CA Spectrum®

Modeling Gateway Toolkit ガイド

リリース 9.3

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 Spectrum®（CA Spectrum）
- CA Spectrum® Modeling Gateway Toolkit （Modeling Gateway）
- CA CMDB

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: Modeling Gateway の概要

このセクションには、以下のトピックが含まれています。

Modeling Gateway の前提条件

CA Spectrum Modeling Gateway ツールキットを使用する前に、以下のことを確認してください。

- CA Spectrum の操作経験が豊富である。
- 「コンセプト ガイド」を読んでいる。
- XML について実践的な知識を持っている。
- 文書型定義（DTD）の概念を理解している。
- インポートするネットワーク トポロジについて詳細に理解している。
- UNIX または Windows を使用して、ファイル システムのナビゲート、ファイルのコピーと削除、およびテキスト ファイルの作成と削除を行うことができる。

Modeling Gateway Toolkit について

CA Spectrum Modeling Gateway Toolkit を使用して、CA Spectrum との間でネットワーク トポロジ データをインポートまたはエクスポートできます。このツールキットには、XML エレメントと属性を定義する文書型定義（DTD）が含まれています。 また、CA Spectrum 構文、およびインポートまたはエクスポートする情報を定義するリソース ファイルも含まれています。

トポロジのインポートでは、DTD エレメントを使用すると、デバイス、ポート、およびネットワークの接続を説明する XML ファイルを作成できます。この XML ファイルでは、CA Spectrum での新しいトポロジ データの作成、既存のデータの更新、または正しくなくなったデータの破棄を行うことができます。 さらに、XML 構文で使用されるエレメントと属性は、ほとんどの統合のニーズに合わせて拡張およびカスタマイズできます。

また、ツールキットでは、カンマ区切りの ASCII テキスト ファイルを使用して、フレーム リレー接続または ATM 接続をインポートできます。 前に説明した XML 機能を使用して、この接続情報をインポートすることもできます。

ネットワーク トポロジ データが CA Spectrum に存在していれば、手動またはディスカバリで作成される他のモデルのように、これらのデバイスを管理できます。 インポートの結果は、各インポートに関する診断情報と共に確認できます。

Modeling Gateway Toolkit では、XML ファイルを使用して、CA Spectrum からトポロジ情報および設定をエクスポートすることもできます。 この情報は、Modeling Gateway を介して、指定された SpectroSERVER にインポートできます。

動的なネットワーク トポロジ情報による CA Spectrum の継続的な生成は、以前は困難なタスクでした。 ディスカバリと手動モデリングは、変化する環境で必要な定期的な更新に適していません。 また、ディスカバリを使用したモデリング接続も、さまざまな物理インフラストラクチャ（これらの環境内にあるものなど）では課題となることもあります。

- ケーブル MSO （マルチサービス オペレータ）
- ATM （非同期転送モード）
- フレーム リレー

CA Spectrum Modeling Gateway は、これらの問題に対する効果的なソリューションです。

アーキテクチャのインポート

インポートでは、インポートの統合プロセス中に、サードパーティ データベースからデータを受け取り、入力ファイルを作成します。 この入力ファイルは、コンテンツに応じて、XML ファイルまたはカンマ区切りの ASCII ファイルになります。 XML 入力ファイルには最も広い範囲のインポート オプションがあるので、このガイドで重点的に説明します。 カンマ区切りのファイルでは、フレーム リレーと ATM 回路用の接続を作成できます。

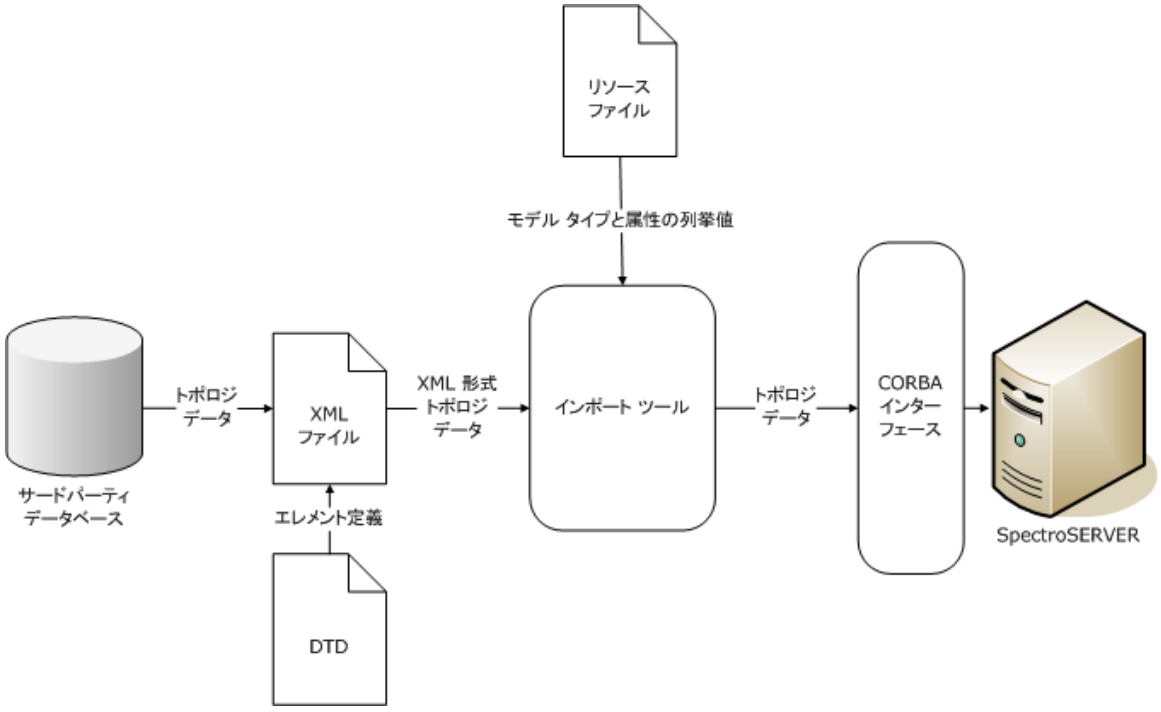
XML 入力ファイルを作成するときは、提供された文書型定義（DTD）ファイルおよび .modelinggatewayresource.xml ファイルを操作します。 DTD は XML エレメント、属性、およびそれらの関連する構文ルールを定義します。 .modelinggatewayresource.xml ファイルでは、使用可能な CA Spectrum モデル タイプおよび属性が示されます。 このファイルは、CA Spectrum モデル タイプ名および属性名を、CA Spectrum がそのモデル タイプまたは属性に使用する一意の 16 進数の識別子に関連付けます。.modelinggatewayresource.xml ファイルは、特定の統合ニーズに合わせてカスタマイズできます。

最初の入力ファイルを作成すると、そのファイルは同じタイプの入力を表す複数のデータ セット用のテンプレートとして動作できます。たとえば、インポートするデバイス用の XML ファイルを作成し、デバイスに固有のトポロジ データを代入することで、このファイルを繰り返し使用できます。

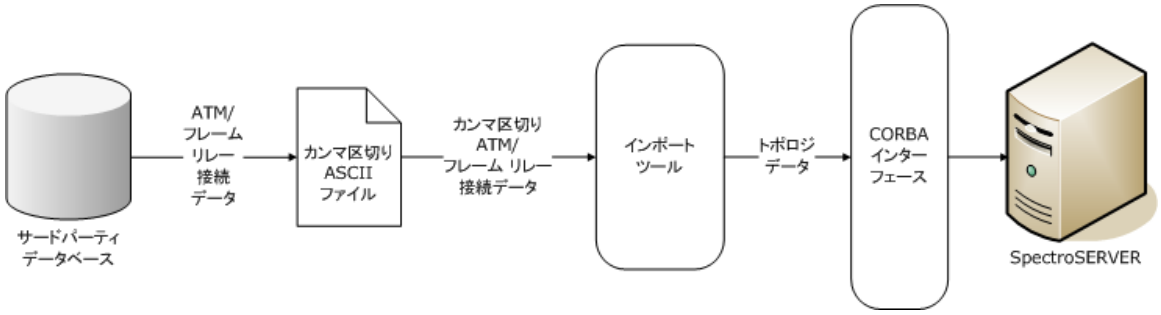
modelinggateway ツールは、入力ファイルからネットワーク トポロジ情報を読み取り、SpectroSERVER データベースにデータを送信するコマンド ライン ユーティリティです。 このインポート ツールは、XML ファイルからのデータを使用して、接続、デバイス、およびコンテナ モデルを作成、破棄、および更新できます。 また、CA Spectrum からデータをエクスポートするためにも、このツールを使用できます。

CA Spectrum Modeling Gateway は、各データベース インポートの安全性および正確性を検証するためのメカニズムも提供します。 たとえば、行われた作成、削除、関連付け、および更新のレコードが含まれる監査証跡を維持できます。 OneClick 内で、インポートに関する情報を表示できます。 CA Spectrum は、任意のタイプの重大なエラーがインポート プロセス中に発生した場合、イベントを生成することによりエラーを報告します。 すべてのエラーと考えられる原因は、エラー ログ ファイルに記録されます。 また、ユーザが問題または不正確なソースを見つけるのを支援できるデバッグ ログをオンにできます。

以下の図は、データのインポートで Modeling Gateway が XML ファイルを使用する方法を示しています。 データはサードパーティ データベースから流れ、構文のために DTD と .modelinggatewayresource.xml を使用して XML ファイルでフォーマットされます。 次に、インポート ツールが XML ファイルを解釈し、CA Spectrum CORBA インターフェースを介して CA Spectrum へデータを送信します。



以下の図は、フレーム リレーおよび ATM 接続情報をインポートするために、Modeling Gateway がカンマ区切りの ASCII テキスト ファイルを使用する方法を示しています。



アーキテクチャのエクスポート

エクスポートの場合、Modeling Gateway は、SpectroSERVER をリソースとして使用し、トポロジと設定のデータを XML ファイルにエクスポートできます。次に、この XML ファイルをサードパーティ ツールに統合するか、別の SpectroSERVER に再インポートできます。 たとえば、ファイルは CA Spectrum パーティションまたはランドスケープ ハンドルの変更のために再インポートできます。

第 2 章: CA Spectrum へのトポロジ データのインポート

このセクションには、以下のトピックが含まれています。

Modeling Gateway でインポートを行う方法

Modeling Gateway を使用して、サードパーティのトポロジ データを CA Spectrum にインポートできます。 まず、サードパーティ データベースからトポロジ データを抽出し、入力ファイルのデータをフォーマットします。

Modeling Gateway を使用して、サードパーティのトポロジ データを CA Spectrum にインポートするには、以下のタスクを実行します。

1. トポロジ データを抽出します。

   サードパーティ データベースからネットワーク トポロジ データを抽出します。データベース システムはそれぞれ異なるので、この手順を完了する方法については、操作しているデータベースのドキュメントを参照してください。

2. 入力ファイルのデータをフォーマットします (P. 16)。

   インポート ツール用にデータをフォーマットするには、XML ファイルまたはカンマ区切りの入力ファイルを作成します。

3. （オプション） modelinggateway ツールを移動します。

   注: 別のサーバで modelinggateway ツールを実行するには、そのサーバに modelinggateway ツールとそのすべてのサポート ファイルを移動します。 詳細については、「分散 SpectroSERVER 管理者ガイド」を参照してください。

4. modelinggateway ツールを実行します (P. 37)。

   入力ファイルが作成されたら、インポート ツールを使用してデータを CA Spectrum に送信します。

5. （オプション）カンマ区切りファイルをインポートします (P. 39)。

   インポート ツールを使用せずに、OneClick インターフェースからフレーム リレー接続データと ATM 接続データをインポートできます。

6. インポート情報の表示 (P. 40)

   OneClick のインポートの進捗状況および結果を確認します。

注: CA Spectrum の 1 つのバージョンから別のバージョンにモデルをマイグレートするために Modeling Gateway を使用しないでください。 エンティティを識別し、モデリングするために使用される方法は CA Spectrum のバージョンによって異なる場合があります。 そのため、Modeling Gateway を使用して、別のバージョンの CA Spectrum からエクスポートされる XML ファイルをインポートしないでください。

入力ファイルのデータのフォーマット

Modeling Gateway を使用したデータのインポートには、XML ファイルとカンマ区切りのファイルという 2 種類の入力ファイルが使用されます。XML 入力ファイルは、モデルと接続を作成または破棄し、属性値と接続情報を更新するために使用できます。 Modeling Gateway で提供される DTD の構文により、XML 入力ファイルで使用するエレメントが定義されます。 カンマ区切りファイルは、ATM 接続とフレーム リレー接続を作成するためにのみ使用できます。 以下のセクションでは、それぞれのタイプの入力ファイルを作成する方法の概要について説明します。

XML 入力ファイル

XML 入力ファイルを作成するエレメントについて理解するには、ネットワーク インフラストラクチャをモデリングするために CA Spectrum が使用するプロセスを把握する必要があります。 以下のセクションでは、このプロセスの概要と、XML 入力ファイルで使用される XML エレメントにプロセスがどのように適用されるかについて説明します。 これらの概念について十分に理解している場合は、このセクションをスキップし、「XML 入力ファイルの構文 (P. 20)」に進んでください。

詳細情報:

XML 入力ファイルの構文 (P. 20)

階層ビュー

CA Spectrum 内のビューは、データを表示または操作できるように編成します。階層ビューはネットワーク データの編成方法を表します。XML ファイル内のネットワーク データを編成する場合は、各階層ビューを表すエレメントから選択します。階層ビューには、トポロジ ビューと場所ビューの 2 つのタイプがあります。

トポロジ ビュー

トポロジ ビューは、実際にはネットワーキング コンポーネントを抽象化したものです。このビューを操作する場合、ネットワークの物理コンポーネントまたは論理コンポーネントを表し、これらのコンポーネントを論理的な接続に結び付けて考えます。 また、ポート レベルまたはデバイス レベルでデバイスがどのように接続されるかを示すパイプを視覚的に使用して、接続を表すこともできます。 OneClick では、このビューはユニバース トポロジとして表示されます。

注: OneClick で使用できるトポロジ ビューの詳細については、「IT インフラストラクチャのモデリング/管理 - 管理者ガイド」を参照してください。

場所ビュー

場所ビューでは、物理的な場所別にネットワーク データが整理されます。このビューを使用すると、地理的な観点からネットワークを示すことができます。 グローバルなオフィスから開始し、 オフィスが位置する各領域内の各ビルディングの各階にあるワイヤリング クローゼットに直行することができます。 OneClick では、このビューはワールド トポロジとして表示されます。

注: OneClick で使用できるトポロジ ビューの詳細については、「IT インフラストラクチャのモデリング/管理 - 管理者ガイド」を参照してください。

モデルとモデル タイプ

CA Spectrum では、多数のモデル タイプが事前定義されています。 モデル タイプが特定のネットワーク エンティティを表すために CA Spectrum インターフェースでインスタンス化される場合、それらをモデルと呼びます。 モデル タイプの 2 つのメジャー カテゴリは以下のとおりです。

- インテリジェントなモデル タイプ
- コンテナ モデル タイプ

インテリジェントなモデル タイプは、ネットワークで動作する実際のデバイスを表すためにインスタンス化できます。 このモデルは IP アドレスと MAC アドレスを持ち、CA Spectrum は SNMP を使用して、これらのデバイスと直接通信できます。コンテナ モデル タイプは、主にモデルをグループ化するために使用されるモデルにインスタンス化されます。

モデルは、使用されている階層ビューのタイプに基づいてグループ化できます。 たとえば、トポロジ ビューを使用して、ネットワークのセグメントにある特定のデバイスをグループ化する LAN モデルを作成できます。 または、場所ビューを使用して、ビルディングの 1 つの部屋にあるデバイスをグループ化する Room モデルを作成できます。

コンテナ モデルには、特定のモデル タイプに応じて、他のコンテナ モデル、インテリジェントなモデル、または両方を含めることができます。 たとえば、ネットワーク コンテナ モデルには、ルータを表すためにインテリジェントなモデルを含めることができます。 また、ネットワーク コンテナ モデルには、一連の IP アドレスを表すために LAN コンテナ モデルを含めることもできます。 一方、Building モデルには、Floor、Section、Room などのコンテナ モデルのみを含めることができます。

DTD のエレメントでは、階層ビューおよびそれぞれの任意のコンテナ モデル タイプを使用して、ネットワーク トポロジを示すことができます。 また、インスタンス化可能な任意のインテリジェントなモデル タイプを使用することもできます。 インテリジェントなモデル タイプは、使用される階層ビューのタイプに依存しません。

注: 必要に応じて、モデルを識別するためにモデル タイプではなくモデル ハンドルを指定できます。 モデル ハンドルを指定する場合、Modeling Gateway は他のモデル識別子を無視し、モデルを識別するためにモデル ハンドルのみを使用します。

CA Spectrum ナレッジ ベースで定義されているすべてのモデル タイプを、実際に OneClick でモデルを作成するために使用することはできません。一部は、他のモデル タイプの派生元となるベース モデル タイプとして使用されます。

注: 詳細については、「コンセプト ガイド」を参照してください。

モデリング方法

CA Spectrum のデバイスをモデリングするためには、2 つの方法が使用されます。 最初の方法では、デバイスの IP アドレスまたは DNS 名を使用します。 CA Spectrum はこの情報を使用してデバイスに接続し、デバイスの機能を最もよく表しているモデル タイプを使用して、モデルを作成します。

2 番目の方法では、デバイス モデル作成用のモデル タイプを提供します。この場合でも、CA Spectrum がデバイスと通信できるように、IP アドレスまたは DNS 名を指定する必要があります。 ただし、選択されたモデル タイプまたはモデル ハンドルは、デバイス機能の CA Spectrum 評価にかかわらずインスタンス化されます。

コンテナ モデルなどの非デバイス モデルを作成するには、インポートでモデル タイプおよびモデル名を指定する必要があります。

CA Spectrum 属性

各モデル タイプには関連する属性のセットがあります。 各属性は、何らかの方法でモデル タイプについて説明します。 インスタンス化されたモデル内の属性は、モデルが表すデバイスを反映し、モデルの現在の状態について説明する値を受け取ります。 たとえば、モデル タイプ Host_Sun には IPAddress 属性があります。タイプが Host_Sun のモデルがインスタンス化される場合、この属性の値はモデルが表すデバイスの IP アドレスを反映します。

XML 構文は、期間属性も使用します。 XML 属性は、エレメントに関するより多くの情報を示します。 CA Spectrum Modeling Gateway XML 構文で、一部の XML 属性は CA Spectrum 属性に値を提供するために使用されます。

注: CA Spectrum が定義する属性は CA Spectrum 属性と呼ばれ、一般的な XML 属性は単に属性と呼ばれます。

XML 入力ファイルの構文

XML ファイルを生成するときは、DTD ファイルで定義されている構文ルールを使用します。 以下のセクションでは、各 DTD エレメントの機能の概要について説明します。

以下の説明および例が各エレメントのすべての属性を網羅するとは限りません。 各エレメントおよびその属性の詳細な説明については、「文書型定義エレメント (P. 51)」を参照してください。

ルート エレメント

DTD で定義されているエレメントは、CA Spectrum 内のネットワーク表現と類似した階層構造に存在します。 各 XML インポート ファイルと共に使用される必要があるルート エレメントは Import エレメントです。 XML 構文ルールは、ルート エレメントが最も外側のエレメントであり、XML ファイルの先頭および終わりを示すことを指定します。 そのため、Import エレメントは、ドキュメントで使用されているその他の XML エレメントを囲みます。

モデル指向型エレメント

モデル指向型エレメントは、ネットワークの物理コンポーネントまたは論理コンポーネントを定義します。 それらは、ネットワーク エレメントの論理的なグループ化方法を定義するモデルを作成するために使用される、コンテナ タイプ エレメントです。 グループ化は、エレメントが存在する CA Spectrum 階層ビューのタイプに基づきます。 これらの各コンテナ タイプ エレメントは、特定の階層ビューの 1 つに存在できます。

- Topology_Container
- Location_Container
- Device
- Schedule
- Port
- Connection
- GenericView_Container

Topology_Container

Topology_Container エレメントは、物理トポロジまたは論理トポロジに従って他のモデルをグループ化するモデルを作成します。
Topology_Container エレメントはコンテナ モデルを作成するので、使用する特定のコンテナを識別するには model_type 属性またはモデル ハンドルを使用します。 可能な model_type 値は DTD に列挙されます。 LAN は Topology_Container model_type 値の例です。name 属性を使用して、Topology_Container の名前を指定します。 name 属性と model_type 属性は、作成されたモデルを一意に識別します。 モデル ハンドルを指定する場合、name 属性と model_type 属性は無視されます。
Topology_Containers には、他の Topology_Container エレメント、デバイス、または接続を含めることができます。 Topology_Container モデルは、常に OneClick Topology ビューに配置されます。

注: デフォルトでは、name 属性と model_type 属性は CA Spectrum 属性 Model_Name および Modeltype_Name の値を指定します。 ただし、.modelinggatewayresource.xml ファイルを編集することにより、name 属性が値を指定する CA Spectrum 属性を変更できます。（モデル タイプと共に）選択する新しい CA Spectrum 属性にかかわらず、その属性が、コンテナを一意に識別するために使用されます。 この変更により、2 つのコンテナのモデル名が同じになります。

