）。

例:

```
<ItemReconciliation>
    <SourceAgentScopedReconciliation>
        <MatchAlgorithms>
           <ExactMatch>
              <MatchAttribute
        name="{http://im.ca.com/inventory}SourceAgentInfo.SourceAgentIndexes" />
           </ExactMatch>
        </MatchAlgorithms>
    </SourceAgentScopedReconciliation>
    <GlobalScopedReconciliation matchDevices="true" />
</ItemReconciliation>
```

SourceAgentScopedReconciliation

アイテムを照合するために使用される一致アルゴリズムを定義します。

更新の可否： はい

更新による影響： アイテム照合ロジックを変更します。

更新が有効化されるタイミング： 再検出時

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

GlobalScopedReconciliation

ソース エージェントに対してアイテムを照合できなかった場合に使用される一致アルゴリズムを定義します。 GlobalScopedReconciliation アルゴリズムは、他のエージェント用に作成されたが、潜在的な新しいアイテムに一致するアイテムを見つけるために使用されます。 matchDevices プロパティが true に設定されている場合、システム デフォルト（XML には表示されない標準仕様）の ComponentReconciliationInfo 一致アルゴリズムが使用されます。一致アルゴリズムはデバイス プライマリ IP アドレスおよびホスト名に基づきます。

更新の可否： はい

更新による影響： アイテム照合ロジックを変更します。

更新が有効化されるタイミング： 再検出時

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

詳細:

ItemFacets (P. 38)
MatchAlgorithms (P. 54)

MatchAlgorithms

コンポーネント照合およびアイテム照合は一致アルゴリズムを使用します。 2 つの一致アルゴリズムがサポートされています。

- ExactMatch -- アイテムが新しいデータに一致しているとみなすには、すべての指定した属性が一致する必要があります。
- BestOfMatch -- leastMatchCount 値を使用して、一致する属性の最小数を指定する必要があります。また、属性にはそれぞれ必須のキー プロパティがあります。 必須プロパティが true に設定されている場合、一致と見なすには、その属性が一致している必要があります。

複数のアルゴリズムがメトリック ファミリに対して提供されている場合、アルゴリズムには照合の優先順位があります。 アルゴリズムの順序が優先順位を決定します。 一番上のアルゴリズムの優先順位が最も高くなります。 一番下のアルゴリズムの優先順位が最も低くなります。

各アルゴリズムには少なくとも 1 つの一致属性が必要です。データが複数のアイテムに同じアルゴリズムで一致する場合、一致属性が最も多いアイテムが選ばれます。 複数の一致アイテムに同数の一致属性がある場合、これらのアイテムから任意に選ばれます。

例: 照合の方法

2 つの一致アルゴリズム alg1 および alg2 がメトリック ファミリに対して提供されます。 alg1 は alg2 より優先順位が高くなります。 メトリック ファミリに 3 つの既存のコンポーネント アイテム 1、2、3 があります。メトリック ファミリの再検出により、3 つのエントリ A、B、C が見つかりました。 ここで、2 つのアルゴリズムを適用し、どのエントリが新しいか、変更されているか、未変更であるかを判断します。

```
   照合メタデータ                          新しいデータ          既存コンポーネント
<ComponentReconciliation>                          A                    1
  <MatchAlgorithms>                                B                    2
    <MatchAlgorithm1>                              C                    3
      <MatchAttribute name="attr1"/>
      <MatchAttribute name="attr2"/>
    </MatchAlgorithm1>

    <MatchAlgorithm2>
      <MatchAttribute name="attr1"/>
      <MatchAttribute name="attr3"/>
      <MatchAttribute name="attr4"/>
    </MatchAlgorithm2>

  </MatchAlgorithms>
</ComponentReconciliation>
```

<MatchAlgorithm1> および <MatchAlgorithm2> は、<ExactMatch> または <BestOfMatch> のいずれかになる可能性があります。 2 つの一致アルゴリズムの順序は、MatchAlgorithm1 が MatchAlgorithm2 より優先順位が高いことを示します。

ケース 1: 一意の 1 対 1 の 一致

エントリ A がアイテム 1 に一致し、アイテム 1 にはその他の一致がありません。

```
A -----> 1
```

この例は最も単純なケースです。 この一致は一意です。したがって、alg1 または alg2 にも一致したとしても関係ありません。エントリ A がアイテム 1 に一致します。

良好な一致アルゴリズムでは、より多くの一意の一致が生じます。

ケース 2: 1 つのエントリに複数の一致があります

エントリ A が alg1 によりアイテム 1 に一致し、alg2 によりアイテム 2 に一致します。

```
          ---> 1 （alg1） （1 を採用）
    /
A                    alg1 の方が優先順位が高いため、アイテム 1 に一致となります。
  ¥
          ---> 2 （alg2）
```

ケース 3: 複数のエントリが異なるアルゴリズムで同じアイテムに一致します

エントリ A が alg1 によりアイテム 1 に一致し、エントリ B も alg2 によりアイテム 1 に一致します。

```
A ------> 1 （alg1） （A を採用）
B ------> 1 （alg2）
```

alg1 の方が優先順位が高いため、エントリ A が採用されます。

ケース 4: 複数のエントリが、同じアルゴリズムで同じアイテムに一致し、一致した属性の数が異なります

A も B も alg1 により 1 に 一致します。

```
A ------> 1 （alg1、一致した属性の数： 2） （A を採用）
B ------> 1 （alg1、一致した属性の数： 1）
```

A の方が、より多くの一致した属性があるため、A が採用されます。

一致属性の数が同じである場合、ランダムに選ばれ、警告が出されます。

ケース 5: 混合一致 1

```
alg1
 A ----------------------------------> 1
   /  alg2（一致属性数： 3）
 B
   ¥  alg2（一致属性数： 2）
          -------------------------------> 2
```

A は優先順位の高いアルゴリズムで一致するため、1 に一致します。

1 が A に一致したので、B は 2 に一致します。

ケース 6: 混合一致 2

```
        ---------> 3
   /  alg1
A                  ==> alg1 の方が優先順位が高いため、A は 3 に一致します
   ¥  alg2
        ---------> 1
   /  alg2
B                  ==> alg1 の方が優先順位が高いため、B は 2 に一致します
   ¥  alg1
        ---------> 2
   /  alg2
C                  ==> 2 が B に一致し、3 が A に一致したため、C の一致はありません
   ¥  alg2
        ---------> 3
```

エントリ C は新しいコンポーネントとして扱われます。 1 は不一致アイテムと見なされます。

ケース 1 （一意の一致）の一致が多いほど、良好な一致アルゴリズムといえます。

更新の可否： はい

更新による影響： コンポーネント照合ロジックを変更します。

更新が有効化されるタイミング： 再検出時

更新の有効化に必要なアクション： メトリック ファミリを更新、またはベンダー認定優先度を変更します。

詳細:

ComponentReconciliation (P. 52)
ItemReconciliation (P. 52)

ReconfigDetectionAttr

このエレメントは、変更検出に使用されるメトリック ファミリ属性を定義します。 監視プロファイルに対して変更検出を有効にできます。 Data Aggregator は、再検出をすべて実行するのではなく、ターゲット デバイスが変更されたかを確認するためだけに属性をポーリングします。 この機能はパフォーマンスの向上に役立ち、ネットワーク トラフィックを軽減させます。

更新の可否： はい

可能な値： メトリック ファミリ属性のフル ネーム。 指定されたメトリック ファミリ属性フラグでは、cached、persistent、external フラグが true に設定されている必要があります。

更新による影響： コンポーネント再設定検出の変更。

更新が有効化されるタイミング： 再検出後。

更新の有効化に必要なアクション： メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

カスタム メトリック ファミリのサポートされていないプロパティ

以下のプロパティは、カスタム メトリック ファミリに対してサポートされていません。

- Variance
- RollupExpression

重要: エクスポートされたファクトリ メトリック ファミリをテンプレートとして使用する場合は、サポートされていないエレメントをすべて削除するか、それらをカスタム メトリック ファミリ内で false に設定します。

カスタム メトリック ファミリのインポート

カスタム メトリック ファミリを有効にするには、Data Aggregator にインポートします。

CA Performance Center は、IMDBCertificationFacet.xsd スキーマに直接基づくメトリック ファミリのインポートをサポートするのユーザ インターフェースをまだ提供していません。 そのため、このタスクを実行するために Data Aggregator が提示する Web サービスを活用します。

REST クライアントを使用します。 RESTClient を CA Performance Management と使用することをお勧めします。インストールしていない場合、http://code.google.com/p/rest-client/ から RESTClient GUI JAR ファイルを取得できます。

RESTClient を使用する場合、以下の情報を考慮してください。

- JAR ファイルをダブルクリックして起動します。
- POST XML を実行する場合、Charset が UTF-8 に設定されていることを確認します。

  この設定を表示して確認するには、［Edit Content-Type & Charset］ボタンをクリックします。
- また以下のように Response Body をオート インデントできます。
  1. ［Tools］メニューの［Options］をクリックします。
  2. ［Etc.］タブを選択します。
  3. ［Auto-indent Response Body］を選択し、［OK］をクリックします。

次の手順に従ってください:

1. http://*da-hostname*:8581/typecatalog/metricfamilies を URL として入力します。
2. メソッドとして POST を選択します。
3. Body 設定内の［Body Content-type］として「application/xml」を選択します。

   重要: Content-type の設定に失敗すると 404 エラーとなります。
4. ［Body］タブにカスタム メトリック ファミリ XML をコピーして貼り付けます。
5. メソッドを実行します。

   カスタム メトリック ファミリがインポートされます。 エラーが発生しない場合、［HTTP Response］セクション内の［Status］フィールドには以下が表示されます。

   ```
   HTTP/1.1 200 OK
   ```

   注: 他のいずれのリターン コードも、カスタム メトリック ファミリを更新する間にエラーが発生したことを示します。 エラーを修正し、別の POST を実行することでメトリック ファミリの更新を再試行します。

   重要: データの損失を回避するには、ベンダー認定、メトリック ファミリ、またはコンポーネントを作成または更新するたびに、展開ディレクトリを必ずバックアップします。

カスタム メトリック ファミリ結果の確認

カスタム メトリック ファミリ XML をインポートした後、結果を確認します。 この例では、フレーム リレー メトリック ファミリが正常にインポートされたことを確認します。

次の手順に従ってください:

1. REST クライアントに以下の URL を入力します。

   `http://da_hostname:8581/typecatalog/metricfamilies/name`

   name

   使用するカスタム メトリック ファミリの名前です。 この例では FrameRelay です。

   カスタムのフレーム リレー メトリック ファミリ XML がページに表示されます。

また、Data Aggregator データ ソースに対して、[監視設定] メニューの [メトリック ファミリ] タブでカスタム メトリック ファミリがリスト表示されていることを確認できます。

追加メトリックのサポート

追加のメトリックを既存のカスタム メトリック ファミリに追加したい場合があります。 前述の例でいえば、［ビット数 （イン）］および［ビット数 （アウト）］のメトリックを追加します。

更新された frPVCInfo カスタム メトリック ファミリの例を以下に示します。 この手順で加えた変更は太字で示されます。

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Auto-generated by the type catalog local manager.-->
<DataModel namespace="http://im.ca.com/normalizer"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="IMDBCertificationFacet.xsd">
  <FacetType name="frPVCInfo"
descriptorClass="com.ca.im.core.datamodel.certs.NormalizedFacetDes
criptorImpl">
   <Documentation>Frame Relay Permanent Virtual
Circuit</Documentation>
   <FacetOf namespace="http://im.ca.com/core" name="Item" />
   <AttributeGroup name="AttributeGroup" external="true"
list="true">
     <Documentation />
     <Attribute name="Indexes" type="ObjectID[]">
       <Documentation />
       <Polled>false</Polled>
       <Baseline>false</Baseline>
       <IsDbColumn>false</IsDbColumn>
       <Variance>false</Variance>
       <StandardDeviation>false</StandardDeviation>
       <Minimum>false</Minimum>
       <Maximum>false</Maximum>
       <WriteOnPoll>false</WriteOnPoll>
       <RollupStrategy />
       <AttributeDisplayName />
       <Percentile>0</Percentile>
     </Attribute>
     <Attribute name="Names" type="String">
        <Documentation>The name of the frame relay
circuit</Documentation>
        <Polled>false</Polled>
        <Baseline>false</Baseline>
        <IsDbColumn>false</IsDbColumn>
        <Variance>false</Variance>
        <StandardDeviation>false</StandardDeviation>
        <Minimum>false</Minimum>
        <Maximum>false</Maximum>
        <WriteOnPoll>false</WriteOnPoll>
        <RollupStrategy />
        <AttributeDisplayName />
        <Percentile>0</Percentile>
      </Attribute>
      <Attribute name="Description" type="String">
        <Documentation>A description for the frame relay
circuit</Documentation>
```

