CHAPTER 4

Cisco ANA Network Service Activation スクリプトのカスタマイズ

次のトピックでは、Cisco ANA NSA スクリプトの変更に関する一般的な情報について説明します。一般に、詳細な知識と経験が必要となるため、スクリプトを変更することはお勧めできません。トピックは次のとおりです。

スクリプトのカスタマイズを始める前に

スクリプトの変更を始める前に、『*Cisco Active Network Abstraction 3.7 Customization Guide*』の「Managing and Deploying Configuration Changes Using Command Builder」の章を読み、参照できるように用意してください。すべての適切な権限があることを確認し、さまざまなジョブ ポリシー、展開およびスケジューリング手順を必ず理解しておいてください。

Cisco ANA NSA のアクティベーション ウィザードおよびワークフローは、パッケージのウィザードおよびワークフロー用に記述され、テストされたものです。アクティベーション スクリプトをカスタマイズした場合、Cisco ANA NSA のワークフローおよびウィザードのカスタマイズが必要となることがあります。

（注） この項の情報は、『*Cisco Active Network Abstraction 3.7 Customization Guide*』に記載された情報に代わるものではありません。

Cisco ANA マクロ言語および BeanShell

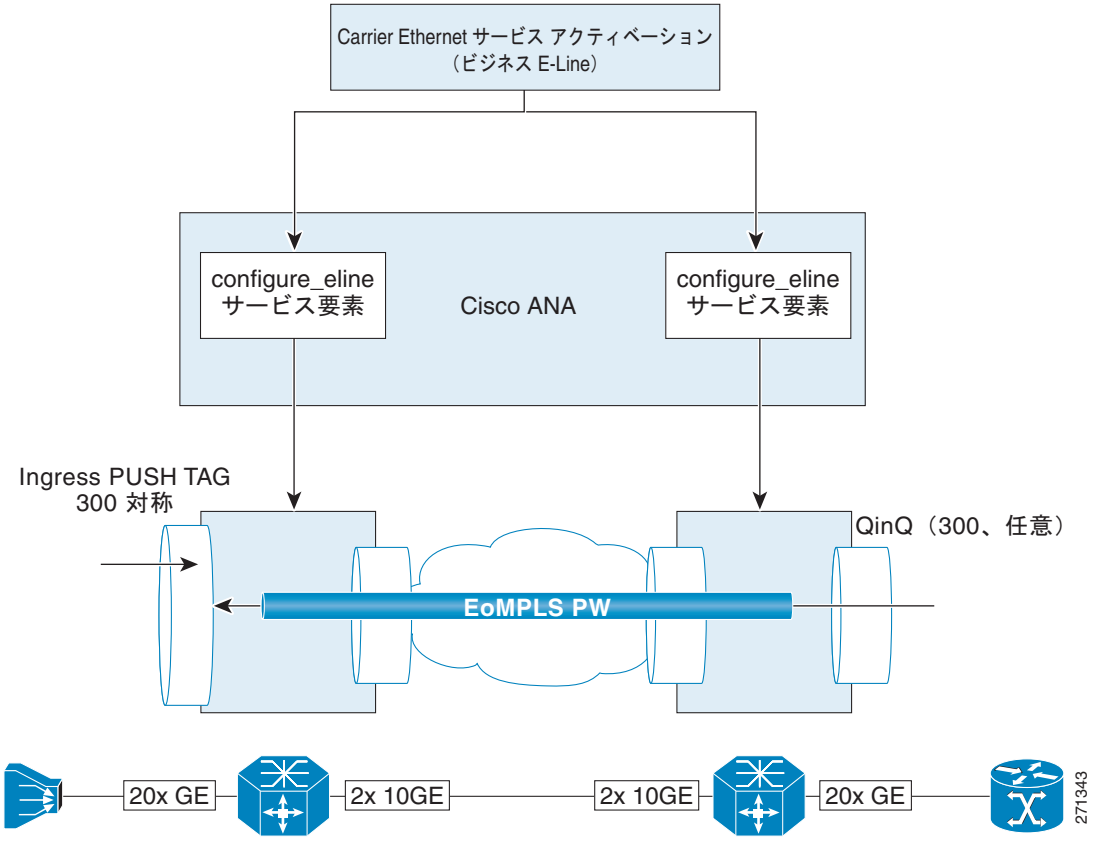
Cisco ANA NSA スクリプトは、Cisco ANA マクロ言語または BeanShell スクリプト言語を使用して変更できます。Cisco ANA マクロ言語については、『*Cisco Active Network Abstraction 3.7 Customization Guide*』の「Cisco ANA Macro Language」を参照してください。BeanShell の使用方法に関する最も完全な資料は、http://www.beanshell.org/ からオンラインで入手できます。この項では、デバイスとのやり取りに使用される Cisco ANA NSA コマンドを作成するために BeanShell を使用する場合のガイドラインだけを示します。