Location_Container

Location_Container エレメントは物理的位置または地理的位置に従って他のモデルをグループ化します。 Building と Room は両方とも Location_Container エレメント モデル タイプ値の例です。
Location_Container エレメントはコンテナ モデルを作成するので、使用する特定のコンテナを識別するには、model_type 属性またはモデル ハンドルを使用します。 可能な model_type 値は DTD に列挙されます。 name 属性を使用して、Location_Container の名前を指定します。 name 属性と model_type 属性は、作成されたモデルを一意に識別します。モデル ハンドルを指定する場合、name 属性と model_type 属性は無視されます。

注: デフォルトでは、name 属性と model_type 属性は CA Spectrum 属性 Model_Name および Modeltype_Name の値を指定します。 ただし、.modelinggatewayresource.xml ファイルを編集することにより、name 属性が値を指定する CA Spectrum 属性を変更できます。（モデル タイプと共に）選択する新しい CA Spectrum 属性にかかわらず、その属性が、コンテナを一意に識別するために使用されます。 この変更により、2 つのコンテナのモデル名が同じになります。

Device

Device エレメントは、ネットワーク上のデバイスを定義します。 CA Spectrum のデバイス モデルのインスタンスを作成、更新、または破棄するには、このエレメントを他のエレメントと共に使用します。SNMP デバイスを操作する場合、ip_dnsname 属性を使用してデバイスを一意に識別するために、有効で一意の IP アドレスまたは DNS 名を指定します。 この一意の識別により CA Spectrum はデバイスと通信し、デバイス機能に基づいて最も適切なモデル タイプを選択します。ip_dnsname を有効な文字列に設定します。ip_dnsname が無効または接続不可能な場合、デバイス モデルの作成は失敗する可能性があります。 無効または接続不可能な ip_dnsname で model_type を指定した場合でも、デバイス モデルは指定されたモデル タイプで作成されます。 ただし、デバイス モデルはアクティブ化されず、有効なネットワーク情報またはステータスを提供しません。 デバイスに対する可能な model_type 値は .modelinggatewayresource.xml ファイルに列挙されます。

Schedule

Schedule エレメントは、デバイス モデルがメンテナンス モードになるタイミングを定義します。 デバイス モデルがメンテナンス モードの場合、デバイスとそのコンポーネントへの管理トラフィックは停止されます。 トラフィックの停止により、デバイスでメンテナンスを実行している間、CA Spectrum はデバイス モデルでイベントまたはアラームを生成できなくなります。

Port

デバイス モデルを作成すると、ポートはデバイスに対して自動的に作成されます。 Port エレメントにより、いくつかの CA Spectrum ポート属性値を変更するか、ポート レベル接続を指定できます。フレーム リレーまたは ATM 回路を含めて、異なる種類のポートを指定できます。デバイスのポートを識別するには、identifier_name 属性と identifier_value 属性の値を指定します。 identifier_name に対する可能な値は DTD で列挙されます。 identifier_value は identifier_name 属性で選択される識別子の値です。 Port エレメントを、Device エレメントの子として常に指定する必要があります。

Connection

Connection エレメントは、WAN リンク接続を含めて、2 つのデバイス間の物理的または論理的な接続を定義するため、このエレメントには 2 つの子 Device エレメントを含める必要があります。 Port エレメントが Device エレメントで指定される場合、接続はそのデバイス用の指定されたポートで解決されます。Device エレメントが Port エレメントを指定しない場合、CA Spectrum ディスカバリは接続で解決するポートの特定を試みます。

GenericView_Container

一般的なビューでコンテナ モデルを作成するには、GenericView_Container エレメントを使用します。 GenericView と GenericView_Container の両方のエレメントは、カスタマイズされたビューを作成するために使用されます。 そのため、インテグレータとして、このコンテナを使用するタイミングと方法を決定します。

タスク指向型エレメント

DTD で定義されている他のエレメントは、タスク指向型エレメントです。これらのエレメントとそれらの属性は、入力ファイルが生成するアクションのタイプの定義に役立ちます。 それらを使用すると、新しいトポロジ情報を作成するか、既存のトポロジ情報を更新、上書き、または削除できます。 個別の入力ファイルは、Connection エレメントを除いて、これらの各エレメントを使用しないか、1 つを使用できます。 必要に応じて任意の数の Connection エレメントを使用できます。

- Topology
- Location
- GenericView
- Connection
- Update
- Destroy

新しいトポロジ データ

タスク指向のエレメントは、XML ファイルで実行するアクションを定義します。CA Spectrum で新しいネットワーク トポロジ データを作成する場合は、Topology エレメントと Location エレメントを使用します。これらのエレメントは、データを作成する場所の階層ビューを定義します。その後、ネットワーク エンティティ用のモデルを作成するために、子エレメントとして対応するモデル エレメントを使用できます。 特定の統合ニーズに応じてビューをカスタマイズするには、GenericView エレメントを使用します。

場所ビューの作成

OneClick でワールド トポロジで表示するために、場所ビューでトポロジ情報を作成できます。場所ビューでこの情報を作成するには、Import root エレメントの内部で Location エレメントを使用して、XML ファイルを作成します。 Location エレメントには、特定のコンテナ モデルを作成する Location_Container エレメント、またはデバイス モデルを作成する Device エレメントを含めることができます。

例: 場所ビューの Site コンテナ

以下の例では、場所ビューに Site コンテナを作成し、そのコンテナ内にデバイスを作成します。

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
        <Location>
                <Location_Container  model_type = "Site" name = "My_Town" >
                        <Device ip_dnsname= "10.253.9.18"
                                community_string="public"/>
                </Location_Container>
        </Location>
</Import>
```

Import エレメントはルート エレメントであり、常に入力ファイルに含まれます。

Location エレメントは、場所ビューでモデルを作成することを示します。

Location_Container エレメントはコンテナ モデルを作成します。 このモデルは、ネットワークの物理コンポーネントではなく論理コンポーネントです。 したがって、CA Spectrum はこのコンポーネントに接続することができず、IP アドレスまたは DNS 名を使用してモデル タイプを定義することができません。作成するコンテナ モデルのタイプを示すには、model_type 属性および name 属性の値を指定します。 可能な model_type 属性値は、DTD および Location_Container (P. 67) セクションにリストされます。 name 属性は必須であり、モデルの一意の名前を指定する必要があります。

注: モデル ハンドルを指定する場合、モデル ハンドルはコンテナ モデルを識別するために使用されます。そのため、name 属性と model_type 属性は指定しても無視されます。

Device エレメントは Site Location_Container モデルの内部でモデルを作成します。 ip_dnsname 属性は Device エレメントの必須の属性です。 デバイスに接続できる場合、CA Spectrum はデバイスを検索するために IP アドレスまたは DNS 名を使用します。

これらのエレメントおよびそれらの可能な属性の完全な詳細については、「文書型定義エレメント (P. 51)」を参照してください。

詳細情報:

ディスカバリを使用した接続の作成 (P. 28)
Connection エレメントを使用した接続の作成 (P. 28)
文書型定義エレメント (P. 51)

トポロジ ビューの作成

トポロジ ビューにネットワーク データをインポートし、OneClick のユニバース トポロジで表示できます。トポロジ ビューにインポートするには、Import root エレメントの内部で Topology エレメントを使用して XML ファイルを作成します。Topology エレメントには以下のものを含めることができます。

- 特定のタイプのコンテナ モデルを作成する Topology_Container エレメント。
- 特定のタイプのデバイス モデルを作成する Device エレメント。
- 2 つのデバイス間の接続を作成する Connection エレメント。

DTD で説明されている階層と構文のルールを使用して、ネットワークの物理的および論理的な接続を正確に表すことができます。

例: トポロジ ビューの LAN コンテナ

この例では、トポロジ ビューに LAN コンテナを作成し、そのコンテナ内にデバイスを作成します。

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
        <Topology>
                <Topology_Container  model_type = "Lan"
                        name = "Sample_LAN" Security_String = "public"
                        subnet_address= "10.253.9.0" subnet_mask =
"255.255.255.0">
                        <Device ip_dnsname= "10.253.9.18"
                                community_string="public"/>
                </Topology_Container>
        </Topology>
</Import>
```

Import エレメントはルート エレメントであり、常に入力ファイルに含まれます。

Topology エレメントは、トポロジ ビューでモデルを作成することを示します。

Topology_Container エレメントはコンテナ モデルを作成します。このモデルは、ネットワークの物理コンポーネントではなく論理コンポーネントです。 したがって、CA Spectrum はこのコンポーネントに接続することができず、IP アドレスまたは DNS 名を使用してモデル タイプを定義することができません。作成するコンテナ モデルのタイプを示すには、model_type 属性および name 属性の値を指定します。 可能な model_type 属性値は、DTD および Topology_Container (P. 91) セクションにリストされます。name 属性は必須であり、モデルの一意の名前を指定する必要があります。 指定される他の属性はオプションです。

注: モデル ハンドルを指定する場合、モデル ハンドルはコンテナ モデルを識別するために使用されます。そのため、name 属性と model_type 属性は指定しても無視されます。

Device エレメントは LAN Topology_Container モデル内部でモデルを作成します。 ip_dnsname 属性は Device エレメントの必須の属性です。 CA Spectrum がデバイスに接続できる場合、デバイスを検索するために IP アドレスまたは DNS 名が使用されます。 CA Spectrum がデバイスを見つけるときに、モデルの作成に使用する適切なモデル タイプを決定します。

複数のビューでの同じデバイスの表現

複数のビューでデバイスを表す XML ファイルを作成できます。 このファイルを作成するには、このデバイスのモデルを作成するために使用する各 Device エレメントの属性と属性値が同じかどうか確認することをお勧めします。 同じでない場合、インポート ツールは Device エレメントの属性と値をマージし、一貫性のある属性と値のセットの作成を試みます。 これらの各 Device エレメントで属性が指定されているが、別の値が使用される場合があります。この場合、XML ファイルでリストされる最後の Device エレメントの値が、そのデバイスのモデルを作成するために使用される他の Device エレメントの、その属性に対するそれ以前のすべての値をオーバーライドします。

いくつかの属性にはデフォルト値があることに注意してください。 たとえば、属性 community_string のデフォルト値は Public です。 そのため、デバイス A を表すために Device エレメント属性および値を指定する場合は、デバイス A を表す他の Device エレメントでその属性を指定することをお勧めします。 このようにすると、その属性のデフォルト値が、以前に指定された値をオーバーライドするために使用されなくなります。

例: トポロジ ビューと場所ビューでのデバイスの作成

以下の例では、デバイス 10.253.9.16 がトポロジ ビューと場所ビューの両方で作成されます。 デバイスを説明するために使用される属性および値は、両方のビューで同じです。

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
<!-- Topology View import -->
        <Topology discover_connections="false" complete_topology="false">
                <Device ip_dnsname="10.253.9.16" community_string="zippo" />
        </Topology>
<!-- Location View import -->
        <Location complete_topology="true">
                <Location_Container model_type="Site" name="Durham">
                        <Device ip_dnsname="10.253.9.16"
community_string="zippo"/>
                </Location_Container>
        </Location>
</Import>
```

ディスカバリを使用した接続の作成

CA Spectrum 接続は、2 つのデバイス間の物理的または論理的なリンクを表します。 XML 入力ファイルで、2 つの異なる方法で接続を作成できます。最初の方法では Topology エレメントの discover_connections 属性を使用して、CA Spectrum ディスカバリを使用します。 discover_connections 属性が true に設定される場合、ディスカバリは新しく作成されたデバイス モデルで実行されます。 その後、ディスカバリは、これらのデバイスの接続を確立します。

注: ディスカバリの詳細については、「IT インフラストラクチャのモデリング/管理 - 管理者ガイド」を参照してください。

例: ディスカバリを使用した接続の作成

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
        <Topology >
                <Topology_Container  model_type = "Lan" name = "Sample_LAN"
discover_connections= "true"
                        Security_String = "public" subnet_address= "10.253.9.0"
                        subnet_mask = "255.255.255.0" >
                        <Device ip_dnsname= "10.253.9.18"
                                community_string="public"/>
                        <Device ip_dnsname= "10.253.9.20"
                                community string="public"/>
                </Topology_Container>
        </Topology>
</Import>
```

Connection エレメントを使用した接続の作成

接続を作成する 2 番目の方法は Connection エレメントを使用することです。この場合、作成済みのデバイスを接続します。接続は 2 つのポート間、デバイスとポート間、または 2 つのデバイスの間で指定できます。

デバイス間の接続を指定すると、CA Spectrum はデバイスの障害を分離できますが、ポート レベル接続を指定することをお勧めします。 ポート レベル接続はより高いグレードの接続で、CA Spectrum が接続を解決し、ポート レベルで障害を分析できるようにします。 CA Spectrum は、2 つのデバイス間の接続を指定すると自動的にポートの決定を試みますが、ユーザは、接続で使用される 1 つまたは両方のポートを指定しません。このプロセスが成功すると、CA Spectrum はポート レベルで接続を解決します。

CA Spectrum は、これらの条件が両方とも当てはまる場合、接続の失敗を示すエラーを生成します。

- CA Spectrum は、接続で使用される両方のポートを決定できません。
- これらのデバイスの少なくとも 1 つは管理可能デバイスです。

エラーはエラー ログ ファイルに書き込まれます。

CA Spectrum がポートの 1 つのポートのみを決定できる場合、接続は、接続の一方のポート レベルでのみ解決されます。 反対側は引き続きデバイス レベルで解決されます。

両方のデバイスが管理不可能なデバイスである場合、CA Spectrum はデバイス レベルで接続を確立します。

例: 既存のポート間の接続の作成

以下の例は、各ポートが別のデバイスに属する 2 つの既存のポート間の接続を作成します。 Connection エレメントは、リンクされる Port エレメントと Device のエレメントの両方を識別します。 Port エレメントが各 Device エレメント内で指定されるので、その接続は両方のデバイスに対してポート レベルで解決されます。

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
        <Connection>
                <Device ip_dnsname= "172.19.57.93">
                        <Port identifier_name = "frCircuitTableInstance"
                                identifier_value="4.161"/>
                </Device>
                <Device ip_dnsname = "192.168.125.161">
                        <Port identifier_name= "frCircuitTableInstance"
                                identifier_value= "2.861"/>
                </Device>
        </Connection>
</Import>
```

前の例では、DLCI ポート間の接続を指定しています。 identifier_name 属性の値が frCircuitTableInstance なので、ポートは、MIB で frCircuitTable オブジェクトからの OID インスタンス値を使用して識別されます。OID インスタンス値は identifier_value 属性を使用して指定されます。

デバイスの階層配置を示すため、Connection エレメントは Topology エレメントまたは Topology_Container エレメントに含まれます。 この場合、入力ファイルの結果は変更されません。

重要: 同じ Port エレメントを使用して、複数の Connection エレメントが含まれる XML ファイルをインポートしようとした場合、Modeling Gateway はエラーをレポートしません。1 つのポートは複数の接続を持つことができません。 同じポートが複数の Connection エレメントで指定される場合、XML ファイル内の最後の Connection エレメントが、そのポートを指定する以前の Connection エレメントをすべてオーバーライドします。

詳細情報:

エラー ログ (P. 42)

WA_Link 接続の作成

WA_Link 接続を作成するには、以下の構文を使用します。

```
<Connection>
    <Device ip_dnsname=10.253.9.18/>
    <Device ip_dnsmame=10.253.9.100 model_type="WA_Link">
</Connection>
```

Modeling Gateway は自動的に WA_Segment を作成します。リンクはセグメントとデバイスの間で作成されます。2 番目のデバイスとリンクの間の接続を指定するには、2 番目の接続タグをインポート ファイルに追加します。

CA Spectrum とサードパーティ データベース間の情報の同期

Topology、Location、Topology_Container、および Location_Container の各エレメントには、complete_topology という名前の属性があります。 この属性の値を true に設定すると、XML ファイルが、CA Spectrum が識別する必要があるすべてのモデルと接続を定義することを示します。 XML ファイルが CA Spectrum にインポートされた場合、XML ファイルで表されていない CA Spectrum ビューの任意のモデルはロスト ファウンドに送信されます。 ビューにサブコンテナがある場合、CA Spectrum はサブコンテナを指定するエレメントで設定された complete_topology 属性の値を参照します。 complete_topology 属性値がサブコンテナ エレメントで指定されない場合、値は親エレメントから継承されます。 したがって、親エレメントで complete_topology が true に設定され、サブコンテナ エレメントで complete_topology の設定を指定していない場合、サブコンテナの complete_topology 値も true になります。

CA Spectrum が XML ファイルをインポートするときに、モデルは以下の条件下でロスト ファウンドに送信されます。

- モデルは、インポート先のビューまたはそのビューのサブコンテナに直接存在している。
- モデルが XML ファイルには存在していない。

この動作は、サードパーティ データベースのデータを CA Spectrum のデータと同期する場合に役立ちます。

例: Complete_Topology を True に設定

以下の例では、complete_topology を Topology エレメント内で true に設定します。この入力ファイルで指定されるモデルを除いて、トポロジ ビューの既存のすべてのモデルはロスト ファウンドに送信されます。 このサンプル入力ファイルでは、以下の 2 つのモデルのみが指定されます。

- LAN Topology_Container
- IP アドレスが 10.253.9.18 のデバイス

これらのモデルが存在しない場合は作成されます。 存在する場合、CA Spectrum は入力ファイルの属性値を使用して、それらの CA Spectrum 属性値を更新します。 トポロジ ビューに存在するその他のモデル（VNM を除く）は、ロスト ファウンドに送信されます。

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
        <Topology complete_topology="true">
                <Topology_Container  model_type = "Lan" name ="Sample_LAN"
                        Security_String = "public" subnet_address= "10.253.9.0"
                        subnet_mask = "255.255.255.0">
                        <Device ip_dnsname= "10.253.9.18"
                                community_string="public"/>
                </Topology_Container>
        </Topology>
</Import>
```

complete_topology 属性が Topology エレメントではなく Topology_Container エレメントで使用された場合、CA Spectrum は未指定のモデルのみを Topology_Container で階層から削除します。

情報の更新

既存のモデル用の CA Spectrum 属性および関連付け情報を更新するには、Update エレメントを使用します。Update エレメントは Container エレメント、Device エレメント、および Association エレメントを囲むことができます。 Port 属性の値は適切な Device エレメントを使用して更新されます。

例: 2 つの異なるモデル用の属性の更新と、関連付けの作成

以下の例では、Update 入力ファイルを示します。 この場合、2 つの異なるモデル用の 2 つの属性が更新され、関連付けは 2 つのモデル間で作成されます。

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
  <Update>

    <Topology_Container model_type="Lan" name="Sample"
                        model_name = "newLAN"/>
    <Device ip_dnsname= "Test1" poll_interval= "1108"/>

    <Association relation="0x10002">
       <Left_Model> <Topology_Container name="Net"
                                        model_type="Network" /></Left_Model>
       <Right_Model> <Device ip_dnsname="172.24.94.94" /></Right_Model>
    </Association>
  </Update>
</Import>
```

最初に更新される属性は、LAN コンテナ モデルの model_name 属性です。モデル名は Sample から newLAN に変更されます。 name 属性と model_name 属性の使用について注意してください。 これらの属性は両方とも CA Spectrum 属性 Model_Name を変更するために存在します。コンテナ モデルを識別し、model_name 属性を使用してコンテナ モデルの新しい名前を指定するには、値として現在の名前を持つ name 属性を使用します。

次に、デバイスのポーリング間隔の値を Test1 から 1108 に変更します。poll_interval 属性に新しい値を割り当てると、古い値が上書きされます。

この例では、両方のモデルが SpectroSERVER に存在する限り、Network モデル タイプのコンテナ モデル "Net" とデバイス モデル 172.24.94.94 の関係 0x10002 の関連付けを作成します。

情報の破棄

コンテナ モデル、デバイス モデル、接続、および関連付けを削除するには、Destroy エレメントを使用します。 デバイスを破棄すると、デバイスと関連付けられたすべてのポートとアプリケーション モデルも破棄されます。

例: LAN コンテナ、接続、および関連付けの破棄

以下の例では、newLAN という名前の LAN トポロジ コンテナが破棄されます。 これらが破棄されるように指定されない限り、このコンテナ内のすべてのモデルはロスト ファウンドに送信されます。 また、この例では、ポート レベルで指定される 2 つのデバイス Test1 および Test2 間の接続も破棄します。 モデル タイプ Network のコンテナ モデル "Net" およびデバイス 172.24.94.94 の間に Collects 関連付けが存在する場合、この関連付けは破棄されます。

```
<?xml version = "1.0" standalone = "no" ?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">

<Import>
    <Destroy>
        <Topology_Container model_type="Lan" name="newLAN"/>

        <Connection>
          <Device ip_dnsname= "Test1">
           <Port identifier_name= "ifIndex" identifier_value= "1"/>
          </Device>
          <Device ip_dnsname= "Test2">
           <Port identifier_name="ipAddress"
              identifier_value = "10.253.8.18"/>
          </Device>
        </Connection>

        <Association relation="Collects">
       <Left_Model> <Topology_Container name="Net"
                                        model_type="Network" /></Left_Model>
       <Right_Model> <Device ip_dnsname="172.24.94.94" /></Right_Model>
    </Association>

