HP Service Activator

Putting Service Activator to Work: A Sample Service Scenario for VPLS

Edition: V62-1A

for Microsoft Windows® Server 2008 R2, HP-UX 11i v3, and Red Hat Enterprise Linux 6.4 operating systems

Manufacturing Part Number: None

October 14, 2013

Legal Notices

Document id: HPSA p158-pd001008

Contents

Install Location Descriptors

The following names are used to define install locations throughout this guide.

Descriptor	**What the Descriptor Represents**
$ACTIVATOR_OPT	The base install location of Service Activator. The UNIX® location is `/opt/OV/ServiceActivator` The Windows® location is `<`*`install drive`*`>:\HP\OpenView\ServiceActivator`
$ACTIVATOR_ETC	The install location of specific Service Activator files. The UNIX location is `/etc/opt/OV/ServiceActivator` The Windows location is `<`*`install drive`*`>:\HP\OpenView\ServiceActivator\etc`
$ACTIVATOR_VAR	The install location of specific Service Activator files. The UNIX location is `/var/opt/OV/ServiceActivator` The Windows location is `<`*`install drive`*`>:\HP\OpenView\ServiceActivator\var`
$ACTIVATOR_BIN	The install location of specific Service Activator files. The UNIX location is `/opt/OV/ServiceActivator/bin` The Windows location is `<`*`install drive`*`>:\HP\OpenView\ServiceActivator\bin`

$JBOSS_HOME	The install location for JBoss.
	The UNIX location is
	`/opt/HP/jboss`
	The Windows location is
	`<`*`install drive`*`>:\HP\jboss`

$JBOSS_DEPLOY	The install location of the Service Activator J2EE components.
	The UNIX location is
	`/opt/HP/jboss/standalone/deployments`
	The Windows location is
	`<`*`install drive`*`>:\HP\jboss\standalone\deployments`

$JBOSS_EAR_LIB	Location for libraries (java *.jar files) to be executed by the HPSA engine (workflow manager and resmgr) `$JBOSS_DEPLOY/hpsa.ear/lib`
$ACTIVATOR_DB_USER	The database user name you define. Suggestion: `activator`
$ACTIVATOR_SSH_USER	The Secure Shell user name you define. Suggestion: `actusr`

Conventions

The following typographical conventions are used in this guide.

Font	What the Font Represents	Example
Italic	Book or manual titles, and manpage names	Refer to the *HP Service Activator—Workflows and the Workflow Manager* and the *Javadocs* for more information
	Provides emphasis	You *must* follow these steps.
	Specifies a variable that you must supply when entering a command	Run the command: `InventoryBuilder` *<sourceFiles>*
	Parameters to a method	The assigned_criteria parameter returns an ACSE response.
`Computer`	Text and items on the computer screen	The system replies: `Press Enter`
	Command names	Use the `InventoryBuilder` command
	Method names	The `get_all_replies()` method does the following…
	File and directory names	Edit the file `$ACTIVATOR_ETC/config/mwfm.xml`
	Window/dialog box names	In the `Test and Track` dialog…
`Computer Bold`	Text that you must type	At the prompt, type: **`ls -l`**
Keycap	Keyboard keys	Press **Return**
`[Button]`	Buttons on the user interface	Click `[Delete]`. Click the `[Apply]` button.
`Menu Items`	A menu name followed by a colon (`:`) means that you select the menu, then the item. When followed by an arrow (`->`), a cascading menu follows.	Select `Locate:Objects->by Comment`

In This Guide

This guide describes the HP Service Activator sample service scenario for VPLS activation. The scenario is an end-to-end example that makes use of most aspects of the HP Service Activator framework, including workflows, and inventory system, template files, and a plug-in archive (PAR) for activating VPLS services on PE routers. The scenario does not implement a real solution, but is intended as a resource to leverage the design of service activation solutions.

Audience

The audience for this guide is:

- Systems Integrator, using it as a resource for building new solutions.
- Pre-sales staff, using it as a demonstration for customers.
- Educational staff, using it as student material in customer training.

The reader must understand the architecture, tools, and service delivery processes described in the *HP Service Activator – Introduction & Overview Guide*.

In addition, the reader should have a combination of some or all of the following:

- Understands and has a solid working knowledge of:
  - UNIX commands (for HP-UX and Solaris)
  - Microsoft Windows system administration
- Understands networking concepts and language
- Understands network and system security issues

References

MPLS: Technology and Application by Bruce Davie (Author), Yakov Rekhter

The Switch Book: The Complete Guide to LAN Switching Technology by Rich Seifert

The Internet Engineering Task Force RFC Document: Virtual Private LAN Services over MPLS by Marc Lasserre and Vach Kompella

Manual Organization

This guide contains the following chapters:

Chapter 1, “Understanding the VPLS Sample Service Scenario”, which describes the overall purpose of the VPLS sample service scenario and gives a brief overview of VPLS.

Chapter 2, “Installing the Sample Service Scenario”, which provides detailed instructions on how to install and configure scenario.

Chapter 3, “Sample Service Scenario Details”, which describes the sample service scenario in details.

Chapter 4, “Running the VPLS Sample Service Scenario”, which guides the reader through running the sample service scenario.

Chapter 5, “Uninstalling the Sample Service Scenario”, which describes how to completely remove the sample service scenario from your HP Service Activator installation.

1 Understanding the VPLS Sample Service Scenario

HP Service Activator is a framework that supplies most of the components needed to build service activation solutions for a wide variety of solution domains.

The VPLS Sample Service Scenario is an end-to-end example illustrating how many aspects of the Service Activator framework and the activation plug-in archives (PARs) that are used to deliver a service activation solution for an MPLS network provider.

The sample scenario includes:

- Sample customer relationship management (CRM) input messages.
- Controller and activation workflows.
- Inventory schemas and sample data.
- PE router plug-in.
- Simulation of PE devices.

Keep in mind that the scenario is not intended as the following:

- A complete demonstration. The scenario models a VPLS service which represents a simplification of actual VPLS services that a network service provider might offer and covers only a few of complexities of data modeling and workflows design for this type of services.
- A working solution that can be used in “real life“. The plug-in does not interact with an actual PE device, but the intention is to run the scenario on a single computer. In addition, the workflows do not handle all kinds of errors that could occur.
- A tutorial in and of itself. This document describes the scenario, but does not endeavor to explain all of the design choices or implementation details. You can learn a lot about Service Activator by studying this example, and many techniques used in the example would be wisely employed in any Service Activator solution.

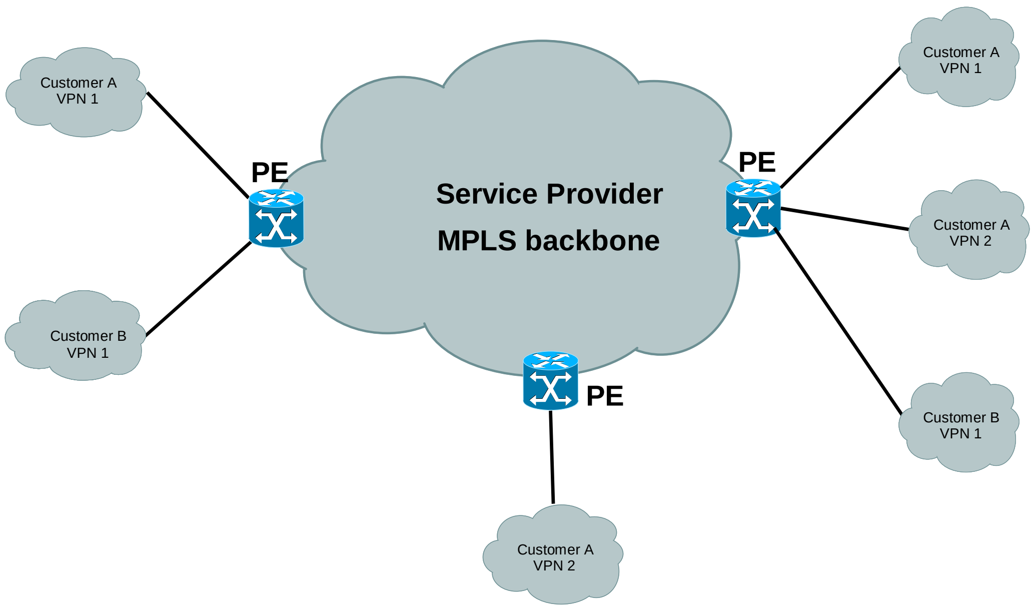
A complete description of the sample service scenario is presented in Chapter 3 on page 20.

VPLS Overview

VPLS is an Ethernet-based point-to-multipoint Layer 2 VPN. It allows you to connect geographically dispersed Ethernet LAN sites to each other across a Metropolitan or even long distance provider network. For customers who implement VPLS (Virtual Private LAN Services), all sites appear to be in the same Ethernet LAN even though traffic travels across the service provider's network.

Figure 1-1 is an illustration of customer sites being attached to PE routers. The backbone transport layer is MLPS-based (Multiprotocol Label Switching).

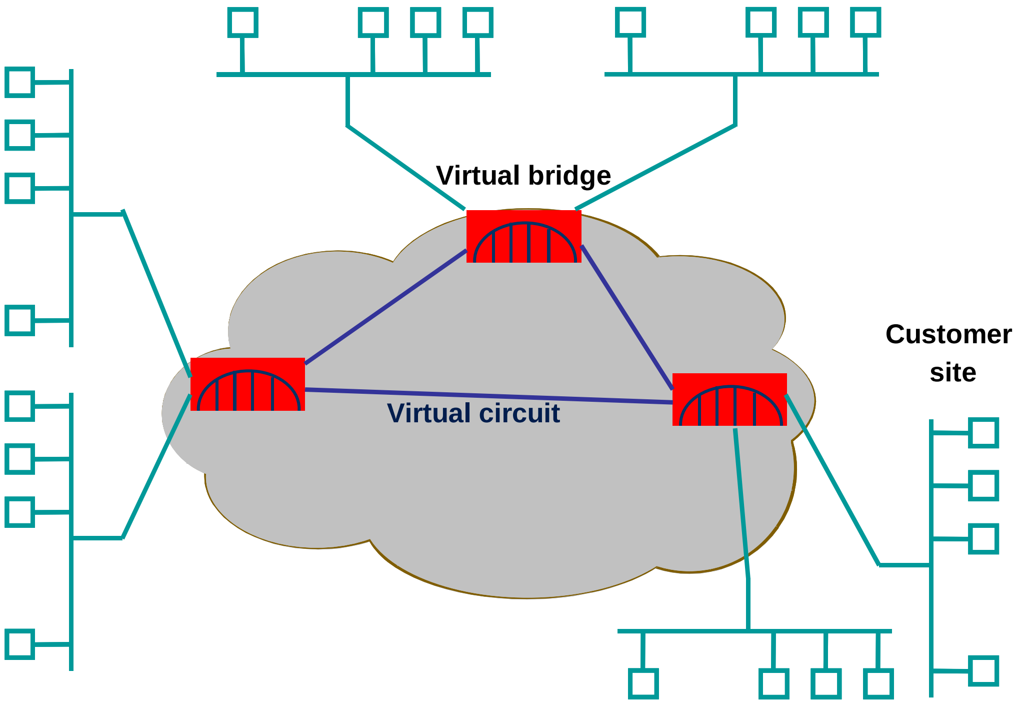
Figure 1-1 **Customer site interconnected over an MPLS network**



Each customer is attached to an Ethernet port on a Provider Edge (PE) device. The provider network then emulates a single bridge interconnecting all the customer Ethernet segments to form effectively a single LAN, which we call a VPN (Virtual Private Network); see Figure 1-2.