- Cisco ANA マクロ言語とは異なり、BeanShell ユーザ引数のインベントリ プロパティは $...$ 記号内に*埋め込まないでください*。
- 次の例は、デバイスとのやり取りに使用できる Telnet 環境の定義済みオブジェクトを示します。
  - telnetInterface.config ("prompt","telnet command") : "prompt" はコマンド完了後の所定のプロンプト、"telnet command" は実行される実際のコマンドです。
  - telnetInterface.setStatus (*1 または 2*) : 1 は成功、2 は失敗です。設定コマンドが成功または失敗したことをワークフロー マネージャに知らせる場合などに使用されます。

アクティベーション スクリプトの例

この項では、E-Line ポイントツーポイント サービスのアクティベーションを例に挙げ、E-Line ポイントツーポイント スクリプトを使用してサービスをプロビジョニングする方法を示します。図 4-1 に、一般的なフローを示します。

図 4-1　キャリア イーサネット サービスのアクティベーション

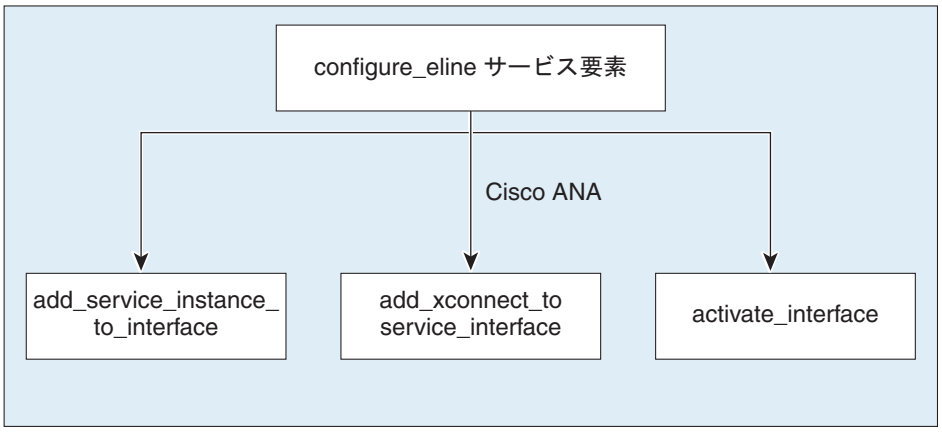


Cisco ANA NSA アクティベーション スクリプトは、モジュラー型で再利用できるように設計されています。使用可能なユーティリティ スクリプトを再利用して、新しいサービス要素スクリプトをカスタマイズまたは作成できます。スクリプトは次のものとして実装されます。

- サービス要素スクリプト：1 つのネットワーク要素のコンテキスト内でサービス ロジックを実装します。
- ユーティリティ スクリプト：デバイス上の 1 つのアトミック アクションを実装し、多くのサービス要素スクリプトから再利用できる、より粒度の大きいスクリプトです。ユーティリティ スクリプトはアクションのロールバック ロジックも実装しているので、アクションはアトミックで完全に独立したものになります。ユーティリティ スクリプトが成功したら、そのタスクの Command-Line Interface（CLI; コマンドライン インターフェイス）がデバイス上に設定されます。同様に、ユーティリティ スクリプトが失敗した場合は、ユーティリティ スクリプトがそのタスクのロールバックを実行し、デバイス上には何も設定されません。