    </Destroy>

</Import>
```

重要: ネットワークからすでに削除されたデバイスを表すモデルを破棄するには、XML ファイルの Destroy エレメントで Device の ip_dnsname 属性を指定するときに、DNS 名ではなくデバイスの IP アドレスを使用します。デバイスがネットワークから削除されると、そのデバイスの DNS エントリは存在しなくなります。また、Modeling Gateway は削除する適切なモデルを識別できません。

.modelinggatewayresource.xml ファイル

Topology_Container、Location_Container、および Device の各エレメントには、有効な CA Spectrum モデル タイプに等しい値を持っている必要がある model_type 属性があります。 CA Spectrum は一意に 16 進数を使用してモデル タイプを識別します。 これらの 16 進値はリソース ファイル .modelinggatewayresource.xml で列挙されています。 このファイルは、一意の 16 進の識別子でモデル タイプ用のテキスト値をペアにします。その後、テキスト値は DTD に表示されます。

DTD で定義されている属性の多くは CA Spectrum 属性に対応します。 CA Spectrum 属性は、16 進数を使用して、CA Spectrum で一意に識別されます。 .modelinggatewayresource.xml ファイルは DTD または XML ファイルでこれらの 16 進値を使用しません。 代わりに、このファイルは、より直観的なテキスト ベースの名前で CA Spectrum 属性の 16 進の識別子をペアにします。

.modelinggatewayresource.xml ファイルの ModelType エレメントおよび Attribute エレメントは両方ともカスタマイズできます。

注: .modelinggatewayresource.xml ファイルは CA Spectrum からのトポロジ データのエクスポートにも使用されます。詳細については、「CA Spectrum からのトポロジ データのエクスポート (P. 43)」を参照してください。

文字セット エンコーディングの定義

CA Spectrum XML 入力ファイルはデフォルトでは UTF-8 でエンコードされます。特殊文字または外国語をインポートするには、以下の例のように、XML ファイル ヘッダで適切な文字セット エンコーディングを指定します。

例: 入力ファイル文字エンコードをギリシャ語に設定

以下の例は、文字エンコードをギリシャ語に設定するために XML ファイルを変更する方法を示しています。

```
<?xml version="1.0" encoding="ISO-8859-7" standalone="no"?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
```

CA Spectrum 文字セット エンコーディング情報の表示

CA Spectrum が使用する文字セットのエンコーディングを判断する必要がある場合、OneClick 管理ページから行うことができます。

CA Spectrum 文字セット エンコーディング情報を表示する方法

1. OneClick ホーム ページで［管理］をクリックします。

   ［管理ページ］が開きます。

2. 左側のパネルで［文字セット］をクリックします。

   ［文字セット エンコーディング］ページが開き、エンコーディングと該当する言語のリストが表示されます。

カンマ区切り入力ファイル

Modeling Gateway は、XML 入力ファイルで ATM 接続およびフレーム リレー接続を指定できます。 ツールキットは、カンマ区切りの ASCII テキスト ファイルから ATM 接続およびフレーム リレー接続の情報を使用することもできます。 このファイルは、以下の間の接続に関する情報をインポートするために使用できます。

- 2 つの ATM 回路
- 2 つのフレーム リレー回路
- ATM およびフレーム リレー回路

接続を表すために、OneClick でライブ パイプの作成を指定できます。 複数の接続を同じ入力ファイルで指定できます。

注: これらの接続に関係するデバイス モデルは、以前に CA Spectrum に存在している必要があります。

カンマ区切りファイルの構文

以下の例では、カンマ区切りの入力ファイルで使用される形式を示します。

```
<Device_IP>, <OID>, <Device_IP>, <OID>, <CircuitName>, <CircuitID>, <Pipe>
```

Device_IP

接続に関係する各デバイスの IP アドレスを指定します。 必須です。

OID

デバイス上の回路リンクを指定するには、frCircuitTable、atmVclTable、または atmVplTable の OID インスタンスを指定します。

CircuitName

（オプション）含まれる回路の名前を指定します。

CircuitID

（オプション）含まれる回路の ID を指定します。

Pipe

（オプション）有効な 2 つの値は、CREATE_PIPE または NO_CREATE_PIPE です。 値を CREATE_PIPE に設定すると、指定された接続間でライブ パイプが作成されます。 値を NO_CREATE_PIPE に設定すると、指定された接続間でライブ パイプは作成されません。値がこのパラメータに対して指定されない場合、CREATE_PIPE がデフォルト値として見なされます。

例: フレーム リレー回路間の指定された接続

以下の例では、2 つのフレーム リレー回路間の接続を指定する入力ファイルを示します。ライブ パイプはこれらの 2 つのポート間で作成されます。

```
172.19.57.93, 4.161, 192.168.125.161, 2.161, FR_Circuit_Name, Circuit_Id_123,
CREATE_PIPE
```

インポートのための modelinggateway ツールの実行

インポートのために modelinggateway ツールを実行するには、以下の構文を使用します。

Windows

```
modelinggateway.bat -vnm vnm_name -i import_file [-o outputfile] [-debug debugfile]
```

Solaris/Linux

```
modelinggateway -vnm vnm_name -i import_file [-o outputfile] [-debug debugfile]
```

-vnm *vnm_name*

SpectroSERVER ホストの名前を指定します。

-i *import_file*

.modelinggateway.dtd でコンパイルされる必要な入力情報が含まれた XML ファイルの名前を指定します。

-o *outputfile*

（オプション）*outputfile* パラメータで名前を付けられたファイルにエラー情報を記録します。 このオプションを使用しない場合、エラー情報は *import_file.log* という名前のファイルに記録されます。 *Import_file* は XML ファイルの名前です。

-debug *debugfile*

（オプション）インポート プロセス中にデバッグ出力ファイルを作成することを示します。 -debug オプションを使用する場合、出力に独自のデバッグ ファイル名を指定できます。*debugfile* に値を指定しない場合、デバッグ ファイル名はデフォルトで ".debug" というサフィックスが付けられた *import_file* 名になります。

注: -debug オプションでは、Modeling Gateway が実行されるマシン上に空きディスク容量が必要です。 たとえば、*import_file* のモデルの数が多い場合、またはデバイス モデルが大きなインターフェース密度を持っている場合、大きなデバッグ出力ファイルが生成される可能性があります。

ImportConfiguration エレメント

Modeling Gateway がデータをインポートする方法の特定の要素を制御するには、ImportConfiguration エレメントを使用します。

ImportConfiguration エレメントの構文は以下のとおりです。

```
<ImportConfiguration
    do_not_process_pre_existing_devices_under_container_node = "false"
    import_to_primary_ss_only = "false"
    max_device_creation_threads="50"
/>
```

do_not_process_pre_existing_devices_under_container_node

CA Spectrum が、以前に CA Spectrum に存在した Container エレメントで見つかったデバイスを処理するかどうかを指定します。

デフォルト： false

import_to_primary_ss_only

プライマリ SpectroSERVER がダウンしている場合、Modeling Gateway がセカンダリ SpectroSERVER に接続するかどうかを指定します。

デフォルト： false

max_device_creation_threads

同時に作成し、アクティブにできるデバイス モデルの数を指定します。

注: この値を 50 を超える数値に設定すると、SNMP トラフィックが大きくなりすぎる可能性があります。

デフォルト： 50

カンマ区切りファイルのインポート

前のセクションでは、modelinggateway インポート ツールを使用して、入力ファイルをインポートする方法について説明しました。 OneClick インターフェースからフレーム リレー接続データと ATM 接続データをインポートすることもできます。

次の手順に従ってください:

1. OneClick コンソールで該当する VNM モデルをクリックします。
2. コンポーネント詳細画面で［情報］タブをクリックします。
3. ［論理接続インポート］をクリックしてセクションを展開します。
4. ［インポート］をクリックし、CA Spectrum にインポートするデータが含まれているカンマ区切りのファイルを見つけ、［開く］をクリックします。

   データがインポートされます。［インポート結果］ダイアログ ボックスが表示され、インポートの成功に関する情報が示されます。

インポート情報の表示

CA Spectrum Modeling Gateway は、各データベース インポート操作の安全性および正確性を確保するためのメカニズムを提供します。 CA Spectrum Modeling Gateway は、行われた作成、削除、関連付け、および更新のレコードが含まれる監査証跡を維持します。 OneClick 内でインポートに関するデータを表示できます。また、エラーとデバッグ ログのインポート問題に関する情報を追跡することもできます。

OneClick での Modeling Gateway 結果の表示

VNM モデルからの Modeling Gateway インポートの結果は、OneClick コンソールの［情報］タブで確認できます。

次の手順に従ってください:

1. OneClick コンソールで該当する VNM モデルをクリックします。
2. コンポーネント詳細画面で［情報］タブをクリックします。
3. Modeling Gateway をクリックしてセクションを展開します。
4. 最近のインポートの詳細については、Modeling Gateway セクションのテーブルを確認してください。 このテーブルには、以下の情報が含まれます。

   ファイルのインポート

   インポート ファイルの名前が表示されます。

   ログ ファイル

   ログ ファイルの名前が表示されます。

   開始時刻

   トポロジ インポート プロセスが開始した時刻を示します。

   終了時刻

   トポロジ インポート プロセスが終了した時刻を示します。

進捗

進捗状況フィールドは、まだ完了していないトポロジ インポートのステータスを表示します。 このフィールドに対する可能な値は以下のとおりです。

- 初期化中
- モデルを識別中
- モデルを作成中
- モデルをアクティブ化中
- 接続をマップ中
- モデルを配置中
- 接続を作成中
- モデルを更新中
- モデルを破棄中
- 完了
- 切断

エラー数

インポート プロセス中に生成されたエラーの数を表示します。

作成されたモデル数

インポート プロセス中に作成されたモデルの数を表示します。

破棄されたモデル数

インポート プロセス中に破棄されたモデルの数を表示します。

更新されたモデル数

インポート プロセス中に更新されたモデルの数を表示します。

作成された接続数

インポート プロセス中に作成された接続の数を表示します。

削除された接続数

インポート プロセス中に削除された接続の数を表示します。

5. テーブルでリストされるインポート ファイルの数を変更するには、［最大レコード数］設定リンクをクリックします。

6. 必要に応じてデータを並べ替えるには、いずれかのテーブル列ヘッダをクリックします。
7. インポート データを特定の条件に制限するには、［フィルタ］フィールドにテキストを入力します。
8. 処理中にインポートのステータスを確認するには、［更新］をクリックします。

   画面が更新され、使用可能な最新のインポート情報が表示されます。

エラー ログ

すべてのエラーと考えられる原因は、エラー ログ ファイルに記録されます。 デフォルトでは、インポート ツールは、*<nameofimportfile>*.log という名前のエラー ログを作成します。*<nameofimportfile>* はインポート ファイルの名前です。 また、インポート ツールのセクションで指定される構文を使用して、ログ ファイルに特定の名前を指定することもできます。インポートが完了すると、ログ ファイルが SS-Tools ディレクトリに表示されます。 ログ ファイルには、モデルの正常な作成数、削除数、および更新数と、接続数が記録されます。 また、インポート プロセス中に発生した単一の障害も個別に記録されます。

第 3 章: CA Spectrum からのトポロジ データのエクスポート

このセクションには、以下のトピックが含まれています。

CA Spectrum からのトポロジ データのエクスポートについて

Modeling Gateway は SpectroSERVER からのトポロジ情報および設定のエクスポートをサポートしています。 情報は XML でフォーマットされたファイルにエクスポートされます。このファイルは Modeling Gateway を使用して、指定された SpectroSERVER にインポートできます。

デフォルトでは、以下の種類の情報がエクスポートされます。

- Device エレメントと設定情報。
- Port エレメントと設定情報。
- Container エレメントと設定情報。
- 接続（解決済みおよび未解決の WA_Link 接続）。
- ユニバース トポロジ階層。
- 注釈とズーム情報を含み、背景画像を含まないユニバース トポロジの各ビューのレイアウト。
- ユーザ モデルとユーザ スキームの全体（ユーザ関連の関係、属性、および LicenseRole、AccessGroup、PrivilegeRole、UserGroup などのモデル）。
- ディスカバリ設定。
- サービス管理スキームと属性。
- 各グローバル コレクション内のすべてのモデル、すべての動的なコレクション条件、ズームされたリスト、グループ化されたリスト、およびトポロジ レイアウトを含む、静的および動的なグローバル コレクション。

重要: コンバータ ツールの目的は、2 つ以上の SpectroSERVER にわたって SpectroSERVER データベースを分割するか、単一の SpectroSERVER データベースのランドスケープ ハンドルを変更することです。Modeling Gateway エクスポート機能をコンバータ ツールと同じ目的に使用している場合は、すべての情報がエクスポートされるとは限らないことに注意してください。 動作を設定し、エクスポート機能で現在サポートされていないデータ用のモデリングをセットアップするため、エクスポートされたデータがインポートされた後で、いくつかの手動の作業が必要になります。 エクスポートされないデータのタイプの例には、サービス パフォーマンス マネージャ（SPM）テストや、Policy Manager ポリシーなどが含まれます。

エクスポート設定

.modelinggatewayresource.xml ファイルの ExportConfiguration エレメントを変更することにより、エクスポートされるものを制御できます。 デフォルトでは、ユニバース コンテナのすべてのトポロジ情報とモデリング情報がエクスポートされます。 エクスポート元の別のルート コンテナを指定するには、RootContainerToExport エレメントを変更できます。ルート コンテナおよびその各サブコンテナのすべてのコンテンツがエクスポートされます。

注: データのエクスポートに DTD を使用する必要はありません。DTD はデータのインポートにのみ使用されます。

デバイス、コンテナ、およびポートに対してエクスポートされる属性は、DeviceExportAttributes、ContainerExportAttributes、および PortExportAttributes の各エレメントで定義されます。 必要に応じて、これらのエレメントに対して属性を追加または削除します。

SpectrumConfigurationExport エレメントは、ExportConfiguration エレメントの export_spectrum_settings フラグが true に設定される場合、エクスポートされる CA Spectrum 設定データのタイプを制御します。

たとえば、以下のエレメントは、LostFound モデルに対してエクスポートされる属性を制御します。

```
<SpectrumConfigurationExport model_type="LostFound" >
     <Automatic_Model_Destruction attribute_id="0x11de1" />
     <Model_Destruction_Interval_Hours attribute_id= "0x11de3"/>
     <Model_Destruction_Interval_Minutes attribute_id="0x11de4" />
</SpectrumConfigurationExport>
```

詳細情報:

ExportConfiguration エレメント (P. 45)

ExportConfiguration エレメント

エクスポート動作を制御するには、.modelinggatewayresource.xml ファイルで ExportConfiguration エレメントを使用します。

ExportConfiguration エレメントの構文は以下のとおりです。

```
<ExportConfiguration
    export_devices           = "true"
    export_containers        = "true"
    export_port_attributes   = "true"
    export_links             = "true"
    export_topology_layout   = "true"
    export_annotation        = "true"
    export_WA_Link_models    = "true"
    export_spectrum_settings = "true"
    export_user_models       = "true"
    export_service_modeling  = "true"
    export_schedules         = "true"
    export_global_collections="true"
    export_discovery_configs = "true"
    export_from_primary_ss_only = "false"
    export_policy_manager = "true"

/>
```

export_devices

デバイス モデルをエクスポートします。

export_containers

コンテナ モデルをエクスポートします。

export_port_attributes

ポート属性をエクスポートします。

export_links

デバイス リンクをエクスポートします。

export_topology_layout

トポロジでデバイスおよびコンテナ モデルの x,y 座標をエクスポートします。

export_annotation

注釈およびモデル グループ情報をエクスポートします。

export_WA_Link_models

WA_Link モデルをエクスポートします。 WA_Link モデルをエクスポートしない場合、モデルは透過的に処理されます。 2 つのデバイス モデル間のワイド エリア リンクは直接リンクとしてエクスポートされます。

export_spectrum_settings

障害分離、ディスカバリ、および VNM コントロール用の設定などの CA Spectrum 設定をエクスポートします。

export_user_models

ユーザ モデル、ユーザ ライセンス、ユーザ権限、ユーザ設定、および他のすべてのユーザ関連の関係属性とモデルをエクスポートします。

export_servicemodeling

サービス管理スキームおよび属性をエクスポートします。

注: Service Manager の詳細については、「Service Manager ユーザ ガイド」を参照してください。

export_schedules

スケジュールをエクスポートします。

export_global_collections

各グローバル コレクション内のすべてのモデル、すべての動的なコレクション条件、ズームされたリスト、グループ化されたリスト、およびトポロジ レイアウトを含む、静的および動的なグローバル コレクションをエクスポートします。

export_discovery_configs

ディスカバリ設定をエクスポートします。

export_from_primary_ss_only

プライマリ SpectroSERVER がダウンしている場合、Modeling Gateway がセカンダリ SpectroSERVER に接続するかどうかを指定します。

export_policy_manager

Policy Manager ポリシーをエクスポートします。 関連するすべてのモデル、ポリシー、ルール、権限、およびテンプレートが含まれます。

例:

以下の例では、ポート属性情報を除くすべてをエクスポートします。 この例では、プライマリがダウンしている場合に、セカンダリ SpectroSERVER に接続しないように Modeling Gateway に伝えます。

```
<ExportConfiguration
    export_devices           = "true"
    export_containers        = "true"
    export_port_attributes   = "false"
    export_links             = "true"
    export_topology_layout   = "true"
    export_annotation        = "true"
    export_WA_Link_models    = "true"
    export_spectrum_settings = "true"
    export_user_models       = "true"
    export_service_modeling  = "true"
    export_schedules         = "true"
    export_global_collections="true"
    export_discovery_configs = "true"
    export_from_primary_ss_only = "true"
    export_policy_manager = "true"

/>
```

エクスポート用の modelinggateway ツール

Modeling Gateway のコマンド ライン ツール 'modelinggateway' （Windows 上の modelinggateway.bat）は、SS-Tools ディレクトリにあります。 エクスポートの構文は以下のとおりです。

Windows

```
modelinggateway.bat -vnm vnm_name [-cmdb] -e export_file [-o outputfile] [-debug debugfile]
```

Solaris/Linux

```
modelinggateway -vnm vnm_name [-cmdb] -e export_file [-o outputfile] [-debug debugfile]
```

-vnm *vnm_name*

SpectroSERVER ホストの名前を指定します。

-cmdb

（オプション） CA CMDB に CA Spectrum を統合する場合に使用できる形式で、SpectroSERVER のコンテンツをエクスポートします。 この統合の実装の詳細については、CA サポートにお問い合わせください。

-e *export_file*

CA Spectrum トポロジ データをエクスポートします。

-o *outputfile*

（オプション）*outputfile* パラメータで名前を付けられたファイルにエラー情報を記録します。 このオプションを使用しない場合、エラー情報は *export_file.log* という名前のファイルに記録されます。 *Export_file* は XML ファイルの名前です。

-debug *debugfile*

（オプション）エクスポート プロセス中にデバッグ出力ファイルを作成することを示します。 -debug オプションを使用する場合、出力に独自のデバッグ ファイル名を指定できます。*debugfile* の値を指定しない場合、デバッグ ファイル名はデフォルトで .debug というサフィックスが付いた *export_file* 名になります。

注: -debug オプションでは、Modeling Gateway が実行されるマシン上に空きディスク容量が必要です。*export_file* 内のモデルの数は、デバッグ出力ファイルのサイズに影響します。データベース内のモデルの数が多いほど、生成されるデバッグ ファイルは大きくなります。

注: 別のサーバで modelinggateway ツールを実行するには、そのサーバに modelinggateway ツールとそのすべてのサポート ファイルを移動します。詳細については、「分散 SpectroSERVER 管理者ガイド」を参照してください。

CA Spectrum トポロジ データのエクスポート

modelinggateway ツールを使用して、CA Spectrum トポロジ データをエクスポートします。

次の手順に従ってください:

CA Spectrum トポロジ データをエクスポートするには、-e フラグを使用します。 たとえば、以下のコマンドを実行すると、NOC1_Spectrum 上の SpectroSERVER から、NOC1_data.xml という名前の Modeling Gateway 形式の XML ファイルにデータがエクスポートされます。

```
modelinggateway -vnm NOC1_Spectrum -e NOC1_data.xml
```

Modeling Gateway XML ファイルのインポート

Modeling Gateway 形式の XML ファイルから、データを CA Spectrum にインポートできます。

次の手順に従ってください:

Modeling Gateway 形式の XML ファイルをインポートするには、-i フラグを使用します。 たとえば、以下のコマンドを実行すると、NOC1_data.xml から、NOC2_Spectrum の SpectroSERVER にデータがインポートされます。