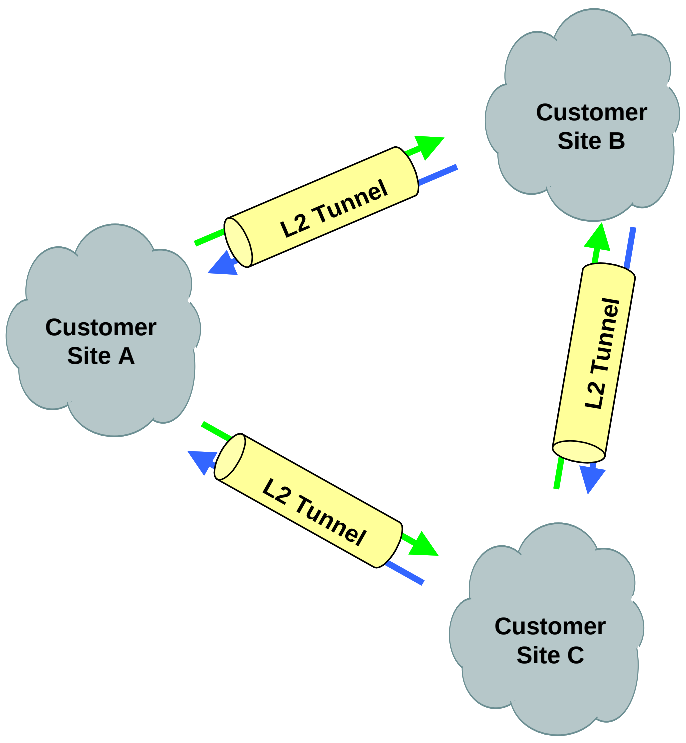
The physical interconnection of a customer site and a PE device port allows a site to be logically connected to multiple logical VPNs. Typically the traffic belonging to the different VPNs would be separated by means of VLAN tagging; this aspect is not dealt with in the scenario.

Figure 1-2 **Virtual Private Bridges**



All PEs in the network are connected together in a full mesh of MPLS Label Switched Path (LSP) tunnels with each tunnel carrying multiple pseudowires (see Figure 1-3). Pseudowires are point-to-point connection between PE devices set up for each customer service between the PEs. The example does not deal with configuration of tunnels in the core network; it handles setting up the pseudowires, but assumes that tunnels have been created by the service provider.

Figure 1-3 Full mesh of MPLS LSPs



When a PE device is used by a VPN for the first time, a full mesh of pseudowires must be created to the PE devices already in the VPN. The Label Distribution Protocol (LDP) is used to setup pseudowires in the service provider network and since each LSP is unidirectional, a pseudowire requires one LSP in each direction.

As the network offers a Layer 2 service to the end user, each PE is fully capable of learning all locally attached MAC addresses and associating learned remote addresses with a pseudowire. All unknown unicast, multicast and broadcast packets are flooded to all the PEs participating in a customer VPN.

2 Installing the Sample Service Scenario

The VPLS sample service scenario is included as part of the main Service Activator installation. It is packed in a single zip file that can be deployed on UNIX as well as Windows using the Deployment Manager.

After installing Service Activator, you can find the scenario zip file `VPN_Ex.zip` in the `$ACTIVATOR/examples/vpn_example` directory. You can *not* install (unpack) the scenario directly into a Service Activator installation; you need to deploy it using the Deployment Manager or by manually deploying the constituent parts yourself.

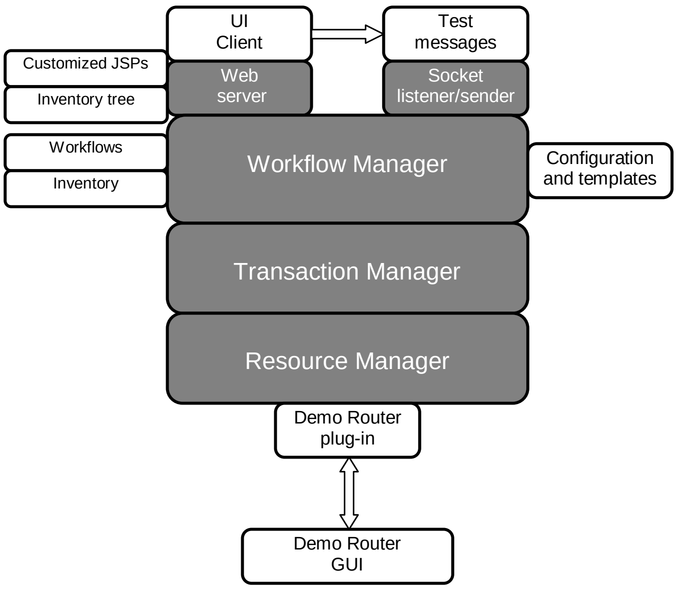
Before you deploy the solution you should unzip the file to a location of your choosing. From there, you can examine the files and see if there a risk of overwriting components that you may already have deployed into Service Activator.

NOTE The following instructions assume an installation of the scenario into an existing Service Activator installation.

Sample Service Scenario Contents

Figure 2-1 shows the components included in the scenario and their relations to each other. Components in the white boxes are supplied in this example, whereas the dark boxes are standard Service Activator components.

Figure 2-1 Sample Service Scenario components



The VPN Example (VE) consists of the following pieces, which are available after the scenario ZIP file is unpacked:

- **Workflows:** A complete collection of controller and activation workflows that work together to carry out the requested activations (or deactivations). Located in the `etc/workflows` directory.
- **Template files:** The template files referenced from the workflows. Located in the `etc/template_files/VPN_Ex` directory
- **Inventory components:** The components necessary to run the scenario, including:
    - XML resource definition files used by Inventory Builder to define inventory beans. These files are located in the `inventory` directory.
    - Two inventory tree XML files defining how inventory objects are displayed in the Operator UI. The files are located in the `etc/config/inventoryTree` directory.
    - SQL file used to populate the inventory database. Located in `etc/sql/populate_VPN_inventory.sql`.
    - An image file that is used to represent a port that has been reserved. The image file is located in the `UI/images/inventory-gui/tree` directory.
- **DTD file:** The definition of the valid structure of the XML messages that can be passed from the CRM. Located in `etc/config/vpn_message.dtd`.
- **XML messages:** The messages from a sample CRM system that can be injected into a running Service Activator installation to start activation. Located in `etc/tests/messages/VPN_Ex` for use within the Operator GUI.
- **Workflow Manager configuration file:** Contains the appropriate contents to be placed in your `mwfm.xml` file to configure the `SocketListenerModule` and the `SocketSenderModule` to exchange messages between Service Activator and a CRM system. Located in `etc/newconfig/mwfm_VPN_Example.xml`.
- **Plug-in archive:** Contains the plug-in (PAR-file) for the PE router simulator used in the VPLS example. Located in `plugins/VPN_Ex.DemoRouter.par`.
- **PE Router simulator and GUI:** The simulator Java classes are located in the `classes` directory. Two scripts (`runDemoRouter` for UNIX and `runDemoRouter.bat` for Windows) for launching the simulator GUI are located in the `bin` directory. In addition the `bin` directory contains a router icon and a Visual Basic wrapper script (`runDemoRouter_ui.vbs`) that can be used on Windows to launch the simulator GUI without opening an extra window for STDOUT and STDERR messages.

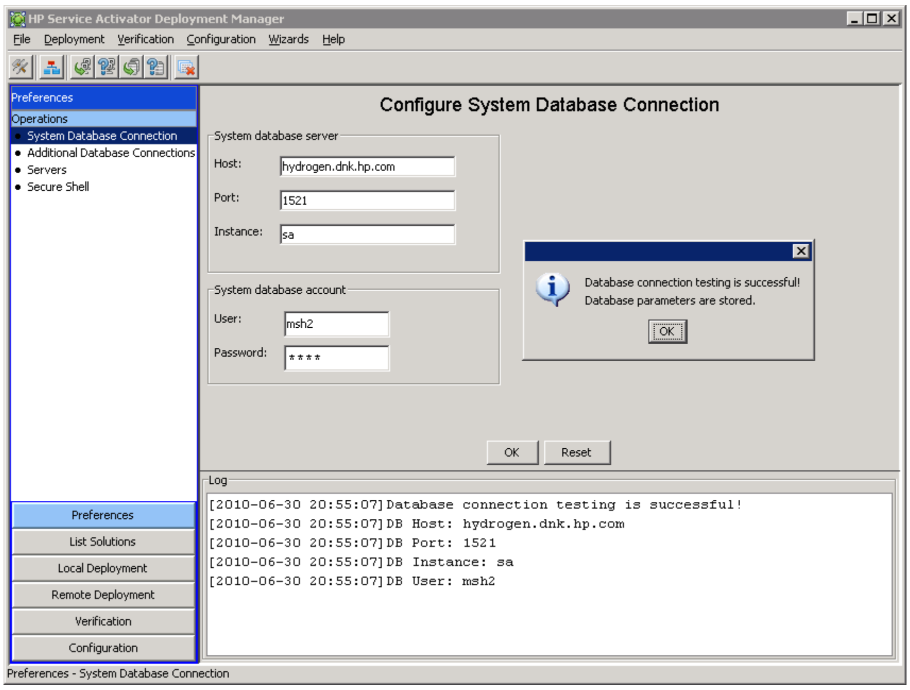
Installing the Scenario Using the Deployment Manager

This section illustrates how you can deploy the VPLS Sample Service Scenario on a single server using the Deployment Manager. The Deployment Manager supports many other operations than those shown here; for instance, it can be used to deploy a solution on an entire cluster environment and to verify that the cluster nodes are in sync. For more details about the Deployment Manager you should read the *Solution Separation and the Deployment Manager* document.

The Deployment Manager can be launched via the Deployment Manager desktop icon (on Windows) or by typing `deploymentmanager` on the command line.

Upon launching the Deployment Manager you should click “Configure Database Connection” under “Preferences”. This will bring up a screen allowing you to enter the username and the password for the Service Activator system database; see Figure 2-2.

Figure 2-2 Specifying database username and password


HP Service Activator Deployment Manager
File Deployment Verification Configuration Wizards Help
Preferences
Operations
System Database Connection
Additional Database Connections
Servers
Secure Shell
Preferences
List Solutions
Local Deployment
Remote Deployment
Verification
Configuration
OK
Reset
Log
Preferences - System Database Connection

Configure System Database Connection

System database server

Host: hydrogen.dnk.hp.com

Port: 1521

Instance: sa

System database account

User: msh2

Password: ****

Database connection testing is successful!
Database parameters are stored.

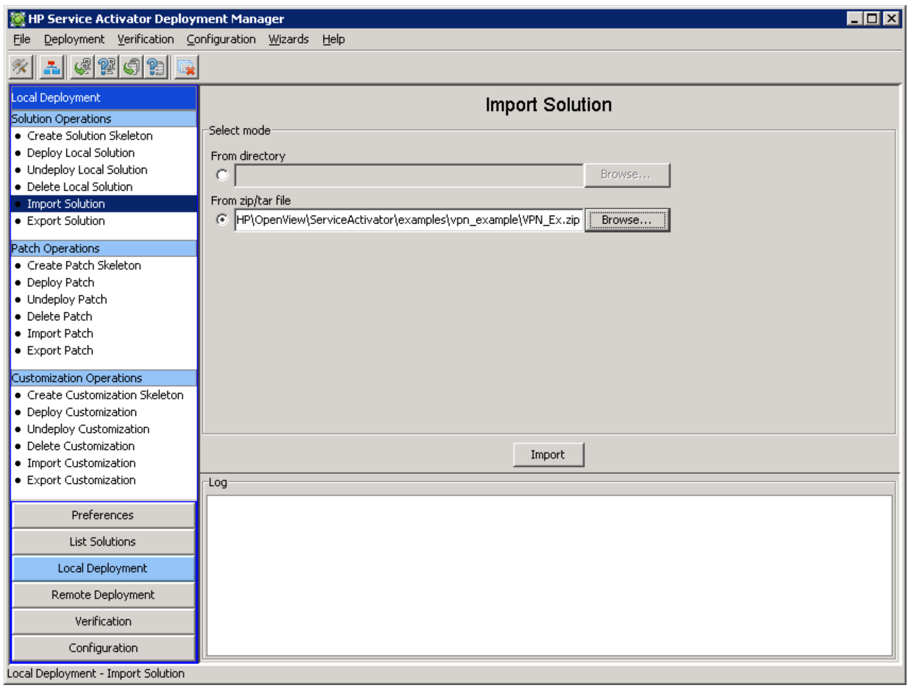
```
[2010-06-30 20:55:07]Database connection testing is successful!
[2010-06-30 20:55:07]DB Host: hydrogen.dnk.hp.com
[2010-06-30 20:55:07]DB Port: 1521
[2010-06-30 20:55:07]DB Instance: sa
[2010-06-30 20:55:07]DB User: msh2
```

Import the VPLS Sample Service Scenario into the Deployment Manager by following these steps:

- Select the “Import Solution” operation under “Local Deployment”.
- Click the “From a zip/tar file” radio button and click the `[Browse…]` button.
- Locate the `VPN_Example.zip` file in the `$ACTIVATOR_OPT/examples/vpn_example` directory.
- Finally, click the `[Import]` button.

A screenshot illustrating this is shown in Figure 2-3.