図 4-2 に、E-Line サービス要素スクリプトと使用されるユーティリティ スクリプトを示します。

図 4-2　Configure E-Line サービス要素スクリプト



確認チェック

Cisco ANA NSA スクリプトでは、パラメータ値に一連の確認手順を実行することによってエラーを解決し、デバイス上の設定エラーの可能性を低減します。次の確認チェックが含まれます。

- すべての必須パラメータが入力されていることの確認。
- パラメータの値がデバイスで求められる範囲に入っていることの確認（VLAN ID、ブリッジ ドメイン ID、サービス インスタンス ID など）。
- 値のシンタックスがデバイスのシンタックスを満たしていることの確認。

これらの確認が完了したら、別の一連のチェックがデバイスと照合して実行されます。実行される可能性のあるチェックの一部を次に示します。

- デバイス上で show コマンドを使用して、サービスに必要なインターフェイスが存在することを確認。
- サービス インスタンス ID が設定済みであることの確認。
- VFI が存在することの確認。
- サービス インスタンスに付加される QoS ポリシーがデバイスに定義されていることの確認。

すべてのチェックが完了したら、スクリプトは、CLI をデバイスにダウンロードするユーティリティ スクリプトの使用を開始します。CLI のダウンロード中にルータがエラー メッセージを返した場合は、エラー処理およびロールバックが実行されます。ユーティリティ スクリプトは、エラーをサービス要素スクリプトに返し、サービス要素レベルのロールバックを実行します。これには、ダウンロードされ適用されたすべての設定の「削除」ユーティリティ スクリプトなどが含まれます。Cisco ANA NSA スクリプトは、CLI がデバイス上で正しく設定されるようにします。エラーが発生した場合は、ロールバックが実行され、適切なエラー メッセージが生成されます。

サービス アクティベーション用スクリプトの記述

表 4-1 に、サービス アクティベーション スクリプトを記述するための一般的な手順を示します。説明には、「Configure E-Line シナリオのスクリプト ファイル注釈」（P.4-8）に記載された E-Line ポイントツーポイント スクリプト例への参照が含まれています。

表 4-1　スクリプトのデザインおよび実装の概要

手順	タスク	説明
ステップ 1	入力パラメータを必要に応じてデータ型に変換する。	Cisco ANA は、文字列、整数、ロング、コンボなどを含む多くのデータ型をサポートしています。初期化し、スクリプトで必要となる型に変換します。 「Configure E-Line シナリオのスクリプト ファイル注釈」（P.4-8）（項目 5）を参照してください。
ステップ 2	すべての必須パラメータが入力されていることをチェックする。	スクリプトの実行に必要なすべてのパラメータがユーザから与えられていることを確認します。与えられていない場合は、適切なエラー メッセージを提供します。 「Configure E-Line シナリオのスクリプト ファイル注釈」（P.4-8）（項目 6）を参照してください。
ステップ 3	パラメータ値の範囲およびシンタックスをチェックする。	この段階で、スクリプト内で第 1 レベルのチェックを実行できます。これには、VLAN ID が適切な範囲内にあることの確認、入力されたパラメータのシンタックスがデバイスの許容範囲内にあることの確認など、簡単な確認が含まれます。 「Configure E-Line シナリオのスクリプト ファイル注釈」（P.4-8）（項目 7）を参照してください。
ステップ 4	必要に応じて、オブジェクトの存在をデバイスと照合してチェックする。	入力されたパラメータ値をデバイスと照合してチェックします。チェックには、スクリプトから求められるオブジェクトの存在を確認するために、デバイス表示要求を含めることができます。オブジェクトには、インターフェイス、サービス インスタンス、xConnect、QoS ポリシーなどが含まれます。 「Configure E-Line シナリオのスクリプト ファイル注釈」（P.4-8）（項目 7）を参照してください。
ステップ 5	ユーティリティ スクリプトを起動し、エラーを処理する。	ユーティリティ スクリプトを呼び出して、該当する CLI をデバイスにダウンロードします。ユーティリティ スクリプトはロールバックを実装しているので、CLI のダウンロード中にエラーが発生した場合は、ダウンロードされたすべての CLI が削除されます。 「Configure E-Line シナリオのスクリプト ファイル注釈」（P.4-8）（項目 7）を参照してください。
ステップ 6	エラーが発生した場合は、ロールバックする。	サービス要素スクリプトを使用して、エラー条件を処理し、対応する削除ユーティリティ スクリプトを起動してエラー発生前に実行されたアクションを元に戻します。これにより、エラー条件下で、デバイスにダウンロードされたすべての CLI が確実に削除されます。 「Configure E-Line シナリオのスクリプト ファイル注釈」（P.4-8）（項目 2 および 7）を参照してください。

