Cisco UCS ラックマウント サーバ CIMC XML API プログラマ ガイド

初版：2011 年 09 月 06 日

シスコシステムズ合同会社
〒107-6227 東京都港区赤坂9-7-1 ミッドタウン・タワー
http://www.cisco.com/jp
お問い合わせ先：シスコ コンタクトセンター
0120-092-255 （フリーコール、携帯・PHS含む）
電話受付時間：平日 10:00～12:00、13:00～17:00
http://www.cisco.com/jp/go/contactcenter/

Text Part Number: OL-25396-01-J

【注意】シスコ製品をご使用になる前に、安全上の注意（**www.cisco.com/jp/go/safety_warning/**）をご確認ください。本書は、米国シスコ発行ドキュメントの参考和訳です。リンク情報につきましては、日本語版掲載時点で、英語版にアップデートがあり、リンク先のページが移動/変更されている場合がありますことをご了承ください。あくまでも参考和訳となりますので、正式な内容については米国サイトのドキュメントを参照ください。また、契約等の記述については、弊社販売パートナー、または、弊社担当者にご確認ください。

目 次

はじめに

この前書きは、次の項で構成されています。

対象読者

このマニュアルは、プログラミングと API の使用について背景知識を持つソフトウェア エンジニアを対象としています。エンジニアは、XML、データ システム、ネットワーキング プロトコル、およびストレージ プロトコルに関する知識を持っている必要があります。

マニュアルの構成

この XML API リファレンス ガイドは、次の章で構成されています。

関連資料

すべての C-Series マニュアルの完全なリストについては、次の URL で入手可能な『*Cisco UCS C-Series Servers Documentation Roadmap*』を参照してください。http://www.cisco.com/go/unifiedcomputing/c-series-doc

（注） 『*Cisco UCS C-Series Servers Integrated Management Controller GUI Configuration Guide*』および『*Cisco UCS C-Series Servers Integrated Management Controller CLI Command Reference*』では、CIMC の概要について説明します。 これは XML API ソフトウェア開発者向けの重要な背景説明です。

マニュアルに関するフィードバック

このマニュアルに関する技術的なフィードバック、または誤りや記載もれなどお気づきの点がございましたら、HTML ドキュメント内のフィードバック フォームよりご連絡ください。ご協力をよろしくお願いいたします。

第 1 章

Cisco CIMC XML API

この章の内容は、次のとおりです。

Cisco CIMC XML API の概要

Cisco Integrated Management Controller（CIMC）XML API は、C シリーズ ラックマウント サーバの CIMC へのプログラマチック インターフェイスです。 この API は、HTTP または HTTPS 経由で XML ドキュメントを受け取ります。 開発者は、任意のプログラミング言語を使用して API メソッドを含む XML ドキュメントを生成できます。 CIMC の設定およびステータス情報は、XML API を介して完全にアクセスできる、管理情報ツリーと呼ばれる階層ツリー構造に格納されます。

Cisco CIMC XML API は、Cisco UCS Manager XML API で利用可能なメソッドのサブセットおよび管理情報モデルを実装します。 両方の API の動作は構文およびセマンティクスの面で似ており、両方で同じクライアント開発ツールおよび技術を使用できます。 Cisco CIMC XML API の範囲は、Cisco UCS Manager XML API とは対照的に単一の C シリーズ ラックマウント サーバ に制限されており、スイッチ、FEX モジュール、サーバ、およびその他のデバイスで構成される UCS 環境全体を制御します。

Cisco CIMC XML API を使用して、ユーザはサーバを設定、管理、およびモニタするために CIMC にプログラムでアクセスします。 API は、CIMC CLI および GUI インターフェイス経由でアクセスできるものと同じ機能を提供します。

API の動作はトランザクション型で、CIMC で保持される単一のデータ モデルで終了します。

API モデルには、次のプログラマチック エンティティが含まれます。

- クラス：管理情報ツリーのオブジェクトのプロパティおよび状態を定義します。
- メソッド：1 つまたは複数のオブジェクトに対して API が実行するアクションです。
- タイプ：オブジェクト ステート（たとえば、`equipmentPresence`）に値をマッピングするオブジェクトのプロパティです。

一般的な要求は CIMC に着信し、FIFO の順にトランザクタ キューに配置されます。 トランザクタはこのキューから要求を取得し、要求を解釈して認可チェックを実行します。 要求が確認されると、トランザクタは管理情報ツリーを更新します。 このすべての動作は、1つのトランザクションで行われます。

イベント サブスクリプションがサポートされます。 最大で 4 つの Cisco CIMC XML API クライアントが、CIMC からのイベント通知を受信するようにサブスクライブできます。 イベント サブスクリプション操作によって接続セッションが確立され、CIMC によって非同期的に送信される XML 形式のイベント通知メッセージをクライアントが受信できるようになります。

（注） リリース 1.4(1.x) では、 Cisco CIMC XML API はエラーに関連するイベントのみについてイベント通知を送信します。

Cisco UCS 管理情報モデル

Cisco UCS を構成するすべての物理および論理コンポーネントは、管理情報ツリーと呼ばれる階層型管理情報モデルで表されます。 このツリー内の各ノードは、管理ステータスと動作ステータスを含む、管理対象オブジェクト（MO）またはオブジェクトのグループを表します。

階層構造は最上部（`sys`）から始まり、親ノードと子ノードを含みます。 このツリー内の各ノードは管理対象オブジェクトであり、Cisco UCS 内の各オブジェクトは、オブジェクトとツリー内の位置を示す一意の識別名（DN）を持ちます。 管理対象オブジェクトはCPU、DIMM、アダプタ カード、ファン、および電源装置などの Cisco UCS リソースを抽象化したものです。

設定ポリシーは、システム内のポリシーの大半を占め、さまざまな Cisco UCS コンポーネントの設定を説明します。 ポリシーは、ある環境下でシステムがどのように動作するかを決定します。 特定の管理対象オブジェクトはユーザが作成せず、自動的に Cisco UCS によって作成されます（電源オブジェクトやファン オブジェクトなど）。 API を起動することによって、管理情報モデル（MIM）にオブジェクトの読み取りと書き込みを行います。

CIMC 管理情報モデル

CIMC 管理情報モデルは、Cisco UCS 管理情報モデルのサブセットです。 C シリーズ ラックマウント サーバ は、次の例のように管理情報ツリーの `sys/rack-unit-1` からモデル化されています。

図 *1*：*CIMC MIM* 構造の図

```
Tree (topRoot):————————————-Distinguished Name:
```

```
|—sys———————————— (sys)
     |—rack-unit-1————————(sys/rack-unit-1)
          |—adaptor-1————————(sys/rack-unit-1/adaptor-1)
          |—psu-1————————(sys/rack-unit-1/psu-1)
          |—psu-2————————(sys/rack-unit-1/psu-2)
```

Cisco CIMC XML API サンプル フロー

一般的な要求は CIMC に着信し、FIFO の順にトランザクタ キューに配置されます。 トランザクタはこのキューから要求を取得し、要求を解釈して認可チェックを実行します。 要求が確認されると、トランザクタは管理情報ツリーを更新します。 このプロセスは、単一のトランザクションで実行されます。

次の図に、CIMC がサーバの起動要求をどのように処理するかを示し、それに続く表に、サーバの起動要求に含まれる手順を示します。

図 *2*：サーバの起動要求のサンプル フロー

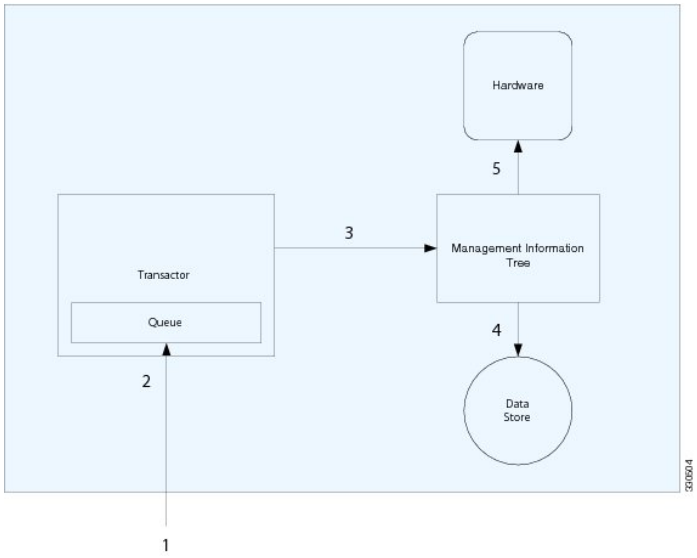

表 *1*：サーバの起動要求の説明

ステップ	コマンド/プロセス	MO（サーバ）の動作上の電源状態
1	CMD 要求：サーバの起動	Down
2	要求をキューに投入	Down
3	管理情報ツリーでのステートの変更	Down

ステップ	コマンド/プロセス	MO（サーバ）の動作上の電源状態
4	管理対象オブジェクト（MO）のステート変更の永続化	Down
5	ブート スティミュラスの適用	Up