```
            <Polled>false</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>false</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy />
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
          <Attribute name="BECNIn" type="Double">
             <Documentation>Backward congestion since the virtual
circuit was created</Documentation>
            <Polled>true</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
          <Attribute name="FECNIn" type="Double">
             <Documentation>Forward congestion since the virtual
circuit was created</Documentation>
            <Polled>true</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
          <Attribute name="FramesIn" type="Double">
             <Documentation>Frames received since the virtual circuit
was created</Documentation>
            <Polled>true</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
```

```
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
          <Attribute name="FramesOut" type="Double">
            <Documentation>Frames sent since the virtual circuit was
created</Documentation>
            <Polled>true</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
          <Attribute name="BytesIn" type="Double">
            <Documentation>Bytes received since the virtual circuit
was created</Documentation>
            <Polled>true</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
          <Attribute name="BytesOut" type="Double">
            <Documentation>Bytes sent since the virtual circuit was
created</Documentation>
            <Polled>true</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
```

```
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
          <Attribute name="BitsIn" type="Double">
             <Documentation>Bits received since the virtual circuit
was created</Documentation>
            <Polled>true</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
          <Attribute name="BitsOut" type="Double">
            <Documentation>Bits sent since the virtual circuit was
created</Documentation>
            <Polled>true</Polled>
            <Baseline>false</Baseline>
            <IsDbColumn>true</IsDbColumn>
            <Variance>false</Variance>
            <StandardDeviation>false</StandardDeviation>
            <Minimum>false</Minimum>
            <Maximum>false</Maximum>
            <WriteOnPoll>false</WriteOnPoll>
            <RollupStrategy>Sum</RollupStrategy>
            <AttributeDisplayName />
            <Percentile>0</Percentile>
          </Attribute>
        </AttributeGroup>
        <Attribute name="SourceFacetTypes" cached="true"
list="true" persistent="true" type="QName">
          <Documentation />
        </Attribute>
        <DisplayName>Frame Relay PVC</DisplayName>
        <Expressions>
          <ExpressionGroup destCert="{http://im.ca.com/core}Item">
             <Expression destAttr="Name">Names</Expression>
          </ExpressionGroup>
          <ExpressionGroup
destCert="{http://im.ca.com/inventory}DeviceComponent">
            <Expression destAttr="IndexList">Indexes</Expression>
          </ExpressionGroup>
        </Expressions>
```

```
        <TableName>FR_PVC_INFO</TableName>
        <ComponentFacets>
          <Facet>{http://im.ca.com/inventory}frPVC</Facet>
        </ComponentFacets>
        <Protocol>IMDB</Protocol>
        <Normalized>true</Normalized>
      </FacetType>
</DataModel>
```

これらの変更は、カスタム メトリック ファミリをインポートした後に行われたとします。そのため、変更後にメトリック ファミリを更新します。メトリック ファミリを更新した後、メトリック ファミリが関連付けられているベンダー認定も更新する必要があります。

詳細:

カスタム ベンダー認定の更新 (P. 101)
カスタム メトリック ファミリの更新 (P. 65)

カスタム メトリック ファミリの更新

既存のカスタム メトリック ファミリは更新できます。 この例では、メトリックをカスタム メトリック ファミリ frPVCInfo に追加していました。そこで、メトリック ファミリを更新して変更を有効にします。

注: カスタム メトリック ファミリを更新する場合のタグまたは属性を更新する影響の詳細については、「カスタム メトリック ファミリ XML の詳細」を確認してください。 具体的には、特定の属性説明を確認します。

次の手順に従ってください:

1. URL フィールドに以下のアドレスを入力します。

   `http://da_hostname:8581/typecatalog/metricfamilies/name`

   name

   更新するカスタム メトリック ファミリの名前です。 この例では、メトリック ファミリの名前は frPVCInfo です。

2. メトリック ファミリがその URL に存在することを確認するには、［Method］タブで GET を選択します。

   注: メトリック ファミリが存在しない場合、カスタム メトリック ファミリをインポート (P. 58)できます。

3. メトリック ファミリが URL に存在することを確認したら、［Method］タブで PUT を選択します。

4. 更新したカスタム メトリック ファミリ XML を［Body］タブ（［編集］フィールド）にコピーして貼り付け、Content-type を application/xml に設定します。

   重要: Content-type の設定に失敗すると 404 エラーとなります。

5. ［URL］フィールドの隣の［Go］ボタンをクリックします。

   カスタム メトリック ファミリが更新されます。 エラーが発生しない場合、［HTTP Response］セクションの［Status］フィールドには以下のメッセージが表示されます。

   ```
   HTTP/1.1 200 OK
   ```

   その他のリターン コードは、カスタム メトリック ファミリを更新する間にエラーが発生したことを示します。 エラーを修正し、別の PUT を実行することによりメトリック ファミリの更新を再試行します。

   重要: データの損失を回避するには、ベンダー認定、メトリック ファミリ、またはコンポーネントを作成または更新するたびに、展開ディレクトリを必ずバックアップします。

6. 次に、このメトリック ファミリが関連付けられているカスタム ベンダー認定を更新します (P. 101)。

詳細:

メトリック ファミリ XML 構造について (P. 31)
追加メトリックのサポート (P. 60)

第 5 章: カスタム ベンダー認定の作成

新規ベンダー認定が問題なく作成されたら、指定したメトリック ファミリについて新しいベンダー認定が、優先度リストの最後に自動的に追加されます。 優先度リストを変更し、必要に応じてベンダー認定を高優先度に移動させることができます。 指定されたメトリック ファミリに対してデバイス サポートを検出する場合、ベンダー認定が使用されます。

注: Data Aggregator には、カスタムのベンダー認定とメトリック ファミリを作成するための基本的な方法と高度な方法があります。 基本的方法はより単純なプロセスであり、ユーザ インターフェースを使用して、サポートされた既存の技術（メトリック ファミリ）のベンダー サポートを追加することから構成されます。 この方法は、多くのユーザの要件を満たします。 一方、高度な方法は、ファクトリ認定の形式に基づいており、完全な機能セットを利用できます。 このガイドでは、高度な認定方法について説明します。 基本的な認定方法の詳細については、「Data Aggregator 自己認定ガイド」を参照してください。

このセクションには、以下のトピックが含まれています。

ベンダー認定 XML テンプレートの作成

適切な REST クライアントを使用し、カスタム ベンダー認定の作成にテンプレートとして使用できる XML ファイルのサンプルを作成します。

まず最初に、既存のベンダー認定のリストを取得します。 その後、必要とするベンダー認定がすでにサポートされているかどうかを確認できます。

次の手順に従ってください:

1. Data Aggregator サーバに接続されている REST クライアントをセットアップします。
2. REST クライアントで、Data Aggregator Web サービス API に対して以下の URL を入力します。

   `http://da_hostname:8581/typecatalog/certifications/snmp`

   すべての使用可能なベンダー認定のリストが表示されます。
3. 必要とするベンダー認定がすでにサポートされているかどうかを確認します。

必要とするベンダー認定がまだサポートされていない場合、作成するカスタム ベンダー認定に類似しているベンダー認定を参照してエクスポートします。

次の手順に従ってください:

1. 必要とするメトリック ファミリに類似している特定のベンダー認定を取得するために以下の URL を入力します。

   `http://da_hostname:8581/typecatalog/certifications/snmp/name`

   name

   CiscoCPUMibRev など、ベンダー認定の名前です。
2. ［Method］タブで GET を選択します。
3. メソッドを実行します。

   返される XML にはベンダー認定情報が含まれます。

   カスタム ベンダー認定を作成するためのテンプレートとしてこの XML を使用できます。

4. ベンダー認定 XML をテキスト ファイルにコピーし、必要に応じて変更します。XML 構造の例については、「ベンダー認定 XML 構造について (P. 69)」を参照してください。

ベンダー認定 XML 構造について

ベンダー認定は、ベンダーおよびデバイス特異的データをパフォーマンス メトリックおよびメトリック ファミリに定義されている設定にマップします。 さまざまなソースから、「正規化された」メトリック ファミリの値へこのデータをマッピングすると、Data Aggregator がデバイス ベンダーに関係なく、このデータを同じようにレポートするうえで有用です。

フレーム リレー PVC の例をサポートするカスタム ベンダー認定 XML の例を以下に示します。 サンプルのカスタム メトリック ファミリ（frPVCInfo）が［ExpressionGroup］セクションにどのように含まれているかに注目してください（例を示す目的で太字になっています）。

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Auto-generated by the type catalog local manager.-->
<DataModel namespace="http://im.ca.com/certifications/snmp"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="SNMPCertificationFacet.xsd">
  <FacetType name="frPVCInfoCustom"
descriptorClass="com.ca.im.core.datamodel.certs.CertificationFacet
DescriptorImpl">
    <Documentation>Frame Relay PVC Vendor
Certification</Documentation>
    <FacetOf namespace="http://im.ca.com/core" name="Item" />
    <AttributeGroup name="AttributeGroup" external="true"
list="true">
      <Documentation />
      <Attribute name="INDEX" type="ObjectID">
      <Documentation />
      <Source>1.3.6.1.2.1.10.32.2.1.4</Source>
      <IsIndex>true</IsIndex>
      <IsKey>false</IsKey>
      <NeedsDelta>false</NeedsDelta>
    </Attribute>
    <Attribute name="frCircuitReceivedBECNs" type="Long">
      <Documentation />
      <Source>1.3.6.1.2.1.10.32.2.1.5</Source>
      <IsIndex>false</IsIndex>
      <IsKey>true</IsKey>
      <NeedsDelta>true</NeedsDelta>
    </Attribute>
    <Attribute name="frCircuitSentFrames" type="Long">
      <Documentation />
      <Source>1.3.6.1.2.1.10.32.2.1.6</Source>
      <IsIndex>false</IsIndex>
      <IsKey>true</IsKey>
      <NeedsDelta>true</NeedsDelta>
    </Attribute>
    <Attribute name="frCircuitSentOctets" type="Long">
      <Documentation />
      <Source>1.3.6.1.2.1.10.32.2.1.6</Source>
      <IsIndex>false</IsIndex>
      <IsKey>true</IsKey>
      <NeedsDelta>true</NeedsDelta>
    </Attribute>
    <Attribute name="frCircuitReceivedFrames" type="Long">
      <Documentation />
```

```
      <Source>1.3.6.1.2.1.10.32.2.1.8</Source>
      <IsIndex>false</IsIndex>
      <IsKey>true</IsKey>
      <NeedsDelta>true</NeedsDelta>
    </Attribute>
    <Attribute name="frCircuitReceivedOctets" type="Long">
      <Documentation />
      <Source>1.3.6.1.2.1.10.32.2.1.9</Source>
      <IsIndex>false</IsIndex>
      <IsKey>true</IsKey>
      <NeedsDelta>true</NeedsDelta>
    </Attribute>
  </AttributeGroup>
  <Protocol>SNMP</Protocol>
  <DisplayName>Frame Relay PVC Certification</DisplayName>
  <Expressions>
    <ExpressionGroup
destCert="{http://im.ca.com/normalizer}frPVCInfo"
name="frPVCInfoDS">
      <Expression destAttr="Indexes">INDEX</Expression>
      <Expression destAttr="Names">"Frame Relay " +
INDEX</Expression>
      <Expression
destAttr="FECNIn">frCircuitReceivedFECNs</Expression>
      <Expression
destAttr="BECNIn">frCircuitReceivedBECNs</Expression>
      <Expression
destAttr="FramesIn">frCircuitReceivedFrames</Expression>
      <Expression
destAttr="FramesOut">frCircuitSentFrames</Expression>
      <Expression
destAttr="BytesIn">frCircuitReceivedOctets</Expression>
      <Expression
destAttr="BytesOut">frCircuitSentOctets</Expression>
    </ExpressionGroup>
  </Expressions>
  <MIB>RFC1315-MIB</MIB>
 </FacetType>
</DataModel>
```

基本プロパティ

カスタム ベンダー認定の基本プロパティは、ユーザが作成したその他のカスタム ベンダー認定 ファミリとの区別に役立ちます。 また、これらのプロパティは、どのベンダー MIB からメトリック データを収集しているかを示します。

基本のプロパティを確定する場合、以下の制限を考慮します。

- FacetType/name および FacetType/DisplayName プロパティ は各ベンダー認定に対して一意である必要があります。
- Protocol タグは常に SNMP です。
- 前の図の XML 例 で表示されるように、FacetType/descriptorClass プロパティおよびすべての DataModel と FacetOf プロパティを設定します。