スクリプトのカスタマイズ フロー

次の手順は、スクリプトをカスタマイズするための一般的なフローを示します。

1. リソース要素（ユーティリティ）BeanShell スクリプトを作成します（必要な場合）。
2. サービス要素 BeanShell スクリプトを作成します。
3. Cisco ANA NetworkVision Command Builder を使用して、GUI および BQL 用のインターフェイスを定義する BeanShell スクリプトを作成します。
   a. コマンドを実行してデバッグおよび確認を行います。
   b. 次の手順で取り出される情報が含まれるファイルをエクスポートします。
4. 次の XML ファイルおよび jar ファイルを作成します。
   a. site-ps.xml :「リポジトリ」ファイルへのデバイス別ポインタが含まれます（必要に応じて変更）。
   b. cisco-<technology>-<device type>-scripts-repository.xml（新しいスクリプトを追加）：

      サービス スクリプトのメタデータ定義が含まれます。

      手順 3 で作成した情報を Cisco ANA NSA スクリプト形式に変換します。
   c. cisco-<technology>-DSP.xml（新しいスクリプトを追加）：

      サービス スクリプトの表示情報が含まれます。

      手順 c）で作成した情報を Cisco ANA NSA DSP 形式に変換します。
   d. クライアント jar ファイルは DSP ファイルで構成されます。
5. インストール：
   a. スクリプトを Cisco ANA ゲートウェイの scripts ディレクトリにコピーします。
   b. XML ファイルを Cisco ANA ゲートウェイの registry ディレクトリにコピーします。
   c. クライアント jar を webstart/jar ディレクトリにコピーし、jnlp ファイルをアップデートします。
   d. サーバ/クライアントを再起動します。
6. 確認：