オブジェクトの命名

特定のオブジェクトは、識別名（DN）または相対名（RN）で識別できます。

識別名

識別名を使用すると、明確にターゲットオブジェクトを識別することができます。識別名は、一連の相対名から構成される次の形式を持ちます。

```
dn = {rn}/{rn}/{rn}/{rn}...
```

次の例で DN は、オブジェクト ツリーの最上位からオブジェクトまで、`adaptor-1` の完全修飾パスを提供します。 DN は、API コールが動作する管理対象オブジェクトを指定します。

```
< dn =”sys/rack-unit-1/adaptor-1” />
```

相対名

相対名は、親オブジェクトのコンテキスト内でオブジェクトを識別します。 識別名は、一連の相対名で構成されます。

次の識別名を例にします。

```
<dn = "sys/rack-unit-1/adaptor-1/host-eth-2"/>
```

これは、次の相対名で構成されます。

```
topSystem MO: rn="sys"
computeRackUnit MO: rn ="rack-unit-1"
adaptorUnit MO: rn="adaptor-<id>"
adaptorHostEthIf MO: rn="host-eth-<id>"
```

API メソッド カテゴリ

各メソッドは XML ドキュメントに対応します。

（注） このマニュアルのいくつかのコード例では、用語 <real_cookie> は 1217377205/85f7ff49-e4ec-42fc-9437-da77a1a2c4bf などの実際の Cookie に置き換えられます。XML API の Cookie は 47 文字の文字列です。これは、セッション情報を維持するために Web ブラウザがローカルに保存する Cookie とは種類が異なります。

認証方法

認証メソッドは、アクティブセッションを開始し、維持します。 他の API コールが許可される前に、正常な認証 Cookie がシステムによって返される必要があります。

セッションを認証するには、次のメソッドを使用します。

- aaaLogin：成功した場合は、セッションを初期化し認証 Cookie を返します。 Cookie は、600 秒（10 分）の間有効で、期限が切れないようにするためにセッション期間中に更新する必要があります。 最大 4 つの CIMC へのセッションを一度に開くことができます。
- aaaRefresh：以前にアクティブなセッションの代わりに新しいセッションを作成します。 新しいセッションが作成されると、新しい認証 Cookie が返されます。
- aaaKeepAlive：セッションを維持し、認証 Cookie を再び 600 秒間アクティブなままにします。
- aaaLogout：現在のセッションを終了し、対応する認証 Cookie を非アクティブ化します。

操作は、TCP 上で HTTP の post メソッドを使用して実行されます（CIMC は HTTP 要求と HTTPS 要求の両方をサポートします）。 HTTP および HTTPS が別のポート番号を使用するように設定できますが、TCP/80（またはセキュア接続用に TCP/443）がデフォルトで使用されます。 HTTP のエンベロープには XML の設定が含まれます。

ヒント CIMC では、HTTP から HTTPS へのリダイレクトはデフォルトでイネーブルです。 クライアント アプリケーションと CIMC の間の HTTP パケットをキャプチャするには、CIMC GUI または CLI のリダイレクトをディセーブルにします。

クエリー メソッド

クエリー メソッドは、オブジェクトの現在の設定状態情報を取得します。 次に示すのは、クエリー メソッド（リリース 1.4(1.x) でサポート）です。

- configResolveDn：DN によりオブジェクトを取得します。
- configResolveClass：該当するクラスのオブジェクトを取得します。
- configResolveChildren：オブジェクトの子オブジェクトを取得します。

- configResolveParent：オブジェクトの親オブジェクトを取得します。

ほとんどのクエリー メソッドは、引数 inHierarchical（ブール値 true/yes または false/no）を持ちます。 true の場合、inHierarchical 引数はすべての子オブジェクトを返します。

```
<configResolveDn … inHierarchical="false"></>
<configResolveDn … inHierarchical="true"></>
```

CIMC から返されるデータ量は非常に大きいことがあるため、inHierarchical 引数は慎重に使用してください。たとえば、クエリーメソッドが、管理情報ツリーの上部にある管理対象オブジェクト（MO）を参照するクラスまたは DN で使われていて、inHierarchical が true に設定されている場合、応答には CIMC の設定全体のほとんどが含まれる可能性があります。 CIMC が要求を処理するために必要なリソースが多くなると、CIMC の応答にかかる時間が長くなります。 遅延を回避するには、クエリー メソッドを少数の MO に関連する小さな規模で実行する必要があります。

ヒント クエリーメソッドが応答しない、または応答に時間がかかる場合は、クライアントアプリケーションのタイムアウト期間を長くするか、関連する MO の数を減らすようにクエリー メソッドを調整します。

クエリーの API メソッドには、コールを再帰的にするかどうかを指定するために inRecursive 引数が含まれる場合があります（他のオブジェクトまたは親オブジェクトをポイントし返す場合など）。

この API は、クエリー メソッドの有用性を高めるためにフィルタ セットを提供します。 これらのフィルタは、クエリーの一部として渡すことができ、必要な結果セットを特定するために使用されます。

（注） リリース 1.4(1.x) では、inRecursive 引数およびクエリー フィルタはサポートされず、指定された場合は無視されます。

（注） ホストの電源が少なくとも 1 回投入されるまでに、CIMC はインベントリおよびステータス情報の取得を完了していない場合があります。 たとえば、CIMC がリセットされた場合は、ホストの電源が次にオンになるまで、CPU、メモリ、またはアダプタの詳細なインベントリ情報は取得されません。 使用できないデータに対応する MO でクエリー メソッドが実行された場合、応答は空白になります。

設定メソッド

Cisco CIMC XML API では、管理対象オブジェクトの設定変更を行うために単一のメソッドのみがサポートされます。

- configConfMo：管理情報ツリーの 1 つの管理対象オブジェクト（たとえば、DN）に影響します。

イベント サブスクリプション メソッド

アプリケーションは、通常のポーリングまたはイベント サブスクリプションによってステート変更に関する情報を取得します。 リソースをより効率的に使うために、イベント サブスクリプションは通知に最適な方法です。 ポーリングは非常に限定的な状況にある場合だけ使用してください。

次の例に示すように、イベントに対して登録するために eventSubscribe を使用します。

```
<eventSubscribe
    cookie="<real_cookie>">
</eventSubscribe>
```

通知を受信するには、TCP 上で HTTP または HTTPS セッションを開き、このセッションを開いたままにします。 eventSubscribe の受信時に、CIMC は新しいイベントが発生すると、それらすべての送信を開始します。

各イベントは一意のイベント ID を持ちます。 イベント ID はカウンタとして動作し、すべてのメソッドの応答に含まれます。 イベントが生成されると、イベント ID カウンタが増加し、新しいイベント ID が割り当てられます。 このシーケンス番号により、イベントの追跡が可能になり、イベントを見逃すことがなくなります。

ユーザが開始したイベント チャネル接続は、イベント チャネル セッションの Cookie に関連付けられた非アクティビティが 600 秒経過してから、CIMC によって自動的に切断されます。 イベント チャネル接続が CIMC によって自動的に閉じられることを防ぐには、ユーザは 600 秒以内に同じイベント チャネル セッション Cookie に対して aaaKeepAlive 要求を送信するか、または同じイベント チャネル セッション Cookie を使用して CIMC に他の XML API メソッドを送信する必要があります。

（注） リリース 1.4(1.x) では、 Cisco CIMC XML API はエラーに関連するイベントのみについてイベント通知を送信します。

成功または失敗の応答

CIMC が XML API 要求に応答する場合、応答は要求が完了できない場合に失敗を示します。 成功の応答は、要求が有効かどうかだけを示し、操作が完了したことは示しません。 たとえば、電源投入の要求をサーバが完了するには時間がかかることがあります。 電源状態は、サーバの電源が実際に投入されてからのみダウンからアップに変更されます。

成功した要求

要求が正常に実行されると、CIMCは要求された情報または変更が行われたことの確認を含むXMLドキュメントを返します。 次に、識別名 sys/rack-unit-1/adaptor-2/ext-eth-0 での configResolveDn クエリーの例を示します。

```
<configResolveDn
     dn="sys/rack-unit-1/adaptor-2/ext-eth-0"
     cookie="<real_cookie>"
     inHierarchical="false"/>
```

応答には、次の情報が含まれます。

```
<configResolveDn
    cookie="<real_cookie>"
    response="yes"
    dn="sys/rack-unit-1/adaptor-2/ext-eth-0">
    <outConfig>
        <adaptorExtEthIf
            id="0"
            ifType="physical"
            linkState="up"
            mac="00:22:BD:D6:42:DA"
            name=""
            operState="up"
            portId="0"
            purpose="general"
            transport="CE"
            type=""
            dn="sys/rack-unit-1/adaptor-2/ext-eth-0" >
        </adaptorExtEthIf>
    </outConfig>
</configResolveDn>
```