FacetType/name

ベンダー認定を一意に識別します。

推奨事項：<MibName><TableName>Mib に準拠します。

更新の可否： いいえ

可能な値： 英数字およびアンダースコア。 ドットおよびダッシュは使用できません。

［FacetType］セクションは特定のベンダー認定を明示します。同じ XML ドキュメントには複数の［FacetType］セクションを含むことができます。それらのベンダー認定が MIB-2 実装から TCP および UDP の統計などのベンダー固有のデバイスのさまざまな要素を公開する場合です。

［FacetType］セクションにはいくつかの基本プロパティが含まれます。たとえば、このセクションにはベンダー MIB の名前が含まれ、1 つ以上の［AttributeGroup］セクションが続きます。 これらの［AttributeGroup］セクションは、この認定が MIB からどの属性を使用するかを定義します。最後に、［AttributeGroup］セクションからメトリック ファミリで指定されたメトリックに属性をマップする 1 つ以上の［ExpressionGroup］セクションがあります。

FacetType/Documentation

ベンダー認定で何が認定されるかについて説明します。

推奨事項： ベンダー、MIB 名およびテーブル名に関する詳細を含めます。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： なし

FacetType/MIB

［DEFINITIONS］節が ASN.1 ファイルで定義する MIB の名前を指定します。

推奨事項： <MibName> に準拠します

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： 管理者ユーザ インターフェースの［ベンダー認定］タブの［SNMP MIB 名］列を変更します。

更新が有効化されるタイミング： 即時

更新を有効にするのに必要なアクション: ユーザ インターフェースをリフレッシュします。

FacetType/DisplayName

CA Performance Center で表示されるように、ベンダー認定の名前を指定します。

推奨事項： ベンダー名で始まり、MIB および機能情報を含めます。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響: 管理者ユーザ インターフェースでの名前を変更します。

更新が有効化されるタイミング： 即時

更新を有効にするのに必要なアクション: ユーザ インターフェースをリフレッシュします。

AttributeGroup（ベンダー認定）

以下の例では、カスタム ベンダー認定の［AttributeGroup］セクションを示します。 このセクションでは、raw デバイス データをマップするために使用されるベンダー MIB 内の特定のテーブルの属性（変数 OID）を識別します。 このデータは、パフォーマンス メトリック、およびメトリック ファミリで定義されている設定データにマップされます。

前の図の XML 例 で示されるように、AttributeGroup/list および AttributeGroup/external プロパティを true に設定します。 これらのプロパティは、属性がそれぞれ外部ソース（MIB テーブル）から取得される値のリストを表すことを指定します。 以下の情報は、カスタマイズする ItemProperty XML エレメントの一覧を示したものです。

AttributeGroup/name

属性グループ名を指定します。

推奨事項： <FacetType/name>Group に準じます。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： なし

Documentation

（オプション）属性グループの説明を指定します。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： なし

UseIndex

複数の MIB テーブルを結合するために、この属性グループ用のインデックスとして使用される属性の名前を指定します。

推奨事項： AttributeGroup/name プロパティの値を設定します。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： なし

詳細:

AttributeGroup（複数の MIB テーブル） (P. 99)
IndexTagList (P. 85)

一般的な属性（ベンダー認定）

すべてのベンダー認定用の一般的な属性を以下に示します。

Attribute/name

属性名を指定します。

推奨事項： MIB 変数名に設定します（OBJECT-TYPE 節は ASN.1 ファイルで定義）。

更新の可否： はい

可能な値： 英数字およびアンダースコア。ドットおよびダッシュは使用できません。

更新による影響： この属性を参照するあらゆる式を更新する必要があります。

更新が有効化されるタイミング： 即時

更新の有効化に必要なアクション： なし

Attribute/type

この属性のデータ型を指定します。

推奨事項： SYNTAX 節が ASN.1 ファイルに定義する変数タイプに最も良く一致する属性タイプを使用します。

更新の可否： はい

可能な値： Boolean、Int、Long、Double、BigInteger、String、DateTime、IPAddress、MACaddress、IPSubnet、OctetString、ObjectID

更新による影響： ポーリングされた SNMP データはこのタイプに変換されます。

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

Documentation

（オプション）属性の説明を指定します。これは、MIB 変数のセマンティクス（単位など）を文書化します。

推奨事項： MIB ASN.1 ファイルからの説明を使用します。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： なし

IsKey

（オプション） MIB 変数がテーブルに対するサポートを判断するためのキーかどうかを示すフラグを使用します。複数のフィールドがキーとして指定される場合、すべてのフィールドがあわせて複合キーと見なされます。

デフォルト： false

推奨事項: コンポーネント ディスカバリ用のキー MIB オブジェクトである場合、true に設定します。

更新の可否： はい

可能な値： true、false

更新による影響： コンポーネントが新しいベンダー認定に変更される可能性があります。

更新が有効化されるタイミング： コンポーネント再検出後。

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

IsIndex

（オプション）この変数が MIB テーブルに対するインデックスかを示すフラグを使用します。

デフォルト： false

推奨事項： インデックス属性に対しては true に設定します。

更新の可否： はい

可能な値： true、false

更新による影響: コンポーネント インデックスは変わる可能性があります。

更新が有効化されるタイミング： コンポーネント再検出後。

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

NeedsDelta

（オプション）MIB 変数のデルタを取得するかどうかを示すフラグを使用します（カウンタに対して最新および前回のポーリングの差異を記録）。

デフォルト： false

推奨事項： MIB で変数が Counter、Counter32、Counter64、または TimeTicks 量として定義されている場合、true に設定します。

更新の可否： はい

可能な値： true、false

更新による影響： ポーリングされたデータが変更されます。

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

Source

属性の ObjectID を指定します。

推奨事項： OBJECT-TYPE が定義する完全修飾 MIB 変数 OID に設定します。

更新の可否： はい

可能な値： ドットで区切られた数（たとえば 1.3.6.1.4.1...）

更新による影響： データは指定された OID からポーリングされます。

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

Version

ベンダー認定のバージョンを指定します。 アップグレード時にベンダー認定をインストールすると、認証のバージョンがインストーラによって確認されます。バージョンがインストーラ パッケージ内のものより新しい場合、インストーラは既存のベンダー認定を変更しません。

デフォルト： 1.0

更新の可否： はい

可能な値： ドットで区切られた数（たとえば 1.3.6.1.4.1...）

更新による影響： バージョン属性が更新されます。

更新が有効化されるタイミング： 即時

更新の有効化に必要なアクション： なし

Author

ベンダー認定の作成者を指定します。

デフォルト： 「Custom」

更新の可否： はい

可能な値： 任意の英数文字列。

更新による影響： 作成者属性が更新されます。

更新が有効化されるタイミング： 即時

更新の有効化に必要なアクション： なし

属性のリストは、このベンダー認定によってサポートされている場合にメトリック ファミリが収集するデータのセットを指定します。 通常、このデータは 2 つのカテゴリに分類されます。

- ディスカバリ時にのみ収集されるデバイス コンポーネントの設定データ（名前またはインデックスなど）。
- すべてのポーリング サイクル時に収集されるパフォーマンス データ。

詳細:

複数の MIB テーブルのサポート (P. 94)
フィルタリングのサポート (P. 90)

設定データ属性

名前が INDEX およびタイプが ObjectID である属性は、ターゲット メトリック ファミリのインデックス属性にマップされます。 Source タグの値は、任意の変数 OID に設定できます。ただし、通常はテーブルの INDEX 節に表示される変数の 1 つを使用します。 たとえば、MIB-2 のインターフェース テーブル内の ifIndex を検討します。 この変数は同じ MIB テーブルの他の変数に対するインデックスとして機能します。 さらに、この属性用の IsIndex タグ（通常 IsKey タグ）は、true に設定されます。

この例では、ifDesc または ifType などの属性は、インターフェースについてのより多くの設定情報を提供します。 そのため、これらの属性は、ターゲット メトリック ファミリの Names および Description 属性に役立ちます。

パフォーマンス データ属性

これらの属性は、ターゲット メトリック ファミリ内のパフォーマンス メトリックに対して raw データを提供します。 以下の点について考慮してください。

- これらの属性は、メトリック ファミリ パフォーマンス メトリックに直接マップできます。
- 他の属性と共に式で使用してメトリックの値を計算することもできます。

ExpressionGroup

ExpressionGroup は以下のように属性をマップします。

- AttributeGroup から（SNMP MIB からメトリックを取得する方法を定義）
- メトリック ファミリで指定されたメトリックへ（属性がデータベースに格納される方法を定義）

MIB 値がデバイスから受信されたら、またはいくつかの正規化操作が実行されたら、データベースに MIB 値を格納できます。 たとえば、正規化操作には、値をキロバイトに変換またはキロバイトから変換するために、1024 で割るまたは 1024 を掛けることが含まれます。

ExpressionGroup/name

（オプション）式グループ名を指定します。

更新の可否： はい

可能な値： プレーン テキスト

更新による影響： なし

ExpressionGroup/destCert

入力する destAttrs が含まれるメトリック ファミリを指定します。

更新の可否： はい

可能な値： 任意の有効なメトリック ファミリ

更新による影響： 許容可能な式 destAttr を変更します。

更新が有効化されるタイミング： 即時

更新の有効化に必要なアクション： なし

ExpressionGroup/Filter

（オプション）どのコンポーネントが検出されるかを指定します。フィルタを使用して、管理されるコンポーネントの数を減らすことができます。

更新の可否： はい

可能な値： 提供されている属性を使用したブール MVEL 式

更新による影響： どのコンポーネントが検出されるかを変更します。

更新が有効化されるタイミング： コンポーネント再検出後。

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

ExpressionGroup 内のフィルタ

以下のリストでは、フィルタ例について説明します。

例 1:

```
<Filter> hrStorageType.toString() == "1.3.6.1.2.1.25.2.1.4" &amp;&amp;
        hrStorageSize != 0
</Filter>
```

この例では、サイズが 0 以外、および storageType が hrStorageFixedDisk であるパーティション インスタンスのみが、データ リポジトリで作成されます。

例 2:

```
<Filter> (jnxOperatingCPUIndex.toString() == "9.2.0.0" ||
         jnxOperatingCPUIndex.toString() == "9.1.0.0" ) &amp;&amp;
        (jnxOperatingState != 1 || jnxOperatingCPU > 0 )
</Filter>
```

例 3:

```
<Filter> (rttMonCtrlAdminRttType==9) &amp;&amp;
        ( !(rttMonCtrlAdminOwner.toString() contains "Network Health") )
</Filter>
```

例 4:

```
<Filter> (snmpOIDParser(sonetIndex,2,2).toString()=="1") </Filter>
```

この例では、OID を解析するためにベンダー認定ユーティリティ関数を使用します。

注: 全関数のリストについては、このガイドの付録を参照してください。

Expression/destAttr メトリック

以下の情報では Expression/destAttr メトリックを説明します。

Indexes

Indexes メトリック ファミリ属性の値を提供する MVEL 式を定義するために ObjectID のベンダー認定属性を使用するように指定します。

推奨： INDEX に設定します。

更新の可否： はい

可能な値： <IsIndex>true</IsIndex> を持つすべての属性。

更新による影響: コンポーネント インデックスが変更される可能性があります。

更新が有効化されるタイミング： コンポーネント再検出後

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

Names

設定データを収集するためにベンダー認定属性を使用するように指定します。この設定データは、Names メトリック ファミリ属性に対する値を指定するための MVEL 式の定義に役立ちます。

推奨事項： インスタンスを一意に識別するために必要となる情報をすべて含めます。

更新の可否： はい

可能な値： 使用可能な属性を使用している文字列 MVEL 式

更新による影響： コンポーネント名の変更

更新が有効化されるタイミング： コンポーネント再検出後

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

Descriptions

（オプション）設定データを収集するためにベンダー認定属性を使用するように指定します。 この設定データは、Descriptions メトリック ファミリに対する値を指定するための MVEL 式の定義に役立ちます。すべてのメトリック ファミリが Descriptions 属性をサポートするとは限りません。

推奨事項： インスタンスを説明できる量の情報を含めます。

更新の可否： はい

可能な値： 使用可能な属性を使用している文字列 MVEL 式

更新による影響： コンポーネント説明の変更

更新が有効化されるタイミング： コンポーネント再検出時

更新の有効化に必要なアクション： メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

その他のメトリック

設定またはパフォーマンスのデータを収集するためにベンダー認定属性を使用するように指定します。このデータは、メトリック ファミリ属性に対する値を指定する MVEL 式を定義するために使用されます。