Figure 2-3 **Importing the VLPS Sample Service Scenario into Deployment Manager**

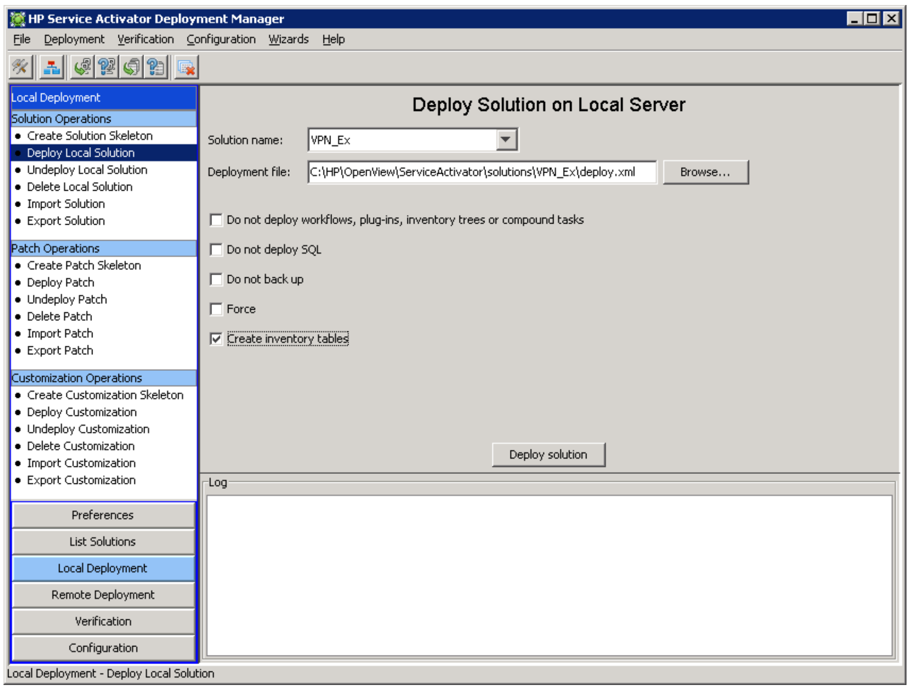


From zip/tar file: HP\OpenView\ServiceActivator\examples\vpn_example\VPN_Ex.zip

After having imported the solution into Deployment Manager you need to click on the “Deploy Local Solution” operation. In the “Deploy Local Solution” screen follow these steps (see Figure 2-4):

- Select the `VPN_Ex` solution from the dropdown list
- Click on the `[Browse...]` button and select the `deploy.xml` that specifies the components to be deployed (the Deployment Manager automatically selects the `deploy.xml` file if it exists).
- Make sure that “Create inventory tables” is checked
- Click on the `[Deploy Solution]` button

Figure 2-4 Deploying a solution on the local server using Deployment Manager



Deploy Solution on Local Server

Solution name: VPN_Ex

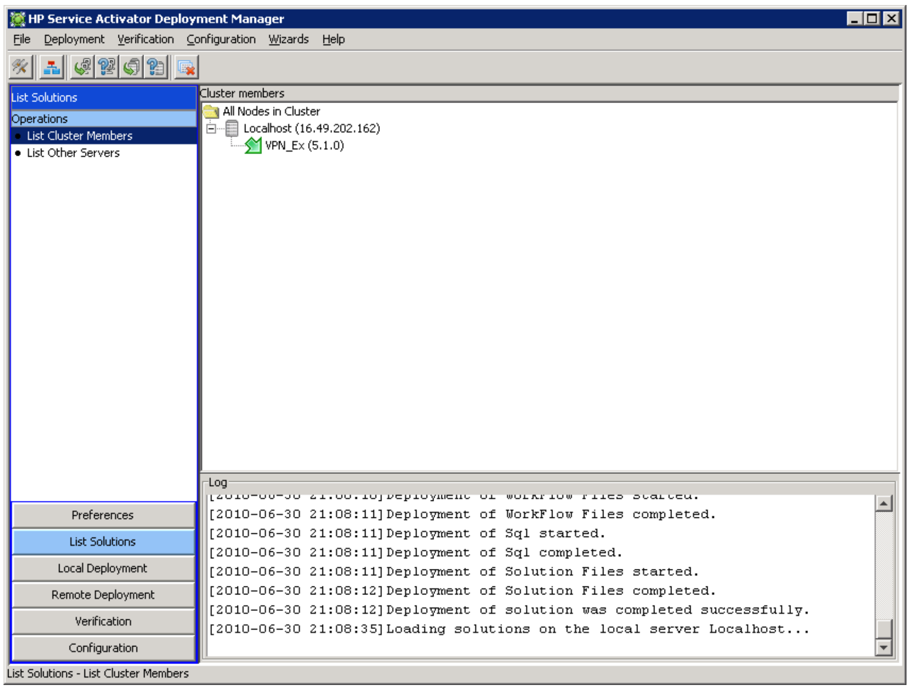
Deployment file: C:\HP\OpenView\ServiceActivator\solutions\VPN_Ex\deploy.xml

- [ ] Do not deploy workflows, plug-ins, inventory trees or compound tasks
- [ ] Do not deploy SQL
- [ ] Do not back up
- [ ] Force
- [x] Create inventory tables

After the deploy operation has finished you should see a dialog indicating that the operation went well.

Finally, after having deployed the VPLS scenario you can use the Deployment Manager’s “List Solution” feature to list the solutions deployed on your server; see Figure 2-5.

Figure 2-5 Listing solutions using the Deployment Manager



```
All Nodes in Cluster
  Localhost (16.49.202.162)
    VPN_Ex (5.1.0)
```

```
[2010-06-30 21:08:11]Deployment of WorkFlow Files completed.
[2010-06-30 21:08:11]Deployment of Sql started.
[2010-06-30 21:08:11]Deployment of Sql completed.
[2010-06-30 21:08:11]Deployment of Solution Files started.
[2010-06-30 21:08:12]Deployment of Solution Files completed.
[2010-06-30 21:08:12]Deployment of solution was completed successfully.
[2010-06-30 21:08:35]Loading solutions on the local server Localhost...
```

Completing the Scenario Installation

You need to carry out the following tasks to make the scenario functional.

1. **Configure the `SocketListenerModule`**

   Configure the `SocketListenerModule` of the Workflow Manager to receive messages from the CRM. To do this you will need to edit the file `$ACTIVATOR_ETC/config/mwfm.xml` and merge in the contents of `$ACTIVATOR_OPT/solutions/VPN_Ex/etc/newconfig/mwfm_VPN_Example.xml`. See the notes in `mwfm_VPN_Example.xml` for more information.

2. **Configure the `SocketSenderModule`**

   It is advised to configure the `SocketSenderModule` as well. The Controller workflow returns a response via the `SocketSenderModule`, and error messages will be logged if the module is not included. See the notes in the file `mwfm_VPN_Example.xml` for more information.

3. **Configure the `DBAuditModule`**

   It is recommended to enable the `DBAuditModule` as well. This is done by editing the file `$ACTIVATOR_ETC/config/mwfm.xml` and removing the comment tags surrounding the module named `auditor` and setting the `store_audit` attribute to `true`.

4. **Configure the Operator UI to enable a special testing page**

   Configure the Operator UI to allow for injecting tests messages into the Workflow Manager from the UI. To do this, edit the file `$JBOSS_DEPLOY/hpovact.sar/activator.war/WEB-INF/web.xml`. In the `login` servlet part, set the value of the `tests` parameter to “`true`”; see the example below:

```
<init-param>
  <param-name>tests_dir</param-name>
  <param-value>
    C:/HP/OpenView/ServiceActivator/etc/tests/messages
  </param-value>
</init-param>

<!-- set the <param-value> to true to enable the tests ->
<init-param>
  <param-name>tests</param-name>
  <param-value>false</param-value>
</init-param>
```

5. **Configure the `runDemoRouter.bat` script (Windows only, optional)**

   If you have not installed HP Service Activator on the C: drive you will need to edit the file `runDemoRouter.bat` located in the `$ACTIVATOR_OPT/solutions/VPN_Ex/bin` directory. Change the drive-letter to match your Service Activator installation.

After you have completed all these steps it is necessary to restart Service Activator to make the changes take effect.

NOTE Most components in a typical Service Activator solution are hot-deployable. However, the Java bean classes generated by the Inventory Builder cannot be hot-deployed.

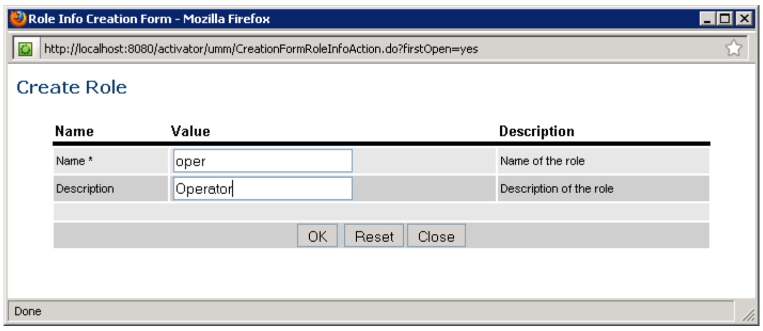
6. **Assign the VPN example’s inventory trees to a role (if authentication is enabled)**

If you have enabled authentication in the Workflow Manager’s configuration file the VPN Scenario’s inventory trees will not be immediately visible. In order to be able to see (and interact with) the inventory trees you need to use the User Management Interface to assign the “VPN Example” inventory trees to a particular role.

The following steps will help you create a new role “oper”, assign it to the user “sys” (the system user in this example) and then permit the “sys” user to access the inventory trees.

Figure 2-6 shows the form to create a new role. The form is opened by clicking on the “Add Role” link under “User Management”. Create a new role named “oper” and hit `[OK]`.

Figure 2-6 Creating a new Role


Role Info Creation Form - Mozilla Firefox
http://localhost:8080/activator/umm/CreationFormRoleInfoAction.do?firstOpen=yes
Done

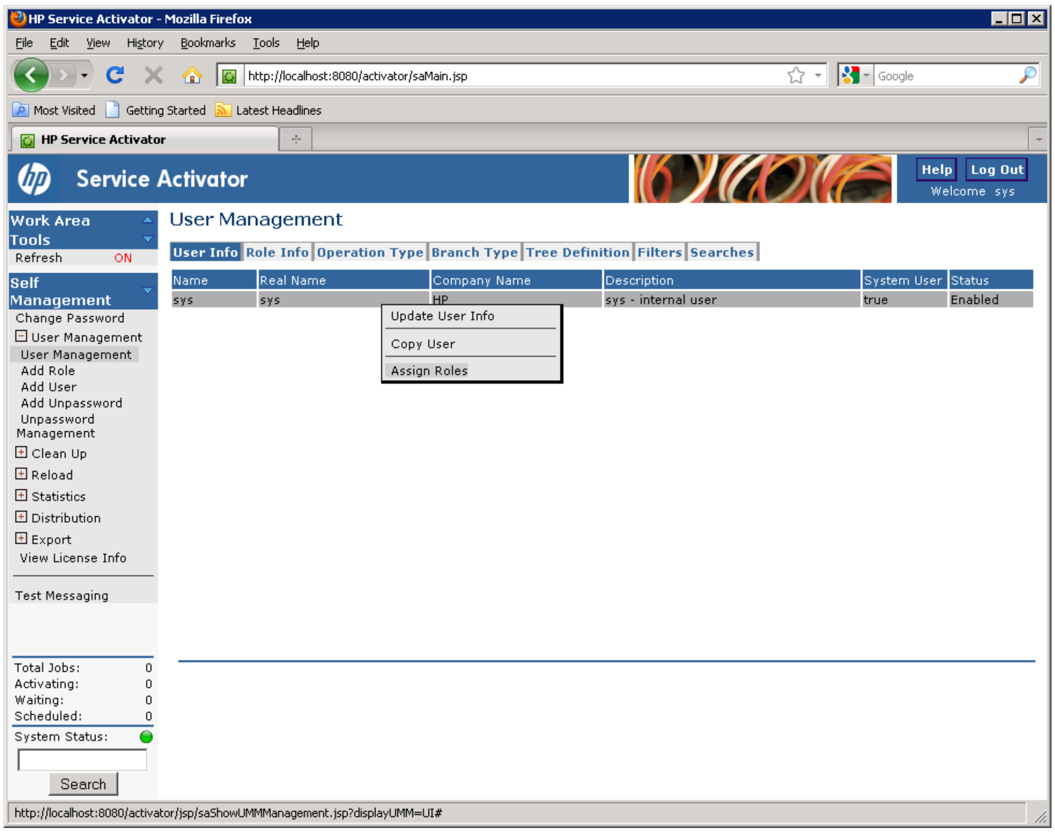
Create Role

Name	Value	Description
Name *	oper	Name of the role
Description	Operator	Description of the role

Next, assign the role “oper” to user “sys” by right clicking on the user in the “User Management” menu and selecting the `Assign Roles` menu item. See figure Figure 2-7.