失敗した要求

失敗した要求への応答には、errorCode および errorDescr の XML 属性が含まれます。 次に、失敗した要求に対する応答の例を示します。

```
<configConfMo dn="sys/rack-unit-1/adaptor-1/ext-eth-0"
     cookie="<real_cookie>"
     response="yes"
     errorCode="103"
     invocationResult="unidentified-fail"
     errorDescr="can't create; object already exists.">
</configConfMo>
```

空の結果

存在しないオブジェクトに対するクエリー要求は、CIMC によって失敗として扱われません。 オブジェクトが存在しない場合、CIMC は成功メッセージを返しますが、XML ドキュメントには、要求されたオブジェクトが見つからなかったことを示すために空のデータ フィールド（<outConfig>

</outConfig>）が含まれます。 次に、存在しないオブジェクトの識別名を解決する試みに対する応答の例を示します。

```
<configResolveDn
    cookie="<real_cookie>"
    response="yes"
    dn="sys/rack-unit-1/adaptor-9999">
    <outConfig>
    </outConfig>
</configResolveDn>
```

空の結果

第 2 章

Cisco CIMC XML API メソッドの使用

この章の内容は、次のとおりです。

認証方法

認証は CIMC との XML API の交換を可能にします。 認証を使用すると、権限を設定し、実行できる操作を制御できます。

（注） このマニュアルのほとんどのコード例では、用語 `<real_cookie>` は 1217377205/85f7ff49-e4ec-42fc-9437-da77a1a2c4bf などの実際の Cookie に置き換えられます。 XML API の Cookie は 47 文字の文字列です。これは、セッション情報を維持するために Web ブラウザがローカルに保存する Cookie とは種類が異なります。

ログイン

ログインするために、XML API クライアントは CIMC HTTP（または HTTPS）サーバへの TCP 接続を確立し、aaaLogin メソッドを格納している XML ドキュメントをポストします。

次の例では、IP アドレス 192.0.20.72 で CIMC のポート 80 への TCP 接続を確立するために、Telnet ユーティリティを使用しています。 使用するパスは /nuova です。

```
$ telnet 192.0.20.72 80
POST /nuova HTTP/1.1
USER-Agent: lwp-request/2.06
HOST: 192.0.20.72
Content-Length: 62
Content-Type: application/x-www-form-urlencoded
```

次に、クライアントは aaaLogin メソッドを指定し、ユーザ名とパスワードを提供します。

```
<aaaLogin
    inName='admin'
    inPassword='password'>
</aaaLogin>
```

（注） XML API ドキュメントに XML バージョンまたは DOCTYPE の行を含めないでください。inName 属性と inPassword 属性はパラメータです。

各 XML API ドキュメントは、実行する操作を表します。 要求が XML API ドキュメントとして受け取られると、CIMC は要求を読み取り、メソッドで指定されているアクションを実行します。CIMC は、XML ドキュメント形式のメッセージで応答し、要求の成否を示します。

次に、一般的な成功応答を示します。

```
1 <aaaLogin
2    response="yes"
3    outCookie="<real_cookie>"
4       outRefreshPeriod="600"
5       outPriv="admin">
6 </aaaLogin>
```

応答の各行は、次のように解釈されます。

1. ログインに使用するメソッドを指定します。
2. これが応答であることを確認します。
3. セッションの Cookie を提供します。
4. 推奨される Cookie のリフレッシュ間隔を指定します。 デフォルトのログイン セッションの長さは 600 秒です。
5. ユーザアカウントに割り当てられる権限レベルを指定します（admin、user、または readonly）。
6. 終了タグ。

また、次の例に示すように XML API へのログインに cURL ユーティリティを使用できます。

```
curl -d "<aaaLogin inName='admin' inPassword='password'></aaaLogin>" http://192.0.20.72/nuova
```

HTTPS を有効にした場合は、次の例に示すように、cURL のコマンドで HTTPS を使用する必要があります。

```
curl -d "<aaaLogin inName='admin' inPassword='password'></aaaLogin>" https://192.0.20.72/nuova
```

セッションの更新

セッションは、aaaLogin 応答または以前の更新から取得された 47 文字の Cookie を使用して、aaaRefresh メソッドで更新されます。

```
<aaaRefresh
    cookie="<real_cookie>"
    inCookie="<real_cookie>"
    inName='admin'
```

```
    inPassword='password'>
</aaaRefresh>
```

セッションからのログアウト

セッションからログアウトするには、次のメソッドを使用します。

```
<aaaLogout
    cookie="<real_cookie>"
    inCookie="<real_cookie>"
</aaaLogout>
```

失敗応答の例

失敗したログイン：

```
<aaaLogin
    cookie=""
    response="yes"
    errorCode="551"
    invocationResult="unidentified-fail"
    errorDescr="Authentication failed">
</aaaLogin>
```

存在しないオブジェクト（空白が返される場合は、指定した DN オブジェクトが存在しないことを示します）：

```
<configResolveDn
    cookie="<real_cookie>"
    response="yes"
    dn="sys/rack-unit-1/adaptor-9999">
    <outConfig>
    </outConfig>
</configResolveDn>
```

不正な要求：

```
<configConfMo
    cookie="<real_cookie>"
    response="yes"
    dn="sys/rack-unit-1/adaptor-1/ext-eth-0">
    errorCode="103"
    invocationResult="unidentified-fail"
    errorDescr="can't create; object already exists.">
</configConfMo>
```

クエリー メソッド

configResolveChildren の使用

管理情報ツリーの子オブジェクトの解決時に、次のことに注意してください。

- このメソッドが、名前付きクラスのインスタンスである名前付きオブジェクトのすべての子オブジェクトを取得している。 クラス名を省略すると、名前付きオブジェクトのすべての子オブジェクトが返されます。
- inDn 属性により、子オブジェクトが取得される名前付きオブジェクトが指定されている（必須）。
- classId 属性により、返される子オブジェクト クラスの名前が指定されている（任意）。
- 認証 Cookie（aaaLogin または aaaRefresh から）が必須である。
- inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。
- 列挙値、`classIds`、およびビット マスクが文字列として表示されている。

configResolveChildren, （23 ページ）に示す要求または応答の例を参照してください。

configResolveClass の使用

クラスの解決時には、次のことに注意してください。

- 指定したクラス タイプのすべてのオブジェクトが取得されている。
- classId が、返されるオブジェクト クラスの名前を指定している（必須）。
- 認証 Cookie（aaaLogin または aaaRefresh から）が必須である。
- inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。
- 列挙値、`classIds`、およびビット マスクが文字列として表示されている。

結果セットは大きくなることがあります。結果セットは正確に定義してください。たとえば、アダプタのリストだけを取得する場合は、クエリーで classId の属性値として adaptorUnit を使用します。 この例は、`adaptorUnit` クラスのすべてのインスタンスについて問い合わせます。

```
<configResolveClass
 cookie="real_cookie"
 inHierarchical="false"
 classId="adaptorUnit"/>
```

configResolveClass, （25 ページ）に示す要求または応答の例を参照してください。

configResolveDn の使用

DN を解決するときは、次のことに注意してください。

- DN により指定されたオブジェクトが取得されている。
- 指定された DN が、解決するオブジェクト インスタンスを識別している（必須）。

- 認証 Cookie（aaaLogin または aaaRefresh から）が必須である。
- inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。
- 列挙値、`classIds`、およびビット マスクが文字列として表示されている。

configResolveDn, （26 ページ）に示す要求または応答の例を参照してください。

configResolveParent の使用

オブジェクトの親オブジェクトの解決時に、次のことに注意してください。

- このメソッドが、指定 DN の親オブジェクトを取得している。
- dn 属性が子オブジェクトの DN である（必須）。
- 認証 Cookie（aaaLogin または aaaRefresh から）が必須である。
- inHierarchical 属性（デフォルト値は false）が true の場合は、結果が階層形式であることが指定されている。
- 列挙値、`classIds`、およびビット マスクが文字列として表示されている。

configResolveParent, （27 ページ）に示す要求または応答の例を参照してください。

第 3 章

Cisco CIMC XML API メソッドの説明

この章の内容は、次のとおりです。

aaaKeepAlive

aaaKeepAlive メソッドは、メソッド呼び出し後に同じ Cookie を使用して、デフォルトのセッション時間が経過するまでセッションをアクティブなままにします。

要求構文

```
<xs:element name="aaaKeepAlive" type="aaaKeepAlive" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaKeepAlive" mixed="true">
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

応答構文

```
<xs:element name="aaaKeepAlive" type="aaaKeepAlive" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaKeepAlive" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

例

要求

```
<aaaKeepAlive
    cookie="<real_cookie>"
</aaaKeepAlive>
```

応答

```
<aaaKeepAlive
    cookie="<real_cookie>"
    response="yes">
</aaaKeepAlive>
```

aaaLogin

aaaLogin メソッドはログイン プロセスで、セッションを開始するために必要です。 この動作は、クライアントと CIMC の間の HTTP（または HTTPS）セッションを確立します。