追加の可否： はい（destAttr がメトリック ファミリに存在する場合）。

更新の可否： はい

可能な値： 宛先属性のタイプに一致する値を生成し、使用可能な属性を使用する MVEL 式。

更新による影響： ポーリングされた値の変更

更新が有効化されるタイミング： 次のポーリング時

更新の有効化に必要なアクション： なし

メトリック ファミリは URI （たとえば {http://im.ca.com/normalizer}*FamilyName.AttributeName*）を提示します。これは、ExpressionGroup で個別に参照されます。ExpressionGroup/destCert プロパティは、URI（たとえば {http://im.ca.com/normalizer}*FamilyName*）に設定され、Expression/destAttr は *AttributeName* に設定されます。

詳細:

設定データ属性 (P. 79)

HierarchyList

以下の情報は階層動作を定義します。

Hierarchy/ParentFacet

候補の親アイテムを検索するために使用されるファセットの QName を指定します。

更新の可否： はい

可能な値： 任意の有効なファセット

更新による影響： 階層構築を変更します。

更新が有効化されるタイミング： コンポーネント再検出後

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

Hierarchy/ParentAttribute

特定の親アイテムを識別するために使用される属性の QName を指定します。

更新の可否： はい

可能な値： 任意の有効な属性 QName

更新による影響： 階層構築を変更します。

更新が有効化されるタイミング： コンポーネント再検出後

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

Hierarchy/ChildAttribute

親アイテム上の ParentAttribute に一致させるために使用される子アイテム上の属性の QName を指定します。

更新の可否： はい

可能な値： 任意の有効な属性 QName

更新による影響： 階層構築を変更します。

更新が有効化されるタイミング： コンポーネント再検出後

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

IndexTagList

複数の MIB テーブルからの属性をポーリングするには、これらの属性が含まれる MIB テーブルごとに属性グループが必要です。 インデックス タグ リストは、異なるインデックスの 2 つの属性グループ（または MIB テーブル）を関連付けるためのメカニズムを提供します。 1 つのテーブルの 1 つのアイテム（行）が別のテーブル内の対応する行にリンクされるように、グループが関連付けられます。

PrimaryTag

プライマリ属性グループを参照します（ObjectID タイプのインデックス属性を定義するグループ）。 このエレメントの値は、プライマリ グループ用の属性グループの「UseIndex」タグと等しい必要があります。

更新の可否： はい

可能な値： プライマリ属性グループに対応する属性グループの「UseIndex」タグ。

更新による影響： インデックスを変更します

更新が有効化されるタイミング： コンポーネント再検出後

更新の有効化に必要なアクション： メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

IndexTag

セカンダリ グループの行にプライマリ グループ（または MIB テーブル）の行を関連付ける方法を定義します。 このエレメントは、一致させる必要がある両方のグループの属性を指定することで行を関連付けます。

IndexTag/Name

セカンダリ グループ（または MIB テーブル）を参照します。 このエレメントの値は、プライマリ属性グループで関連付けようとしているセカンダリ属性グループの「UseIndex」タグと等しくする必要があります。

更新の可否： はい

可能な値： セカンダリ属性グループの「UseIndex」タグ。

更新による影響： インデックスを変更します

更新が有効化されるタイミング： コンポーネント再検出後

更新の有効化に必要なアクション： メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

IndexTag/PrimaryKeyExpression

プライマリ属性グループの属性、または以前に定義された IndexTag のいずれかに対応する属性グループの属性が含まれる MVEL 式を指定します。 計算値は「ThisTagKeyExpression」と照合されます。 一致する場合、両方の属性グループ（または MIB テーブル）の行がリンクされます。その後、これらの属性は、destAttr（またはメトリック）をサポートする「Expression」で共に使用できます。

更新の可否： はい

可能な値： 有効な MVEL 式

更新による影響： インデックスを変更します

更新が有効化されるタイミング： コンポーネント再検出後

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

IndexTag/ThisTagKeyExpression

セカンダリ属性グループの属性が含まれる MVEL 式を指定します。 計算値は「PrimaryKeyExpression」と照合されます。 一致する場合、両方のグループ（または MIB テーブル）の行がリンクされます。 その後、これらの属性は、destAttr （またはメトリック）をサポートする「Expression」で共に使用できます。

更新の可否： はい

可能な値： 有効な MVEL 式

更新による影響： インデックスを変更します

更新が有効化されるタイミング： コンポーネント再検出後

更新の有効化に必要なアクション: メトリック ファミリを更新するか、またはベンダー認定優先度を変更します。

カスタム ベンダー認定のインポート

ベンダー認定を有効にするには、それを Data Aggregator にインポートします。 ベンダー認定をインポートするには、以下の 2 つの方法があります。

- REST クライアントを使用してカスタム ベンダー認定をインポートする (P. 87)。
- カスタム ベンダー認定インストーラを使用する (P. 88)

REST クライアントを使用してカスタム ベンダー認定をインポートする

任意の REST クライアントを使用してベンダー認定をインポートします。 CA Performance Management 内で RESTful Web サービスへの GET、POST、PUT およびその他のメソッドを実行するために RESTClient を使用することをお勧めします。 インストールしていない場合、 http://code.google.com/p/rest-client/ から RESTClient GUI JAR ファイルを取得できます。

RESTClient を使用する場合、以下の情報を考慮してください。

- JAR ファイルをダブルクリックして起動します。
- POST XML を実行する場合、Charset が UTF-8 に設定されていることを確認します。

  この設定を表示して確認するには、［Edit Content-Type & Charset］ボタンをクリックします。
- また以下のように Response Body をオート インデントできます。
  1. ［Tools］メニューの［Options］をクリックします。
  2. ［Etc.］タブを選択します。
  3. ［Auto-indent Response Body］を選択し、［OK］をクリックします。

次の手順に従ってください:

1. http://*da-hostname*:8581/typecatalog/certifications/snmp を URL として入力します。
2. ［Method］タブで POST を選択します。
3. Body 設定内の「Body Content-type」として「application/xml」を選択します。

   重要: Content-type の設定に失敗すると 404 エラーとなります。
4. ［Body］タブにカスタム ベンダー認定 XML をコピーして貼り付けます。
5. メソッドを実行します。

   ベンダー認定がインポートされます。 エラーが発生しない場合、［HTTP Response］セクション内の［Status］フィールドには以下が表示されます。

   ```
   HTTP/1.1 200 OK
   ```

注: その他のリターン コードは、カスタム コンポーネントを更新する間にエラーが発生したことを示します。エラーを修正し、別の POST を実行することによりコンポーネントを更新することを再試行します。

重要: データの損失を回避するには、ベンダー認定、メトリック ファミリ、またはコンポーネントを作成または更新するたびに、展開ディレクトリを必ずバックアップします。

カスタム ベンダー認定インストーラを使用する

CA Performance Management 認定インストーラ（Data Aggregator 上にある）を使用して認定をインポートします。インストーラでは、以下の 3 つの方法で認定をインストールできます。

- 単一の認定
- ZIP ファイル形式での認定のコレクション
- ディレクトリ内のすべての認定（認定が含まれるフォルダを直接指定することが必要）

次の手順に従ってください:

1. インストーラをダウンロードして実行します。

   インストーラは以下の URL で入手できます。
   `http://<DA>:8581/cert/install.htm`

   <DA>

   Data Aggregator のホスト名。

   注: デフォルトの REST ポートは 8581 です。

2. インストーラ用の言語を選択します。
3. 対応するプロンプトが表示されたら、以下の情報を入力します。
   - DA ホスト名
   - DA REST ポート
4. 単一の認定、認定のコレクション（ZIP）、またはディレクトリ内のすべての認定をインストールするかどうかを選択します。

   ディレクトリからインストールする場合は、認定が置かれているフォルダを選択します。認定インストーラはサブディレクトリ内の認定は検索しません。

5. インストーラは、インストール プロセス中に各認定のバージョンを確認します。

   認定がインストール済みであることが Data Aggregator によって示されると、認定インストーラは認定のアップグレードを試みます。 インストーラが認定をアップグレードできない場合は、障害の原因を明示するエラーが表示されます。

6. インストーラが認定のインストールを完了した場合、検出されたエラーがあれば表示されます。

   インストーラがコンピュータの Home ディレクトリに作成したログ ファイルでエラーの詳細を確認できます。

注: Linux コマンド ラインまたは UI インストーラを使用して認定をインストールすることもできます。

カスタム ベンダー認定結果の確認

カスタム ベンダー認定 XML をインポートしたら、結果を確認します。 この例では、rPVCInfoCustom カスタム ベンダー認定が正常にインポートされたことを確認します。

次の手順に従ってください:

1. REST クライアントに以下の URL を入力します。

   `http://da_hostname:8581/typecatalog/certifications/snmp/name`

   name

   カスタム ベンダー認定の名前です。この例では frPVCInfoCustom です。

   カスタムの frPVCInfoCustom ベンダー認定 XML がページに表示されます。

2. 以下の URL を REST クライアントに入力します。

   `http://da-hostname:8581/rest/vendorpriorities`

3. 表示される XML ドキュメント内でベンダー認定を検索します。

4. MetricFamilyVendorPriority XML に、カスタム ベンダー認定とメトリック ファミリの両方がリスト表示されていることを確認します。

Data Aggregator ユーザ インターフェースを使用してカスタム ベンダー認定を確認する準備ができました。

次の手順に従ってください:

1. カスタム ベンダー認定が Data Aggregator で使用できることを確認します。
   a. Data Aggregator データ ソースに対して、［監視設定］メニューから［ベンダー認定］をクリックします。

      該当するカスタム ベンダー認定がリスト表示されていることを確認します（またはベンダー認定を検索します）。
   b. ベンダー認定のリストからカスタム ベンダー認定を選択します。

      カスタム メトリック ファミリが［メトリック ファミリ］ビューに表示されていることを確認します。
   c. ［メトリック ファミリ］ビューで対象のメトリック ファミリをクリックします。
   d. ［ベンダー認定優先度］タブをクリックします。

      カスタム ベンダー認定が「すべての管理可能デバイス」コレクションと関連付けられていることを確認します。
2. カスタム ベンダー認定がサポートするメトリック ファミリに対して、監視プロファイルおよびデバイス コレクションを設定します。
   a. ディスカバリによって、期待される適切なアイテムが作成されることを確認します。
   b. アイテムが、メトリック ファミリ XML で指定されたメトリック データをポーリングし、収集することを確認します。
   c. 収集されたデータが正しいかどうかを検証します。
   d. アイテムの同期が設定されている場合は、アイテム データが CA Performance Center と正しく同期されることを確認します。
   e. ディスカバリをもう一度実行し、情報が正しく更新されていることを確認します。

フィルタリングのサポート

ExpressionGroup セクションの Filter タグは、コンポーネントのディスカバリ プロセスに対してフィルタリングを適用するために使用されます。

前述の例でいえば、アクティブなフレーム リレー PVC のみを監視したいとします。 そのためには、frPVCInfoCustom カスタム ベンダー認定にフィルタ タグを適用し、これらの特定のコンポーネントを監視します。 フィルタは ExpressionGroup に追加されます。

注: フィルタ式には MVEL 構文を使用できます。 MVEL 構文、ベンダー認定ユーティリティ関数、ベンダー認定グローバル変数の詳細については、このガイドの付録を参照してください。

更新された frPVCInfoCustom カスタム ベンダー認定の例を以下に示します。この手順で加えた変更は太字で示されます。

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Auto-generated by the type catalog local manager.-->
<DataModel namespace="http://im.ca.com/certifications/snmp"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="SNMPCertificationFacet.xsd">
  <FacetType name="frPVCInfoCustom"
descriptorClass="com.ca.im.core.datamodel.certs.CertificationFacet
DescriptorImpl">
    <Documentation>Frame Relay PVC Vendor
Certification</Documentation>
    <FacetOf namespace="http://im.ca.com/core" name="Item" />
    <AttributeGroup name="AttributeGroup" external="true"
list="true">
      <Documentation />
      <Attribute name="INDEX" type="ObjectID">
        <Documentation />
        <Source>1.3.6.1.2.1.10.32.2.1.4</Source>
        <IsIndex>true</IsIndex>
        <IsKey>false</IsKey>
        <NeedsDelta>false</NeedsDelta>
      </Attribute>
      <Attribute name="frCircuitReceivedFECNs" type="Long">
        <Documentation />
        <Source>1.3.6.1.2.1.10.32.2.1.4</Source>
        <IsIndex>false</IsIndex>
        <IsKey>true</IsKey>
        <NeedsDelta>true</NeedsDelta>
      </Attribute>
      <Attribute name="frCircuitReceivedBECNs" type="Long">
        <Documentation />
        <Source>1.3.6.1.2.1.10.32.2.1.5</Source>
        <IsIndex>false</IsIndex>
        <IsKey>true</IsKey>
        <NeedsDelta>true</NeedsDelta>
      </Attribute>
      <Attribute name="frCircuitSentFrames" type="Long">
        <Documentation />
        <Source>1.3.6.1.2.1.10.32.2.1.6</Source>
        <IsIndex>false</IsIndex>
        <IsKey>true</IsKey>
        <NeedsDelta>true</NeedsDelta>
      </Attribute>
      <Attribute name="frCircuitSentOctets" type="Long">
        <Documentation />
        <Source>1.3.6.1.2.1.10.32.2.1.6</Source>
        <IsIndex>false</IsIndex>
```