   新しいサービス スクリプトが Command Builder で使用できる必要があります。

   新しいコマンドで Cisco ANA NetworkVision Command Builder を実行して確認します。

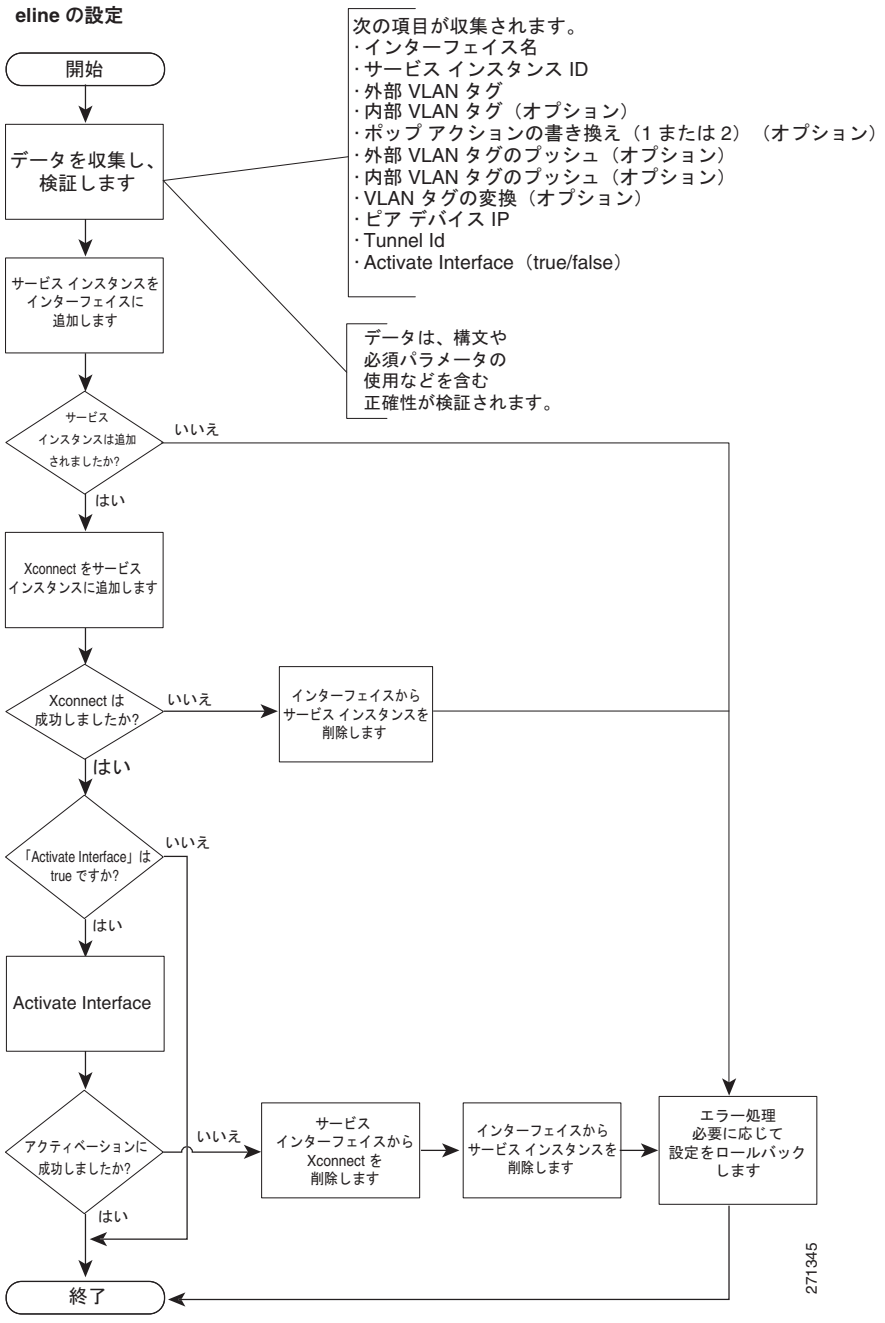
E-Line フロー チャート

E-Line アクティベーションには、次のコンポーネント スクリプトが含まれています。

- configure_eline、remove_eline
- add_service_instance_to_interface、remove_service_instance_from_interface
- add_xconnect_to_service_instance、remove_xconnect_from_service_instance

図 4-3 に、E-Line 設定スクリプト フローを示します。

図 4-3　Configure E-Line



E-Line CLI

E-Line スクリプトが成功した場合に、デバイス上で実行される CLI です。

```
conf t
int <Interface Name>
service instance <Service Instance Id> ethernet
encapsulation dot1q <Outer VLAN Tag> [ second-dot1q <Inner VLAN Tag >]
[rewrite ingress tag pop <Rewrite Pop Action>  symmetric]
[rewrite ingress tag push dot1q <Push Outer VLAN Tag>  symmetric]
[rewrite ingress tag push dot1q <Push Outer VLAN Tag>  second-dot1q
<Push Inner VLAN Tag> symmetric]
[rewrite ingress tag translate 1-to-1 dot1q <Translate VLAN Tag >  symmetric]
xconnect <Peer Device IP> <Tunnel Id> encapsulation mpls
exit
exit
exit
```