要求構文

```
<xs:element name="aaaLogin" type="aaaLogin" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaLogin" mixed="true">
            <xs:attribute name="inName" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inPassword" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="stringMin0Max47"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

応答構文

```
<xs:element name="aaaLogin" type="aaaLogin" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaLogin" mixed="true">
```

```
        <xs:attribute name="outCookie" type="xs:string"/>
        <xs:attribute name="outRefreshPeriod" type="xs:unsignedInt"/>
    <xs:attribute name="outPriv">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:pattern value="(read-only|admin|user){0,1}"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
        <xs:attribute name="outDomains" type="xs:string"/>
        <xs:attribute name="outChannel">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                     <xs:enumeration value="fullssl"/>
                     <xs:enumeration value="noencssl"/>
                     <xs:enumeration value="plain"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="outEvtChannel">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                     <xs:enumeration value="fullssl"/>
                     <xs:enumeration value="noencssl"/>
                     <xs:enumeration value="plain"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="outSessionId">
            <xs:simpleType>
                <xs:restriction base="xs:string">
                     <xs:minLength value="0"/>
                     <xs:maxLength value="32"/>
                </xs:restriction>
            </xs:simpleType>
        </xs:attribute>
        <xs:attribute name="outVersion" type="xs:string"/>
        <xs:attribute name="cookie" type="xs:string"/>
        <xs:attribute name="response" type="YesOrNo"/>
        <xs:attribute name="errorCode" type="xs:unsignedInt"/>
        <xs:attribute name="errorDescr" type="xs:string"/>
        <xs:attribute name="invocationResult" type="xs:string"/>
    </xs:complexType>
```

例

要求

```
<aaaLogin
    inName='admin'
    inPassword='password'>
</aaaLogin>
```

応答

```
<aaaLogin
    cookie=""
    response="yes"
    outCookie="<real_cookie>"
    outRefreshPeriod="600"
    outPriv="admin">
</aaaLogin>
```

aaaLogout

aaaLogout メソッドは、入力としてセッションの Cookie を渡すことによって Web セッションを閉じるプロセスです。 これは自動では行われません。ユーザはセッションを終了するために、aaaLogout メソッドを明示的に呼び出す必要があります。

要求構文

```
<xs:element name="aaaLogout" type="aaaLogout" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaLogout" mixed="true">
            <xs:attribute name="inCookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="cookie" type="stringMin0Max47"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

応答構文

```
<xs:element name="aaaLogout" type="aaaLogout" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaLogout" mixed="true">
            <xs:attribute name="outStatus">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="success"/>
                         <xs:enumeration value="failure"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

例

要求

```
<aaaLogout
    cookie="<real_cookie>"
    inCookie="<real_cookie>"
</aaaLogout>
```

応答

```
<aaaLogout
    cookie="<real_cookie>"
    response="yes"
    outStatus="success">
</aaaLogout>
```

aaaRefresh

aaaRefresh メソッドは、ユーザ アクティビティによってセッションをアクティブなままにします（デフォルトのセッション期間中）。 デフォルトでは、アクティビティがない時点から 600 秒カウントダウンされます。 600 秒を超えると、CIMC はスリープ モードに入ります。 戻るためにはサインインが必要で、その後カウントダウンが再開されます。 これは同じセッション ID を引き続き使用します。

（注） このメソッドを使用すると、以前の Cookie の有効期限が切れ、新しい Cookie が発行されます。

要求構文

```
<xs:element name="aaaRefresh" type="aaaRefresh" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaRefresh" mixed="true">
            <xs:attribute name="inName" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:pattern value="[\-\.:_a-zA-Z0-9]{0,16}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inPassword" use="required">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:minLength value="0"/>
                         <xs:maxLength value="510"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="inCookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="cookie" type="stringMin0Max47"/>
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

応答構文

```
<xs:element name="aaaRefresh" type="aaaRefresh" substitutionGroup="externalMethod"/>
        <xs:complexType name="aaaRefresh" mixed="true">
            <xs:attribute name="outCookie" type="xs:string"/>
            <xs:attribute name="outRefreshPeriod" type="xs:unsignedInt"/>
            <xs:attribute name="outPriv">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                        <xs:pattern value="(read-only|admin|user){0,1}"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="outDomains" type="xs:string"/>
            <xs:attribute name="outChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
```

```
            </xs:attribute>
            <xs:attribute name="outEvtChannel">
                <xs:simpleType>
                    <xs:restriction base="xs:string">
                         <xs:enumeration value="fullssl"/>
                         <xs:enumeration value="noencssl"/>
                         <xs:enumeration value="plain"/>
                    </xs:restriction>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

例

要求

```
<aaaRefresh
    cookie="<real_cookie>"
    inCookie="<real_cookie>"
    inName='admin'
    inPassword='password'>
</aaaRefresh>
```

応答

```
<aaaRefresh
    cookie="<real_cookie>"
    response="yes"
    outCookie="<real_cookie>"
    outRefreshPeriod="600"
    outPriv="admin">
</aaaRefresh>
```

configConfMo

configConfMo メソッドは 1 つのサブツリーで指定の管理対象オブジェクトを設定します（DN など）。

要求構文

```
<xs:element name="configConfMo" type="configConfMo" substitutionGroup="externalMethod"/>
        <xs:complexType name="configConfMo" mixed="true">
            <xs:all>
                <xs:element name="inConfig" type="configConfig" minOccurs="1"/>
            </xs:all>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean YesOrNo"/>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject" use="required"/>
        </xs:complexType>
```

応答構文

```
<xs:element name="configConfMo" type="configConfMo" substitutionGroup="externalMethod"/>
        <xs:complexType name="configConfMo" mixed="true">
            <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

例

要求

```
<configConfMo
    cookie="<real_cookie>"
    dn='sys/rack-unit-1/locator-led'>
    <inConfig>
        <equipmentLocatorLed
            adminState='on'
            dn='sys/rack-unit-1/locator-led'>
        </equipmentLocatorLed>
    </inConfig>
</configConfMo>
```

応答

```
<configConfMo
    dn="sys/rack-unit-1/locator-led"
    cookie="<real_cookie>"
    response="yes">
    <outConfig>
        <equipmentLocatorLed
            dn="sys/rack-unit-1/locator-led"
            adminState="inactive"
            color="unknown"
            id="1"
            name=""
            operState="off">
        </equipmentLocatorLed>
    </outConfig>
</configConfMo>
```

configResolveChildren

configResolveChildren メソッドは、管理情報ツリーの特定の DN 下の管理対象オブジェクトの子を取得します。 返される子の数を減らすためにフィルタを使用できます。

要求構文

```
<xs:element name="configResolveChildren" type="configResolveChildren"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveChildren" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
```

```
            <xs:attribute name="inDn" type="referenceObject" use="required"/>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean YesOrNo"/>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="classId" type="namingClassId"/>
        </xs:complexType>
```

応答構文

```
<xs:element name="configResolveChildren" type="configResolveChildren"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveChildren" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="classId" type="namingClassId"/>
        </xs:complexType>
```

例

要求

```
<configResolveChildren
    cookie="<real_cookie>"
    inDn='sys/rack-unit-1/boot-policy'
    inHierarchical='false'>
</configResolveChildren>
```

応答

```
<configResolveChildren
    cookie="0000227746/06bc6a70-0035-1035-800c-cdac38e14388"
    response="yes"
    dn="sys/rack-unit-1/boot-policy">
    <outConfig>
        <lsbootVirtualMedia
            access="read-write"
            order="2"
            type="virtual-media"
            dn="sys/rack-unit-1/boot-policy/vm-read-write">
        </lsbootVirtualMedia>
        <lsbootLan
            access="read-only"
            order="1" prot="pxe"
            type="lan"
            dn="sys/rack-unit-1/boot-policy/lan-read-only">
        </lsbootLan>
        <lsbootStorage
            access="read-write"
            order="4"
            type="storage"
            dn="sys/rack-unit-1/boot-policy/storage-read-write">
        </lsbootStorage>
        <lsbootEfi
            access="read-only"
            order="3"
            type="efi"
```

```
            dn="sys/rack-unit-1/boot-policy/efi-read-only">
        </lsbootEfi>
    </outConfig>
</configResolveChildren>
```

configResolveClass

configResolveClass メソッドは、特定のクラスの要求された管理対象オブジェクトを返します。inHierarchical=true の場合、結果には子が含まれます。

要求構文

```
<xs:element name="configResolveClass" type="configResolveClass"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveClass" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean YesOrNo"/>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="classId" type="namingClassId" use="required"/>
        </xs:complexType>
```

応答構文

```
<xs:element name="configResolveClass" type="configResolveClass"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveClass" mixed="true">
            <xs:all>
                <xs:element name="outConfigs" type="configSet" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="classId" type="namingClassId"/>
        </xs:complexType>
```

例

要求

```
<configResolveClass
    cookie="<real_cookie>"
    classId='computeRackUnit'
    inHierarchical='false'>
</configResolveClass>
```