NOTE Since the user in this example is the system user, there’s no need to assign roles to the user. The system user will automatically have all roles.

Figure 2-7 **Assign Role to User**



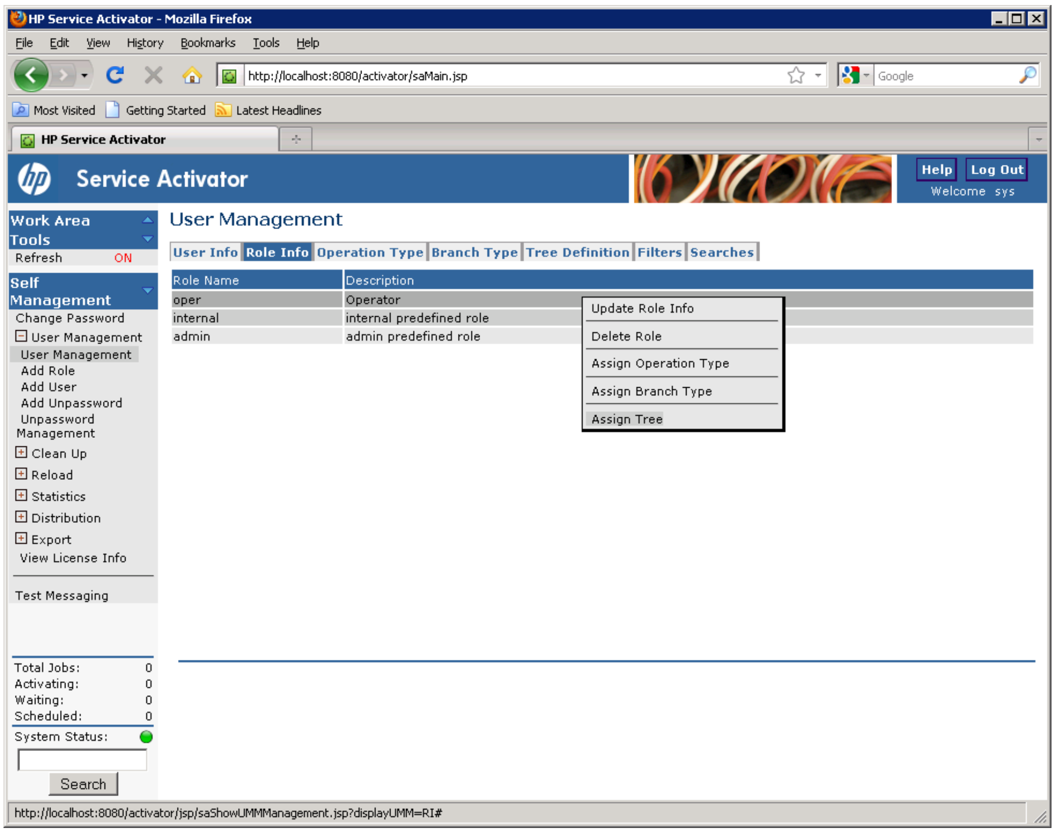
Total Jobs: 0
Activating: 0
Waiting: 0
Scheduled: 0
System Status: ●

Name	Real Name	Company Name	Description	System User	Status
sys	sys	HP	sys - internal user	true	Enabled

http://localhost:8080/activator/jsp/saShowUMMManagement.jsp?displayUMM=UI#

Figure 2-8 shows how to assign an inventory tree to a role. In the figure below, the inventory tree is assigned to the “oper” role by right-clicking on “oper” and selecting the `Assign Tree` menu item. This will bring up a window that allows you to assign inventory trees to the selected roles; see Figure 2-9.

Figure 2-8 **Assigning a inventory trees to a role using the User Management Interface**



HP Service Activator - Mozilla Firefox

http://localhost:8080/activator/saMain.jsp

User Management

User Info | Role Info | Operation Type | Branch Type | Tree Definition | Filters | Searches

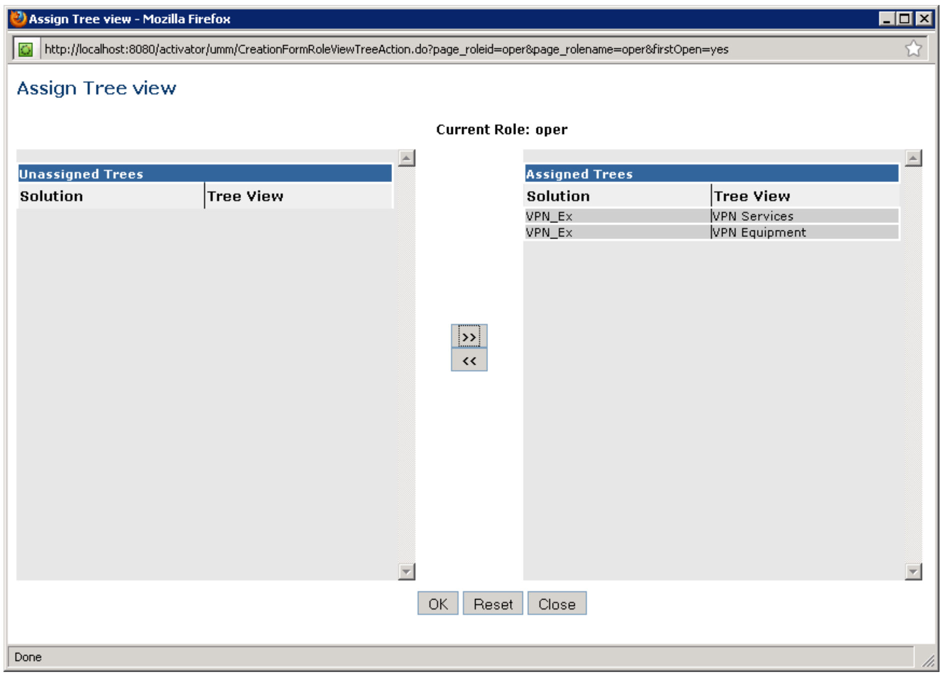
Role Name	Description
oper	Operator
internal	internal predefined role
admin	admin predefined role

Update Role Info
Delete Role
Assign Operation Type
Assign Branch Type
Assign Tree

Total Jobs:	0
Activating:	0
Waiting:	0
Scheduled:	0
System Status:	●

http://localhost:8080/activator/jsp/saShowUMMManagement.jsp?displayUMM=RI#

Figure 2-9 **Assigning two inventory tree to the role "oper"**



Assign Tree view - Mozilla Firefox

http://localhost:8080/activator/umm/CreationFormRoleViewTreeAction.do?page_roleid=oper&page_rolename=oper&firstOpen=yes

Assign Tree view

Current Role: oper

Unassigned Trees

Solution	Tree View

Assigned Trees

Solution	Tree View
VPN_Ex	VPN Services
VPN_Ex	VPN Equipment

3 Sample Service Scenario Details

Before you get started with this chapter, read Chapter 1 in this guide. Chapter 1 describes the network model on which the sample service scenario is based.

This chapter describes the details of the sample service scenario. In this Service Activator solution example, we model a Service Provider offering VPLS services to the customers.

In our model, customers can get the following service offerings:

- **Layer 2 VPN:** A full mesh of Virtual Connections interconnecting the PE devices where its sites are attached.
- **Layer 2 Site:** Sites are attached to the provider's network using Ethernet ports.

The workflows that implement the activation of these services are triggered by messages sent by a CRM system.

Understanding the CRM Message Format and Example Messages

The scenario assumes that a CRM exists and is capable of composing and sending XML messages that indicate the type of service that should be activated. In order to run the example without a CRM, we use the testing facilities built into Service Activator. With this facility it is possible to inject test messages into the running installation and in that way simulating a CRM system.

The Document Type Definition (DTD) supplied with this scenario defines the valid messages that start the activations. You can find the DTD in `$ACTIVATOR_ETC/config/vpn_message.dtd`.

Every message from the CRM contains a header and, optionally, a body. The header contains:

- **A message identifier `<message_id>`:** This ID provides unique identification of the message from the CRM. When the activation is complete, the workflows issue a response message that contains the same message identifier and an indication of whether or not the activation was successful.
- **A service identifier `<service_id>`:** This ID provides unique identification of the service being created or modified.
- **An action indicator `<action>`:** This indicates what action to perform. The valid actions are `CreateVPN`, `DeleteVPN`, `CreateSite`, `DeleteSite`, `AddSite` and `RemoveSite`.

The content of the body of the message is dependent on the action performed:

- For `CreateVPN` the body includes VPN name and customer id.
- For `CreateSite` the body includes site name, customer id, and optionally the PE device and port where the site will be connected to the provider network.

- For `AddSite` the body includes the VPN id and optionally the parameters “RateLimit” and “ClassOfService”.
- For `RemoveSite` the body includes the VPN id where the Site is removed from.
- `DeleteSite` and `DeleteVPN` have no body.

The following example shows a message for adding a Site to a VPN with Gold service and a rate limit of 2 Mbps:

```
<msg>
  <header>
    <message_id>mess001</message_id>
    <customer_id>site001</customer_id>
    <action>AddSite</action>
  </header>
  <body>
    <AddSite>
      <VPNId>vpn001</VPNId>
      <CoS>Gold</CoS>
      <BW>2M</BW>
    </AddSite>
  </body>
</msg>
```

Review the DTD and the example messages in `$ACTIVATOR_ETC/tests/messages/VPN_Ex` for a more complete understanding of what these messages contain. There are examples for all possible kinds of messages supported by the scenario.

When the workflow has finished a response message is returned including the result of the activation. The following example shows an “OK” and an “ERROR” message:

```
<response_msg>
  <header>
    <message_id>mess001</message_id>
    <service_id>site001</service_id>
  </header>
  <body>
    <message>OK</message>
  </body>
</response_msg>
```

```
<response_msg>
  <header>
    <message_id>mess001</message_id>
    <service_id>site001</service_id>
  </header>
  <body>
    <message>ERROR</message>
    <description>Port does not exist</description>
  </body>
</response_msg>
```

Understanding Service Mapping

The scenario strives to ensure that the solution is flexible enough to allow new services and options to be added easily without having to alter the logic of any of the existing workflows. The scenario achieves this through the use of service mapping.

In this example, the controller workflow simple appends value of the `<action>` element to “VE_” in order to determine the name of the service workflow that will can handle this service. So, if the value of the `<action>` element is `AddSite`, the controller workflow will spawn the service workflow `VE_AddSite`.

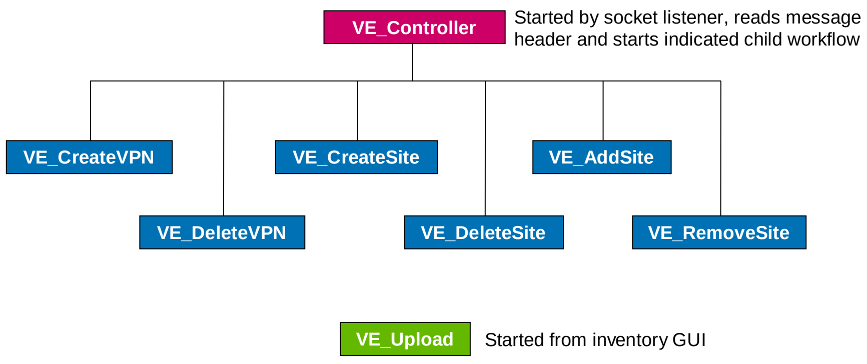
NOTE A real solution would typically implement service mapping with a database table that determines how actions map to service workflow.

Understanding Workflows

The workflows supplied in the scenario are typically categorized as either **controller** workflows or **activation** workflows. A controller workflow does not perform any activation. Instead, it gathers data and directs the actions of the workflows that perform the activations—the activation workflows.

Figure 3-1 shows all of the workflows that are part of the scenario. Those shown in magenta are controller workflows; those shown in blue are activation workflows. Finally there is also an Upload workflow which is used to discover the hardware configuration of a PE device. This workflow is not initiated by an order message, but from the Service Activator inventory GUI.