```
          <IsKey>true</IsKey>
          <NeedsDelta>true</NeedsDelta>
        </Attribute>
        <Attribute name="frCircuitReceivedFrames" type="Long">
          <Documentation />
          <Source>1.3.6.1.2.1.10.32.2.1.8</Source>
         <IsIndex>false</IsIndex>
         <IsKey>true</IsKey>
         <NeedsDelta>true</NeedsDelta>
        </Attribute>
        <Attribute name="frCircuitReceivedOctets" type="Long">
          <Documentation />
          <Source>1.3.6.1.2.1.10.32.2.1.9</Source>
          <IsIndex>false</IsIndex>
          <IsKey>true</IsKey>
          <NeedsDelta>true</NeedsDelta>
        </Attribute>
        <Attribute name="frCircuitState" type="int">
          <Documentation />
          <Source>1.3.6.1.2.1.10.32.2.1.3</Source>
          <IsIndex>false</IsIndex>
          <IsKey>false</IsKey>
          <NeedsDelta>false</NeedsDelta>
        </Attribute>
      </AttributeGroup>
      <Protocol>SNMP</Protocol>
      <DisplayName>Frame Relay PVC Certification</DisplayName>
      <Expressions>
        <ExpressionGroup
destCert="{http://im.ca.com/normalizer}frPVCInfo"
name="frPVCInfoDS">
          <Filter>(frCircuitState==2)</Filter>
          <Expression destAttr="Indexes">INDEX</Expression>
          <Expression destAttr="Names">"Frame Relay " +
INDEX</Expression>
          <Expression
destAttr="FECNIn">frCircuitReceivedFECNs</Expression>
          <Expression
destAttr="BECNIn">frCircuitReceivedBECNs</Expression>
          <Expression
destAttr="FramesIn">frCircuitReceivedFrames</Expression>
          <Expression
destAttr="FramesOut">frCircuitSentFrames</Expression>
          <Expression
destAttr="BytesIn">frCircuitReceivedOctets</Expression>
          <Expression
destAttr="BytesOut">frCircuitSentOctets</Expression>
          <Expression
destAttr="BitsIn">frCircuitReceivedOctets*8</Expression>
```

```
        <Expression
destAttr="BitsOut">frCircuitSentOctets*8</Expression>
      </ExpressionGroup>
    </Expressions>
    <MIB>RFC1315-MIB</MIB>
   </FacetType>
</DataModel>
```

これらの変更は、カスタム ベンダー認定をインポートした後に行われたと見なします。 そのため、これらを変更した後、ベンダー認定を更新します。

詳細:

カスタム ベンダー認定の更新 (P. 101)

複数の MIB テーブルのサポート

ベンダー MIB を認定する場合、2 つ以上のテーブルから特定のメトリック ファミリ用の raw データを収集する必要がある場合があります。

CA Performance Management は、複数の MIB テーブルへのアクセスを必要とするベンダー認定に対してサポートを提供します。 このサポートは、単一テーブルの認定ドキュメント構造である XML への拡張機能に基づきます。これにより、共通キー（インデックス）を使用して、複数のテーブルから収集されるデータを結合することができます。 前述の例でいえば、アクティブなフレーム リレー PVC のみを監視したいとします。 複数の MIB テーブル サポートを追加するために、frPVCInfoCustom カスタム ベンダー認定を変更します。

この例では、フレーム リレー PVC の名前は、ifXTable 内の ifName MIB オブジェクトと、この PVC に対してデータ リンク接続識別子（DLCI）を提供する frCircuitDlci オブジェクトの組み合わせを使用して命名できます。 この種の命名規則は、PVC がどのフレーム リレー インターフェースにリレーされるかを決めるのに役立ちます。

以下の情報を追加するためにカスタム ベンダー認定を変更します。

- frCircuitDlci MIB オブジェクトを表すために*新しい*属性を既存の AttributeGroup に追加します。
- 使用する ifName MIB オブジェクトは、ユーザのカスタム ベンダー認定に含まれていない MIB 由来です。 新しい AttributeGroup （この場合 ifXTable）を追加し、次に新しい属性（ifName）を追加します。

更新された frPVCInfoCustom カスタム ベンダー認定の例を以下に示します。この手順で加えた変更は太字で示されます。

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Auto-generated by the type catalog local manager.-->
<DataModel namespace="http://im.ca.com/certifications/snmp"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="SNMPCertificationFacet.xsd">
   <FacetType name="frPVCInfoCustom"
descriptorClass="com.ca.im.core.datamodel.certs.CertificationFacet
DescriptorImpl">
     <Documentation>Frame Relay PVC Vendor
Certification</Documentation>
     <FacetOf namespace="http://im.ca.com/core" name="Item" />
     <AttributeGroup name="ifXTableGroup" external="true"
list="true">
     <Documentation>This pulls data from the ifXTable so that the
ifName corresponding to the PVC can be referenced</Documentation>
     <UseIndex>ifXIndexTag</UseIndex>
     <Attribute name="ifXTableIndex" type="ObjectID">
       <Documentation />
       <IsKey>false</IsKey>
       <IsIndex>true</IsIndex>
       <Source>1.3.6.1.2.1.31.1.1.1.1</Source>
       <Polled>false</Polled>
     </Attribute>
     <Attribute name="ifName" type="OctetString">
       <Documentation />
       <IsKey>false</IsKey>
       <IsIndex>false</IsIndex>
       <Source>1.3.6.1.2.1.31.1.1.1.1</Source>
       <Polled>false</Polled>
      </Attribute>
     </AttributeGroup>
     <IndexTagList>
       <PrimaryTag>PVCIndexTag</PrimaryTag>
       <IndexTag>
         <Name>ifXIndexTag</Name>

<PrimaryKeyExpression>snmpOIDParser(INDEX,1,1)</PrimaryKeyExpressi
on>

<ThisTagKeyExpression>ifXTableIndex</ThisTagKeyExpression>
       </IndexTag>
     </IndexTagList>
     <AttributeGroup name="AttributeGroup" external="true"
list="true">
       <Documentation />
       <UseIndex>PVCIndexTag</UseIndex>
```

```
<Attribute name="INDEX" type="ObjectID">
  <Documentation />
  <Source>1.3.6.1.2.1.10.32.2.1.4</Source>
  <IsIndex>true</IsIndex>
  <IsKey>false</IsKey>
  <NeedsDelta>false</NeedsDelta>
</Attribute>
<Attribute name="frCircuitReceivedFECNs" type="Long">
  <Documentation />
  <Source>1.3.6.1.2.1.10.32.2.1.4</Source>
  <IsIndex>false</IsIndex>
  <IsKey>true</IsKey>
  <NeedsDelta>true</NeedsDelta>
</Attribute>
<Attribute name="frCircuitReceivedBECNs" type="Long">
  <Documentation />
  <Source>1.3.6.1.2.1.10.32.2.1.5</Source>
  <IsIndex>false</IsIndex>
  <IsKey>true</IsKey>
  <NeedsDelta>true</NeedsDelta>
</Attribute>
<Attribute name="frCircuitSentFrames" type="Long">
  <Documentation />
  <Source>1.3.6.1.2.1.10.32.2.1.6</Source>
  <IsIndex>false</IsIndex>
  <IsKey>true</IsKey>
  <NeedsDelta>true</NeedsDelta>
</Attribute>
<Attribute name="frCircuitSentOctets" type="Long">
  <Documentation />
  <Source>1.3.6.1.2.1.10.32.2.1.6</Source>
  <IsIndex>false</IsIndex>
  <IsKey>true</IsKey>
  <NeedsDelta>true</NeedsDelta>
</Attribute>
<Attribute name="frCircuitReceivedFrames" type="Long">
  <Documentation />
  <Source>1.3.6.1.2.1.10.32.2.1.8</Source>
  <IsIndex>false</IsIndex>
  <IsKey>true</IsKey>
  <NeedsDelta>true</NeedsDelta>
</Attribute>
<Attribute name="frCircuitReceivedOctets" type="Long">
  <Documentation />
  <Source>1.3.6.1.2.1.10.32.2.1.9</Source>
  <IsIndex>false</IsIndex>
  <IsKey>true</IsKey>
  <NeedsDelta>true</NeedsDelta>
 </Attribute>
```

```
          <Attribute name="frCircuitState" type="int">
            <Documentation />
            <Source>1.3.6.1.2.1.10.32.2.1.3</Source>
            <IsIndex>false</IsIndex>
            <IsKey>false</IsKey>
            <NeedsDelta>false</NeedsDelta>
          </Attribute>
          <Attribute name="frCircuitDlci" type="int">
            <Documentation />
            <Source>1.3.6.1.2.1.10.32.2.1.2</Source>
            <IsIndex>false</IsIndex>
            <IsKey>false</IsKey>
            <NeedsDelta>false</NeedsDelta>
          </Attribute>
        </AttributeGroup>
        <Protocol>SNMP</Protocol>
        <DisplayName>Frame Relay PVC Certification</DisplayName>
        <Expressions>
         <ExpressionGroup
destCert="{http://im.ca.com/normalizer}frPVCInfo"
name="frPVCInfoDS">
         <Filter>(frCircuitState==2)</Filter>
         <Expression destAttr="Indexes">INDEX</Expression>
         <Expression destAttr="Names">isdef(ifName)?
(isdef(frCircuitDlci) ? ifName + " DCLI:" + frCircuitDlci : "Frame
Relay " + INDEX) : "Frame Relay " + INDEX</Expression>
         <Expression
destAttr="FECNIn">frCircuitReceivedFECNs</Expression>
         <Expression
destAttr="BECNIn">frCircuitReceivedBECNs</Expression>
         <Expression
destAttr="FramesIn">frCircuitReceivedFrames</Expression>
         <Expression
destAttr="FramesOut">frCircuitSentFrames</Expression>
         <Expression
destAttr="BytesIn">frCircuitReceivedOctets</Expression>
         <Expression
destAttr="BytesOut">frCircuitSentOctets</Expression>
         <Expression
destAttr="BitsIn">frCircuitReceivedOctets*8</Expression>
         <Expression
destAttr="BitsOut">frCircuitSentOctets*8</Expression>
        </ExpressionGroup>
       </Expressions>
      <MIB>RFC1315-MIB</MIB>
     </FacetType>
</DataModel>
```

これらの変更は、カスタム ベンダー認定をインポートした後に行われたと見なします。 そのため、これらを変更した後、ベンダー認定を更新します (P. 101)。

AttributeGroup（複数の MIB テーブル）

テーブルはそれぞれの AttributeGroup セクションに入る必要があります。そのテーブル上の各属性は、その AttributeGroup の子として追加されます。

以下の情報についてはこれらのセクションを参照してください。

- AttributeGroup 情報 - プライマリおよびセカンダリのテーブル属性を定義するために使用される XML エレメントに関する詳細。
- UseIndex および IndexTagList 情報 - プライマリおよびセカンダリの属性グループを結合するための XML エレメントに関する詳細。

このシナリオでは、プライマリ属性グループは、より多くの情報で拡張したいテーブルを表します。 セカンダリ属性グループには、プライマリに対する拡張情報が含まれます。

プライマリ AttributeGroup には、セカンダリ AttributeGroup への共通の「キー」として役立つ MIB テーブル変数を識別する属性が含まれます。

セカンダリ AttributeGroup には、プライマリ テーブルに対して「拡張」情報を有するすべての MIB テーブル変数の属性定義が含まれます。 さらに、プライマリ AttributeGroup からの共通の「キー」に一致する変数を識別する属性があります。

詳細:

AttributeGroup （ベンダー認定） (P. 73)
UseIndex (P. 100)
IndexTagList（複数の MIB テーブル） (P. 100)

