

AVP-2934

Introduction to Integrating WTX with DataPower

WTX/DataPower Integration Lab

What this lab is about.....	2
Lab requirements	3
What you should be able to do	3
Lab Startup.....	4
Part 1: Configuring WTX Design Studio for DataPower Integration	5
Part 2: Configuring DataPower for Integration with WTX Design Studio	11
Part 3: Test Design Studio Integration with DataPower.....	25
Part 4: Configuring Design Studio to Upload Maps to DataPower	39
Part 5: Setting Up a MultiCard Project in WTX Design Studio.....	51
Part 6: Configuring DataPower XML Firewall To Use A MultiCard Map.....	58
Part 7: Converting One Input Source to a Local Document.....	102
Part 8: Troubleshooting Invalid Input	108

What this lab is about

Websphere Transformation Extender (WTX) is commonly used to provide message transformations between differing data formats. These transformations (maps) developed using WTX's development environment -- WTX Design Studio -- can be integrated into services running on DataPower devices to complement the rich set of capabilities provided by the DataPower runtime environment.

This lab will work through setting up Design Studio and configuring DataPower to:

- Integrate Design Studio and DataPower:
 - Parts 1 through 4
- Build, test, and troubleshoot a DataPower service which incorporates several examples of WTX maps
 - Parts 5 through 8

Lab requirements

This is the list of system and software required for the attendee to complete the lab. All the necessary components are provided in the prebuilt VMware image you will use for the exercise:

- Websphere Transformation Extender v8.3.0.3
 - Design Studio running in Windows environment
- DataPower XI50
 - Firmware version 3.8.2.1
- Networking infrastructure for communication with DataPower device and Windows
- cURL – the command line utility (program) needed to submit requests to the DataPower service.
- Internet Explorer – browser for accessing the DataPower GUI interface

Each workstation has a two-digit identifier (xx) assigned to it. You will need to substitute that value in a number of places throughout the lab so that all lab attendees can work concurrently; the lab materials will point out where the value needs to be substituted.

What you should be able to do

At the end of this lab you should be able to :-

- Use WTX Design Studio to:
 - Configure communications between Design Studio and DataPower for integrated testing
 - Build and submit WTX maps to the DataPower device to incorporate them into DataPower services
 - Implement and configure a DataPower service (XML Firewall) that uses a WTX map containing multiple inputs and outputs
 - Test and troubleshoot different configuration and runtime issues with the integration steps, and with the implemented service, using both DataPower and WTX Design Studio facilities
-

Lab Startup

Team members from Accelerated Value have started the appropriate VMware image on each of the workstations for the lab activity. If the VMware image is not running, or you have other questions about using VMware, please contact one of the AV team members who will be assisting during the lab.

Please take note of your workstation ID. This is a two-digit number (01 through 20) that you will need to use when configuring various components during the lab activities.

Once the VMware image is started, use the VMWare workstation menu (*VM / Send Ctrl+Alt+Del*) to bring up the login screen, then login to the VMWare image with the following username/password:

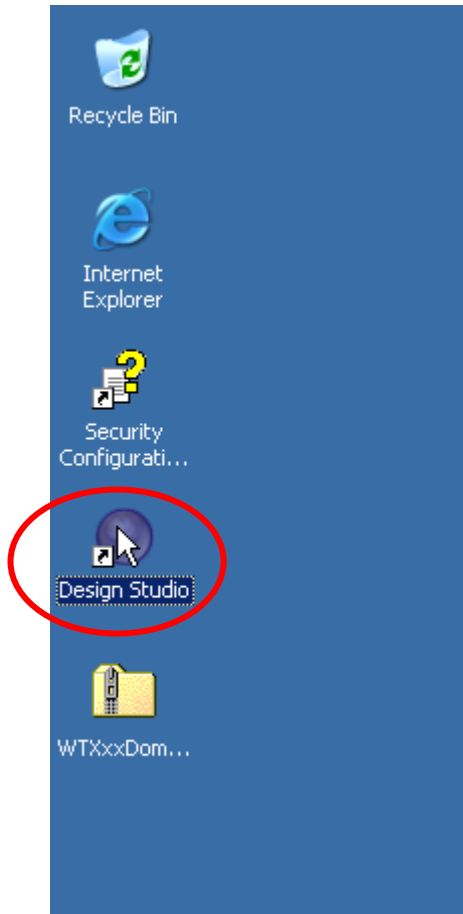
Username : **Administrator**

Password : **Happy2Be**

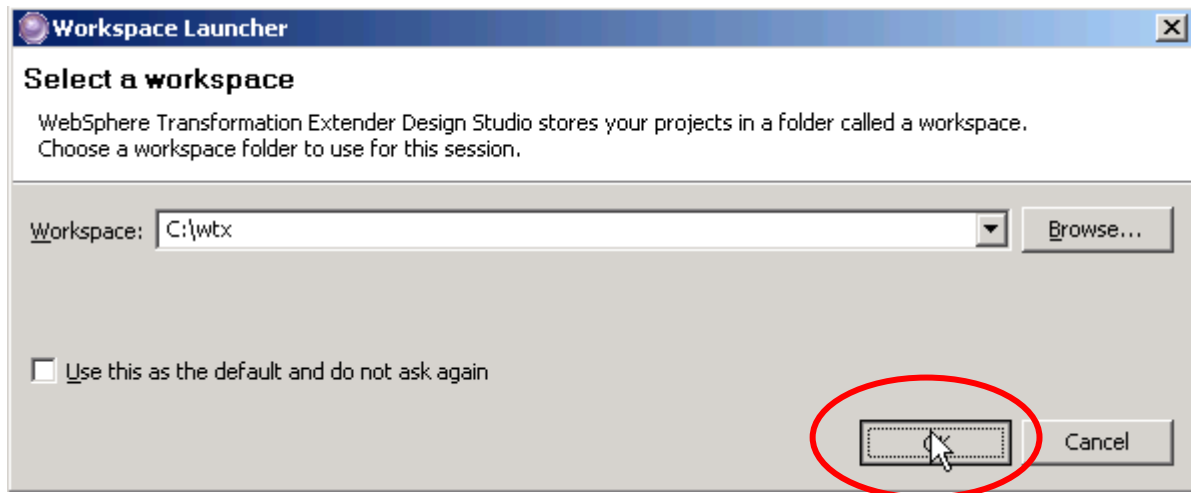
NOTE: Due to the physical memory on the VMware image being used for this lab please understand that certain operations may take time to perform – please be patient.

Part 1: Configuring WTX Design Studio for DataPower Integration

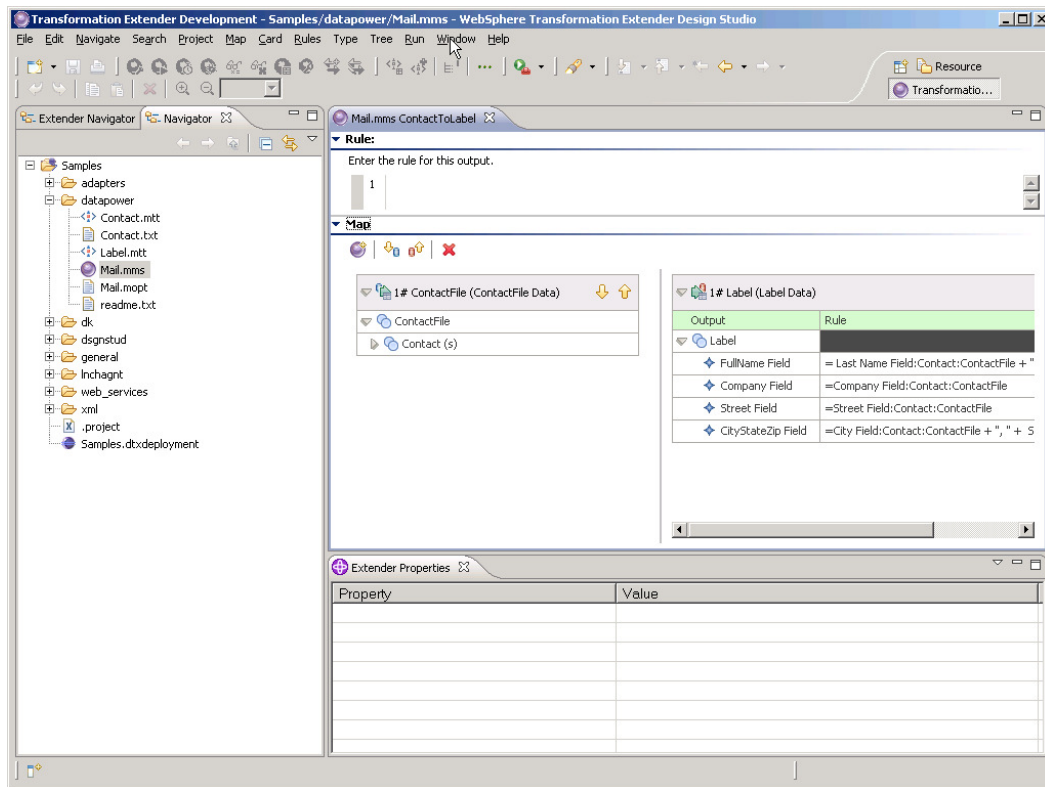
Start WTX Design Studio by double-clicking on the Design Studio desktop icon:



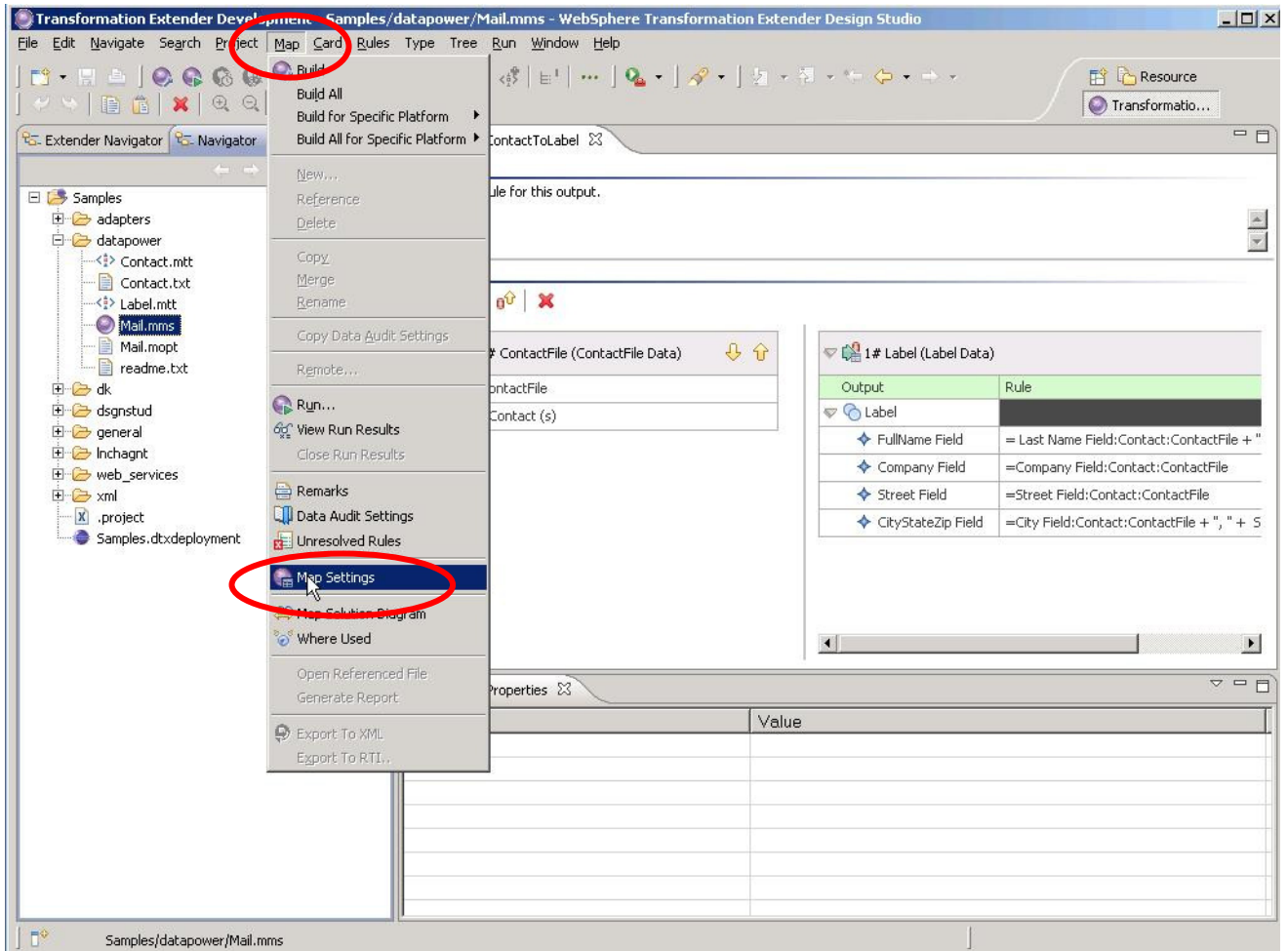
- Accept the workspace location that has been configured:



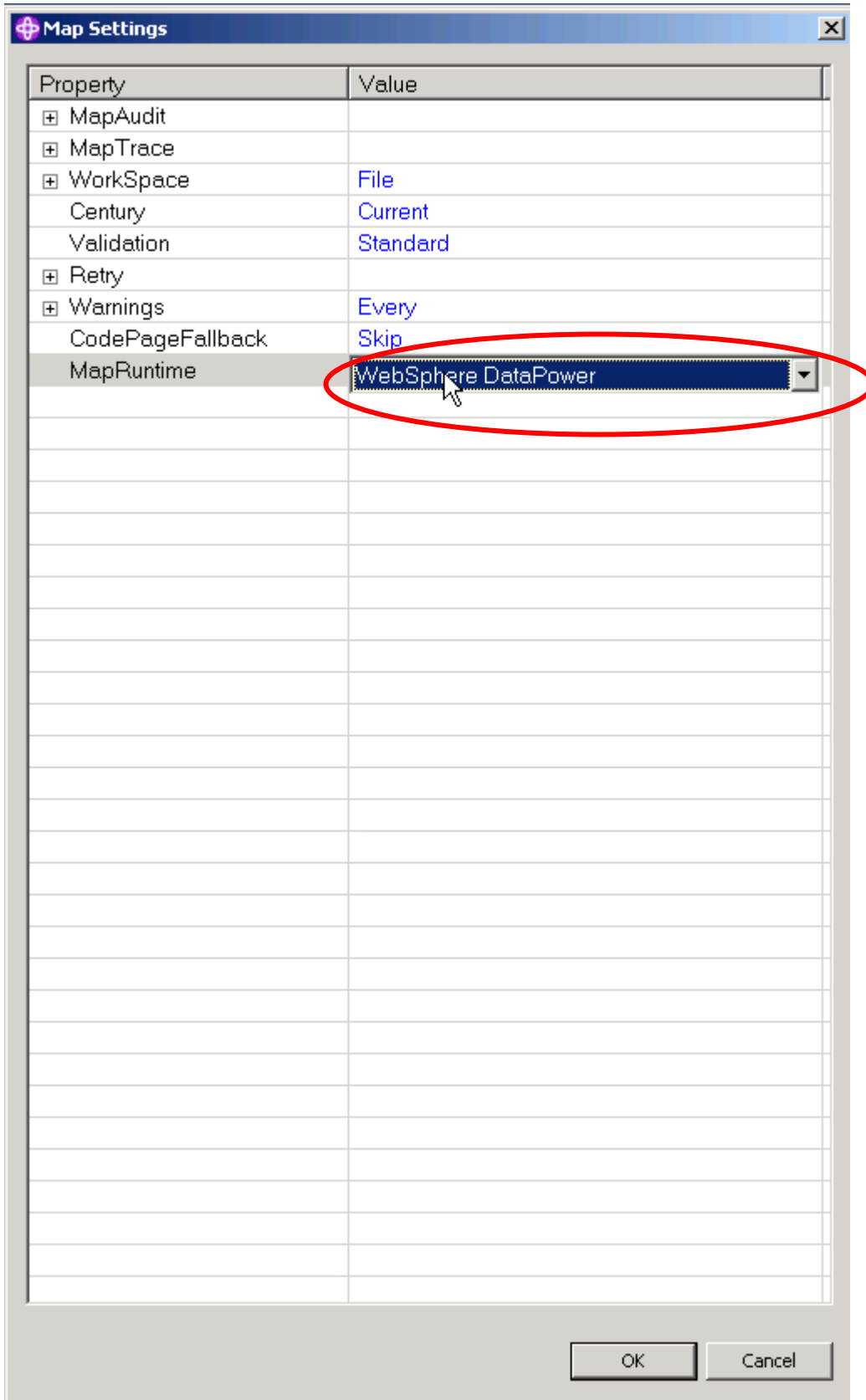
- Design Studio will open to a DataPower sample project that was previously imported:



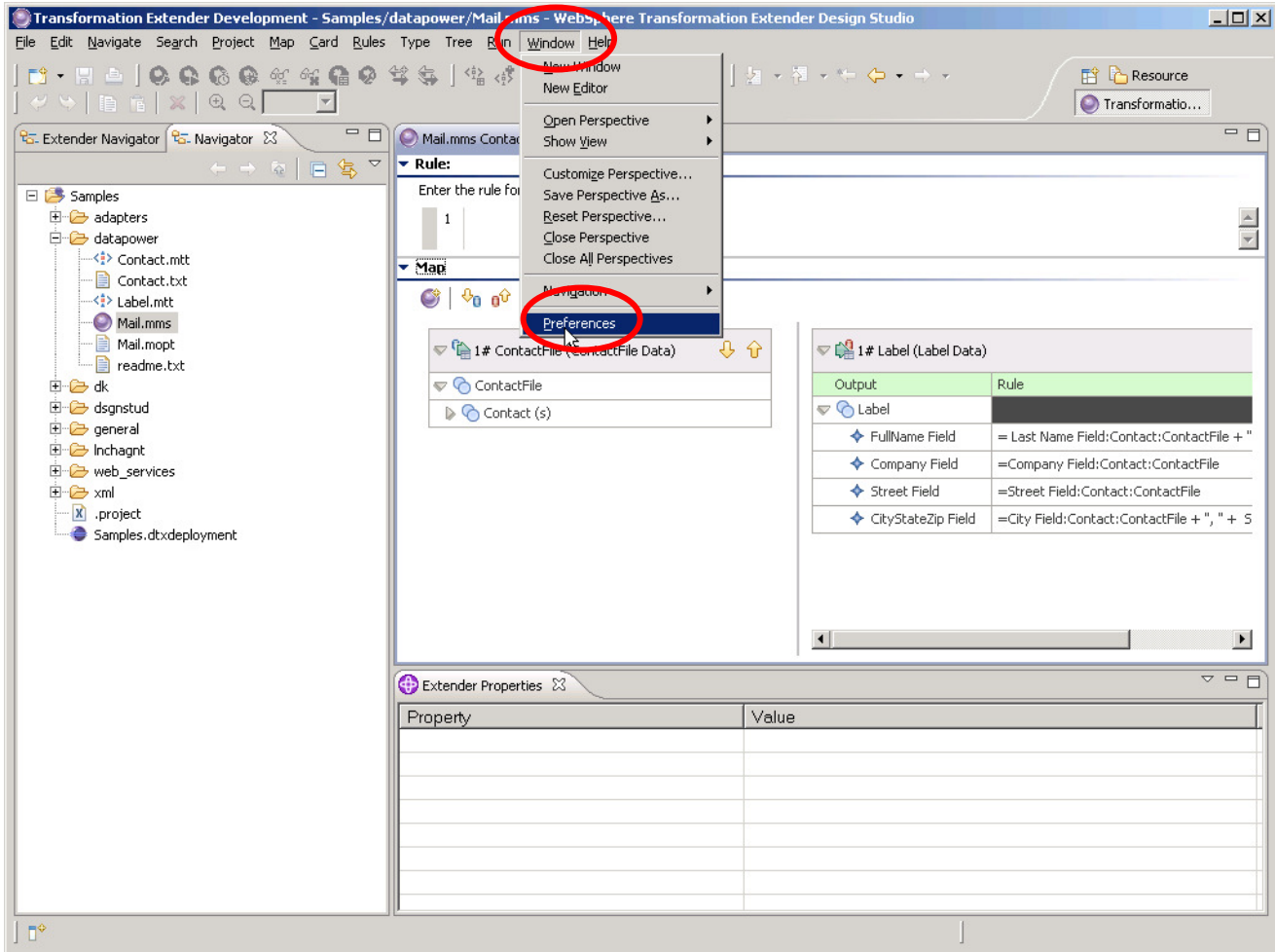
- Check that the Map Settings have been changed to build maps for DataPower:



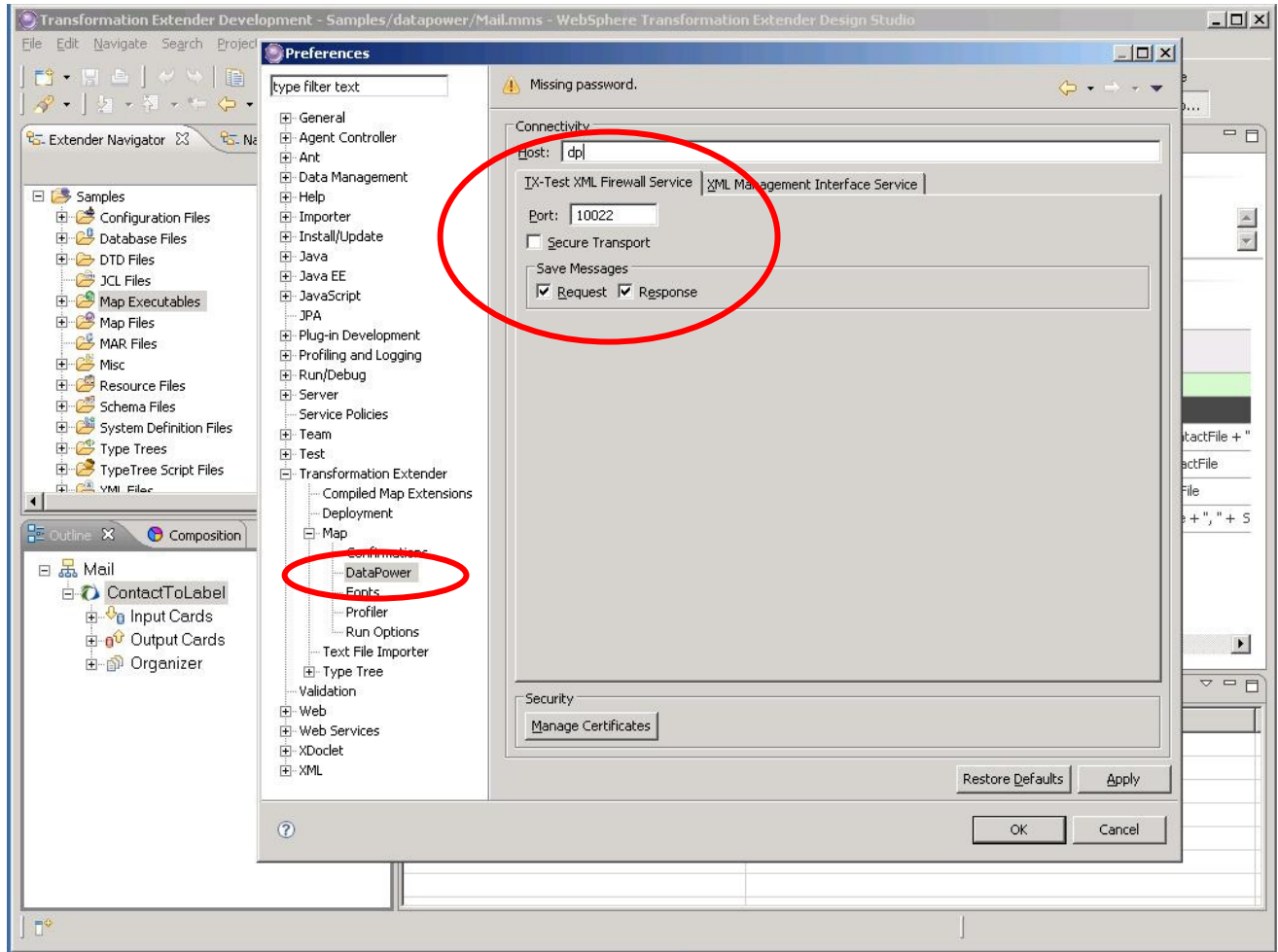
- Ensure that the Map Runtime is set to build maps for DataPower:



- Click OK, then open the Preferences dialog window:



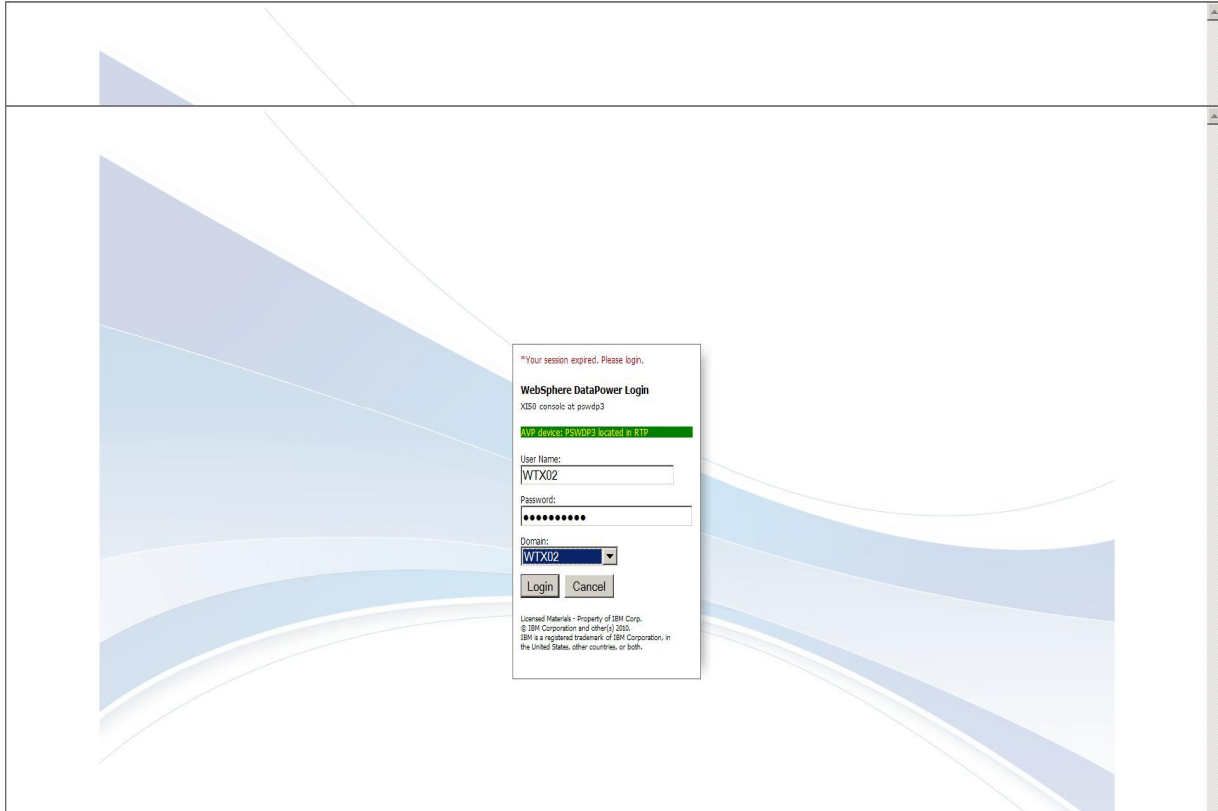
- And then, using the left-hand menu, navigate to Transformation Extender / Map / DataPower. If necessary, click on the tab labelled *TX-Test XML Firewall Service*:



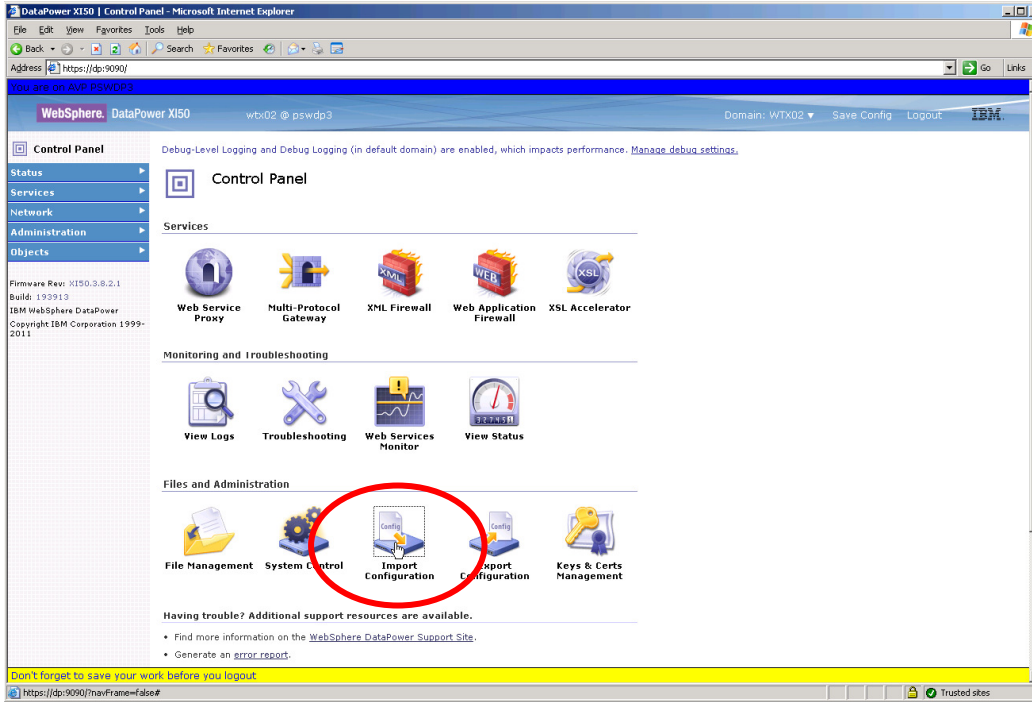
- Enter the host (“*dp*”) and then the port number. Use the port number for this workstation – this will be **8xx2**, where **xx** is the two-digit number that was assigned to this workstation. Ensure that ‘Secure Transport’ checkbox is **unchecked**, and **check** both the Request and Response checkboxes under Save Messages. Click OK.
- Next, you will set up and configure the DataPower services to allow integrated testing from Design Studio. Minimize Design Studio while you configure the DataPower device.