Configure E-Line シナリオのスクリプト ファイル注釈

次に、configure_eline サービス要素スクリプトの BeanShell コードの一部を注釈付きで示します。省略記号（...）は、スクリプト デザインの原則を示す上で必要ではないために省略されたコードを表します。configure_eline の全テキストとコンポーネント ユーティリティ スクリプトについては、「Configure E-Line サービスのアクティベーション スクリプト」（P.B-1）を参照してください。

/******* Script to configure e-line *******/

1. configure_eline ユーティリティ スクリプトには、次のものが含まれます。

```
String sep = File.separator;
...
source("." + sep + "scripts" + sep + "configuration" + sep + "cisco" + sep +
"add_service_instance_to_interface.bsh");
source("." + sep + "scripts" + sep + "configuration" + sep + "cisco" + sep +
"add_xconnect_to_serviceInstance.bsh");
...
```

2. 次の関数は、configure_eline サービス要素スクリプトでエラーが発生し、ロールバックする必要が生じた場合に、このスクリプトから起動されます。failureLevel パラメータを使用して、エラーの場所情報が渡されます。その後、エラーの原因となった動作を元に戻すために、適切なユーティリティ スクリプトが起動されます。

```
...
void rollback(String  interfaceName,Integer intServiceInstanceId, String failureLevel)
{
            // Function entry message
        telnetInterface.println ("Inside rollback_configure_eline...");
...
        if (failureLevel =="xconnect")
        {
              // invoking utility script to remove service instance from
                // interface
             remove_service_instance_from_interface(interfaceName,strServiceInstanceId);
        }
        else if(failureLevel == "activate-interface")
        ...
```

3. この関数の適切なステータスを次のように設定します。

```
        telnetInterface.setStatus(CARRIERETHERNET_PROVISIONING_STATUS_FAILURE );
// Function exit message
        telnetInterface.println ("rollback_configure_eline done");
        sendResult(false, "The script --configure_eline --failed");
```

4. この関数は、**flex uni eline** コマンド セットをデバイスに設定します。エラーが発生した場合は、デバイスに展開された CLI がすぐに削除されます。

```
boolean configure_eline(String interfaceName,Integer intServiceInstanceId,String
outerVlanTag,String  innerVlanTag,String rewritePopAction,Integer
intPushOuterVlanTag,Integer intPushInnerVlanTag,Integer intTranslateVlanTag,String
peerDeviceIp,Integer intTunnelId,String activateInterface)
{
    telnetInterface.println("Inside Configure E-Line...");
```

5. 入力パラメータを必要に応じてデータ型に変換します。

```
...
String strPushInnerVlanTag = "";
             if (intPushInnerVlanTag!= null)
                     strPushInnerVlanTag = String.valueOf(intPushInnerVlanTag);
...
    // handling 'activateInterface' combo input
    String strActivateInterface="";
    if(activateInterface.equals("true"))
     {
        strActivateInterface="true";
     }
    else if(activateInterface.equals("false"))
     {
        strActivateInterface="false";
     }
```

6. 必須パラメータをチェックします。

```
            // checking if mandatory input interface name is left blank
    if(!checkBlank(interfaceName,"Interface Name"))
      {
        telnetInterface.setStatus(CARRIERETHERNET_PROVISIONING_STATUS_FAILURE );
        sendResult(false, "The script --configure_eline -- failed");
        return false;
      }
        ...
```