```
modelinggateway -vnm NOC2_Spectrum -i NOC1_data.xml
```

付録 A: 文書型定義エレメント

このセクションでは、Document Type Definition （DTD）で定義されている各エレメントの機能について説明します。 このセクションでは、各機能のコンテキストも示します。このセクションでは、DTD で使用される XML 構文については説明しません。 構文情報については、XML のリファレンスを参照してください。

Association

構文

親エレメント：

- Update
- Destroy

子エレメント：

- Left_Model
- Right_Model

ルール： Association エレメントには、1 つの Left_Model エレメントおよび 1 つの Right_Model エレメントが含まれている必要があります。

使用法

Association エレメントは、モデル間の関連付けを作成または破棄します。

Association エレメントが Destroy エレメントの子として使用された場合、指定された関連付けは破棄されます。 Association エレメントが Update エレメントの子として使用された場合、指定された関連付けは作成されます。

属性

関係

この関連付けでの Left_Model と Right_Model の間の CA Spectrum 関係の名前またはハンドルを指定します。

Connection

構文

親エレメント：

- Topology
- Topology_Container
- Update
- Destroy

子エレメント： Device

ルール： Connection エレメントには、2 つの Device エレメントを含める必要があります。

使用法

Connection エレメントは、2 つのデバイス間の接続を指定します。 Connection エレメントには常に 2 つの Device エレメントを含める必要があり、これらの各 Device エレメントにはゼロまたは 1 つの Port エレメントを含めることができます。 ポートが指定されている場合、接続はポート レベルで解決されます。ポートが指定されていない場合、接続用のポートを見つけるために、ディスカバリがトリガされます。

Connection エレメントが Destroy エレメントの子として使用される場合、指定された接続は破棄されます。 接続が他のコンテキストで使用される場合、接続が作成されます。

属性

create_pipe

指定された接続のグラフ表示が OneClick で行われるかどうかを示します。

デフォルト： True

Correlation

構文

親エレメント： Import

子エレメント： Correlation_Domain

ルール： Correlation エレメントには、任意の数の子エレメントを含めることができます。

使用法

Correlation エレメントは Correlation Manager モデルを表し、CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

なし。

Correlation_Domain

構文

親エレメント： Correlation

子エレメント：

- Device
- Port
- Model_Attr
- GenericView_Container

ルール： Correlation_Domain エレメントには、任意の数の子エレメントを含めることができます。

使用法

Correlation_Domain エレメントは CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name *（必須）*	文字	N/A	N/A

CustomerManager

構文

親エレメント： SM_Service_Mgt

子エレメント：

- SM_Customer
- SM_CustomerGroup

ルール： CustomerManager エレメントには、任意の数のこれらの子エレメントを含めることができます。

使用法

CustomerManager エレメントは CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

注: 属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name *(必須)*	文字	N/A	N/A
containment_relation	文字	Groups_Customers	N/A
model_type	文字	CustomerManager	N/A

Destroy

構文

親エレメント： Import

子エレメント：

- Topology_Container
- Location_Container
- GenericView_Container
- Device
- Model
- Connection
- EventModel
- SM_Service
- SM_AttrMonitor
- SM_LatencyMon
- SM_ConnectMon
- SM_SLA
- SM_Guarantee

- SM_Customer
- Association

ルール： Destroy エレメントには、必要な数のこれらの子エレメントを含めることができます。

使用法

コンテナ モデル、デバイス モデル、接続、および関連付けを削除するには、Destroy エレメントを使用します。階層を表現または破棄するために、Destroy エレメントでエレメントをネストすることはできません。 Destroy エレメントで許可されている唯一の階層は、破棄する接続をポート レベルで指定する Connection-Device-Port 階層です。 デバイス モデル、またはその内部にあるその他のコンテナ モデルを破棄せずに、コンテナ モデルを破棄することはできません。この場合、残りのモデルは、CA Spectrum ロスト ファウンドに配置されます。

属性

Destroy エレメントには属性が含まれません。

Device

構文

親エレメント：

- Topology
- Location
- Topology_Container
- Left_Model
- Right_Model
- Location_Container
- GenericView
- GenericView_Container
- Connection
- Update

- Destroy
- SM_Service
- SM_AttrMonitor

子エレメント：

- Port
- Schedule

ルール：

- **ポート：** Device エレメントが Connection エレメントに含まれている場合、1 つの Port エレメントのみが許可されます。 ポートを更新するために Update エレメントに Device エレメントが含まれている場合、複数の Port エレメントが許可されます。 Device エレメントが View、Container、または Destroy エレメントに含まれている場合、ポートは無視されます。
- **スケジュール：** Device エレメントには 1 つの Schedule エレメントを含めることができます。

使用法

デバイス モデルを作成、破棄、または更新するには、Device エレメントを使用します。 Device エレメントでは、IP アドレスまたは DNS 名を使用して、デバイス モデルを定義できます。

注: デバイスの作成中に community_string 属性および agent_port 属性が指定されない場合、CA Spectrum は事前定義済み SNMP 認証情報を使用して、デバイスを作成します。 これらの認証情報は、OneClick の VNM モデルの［情報］タブで、自動ディスカバリ制御サブビューの［モデリングとプロトコルのオプション］セクションで設定されます。

属性

ip_dnsname

デバイスの IP アドレスまたは DNS 名を指定します。デバイスが SNMP 通信をサポートしない場合、指定された model_type と共にここで一意の文字列を使用できます。

secdomain_ipname

（オプション）デバイスがあるセキュア ドメイン内の SDConnector を実行するホストの IP アドレスを指定します。

デフォルト： 0.0.0.0

model_handle

（オプション）既存のデバイス モデルを識別するために model_handle を指定します。

注: model_handle を指定する場合、ip_dnsname の値は無視されます。

model_type

（オプション）デバイスをモデリングするために使用される CA Spectrum モデル タイプ。 このデバイス モデル は .modelinggatewayresource.xml ファイルで定義されている任意のインテリジェントなモデル タイプとすることができます。

注: 有効な IP アドレスまたは DNS 名を指定済みの場合、ここで値を指定する必要はありません。

community_string

（オプション）デバイスのコミュニティ文字列を指定します。

注: community_string が含まれていない場合、CA Spectrum は、デバイスを作成するために最初の SNMP コミュニティ文字列値を使用します。これらの値は、VNM モデルの［情報］タブ、自動ディスカバリ制御サブビュー、OneClick のモデリングとプロトコルのオプション サブビューで指定します。

agent_port

（オプション）デバイスの SNMP エージェントと通信するときに使用されるポート番号を制御します。

注: agent_port が含まれていない場合、CA Spectrum は、デバイスを作成するために最初の SNMP ポート値を使用します。 これらの値は、VNM モデルの［情報］タブ、自動ディスカバリ制御サブビュー、OneClick のモデリングとプロトコルのオプション サブビューで指定します。

is_managed

（オプション）true に設定された場合、デバイス モデルをメンテナンス モードにします。

デフォルト： True

poll_interval

（オプション）フラグが POLLED として立てられたデバイス モデルのすべての属性を SpectroSERVER が読み取る間隔を秒単位で指定します。

log_ration

（オプション）ポーリング結果をデータベースに記録する前に発生する、デバイスの SpectroSERVER ポーリングの数を指定します。

poll_status

（オプション）ポーリング ステータスを false に設定することによりデバイスの SpectroSERVER ポーリングを無効にします。

model_name

（オプション）モデルの名前を指定します。

DeviceType

（オプション）モデルのデバイス タイプを指定します。

注: デバイス タイプの詳細については、「認定ユーザ ガイド」を参照してください。

reconfig

（オプション）デバイス モデルを再設定するために SpectroSERVER に Modeling Gateway がアクションを送信するかどうかを指定します。

discover_connections

（オプション） true に設定すると、モデル接続を自動的にマップするために新しく作成された任意のデバイス モデルでディスカバリを実行します。

EventModel

構文

親エレメント：

- Topology_Container
- Left_Model
- Right_Model

子エレメント： なし

ルール： 該当なし

使用法

Southbound Gateway 統合で使用する EventModel モデルをインポートするには、EventModel エレメントを使用します。 Southbound Gateway の詳細については、「Southbound Gateway Toolkit Guide」を参照してください。

属性

model_name

インスタンス化または識別されるモデルの一意の名前を指定します。

unique_id

この EventModel モデルが表すイベント ソースを一意に定義するために使用される識別子を指定します。**注**: 詳細については、「Southbound Gateway Toolkit Guide」を参照してください。

model_handle

（オプション）既存のデバイス モデルを識別するために model_handle を指定します。

Security_String

EventModel モデルのセキュリティ文字列を指定します。

デフォルト： public

manager_name

（オプション）Southbound Gateway を使用しているサードパーティ アプリケーションの名前を指定します。

注: ここでリストされないアプリケーションにはデフォルト値を使用します。

デフォルト： 0

1

NetMentor

2

SSM

3

Omni2000

GenericView

構文

親エレメント： Import

子エレメント：

- GenericView_Container
- Device

ルール： GenericView エレメントには、任意の数のこれらの子エレメントを含めることができます。

使用法

トポロジ ビューおよび場所ビュー以外のカスタマイズされた階層ビューを作成するには、GenericView エレメントを使用します。 統合のニーズに合わせてこのエレメントを変更できます。

属性

containment_relation

このビュー内で格納関係を定義する CA Spectrum 関係を定義する関係ハンドルを指定します。

制限： CA Spectrum 格納関係である必要があります。

model_type

このビューに対して定義されるトップ コンテナ モデルを表すモデル タイプを指定します。 .modelinggatewayresource.xml ファイルで、この model_type をモデル ハンドルと共に指定する必要があります。

name

GenericView 階層で最も高いインスタンス化されたコンテナ モデルの一意の名前を指定します。

complete_topology

（オプション）true に設定すると、GenericView ビューの未指定の既存のコンテナおよびデバイス モデルが破棄されます。また、そのビューのサブコンテナにあるこれらのモデルが破棄されます。

GenericView_Container

構文

親エレメント： GenericView

子エレメント：

- GenericView_Container
- Device

ルール： GenericView_Container エレメントには、任意の数の子エレメントを含めることができます。

使用法

Generic ビューでコンテナ モデルを作成するには、GenericView_Container エレメントを使用します。 GenericView と GenericView_Container の両方のエレメントは、カスタマイズされたビューを作成するために使用されます。そのため、インテグレータとして、このコンテナを使用するタイミングと方法を決定します。

属性

name

インスタンス化または識別されるモデルの名前を指定します。 model_type 属性および name 属性は GenericView_Container を一意に識別するために必須です。 デフォルトでは、この属性は CA Spectrum モデル名属性（属性 ID 0x1006e）の値を設定するために使用されます。ただし、この属性は、.modelinggatewayresource.xml ファイルで他の任意の属性に変更できます。 名前が別の属性にマップするように、.modelinggatewayresource.xml を変更できます。 この場合、その新しい属性が（モデル タイプと共に）コンテナを識別するために使用されます。この動作により、2 つのコンテナが同じモデル名を持ちます。

model_type

モデルを作成するために使用される CA Spectrum モデル タイプを指定します。.modelinggatewayresource.xml ファイルで、この model_type をモデル ハンドルと共に指定する必要があります。model_type 属性および name 属性は GenericView_Container を一意に識別するために必須です。

containment_relation

（オプション）コンテナ内の Generic_Container とモデルの間に存在する CA Spectrum 関係の名前。 この属性の値が指定されない場合、親モデルの包含関係が継承されます。

GlobalCollection

構文

親エレメント： Import

子エレメント：

- Device
- Topology_Container
- Location_Container

使用法

GlobalCollection モデルを表します。

属性

name

このグローバル コレクションの名前を指定します。

containment_relation

このビュー内で格納関係を定義する CA Spectrum 関係を定義する関係ハンドルを指定します。

デフォルト： "GlobalCollect"

collectionDescription

（オプション）グローバル コレクションを説明します。

Security_String

（オプション）グローバル コレクション用のセキュリティ文字列を指定します。

インポート

構文

親エレメント： なし

子エレメント：

- Topology
- Location
- GenericView
- Update
- Destroy
- SM_Service_Mgt
- Correlation
- GlobalCollection

ルール： Import エレメントには、これらの子エレメントの 1 つを含めることができます。

使用法

Import エレメントはルート エレメントであり、各入力ファイルに含まれている必要があります。

属性

model_activation_time

各デバイス モデルのアクティブ化で許可されている最大の分数を指定します。

データ型： 文字

デフォルト： 5 分

Left_Model

構文

親エレメント： Association

子エレメント：

- Device
- Port
- Topology_Container
- Location_Container
- EventModel
- Model

ルール： Left_Model エレメントには 1 つの子エレメントのみを含めることができます。

使用法

Left_Model エレメントは関連付けで左側のモデルを定義します。

属性

なし。

List_Value

構文

親エレメント： Model_Attr

子エレメント： なし

ルール： 該当なし

使用法

CA Spectrum リスト属性値を指定するには、List_Value エレメントを使用します。

属性

なし。

Location

構文

親エレメント： Import

子エレメント：

- Location_Container
- Device

ルール： Location エレメントには、任意の数の子エレメントを含めることができます。

使用法

OneClick の場所ビュー（ワールド トポロジ）でモデルを作成することを指定するには、Location エレメントを使用します。

属性

complete_topology

（オプション） true に設定すると、場所ビューまたは任意のサブコンテナの未指定の既存のコンテナおよびデバイス モデルは、インポート中に破棄されます。

デフォルト： False

データ タイプ： ブール

Location_Container

構文

親エレメント：

- Location
- Location_Container
- Left_Model
- Right_Model
- GlobalCollection

子エレメント：

- Location_Container
- Device

ルール： Location_Container エレメントには、任意の数の子エレメントを含めることができます。

使用法

場所ビュー内のモデルおよび他の場所コンテナをグループ化するために使用される Location_Container モデルを作成または指定するには、Location_Container エレメントを使用します。

属性

name

インスタンス化または識別されるモデルの名前。model_type 属性および name 属性は、Location_Container を一意に識別するのために必須です。
デフォルトでは、この属性は CA Spectrum モデル名属性（属性 ID 0x1006e）の値を設定するために使用されます。 ただし、この属性は、.modelinggatewayresource.xml ファイルで他の任意の属性に変更できます。 名前が別の属性にマップするように、.modelinggatewayresource.xml を変更できます。 この場合、その新しい属性が（モデル タイプと共に）コンテナを識別するために使用されます。この動作により、2 つのコンテナが同じモデル名を持ちます。

model_type

作成するモデルのタイプを示します。 model_type 属性および name 属性は、Location_Container を一意に識別するのために必須です。 可能な値には以下のものが含まれます。

- Country
- Region
- Site
- Building
- Floor
- Section
- Room

model_handle

（オプション）モデルを識別するために使用できます。 model_handle を指定する場合、name と model_type の値は無視されます。

Security_String

（オプション） CA Spectrum ユーザによるモデルへのアクセス要件を定義します。 各セキュリティ文字列は 1 つ以上の Security Community エントリから構成され、モデルに割り当てられます。

model_name

（オプション）モデルの名前を変更するには、name 属性および model_name 属性を使用します。 name 属性は古い名前を指定し、model_name 属性は新しい名前を指定します。

model_modify_author

（オプション） CA Spectrum 属性 mdl_modfy_athr にデータを書き込みます。

complete_topology

（オプション） true に設定すると、場所ビューまたは任意のサブコンテナの未指定の既存のコンテナおよびデバイス モデルは、インポート中に破棄されます。

デフォルト： False

Model_Attr

構文

親エレメント： Correlation_Domain

子エレメント： List_Value

ルール： Model_Attr エレメントには、任意の数の子エレメントを含めることができます。

使用法

値に複数行のテキストまたは値のリストが含まれる CA Spectrum 属性を指定するには、Model_Attr エレメントを使用します。

属性

attr_id

指定する属性の CA Spectrum 属性 ID を示します。

データ型： 文字

Model

構文

親エレメント：

- Left_Model
- Right_Model

子エレメント： なし

使用法

任意の CA Spectrum モデルを表すには、Model エレメントを使用します。

属性

name

このモデルの名前を指定します。

model_type

このモデルのモデル タイプを指定します。

model_handle

（オプション）このモデル用の model_handle を指定します。model_handle 値が指定される場合、name と model_type の値は無視されます。

MonitorPolicy_Attr

構文

親エレメント：

- SM_Service
- SM_AttrMonitor

子エレメント： なし

ルール： 該当なし

使用法

MonitorPolicy_Attr エレメントは CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

なし。

Port

構文

親エレメント：

- Device
- Left_Model
- Right_Model
- Correlation_Domain
- SM_Service
- SM_AttrMonitor

子エレメント： なし

ルール： 該当なし

使用法

Port エレメントは、ポート レベルで接続を指定するか、ポート属性を更新するために使用されます。更新される場合、親 Device エレメントは Update エレメントに含まれます。 接続を指定する場合、親 Device エレメントは Connection エレメントに含まれます。

属性

identifier_name

ポートを一意に識別するために identifier_value 属性と共に動作します。identifier_name は、［有効な値］列に表示されているいずれかの MIB OID 名となります。 portID 値は、その Component_OID 属性（0x1006a）によってポートを識別するために使用できます。 Port エレメントがフレーム リレー仮想回路を表す場合は、frCircuitTableInstance を使用します。 Port エレメントが ATM 仮想チャネルまたはパス リンクを表す場合は、atmVclTableInstance を使用します。

portID 値は、その Component_OID 属性（0x1006a）によってポートを識別するために使用できます。Port エレメントがフレーム リレー仮想回路を表す場合は、frCircuitTableInstance を使用します。Port エレメントが ATM 仮想チャネルまたはパス リンクを表す場合は、atmVclTableInstance を使用します。 可能な値には以下のものが含まれます。

- ifIndex
- ipAddress
- ifPhysAddress
- ifName
- ifAlias
- model_name
- portDescription
- portID
- frCircuitTableInstance
- atmVclTableInstance
- atmVplTableInstance

identifier_value

identifier_name の選択内容の値を指定します。

model_handle

（オプション）既存のモデルを識別するために model_handle を指定します。

注: model_handle を指定する場合、identifier_name と identifier_value の値は無視されます。

ip_dnsname

（オプション）ポート モデルの IP アドレスまたは DNS 名を指定します。 ポート モデルが SNMP 通信をサポートしない場合、指定された model_type と共に、ここで一意の文字列を使用できます。

model_name

（オプション）更新するモデルの名前を指定します。

circuit_id

（オプション） ATM 接続またはフレーム リレー接続に関係する回路を ID で識別します。

circuit_name

（オプション） ATM 接続またはフレーム リレー接続に関係する回路を名前で識別します。

log_ratio

（オプション）ポーリング結果をデータベースに記録する前に発生するポート モデル ポーリングの数を指定します。

poll_interval

（オプション）フラグが POLLED として立てられたポート モデルのすべての属性を、SpectroSERVER が読み取る間隔を秒単位で指定します。

poll_status

（オプション）ポーリング ステータスを False に設定することにより、管理者がポート モデル ポーリングを無効にするようにできます。

Right_Model

構文

親エレメント： Association

子エレメント：

- Device
- Port
- Topology_Container
- Location_Container

- EventModel
- Model

ルール： Right_Model エレメントには 1 つの子エレメントのみを含めることができます。

使用法

Right_Model エレメントは関連付けの右側のモデルを定義します。

属性

なし。

RTM_Test

構文

親エレメント：

- SM_Service
- SM_AttrMonitor

子エレメント： なし

ルール： 該当なし

使用法

RTM_Test エレメントは CA Spectrum Service Manager と共に使用されます。使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name *（必須）*	文字	N/A	N/A
model_type	文字	RTM_Test	N/A

Schedule

構文

親エレメント： Device

子エレメント： なし

使用法

特定のデバイス モデル用のメンテナンス モード スケジュールを作成するには、Schedule エレメントを使用します。

属性

name

スケジュールの名前を指定します。

SCHED_Recurrence

デバイス モデルがメンテナンス モードになる頻度を指定します。

1

常時（24 x 7）

2

日単位

3

週単位

4

月単位

5

年単位

デフォルト： 1

SCHED_Start_Hour

（オプション）デバイスがメンテナンス モードになる時間を指定します。

制限： 0 ～ 23

SCHED_Start_Minute

（オプション）デバイスがメンテナンス モードになる分を指定します。

制限： 0 ～ 59

SCHED_Start_DoW

（オプション）デバイスがメンテナンス モードになる曜日を指定します。

0

日曜日

1

月曜日

2

火曜日

3

水曜日

4

木曜日

5

金曜日

6

土曜日

SCHED_Start_DoM

（オプション）デバイスがメンテナンス モードになる日を指定します。

制限： 1 ～ 31

SCHED_Start_Month

（オプション）デバイスがメンテナンス モードになる月を指定します。

0

1 月

1

2 月

2

3 月

3

4 月

4

5 月

5

6 月

6

7 月

7

8 月

8

9 月

9

10 月

10

11 月

11

12 月

SCHED_Duration

（オプション）デバイスがメンテナンス モードになる時間の長さを指定します（秒単位で定義）。

デフォルト： 0

SCHED_Recurrence_Multiplier

（オプション）定期保守モードの開始間隔を決定する単位（日、週、月、年）の数値を指定します。

デフォルト： 1

SCHED_Daily_Repeat_Limit

（オプション）各期間中に定期保守（SCHED_Start_Hour と SCHED_Start_Minute によって指定）を繰り返し実行する連続の日数を指定します。 この属性は、週、月、または年単位の繰り返しのみに適用されます。

SM_AttrMonitor

構文

親エレメント： SM_Service

子エレメント： なし

ルール： 該当なし

使用法

SM_AttrMonitor エレメントは CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name *(必須)*	文字	N/A	N/A
containment_relation	文字	N/A	SLMMonitors SLMWatchesContainer
AttrToWatch	文字	N/A	N/A
MonitorPolicy_ID	文字	N/A	N/A
is_managed	ブール値	N/A	True False
Generate_Service_Alarms	ブール値	N/A	True False
model_type	文字	SM_AttrMonitor	N/A

SM_Customer

構文

親エレメント：

- SM_CustomerGroup
- CustomerManager

子エレメント： なし

ルール： 該当なし

使用法

SM_Customer エレメントは CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name *（必須）*	文字	N/A	N/A
containment_relation	文字	N/A	SlmAgreesTo SlmUses
Security_String	文字	N/A	N/A
CustomerID	文字	N/A	N/A
Criticality	文字	N/A	N/A
CustomerField4	文字	N/A	N/A
CustomerField5	文字	N/A	N/A
CustomerField6	文字	N/A	N/A
CustomerField7	文字	N/A	N/A
Contact_Name	文字	N/A	N/A
Contact_Title	文字	N/A	N/A
Contact_Location	文字	N/A	N/A
Email_Address	文字	N/A	N/A

属性	データ タイプ	デフォルト値	有効な値
Phone_Number	文字	N/A	N/A
Mobile_Phone_Number	文字	N/A	N/A
Pager_Number	文字	N/A	N/A
Fax_Number	文字	N/A	N/A
User_Defined_1	文字	N/A	N/A
User_Defined_2	文字	N/A	N/A
User_Defined_3	文字	N/A	N/A
User_Defined_4	文字	N/A	N/A
Secondary_Contact_Name	文字	N/A	N/A
Secondary_Contact_Location	文字	N/A	N/A
Secondary_Email_Address	文字	N/A	N/A
Secondary_Phone_Number	文字	N/A	N/A
Secondary_Mobile_Phone_ Number	文字	N/A	N/A
Secondary_Pager_Number	文字	N/A	N/A
Secondary_Fax_Number	文字	N/A	N/A
Secondary_User_Defined_1	文字	N/A	N/A
Secondary_User_Defined_2	文字	N/A	N/A
Secondary_User_Defined_3	文字	N/A	N/A

属性	データ タイプ	デフォルト値	有効な値
Secondary_User_Defined_4	文字	N/A	N/A
model_type	文字	SM_Customer	N/A

SM_CustomerGroup

構文

親エレメント：

- CustomerManager
- SM_CustomerGroup

子エレメント：

- SM_CustomerGroup
- SM_Customer

ルール： SM_CustomerGroup エレメントには、任意の数のこれらの子エレメントを含めることができます。

使用法

SM_CustomerGroup エレメントは CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name *（必須）*	文字	該当なし	N/A
containment_relation	文字	Groups_Customer	N/A

属性	データ タイプ	デフォルト値	有効な値
model_type	文字	SM_CustomerGroup	N/A

SM_Guarantee

構文

親エレメント： SM_SLA

子エレメント： なし

ルール： 該当なし

使用法

SM_Guarantee エレメントは CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name *（必須）*	文字	N/A	N/A
containment_relation	文字	SlmIsMeasuredBy	N/A
is_managed	ブール値	N/A	True False
DegradedTimeViolationLevel	文字	N/A	N/A
DegradedTimeWarningLevel	文字	N/A	N/A
DownTimeViolationLevel	文字	N/A	N/A

属性	データ タイプ	デフォルト値	有効な値
DownTimeWarningLevel	文字	N/A	N/A
LorTimeViolationLevel	文字	N/A	N/A
LorTimeWarningLevel	文字	N/A	N/A
model_type	文字	SM_Guarantee	N/A

SM_LatencyMon

構文

親エレメント：

- SM_Guarantee
- SM_AttrMonitor
- SM_Service

子エレメント：

- Topology_Container
- MonitorPolicy_Attr

ルール： SM_LatencyMon エレメントには、任意の数のこれらの子エレメントを含めることができます。

使用法

SM_LatencyMon エレメントは CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name *（必須）*	文字	N/A	N/A
containment_relation	文字	N/A	SlmMonitors SlmWatchesContainer
is_managed	ブール値	N/A	True False
DefaultMaxRTT	文字	N/A	N/A
DefaultMeasureInterval	文字	N/A	N/A
mode_type	文字	SM_LatencyMon	N/A

SM_Service

構文

親エレメント：

- SM_Service
- SM_ServiceMgr

子エレメント：

- SM_Service
- SM_AttrMonitor

ルール： SM_Service エレメントには、任意の数のこれらの子エレメントを含めることができます。

使用法

SM_Service エレメントは CA Spectrum Service Manager と共に使用されます。使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name *（必須）*	文字	N/A	N/A
Criticality	文字	N/A	N/A
containment_relation	文字	N/A	SlmMonitors SlmWatchesContainer
AttrToWatch	文字	N/A	N/A
MonitorPolicy_ID	文字	N/A	N/A
is_managed	ブール値	N/A	True False
Generate_Service_Alarms	ブール値	N/A	True False
Security_String	文字	N/A	N/A
model_type	文字	SM_Service	N/A

SM_Service_Mgt

構文

親エレメント： Import

子エレメント：

- SM_ServiceMgr
- CustomerManager
- SM_SLA_Mgr

ルール： これらの子エレメントの単一のインスタンスのみが SM_Service_Mgt に存在できます。

使用法

SM_Service_Mgt エレメントは CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name *（必須）*	文字	サービス管理	N/A
containment_relation	文字		N/A
model_type	文字		N/A

SM_ServiceMgr

構文

親エレメント： SM_Service_Mgt

子エレメント： SM_Service

ルール： SM_ServiceMgr エレメントには、任意の数のこの子エレメントを含めることができます。

使用法

SM_ServiceMgr エレメントは CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name	文字	N/A	N/A
containment_relation	文字	SlmContains	N/A
model_type	文字	SM_ServiceMgr	N/A

SM_SLA

構文

親エレメント： SM_SLA_Mgr

子エレメント： SM_Guarantee

ルール： SM_SLA エレメントには、任意の数のこの子エレメントを含めることができます。

使用法

SM_SLA エレメントは CA Spectrum Service Manager と共に使用されます。使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name *（必須）*	文字	N/A	N/A
containment_relation	文字	SlmContains	N/A
is_managed	ブール値	N/A	True False
Security_String	文字	N/A	N/A
model_type	文字	SM_SLA	N/A

SM_SLA_Mgr

構文

親エレメント： SM_Service_Mgt

子エレメント： SM_SLA

ルール： SM_SLA_Mgr エレメントには、任意の数のこの子エレメントを含めることができます。

使用法

SM_SLA_Mgr エレメントは CA Spectrum Service Manager と共に使用されます。 使用の詳細については、「Service Manager ユーザ ガイド」を参照してください。

属性

属性の定義については、「Service Manager ユーザ ガイド」を参照してください。

属性	データ タイプ	デフォルト値	有効な値
name	文字		N/A
containment_relation	文字	SlmContains	N/A
model_type	文字	SM_ServiceMgr	N/A

Topology

構文

親エレメント： Import

子エレメント：

- Topology_Container
- Device
- Connection

ルール： Topology エレメントには、任意の数のこれらの子エレメントを含めることができます。

使用法

OneClick トポロジ ビュー（ユニバース トポロジ）でモデルを作成するには、Topology エレメントを使用します。

属性

complete_topology

true に設定すると、トポロジ ビューの未指定の既存のコンテナおよびデバイス モデルはインポート中に破棄されます。 また、そのビューのサブコンテナも破棄されます。

デフォルト： False

discover_connections

true に設定すると、モデルの接続を自動的にマップするため、新しく作成されたデバイス モデルでディスカバリが実行されます。

デフォルト： False

Topology_Container

構文

親エレメント：

- Topology
- Topology_Container
- SM_Service
- SM_AttrMonitor

子エレメント：

- Topology_Container
- Device
- EventModel
- Connection

ルール： Topology_Container エレメントには、任意の数のこれらの子エレメントを含めることができます。

使用法

トポロジ ビューのモデルおよび他のトポロジ コンテナをグループ化するために使用される Topology_Container モデルを作成または指定するには、Topology_Container エレメントを使用します。 有効なモデル タイプは、model_type セクション内の以下のテーブルにリストされます。

属性

model_type

作成するモデルのタイプを示します。 model_type 属性および name 属性は、Topology_Container を一意に識別するために必須です。

- ネットワーク
- LAN
- IPClassA
- IPClassB
- IPClassC
- LAN_802_3
- LAN_803_5
- EventAdmin
- ATM_Network

model_handle

（オプション）既存のモデルを識別するために使用できます。model_handle を指定する場合、model_type と model_name の値は無視されます。

Security_String

（オプション）モデルに割り当てられた CA Spectrum セキュリティ レベルを指定します。

subnet_address

デバイスのサブネット アドレスを指定します。

subnet_mask

（オプション）デバイスの IP アドレスが属するサブネットを決定するマスクを指定します。

model_name

（オプション）コンテナ モデル用のモデル名を指定します。

trapIPAddress

（オプション） EventAdmin モデルのみが対象です。

x_coordinate

（オプション）トポロジでモデルの x 座標を指定します。

y_coordinate

（オプション） トポロジでモデルの y 座標を指定します。

complete_topology

（オプション） True に設定すると、この Topology_Container の未指定の既存のコンテナとデバイス モデル、およびそのサブコンテナは、インポート中に破棄されます。

デフォルト： False

discover_connections

（オプション） CA Spectrum が、このモデルに接続されたデバイスを検出し、モデリングするかどうかを指定します。

Update

構文

親エレメント： Import

子エレメント：

- Topology_Container
- Location_Container
- GenericView_Container
- Connection
- Device
- Model
- EventModel
- SM_Service
- SM_AttrMonitor
- SM_LatencyMon
- SM_ConnectMon
- SM_SLA
- SM_Guarantee

- SM_Customer
- Association

ルール： Update エレメントには、任意の数のこれらの子エレメントを含めることができます。

使用法

任意のデバイス、コンテナ、またはポート サブエレメント用の属性を更新するには、Update エレメントを使用します。

注: Device エレメント内で Port エレメントを使用する場合を除いて、このエレメントを使用する場合、階層指定は許可されません。

属性

なし。

付録 B: 文書型定義ファイル

このセクションには、インポート用の XML エレメントおよび属性を定義する、文書型定義（DTD）が含まれます。

注: 以下のコードはこのファイルの最新バージョンではない可能性があります。DTD の最新バージョンについては、Modeling Gateway Toolkit に用意されている実際のファイルを使用してください。 このファイルは SS-Tools ディレクトリにあり、.modelinggateway.dtd という名前です。