Figure 3-1 Workflow invocation diagram


VE_Controller
Started by socket listener, reads message header and starts indicated child workflow
VE_CreateVPN
VE_CreateSite
VE_AddSite
VE_DeleteVPN
VE_DeleteSite
VE_RemoveSite
VE_Upload
Started from inventory GUI

Understanding Controller Workflows

The distinction between controller and activation workflows is somewhat arbitrary. There is no difference in the workflows from the perspective of the Workflow Manager, but the distinction may help in the understanding of a complex network of workflows as in the scenario. Controller workflows typically perform some preparatory operations, start a child workflow, and then wait for the child workflow to complete.

Understanding Activation Workflows

There are two kinds of activation workflows in our scenario: Those that activate or remove a new service, and those that create or modify service instances in the database.

The workflows that do the real activation always use a single task list to ensure that the entire service is functional, i.e. it constituent atomic tasks are rolled back in case of errors.

The workflows that does the database operations (`Create-`, `Delete-`) normally validates that changing the service instances in the data base is consistent with the network, i.e. validate VPN has no Sites connected before deleting it.

The following list gives a short description of each workflow in the scenario:

- `VE_Controller` – Receives message from CRM (using the `SocketListenerModule`), determines which workflow to run and then waits for a child workflow to complete. When the child workflow finishes, a response message is sent back including the success or failure of the request.
- `VE_CreateVPN` – An empty VPN instance is created in the inventory.
- `VE_DeleteVPN` – The VPN is deleted in the inventory. It is verified that no Sites are currently connected to the VPN.
- `VE_CreateSite` – Create a Site in the inventory. Optional parameters are the PE router and port used to connect to the provider network. If not present the operator are prompted for the information.
- `VE_DeleteSite` – Delete a Site in the inventory. It is verified that the Site is not connected to any VPN.
- `VE_AddSite` – Add a Site to the VPN, i.e. allow traffic to/from other sites of the VPN to pass to/from the site via the physical connection which exists already.
  To activate this service, there are two basic tasks that must be performed on PE devices. In the initial phase the workflow builds a Tasklist and along the way appends atomic tasks:
  - Build the infrastructure of virtual circuits needed between different PEs which all serves the same VPN: Whenever a VPN grows to involve a new PE, it must be connected by virtual circuits (pseudowires) to all those previously involved. To setup a pseudowire two atomic tasks are needed. Since it is bidirectional, both ends must be configured.
  - Configure the bridging of traffic for a specific VPN to/from a specific port; this must be done whenever a site joins a VPN.
- `VE_RemoveSite` – Remove a Site from the VPN. Validate that the Site is currently not connected to any VPN. Configure the PE devices:
  - Remove the connections between different PEs when the removed Site is the last one on the PE. Just as when the pseudowire was set up, two atomic tasks are needed to remove it.
  - Remove the bridging of traffic for a specific VPN to/from a specific port; this must be done when a site is removed from the last VPN.
- `VE_Upload` – Request port information from the PE device. The DemoRouter plug-in returns a list of port names, which is used by the workflow to create the corresponding port instances on the PE device. The workflow is not activated by the controller workflow but from the Inventory UI.

Resource Reservation

The scenario uses the ability of the inventory system to provide for resource reservation (see the “Understanding Inventory” section for more details). The Port object in the scenario inventory is reserved in the `VE_CreateSite` workflow and released in the `VE_DeleteSite` workflow.

Synchronization

In all cases when a parent workflow starts a child workflow, the parent waits for the child to complete. In this scenario, the parent workflows always wait in the `sync` queue, asking for `sync_status` and `sync_description`. The body of the child workflows does *not* contain `Sync` nodes. Instead, the child workflows use the `SyncHandler` as an end handler to ensure that no matter how the workflow is ended it will consistently synchronize with its waiting parent.

Error Handling

The scenario attempts to include appropriate handling for most expected errors, such as an inability to reserve a resource or an activation error. However, there are some error cases that the scenario does not handle completely. Study the error handling techniques found in the scenario and apply them carefully throughout your workflows.

Diagnostics

The scenario includes a significant number of workflow nodes that output informational messages to the operator. These workflows are helpful for debugging, but are probably not appropriate in your final implementation. You can include the workflow nodes that you find helpful during implementation of your solution, and then make the extraneous workflow nodes inactive before delivering the service. This allows reactivation of the nodes if you need more detailed output of workflow progress.

Understanding Inventory

The scenario provides a set of inventory objects that fall into two categories:

- VPLS Service entities (Customer, VPN, Site and SiteConnection)
- Equipment entities (PERouter and Port)

The definition files for the inventory objects provided with the scenario are found in the `$ACTIVATOR_OPT/solutions/VPN_Ex/inventory`. The objects provided are:

- `Customer` – Name of the Customer who ordered the VPN.
- `VPN` – A collection of interconnected sites.
- `Site` – A LAN connected to a port on a PE device. Includes the Name of the Site and the Router and port used to connect to the provider network.
- `Contact` – A contact person that can be associated with a Site. Sites created by the included test messages do not use the Contact attribute, but from the Inventory UI it can be selected from a drop down list.
- `SiteConnection` – Logical connection of a Site to a VPN. Includes name of the VPN and Site, and optionally ClassOfService and bandwidth.
- `PERouter` – The PE device used to connect to the provider network. Includes Name, IP address, username and password for logging into the device.
- `Port` – The port on the PE device. Includes Name, port number and reservation count.

These inventory objects are minimal because they store only the information needed by the Sample Service Scenario. In a real solution, you might want to include a significant list of other details such as model numbers, serial numbers, physical location, and so on.

Understanding the DemoRouter Plug-in

A DemoRouter plug-in has been implemented for the Sample Service Scenario. It has the following atomic tasks:

- **VPN_Ex.DemoRouter_Connect_Site:** Parameters (PERouter, port, password, vpn_id, cos and bw).
  The purpose of the task is to connect the Site to the provider network via the PERouter and port. The parameters are used to identify the PE device and log in to it. Also the VPN being connected to is identified (vpn_id) and the QualityOfService and bandwidth the customer has ordered for the connection.

- **VPN_Ex.DemoRouter_Disconnect_Site:** Parameters (PERouter, port, password and vpn_id). The purpose of the task is to disconnect the Site from the provider network.
- **VPN_Ex.DemoRouter_Connect_peer:** Parameters (PERouter, port, password, peerRouter and vpn_id). The purpose of the task is to build the pseudowire in the direction from PERouter to peerRouter. The parameters are used to identify the PE device and log into it. In addition the peer router is defined; i.e. the other end of the pseudowire that connects the two PE devices.
- **VPN_Ex.DemoRouter_Disconnect_peer:** Parameters (PERouter, port, password, peerRouter and vpn_id). The purpose of the task is to remove the pseudowire going from PERouter to peerRouter.
- **VPN_Ex.DemoRouter_Upload:** Parameters (PERouter, port, password). The purpose of the task is to upload the available ports on the PERouter. The result is returned as a list of space separated names in a case packet variable called “ports”. The parameters are used to identify the PE device and log in to it.

The scenario does not use compound tasks to define the list of atomic task. Compound tasks are statically defined whereas this scenario needs to define dynamic lists of tasks. To dynamically build lists of tasks the `VE_AddSite` and `VE_RemoveSite` workflows use the TaskList feature that has been available since Service Activator version 4.0. The following is an example of the resulting Tasklist in the `VE_AddSite` workflow where a Site is connected to device A and the VPN has already Sites on devices B and C (not all arguments are shown):

```
DemoRouter_Connect_Site(A)        ; Connect the site to the PE device and port
DemoRouter_Connect_peer(A, B)     ; Create the A-end of the wire between A and B
DemoRouter_Connect_peer(B, A)     ; Create the B-end of the wire between A and B
DemoRouter_Connect_peer(A, C)     ; Create the A-end of the wire between A and C
DemoRouter_Connect_peer(C, A)     ; Create the C-end of the wire between A and C
```

Understanding the DemoRouter GUI

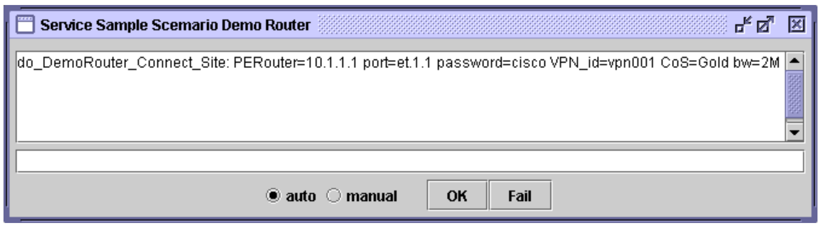
The DemoRouter GUI works together with the VPN_Ex.DemoRouter plug-in. It has two purposes:

- Server program, simulating all PE devices.
- Display the activation of atomic tasks and control the behavior of the server.
- After configuring the plug-in as described in Chapter 2, start the DemoRouter GUI by going to the directory `$ACTIVATOR_OPT/solutions/VPN_Ex/bin` and run the script `runDemoRouter.sh` (UNIX) or `runDemoRouter.bat` (Windows).

As it is shown in Figure 3-2, the DemoRouter GUI has the following elements:

- A big text area where every execution of an atomic task is logged. The log includes the task name and the value of all arguments.

Figure 3-2 DemoRouter GUI

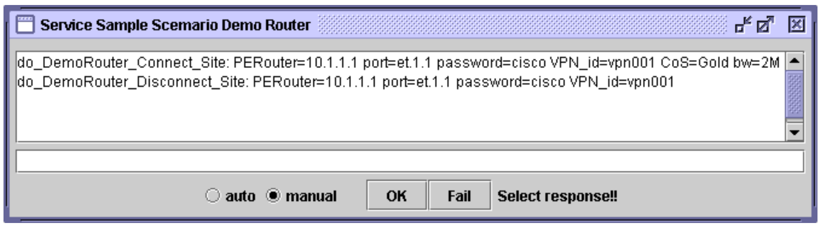

Service Sample Scemario Demo Router

```
do_DemoRouter_Connect_Site: PERouter=10.1.1.1 port=et.1.1 password=cisco VPN_id=vpn001 CoS=Gold bw=2M
```

[✓ auto] manual

- A radio button to control the behavior of the simulator. In the automatic mode activations are returned at once with a successful result. In the manual mode activations are returned when one of the buttons `[OK]` or `[Fail]` is pushed, and with a result in accordance with the button. As shown in Figure 3-3 the message “*Select response!!*” is displayed to notify that a request is waiting.

Figure 3-3 **DemoRouter GUI - Request User Response**

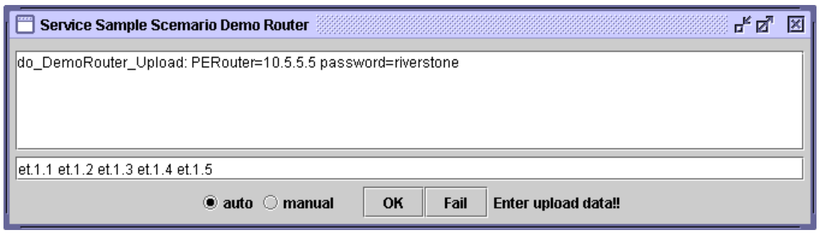


```
do_DemoRouter_Connect_Site: PERouter=10.1.1.1 port=et.1.1 password=cisco VPN_id=vpn001 CoS=Gold bw=2M
do_DemoRouter_Disconnect_Site: PERouter=10.1.1.1 port=et.1.1 password=cisco VPN_id=vpn001
```

auto [✓ manual] OK Fail Select response!!

- Finally a one-line text input field is used to return uploaded data. Data must be a list of space separated port names. As shown in Figure 3-4, a message “*Enter upload data!!*” is displayed to notify that upload data is requested by the user.

Figure 3-4 **DemoRouter GUI - Request upload data**



```
do_DemoRouter_Upload: PERouter=10.5.5.5 password=riverstone
```

```
et.1.1 et.1.2 et.1.3 et.1.4 et.1.5
```

[✓ auto] manual OK Fail Enter upload data!!

4 Running the VPLS Sample Service Scenario

The scenario is well suited for use as a demonstration, in fact, if a local Oracle database is available, the scenario can be demonstrated on a single Windows PC.

Of course, the main value proposition of Service Activator lies in its ability to configure network devices, but an effective demonstration can be presented that does not require any additional hardware beyond the PC that is running Service Activator.

Understanding the test messages

Using the testing facilities built into Service Activator, it is possible to inject test messages into the running installation to cause the scenario workflows to start. The scenario comes with a complete set of test messages and uses the ability of the Workflow Manager to run workflows as if activations were taking place.