応答

```
<configResolveClass
    cookie="<real_cookie>"
    response="yes"
```

```
    classId="computeRackUnit">
    <outConfig>
        <computeRackUnit
            dn="sys/rack-unit-1"
            adminPower="policy"
            availableMemory="16384"
            model="R210-2121605W"
            memorySpeed="1067"
            name="UCS C210 M2"
            numOfAdaptors="2"
            numOfCores="8"
            numOfCoresEnabled="8"
            numOfCpus="2"
            numOfEthHostIfs="5"
            numOfFcHostIfs="2"
            numOfThreads="16"
            operPower="on"
            originalUuid="00C9DE3C-370D-DF11-1186-6DD1393A608B"
            presence="equipped"
            serverID="1"
            serial="QCI140205Z2"
            totalMemory="16384"
            usrLbl="C210 Row-B Rack-10"
            uuid="00C9DE3C-370D-DF11-1186-6DD1393A608B"
            vendor="Cisco Systems Inc" >
        </computeRackUnit>
    </outConfig>
</configResolveClass>
```

configResolveDn

configResolveDn メソッドは、指定された DN の 1 つの管理対象オブジェクトを取得します。

要求構文

```
<xs:element name="configResolveDn" type="configResolveDn" substitutionGroup="externalMethod"/>

        <xs:complexType name="configResolveDn" mixed="true">
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean YesOrNo"/>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject" use="required"/>
        </xs:complexType>
```

応答構文

```
<xs:element name="configResolveDn" type="configResolveDn"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveDn" mixed="true">
            <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

例

要求

```
<configResolveDn
    dn='sys/rack-unit-1/adaptor-2/ext-eth-0'
    cookie="<real_cookie>"
    inHierarchical='false'>
</configResolveDn>
```

応答

```
<configResolveDn
    cookie="<real_cookie>"
    response="yes"
    dn="sys/rack-unit-1/adaptor-2/ext-eth-0">
    <outConfig>
        <adaptorExtEthIf
            id="0"
            ifType="physical"
            linkState="up"
            mac="00:22:BD:D6:42:DA"
            name=""
            operState="up"
            portId="0"
            purpose="general"
            transport="CE"
            type=""
            dn="sys/rack-unit-1/adaptor-2/ext-eth-0" >
        </adaptorExtEthIf>
    </outConfig>
</configResolveDn>
```

configResolveParent

指定された DN について、configResolveParent メソッドは管理対象オブジェクトの親を取得します。

要求構文

```
<xs:element name="configResolveParent" type="configResolveParent"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveParent" mixed="true">
            <xs:attribute name="inHierarchical">
                <xs:simpleType>
                    <xs:union memberTypes="xs:boolean YesOrNo"/>
                </xs:simpleType>
            </xs:attribute>
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="dn" type="referenceObject" use="required"/>
        </xs:complexType>
```

応答構文

```
<xs:element name="configResolveParent" type="configResolveParent"
substitutionGroup="externalMethod"/>
        <xs:complexType name="configResolveParent" mixed="true">
            <xs:all>
                <xs:element name="outConfig" type="configConfig" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="xs:string"/>
```

```
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
            <xs:attribute name="dn" type="referenceObject"/>
        </xs:complexType>
```

例

要求

```
<configResolveParent
    cookie="<real_cookie>"
    dn='sys/rack-unit-1/boot-policy/efi-read-only'
    inHierarchical='false'>
</configResolveParent>
```

応答

```
<configResolveParent
    cookie="<real_cookie>"
    response="yes"
    dn="sys/rack-unit-1/boot-policy/efi-read-only">
    <outConfig>
        <lsbootDef
            dn="sys/rack-unit-1/boot-policy"
            name="boot-policy"
            purpose="operational"
            rebootOnUpdate="no">
        </lsbootDef>
    </outConfig>
</configResolveParent>
```

eventSubscribe

eventSubscribe メソッドによって、CIMC が生成した非同期のシステム イベント ログ（SEL）イベントをクライアントがサブスクライブできるようになります。

イベント サブスクリプションでは、クライアント アプリケーションが CIMC からのイベント通知を受けるように登録できます。 イベントが発生したときに、CIMC はクライアント アプリケーションにイベントとそのタイプを通知します。 実際の変更情報だけが送信されます。 オブジェクトの影響を受けない属性は含まれません。

次の例に示すように、イベントに登録するために eventSubscribe を使用します。

```
<eventSubscribe
    cookie="<real_cookie>">
</eventSubscribe>
```

要求構文

```
<xs:element name="eventSubscribe" type="eventSubscribe" substitutionGroup="externalMethod"/>

        <xs:complexType name="eventSubscribe" mixed="true">
            <xs:all>
                <xs:element name="inFilter" type="filterFilter" minOccurs="0"/>
            </xs:all>
            <xs:attribute name="cookie" type="stringMin0Max47" use="required"/>
```

```
            <xs:attribute name="response" type="YesOrNo"/>
        </xs:complexType>
```

応答構文

```
<xs:element name="eventSubscribe" type="eventSubscribe"
substitutionGroup="externalMethod"/>
        <xs:complexType name="eventSubscribe" mixed="true">
            <xs:attribute name="cookie" type="xs:string"/>
            <xs:attribute name="response" type="YesOrNo"/>
            <xs:attribute name="errorCode" type="xs:unsignedInt"/>
            <xs:attribute name="errorDescr" type="xs:string"/>
            <xs:attribute name="invocationResult" type="xs:string"/>
        </xs:complexType>
```

例

要求

```
<eventSubscribe
    cookie="<real_cookie>">
</eventSubscribe>
```

応答

```
NO RESPONSE OR ACKNOWLEDGMENT.
```

eventSubscribe

第 4 章

CIMC XML オブジェクト アクセス権限

この章の内容は、次のとおりです。

権限のサマリー テーブル

ユーザがロールに割り当てられると、そのロールが特定の権限を許可します。 これらの権限は、特定のシステム リソースへのアクセスをユーザに許可し、これらのリソースでタスクを実行する権限を許可します。 次の表に、各権限と、その権限に付与されている初期のデフォルト ユーザ ロールを示します。

表 *2*：権限のサマリー

内部名	Label	説明
admin，（32 ページ）	ADMIN	すべてへのアクセス
read-only，（32 ページ）	READ_ONLY	読み取り専用アクセス権
user，（32 ページ）	USER	制限付きコンフィギュレーション アクセス

権限

admin

目的

システム管理

担当するロール

Administrator

制御対象オブジェクト

このロールは、システム レベルです。 管理者は、すべてのオブジェクトを制御します。

read-only

目的

読み取り専用アクセス権

担当するロール

これは、選択可能な権限ではありません。 すべてのロールがすべてのオブジェクトへの読み取り専用アクセス権を持ちます。 一部のオブジェクトの読み取り/書き込み権限を持つロールが、その他すべてのオブジェクトに対する読み取り専用アクセス権を持ちます。

user

目的

設定の制限

担当するロール

テキストが必要

制御対象オブジェクト

このロールは、次のタスクを実行できます。

- すべての情報を表示する
- 電源のオン、電源再投入、電源のオフなどの電力制御オプションを管理する

- KVM コンソールと仮想メディアを起動する
- すべてのログをクリアする
- ロケータ LED を切り替える

user

付 録 A

共通サーバ管理タスクの例

この章の例は、 Cisco CIMC XML API を使用して共通サーバ管理タスクを実行する方法を示します。 各例は、XML API 要求に続いて CIMC からの応答を示しています。

この章には、次の例があります。

サーバの要約情報とホストの電源状態の取得

要求：

```
<configResolveClass cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388"
inHierarchical="false" classId="computeRackUnit"/>
```

応答：

```
<configResolveClass cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388"
response="yes" classId="computeRackUnit">
  <outConfig>
    <computeRackUnit dn="sys/rack-unit-1" adminPower="policy" availableMemory="49152"
     model="R250-2480805W" memorySpeed="1067" name="UCS C250 M2" numOfAdaptors="1"
     numOfCores="8" numOfCoresEnabled="8" numOfCpus="2" numOfEthHostIfs="8"
     numOfFcHostIfs="2" numOfThreads="16" operPower="on"
     originalUuid="100DC440-0EBC-11DF-3B97-8843E1C2615E" presence="equipped" serverId="1"

     serial="PGS140601CS" totalMemory="49152" usrLbl="Cisco C250 Server"
     uuid="100DC440-0EBC-11DF-3B97-8843E1C2615E" vendor="Cisco Systems Inc"/>
  </outConfig>
</configResolveClass>
```

サーバ コンポーネントの現在実行中のファームウェア バージョンの取得

要求：

```
<configResolveClass cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388"
inHierarchical="false" classId="firmwareRunning"/>
```

応答：