```
<!-- ********************************************************** -->
<!-- ルートの Import エレメントには、Topology、Location、-->
<!-- GenericView、Update、Destroy の各エレメント、-->
<!-- SM_Service_Mgt、相関、およびグローバル コレクションの 0 または 1 が含まれます。      -->
<!--                                                              -->
<!-- このエレメントには 1 つの属性 model_activation_time があります。-->
<!-- これは、各デバイス モデルのアクティブ化のための最大の待機時間を-->
<!--分数で示します。 デフォルトで 5 分になります。                    -->
<!-- ********************************************************** -->

<!ELEMENT Import ( ( Topology       |
                     Location       |
                     GenericView    |
                     Update         |
                     Destroy        |
                     SM_Service_Mgt |
                     Correlation    |
                     GlobalCollection )*) >

<!ATTLIST Import model_activation_time  CDATA     "5">

<!-- ************************************************** -->
<!-- トポロジ ビューに使用される Topology エレメントには、 -->
<!-- 任意の数の Topology_Container、      -->
<!-- Device、および Connection エレメントが含まれます。          -->
<!--                                                    -->
<!-- このエレメントには、2 つの属性 complete_topology   -->
<!-- および discover_connection があります。                -->
<!--                                                    -->
<!-- ************************************************** -->

<!ELEMENT Topology ((Topology_Container | Device | Connection)*) >
<!ATTLIST Topology
        complete_topology    (false | true)     #IMPLIED
        discover_connections (false | true)     #IMPLIED>
```

```
<!-- ************************************************** -->
<!-- Topology_Container エレメントは、-->
<!-- トポロジ コンテナ モデルに使用され、任意の数の -->
<!-- Topology_Container、Device、および Connection エレメントが含まれます。  -->
<!--                                                    -->
<!-- "model_type" および "name" は、一意に  -->
<!-- Topology_Container モデルを識別するために必須の属性です。       -->
<!--                                                    -->
<!-- "model_handle" もモデルを識別するために使用できます。  -->
<!-- "model_handle" を指定すると、"name"  -->
<!-- および "model_type" の値は無視されます。                    -->
<!--                                                    -->
<!-- "trapIPAddress" 属性は    -->
<!-- EventAdmin モデルのみに使用します。                                  -->
<!--                                                    -->
<!-- ************************************************** -->

<!ELEMENT Topology_Container ((Topology_Container |
                               Device |
                               EventModel |
                               Connection )*) >

<!ATTLIST Topology_Container
        name          CDATA                 #REQUIRED
        model_type          ( Network      |
                              Lan          |
                              IPClassA     |
                              IPClassB     |
                              IPClassC     |
                              LAN_802_3    |
                              LAN_803_5    |
                              EventAdmin   |
                              ATM_Network  ) #REQUIRED

        model_handle          CDATA          #IMPLIED
        Security_String       CDATA          #IMPLIED
        subnet_address        CDATA          #IMPLIED
        subnet_mask           CDATA          #IMPLIED
        model_name            CDATA          #IMPLIED
        trapIPAddress         CDATA          #IMPLIED
        x_coordinate          CDATA          #IMPLIED
        y_coordinate          CDATA          #IMPLIED
        complete_topology    (false | true)  #IMPLIED
        discover_connections (false | true)  #IMPLIED >
```

```
<!-- ************************************************** -->
<!-- Location エレメントは場所ビューに使用され、          -->
<!-- 任意の数の Location_Container および                 -->
<!-- Device エレメントを含みます                                  -->
<!--                                                    -->
<!-- このエレメントには属性 complete_topology があります。          -->
<!-- ************************************************** -->

<!ELEMENT Location ( (Location_Container  | Device )* ) >
<!ATTLIST Location
        complete_topology    (false | true)     #IMPLIED>

<!-- ************************************************** -->
<!-- Location_Container エレメントは場所コンテナに     -->
<!-- 使用され、任意の数の           -->
<!-- Location_Container および Device エレメントを含めることができます。              -->
<!--                                                    -->
<!-- "model_type" および "name" は、一意に  -->
<!-- Location_Container モデルを識別するための必須の属性です       -->
<!--                                                    -->
<!-- "model_handle" もモデルを識別するために使用できます。   -->
<!-- "model_handle" を指定すると、"name"  -->
<!-- および "model_type" の値は無視されます。                      -->
<!-- ************************************************** -->

<!ELEMENT Location_Container ( ( Location_Container | Device )* )>

<!ATTLIST Location_Container
        name          CDATA                 #REQUIRED
        model_type        ( Country    |
                            Region     |
                            Site       |
                            Building   |
                            Floor      |
                            Section    |
                            Room       )     #REQUIRED

        model_handle          CDATA          #IMPLIED
        Security_String       CDATA          #IMPLIED
        model_name            CDATA          #IMPLIED
        model_modify_author CDATA            #IMPLIED
        complete_topology   (false | true)   #IMPLIED >

<!-- ********************************************************** -->
<!-- Device エレメントはデバイス モデルに使用されます。                    -->
<!--                                                          -->
```

```
<!-- "ip_dnsname" は、デバイス モデルを一意に識別する       -->
<!-- ための必須の属性です。                                  -->
<!--                                                        -->
<!-- "model_handle" は、デバイス モデルを識別するためにも使用できます。    -->
<!-- "model_handle" を指定すると、"ip_dnsname" の値は      -->
<!-- 無視されます。                                         -->
<!--                                                        -->
<!-- 注意 :                                                 -->
<!--                                                        -->
<!-- 1. 属性 is_managed を false に設定すると、デバイスに -->
<!--    接続できなくなります。  そのため、model_type 属性を -->
<!--    設定する必要があります。                                    -->
<!--                                                        -->
<!-- 2. Device には 0、1、またはそれ以上の数のポートを、       -->
<!--    以下の状況で含めることができます。                    -->
<!--                                                        -->
<!-- (a) Device が Container または Destroy エレメントにある場合、 -->
<!--    Port エレメントは必要ではありません。  Port       -->
<!-- エレメントが指定されている場合、それらは無視されます。      -->
<!--                                                        -->
<!-- (b) Device が Connection エレメントにある場合、1 つの Port    -->
<!--     エレメントのみが許可されます。                          -->
<!--                                                        -->
<!-- (c) ポートを更新するために Device が Update エレメントに  -->
<!--     含まれている場合、複数の Port エレメントを使用できます。    -->
<!--                                                        -->
<!-- (d) デバイス タグで discover_connections="true" を   -->
<!--     指定する場合は、そのデバイスの親コンテナでもこれを       -->
<!--     指定しないでください。  これにはパフォーマンスと効率に関連する問題が  -->
<!--     あり、この属性をコンテナで指定すると、        -->
<!--     Spectrum はそのコンテナの各モデルで     -->
<!--     接続を検出します。                                  -->
<!--                                                        -->
<!-- ********************************************************** -->

<!ELEMENT Device ( Port* | Schedule ) >

<!ATTLIST Device
        ip_dnsname        CDATA             #REQUIRED
        secdomain_ipname  CDATA             #IMPLIED

        model_handle          CDATA         #IMPLIED
        model_type        CDATA             #IMPLIED
        community_string  CDATA             #IMPLIED
       agent_port        CDATA             #IMPLIED
        poll_interval     CDATA             #IMPLIED
        log_ratio         CDATA             #IMPLIED
        model_name            CDATA         #IMPLIED
        DeviceType        CDATA             #IMPLIED
```

```
        x_coordinate          CDATA          #IMPLIED
        y_coordinate          CDATA          #IMPLIED
         is_managed        (true  | false)    #IMPLIED
         reconfig          (true  | false)    #IMPLIED
         poll_status       (true  | false)    #IMPLIED
         discover_connections (false | true)  #IMPLIED >

<!-- ********************************************************** -->
<!-- Port エレメントはデバイス ポート モデルに使用されます。                    -->
<!--                                                            -->
<!-- "identifier_name" および "identifier_value" は、     -->
<!-- 一意にポート モデルを識別するための必須の属性です。                    -->
<!--                                                            -->
<!-- "model_handle" もポート モデルを識別するために使用できます。      -->
<!-- "model_handle" を指定した場合、identifier_name  -->
<!-- および identifier_value の値は無視されます。                          -->
<!-- ********************************************************** -->

<!ELEMENT Port ( Port* ) >

<!ATTLIST Port
        identifier_name ( portDescription        |
                          model_name             |
                          ifIndex                |
                          ipAddress              |
                          ifPhysAddress          |
                          ifName                 |
                          ifAlias                |
                          portID                 |
                          frCircuitTableInstance |
                          atmVclTableInstance    |
                          atmVplTableInstance    ) #REQUIRED
        identifier_value  CDATA                    #REQUIRED

        model_handle      CDATA                    #IMPLIED
        ip_dnsname        CDATA                    #IMPLIED
        model_type        CDATA                    #IMPLIED
        model_name        CDATA                    #IMPLIED
        circuit_id        CDATA                    #IMPLIED
        circuit_name      CDATA                    #IMPLIED
        log_ratio         CDATA                    #IMPLIED
        poll_interval     CDATA                    #IMPLIED
        poll_status       (false | true)           #IMPLIED >
```

```
<!-- ********************************************************************** -->
<!-- スケジュール モデルを表します                                          -->
<!--                                                                        -->
<!-- SCHED_Recurrence には、以下の値があります                              -->
<!--                                                                        -->
<!--  1 = 常時（24 x 7）                                                    -->
<!--  2 = 日単位                                                            -->
<!--  3 = 週単位                                                            -->
<!--  4 = 月単位                                                            -->
<!--  5 = 年単位                                                            -->
<!-- 6 = 一度のみ                                                           -->
<!--                                                                        -->
<!--  SCHED_Start_Hour： 値の範囲 0 ～ 23                                   -->
<!--  SCHED_Start_Minute： 値の範囲 0 ～ 59                                 -->
<!--  SCHED_Start_DoW： 週単位の繰り返しの曜日     -->
<!--                  （範囲は 0 ～ 6、日曜日は 0）                         -->
<!-- SCHED_Start_DoM： 月単位および年単位の繰り返しの   >--
<!--                  日                                                    -->
<!-- SCHED_Start_Month： 年単位の繰り返しの場合の範囲 0 ～ 11          >--
<!--                  1 月は 0                                              -->
<!-- SCHED_Duration： 秒単位のアクティブな期間。 0 （デフォルト）である可能性があります -->
<!--  SCHED_Recurrence_Multiplier: アクティブな各期間の          -->
<!--                               間隔を決定する -->
<!--                               繰り返し単位の数。                -->
<!--                              デフォルトは 1 です。                      -->
<!-- SCHED_Daily_Repeat_Limit： 各繰り返し期間の開始時に日単位のスケジュール    >--
<!--                            を繰り返す連続した日数                  -->
<!--                            （SCHED_Start_Hour と SCHED_Start_Minute     -->
<!--                            で指定）       -->
<!--                            週単位、月単位、または         -->
<!--                            年単位のみに適用されます。                  -->
<!-- SCHED_DayBitMask： WEEKLY Schedule をアクティブにする>--
<!--                  曜日。   有効な値：                          -->
<!--                  日曜日 = 1、                                          -->
<!--                  月曜日 = 2、                                          -->
<!--                  火曜日 = 4、                                         -->
<!--                  水曜日 = 8、                                        -->
<!--                  木曜日 = 16、                                        -->
<!--                  金曜日 = 32、                                         -->
<!--                  土曜日 = 64                                         -->
<!--                  たとえば、月曜日、水曜日、および金曜日の場合、値は、 -->
<!--                  2+8+32=42 となります                                  -->
<!--  SCHED_Start_MoY、                                                     -->
<!--  SCHED_START_YEAR、                                                    -->
```