The messages to inject that drive the scenario are found in `$ACTIVATOR_ETC/tests/messages/VPN_Ex`. The Operator UI allows the user to process a message by right clicking on the message.

NOTE Before beginning the demo, make sure the plug-in simulator has been started; go to the `$ACTIVATOR_OPT/solutions/VPN_Ex/bin` directory and run the script `runDemoRouter.sh` (UNIX) or `runDemoRouter.bat` (Windows).

The following is a list of the test messages and a description of their functionality:

- `CreateVPN1` – Create a VPN in the inventory with the names: “`VPN 1`”.
- `CreateVPN2` – Similar to `CreateVPN1`.
- `CreateSite1` – Create a Site with the name “`Site 1`”. Included in the message is the name of a router and port used for connection to the provider network. (RS3000-1, et.1.1), i.e. the `VE_AddSite` workflow is performing a flow-through activation without requesting further information.
- `CreateSite2` – Similar to `CreateSite1`.
- `CreateSite3` – Create a Site with the name “`Site 3`”. The message does not include any router information. The workflow `VE_AddSite` will request this information and the user will have to interact with the workflow to supply these additional parameters.
  To interact with the `VE_AddSite` workflow, first list the active jobs and then right click on the job in the Step: **Select_router_and_port**.
- `CreateSite4` – Similar to `CreateSite3`.

- `DeleteSite[1-4]` – Delete the Site from inventory that was created by `CreateSite[1-4]`. The workflow is validating that the Site is not currently connected to any VPN.
- `AddSite1ToVPN1` – Connects “`Site 1`” to “`VPN 1`”. Included in the message is the name of the VPN and additional *ClassOfService* and *RateLimit* parameters.
- `AddSite[2-4]ToVPN1` – Connects “`Site [2-4]`” to “`VPN 1`”. The message does not include any *ClassOfService* or *RateLimit* parameters, in the activation the PE Router have to use default values.
- `AddSite4ToVPN2` – Connects “`Site 1`” to “`VPN 1`”.
- `RemoveSitenFromVPNm` – Remove “`Site n`” from “`VPN m`”.

Running the Demo Step by Step

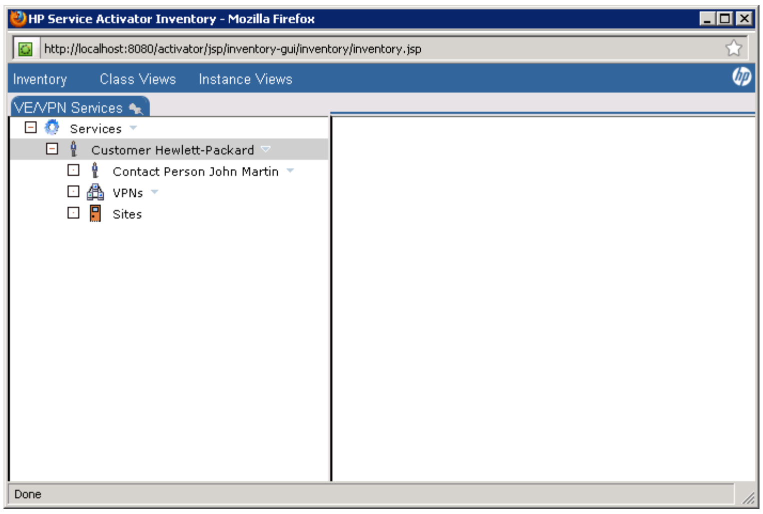
As a help to get started quickly, we have included a step by step guide that lists what you have to do and how the results of the different messages can be verified.

The scenario will create three Sites located on three different routers, and the Sites are afterwards connected to a single VPN. Finally, a new PE router is created in the inventory and ports are uploaded from the device.

Some of the steps are illustrated with screenshots of the Inventory UI and the DemoRouter GUI.

1. First of all the Service Sample Scenario must have been deployed and configured as described in Chapter 2. The deployment process has also loaded some equipment into the inventory database.
2. The DemoRouter GUI is started and running.
3. The Inventory UI displaying the Services instance tree before creating any services is shown in Figure 4-1.

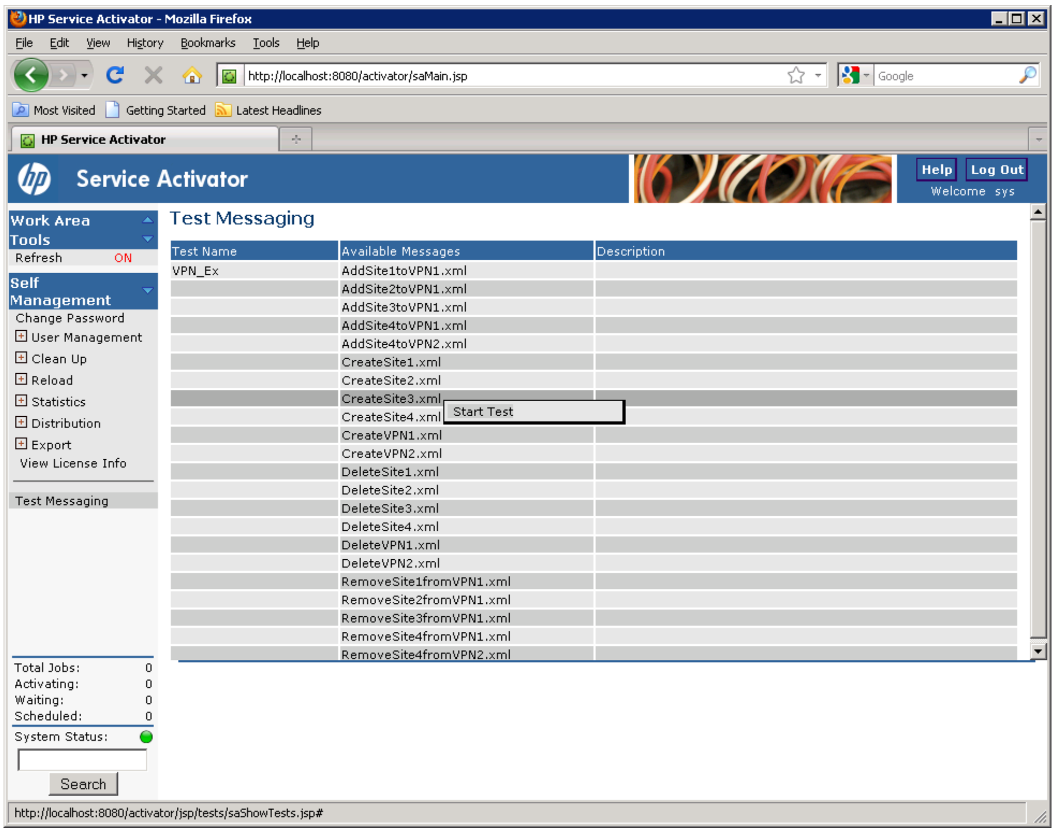
Figure 4-1 **Inventory UI**


HP Service Activator Inventory - Mozilla Firefox
http://localhost:8080/activator/jsp/inventory-gui/inventory/inventory.jsp
Inventory
Class Views
Instance Views
VE/VPN Services
Services
Customer Hewlett-Packard
Contact Person John Martin
VPNs
Sites
Done

4. Before proceeding to the next step, run the `CatchSocketSenderMessages` (UNIX) or `CatchSocketSenderMessages.bat` (Windows) script located in `$ACTIVATOR_OPT/bin`.
5. Run the test messages: `CreateVPN1`, `CreateSite1`, `CreateSite2` and `CreateSite3`.

In order to send test messages, select the `Test Messaging` menu item in the “Self Management” section to get a list of the available test messages. Right click on the desired message and from the popup menu, select the `Start Test` menu item; see Figure 4-2.

Figure 4-2 Injecting test message into Service Activator

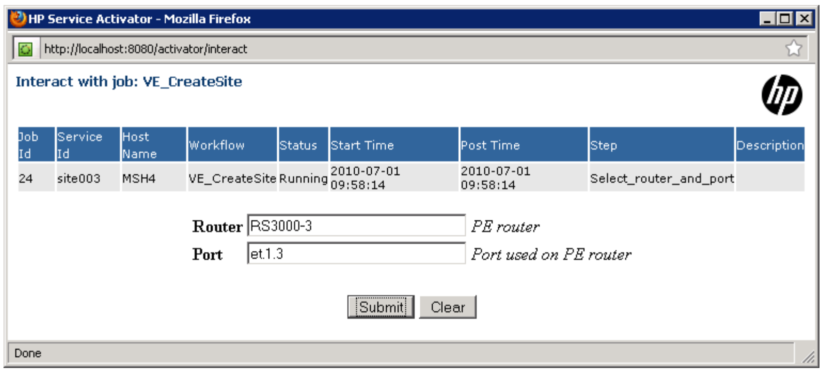


Test Name	Available Messages	Description
VPN_Ex	AddSite1toVPN1.xml	
	AddSite2toVPN1.xml	
	AddSite3toVPN1.xml	
	AddSite4toVPN1.xml	
	AddSite4toVPN2.xml	
	CreateSite1.xml	
	CreateSite2.xml	
	CreateSite3.xml	
	CreateSite4.xml	
	CreateVPN1.xml	
	CreateVPN2.xml	
	DeleteSite1.xml	
	DeleteSite2.xml	
	DeleteSite3.xml	
	DeleteSite4.xml	
	DeleteVPN1.xml	
	DeleteVPN2.xml	
	RemoveSite1fromVPN1.xml	
	RemoveSite2fromVPN1.xml	
	RemoveSite3fromVPN1.xml	
	RemoveSite4fromVPN1.xml	
	RemoveSite4fromVPN2.xml	

Total Jobs: 0
Activating: 0
Waiting: 0
Scheduled: 0
System Status: ●

The last message (shown in Figure 4-3) requires user interaction. Enter the name of the router and port as shown in the figure.

Figure 4-3 Interacting with job



Interact with job: VE_CreateSite

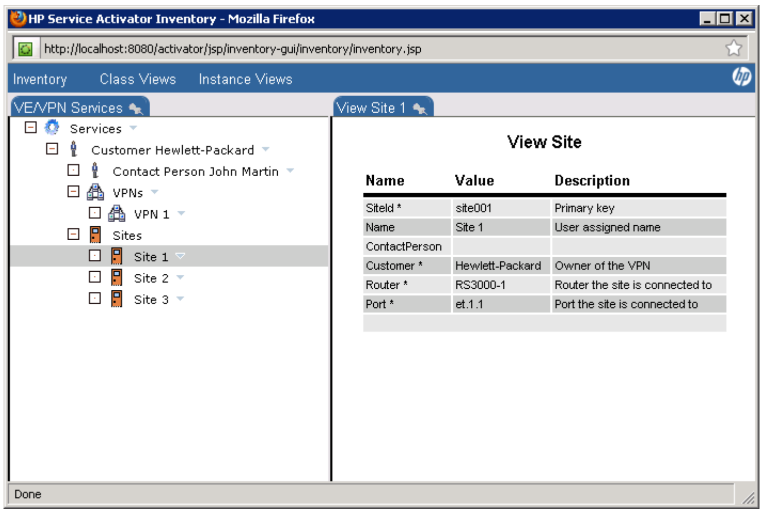
Job Id	Service Id	Host Name	Workflow	Status	Start Time	Post Time	Step	Description
24	site003	MSH4	VE_CreateSite	Running	2010-07-01 09:58:14	2010-07-01 09:58:14	Select_router_and_port	

Router	RS3000-3	*PE router*
Port	et1.3	*Port used on PE router*

HP Service Activator - Mozilla Firefox
http://localhost:8080/activator/interact
Submit
Clear
Done

6. Two screenshots of the inventory UI after the VPN and the three Sites have been created are shown in Figure 4-4 and Figure 4-5.