```
<configResolveClass cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388" response="yes"

classId="firmwareRunning">
  <outConfig>
    <firmwareRunning dn="sys/rack-unit-1/bios/fw-boot-loader" deployment="boot-loader"
        type="blade-bios" version="C250.1.4.0.2 (Build Date: 05/20/2011)"/>
    <firmwareRunning dn="sys/rack-unit-1/mgmt/fw-boot-loader" deployment="boot-loader"
        type="blade-controller" version="66.77(67.1305573810).16"/>
    <firmwareRunning dn="sys/rack-unit-1/mgmt/fw-system" deployment="system"
        type="blade-controller" version="1.4(0.22)"/>
    <firmwareRunning dn="sys/rack-unit-1/adaptor-4/mgmt/fw-boot-loader"
        deployment="boot-loader" type="adaptor" version="1.0(0.152)"/>
    <firmwareRunning dn="sys/rack-unit-1/adaptor-4/mgmt/fw-system"
        deployment="system" type="adaptor" version="1.6(0.11)"/>
  </outConfig>
</configResolveClass>
```

CIMC にインストールされているバックアップ ファームウェア バージョンの取得

要求：

```
<configResolveDn cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388"
inHierarchical="false" dn="sys/rack-unit-1/mgmt/fw-updatable"/>
```

応答：

```
<configResolveDn cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388" response="yes"

        dn="sys/rack-unit-1/mgmt/fw-updatable">
```

```
  <outConfig>
    <firmwareUpdatable dn="sys/rack-unit-1/mgmt/fw-updatable" adminState="triggered"
        deployment="backup" version="1.4(0.21)"/>
  </outConfig>
</configResolveDn>
```

inHierarchical オプションを使用した、設定済みブート順テーブルの取得

要求：

```
<configResolveClass cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388"
inHierarchical="true" classId="lsbootDef"/>
```

応答：

```
<configResolveClass cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388" response="yes"

      classId="lsbootDef">
  <outConfig>
    <lsbootDef dn="sys/rack-unit-1/boot-policy" name="boot-policy"
          purpose="operational" rebootOnUpdate="no" childAction="deleteNonPresent">
      <lsbootVirtualMedia access="read-only" order="3"
          type="virtual-media" rn="vm-read-only" childAction="deleteNonPresent"/>
      <lsbootVirtualMedia access="read-write" order="5"
          type="virtual-media" rn="vm-read-write" childAction="deleteNonPresent"/>
      <lsbootLan rn="lan-read-only" access="read-only" order="2"
          prot="pxe" type="lan" childAction="deleteNonPresent"/>
      <lsbootStorage rn="storage-read-write" access="read-write" order="1"
          type="storage" childAction="deleteNonPresent">
         <lsbootLocalStorage rn="local-storage" childAction="deleteNonPresent"/>
      </lsbootStorage>
      <lsbootEfi rn="efi-read-only" access="read-only" order="4"
          type="efi" childAction="deleteNonPresent"/>
    </lsbootDef>
  </outConfig>
</configResolveClass>
```

サーバのすべての電源装置ユニットに関する詳細の取得

要求：

```
<configResolveClass cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388"
inHierarchical="false" classId="equipmentPsu"/>
```

応答：

```
<configResolveClass cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388" response="yes"

      classId="equipmentPsu">
  <outConfig>
    <equipmentPsu id="0" model="R250-PSU2-750W" operability="operable" power="on"
      presence="equipped" serial="ART1348S039" thermal="unknown" vendor="Cisco Systems
Inc"
      voltage="unknown" dn="sys/rack-unit-1/psu-0" childAction="deleteNonPresent"/>
    <equipmentPsu id="1" model="" operability="unknown" power="off" presence="missing"
      serial="" thermal="unknown" vendor="" voltage="unknown" dn="sys/rack-unit-1/psu-1"

      childAction="deleteNonPresent"/>
  </outConfig>
</configResolveClass>
```

DN によって識別されるファン オブジェクトに関する詳細の取得

要求：

```
<configResolveDn cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388"
inHierarchical="false" dn="sys/rack-unit-1/fan-module-1-2/fan-2"/>
```

応答：

```
<configResolveDn cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388" response="yes"

     dn="sys/rack-unit-1/fan-module-1-2/fan-2">
  <outConfig>
    <equipmentFan id="2" model="" module="2" operability="operable" power="on"
      presence="equipped" serial="" thermal="not-supported" tray="1" vendor=""
      voltage="not-supported" dn="sys/rack-unit-1/fan-module-1-2/fan-2"/>
  </outConfig>
</configResolveDn>
```

SNMP 設定の詳細の取得

要求：

```
<configResolveClass cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388"
inHierarchical="false" classId="commSnmp"/>
```

応答：

```
<configResolveClass cookie="1313086522/c7c08988-aa3e-1a3e-8005-5e61c2e14388"
    response="yes" classId="commSnmp">
  <outConfig>
    <commSnmp dn="sys/svc-ext/snmp-svc" adminState="enabled" community="topSecret"
      descr="SNMP Service" name="snmp" port="161" proto="udp" sysContact="demo@demo.com"

      sysLocation="San Jose"/>
  </outConfig>
</configResolveClass>
```

サーバのローカル ハード ディスクの状態の取得

要求：

```
<configResolveDn cookie="1313146313/b38e04a0-aa4c-1a4c-8008-cdac38e14388"
inHierarchical="false" dn="sys/rack-unit-1/board/disk-4"/>
```

応答：

```
<configResolveDn cookie="1313146313/b38e04a0-aa4c-1a4c-8008-cdac38e14388" response="yes"

     dn="sys/rack-unit-1/board/disk-4">
  <outConfig>
    <storageLocalDiskSlotEp id="4" operability="operable" presence="equipped"
      dn="sys/rack-unit-1/board/disk-4"/>
  </outConfig>
</configResolveDn>
```

サーバ電源の再投入

要求：

```
<configConfMo cookie="1313084260/40ea8058-aa3e-1a3e-8004-5e61c2e14388" dn="sys/rack-unit-1"

     inHierarchical="false">
  <inConfig>
    <computeRackUnit adminPower="cycle-immediate" dn="sys/rack-unit-1">
    </computeRackUnit>
  </inConfig>
</configConfMo>
```

応答：

```
<configConfMo dn="sys/rack-unit-1" cookie="1313084260/40ea8058-aa3e-1a3e-8004-5e61c2e14388"

     response="yes">
  <outConfig>
    <computeRackUnit dn="sys/rack-unit-1" adminPower="policy" availableMemory="49152"
     model="R250-2480805W" memorySpeed="1067" name="UCS C250 M2" numOfAdaptors="1"
     numOfCores="8" numOfCoresEnabled="8" numOfCpus="2" numOfEthHostIfs="0"
     numOfFcHostIfs="0" numOfThreads="16" operPower="off"
     originalUuid="100DC440-0EBC-11DF-3B97-8843E1C2615E" presence="equipped" serverId="1"

     serial="PGS140601CS" totalMemory="49152" usrLbl="Cisco C210 Server"
     uuid="100DC440-0EBC-11DF-3B97-8843E1C2615E" vendor="Cisco Systems Inc"
     status="modified"/>
  </outConfig>
</configConfMo>
```

ブート順テーブルのセカンダリ ブート デバイスとしての EFI の設定

要求：

```
<configConfMo cookie="1313090863/ca79ef88-aa3f-1a3f-8006-5e61c2e14388"
     dn="sys/rack-unit-1/boot-policy/efi-read-only" inHierarchical="false">
  <inConfig>
    <lsbootEfi order="2" status="modified" dn="sys/rack-unit-1/boot-policy/efi-read-only"/>

  </inConfig>
</configConfMo>
```

応答：

```
<configConfMo dn="sys/rack-unit-1/boot-policy/efi-read-only"
      cookie="1313090863/ca79ef88-aa3f-1a3f-8006-5e61c2e14388" response="yes">
  <outConfig>
    <lsbootEfi dn="sys/rack-unit-1/boot-policy/efi-read-only" access="read-only" order="2"

      type="efi" status="modified"/>
  </outConfig>
</configConfMo>
```

フロッピーディスク ドライブのブート デバイスとしてのブート順リストからの削除

要求：

```
<configConfMo cookie="1313092854/412183f8-aa40-1a40-8007-5e61c2e14388"
     dn="sys/rack-unit-1/boot-policy/vm-read-write" inHierarchical="true">
  <inConfig>
    <lsbootVirtualMedia order="5" access="read-write" status="deleted"
        dn="sys/rack-unit-1/boot-policy/vm-read-write"/>
```

```
  </inConfig>
</configConfMo>
```

応答：

```
<configConfMo dn="sys/rack-unit-1/boot-policy/vm-read-write"
      cookie="1313092854/412183f8-aa40-1a40-8007-5e61c2e14388" response="yes">
   <outConfig>
   </outConfig>
</configConfMo>
```

inHierarchical オプションを使用した SNMP 設定の変更と設定済み SNMP トラップ レシーバの取得

要求：

```
<configConfMo cookie="1313090863/ca79ef88-aa3f-1a3f-8006-5e61c2e14388"
      inHierarchical="true" dn="sys/svc-ext/snmp-svc">
 <inConfig>
     <commSnmp dn="sys/svc-ext/snmp-svc" sysContact="TheAdmin@ITDept.com"
        community="demoPrivate" sysLocation="SanJoseCalifornia"/>
  </inConfig>
</configConfMo>
```

応答：