```
<!--  SCHED_START_DAY: SCHED_START_MONTH と共に使用して          -->
<!--                   将来の日付でスケジュールがアクティブになることを     -->
<!--                   示します。   SCHED_START_YEAR および         -->
<!--             SCHED_START_DAY の両方はゼロ以外である必要があります。              -->
<!--                   それ以外の場合、スケジュールは通常どおり動作し、       -->
<!--                 早ければ今日にもアクティブになります。                        -->
<!--                   SCHED_START_YEAR は     -->
<!--                   1900 年以降の年数として指定します。                    -->
<!-- SCHED_Description :  このスケジュールの説明。                        -->
<!--                                                                      -->
<!-- ******************************************************************** -->

<!ELEMENT Schedule ( #PCDATA) >

<!ATTLIST Schedule
        name                        CDATA               #REQUIRED
        SCHED_Recurrence            ( 1 | 2 | 3 |
                                        4 | 5 | 6 )        #REQUIRED
        SCHED_Daily_Repeat_Limit    CDATA               #REQUIRED
        SCHED_Duration              CDATA               #REQUIRED
        SCHED_Recurrence_Multiplier CDATA               #REQUIRED
        SCHED_Start_DoM             CDATA               #REQUIRED
        SCHED_Start_DoW             CDATA               #REQUIRED
        SCHED_Start_Hour            CDATA               #REQUIRED
        SCHED_Start_Minute          CDATA               #REQUIRED
        SCHED_Start_Month           CDATA               #REQUIRED
        SCHED_Start_Day             CDATA               #REQUIRED
        SCHED_DayBitMask            CDATA               #REQUIRED
        SCHED_Start_Year            CDATA               #REQUIRED
        SCHED_Start_MoY             CDATA               #REQUIRED
        SCHED_Description           CDATA               #REQUIRED
        >
<!-- ************************************************************** -->
<!-- ElementModel を表すために使用されるエレメント。                          -->
<!--                                                                  -->
<!-- パフォーマンス上の理由で、一意の ID を指定する必要があります。        -->
<!-- ************************************************************** -->

<!ELEMENT EventModel ( #PCDATA ) >

<!ATTLIST EventModel
        model_name        CDATA        #REQUIRED
        unique_id         CDATA        #REQUIRED
        model_handle      CDATA        #IMPLIED
        Security_String   CDATA        "public"
        manager_name      CDATA        "0">
```

```
<!-- ************************************************************ -->
<!-- Connection エレメントはデバイス接続を表します。              -->
<!-- このエレメントには、接続に関連する 2 つの Device エレメントが含まれます。    -->
<!-- 各 Device にはゼロまたは 1 つの Port エレメントを含めることができます。      -->
<!--                                                              -->
<!-- Connection には、1 つの属性 create_pipe があります。 通常、ATM 回路リンクに対しては、
-->
<!-- create_pipe を false に設定し、          -->
<!-- ビュー内で多数のパイプを作成しないようにします。 この場合、       -->
<!-- 接続を表示するために ATM Manager を使用できます。 create_pipe 用の設定の決定は、   -->
<!-- ユーザが行います。 デフォルトでは、     -->
<!-- パイプが各接続に対して作成されます。                         -->
<!--                                                              -->
<!-- ************************************************************ -->

<!ELEMENT Connection (Device, Device)>

<!ATTLIST Connection  create_pipe  (true | false)  "true" >

<!-- ************************************************************ -->
<!-- Update エレメントは SPECTRUM モデル属性と  -->
<!-- 関連付けを更新します。                                       -->
<!--                                                              -->
<!-- Update エレメントには、任意の数の Container、       -->
<!-- Device、および Association エレメントを含めることができます。     -->
<!-- ポートを更新するには、Port エレメントは  -->
<!-- Device エレメントの内部に配置し、Device エレメントを     -->
<!-- Update エレメントに配置する必要があります。                  -->
<!-- ************************************************************ -->
```

```
<!ELEMENT Update  ( ( Topology_Container           |
                     Location_Container           |
                     GenericView_Container        |
                     Connection                   |
                     Device                       |
                     EventModel                   |
                     SM_Service                   |
                     SM_AttrMonitor               |
                     SM_LatencyMon                |
                     SM_ConnectMon                |
                     SM_SLA                       |
                     SM_Guarantee                 |
                     SM_Customer                  |
                     Association
                   )* ) >

<!-- ********************************************************** -->
<!-- Destroy エレメント： SPECTRUM モデルと関連付けを破棄します。     -->
<!--                                                            -->
<!-- Destroy エレメントには、破棄する任意の数の      -->
<!-- Container、Device、Connection、および Association -->
<!-- エレメントを含めることができます。                                  -->
<!-- ********************************************************** -->

<!ELEMENT Destroy ( ( Topology_Container           |
                     Location_Container           |
                     GenericView_Container        |
                     Device                       |
                     Connection                   |
                     EventModel                   |
                     SM_Service                   |
                     SM_AttrMonitor               |
                     SM_LatencyMon                |
                     SM_ConnectMon                |
                     SM_SLA                       |
                     SM_Guarantee                 |
                     SM_Customer                  |
                     Association

                   )* ) >
```

```
<!-- ********************************************************** -->
<!-- Association エレメントは、作成または破棄のために 2 つのモデル間の -->
<!-- SPECTRUM 関連付けを定義します。                              -->
<!--                                                            -->
<!-- Association エレメントには 1 つの Left_Model と 1 つの       -->
<!-- Right_Model エレメントが含まれます。                                  -->
<!-- ********************************************************** -->

<!ELEMENT Association ((Left_Model | Right_Model)*) >
<!ATTLIST Association relation CDATA #REQUIRED >


<!-- ********************************************************** -->
<!-- Left_Model エレメントは Spectrum 関連付けの左側のモデルを    -->
<!-- 定義します。                                                -->
<!--                                                            -->
<!-- Left_Model エレメントには 1 つの子エレメントのみを含める    -->
<!-- ことができます。                                                  -->
<!-- ********************************************************** -->

<!ELEMENT Left_Model (Device                    |
                      Port                      |
                      Topology_Container        |
                      Location_Container        |
                      EventModel                |
                      Model
                     ) >



<!-- ********************************************************** -->
<!-- Right_Model エレメントは Spectrum 関連付けの右側のモデルを  -->
<!-- 定義します。                                                -->
<!--                                                            -->
<!-- Right_Model エレメントは 1 つの子エレメントのみを含める     -->
<!-- ことができます。                                                  -->
<!-- ********************************************************** -->
<!ELEMENT Right_Model (Device                   |
                       Port                     |
                       Topology_Container       |
                       Location_Container       |
                       EventModel               |
                       Model
                      ) >


<!-- ********************************************************** -->
<!-- Model エレメントを使用して、任意の SPECTRUM モデルも表すことができます。  -->
<!--                                                            -->
<!-- モデルを定義するには、model_type と name を指定する必要があります。  -->
<!-- モデルを一意に識別するために、"model_type" と "name" を  -->
```

```
<!-- 使用する必要があります。    ただし、"model_handle" を指定する場合、 >--
<!-- "model_type" および "name" の値は使用されません。                    -->
<!-- ********************************************************** -->

<!ELEMENT Model ( #PCDATA ) >
<!ATTLIST Model
          name           CDATA    #REQUIRED
          model_type     CDATA    #REQUIRED
          model_handle   CDATA    #IMPLIED >


<!-- ********************************************************** -->
<!-- カスタマイズされたビューに使用される GenericView エレメントには、       -->
<!-- 任意の数の GenericView_Container および Device    -->
<!-- エレメントを含めることができます。                                                  -->
<!--                                                                -->
<!-- このエレメントには、3 つの必須の属性 containment_relation、 -->
<!-- model_type、および name があります。                                  -->
<!-- ********************************************************** -->

<!ELEMENT GenericView ((GenericView_Container | Device )*) >

<!ATTLIST GenericView
        containment_relation  CDATA          #REQUIRED
        model_type            CDATA          #REQUIRED
        name          CDATA                  #REQUIRED
        complete_topology   (false | true)   #IMPLIED >

<!-- ********************************************************** -->
<!-- GenericView コンテナに使用される GenericView_Container       -->
<!-- エレメントには、任意の数の GenericView_Container、            -->
<!-- および Device エレメントを含めることができます。                  -->
<!--                                                              -->
<!-- このエレメントでは model_type および name 属性が必須です。          -->
<!-- 属性 containment_relation は必須ではありません。  指定しない場合、-->
<!-- 親の containment_relation が          -->
<!-- 継承されます。  Model_Attr は複数行のテキスト文字列 -->
<!-- SPECTRUM 属性、またはリスト属性に使用できます。               -->
<!-- ********************************************************** -->

<!ELEMENT GenericView_Container ( GenericView_Container |
        Device
        )*>
<!ATTLIST GenericView_Container
        name                CDATA          #REQUIRED
        model_type          CDATA          #REQUIRED
        containment_relation CDATA         #IMPLIED >
```

```
<!-- ********************************************************** -->
<!-- Model_Attr は複数行のテキスト文字列またはリストの       -->
<!-- SPECTRUM 属性に使用されます。                                  -->
<!--                                                          -->
<!-- このエレメントで SPECTRUM 属性を指定するには、attr_id が       -->
<!-- 必須です。   このエレメントには、SPECTRUM テキスト文字列属性タイプの     -->
<!-- 複数行のテキスト文字列、または SPECTRUM リスト属性の      -->
<!-- 複数の List_Value エレメントを含めることができます。 -->
<!-- ********************************************************** -->

<!ELEMENT Model_Attr ( #PCDATA | List_Value )* >
<!ATTLIST Model_Attr
                        attr_id   CDATA          #REQUIRED >

<!-- ********************************************************** -->
<!-- List_Value は SPECTRUM リスト属性値に使用されます。       -->
<!--                                                          -->
<!-- Each List_Value には PCDATA が含まれ、リスト属性に        -->
<!-- 1 つのインスタンス値を提供します。                          -->
<!-- ********************************************************** -->
<!ELEMENT List_Value ( #PCDATA ) >

<!-- ********************************************************** -->
<!-- サービス レベル管理トポロジ エレメント。                  -->
<!-- ********************************************************** -->

<!ELEMENT CustomerManager ( SM_Customer |
                                SM_CustomerGroup
                                )*>

<!ATTLIST CustomerManager
        name                  CDATA                #IMPLIED
        containment_relation  ( Groups_Customers )  #IMPLIED
        model_type            ( CustomerManager )   #IMPLIED
        >

<!ELEMENT SM_ServiceMgr ( SM_Service )*>

<!ATTLIST SM_ServiceMgr
        name            CDATA                          #IMPLIED
        containment_relation          ( SlmContains )   #IMPLIED
        model_type                    ( SM_ServiceMgr ) #IMPLIED
        >

<!ELEMENT SM_SLA_Mgr ( SM_SLA )*>
```

```
<!ATTLIST SM_SLA_Mgr
        name                    CDATA                   #IMPLIED
        containment_relation    ( SlmContainsSLAs )     #IMPLIED
        model_type              ( SM_SLA_Mgr )          #IMPLIED
        >

<!ELEMENT SM_Service_Mgt ( CustomerManager |
                        SM_ServiceMgr   |
                        SM_SLA_Mgr
                        )*>

<!ATTLIST SM_Service_Mgt
        name            CDATA                               #IMPLIED
        containment_relation    ( SlmHasServiceComponent )  #IMPLIED
        model_type              ( SM_Service_Mgt )          #IMPLIED
        >

<!-- Correlation エレメントは、ルート モデル Correlation_Manager -->
<!-- （0x10469）を表します。 これには Correlation_Domain エレメントのみを含めることができ、
-->
<!-- 属性はありません。  すべての Correlation_Domains は、CORRELATES         -->
<!-- 関係によって Correlation Manager に関連付けられます。                        -->

<!-- Correlation_Manager は一意のモデルであり、このエレメントは     -->
<!-- 実際には SpectroSERVER にモデルを作成させません。  これは     -->
<!--以前に存在したモデルを表します。                                              -->

<!ELEMENT Correlation ( Correlation_Domain )*>

<!-- Correlation_Domain には任意の数の Device、Port、       -->
<!-- Model_Attr、または GenericView_Container を含めることができます。 それらはすべて   -->
<!--CORRELATES 関係によって Correlation_Domain に関連付けられます。  Correlation_   -->
<!-- Domains にはまた属性がありません。                                          -->

<!ELEMENT Correlation_Domain (  Device              |
                                Port                |
                                Model_Attr          |
                                GenericView_Container
                                )*>

<!ATTLIST Correlation_Domain
        name            CDATA                   #REQUIRED
        >

<!ELEMENT RTM_Test ( #PCDATA) >

<!ATTLIST RTM_Test
        name            CDATA               #REQUIRED
        model_type      ( RTM_Test )        #IMPLIED
        >
```

```
<!ELEMENT SM_Service ( SM_Service         |
                       SM_AttrMonitor     |
                       SM_LatencyMon      |
                       SM_ConnectMon      |
                       Device             |
                       Port               |
                       Topology_Container |
                       RTM_Test           |
                       MonitorPolicy_Attr |
                       Schedule
                       )*>

<!-- ********************************************************************** -->
<!-- SM_Service モデルを表します -->
<!--                                                                        -->
<!-- 重大度は以下のいずれかになります                                       -->
<!--                                                                        -->
<!-- 10 - 低                                                             >--
<!-- 15 - やや低                                                            >--
<!-- 20 - 中                                                             >--
<!-- 25 - やや高                                                            >--
<!-- 10 - 高                                                             >--
<!--                                                                        -->
<!-- AttrToWatch は以下のいずれかになります                                 -->
<!--                                                                        -->
<!-- Condition - ほとんどのモデル、ポリシー 1 ～ 5 に使用できます           -->
<!-- Contact_Status - 通常、デバイス モデル、ポリシー 10 ～ 13 に使用されます   -->
<!-- Port_Status - インターフェース モデル、ポリシー 14 ～ 17 に使用されます    -->
<!-- LatestErrorStatus - RTM_Test モデル、ポリシー 18 ～ 21  -->
<!-- または Response_Time に使用されます                                    -->
<!-- RM_Condition - SM_AttrMonitor または SM_Service のモデル、ポリシー 6 ～ 9 -->
<!-- またはサービス ヘルス                                                  -->
<!--                                                                        -->
<!-- MonitorPolicy_ID - GlobalConfig の DefaultPolicies の 1 ～ 21 のインデックス -->
<!--                                                                        -->
<!-- 1 - 状態のロールアップ                                                 >--
<!-- 2 - 状態の冗長性                                                    >--
<!-- 3 - 状態 - 高感度                                                      >--
<!-- 4 - 状態 - 低感度                                                   >--
<!-- 5 - 条件の割合                                                      >--
<!-- 6 - サービス ヘルスの冗長性                                            >--
<!-- 7 - サービス ヘルス - 高感度                                           >--
<!-- 8 - サービス ヘルス - 低感度                                           >--
<!-- 9 - サービス ヘルスのパーセンテージ                                    >--
<!-- 10 - 接続ステータスの冗長性                                            >--
<!-- 11 - 接続ステータス - 高感度                                           >--
<!-- 12 - 接続ステータス - 低感度                                           >--
```

```
<!-- 13 - 接続ステータスの割合                                                    >--
<!-- 14 - ポート ステータスの冗長性                                                 >--
<!-- 15 - ポート ステータス - 高感度                                                >--
<!-- 16 - ポート ステータス - 低感度                                                >--
<!-- 17 - ポート ステータスの割合                                                   >--
<!-- 18 - レスポンス時間の冗長性                                                    >--
<!-- 19 - レスポンス時間 - 高感度                                                   >--
<!-- 20 - レスポンス時間 - 低感度                                                   >--
<!-- 21 - レスポンス時間の低パーセンテージ                                          >--
<!--                                                                            -->
<!-- ************************************************************************ -->

<!ATTLIST SM_Service
        name                        CDATA               #REQUIRED
        containment_relation  ( SlmMonitors |
                                SlmWatchesContainer |
                                MaintenenceScheduledBy ) #IMPLIED
        Criticality  ( 10 | 15 | 20 | 25 | 30 )         #IMPLIED

        AttrToWatch             CDATA                   #IMPLIED
        MonitorPolicy_ID        CDATA                   #IMPLIED
        is_managed              (true  | false)         #IMPLIED
        Generate_Service_Alarms (true  | false)         #IMPLIED
        Security_String         CDATA                   #IMPLIED
        model_type        ( SM_Service )                #IMPLIED
        >

<!ELEMENT SM_AttrMonitor ( SM_Service         |
                        SM_AttrMonitor     |
                        SM_LatencyMon      |
                        SM_ConnectMon      |
                        Device             |
                        Port               |
                        Topology_Container |
                        RTM_Test           |
                        MonitorPolicy_Attr |
                        )*>


<!-- ************************************************************************ -->
<!-- SM_AttrMonitor モデルを表します                                          -->
<!--                                                                          -->
<!-- AttrToWatch は以下のいずれかになります                                   -->
<!--                                                                          -->
<!-- Condition - ほとんどのモデル、ポリシー 1 ～ 5 に使用できます          -->
<!-- Contact_Status - 通常、デバイス モデル、ポリシー 10 ～ 13 に使用されます   -->
<!-- Port_Status - インターフェース モデル、ポリシー 14 ～ 17 に使用されます    -->
<!-- LatestErrorStatus - RTM_Test モデル、ポリシー 18 ～ 21  -->
```

```
<!-- または Response_Time に使用されます                                                    -->
<!-- RM_Condition - SM_AttrMonitor または SM_Service のモデル、ポリシー 6 ～ 9 -->
<!-- またはサービス ヘルス                                                                  -->
<!--                                                                                      -->
<!-- MonitorPolicy_ID - GlobalConfig の DefaultPolicies の 1 ～ 21 のインデックス -->
<!--                                                                                      -->
<!-- 1 - 状態のロールアップ                                                          >--
<!-- 2 - 状態の冗長性                                                              >--
<!-- 3 - 状態 - 高感度                                                               >--
<!-- 4 - 状態 - 低感度                                                              >--
<!-- 5 - 条件の割合                                                                >--
<!-- 6 - サービス ヘルスの冗長性                                                          >--
<!-- 7 - サービス ヘルス - 高感度                                                          >--
<!-- 8 - サービス ヘルス - 低感度                                                          >--
<!-- 9 - サービス ヘルスのパーセンテージ
<!-- 10 - 接続ステータスの冗長性                                                          >--
<!-- 11 - 接続ステータス - 高感度                                                          >--
<!-- 12 - 接続ステータス - 低感度                                                          >--
<!-- 13 - 接続ステータスの割合                                                          >--
<!-- 14 - ポート ステータスの冗長性                                                          >--
<!-- 15 - ポート ステータス - 高感度                                                          >--
<!-- 16 - ポート ステータス - 低感度                                                          >--
<!-- 17 - ポート ステータスの割合                                                          >--
<!-- 18 - レスポンス時間の冗長性                                                          >--
<!-- 19 - レスポンス時間 - 高感度                                                          >--
<!-- 20 - レスポンス時間 - 低感度                                                          >--
<!-- 21 - レスポンス時間の低パーセンテージ                                                  >--
<!--                                                                                      -->
<!-- Special_Cause_List - サービスのサービス ヘルスまたはリソース          -->
<!--                          モニタ モデルに影響する、含める/除外を          -->
<!--                          指定するために使用できるアラームの原因の       -->
<!--                  リストまたは範囲。 AttrToWatch が Condition である場合のみ      -->
<!--                          使用できます。                          -->
<!--                                                                                      -->
<!-- Cause_List_Control - Special_Cause_List の使用方法を指定します。          -->
<!--                          0 - 未使用                                          -->
<!--                          1 - 包括的                                        -->
<!--                          2 - 排他的                                        -->
<!--                                                                                      -->
<!-- ********************************************************************** -->
```

```
<!ATTLIST SM_AttrMonitor
        name                        CDATA               #REQUIRED
        containment_relation    ( SlmMonitors |
                                  SlmWatchesContainer )  #IMPLIED
        is_managed              (true  | false)          #IMPLIED
        AttrToWatch             CDATA                    #IMPLIED
        MonitorPolicy_ID        CDATA                    #IMPLIED
        Generate_Service_Alarms (true  | false)          #IMPLIED
        model_type      ( SM_AttrMonitor )               #IMPLIED
        >

<!ELEMENT MonitorPolicy_Attr ( #PCDATA ) >

<!ELEMENT SM_SLA ( SM_Service   |
                SM_Guarantee |
                Schedule
                )*>

<!-- ************************************************************************ -->
<!-- SM_Guarantee モデルを表します                                          -->
<!--                                                                          -->
<!-- SLAControl には、以下の値があります                                  -->
<!--                                                                          -->
<!--  0 = 非アクティブ                                                      -->
<!--  1 = アクティブ                                                        -->
<!--                                                                          -->
<!-- ************************************************************************ -->

<!ATTLIST SM_SLA
        name              CDATA                  #REQUIRED
        containment_relation  ( SlmHasGuarantee |
                                SlmGuarantees |
                                SlaPeriod )       #IMPLIED
        is_managed          (true  | false)       #IMPLIED
        SLA_Control         ( 0 | 1 )             #IMPLIED
        SLA_ExpirationDate    CDATA               #IMPLIED
        SLA_Notes             CDATA               #IMPLIED
        SLA_Description       CDATA               #IMPLIED
        Security_String       CDATA               #IMPLIED
        model_type          ( SM_SLA )            #IMPLIED
        >

<!ELEMENT SM_Guarantee ( SM_Service      |
                         SM_AttrMonitor  |
                         SM_LatencyMon   |
                         SM_ConnectMon   |
                         Schedule
                         )*>
```