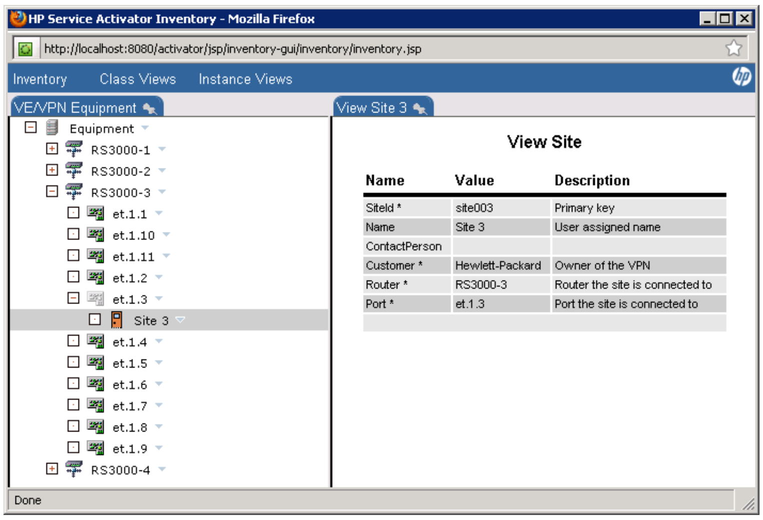
Figure 4-4 Services tree: New VPN Sites


HP Service Activator Inventory - Mozilla Firefox
http://localhost:8080/activator/jsp/inventory-gui/inventory/inventory.jsp
Inventory
Class Views
Instance Views
VE/VPN Services
Services
Customer Hewlett-Packard
Contact Person John Martin
VPNs
VPN 1
Sites
Site 1
Site 2
Site 3
View Site 1
Done

View Site

Name	Value	Description
SiteId *	site001	Primary key
Name	Site 1	User assigned name
ContactPerson		
Customer *	Hewlett-Packard	Owner of the VPN
Router *	RS3000-1	Router the site is connected to
Port *	et.1.1	Port the site is connected to

Figure 4-5 Equipment tree: Site 1 attached to port “et.1.3” on “RS3000-3”

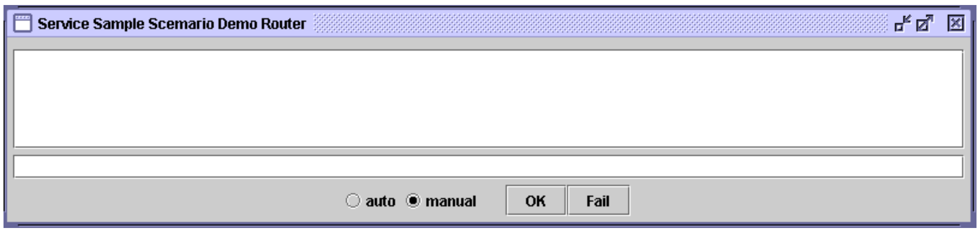


View Site

Name	Value	Description
SiteId *	site003	Primary key
Name	Site 3	User assigned name
ContactPerson		
Customer *	Hewlett-Packard	Owner of the VPN
Router *	RS3000-3	Router the site is connected to
Port *	et.1.3	Port the site is connected to

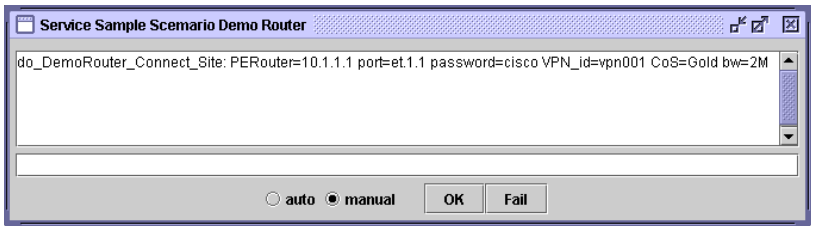
7. The next step is to connect the Sites to “`VPN 1`”. Set the plug-in simulator in the manual position in order to control the result the activation manually as shown in Figure 4-6.

Figure 4-6 DemoRouter simulator



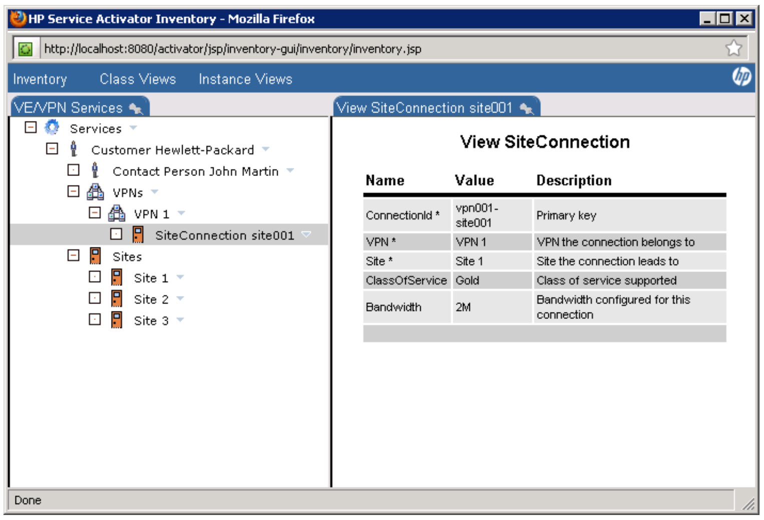
8. Run the test message `AddSite1ToVPN1`. The atomic task is listed in the log area and the user is requested to select the result of the task. For now select the `[OK]` button. Figure 4-7 shows the DemoRouter simulator after an activation command has been received, and Figure 4-8 shows the inventory UI after connecting “`Site 1`” to “`VPN 1`”.

Figure 4-7 **DemoRouter simulator displaying an activation command**


Service Sample Scemario Demo Router

```
do_DemoRouter_Connect_Site: PERouter=10.1.1.1 port=et.1.1 password=cisco VPN_id=vpn001 CoS=Gold bw=2M
```

Figure 4-8 **“`Site 1`” connected to “`VPN 1`”**

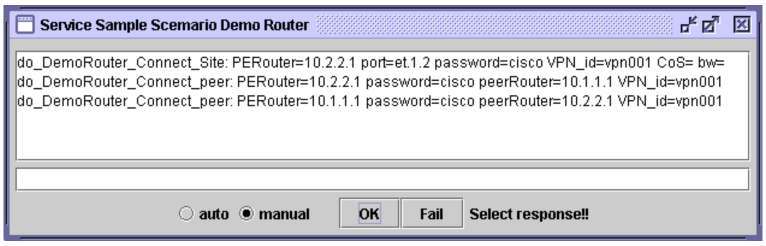

HP Service Activator Inventory - Mozilla Firefox
http://localhost:8080/activator/jsp/inventory-gui/inventory/inventory.jsp
Inventory
Class Views
Instance Views
VE/VPN Services
Services
Customer Hewlett-Packard
Contact Person John Martin
VPNs
VPN 1
SiteConnection site001
Sites
Site 1
Site 2
Site 3
View SiteConnection site001
Done

View SiteConnection

Name	Value	Description
ConnectionId *	vpn001-site001	Primary key
VPN *	VPN 1	VPN the connection belongs to
Site *	Site 1	Site the connection leads to
ClassOfService	Gold	Class of service supported
Bandwidth	2M	Bandwidth configured for this connection

9. Run the test message `AddSite2ToVPN1`. Three atomic tasks are called, one to connect the Site to a PE device and port, and two to create both ends of the pseudo-wire between the two devices. Each task must be terminated by pushing one of the `[OK]`/`[Fail]` buttons. In the example, the `[OK]` button was pushed all three times. Figure 4-9 shows the simulator after processing the two atomic tasks (waiting for the last response).

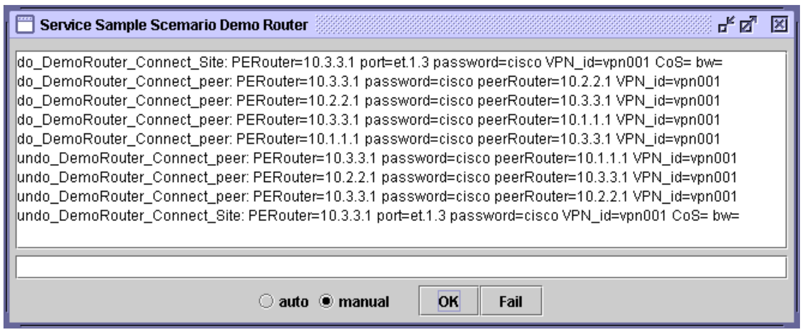
Figure 4-9 **DemoRouter simulator processing atomic tasks**


Service Sample Scemario Demo Router

```
do_DemoRouter_Connect_Site: PERouter=10.2.2.1 port=et.1.2 password=cisco VPN_id=vpn001 CoS= bw=
do_DemoRouter_Connect_peer: PERouter=10.2.2.1 password=cisco peerRouter=10.1.1.1 VPN_id=vpn001
do_DemoRouter_Connect_peer: PERouter=10.1.1.1 password=cisco peerRouter=10.2.2.1 VPN_id=vpn001
```

10. Run the command `AddSite3ToVPN1`. This message is used to demonstrate the rollback feature. “`Site 3`” is connected to the VPN, and five atomic tasks are activated. The first is connecting the Site to the PE device and port, and the next four creates both ends of the two pseudo-wires that connects to the PE devices that already is member of the VPN. In Figure 4-10, all atomic tasks are returned successful except the last one. The failure result causes the undo action to be called, in reverse order, for the already successful tasks.

Figure 4-10 Undo activation commands displayed in plug-in simulator

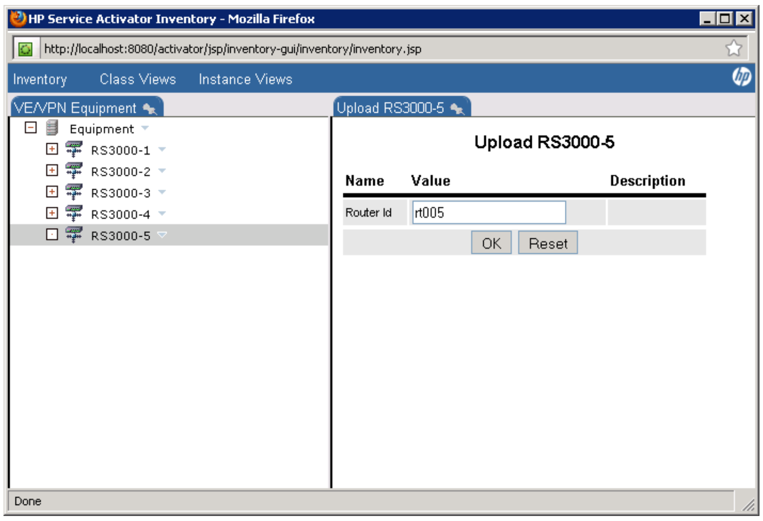


```
do_DemoRouter_Connect_Site: PERouter=10.3.3.1 port=et.1.3 password=cisco VPN_id=vpn001 CoS= bw=
do_DemoRouter_Connect_peer: PERouter=10.3.3.1 password=cisco peerRouter=10.2.2.1 VPN_id=vpn001
do_DemoRouter_Connect_peer: PERouter=10.2.2.1 password=cisco peerRouter=10.3.3.1 VPN_id=vpn001
do_DemoRouter_Connect_peer: PERouter=10.3.3.1 password=cisco peerRouter=10.1.1.1 VPN_id=vpn001
do_DemoRouter_Connect_peer: PERouter=10.1.1.1 password=cisco peerRouter=10.3.3.1 VPN_id=vpn001
undo_DemoRouter_Connect_peer: PERouter=10.3.3.1 password=cisco peerRouter=10.1.1.1 VPN_id=vpn001
undo_DemoRouter_Connect_peer: PERouter=10.2.2.1 password=cisco peerRouter=10.3.3.1 VPN_id=vpn001
undo_DemoRouter_Connect_peer: PERouter=10.3.3.1 password=cisco peerRouter=10.2.2.1 VPN_id=vpn001
undo_DemoRouter_Connect_Site: PERouter=10.3.3.1 port=et.1.3 password=cisco VPN_id=vpn001 CoS= bw=
```

auto [✓ manual]

11. In the last steps, the upload facility is demonstrated. First a new PE device is created by clicking the create icon in the “Equipment” branch. Figure 4-11 shows the inventory UI where the device “`RS3000-5`” has been created and the “Upload” operation has been selected by right-clicking “`RS3000-5`” and selecting `Upload` menu item from the popup menu. Click the `[Submit]` button to launch the `VE_Upload` workflow.