```
<configConfMo dn="sys/svc-ext/snmp-svc"
     cookie="1313090863/ca79ef88-aa3f-1a3f-8006-5e61c2e14388" response="yes">
  <outConfig>
    <commSnmp dn="sys/svc-ext/snmp-svc" adminState="enabled" community="demoPrivate"
         descr="SNMP Service" name="snmp" port="161" proto="udp"
        sysContact="TheAdmin@ITDept.com" sysLocation="SanJoseCalifornia" status="modified"

         childAction="deleteNonPresent">
      <commSnmpTrap adminState="disabled" community="demoPublic" hostname="11.22.33.44"
        id="1" notificationType="informs" version="v1" rn="snmp-trap-1" status="modified"

         childAction="deleteNonPresent"/>
      <commSnmpTrap adminState="disabled" community="demoPublic" hostname="50.60.70.80"
        id="2" notificationType="informs" version="v1" rn="snmp-trap-2" status="modified"

         childAction="deleteNonPresent"/>
     <commSnmpTrap adminState="disabled" community="demoPublic" hostname="0.0.0.0" id="3"

         notificationType="informs" version="v1" rn="snmp-trap-3" status="modified"
         childAction="deleteNonPresent"/>
     <commSnmpTrap adminState="enabled" community="demoPublic" hostname="138.148.198.218"

        id="4" notificationType="informs" version="v1" rn="snmp-trap-4" status="modified"

         childAction="deleteNonPresent"/>
    </commSnmp>
  </outConfig>
</configConfMo>
```

「Select Memory RAS」BIOS トークンの取得

要求：

```
<configResolveClass cookie="1313092854/412183f8-aa40-1a40-8007-5e61c2e14388"
inHierarchical="false" classId="biosVfSelectMemoryRASConfiguration"/>
```

応答：

```
<configResolveClass cookie="1313092854/412183f8-aa40-1a40-8007-5e61c2e14388"
    response="yes" classId="biosVfSelectMemoryRASConfiguration">
  <outConfig>
    <biosVfSelectMemoryRASConfiguration
       dn="sys/rack-unit-1/bios/bios-settings/SelectMemory-RAS-configuration"
       vpSelectMemoryRASConfiguration="maximum-performance" >
    </biosVfSelectMemoryRASConfiguration>
  </outConfig>
</configResolveClass>
```

ミラーリング モードの「Select Memory RAS」BIOS トークンの設定

要求：

```
<configConfMo cookie="1313092854/412183f8-aa40-1a40-8007-5e61c2e14388"
     inHierarchical="false"
     dn="sys/rack-unit-1/bios/bios-settings/SelectMemory-RAS-configuration">
  <inConfig>
    <biosVfSelectMemoryRASConfiguration
       dn="sys/rack-unit-1/bios/bios-settings/SelectMemory-RAS-configuration"
       vpSelectMemoryRASConfiguration="mirroring">
    </biosVfSelectMemoryRASConfiguration>
  </inConfig>
</configConfMo>
```

応答：

```
<configConfMo dn="sys/rack-unit-1/bios/bios-settings/SelectMemory-RAS-configuration"
     cookie="1313092854/412183f8-aa40-1a40-8007-5e61c2e14388" response="yes">
  <outConfig>
    <biosVfSelectMemoryRASConfiguration
       dn="sys/rack-unit-1/bios/bios-settings/SelectMemory-RAS-configuration"
       vpSelectMemoryRASConfiguration="mirroring" status="modified"/>
  </outConfig>
</configConfMo>
```

TFTP を使用した CIMC 設定のエクスポート

要求：

```
<configConfMo dn="sys/export-config"
      cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388" inHierarchical="false">
  <inConfig>
    <mgmtBackup dn="sys/export-config" adminState="enabled" hostname="198.29.210.14"
       remoteFile="/tftpserver/c250_config_export.cfg"/>
  </inConfig>
</configConfMo>
```

応答：

```
<configConfMo dn="sys/export-config"
     cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388" response="yes">
  <outConfig>
    <mgmtBackup dn="sys/export-config" adminState="disabled"
       fsmStageDescr="Completed successfully" fsmRmtInvErrCode=""
       fsmRmtInvErrDescr="NONE"
       fsmDescr="export-config" proto="tftp" hostname="" remoteFile=""
       status="modified"/>
  </outConfig>
</configConfMo>
```

上の要求がエクスポート操作を起動し、バックグラウンド タスクとして実行されます。 次の要求を送信することで、完了ステータスを定期的に照会できます。

ステータス要求：

```
<configResolveClass cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388"
inHierarchical="false" classId=" mgmtBackup"/>
```

完了後のステータス応答：

```
<configResolveClass cookie="1313122298/1c207238-aa47-1a47-8009-5e61c2e14388" response="yes"

      classId="mgmtBackup">
  <outConfig>
     <mgmtBackup dn="sys/export-config" adminState="disabled"
       fsmStageDescr="Completed successfully" fsmRmtInvErrCode="" fsmRmtInvErrDescr="NONE"

        fsmDescr="export-config" proto="tftp" hostname="" remoteFile=""/>
  </outConfig>
</configResolveClass>
```

エクスポートされたコンフィギュレーション ファイルは、次の例のようになります。

```
[root]# cat /tftpserver/c250_config_export.cfg
<root><cimc>
<version>1.4(0.22)</version>
<network>
<hostname>ucs-c250-M2</hostname>
<mode>dedicated</mode>
<redundancy>active-standby</redundancy>
<dns-use-dhcp>no</dns-use-dhcp>
<preferred-dns-server>0.0.0.0</preferred-dns-server>
<alternate-dns-server>0.0.0.0</alternate-dns-server>
<vlan-enabled>no</vlan-enabled>
.
.
.
```

TFTP を使用した CIMC 設定のインポート

要求：

```
<configConfMo dn="sys/import-config"
     cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388"
     inHierarchical="false">
  <inConfig>
     <mgmtImporter dn="sys/import-config" adminState="enabled"
         hostname="198.29.210.14" remoteFile="/tftpserver/c250_config_export.cfg"/>
  </inConfig>
</configConfMo>
```

応答：

```
<configConfMo dn="sys/import-config"
     cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388" response="yes">
  <outConfig>
      <mgmtImporter dn="sys/import-config" adminState="disabled"
          fsmStageDescr="Error" fsmRmtInvErrCode="" fsmRmtInvErrDescr="NONE"
          fsmDescr="import-config" proto="tftp" hostname="" remoteFile=""
          status="modified"/>
  </outConfig>
</configConfMo>
```

上の要求がインポート操作を起動し、バックグラウンド タスクとして実行されます。 次の要求を送信することで、完了ステータスを定期的に照会できます。

ステータス要求：

```
<configResolveClass cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388"
inHierarchical="false" classId="mgmtImporter"/>
```

完了前のステータス応答：

```
<configResolveClass cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388" response="yes"

     classId="mgmtImporter">
  <outConfig>
    <mgmtImporter dn="sys/import-config" adminState="enabled"
       fsmStageDescr="Applying configuration" fsmRmtInvErrCode=""
       fsmRmtInvErrDescr="NONE" fsmDescr="import-config" proto="tftp" hostname=""
       remoteFile=""/>
  </outConfig>
</configResolveClass>
```

繰り返しのステータス要求：

```
<configResolveClass cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388"
inHierarchical="false" classId="mgmtImporter"/>
```

完了後のステータス応答：

```
<configResolveClass cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388" response="yes"

      classId="mgmtImporter">
  <outConfig>
     <mgmtImporter dn="sys/import-config" adminState="disabled"
        fsmStageDescr="Completed successfully" fsmRmtInvErrCode=""
        fsmRmtInvErrDescr="NONE" fsmDescr="import-config" proto="tftp" hostname=""
        remoteFile=""/>
  </outConfig>
</configResolveClass>
```

TFTP を使用した CIMC テクニカル サポート データのエクスポート

要求：

```
<configConfMo dn="sys/rack-unit-1/tech-support"
      cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388" inHierarchical="false">
  <inConfig>
     <sysdebugTechSupportExport dn="sys/rack-unit-1/tech-support" adminState="enabled"
       remoteFile="/tftpserver/c250_techsupport_archive.tgz" hostname="198.29.210.14"/>
  </inConfig>
</configConfMo>
```

応答：

```
<configConfMo dn="sys/rack-unit-1/tech-support"
      cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388" response="yes">
  <outConfig>
     <sysdebugTechSupportExport dn="sys/rack-unit-1/tech-support" adminState="disabled"
        hostname="198.29.210.14" proto="tftp"
        remoteFile="/tftpserver/c250_techsupport_archive.tgz" fsmStageDescr="none"
        fsmProgr="0" fsmStatus="nop" status="modified"/>
  </outConfig>
```

```
</configConfMo>
```

上の要求がエクスポート操作を起動し、バックグラウンド タスクとして実行されます。 次の要求を送信することで、完了ステータスを定期的に照会できます。

ステータス要求：