7. ユーティリティ スクリプトを使用して、サービス インスタンスの追加、サービス インスタンスへの xconnect の追加、または必要な場合、インターフェイスのアクティベーションを行います。これらのユーティリティ スクリプトは、デバイスへの設定前に入力パラメータを確認します。エラーが発生した場合、ユーティリティ スクリプトはロールバックを実行し、サービス要素スクリプトが必要に応じてエラーを処理できるようにエラーを返します。

```
        ...
    if(!add_service_instance_to_interface(interfaceName,strServiceInstanceId
,outerVlanTag,innerVlanTag,strRewritePopAction,strPushOuterVlanTag,strPushInnerVlanTag,str
TranslateVlanTag))
    {
        telnetInterface.setStatus(CARRIERETHERNET_PROVISIONING_STATUS_FAILURE );
        sendResult(false, "The script --configure_eline-- failed");
        return false;
    }
    // invoke the utility script to add xconnect on the service instance. This script
verifies the input parameters locally as well
      // as against the router before configuring xconnect on service instance. If there
is an error, returns false.
      // If there is an error configuring xconnect, rollback is called to remove the
configured service instance.

if(!add_xconnect_to_serviceInstance(interfaceName,strServiceInstanceId,outerVlanTag,innerV
lanTag,peerDeviceIp,strTunnelId))
    {
        rollback(interfaceName,intServiceInstanceId,"xconnect");
```

```
            telnetInterface.setStatus(CARRIERETHERNET_PROVISIONING_STATUS_FAILURE );
            return false;
        }
        // Activate the interface if needed.
        if(strActivateInterface.equals("true"))
        {
            if(!activate_interface(interfaceName))
            {
              telnetInterface.setStatus(CARRIERETHERNET_PROVISIONING_STATUS_FAILURE );
              rollback(interfaceName,intServiceInstanceId,"activate-interface");
              return false;
            }
        }
```

8. 成功メッセージを出力して終了します。

```
    sendResult(true, "The script configure_eline was executed successfully");
    telnetInterface.setStatus(CARRIERETHERNET_PROVISIONING_STATUS_SUCCESS );
    return true;
}
...
```

9. configure_eline を起動します。

```
boolean result = configure_eline(interfaceName,intServiceInstanceId
,outerVlanTag,innerVlanTag,rewritePopAction,intPushOuterVlanTag,intPushInnerVlanTag,intTra
nslateVlanTag,peerDeviceIp,intTunnelId,activateInterface);
```

ヒント　E-Line アクティベーション スクリプトの全テキストと、このシナリオで使用されるユーティリティ スクリプトについては、付録 B「Configure E-Line サービスのアクティベーション スクリプト」のサンプル テキストを参照してください。

レジストリ XML ファイル

Cisco ANA NSA は、基本の Cisco ANA と連携して動作するように設計されています。これを実現するために、Cisco ANA NSA には次のものが含まれています。

- アクティベーション ロジックを実装するスクリプト。
- スクリプトにメタデータを提供し、それらを Cisco ANA ユニットの適切な VNE に登録するレジストリ XML ファイル。

表 4-2 に、Cisco ANA NSA のレジストリ XML ファイルを示します。

表 4-2　Cisco ANA NSA の XML ファイル

ファイル	説明
site-ps.xml	スクリプト ファイルへのデバイス別ポインタ。
cisco-<technology>-<device class>-scripts-**repository.xml**	各スクリプトのメタデータ定義。
cisco-<technology>-**DSP.xml**	各スクリプトの固有情報。

サイト XML ファイル

site-ps.xml ファイルは、Cisco ANA NSA によって実行されるカスタマイズを取り込みます。アクティベーション モジュール パッケージでは、このファイルを編集して、異なるデバイスで新しいアクティベーション スクリプトをサポートします。次に、Cisco 7600 ルータ用の登録が site-ps.xml ファイルによって実行される例を示します。「76xx」のエントリにより、Cisco ANA サーバが、MPLS 方式をサポートするすべての Cisco 7600 VNE 用のスクリプト ファイルを指すようにします。