Figure 4-11 Upload ports for new PE device in inventory



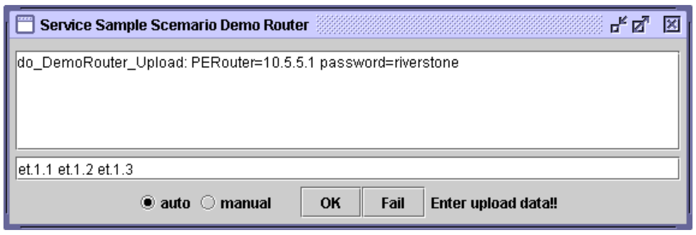
HP Service Activator Inventory - Mozilla Firefox
http://localhost:8080/activator/jsp/inventory-gui/inventory/inventory.jsp
Inventory
Class Views
Instance Views
VE/VPN Equipment
Equipment
RS3000-1
RS3000-2
RS3000-3
RS3000-4
RS3000-5
Upload RS3000-5
Done

Upload RS3000-5

Name	Value	Description
Router Id	rt005	

12. After having launched the VE_Upload workflow from the inventory UI you need to enter ports names (using space as separator) into the DemoRouter simulator as shown Figure 4-12. The uploaded ports displayed in the inventory UI is shown in Figure 4-13.

Figure 4-12 List of port names entered in DemoRouter simulator



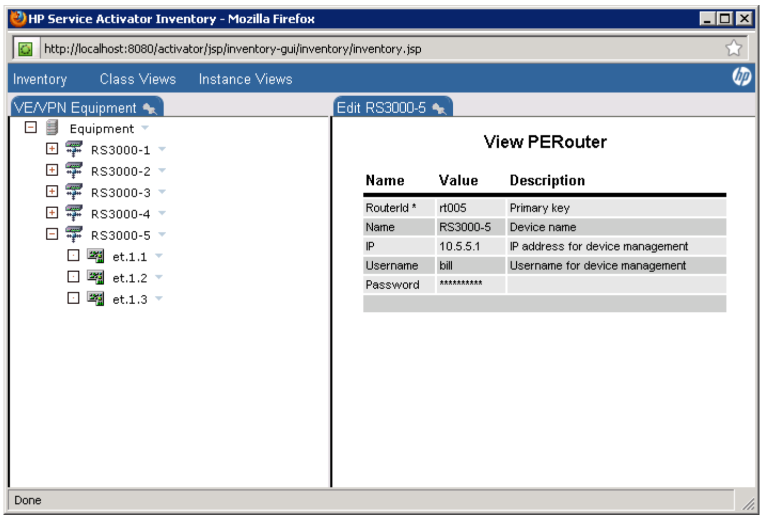
```
do_DemoRouter_Upload: PERouter=10.5.5.1 password=riverstone
```

```
et.1.1 et.1.2 et.1.3
```

[✓ auto] manual

Enter upload data!!

Figure 4-13 Uploaded ports displayed in inventory



View PERouter

Name	Value	Description
RouterId *	rt005	Primary key
Name	RS3000-5	Device name
IP	10.5.5.1	IP address for device management
Username	bill	Username for device management
Password	**********	

5 Uninstalling the Sample Service Scenario

This chapter shows how you can use the Deployment Manager to completely remove the VPN Sample Service Scenario from you HP Service Activator installation.

Undeploying the Scenario using the Deployment Manager

The first step in removing the VPLS Scenario from you Service Activator installation is to undeploy all components in the VPLS Scenario from the Service Activator run-time system. After having undeployed the solution the components in the VPLS Scenario will only be located in the `$ACTIVATOR_OPT/solutions/VPN_Ex` directory.

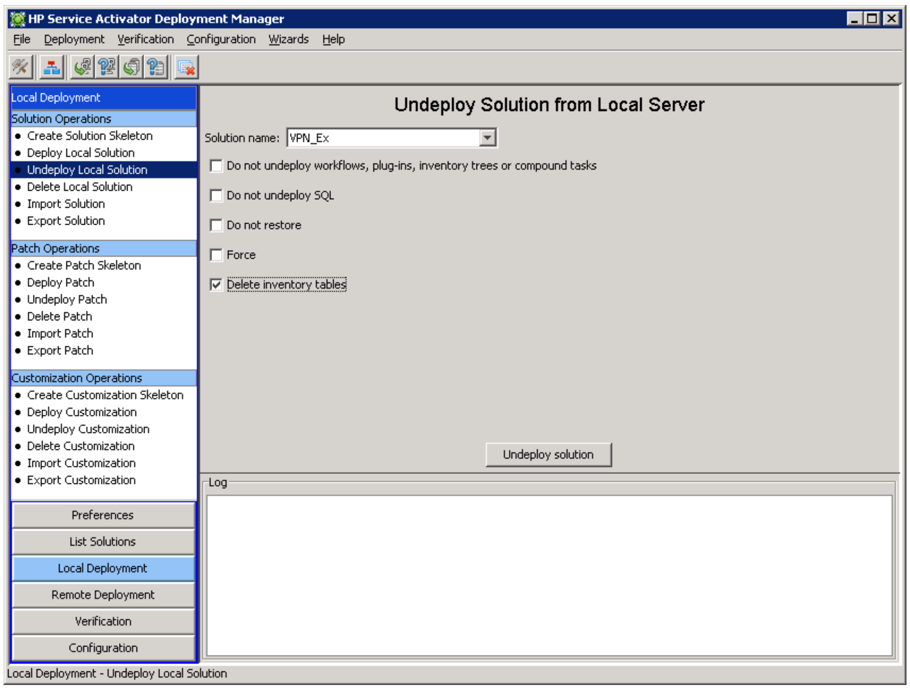
Launch the Deployment Manager by double-clicking the Deployment Manager desktop icon (on Windows) or by typing `deploymentmanager` from a command line prompt.

Upon launching the Deployment Manager, click “Configure Database Connection” under “Preferences” and enter the username and the password for the Service Activator system database.

Now, follow these steps to undeploy the VPLS Scenario (see Figure 5-1):

- Select the “Undeploy Local Solution” operation under “Deployment”.
- Select the “VPN_Ex” from the dropdown list.
- Make sure to select the “Delete inventory tables” checkbox.
- Finally, click the `[Undeploy solution]` button.
- Figure 5-2 shows a screenshot of the Deployment Manager after the VPLS Scenario has been successfully undeployed.

Figure 5-1 **Using Deployment Manager to undeploy the scenario**



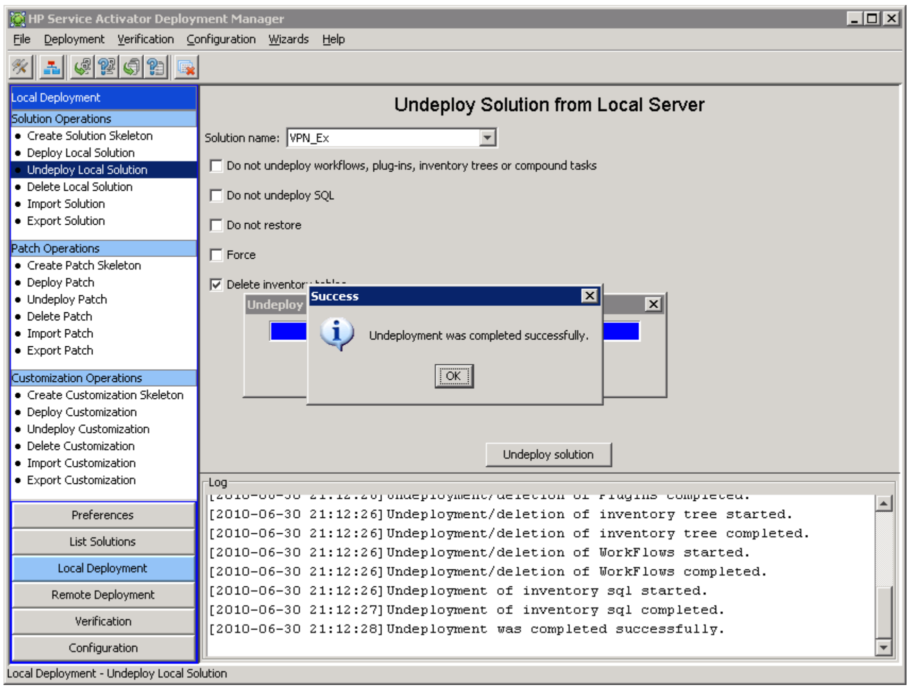
Undeploy Solution from Local Server

Solution name: VPN_Ex

- [ ] Do not undeploy workflows, plug-ins, inventory trees or compound tasks
- [ ] Do not undeploy SQL
- [ ] Do not restore
- [ ] Force
- [x] Delete inventory tables

Local Deployment - Undeploy Local Solution

Figure 5-2 **Scenario successfully undeployed**



Undeploy Solution from Local Server

Solution name: VPN_Ex

- [ ] Do not undeploy workflows, plug-ins, inventory trees or compound tasks
- [ ] Do not undeploy SQL
- [ ] Do not restore
- [ ] Force
- [x] Delete inventor

Undeployment was completed successfully.

```
[2010-06-30 21:12:26]Undeployment/deletion of Plugins completed.
[2010-06-30 21:12:26]Undeployment/deletion of inventory tree started.
[2010-06-30 21:12:26]Undeployment/deletion of inventory tree completed.
[2010-06-30 21:12:26]Undeployment/deletion of WorkFlows started.
[2010-06-30 21:12:26]Undeployment/deletion of WorkFlows completed.
[2010-06-30 21:12:26]Undeployment of inventory sql started.
[2010-06-30 21:12:27]Undeployment of inventory sql completed.
[2010-06-30 21:12:28]Undeployment was completed successfully.
```

Local Deployment - Undeploy Local Solution

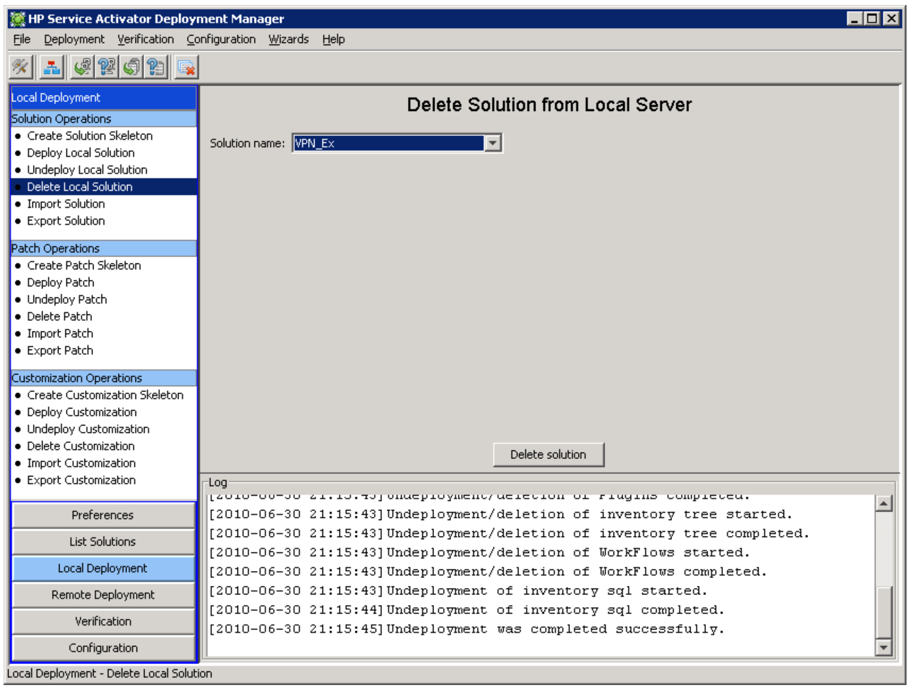
Deleting the Scenario using the Deployment Manager

After you have undeployed the VPLS Scenario you can delete the “VPN_Ex” solution directory using Deployment Manager (or using another other utility of your preference).

Now, follow these steps to delete the VPLS Scenario (see Figure 5-3):

- Select the “Delete Local Solution” operation under “Deployment”.
- Select the “VPN_Ex” from the dropdown list.
- Finally, click the `[Delete solution]` button.

Figure 5-3 Using Deployment Manager to delete the scenario



Solution name: VPN_Ex

```
[2010-06-30 21:15:43]Undeployment/deletion of Plugins completed.
[2010-06-30 21:15:43]Undeployment/deletion of inventory tree started.
[2010-06-30 21:15:43]Undeployment/deletion of inventory tree completed.
[2010-06-30 21:15:43]Undeployment/deletion of WorkFlows started.
[2010-06-30 21:15:43]Undeployment/deletion of WorkFlows completed.
[2010-06-30 21:15:43]Undeployment of inventory sql started.
[2010-06-30 21:15:44]Undeployment of inventory sql completed.
[2010-06-30 21:15:45]Undeployment was completed successfully.
```

NOTE On Windows systems the delete operation will fail if any programs are accessing files or directories in the `$ACTIVATOR_OPT/solutions/VPN_Ex` directory.