```
<configResolveClass cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388"
inHierarchical="false" classId="sysdebugTechSupportExport"/>
```

完了前のステータス応答：

```
<configResolveClass cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388"
      response="yes" classId="sysdebugTechSupportExport">
  <outConfig>
     <sysdebugTechSupportExport dn="sys/rack-unit-1/tech-support" adminState="enabled"
        hostname="198.29.210.14" proto="tftp"
        remoteFile="/tftpserver/c250_techsupport_archive.tgz"
        fsmStageDescr="collecting" fsmProgr="0" fsmStatus="exporting"/>
  </outConfig>
</configResolveClass>
```

繰り返しのステータス要求：

```
<configResolveClass cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388"
inHierarchical="false" classId="sysdebugTechSupportExport"/>
```

完了後のステータス応答：

```
<configResolveClass cookie="1313118253/2b07f100-aa46-1a46-8008-5e61c2e14388"
      response="yes" classId="sysdebugTechSupportExport">
  <outConfig>
     <sysdebugTechSupportExport dn="sys/rack-unit-1/tech-support" adminState="disabled"
        hostname="198.29.210.14" proto="tftp"
        remoteFile="/tftpserver/c250_techsupport_archive.tgz"
        fsmStageDescr="completed" fsmProgr="100" fsmStatus="success"/>
  </outConfig>
</configResolveClass>
```

エクスポートされたテクニカル サポート ファイルは、次の例のようになります。

```
[root]# tar tvfz /tftpserver/c250_techsupport_archive.tgz | more
drwxr-xr-x root/root         0 2011-08-11 13:01:10 obfl/
-rw-r--r-- root/root     76910 2011-08-11 13:00:56 obfl/obfl-log.1
-rw-r--r-- root/root     76835 1970-01-01 09:38:26 obfl/obfl-log.2
-rw-r--r-- root/root     76881 2011-08-08 21:20:55 obfl/obfl-log.3
-rw-r--r-- root/root     76916 1969-12-31 16:07:28 obfl/obfl-log.4
-rw-r--r-- root/root     76846 2011-08-03 21:38:49 obfl/obfl-log.5
-rw-r--r-- root/root     14598 2011-08-11 20:49:57 obfl/obfl-log
.
.
.
```

付 録 B

configConfMo メソッドの使用に関する注意事項

この付録は、次の項で構成されています。

configConfMo メソッドを使用した識別名の定義

configConfMo メソッドを使用して、管理対象オブジェクト（MO）の 1 つ以上のプロパティを設定します。 設定する MO は識別名（DN）によって一意に識別されます。 この章は、configConfMo メソッドを使用して DN を提供する 2 つの方法を示します。

管理対象オブジェクト レベル

管理対象オブジェクト レベルで DN を提供できます。 次の例では、DN「sys/rack-unit-1/locator-led」が MO「equipmentLocatorLed」内に定義されます。

```
<configConfMo
    cookie="<real_cookie>">
    <inConfig>
        <equipmentLocatorLed
            adminState='on'
            dn='sys/rack-unit-1/locator-led'>                    <== MO level
        </equipmentLocatorLed>
    </inConfig>
</configConfMo>
```

メソッドおよび管理対象オブジェクト レベル

メソッドおよび管理対象オブジェクト レベルで DN を提供できます。 次の例では、DN「sys/rack-unit-1/locator-led」が、MO「equipmentLocatorLed」内に configConfMo メソッド レベルで定義されます。

```
<configConfMo
    cookie="<real_cookie>"
    dn='sys/rack-unit-1/locator-led'>                          <== Method level
    <inConfig>
        <equipmentLocatorLed
            adminState='on'
            dn='sys/rack-unit-1/locator-led'>                  <== MO Level
        </equipmentLocatorLed>
    </inConfig>
</configConfMo>
```

（注） メソッド レベルでの DN の指定はオプションで、Cisco UCS Manager XML API の実装と同一になるように Cisco CIMC XML API の実装でサポートされます。

オプションの inHierarchical 属性の使用

configConfMo 要求が CIMC に送信されると、応答には設定される MO の直接のプロパティだけが含まれます。

オプションの inHierarchical 属性が configConfMo 要求に含まれている場合、応答は true に設定された inHierarchical 属性を持つ configResolveDn 要求の応答と同様になります。 応答には、設定される MO のプロパティとともに、その子 MO のプロパティがすべて含まれます。

要求：

```
<configConfMo
    cookie="<real_cookie>"
    inHierarchical="true"
    dn='sys/rack-unit-1/locator-led'>
    <inConfig>
        <equipmentLocatorLed
            adminState='on'
            dn='sys/rack-unit-1/locator-led'>
        </equipmentLocatorLed>
    </inConfig>
</configConfMo>
```

応答：

```
<configConfMo
    dn="sys/rack-unit-1/locator-led"
    cookie="<real_cookie>"
    response="yes">
    <outConfig>
        <equipmentLocatorLed
            dn="sys/rack-unit-1/locator-led"
            adminState="inactive"
            color="unknown"
            id="1"
            name=""
            operState="on">
```

```
        </equipmentLocatorLed>
    </outConfig>
</configConfMo>
```

1 つの管理対象オブジェクトの設定

リリース 1.4(1x) では、 Cisco CIMC XML API の実装は単一の管理対象オブジェクト（MO）上で動作する configConfMo メソッドだけを受け入れます。 CIMC 管理情報モデルの包含関係で定義されている場合でも、複数の MO を含む configConfMo メソッドを指定することは無効です。

次に、単一の MO「lsbootLan」を設定する有効な configConfMo メソッドの例を示します。この例では、ホストは最初のブート オプションとして PXE 起動を使用するように設定されています。

```
<configConfMo
    cookie="<real_cookie>">
    <inConfig>
        <lsbootLan                                          <== Single MO
            order="1"
            status="modified"
            dn="sys/rack-unit-1/boot-policy/lan" >
        </lsbootLan>
    </inConfig>
</configConfMo>
```

次の例の configConfMo メソッドは、親 MO と子 MO が同時に指定されているため無効です。「equipmentLocatorLed」および「solIf」の MO は、管理情報ツリーの「computeRackUnit」MO の子オブジェクトです。 Cisco CIMC XML API の実装は configConfMo メソッドがサブツリーの設定を行うことを許可しません。

要求：

```
<configConfMo
    cookie="1313084260/40ea8058-aa3e-1a3e-8004-5e61c2e14388"
    dn="sys/rack-unit-1" inHierarchical="false">
     <inConfig>
         <computeRackUnit                                    <== Parent MO
             adminPower="cycle-immediate"
             usrLbl="Cisco C210 Server"
             dn="sys/rack-unit-1">
                  <equipmentLocatorLed                       <== Child MO
                      adminState="on"
                      dn="sys/rack-unit-1/locator-led"/>
                  <solIf                                     <== Child MO
                      dn="sys/rack-unit-1/solif"
                      adminState="enable"
                      speed="9600"/>
         </computeRackUnit>
      </inConfig>
</configConfMo>
```

応答：

```
XML PARSING ERROR: Element 'equipmentLocatorLed': This element is not expected.
```

（注） このメソッドは、Cisco UCS Manager XML API の実装では有効ですが、 Cisco CIMC XML API の実装ではサポートされていません。

付 録 C

CIMC Visore ユーティリティ

Visoreは、HTMLブラウザを使用して管理対象オブジェクト（MO）を簡単に参照できる、CIMCに構築されたユーティリティです。 Visore ユーティリティは、 Cisco CIMC XML API クエリーメソッドを使用して CIMC のアクティブな MO を参照します。 Visore ユーティリティは、設定を行うためには使用できません。

Visore へのアクセス方法

Visore にアクセスするには、ブラウザを開き、次のいずれかの URL を入力します。

- http://<CIMC IP Address>/visore.html
- https://<CIMC IP Address>/visore.html

プロンプトが表示されたら、CIMC CLI または GUI ユーザ インターフェイスへのログインと同じクレデンシャルを使用してログインします。

（注） CIMC リリース 1.4(1x) では、Visore アクセスに Firefox および Chrome のブラウザだけがサポートされます。

クラス照会での Visore の使用

特定のクラスを照会するには、クラス名を [Class or DN] フィールドに入力し、[Run Query] をクリックします。 Visore は configResolveClass メソッドを CIMC に送信し、要求された MO が表形式で表示されます。

表示された MO の親および子クラスを取得するには、[<] および [>] ボタンを使用します。 たとえば、[>] をクリックすると、MO の子を照会するために configResolveChildren メソッドが CIMC に送信されます。 [<] をクリックすると、MO の親を照会するために configResolveParent メソッドが CIMC に送信されます。

識別名（DN）を照会するための Visore の使用

特定の DN を照会するには、DN を [Class or DN] フィールドに入力し、[Run Query] をクリックします。 Visore は configResolveDn メソッドを CIMC に送信します。

索 引

し

せ

そ

に

れ