UseIndex

AttributeGroup はそれぞれ UseIndex タグを与えられます。UseIndex タグを使用して、OID を共通の名前にグループ化できます。 この共通名は、指定された変数に関連付けられ、対応する MIB テーブルへの共通キー（インデックス）として機能します。

以下の情報は、カスタマイズする XML エレメントをまとめたものです。

AttributeGroup/UseIndex

IndexTagList セクションで使用されるプライマリおよびセカンダリのタグ名をそれぞれ一意に識別します。

推奨事項： AttributeGroup/name プロパティの値を設定します。

詳細情報:

AttributeGroup（複数の MIB テーブル） (P. 99)

IndexTagList（複数の MIB テーブル）

IndexTagList セクションは、異なるインデックスの 2 つの属性グループ（または MIB テーブル）を関連付けるメカニズムです。 グループが関連付けられた場合、1 つのアイテムには複数のテーブルからの複数のインデックス ID があります。

IndexTagList セクションには、すべてのセカンダリ属性グループに対する IndexTag セクションなど、すべての結合情報が含まれます。

IndexTagList/PrimaryTag

プライマリ属性グループ（または MIB テーブル）を定義します。 プライマリ AttributeGroup の UseIndex プロパティの値に設定します。

IndexTag/Name

セカンダリ属性グループを定義します。 セカンダリ AttributeGroup の UseIndex プロパティの値に設定します。

IndexTag/PrimaryKeyExpression

プライマリ テーブル内の共通キーを生成する式を指定します。指定のプライマリ テーブル インデックス属性から共通キーを派生させるために MVEL 関数を使用することを考慮してください。

IndexTag/ThisTagKeyExpression

セカンダリ テーブル内の共通キーを生成する式を指定します。指定のセカンダリ テーブル インデックス属性から共通キーを派生させるために MVEL 関数を使用することを考慮してください。

このマルチテーブル アプローチは、2 つ以上のテーブルの結合をサポートします。 複数のテーブル結合には、以下の 2 種類の関係が存在します。

- **プライマリ –> セカンダリ #1、プライマリ –> セカンダリ #2**
  セカンダリ テーブルの順序はインデックス タグ リストにおいて重要ではありません。
- **プライマリ –> セカンダリ #1 –> セカンダリ #2**
  テーブルがマージされる方法に基づいて、セカンダリ テーブル #2 の前にセカンダリ テーブル #1 を配置します。

プライマリ テーブル内の 1 つ以上の行は、構造上セカンダリ テーブル内の同じ行にマップできます。 セカンダリ テーブル上のキーは順番に検索され、最初に一致したものが採用されます。

詳細:

複数の MIB テーブルのサポート (P. 94)
AttributeGroup（複数の MIB テーブル） (P. 99)

カスタム ベンダー認定の更新

既存のカスタム ベンダー認定は更新できます。 前述の例でいえば、カスタム メトリック ファミリ、およびカスタム ベンダー認定に若干の変更を加えていました。

メトリック ファミリには、メトリックを追加しました。 ベンダー認定には、MIB テーブルを追加し、カスタム ベンダー認定へのフィルタリングを含めました。 ここで、変更を有効にするためにベンダー認定を更新します。

注: タグまたは属性を更新する影響の詳細については、カスタム ベンダー認定 XML の詳細を確認してください。 具体的には、特定の属性説明を確認します。

次の手順に従ってください:

1. 以下の アドレス を URL フィールドに入力します。

   `http://da_hostname:8581/typecatalog/certifications/snmp/CustomVendorCertName`

   CustomVendorCertName

   更新するカスタム ベンダー認定の名前です。 この例では、ベンダー認定は frPVCInfoCustom という名前です。

2. ベンダー認定がその URL に存在することを確認するには、［Method］タブで GET を選択します。
3. ベンダー認定が URL に存在することを確認した後、［Method］タブで PUT を選択します。
4. 更新したカスタム メトリック ファミリ XML を［Body］タブ（［編集］フィールド）にコピーして貼り付け、Content-type を application/xml に設定します。

   重要: Content-type の設定に失敗すると 404 エラーとなります。

5. URL フィールドの隣の［Go］ボタンをクリックします。

   カスタム ベンダー認定が更新されました。 エラーが発生しない場合、［HTTP Response］セクションの［Status］フィールドには以下のメッセージが表示されます。

   `HTTP/1.1 200 OK`

   注: その他のリターン コードは、カスタム ベンダー認定を更新する間にエラーが発生したことを示します。 エラーを修正し、別の PUT を実行することによりベンダー認定の更新を再試行します。

   重要: データの損失を回避するには、ベンダー認定、メトリック ファミリ、またはコンポーネントを作成または更新するたびに、展開ディレクトリを必ずバックアップします。

詳細:

ベンダー認定 XML 構造について (P. 69)
カスタム メトリック ファミリの更新 (P. 65)

付録 A: ベンダー認定式: 式演算子、関数、およびグローバル変数

このセクションでは、ベンダー認定式で使用できる式演算子、関数、およびグローバル変数の一部について説明します。

ベンダー認定式では MVEL 構文を使用できます。 MVEL は、Java に近い構文を持つ、Java 環境用に公開されている埋め込み可能な式言語です。MVEL では Java の式に似た式をサポートします。

演算子を使用して式を構築したり、中かっこを使用して優先順位を制御したり、セミコロンによってステートメントを終了できます。 MVEL 言語の詳細なリファレンスについては、http://mvel.codehaus.org を参照してください。

MVEL 言語には、ベンダー認定式でも使用できるベンダー認定ユーティリティ関数およびベンダー認定グローバル変数が備わっています。

関数、オペレータ、グローバル変数の使用方法については、CA Performance Center の［ベンダー認定］タブで参照できます。

このセクションには、以下のトピックが含まれています。

式演算子

このセクションでは、ベンダー認定式で使用できる演算子について説明します。

ベンダー認定式では MVEL 構文を使用できます。 MVEL は、Java に近い構文を持つ、Java 環境用に公開されている埋め込み可能な式言語です。MVEL では Java の式に似た式をサポートします。

演算子を使用して式を構築したり、中かっこを使用して優先順位を制御したり、セミコロンによってステートメントを終了できます。 MVEL 言語の詳細なリファレンスについては、http://mvel.codehaus.org を参照してください。

以下の表に、利用可能な演算子の概要を示します。

注: XML ドキュメントでは、XNE （XML Named Entities）表記を使用してください。

オペレータ	XNE	説明	例
=		割り当て	a = 1
==		等しい	"fred" == "fred"
!=		等しくない	"fred" != "tom"
>	&gt;	より大きい	1 > 0 is true
<	&lt;	より小さい	0 < 1 is true
>=		以上	1 >= 0 is true
<=		以下	1 <= 1 is true
contains		左側の値に右側の値が含まれるかどうかを検証します	"tomcat" contains "cat"
isdef		変数が定義済みかどうかをテストします	isdef a
+		追加	1 + 1
+		連結	"one " + "two"
-		減算	2 - 1
*		乗算	2 * 2
/		除算	4 / 2
%		剰余	5 % 2
&&	&&	論理 AND	(x>-1) && (x<1)
||		論理 OR	(x<-1) || (x>1)
&	&amp;	AND ビット操作	17 & 0xF

オペレータ	XNE	説明	例
|		OR ビット操作	4 | 1
^		排他 OR ビット操作	5 ^ 1
!		否定	! True
?		三項演算子	age > 17 ? "allow" : "deny"

関数とグローバル変数

このセクションでは、ベンダー認定式で使用できる式演算子、関数、およびグローバル変数の一部について説明します。

ベンダー認定式では MVEL 構文を使用できます。 MVEL は、Java に近い構文を持つ、Java 環境用に公開されている埋め込み可能な式言語です。MVEL では Java の式に似た式をサポートします。

演算子を使用して式を構築したり、中かっこを使用して優先順位を制御したり、セミコロンによってステートメントを終了できます。 MVEL 言語の詳細なリファレンスについては、http://mvel.codehaus.org を参照してください。

MVEL 言語には、ベンダー認定式でも使用できるベンダー認定ユーティリティ関数およびベンダー認定グローバル変数が備わっています。

関数およびグローバル変数の使用を調べるには、CA Performance Center の［ベンダー認定］タブを使用します。

availabilityWithSysUptime 関数

この関数は、sysUptime およびポーリング期間を使用して可用性の割合（%）を計算し、猶予期間を与えます。 この関数はデバイスの認定に使用します。

構文

この関数の形式は以下のとおりです。

```
Object availabilityWithSysUptime (Long sysUpTime, Long duration)
```

パラメータ

sysUpTime

システムのネットワーク管理コンポーネントが最後に再初期化されてからの経過時間（100 分の 1 秒単位）です。

duration

秒単位のポーリング期間です。 グローバル変数 _rspDuration を使用します。 詳細については、「高度な例」を参照してください。

戻り値

割合（0-100）として可用性を返すか、または無効なデータが渡される場合は「NULL」を返します。

例

以下の式は、sysUpTime が 30000 でポーリング期間が 300 の場合、以下のような結果になります。

式：

```
availabilityWithSysUptime (sysUpTime, duration)
```

結果：

```
100
```

同じ式で、sysUpTime が 6000 でポーリング期間が 300 の場合、以下のような結果になります。

結果：

```
20
```

同じ式で、sysUpTime が 30005 でポーリング期間が 300 の場合、以下のような結果になります。

結果：

```
100
```

高度な例

「System Statistics」ベンダー認定から、以下の式が取得されます。

```
Availability=availabilityWithSysUptime(sysUpTime,_rspDuration)
```

mapModel 関数

この関数は、objectID（sysObjectID）の値を使用してシステム OID をモデル名の文字列にマップします。 この関数はデバイスの認定に使用します。

構文

この関数の形式は以下のとおりです。

```
String  mapModel ( ObjectID sysObjectID )
```

パラメータ

sysObjectID

解析するオブジェクト ID 値です。

戻り値

マップされたモデル名を含む文字列を返します。

例

以下の式で、OID 値が 1.3.6.1.4.1.9.1.223 の場合、以下のような結果になります。

式：

```
mapModel (oid )
```

結果：

```
Cisco7204VXR
```

同じ式で、OID 値が 1.3.6.1.4.1 の場合、以下のような結果になります。

結果：

```
Unknown 1.3.6.1.4.1
```

高度な例

「System Statistics」ベンダー認定から、以下の式が取得されます。

```
Model=mapModel (sysObjectID)
```

mapVendor 関数

この関数は、objectID（sysObjectID）の値を使用してシステム OID をベンダー名の文字列にマップします。 この関数はデバイスの認定に使用します。

構文

この関数の形式は以下のとおりです。

```
String  mapVendor ( ObjectID sysObjectID )
```

パラメータ

sysObjectID

解析するオブジェクト ID 値です。

戻り値

マップされたベンダー名を含む文字列を返します。

例

以下の式は、OID 値が 1.3.6.2.1.2.2636.0 の場合、以下のような結果になります。

式：

```
mapVendor (oid )
```

結果：

```
Juniper
```

同じ式で、OID 値が 1.3.6.2.1.2.1234567.0 の場合、以下のような結果になります。

結果：

不明

高度な例

「System Statistics」ベンダー認定から、以下の式が取得されます。

```
Model=mapVendor (sysObjectID)
```

snmpConstArrayMap 関数

この関数は、値（インデックス）を 1 セットの定数値（配列）にマップします。 必要な場合、この関数は入力値を最も近い整数値に丸めます。 その後、c0、c1 ～ cn-1 のように表される定数値（配列）セットへのインデックスとして整数値を使用します。c の値は整数でなければなりません。この関数は、式が解析されて *cx* が返されたときに、これらの値を確認します。 値が 0 から n-1（この値を含む）までの範囲にない場合、結果は 0 です（エラー メッセージは表示されません）。 この関数はデバイスの認定に使用します。

構文

この関数の形式は以下のとおりです。

```
Integer  snmpConstArrayMap(Double index, Integer[] array)
```

パラメータ

index

配列へのインデックスとして使用される Double 値。

array

任意の範囲の整数値です。

戻り値

配列から整数値を返します。 NULL のインデックス値は NULL を返します。

例

以下の式は、インデックス 2、配列 {5, 6, 7, 8, 9, 4} の場合、以下のような結果になります。

式：

```
snmpConstArrayMap (index, array)
```

結果：

```
7
```

以下の式は、インデックス 4.88、配列 {5, 6, 7, 8, 9, 4} の場合、以下のような結果になります。

式：

```
snmpConstArrayMap (value, array)
```

結果：

```
4
```

高度な例

「Generic Modem」ベンダー認定から、以下の式が取得されます。

```
SpeedOut=snmpConstArrayMap(mdmCsFinalTxLinkRate,{0,110,300,600,1200,2400,4800,7200,9600,12000,14000,16000,19000,38000,75,450,0,57000,21000,24000,26000,28000,0,31000,33000,25333,26666,28000,29333,30666,32000,33333,34666,36000,37333,38666,40000,41333,42666,44000,45333,46666,48000,49333,50666,52000,53333,54666,56000,57333,58666,60000,61333,62666,64000})
```

mvelInfo 関数

この関数は、入力されたパラメータを karaf.log ファイルの INFO レベルに入力します。 この関数を使用して、不正確と思われる結果を返すレポートからのポーリングされた値を記録します。