```
<!-- ********************************************************************** -->
<!-- SM_Guarantee モデルを表します                                          -->
<!--                                                                        -->
<!-- GuaranteeControl には、以下の値があります -->
<!--                                                                        -->
<!--  0 = 非アクティブ                                                      -->
<!--  1 = アクティブ                                                        -->
<!--                                                                        -->
<!-- GuranteeType には、以下の値があります -->
<!--                                                                        -->
<!-- 0 = 可用性 -->
<!-- 1 = パフォーマンス -->
<!-- 2 = 平均修復時間 -->
<!-- 3 = 最大停止時間 -->
<!--                                                                        -->
<!-- ServiceHealthType には、以下の値があります -->
<!--                                                                        -->
<!--  1 - ダウン                                                            -->
<!--  2 - 低下                                                              -->
<!--                                                                        -->
<!-- ********************************************************************** -->

<!ATTLIST SM_Guarantee
        name                        CDATA          #REQUIRED
        containment_relation  ( SlmIsMeasuredBy |
                                SlmSchedulesGuarantee )  #IMPLIED
        is_managed            (true  | false)      #IMPLIED
        GuaranteeControl      ( 0 | 1 )            #IMPLIED
        GuaranteeType         ( 0 | 1 | 2 | 3 )    #REQUIRED
        ServiceHealthType     ( 1 | 2 )            #IMPLIED
        WarningThreshold            CDATA          #IMPLIED
        WarningThresholdPercent     CDATA          #IMPLIED
        ViolationThreshold          CDATA          #IMPLIED
        ViolationThresholdPercent   CDATA          #IMPLIED
        GuaranteeNotes              CDATA          #IMPLIED
        GuaranteeDescription        CDATA          #IMPLIED
        model_type             ( SM_Guarantee )    #IMPLIED
        MOT_Threshold               CDATA          #IMPLIED
        MTTR_Threshold              CDATA          #IMPLIED
        MTBF_Threshold              CDATA          #IMPLIED
        >

<!ELEMENT SM_LatencyMon ( Topology_Container |
                          MonitorPolicy_Attr )*>
```

```
<!ATTLIST SM_LatencyMon
        name                    CDATA                #REQUIRED
        containment_relation  ( SlmMonitors |
                                   SlmWatchesContainer ) #IMPLIED
        is_managed            ( true  | false )      #IMPLIED
        DefaultMaxRTT           CDATA                #IMPLIED
        DefaultMeasureInterval  CDATA                #IMPLIED
        model_type            ( SM_LatencyMon )      #IMPLIED
        >

<!ELEMENT SM_CustomerGroup ( SM_CustomerGroup |
                        SM_Customer
                        )*>

<!ATTLIST SM_CustomerGroup
        name                    CDATA                #REQUIRED
        containment_relation ( Groups_Customers )    #IMPLIED
        model_type           ( SM_CustomerGroup )    #IMPLIED
        >

<!ELEMENT SM_Customer ( SM_Service |
                           SM_SLA     |
                         )*>

<!-- ************************************************************************ -->
<!-- SM_Customer モデルを表します -->
<!--                                                                        -->
<!-- 重大度は以下のいずれかになります                                       -->
<!--                                                                        -->
<!-- 10 - 低                                                             >--
<!-- 15 - やや低                                                            >--
<!-- 20 - 中                                                             >--
<!-- 25 - やや高                                                            >--
<!-- 10 - 高                                                             >--
<!--                                                                        -->
<!-- ************************************************************************ -->

<!ATTLIST SM_Customer
        name                          CDATA  #REQUIRED
        containment_relation  ( SlmAgreesTo |
                                  SlmUses )     #IMPLIED
        Security_String               CDATA  #IMPLIED
        CustomerID                    CDATA  #IMPLIED
        Criticality                   CDATA  #IMPLIED
        CustomerField4                CDATA  #IMPLIED
        CustomerField5                CDATA  #IMPLIED
        CustomerField6                CDATA  #IMPLIED
        CustomerField7                CDATA  #IMPLIED
        Contact_Name                  CDATA  #IMPLIED
```

```
        Contact_Title                 CDATA  #IMPLIED
        Contact_Location              CDATA  #IMPLIED
        Email_Address                 CDATA  #IMPLIED
        Phone_Number                  CDATA  #IMPLIED
        Mobile_Phone_Number           CDATA  #IMPLIED
        Pager_Number                  CDATA  #IMPLIED
        Fax_Number                    CDATA  #IMPLIED
        User_Defined_1                CDATA  #IMPLIED
        User_Defined_2                CDATA  #IMPLIED
        User_Defined_3                CDATA  #IMPLIED
        User_Defined_4                CDATA  #IMPLIED
        Secondary_Contact_Name        CDATA  #IMPLIED
        Secondary_Contact_Title       CDATA  #IMPLIED
        Secondary_Contact_Location    CDATA  #IMPLIED
        Secondary_Email_Address       CDATA  #IMPLIED
        Secondary_Phone_Number        CDATA  #IMPLIED
        Secondary_Mobile_Phone_Number CDATA  #IMPLIED
        Secondary_Pager_Number        CDATA  #IMPLIED
        Secondary_Fax_Number          CDATA  #IMPLIED
        Secondary_User_Defined_1      CDATA  #IMPLIED
        Secondary_User_Defined_2      CDATA  #IMPLIED
        Secondary_User_Defined_3      CDATA  #IMPLIED
        Secondary_User_Defined_4      CDATA  #IMPLIED
        model_type           ( SM_Customer )  #IMPLIED
        >

<!-- ********************************************************************** -->
<!-- GlobalCollection モデルを表します                                      -->
<!--                                                                        -->
<!-- ********************************************************************** -->
<!ELEMENT GlobalCollection (( Device            |
                        Topology_Container |
                        Location_Container )*) >


<!ATTLIST GlobalCollection
        name              CDATA                 #REQUIRED
        containment_relation  CDATA             "GlobalCollect"
        collectionDescription CDATA             #IMPLIED
        Security_String       CDATA             #IMPLIED >
```

付録 C: XML の例

このセクションでは、DTD の操作に役立つ XML の例を示します。

注: 例のエレメント名は太字で強調表示されます。名前の太字は、例を読みやすくすることを目的としていて、XML 入力ファイルに必要な書式を示すものではありません。

例 1: トポロジ ビューへのインポート

この例では、CA Spectrum トポロジ ビューに情報をインポートする基本的な入力ファイルを示します。このファイルは、トポロジ ビューでネットワーク コンテナ モデルを作成します。Network コンテナで、LAN コンテナ モデルが作成されます。LAN コンテナ内では、2 つのデバイスが作成されます。DNS 名デッドロックは 1 つのデバイスを識別し、IP アドレスは別のデバイスを識別します。

Topology エレメントの complete_topology 属性は False に設定されます。この場合、CA Spectrum は、以前にトポロジ ビューに存在した他のモデルを使用します。したがって、このファイルは、XML ファイルでリストされ、すでにモデリングされていないエントリ用のモデルを作成するのみです。作成されるモデルはファイルで指定されたトポロジ階層に配置されます。以前にトポロジ階層に存在したモデルは再発見されず、ファイルで指定されたコンテナに移動されます。

注: complete_topology が false に設定されている場合、コンテナにあるがインポート ファイルにリストされていない既存のモデルは、ロスト ファウンドに送信 *されません*。complete_topology が true に設定されている場合、これらのモデルはロスト ファウンドに送信 *されます*。

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
<!-- ************************************ -->
<!-- この部分はトポロジ ビューのインポート用です -->
<!-- ************************************ -->
```

```
<Topology complete_topology="false">
        <Device ip_dnsname="10.253.9.17" model_type="GnSNMPDev"
                community_string="public"/>
        <Device ip_dnsname="nmcss52-5" />
        <Topology_Container model_type="Network" name="My Network"
                Security_String="public" subnet_address="10.253.0.0"
                subnet_mask="255.255.0.0">
                <Topology_Container model_type="Lan" name="Lan1"
                        Security_String="public" subnet_address="10.253.9.0"
                        subnet_mask="255.255.255.0">
                        <Device ip_dnsname="deadlock" />
                        <Device ip_dnsname="10.253.9.18" poll_interval="333" />
                </Topology_Container>
        </Topology_Container>
</Topology>
</Import>
```

例 2: 接続の作成

この例では、2 つの ATM 回路間、および 2 つのフレーム リレー回路間の接続の作成を示します。 また、FrameRelay DLCI ポートと ATM VCL ポート間の接続も示します。

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
        <Connection create_pipe="false">
                <Device ip_dnsname="10.253.32.225">
                        <Port identifier_name="atmVclTableInstance"
                                identifier_value="5.0.5"
                                circuit_name="ATM Link1"
                                circuit_id = "ATM 5017" />
                </Device>
                <Device ip_dnsname="192.168.52.25">
                        <Port identifier_name="atmVclTableInstance"
                                identifier_value="3.0.12"
                                circuit_name="ATM Link1"
                                circuit_id = "ATM 5017" />
                </Device>
        </Connection>
```

```
        <Connection>
                <Device ip_dnsname="10.253.9.18">
                        <Port identifier_name="frCircuitTableInstance"
                                identifier_value="2.27"/>
                </Device>
                <Device ip_dnsname="nmcss52-5">
                        <Port identifier_name="frCircuitTableInstance
                                identifier_value="4.161"/>
                </Device>
        </Connection>
        <!-- ****************************************** -->
        <!-- FrameRelay DLCI ポートと -->
        <!--ATM VCL ポート間の接続。                              -->
        <!-- ****************************************** -->
        <Connection>
                <Device ip_dnsname="10.253.9.18">
                        <Port identifier_name="frCircuitTableInstance"
                                identifier_value="2.27"/>
                </Device>
                <Device ip_dnsname="10.253.32.225">
                        <Port identifier_name="atmVclTableInstance"
                                identifier_value="5.0.17"/>
                </Device>
        </Connection>
        <Connection>
                <Device ip_dnsname="nmcss52-5">
                        <Port identifier_name="ifIndex" identifier_value="3"/>
                </Device>
                <Device ip_dnsname="10.253.9.17">
                        <Port identifier_name="ifPhysAddress"
                                identifier_value="0:4:27:C:91:C0"/>
                </Device>
        </Connection>
</Import>
```

例 3: 更新と破棄

この例では、Update エレメントおよび Destroy エレメントの使用法を示します。

Update エレメントには Location_Container エレメントが含まれます。 この例は、Location_Container エレメントの name 属性と model_name 属性を使用してモデル名を更新します。 name 属性は現在の名前に等しく設定され、更新されるモデルを識別します。 model_name 属性は、name 属性の値を Peace2 に更新します。

Update エレメントには、Device エレメントと Port エレメントが含まれます。 identifer_name 属性と identifier_value 属性は、更新するポートを識別するために使用されます。 指定される他の属性は、値が更新される属性です。 ポート モデル名はポート 2 に変更され、poll_status は False に変更されます。

Destroy エレメントはデバイス モデルのデッドロックを解除します。 デッドロックと関連付けられる任意の接続またはポートは自動的に破棄されます。 Building コンテナ モデル Durham も破棄されます。 Durham コンテナに含まれているすべてのモデルはロスト ファウンドに送信されます。

また、Destroy エレメントは、デバイス nmcss52-5 上の指定されたポートと nmcss52-3 上の指定されたポートの間の接続を削除します。

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">

<Import>

<!-- ********************************************** -->
<!-- モデルの更新.....................................-->
<!-- ********************************************** -->
        <Update>
<!-- ********************************************** -->
<!-- コンテナ Peace のモデル名を Peace から-->
<!-- Peace2 に変更します ..................................... ....-->
<!-- ********************************************** -->
                <Location_Container model_type="Building" name="Peace"
                        model_name="Peace2"/>
<!-- ****************************************** -->
<!-- デバイス nmcss52-5 上のポート ifIndex=2 を更新します -->
<!-- ****************************************** -->
                <Device ip_dnsname="nmcss52-5">
                        <Port identifier_name="ifIndex" identifier_value="2"
                                model_name="port 2" poll_status="false" />
                </Device>
        </Update>
```

```
<!-- ****************************** -->
<!-- モデルと接続を破棄します。 -->
<!-- ****************************** -->
      <Destroy>
            <Device ip_dnsname="deadlock"/>
            <Location_Container  model_type="Building" name="Durham" />
            <Connection>
                  <Device ip_dnsname="nmcss52-5">
                        <Port identifier_name="ifIndex"
                              identifier_value="1"/>
                  </Device>
                  <Device ip_dnsname="10.253.9.17">
                        <Port identifier_name="ipAddress"
                              identifier_value="10.253.8.18"/>
                  </Device>
            </Connection>
      </Destroy>
</Import>
```

例 4: 作成、更新、および破棄

以下の XML ファイルは、DTD に含まれているエレメントのほとんどの機能を示します。 このファイルはトポロジ ビューと場所ビューの両方でデータを作成し、接続を作成し、属性を更新し、モデルと接続を破棄します。

XML ファイルの最初のセクションでは、トポロジ ビューでモデルを作成し、モデル間の接続を作成します。 セクションは Topology エレメントで始まります（<Topology>）。 最初にコンテナ モデルとデバイス モデルが作成され、次に接続が確立されます。 このセクションは、Topology エレメントが閉じるときに終了します（</Topology>）。

Topology エレメントが閉じられた後、別の接続が作成されるセクションがあります。 このセクションでは、Topology エレメント内で Connection エレメントをネストせずに、接続を作成できることを示します。

ファイルの次のセクションは Location エレメントで始まります（<Location>）。 このセクションでは、場所ビューでのコンテナ モデルとデバイス モデルの作成を示します。 このセクションは、Location エレメントが閉じるときに終了します（</Location>）。

次のセクションは、Update エレメントで始まります（<Update>）。 このセクションでは、コンテナ、デバイス、およびポートの属性値が変更されます。 XML ファイルを見ても、表されている各エレメントの現在の属性値はわかりません。 そのため、更新されているエレメントを識別するのは簡単ではありません。 一般的に、各エレメントには、更新されるモデルまたはポートを一意に識別する属性が含まれます。 他の属性は、値を更新するために指定されます。 たとえば、最初のエレメントは Location_Container エレメントです。name 属性は一意にモデルを識別します。 model_type 属性は、その値を（おそらくは Region から Building に）更新するために指定されます。 Update エレメントで定義できる唯一の階層は、更新するポートを指定する Device/Port 階層です。 このセクションは、Update エレメントが閉じるときに終了します（</Update>）。

このファイルの最後のセクションでは、Destroy エレメントを使用します。Destroy エレメントは以下のものを除外します。

- 10.253.9.19 のデバイス
- コンテナ モデル Durham
- デバイス nmcss52-5 上の指定されたポートと 10.253.9.17 のデバイス間の接続

```
<?xml version="1.0" standalone="no"?>
<!DOCTYPE Import SYSTEM ".modelinggateway.dtd">
<Import>
<!-- ************************************ -->
<!-- この部分はトポロジ ビューのインポート用です -->
<!-- ************************************ -->
        <Topology discover_connections="false" complete_topology="false">
                <Device ip_dnsname="10.253.9.109" model_type="GnSNMPDev"
                        community_string="public" is_managed="false"/>
                <Device ip_dnsname="10.253.9.17"
                        poll_interval="333" log_ratio="11"/>
                <Device ip_dnsname="10.253.9.19" community_string="public"/>
                <Device ip_dnsname="nmcss52-5" />
```

```
<Topology_Container model_type="Network" name="My Network"
        Security_String="public" subnet_address="10.253.0.0"
        subnet_mask="255.255.0.0" complete_topology="true">
        <Topology_Container model_type="Lan"

                name="MyLan" Security_String="public"

                subnet_address="10.253.9.0"
                subnet_mask="255.255.255.0">
                <Device ip_dnsname="10.253.9.18"
                        community_string="public"
                        poll_interval="333"
                        log_ratio="5"/>
        </Topology_Container>
</Topology_Container>
<Topology_Container model_type="IPClassC" name="my_net"
        subnet_address="172.19.57.0">
        <Device  model_type="Pingable"
                ip_dnsname="172.19.57.91"/>
        <Device  model_type="Fanout" ip_dnsname="1.2.3.4"/>
        <Device  ip_dnsname="10.253.9.16"
                community_string="public"/>
</Topology_Container>
<Topology_Container model_type="Lan" name="lan2"
        Security_String="public" subnet_address="10.253.7.0"
        subnet_mask="255.255.255.0" complete_topology="true">
        <Device ip_dnsname="10.253.7.17"
                community_string="public"
                poll_interval="333" log_ratio="11"/>
        <Device ip_dnsname="10.253.32.101"/>
        <Device ip_dnsname="192.168.125.161"
                model_type="GnSNMPDev"/>
</Topology_Container>
<Device ip_dnsname="172.19.57.92" />
<Device ip_dnsname="172.19.57.93" />
<Device ip_dnsname="10.253.32.225" model_type="M46_04"/>
<Connection>
        <Device ip_dnsname="172.19.57.93">
                <Port identifier_name= "frCircuitTableInstance"
                        identifier_value="4.161"/>
        </Device>
        <Device ip_dnsname="192.168.125.161">
                <Port identifier_name ="frCircuitTableInstance"
                        identifier_value="2.161"/>
        </Device>
</Connection>
```

```
                        <Connection create_pipe="false">
                                <Device ip_dnsname="10.253.32.101">
                                        <Port identifier_name ="atmVclTableInstance"
                                                identifier_value="3.1.52"/>
                                </Device>
                                <Device ip_dnsname="10.253.32.225">
                                        <Port identifier_name ="atmVclTableInstance"
                                                identifier_value="5.0.68"
                                                circuit_name="ATM 68"
                                                circuit_id ="ATM ID 68"/>
                                </Device>
                        </Connection>
                        <Connection>
                                <Device ip_dnsname="nmcss52-5">
                                        <Port identifier_name="ifIndex"
                                                identifier_value="1"/>
                                </Device>
                                <Device ip_dnsname="10.253.9.17">
                                        <Port identifier_name="ipAddress"
                                                identifier_value="10.253.8.18"/>
                                </Device>
                        </Connection>
                </Topology>
                <Connection>
                        <Device ip_dnsname="172.19.57.92">
                                <Port identifier_name="ifPhysAddress"
                                        identifier_value="0:E0:63:7C:19:61"/>
                        </Device>
                        <Device ip_dnsname="10.253.9.17">
                                <Port identifier_name="ipAddress"
                                        identifier_value="10.253.8.65"/>
                        </Device>
                </Connection>
        <!-- ********************************** -->
        <!-- この部分は場所ビューのインポート用です -->
        <!-- ********************************** -->
                <Location complete_topology="true">
                        <Location_Container model_type="Country" name="USA"
                                Security_String="whatever">
                                <Location_Container model_type="Region"
                                        name="New Hampshire"
                                        complete_topology="false">
                                        <Location_Container model_type="Site"
                                        name="Durham"
                                                <Device ip_dnsname = "10.253.32.10"/>
                                                <Device ip_dnsname = "172.19.57.93" />
                                        </Location_Container>
                                </Location_Container>
                        </Location_Container>
```

```
                <Location_Container model_type="Building" name="Durham"
                        Security_String="public">
                        <Location_Container model_type="Room" name="my_room"
                                Security_String="hahaha">
                                <Device ip_dnsname="10.253.9.16"
                                        community_string="public"/>
                                <Device ip_dnsname="10.253.9.17" />
                                <Device ip_dnsname = "10.253.9.18"/>
                        </Location_Container>
                </Location_Container>
                <Location_Container model_type="Building" name="Peace"
                        Security_String="aprisma">
                        <Location_Container model_type="Room" name= "Lab 1">
                                <Device ip_dnsname="10.253.7.17"
                                        community_string="public"/>
                                <Device ip_dnsname="192.168.125.161"/>
                        </Location_Container>
                </Location_Container>
        </Location>
<!-- ************************************** -->
<!-- この部分はモデルの更新用です -->
<!-- ************************************** -->
        <Update>
                <Location_Container model_type="Building" name="Peace"
                        model_modify_author="ltang"/>
                <Device ip_dnsname="172.19.57.93" poll_interval="101"
                        model_name="haha" />
    <!-- ************************************** -->
    <!-- この部分ではデバイス nmcss52-5 上の-->
    <!-- ポート ifIndex=2 を更新します                -->
    <!-- ************************************** -->
                <Device ip_dnsname="nmcss52-5">
                        <Port identifier_name="ifIndex" identifier_value="2"
                                model_name="port 2" poll_interval="1103"
                                poll_status="false" log_ratio="12"/>
                </Device>
                <Topology_Container model_type="Lan" name="lan2"
                        Security_String="top secret"/>
        </Update>
```

```
<!-- ********************************************** -->
<!-- この部分ではモデルと接続を削除します -->
<!-- ********************************************** -->
        <Destroy>
                <Device ip_dnsname="10.253.9.19"/>
                <Location_Container model_type="Building" name="Durham"/>
                <Connection>
                        <Device ip_dnsname="nmcss52-5">
                                <Port identifier_name="ifIndex"
                                        identifier_value="1"/>
                        </Device>
                        <Device ip_dnsname="10.253.9.17">
                                <Port identifier_name="ipAddress"
                                        identifier_value="10.253.8.18"/>
                        </Device>
                </Connection>
        </Destroy>
</Import>
```