...

```
<key name="site-ps">
  <entry name="default">.sitedefault</entry>
- <key name="ciscorouter2">
  - <key name="76xx">
    - <key name="ipcore">
        - <key name="software versions">
          - <key name="default version">
            - <key name="imo">
              - <key name="scripts">
               - <key name="com.sheer.imo.IManagedElement">
                 - <key name="add_allowed_vlan_to_interface">
                     <entry
                     name="default">cisco-carrierethernet-76xx-scripts-repository/add_allo
                     wed_vlan_to_interface</entry>
                   </key>
                 - <key name="add_l2pt_to_switchport_interface">
                     <entry
                      name="default">cisco-carrierethernet-76xx-scripts-repository/add_l2p
                      t_to_switchport_interface</entry>
                      </key>
                  - <key name="add_mac_acl_to_service_instance">
                     <entry
                     name="default">cisco-carrierethernet-76xx-scripts-repository/add_mac_
                     acl_to_service_instance</entry>
                   </key>
                 - <key name="add_mac_acl_to_switchport">
                    <entry
                    name="default">cisco-carrierethernet-76xx-scripts-repository/add_mac_a
                    cl_to_switchport</entry>
                  </key>
                 - <key name="add_mac_security_to_service_instance">
                    <entry
                    name="default">cisco-carrierethernet-76xx-scripts-repository/add_mac_s
                    ecurity_to_service_instance</entry>
                  </key>
…
```

リポジトリ XML ファイル

リポジトリ XML ファイルには、スクリプト定義が含まれます。これは Command Builder を使用してスクリプトを作成したときに Cisco ANA によって VNE AVM に記述される XML 設定と同様です。この定義では、次の例に示すようにスクリプトについて記述し、スクリプトで求められるパラメータをリストし、BeanShell ファイルの名前と相対パスを示します。

```
- <key name="cisco-carrierethernet-34xx-scripts-repository">
  - <key name="add_allowed_vlan_to_interface">
      <entry name="default">voids</entry>
      <entry name="metadata-id">76xx</entry>
      <entry name="error-condition" />
```

```
    <entry name="interface">telnet</entry>
    <entry name="script-name">add_allowed_vlan_to_interface</entry>
    <entry name="language">beanshell</entry>
    <entry name="enable">true</entry>
- <key name="parameters">
  - <key name="0">
      <entry name="type">java.lang.String</entry>
      <entry name="name">interfaceName</entry>
  </key>
  - <key name="1">
      <entry name="type">java.lang.String</entry>
      <entry name="name">outerVlanTag</entry>
  </key>
</key>
<key name="enums" />
- <key name="roles">
    <entry name="Administrator" />
    <entry name="OperatorPlus" />
    <entry name="Operator" />
    <entry name="Viewer" />
    <entry name="Configurator" />
</key>
<key name="rollback" />
- <key name="script">
    <entry
  name="file-name">configuration/cisco/ce/add_allowed_vlan_to_interface.bsh</entry>
  </key>
</key>
```

DSP-Scripts XML ファイル

DSP-Scripts ファイルには、Cisco ANA NSA クライアントがスクリプトの名前および設定を正しく表示するために必要となるすべての表示パラメータが含まれます。この内容は、Command Builder を使用してスクリプトを作成したときに Cisco ANA によって site.xml ファイルに記述される XML と同様です。

```
- <key name="cisco-carrierethernet-DSP">
  - <key name="add_allowed_vlan_to_interface">
   - <key name="metadata">
    - <key name="76xx">
        <entry name="short-description">add_allowed_vlan_to_interface</entry>
        <entry name="display-name">add_allowed_vlan_to_interface</entry>
        <entry name="menu-path">Commands</entry>
        <entry      name="class-name">com.sheer.client.common.components.scripts.
        GenericScriptsController</entry>
        <entry name="multiple-selection">false</entry>
        <entry name="icon">toolbar/Run_Command</entry>
        <entry name="visible">true</entry>
        <entry name="location">100</entry>
      - <key name="parameters">
        - <key name="interfaceName">
            <entry name="visible">true</entry>
            <entry name="default-value" />
            <entry name="width">15</entry>
            <entry name="required">true</entry>
            <entry name="location">0</entry>
            <entry name="display-name">Interface Name*</entry>
            <entry name="enabled">true</entry>
```

```
        </key>
      - <key name="outerVlanTag">
          <entry name="visible">true</entry>
          <entry name="default-value" />
          <entry name="required">true</entry>
          <entry name="width">15</entry>
          <entry name="location">1</entry>
          <entry name="display-name">Outer VLAN Tag*</entry>
          <entry name="enabled">true</entry>
        </key>
       </key>
      </key>
     </key>
    </key>
…
```