Data Collector のポーリング ログは、ポーリングされた属性の値のみを表示しますが、レポートは計算の結果のみを表示します。 mvelInfo 関数を使用することにより、入力されたポーリング値を参照し、計算のどこに問題があるのかを判断できます。

構文

この関数の形式は以下のとおりです。

```
String mvelInfo (Array objects)
```

パラメータ

objects

オブジェクト配列は、Data Collector の karaf.log ファイル内の INFO レベルに記録されます。

戻り値

Null

例

以下の式は、karaf.log ファイルに cpmCPUTotal5minRev を記録します。

式：

```
mvelInfo([“cpmCPUTotal5minRev=”, cpmCPUTotal5minRev])
```

結果：

Null

結果（karaf.log）：

MVEL info: cpmCPUTotal5minRev=15

高度な例

```
mvelInfo([“cpmCPUTotal5minRev=”, cpmCPUTotal5minRev, “ cpmCPUTotal10minRev=”,
cpmCPUTotal10minRev]); cpmCPUTotal10minRev;
```

結果：

12

結果（karaf.log）：

MVEL info: cpmCPUTotal5minRev=15 cpmCPUTotal10minRev=12

mvelWarn 関数

入力されたパラメータを karaf.log ファイルの WARN レベルに入力します。この関数を使用して、不正確と思われる結果を返すレポートからのポーリングされた値を記録します。

Data Collector のポーリング ログは、ポーリングされた属性の値のみを表示しますが、レポートは計算の結果のみを表示します。 mvelWarn 関数を使用することにより、入力ポーリング値を参照し、計算のどこに問題があるのかを判断できます。

構文

この関数の形式は以下のとおりです。

```
String mvelWarn (Array objects)
```

パラメータ

objects

オブジェクト配列は、Data Collector の karaf.log ファイル内の WARN レベルに記録されます。

戻り値

Null

例

以下の式は、karaf.log ファイルに cpmCPUTotal5minRev を記録します。

式：

```
mvelWarn([“cpmCPUTotal5minRev=”, cpmCPUTotal5minRev])
```

結果：

Null

結果（karaf.log）：

MVEL warn: cpmCPUTotal5minRev=15

高度な例

```
mvelWarn([“cpmCPUTotal5minRev=”, cpmCPUTotal5minRev, “ cpmCPUTotal10minRev=”,
cpmCPUTotal10minRev]); cpmCPUTotal10minRev;
```

結果：

12

結果（karaf.log）：

MVEL warn: cpmCPUTotal5minRev=15 cpmCPUTotal10minRev=12

mvelError 関数

この関数は、入力されたパラメータを karaf.log ファイルの ERROR レベルに入力します。この関数を使用して、不正確と思われる結果を返すレポートからのポーリングされた値を記録します。

Data Collector のポーリング ログは、ポーリングされた属性の値のみを表示しますが、レポートは計算の結果のみを表示します。 mvelError 関数を使用することにより、入力ポーリング値を参照し、計算のどこに問題があるのかを判断できます。

構文

この関数の形式は以下のとおりです。

```
String mvelError (Array objects)
```

パラメータ

objects

オブジェクト配列は、Data Collector の karaf.log ファイル内の ERROR レベルに記録されます。

戻り値

Null

例

以下の式は、karaf.log ファイルに cpmCPUTotal5minRev を記録します。

式：

```
mvelError(["cpmCPUTotal5minRev=", cpmCPUTotal5minRev])
```

結果：

Null

結果（karaf.log）：

MVEL error: cpmCPUTotal5minRev=15

高度な例

```
mvelError(["cpmCPUTotal5minRev=", cpmCPUTotal5minRev, " cpmCPUTotal10minRev=",
cpmCPUTotal10minRev]); cpmCPUTotal10minRev;
```

結果：

12

結果（karaf.log）：

MVEL error: cpmCPUTotal5minRev=15 cpmCPUTotal10minRev=12

mvelDebug 関数

この関数は、入力されたパラメータを karaf.log ファイルの DEBUG レベルに入力します。この関数を使用して、不正確と思われる結果を返すレポートからのポーリングされた値を記録します。

Data Collector のポーリング ログは、ポーリングされた属性の値のみを表示しますが、レポートは計算の結果のみを表示します。 mvelDebug 関数を使用することにより、入力されたポーリング値を参照し、計算のどこに問題があるのかを判断できます。

構文

この関数の形式は以下のとおりです。

```
String mvelDebug (Array objects)
```

パラメータ

objects

オブジェクト配列は、Data Collector の karaf.log ファイル内の DEBUG レベルに記録されます。

戻り値

Null

例

以下の式は、karaf.log ファイルに cpmCPUTotal5minRev を記録します。

式：

```
mvelDebug([“cpmCPUTotal5minRev=”, cpmCPUTotal5minRev])
```

結果：

Null

結果（karaf.log）：

MVEL debug: cpmCPUTotal5minRev=15

高度な例

```
mvelDebug([“cpmCPUTotal5minRev=”, cpmCPUTotal5minRev, “ cpmCPUTotal10minRev=”,
cpmCPUTotal10minRev]); cpmCPUTotal10minRev;
```

結果：

12

結果（karaf.log）：

MVEL debug: cpmCPUTotal5minRev=15 cpmCPUTotal10minRev=12

mvelDebug 関数

この関数は、入力されたパラメータを karaf.log ファイルの TRACE レベルに入力します。 この関数を使用して、不正確と思われる結果を返すレポートからのポーリングされた値を記録します。

Data Collector のポーリング ログは、ポーリングされた属性の値のみを表示しますが、レポートは計算の結果のみを表示します。 mvelTrace 関数を使用することにより、入力されたポーリング値を参照し、計算のどこに問題があるのかを判断できます。

構文

この関数の形式は以下のとおりです。

```
String mvelTrace (Array objects)
```

パラメータ

objects

オブジェクト配列は、Data Collector の karaf.log ファイル内の TRACE レベルに記録されます。

戻り値

Null

例

以下の式は、karaf.log ファイルに cpmCPUTotal5minRev を記録します。

式：

```
mvelTrace(["cpmCPUTotal5minRev=", cpmCPUTotal5minRev])
```

結果：

Null

結果（karaf.log）：

MVEL trace: cpmCPUTotal5minRev=15

高度な例

```
mvelTrace([“cpmCPUTotal5minRev=”, cpmCPUTotal5minRev, “ cpmCPUTotal10minRev=”,
cpmCPUTotal10minRev]); cpmCPUTotal10minRev;
```

結果：

12

結果（karaf.log）：

MVEL trace: cpmCPUTotal5minRev=15 cpmCPUTotal10minRev=12

snmpCounter64 関数

この関数は 2 つの 32 ビットの数値を評価し、64 ビット表現を含む値を返します。この関数はデバイスの認定に使用します。hiVal は 32 ビット左にシフトされて lowVal が追加され、結果は 64 ビットの変数で返されます。

構文

この関数の形式は以下のとおりです。

```
Object snmpCounter64 (Long hiVal, Long lowVal)
```

パラメータ

hiVal

高位ビットを表す 32 ビットの数値です。

lowVal

低位ビットを表す 32 ビットの数値です。

戻り値

2 つの 32 ビットの数値の 64 ビット表現を返すか、またはどちらの 32 ビットの数値の入力も null の場合に「null」を返します。

例

以下の式は、hiVal が 88 で lowVal が 558 の場合、以下のような結果になります。

式：

```
snmpCounter64 (hiVal, lowVal)
```

結果：

```
377957122606
```

高度な例

「Cisco CBQos ClassMap」ベンダー認定から、以下の式が取得されます。この認定には、snmpMax の例が多数含まれています。

```
PrePolicyPackets=snmpMax(0,snmpCounter64(cbQosCMPrePolicyPktOverflow,cbQosCMPrePolicyPkt))
```

snmpGetUpSinceTime 関数

この関数は、システムがオンにされた時間を現在のエポック以降の秒数に基づいて示します。

構文

この関数の形式は以下のとおりです。

```
snmpGetUpSinceTime(Long upTime)
```

パラメータ

upTime

現在のエポックの開始からの秒数。次の OID からシステム稼働時間を取得できます： 1.3.6.1.2.1.1.3.0. 渡す前に 100 分の 1 秒に変換します。

戻り値

デバイスの電源がオンにされた時間を現在のエポック以降の合計秒数で示します。

snmpMax 関数

この関数は 2 つの 64 ビット値よりも大きい値を返します。 この関数はデバイスの認定に使用します。

構文

この関数の形式は以下のとおりです。

```
Object snmpMax(BigInteger val1, BigInteger val2)
```

パラメータ

val1

1 番目の 64 ビットの BigInteger 値です。

val2

2 番目の 64 ビットの BigInteger 値です。

戻り値

渡された 2 つの BigInteger 値のうち最大値を返すか、またはどちらの BigInteger 値の入力も null の場合に「null」を返します。

例

以下の式は、val1 が 2^32 で val2 が 10 の場合、以下のような結果になります。

式：

```
snmpMax (val1, val2)
```

結果：

```
2^32
```

以下の式は、val1 が 5864 で val2 が 134556890 の場合、以下のような結果になります。

結果：

```
134556890
```

高度な例

「Cisco CBQos ClassMap」ベンダー認定から、以下の式が取得されます。この認定には、snmpMax の例が多数含まれています。

```
PrePolicyPackets=snmpMax(0,snmpCounter64(cbQosCMPrePolicyPktOverflow,cbQosCMPrePolicyPkt))
```

snmpObjectIDToASCIIString 関数

この関数は SNMP OID 値をその文字列に変換します。先頭または末尾のスペースは削除されます。

構文

この関数の形式は以下のとおりです。

```
snmpObjectIDToASCIIString( Object Id oid )
```

パラメータ

oid

文字列に変換するオブジェクト ID。

snmpOIDParser 関数

この関数は objectID（OID）の値を使用し、startIndex 値および endIndex 値を基に OID のサブセットを解析します。インデックスは 1 から始まります。endIndex が -1 であれば、OID の最後に移動します。 この関数はデバイスの認定に使用します。

構文

この関数の形式は以下のとおりです。

```
ObjectID  snmpOIDParser( ObjectID OID, Integer startIndex, Integer endIndex )
```

パラメータ

OID

解析するオブジェクト ID（OID）の値です。

startIndex

解析を開始するインデックスの整数値。

endIndex

解析を停止するインデックスの整数値。

戻り値

解析されたサブセットのオブジェクト ID（OID）を返します。

例

以下の式は、OID 値が 1.2.3.4.5.6.7.8.9.10、startIndex 値が 1、および endIndex 値が 5 の場合、以下のような結果になります。

式：

```
snmpOIDParser(oid, startIndex, endIndex )
```

結果：

```
1.2.3.4.5
```

同じ式で、OID 値が 1.2.3.4.5.6.7.8.9.10、startIndex 値が 6、および endIndex 値が -1 の場合は、以下のような結果になります。

結果：

```
6.7.8.9.10
```

高度な例

「Cisco CBQos ClassMap」ベンダー認定から、以下の式が取得されます。

```
ItemUniqueIDs=snmpOIDParser(cbQosConfigIndex, 2, 2)
```

snmpOctetStringFloat 関数

この機能は SNMP オクテット文字列を浮動小数点値に変換します。この関数はデバイスの認定に使用します。SNMP オクテット文字列は 7 ビットの ASCII 文字列です。

構文

この関数の形式は以下のとおりです。

```
Object snmpOctetStringFloat(byte[] octetString)
```

パラメータ

octetString

SNMP オクテット文字列。

戻り値

変換された文字列の値を返すか、関数が文字列を変換できない場合は「null」を返します。

例

以下の式は、octetString が {0x33, 0x33, 0x2E, 0x33, 0x33} の場合、以下のような結果になります。

式：

```
snmpOctetStringFloat (octetString)
```

結果：

```
33.33
```

同じ式で、octetString が {0x36, 0x36, 0x36} の場合、以下のような結果になります。

結果：

```
666.0
```

snmpProtectedDiv 関数

この関数は、2 つの Double 値を除算し、結果を Double として返します。被除数または除数が NULL または 0.0 である場合、戻り値は 0.0 です。 この関数を使用して、NULL または 0 での除算が行われないように式を保護します。 Data Repository には、ポーリングが失敗した場合などに NULL またはゼロ値が含まれる可能性があります。この場合、この関数を使用して、ゼロによる除算の例外を回避します。