付録 D: .modelinggatewayresource.xml

このセクションには、.modelinggatewayresource.xml ファイルのコピーが含まれています。 ただし、このコピーはこのファイルの最新バージョンではない可能性があります。 最新のバージョンについては、Modeling Gateway Toolkit に用意されている実際のファイルを使用してください。

```
<?xml version="1.0" standalone="no"?>

<TopologyImportExportResourceFile>

<!-- ************************************** -->
<!-- トポロジのエクスポートとインポートに使用される     -->
<!-- SPECTRUM 属性名と ID。  -->
<!-- ************************************** -->

<Attributes
    circuit_id                  = "0xc4042f"
    circuit_name                = "0xc40430"
    community_string            = "0x10024"
    agent_port                  = "0x10023"
    DeviceType                  = "0x23000e"
    is_managed                  = "0x1295d"
    log_ratio                   = "0x10072"
    manager_name                = "0x3dc0009"
    model_modify_author         = "0x11025"
    model_name                  = "0x1006e"
    name                        = "0x1006e"
    poll_interval               = "0x10071"
    poll_status                 = "0x1154f"
    TryCount                    = "0x110c5"
    Security_String             = "0x10009"
    subnet_address              = "0x1027f"
    subnet_mask                 = "0x110b8"
    subnet_list                 = "0x11953"
    TimeOut                     = "0x110c4"
    trapIPAddress               = "0x3dc0007"
    unique_id                   = "0x3dc0004"
    Value_When_Orange           = "0x1000d"
    Value_When_Red              = "0x1000e"
    Value_When_Yellow           = "0x1000c"

    LatestErrorStatus           = "456008c"
    Response_Time               = "456008c"
```

```
AttrToWatch                 = "0x12a43"
MonitorPolicy               = "0x12a3e"
MonitorPolicy_ID            = "0x12a51"
Generate_Service_Alarms     = "0x12a66"
Special_Cause_List          = "0x12b47"
Cause_List_Control          = "0x12d50"

Contact_Status              = "0x10004"
Port_Status                 = "0x10f1b"
RM_Condition                = "0x12a40"
Service_Health              = "0x12a40"
Criticality                 = "0x1290c"
Condition                   = "0x1000a"
Condition_Value             = "0x1000b"

AccumulationMethod          = "0x4500007"
GuaranteeControl            = "0x4500022"
GuaranteeNotes              = "0x4500021"
GuaranteeDescription        = "0x12a4b"
GuaranteeType               = "0x4500018"
ServiceHealthType           = "0x4500019"
ViolationThreshold          = "0x450001e"
ViolationThresholdPercent   = "0x4500024"
WarningThreshold            = "0x450001d"
WarningThresholdPercent     = "0x4500023"

SCHED_Daily_Repeat_Limit    = "0x1299a"
SCHED_Duration              = "0x12993"
SCHED_Recurrence_Multiplier = "0x1299b"
SCHED_Recurrence            = "0x12994"
SCHED_Start_DoM             = "0x12991"
SCHED_Start_DoW             = "0x12990"
SCHED_Start_Hour            = "0x1298f"
SCHED_Start_Minute          = "0x1298e"
SCHED_Start_Month           = "0x12992"
SCHED_Start_Day             = "0x129e4"
SCHED_DayBitMask            = "0x129da"
SCHED_Start_Year            = "0x129e3"
SCHED_Start_MoY             = "0x12b48"
SCHED_Description           = "0x12bbc"

SLA_Control                 = "0x4500015"
SLA_Notes                   = "0x4500017"
SLA_ExpirationDate          = "0x4500025"
SLA_Description             = "0x12a4b"

DefaultMaxRTT               = "0x4500001"
DefaultMeasureInterval      = "0x4500002"
```

```
    CustomerID                    = "0x12a44"
    CustomerField4                = "0x12a39"
    CustomerField5                = "0x12a3a"
    CustomerField6                = "0x12a3b"
    CustomerField7                = "0x12a3c"

    Contact_Name                  = "0x12a20"
    Contact_Title                 = "0x12a21"
    Contact_Location              = "0x12a22"
    Email_Address                 = "0x12a27"
    Phone_Number                  = "0x12a23"
    Mobile_Phone_Number           = "0x12a24"
    Pager_Number                  = "0x12a25"
    Fax_Number                    = "0x12a26"
    User_Defined_1                = "0x12a28"
    User_Defined_2                = "0x12a29"
    User_Defined_3                = "0x12a2a"
    User_Defined_4                = "0x12a2b"

    Secondary_Contact_Name        = "0x12a2c"
    Secondary_Contact_Title       = "0x12a2d"
    Secondary_Contact_Location    = "0x12a2e"
    Secondary_Email_Address       = "0x12a33"
    Secondary_Phone_Number        = "0x12a2f"
    Secondary_Mobile_Phone_Number = "0x12a30"
    Secondary_Pager_Number        = "0x12a31"
    Secondary_Fax_Number          = "0x12a32"
    Secondary_User_Defined_1      = "0x12a34"
    Secondary_User_Defined_2      = "0x12a35"
    Secondary_User_Defined_3      = "0x12a36"
    Secondary_User_Defined_4      = "0x12a37"

    MOT_Threhold                 = "0x450002c"
    MTBF_Threshold                = "0x4500032"
    MTTR_Thresshold                = "0x450002f"

    Policy_Name_List              = "0x12a4a"
    collectionDescription         = "0x12a67"

/>

<!-- ****************************************** -->
<!-- トポロジのインポート XML ファイルで使用できる  -->
<!-- SPECTRUM モデル タイプ名とハンドル。      -->
<!-- ****************************************** -->

<ModelTypes
    Universe                     = "0x10091"
    Network                      = "0x1002e"
```

```
Lan                     = "0x1002d"
IPClassA                = "0x103d5"
IPClassB                = "0x103d6"
IPClassC                = "0x103d7"
LAN_802_3               = "0x1003c"
LAN_802_5               = "0x1003d"
ATM_NETWORK             = "0xaa000f"
EventAdmin              = "0x3dc0000"
GlobalCollection        = "0x10474"

World                   = "0x10040"
Country                 = "0x10041"
Region                  = "0x10042"
Site                    = "0x10043"
Sector                  = "0x10044"
Building                = "0x10045"
Section                 = "0x10046"
Floor                   = "0x10047"
Room                    = "0x10048"

Top_Org                 = "0x102cf"
Enterprise              = "0x102d0"
Subsidiary              = "0x102d1"
Division                = "0x102d2"
Department              = "0x102d3"
Org_Section             = "0x102d4"
Work_Group              = "0x102d5"
Org_Owns                = "0x102da"

Schedule                = "0x10456"
GnSNMPDev               = "0x3d0002"
Fanout                  = "0x100ae"
Pingable                = "0x10290"
WA_Link                 = "0x102e2"
Unplaced                = "0x103d8"

RTM_Test                = "0x4560000"
SM_Service              = "0x1046f"
SM_AttrMonitor          = "0x1046e"
SM_LatencyMon           = "0x4500001"
SM_SLA                  = "0x4500002"
SM_Guarantee            = "0x4500003"
SM_Customer             = "0x1046c"
SM_CustomerGroup        = "0x10477"
SM_ConnectMon           = "0x4500000"
SM_ServiceMgr           = "0x4500006"
CustomerManager         = "0x10478"
SM_Service_Mgt          = "0x4500007"
```

```
    SM_SLA_Mgr                  = "0x4500008"
    Correlation_Domain          = "0x10467"
    Correlation_Manager         = "0x10469"

/>

<Relations
    Collects                    = "0x10002"
    MaintenenceScheduledBy      = "0x10034"
    SlmAgreesTo                 = "0x4500000"
    SlmGuarantees               = "0x4500001"
    SlmHasGuarantee             = "0x4500002"
    SlmIsMeasuredBy             = "0x4500003"
    SlmMonitors                 = "0x4500004"
    SlmOwns                     = "0x4500005"
    SlmUses                     = "0x4500006"
    SlmWatchesContainer         = "0x4500007"
    SlmContains                 = "0x4500008"
    SlaPeriod                   = "0x4500009"
    SlmSchedulesGuarantee       = "0x450000c"
    SlmHasServiceComponent      = "0x450000a"
    SlmContainsSLAs             = "0x450000b"
    Groups_Customers            = "0x1003e"
    GlobalCollect               = "0x1003b"

/>

<!-- do_not_process_pre_existing_devices_under_container_node を -->
<!-- true に設定した場合、Container エレメント下でデバイスが Spectrum に -->
<!-- すでに存在していることがわかると、Modeling Gateway はそのデバイスに -->
<!-- 対して、属性の更新、接続の作成などの処理を実行しません。 -->

<ImportConfiguration
    do_not_process_pre_existing_devices_under_container_node = "false"
    import_to_primary_ss_only = "false"
    max_device_creation_threads="50"
/>

<!-- ******************************************************* -->
<!--                                                         -->
<!-- これは Modeling Gateway のエクスポート設定です。        -->
<!--                                                         -->
<!-- ExportConfiguration は、エクスポートするものを制御する -->
<!-- 設定です。                                              -->
<!--                                                         -->
<!-- export_devices : デバイス モデルをエクスポートするかどうか -->
<!--                                                         -->
<!-- export_containers : コンテナ モデルをエクスポートするかどうか -->
```

```
<!--                                                                -->
<!-- export_port_attributes :  ポート属性をエクスポートするかどうか -->
<!--                                                                -->
<!-- export_links :  デバイス  リンクをエクスポートするかどうか          -->
<!--                                                                -->
<!-- export_topology_layout :  デバイスとコンテナの          >--
<!--                           x,y 座標をエクスポートするかどうか            -->
<!--                                                                -->
<!-- export_annotation :  注釈をエクスポートするかどうか         -->
<!--                                                                -->
<!-- export_WA_Link_models :  WA_Link モデルをエクスポートするかどうか。       -->
<!--                          エクスポートしない場合、WA_Link モデルは -->
<!--                          透過的に処理されます。   WA_Link  -->
<!--                          を通じて行われた 2 つデバイス間の     -->
<!--                          リンクは、直接リンクとして       -->
<!--                          エクスポートされます。                          -->
<!--                                                                -->
<!-- export_spectrum_settings :  SPECTRUM 設定をエクスポートするかどうか -->
<!--                             (障害分離、自動検出、      -->
<!--                             VNM コントロール       -->
<!--                             などの設定)               -->
<!--                                                                -->
<!-- export_user_models: SPECTRUM ユーザ モデル、         -->
<!--                     ユーザ ライセンス、権限などをエクスポートします。  -->
<!--                                                                -->
<!-- export_service_modeling :  SPECTRUM Service モデリングをエクスポートします>--
<!--                                                                -->
<!-- export_schedules :  SPECTRUM スケジュールをエクスポートします。               -->
<!--                                                                -->
<!-- export_discovery_configs :  自動ディスカバリの設定を    >--
<!--                             エクスポートします。                  -->
<!--                                                                -->
<!-- ************************************************************ -->

<ExportConfiguration
    export_devices           = "true"
    export_containers        = "true"
    export_port_attributes   = "true"
    export_links             = "true"
    export_topology_layout   = "true"
    export_annotation        = "true"
    export_WA_Link_models    = "true"
    export_spectrum_settings = "true"
    export_user_models       = "true"
```

```
    export_service_modeling  = "true"
    export_schedules         = "true"
    export_global_collections="true"
    export_discovery_configs = "true"
    export_from_primary_ss_only = "false"
    export_policy_manager = "true"
/>

<!-- ******************************************************** -->
<!--                                                          -->
<!-- RootContainerToExport は、SPECTRUM でエクスポートする -->
<!-- ルート コンテナを指定します。                              -->
<!--                                                          -->
<!-- ******************************************************** -->

<RootContainerToExport model_type="Universe" model_name=""/>

<!-- ******************************************************** -->
<!--                                                          -->
<!-- DeviceExportAttributes は、エクスポートするデバイス属性の -->
<!-- リストです。  上で属性 ID が指定されていない場合、-->
<!-- above, "attribute_id" を割り当てる必要があります。        -->
<!--                                                          -->
<!-- ******************************************************** -->

<DeviceExportAttributes>

    <name/>
    <model_type attribute_id="0x10000"/>
    <community_string/>
    <agent_port/>
    <poll_interval/>
    <is_managed/>
    <poll_status/>
    <Security_String/>
    <TimeOut/>
    <TryCount/>
    <Criticality/>
    <Value_When_Orange/>
    <Value_When_Red/>
    <Value_When_Yellow/>
    <Redundancy_Admin_Status attribute_id="0x11d2c" />
    <Auto_Reconfigure_Interfaces attribute_id="0x11dd4" />
    <Discover_Connection_After_Linkup_Trap attribute_id="0x11d25" />
    <Device_Discovery_After_Reconfiguration attribute_id="0x11d27" />
    <Generate_Redundancy_Alarms attribute_id="0x11dd6" />
    <Create_Sub_Interfaces attribute_id="0x11f3c" />
    <Topology_Relocate_Model attribute_id="0x11a80" />
    <Disable_Trap_Events attribute_id="0x11cd0" />
```

```
    <Enable_Spectrum_Management attribute_id="0x1295d" />
    <Hibernate_Device attribute_id="0x12aca" />
    <Enable_Event_Creation attribute_id="0x129f8" />
    <Redundancy_Admin_Status attribute_id="0x11d2c" />
    <DeviceCPUUtilization_Threshold attribute_id="0x12ab9" />
    <DeviceCPUUtilization_Reset attribute_id="0x12abb" />
    <DeviceCPUUtilization_Duration attribute_id="0x12bce" />
    <DeviceMemoryUtilization_Threshold attribute_id="0x12aba" />
    <DeviceMemoryUtilization_Reset attribute_id="0x12abc" />
    <DeviceMemoryUtilization_Duration attribute_id="0x12bcf" />

</DeviceExportAttributes>

<!-- ********************************************************* -->
<!--                                                           -->
<!-- ContainerExportAttributes は、エクスポートする    -->
<!-- コンテナ属性です。   属性 ID が上で指定されていない場合、     -->
<!-- specified above, "attribute_id" を割り当てる必要があります。       -->
<!--                                                           -->
<!-- ********************************************************* -->

<ContainerExportAttributes>
    <name/>
    <Security_String/>
    <subnet_address/>
    <subnet_mask/>
    <subnet_list/>
    <Value_When_Orange/>
    <Value_When_Red/>
    <Value_When_Yellow/>
    <SelectMP_port attribute_id="0x118e4" />
 </ContainerExportAttributes>

<!-- ********************************************************* -->
<!--                                                           -->
<!-- PortExportAttributes は、エクスポートする       -->
<!-- ポート属性です。   属性 ID が上で指定されていない場合、      -->
<!-- above, "attribute_id" を割り当てる必要があります。                 -->
<!--                                                           -->
<!-- export_changed_attribute_only = "true" の場合、     -->
<!-- 値がデフォルト値と同じでないポート属性のみが      -->
<!-- エクスポートされます。   そうでない場合、指定されたすべての>--
<!-- ポート属性がエクスポートされます。                                  -->
<!--                                                           -->
<!-- ********************************************************* -->
```

```
<PortExportAttributes export_changed_attribute_only="true" >
    <poll_interval/>
    <poll_status/>
    <ok_to_poll attribute_id="0x11dd8" />
    <PollPortStatus attribute_id="0x1280a" />
    <LockConnection attribute_id="0x129f1" />
    <TimeOut/>
    <TryCount/>
    <is_managed/>
    <Enable_Event_Creation/>
    <Criticality/>
    <Alarm_On_Link_Down_Trap attribute_id="0x11fc2" />
    <Assert_Link_Down_Alarm attribute_id="0x12957" />
    <Utilization_Threshold attribute_id="0x1294b" />
    <Utilization_Reset attribute_id="0x1294f" />
    <Utilization_Threshold_Violation_Duration attribute_id="0x12be4" />
    <Inbound_Utilization_Threshold attribute_id="0x12d9f" />
    <Inbound_Utilization_Reset attribute_id="0x12da0" />
    <Inbound_Utilization_Threshold_Violation_Duration attribute_id="0x12da2" />
    <Outbound_Utilization_Threshold attribute_id="0x12da3" />
    <Outbound_Utilization_Reset attribute_id="0x12da4" />
    <Outbound_Utilization_Threshold_Violation_Duration attribute_id="0x12da6" />
    <Total_Packet_Rate_Threshold attribute_id="0x12da7" />
    <Total_Packet_Rate_Reset attribute_id="0x12da8" />
    <Total_Packet_Rate_Threshold_Violation_Duration attribute_id="0x12be3" />
    <Error_Rate_Threshold attribute_id="0x1294d" />
    <Error_Rate_Threshold_Reset attribute_id="0x12951" />
    <Error_Rate_Threshold_Violation_Duration attribute_id="0x12be5" />
    <Discarded_Threshold attribute_id="0x1294e" />
    <Discarded_Threshold_Reset attribute_id="0x12952" />
    <Discarded_Threshold_Violation_Duration attribute_id="0x12be2" />
</PortExportAttributes>

<SpectrumConfigurationExport model_type="VNM">
    <Minimum_Disk_Space attribute_id="0x119d2" />
    <Security_String/>
    <Unmanaged_Trap_Handling attribute_id="0x11cce" />
    <Trap_Storm_Rate attribute_id="0x122db" />
    <Trap_Storm_Length attribute_id="0x122da" />
    <Auto_Connects attribute_id="0x11f99"/>
    <Device_Thresholds attribute_id="0x12acd" />
    <Use_Full_Qualified_Host_Name attribute_id="0x12984" />
    <Allow_Non_Admin_SNMP_Community_Edit attribute_id="0x12042" />
    <Edit_Notes_By_Read_Only_User attribute_id="0x12043" />
    <Set_isManaged_By_Read_Only_User attribute_id="0x129f3" />
    <Consolidate_Users_In_Group attribute_id="0x12a1d" />
```

```
    <Copy_Users_When_Copying_Group attribute_id="0x12a5e" />
    <VLAN_Configuration attribute_id="0x129ad" />
    <Log_When_Device_Cannot_Be_Contacted attribute_id="0x12943" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="TopologyWrkSpc">
    <Create_WA_Link_Model attribute_id="0x25e0033" />
    <Create_LAN_IP_Subnet_Model attribute_id="0x25e000d" />
    <Create_Physical_Addresses attribute_id="0x25e000c" />
    <Create_Fanout_Models attribute_id="0x25e002e" />
    <Run_ATM_Discovery attribute_id="0x25e002d" />
    <IP_Route_Tables attribute_id="0x25e0006" />
    <Source_Addr_Tables attribute_id="0x25e0025" />
    <Spanning_Tree_Tables attribute_id="0x25e0026" />
    <Proprietary_Disc_Tables attribute_id="0x25e002b" />
    <ARP_Tables attribute_id="0x25e003a" />
    <Traffic_Resolution attribute_id="0x25e002f" />
    <Unmanaged_SNMP_Disc attribute_id="0x25e0034" />
    <New_Device_In_Maint_Mode attribute_id="0x25e0035" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="LostFound" >
    <Automatic_Model_Destruction attribute_id="0x11de1" />
    <Model_Destruction_Interval_Hours attribute_id= "0x11de3"/>
    <Model_Destruction_Interval_Minutes attribute_id="0x11de4" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="FaultIsolation">
    <ICMP_Support_Enabled attribute_id="0x11d98" />
    <ICMP_Timeout attribute_id="0x11dab" />
    <ICMP_TryCount attribute_id="0x11dac" />
    <Lost_Device_TryCount attribute_id="0x12a0a" />
    <Contact_Lost_Model_Destruction attribute_id="0x11fa8" />
    <Destruction_Delay attribute_id="0x11fa9" />
    <Destruction_Event_Generation attribute_id="0x11faa" />
    <Router_Redundancy_Retry_Count attribute_id="0x12a09" />
    <Port_Fault_Correlation attribute_id="0x129e6" />
    <Unresolved_Fault_Alarm_Disposition attribute_id="0x129f4" />
    <WA_Link_Fault_Isolation_Mode attribute_id="0x12adc" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="LivePipes" >
    <Live_Pipe_Enabled attribute_id="0x11df9" />
    <Alarm_Linked_Port attribute_id="0x11fbd" />
    <Suppress_Linked_Port_Alarms attribute_id="0x11fbe" />
    <Port_Always_Down_Alarm_Suppression attribute_id="0x129fb" />
</SpectrumConfigurationExport>
```

```
<SpectrumConfigurationExport model_type="AlarmMgmt" >
    <Generate_Alarm_Event attribute_id="0x11f5f" />
    <Add_Event_To_Alarms attribute_id="0x11f5c" />
    <Use_Old_Alarm_Event attribute_id="0x11f5d" />
    <Alarm_Update_by_Read_Only attribute_id="0x11f5e" />
    <Alarm_Ageout_Time attribute_id="0x129ea" />
    <Disable_Initial_Alarms attribute_id="0x11f5a" />
    <Disable_Suppressed_Alarms attribute_id="0x11f5b" />
    <Disable_Maint_Alarms attribute_id="0x11f59" />
    <Alarm_Clear_By_Read_Only attribute_id="0x11fb2" />
    <Ageout_Residual_Alarm_Only attribute_id="0x129ec" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="PolicyManager" >
    <Policy_Distribution_Mode attribute_id="0x4ad0007" />
</SpectrumConfigurationExport>

<SpectrumConfigurationExport model_type="GlobalConfig" >
    <SNMPv3Profiles attribute_id="0x12bd4" />
    <HibernationCommSuccessTries attribute_id="0x12acb" />
</SpectrumConfigurationExport>

</TopologyImportExportResourceFile>
```