構文

この関数の形式は以下のとおりです。

```
Double snmpProtectedDiv(Double val1, Double val2)
```

パラメータ

val1

被除数です。 これは val2 で除算される Double 値（浮動小数点数）です。 （Double は Java のデータ型です。）

val2

除数です。これは Double 値（浮動小数点数）です。（Double は Java のデータ型です。）

戻り値

除算の結果として Double、被除数または除数が NULL または 0.0 の場合は 0.0 を返します（*Double* は Java のデータ型です）。

例

以下の式は、val1 が 7.2 で val2 が 2 の場合、以下のような結果になります。

式：

```
snmpProtectedDiv(val1, val2)
```

結果：

```
3.6
```

以下の式は、val1 が 7.2 で val2 が NULL または 0.0 の場合、以下のような結果になります。

結果：

```
0.0
```

高度な例

ベンダー認定から、以下の式が取得されます。

```
Utilization=snmpProtectedDiv((cpuStatsUser + cpuStatsSys),(cpuStatsUser +
cpuStatsSys + (isdef(cpuStatsWait)?cpuStatsWait:0) + cpuStatsIdle))*100
```

snmpRound 関数

この関数は、数値を近似の整数値に四捨五入します。

構文

この関数の形式は以下のとおりです。

```
Long snmpRound(Double dNumber)
```

パラメータ

dNumber

四捨五入される Double 値（浮動小数点数）（*Double* は Java のデータ型です）。

戻り値

dNumber で提供されている値に最も近い整数値である Long 値を返します（*Long* は Java のデータ型です）。

例

以下の式は、dNumber が 3.5 の場合、以下のような結果になります。

式：

```
snmpRound(dNumber)
```

結果：

```
4
```

同じ式で、dNumber が 3.4 の場合、以下のような結果になります。

結果：

3

高度な例

「Cisco IPSLA Jitter Precision Statistics」ベンダー認定から、以下の式が取得されます。

```
PathAvailability=snmpRound(rttMonJitterStatsNumOfRTT / (rttMonJitterStatsNumOfRTT
+ rttMonJitterStatsPacketLossSD + rttMonJitterStatsPacketLossDS +
rttMonJitterStatsPacketOutOfSequence + rttMonJitterStatsPacketMIA +
rttMonJitterStatsPacketLateArrival + rttMonJitterStatsError +
rttMonJitterStatsBusies + 1/100) * 100)
```

snmpStringParser 関数

この関数は内部使用のみの目的で用意されました。これは、CA Application Insight Module （AIM）から受信された IP アドレスを解析します。 CA は、この関数を内部クラスでのみテストしているため、別のタイプのクラスではサポートされていない可能性があります。

構文

この関数の形式は以下のとおりです。

```
snmpStringParser(Delimiter, Type to convert to, String to parse 1, String to parse
2)
```

パラメータ

Type to convert to

提供された文字列を解析するクラスのタイプ。

Strings to parse 1、2

解析する IP アドレス。 2 つの文字列により、アドレスを IPv4 および IPv6 形式で提供できます。

戻り値

変換された文字列の値を返すか、または関数が文字列を変換できない場合は「null」を返します。

snmpSvcs 関数

この関数は、エージェントの sysObjectOID、sysService、および ipForwarding MIB 変数から値を取得して、SNMP エージェントがサポートするサービスを決定します。 たとえば、SNMP MIB RFC 1213 の定義に従って、ルータ/スイッチ/リピータ/ホストをサポートされるサービスにすることができます。

カスタム デバイス タイプはシステムのデバイス タイプよりも優先順位が高いため、関数からの戻り値は以下のように評価されます。

- sysObjectID OID が DeviceTypes ファイルでマップされる場合、返されるサービスはファイルから取得されます。
- sysObjectID OID が DeviceTypes ファイルでマップされない場合、サポートされるサービスを返すために sysServices および ipForwarding が使用されます。

構文

この関数の形式は以下のとおりです。

```
DeviceService[] snmpSvcs(ObjectID sysObjectID, Integer sysServices, Integer
ipForwarding)
```

パラメータ

sysObjectID

解析するオブジェクト ID 値です。

sysServices

各ビットの整数はスイッチ/リピータ/ホストなどの異なるサービスを表します。

ipForwarding

このエンティティが、データグラムの転送に関わる IP ゲートウェイまたは IP ホストとして機能する整数値。このエンティティは転送されたデータグラムを受信しますが、転送されたデータグラムはこのエンティティにアドレス指定されません。

戻り値

以下の 1 つ以上のデバイス サービスのリストを返します。

- ROUTER
- REPEATER
- SWITCH
- HOST
- UNKNOWN_TYPE

例

以下の式は、sysServices 値が 8、ipForwarding 値が 0、および sysObjectID が DeviceTypes ファイルで見つからない場合、以下の結果になります。

式：

```
snmpSvcs(sysObjectOID,sysServices,ipForwarding)
```

結果：

```
DeviceService[HOST]
```

高度な例

「System Statistics」ベンダー認定から、以下の式が取得されます。

```
Services=snmpSvcs(sysObjectID,isdef(sysServices)?sysServices:0,isdef(ipForwarding
)?ipForwarding:0)
```

storePortReconfig 関数

この関数は、ifNumber、ifTableLastChange、ifStackLastChange の値を表す XML を含む文字列を返します。 XML を使用し、必要に応じてデバイス変更の追跡およびデバイス上のインターフェースの再ディスカバリを行うことができます。

構文

この関数の形式は以下のとおりです。

```
String storePortReconfig ( Integer ifNumber, Long ifTableLastChange, Long
ifStackLastChange )
```

パラメータ

ifNumber

デバイス上のポートの数。

ifTableLastChange

最新のポート テーブルの日時はミリ秒単位で変化し、1970 年 1 月 1 日 GMT の開始日時から計算されます。

ifStackLastChange

最新のポート スタックの日時はミリ秒単位で変化し、1970 年 1 月 1 日 GMT の開始日時から計算されます。

戻り値

以下の例に示す形式の XML を含む文字列を返します。

例

以下の式の結果は、ifNumber 5、ifTableLastChange 123456、および ifStackLastChange 234567 になります。

式：

```
storePortReconfig ( ifNumber, ifTableLastChange, ifStackLastChange )
```

結果：

```
<ReconfigData>
     <ReconfigValue name="ifNumber" value="5"/>
     <ReconfigValue name="ifTableLastChange" value="123456"/>
     <ReconfigValue name="ifStackLastChange" value="234567"/>
</ReconfigData>
```

グローバル変数

Data Aggregator は、以下のグローバル変数をサポートしています。

_ rspDuration

現在のポーリング サイクルの期間（秒単位）を含む Long（Java データ型）。

注: 「System Statistics」ベンダー認定には _rspDuration の使用例が含まれています。

_rspTimestamp

1970 年 1 月 1 日（GMT）に開始された、現在のポーリング サイクル（ミリ秒）のタイムスタンプが含まれている Long （Java データ型）。

付録 B: トラブルシューティング

このセクションには、以下のトピックが含まれています。

トラブルシューティング : ベンダー認定の作成に失敗

症状:

ベンダー認定を作成しようとしたら、失敗したというエラー メッセージが表示されました。

解決方法:

Data Aggregator のインストール ディレクトリ内の karaf ログ ファイルを開き、以下の手順に従います。

1. MIB 名の文字列または選択したメトリック ファミリの名前を検索します。
2. 例外のスタック トレースを確認し、CertManagerException およびエラーの理由を探します。 エラーの理由は例外の後に示されます。

   例： 以下に示すように、式パーサが ++ 以降のトークンを除外しませんでした。

   ```
   Caused by: com.ca.im.dm.certmgr.interfaces.CertManagerException: Tech Cert:
   {http://im.ca.com/normalizer}NormalizedCPUInfo, Unable to compile expression:
   [Error: expected end of statement but encountered: e]
   [Near : {... stemID ++ extremeSystemBoardID ....}]
   ```

3. 提供された理由に基づいてエラーを修正します。以下の要件を満たしているかどうかを確認します。
   - 式グループには、スカラーとテーブル エントリの混合を含めることはできません。
   - 式には正しい構文を含める必要があります。

- メトリック ファミリ変数に対して、少なくとも 1 つの式を定義します。
- 少なくとも 2 つのメトリック ファミリ変数が定義されます。具体的には、名前とインデックスが必須です（スカラーのみのメトリックを除く）。
- 式に使用するベンダー認定変数は、選択された MIB テーブルのもの（ユーザ インターフェースで有効）である必要があります。

トラブルシューティング：メトリック ファミリがサポートされていません

症状：

デバイスのコレクションでメトリック ファミリをポーリングするために、監視プロファイルを作成しました。しかし、［ポーリングされるメトリック ファミリ］テーブルで、これらのメトリック ファミリの 1 つに「サポートされていません」というステータスが表示されます。

解決方法：

問題を解決するには、以下の手順に従います。

1. ポーリングされたデバイスが SNMP クエリに応答することを確認します。
2. サポートされていないメトリック ファミリに移動して、それをクリックします。
3. ベンダー認定でメトリック ファミリがサポートされていることを確認します。 ベンダー認定が定義されていない場合は、カスタム ベンダー認定を作成します。
4. デバイスで、すべてのキー ベンダー認定属性がサポートされることを確認します。すべてのキー ベンダー認定属性がサポートされている場合は、デバイスに移動し、カスタム ベンダー認定を追加したメトリック ファミリを選択し、［メトリック ファミリの更新］をクリックします。

   デバイス設定が更新されます。

トラブルシューティング : メトリック ファミリが不完全

症状:

カスタム メトリック ファミリを正常にインポートしましたが、その後、不完全なメトリック定義が見つかりました。 たとえば、<Name> プロパティの最大長は 32 文字です。 この制限を超過すると、同期の問題が発生する場合があります。

解決方法:

以下の手順に従い、カスタム メトリック ファミリを慎重に削除します。

1. IMDataAggregator/apache-karaf-2.3.0/deploy ディレクトリを見つけます。
2. メトリック ファミリに対して作成および展開された XML ファイルを削除します。 それらの名前を以下に示します。
   - im.ca.com-normalizer-<technology>.xml
   - im.ca.com-inventory-<technology>.xml

   該当する場合、ベンダー認定に対して作成されたファイルも削除します。
   - im.ca.com-certifications-snmp-<vendor>.xml
3. 以下のコマンドを実行して、Data Aggregator を再起動します。

   ```
   service dadaemon restart
   ```

   Data Aggregator が再起動されたら、CA Performance Center に以前インポートしたメトリック ファミリとベンダー認定が表示されないことを確認します。 また、そのカスタム認定に関連して以前検出されたコンポーネントがすべて削除されていることを確認します。
4. CA Performance Center で［管理］ - ［データ ソース］をクリックします。
5. ［Data Aggregator］を選択し、［再同期］ボタンをクリックします。

   残っているメトリック ファミリのコンポーネントが Data Aggregator と CA Performance Center 間で同期されます。
6. カスタム メトリック ファミリの XML ファイルを編集および修正します。
7. 修正したメトリック ファミリの XML ファイルをインポートします。

トラブルシューティング : ベンダー認定式にエラーが存在する

症状:

MVEL コンパイラは、不正な式に対して評価例外（エラー）を生成しない場合があります。 この状況は、一部の構文エラーで発生します。これには、かっこの欠落や複数のアステリスクがある場合などが含まれます。

正しくない式がコンパイルされ、式の評価が適切な変数で実行されるまで、エラー状態は表示されません。 式のターゲットであるデータベース列には値が入力されません。

解決方法:

以下の手順に従って、ExpressionEvaluator のデバッグ ログ記録をオンにします。

1. IMDataAggregator/apache-karaf-2.3.0/etc ディレクトリを見つけます。
2. org.ops4j.pax.logging.cfg ファイルを開き、以下のエントリを作成します。

   ```
   log4j.logger.com.ca.im.core.expressionevaluator=DEBUG
   ```

3. 以下のコマンドを実行して、Data Aggregator を再起動します。

   ```
   service dadaemon restart
   ```

4. IMDataAggregator/apache-karaf-2.3.0/data/log ディレクトリの karaf.log ファイルで評価例外を検索します。