Part 2: Configuring DataPower for Integration with WTX Design Studio

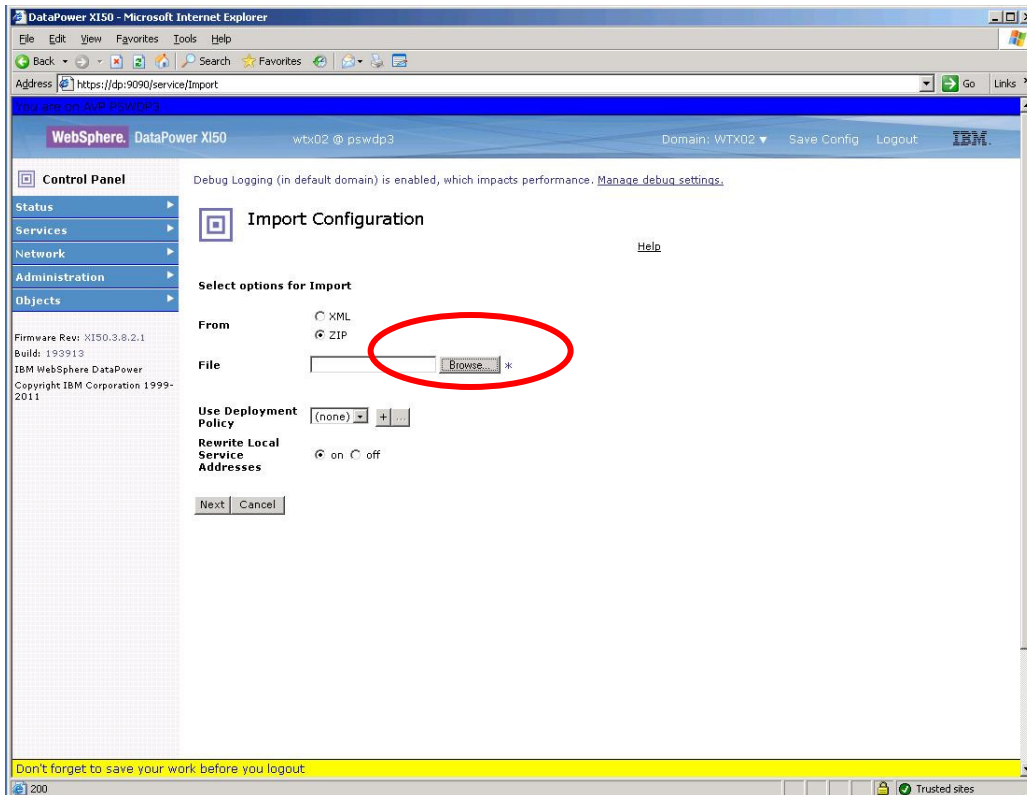
- Start Internet Explorer from the desktop. You may get a message warning you about the certificate; you can ignore this and continue on.
- The DataPower login screen should appear. [if not, the url is *https://dp:9080/*] Log into this workstation's domain (**WTXxx**), using the userid (**WTXxx**) and password provided for this workstation (**WTXxxxx**):



- The main window of the DataPower GUI will paint. Click on *Import Configuration*:

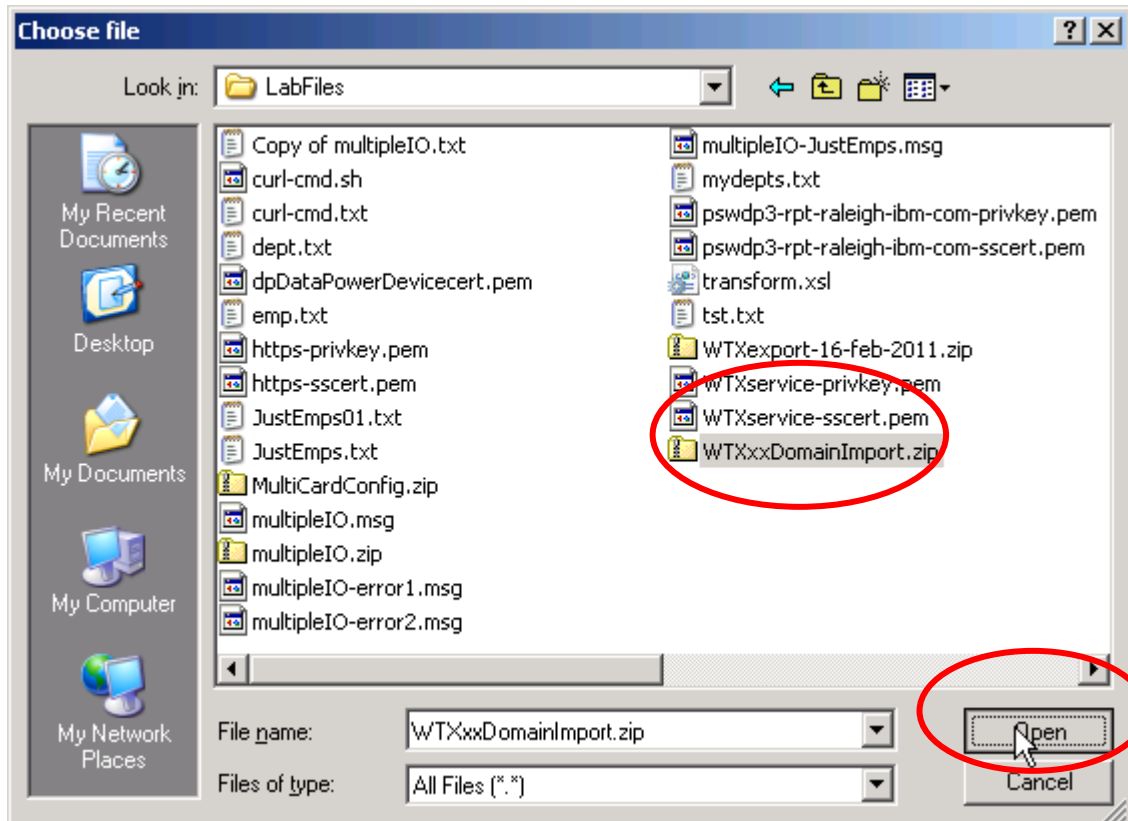


- And then the *Browse* button on the next page:

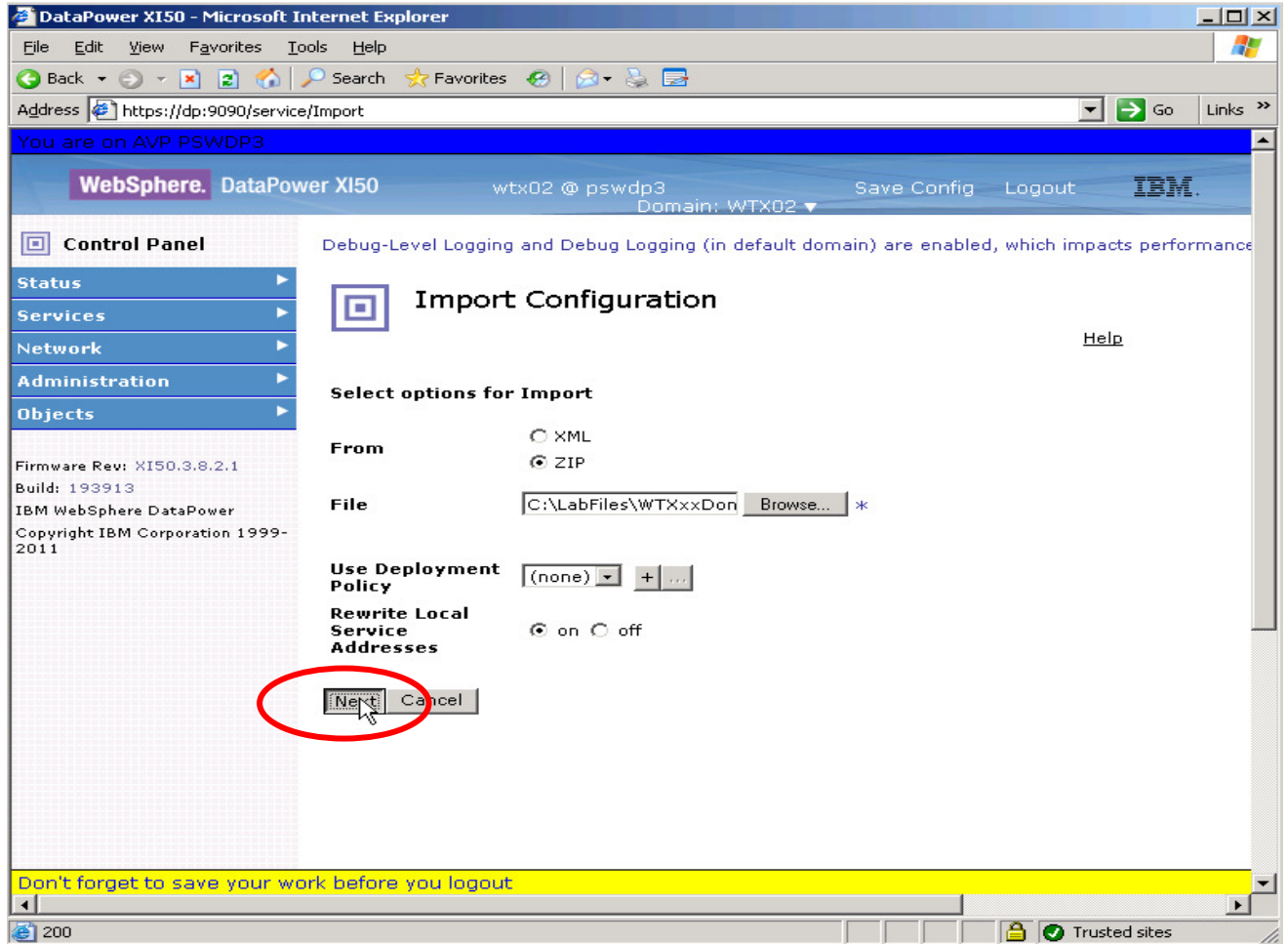


- Select and open the zip file *c:\LabFiles\WTXxxDomainImport.zip*. This zip file is just a [slightly modified] export of the domain setup that is provided on the DataPower Resource CD. Typically, an administrator would set up a single *WTX* domain on a development DataPower

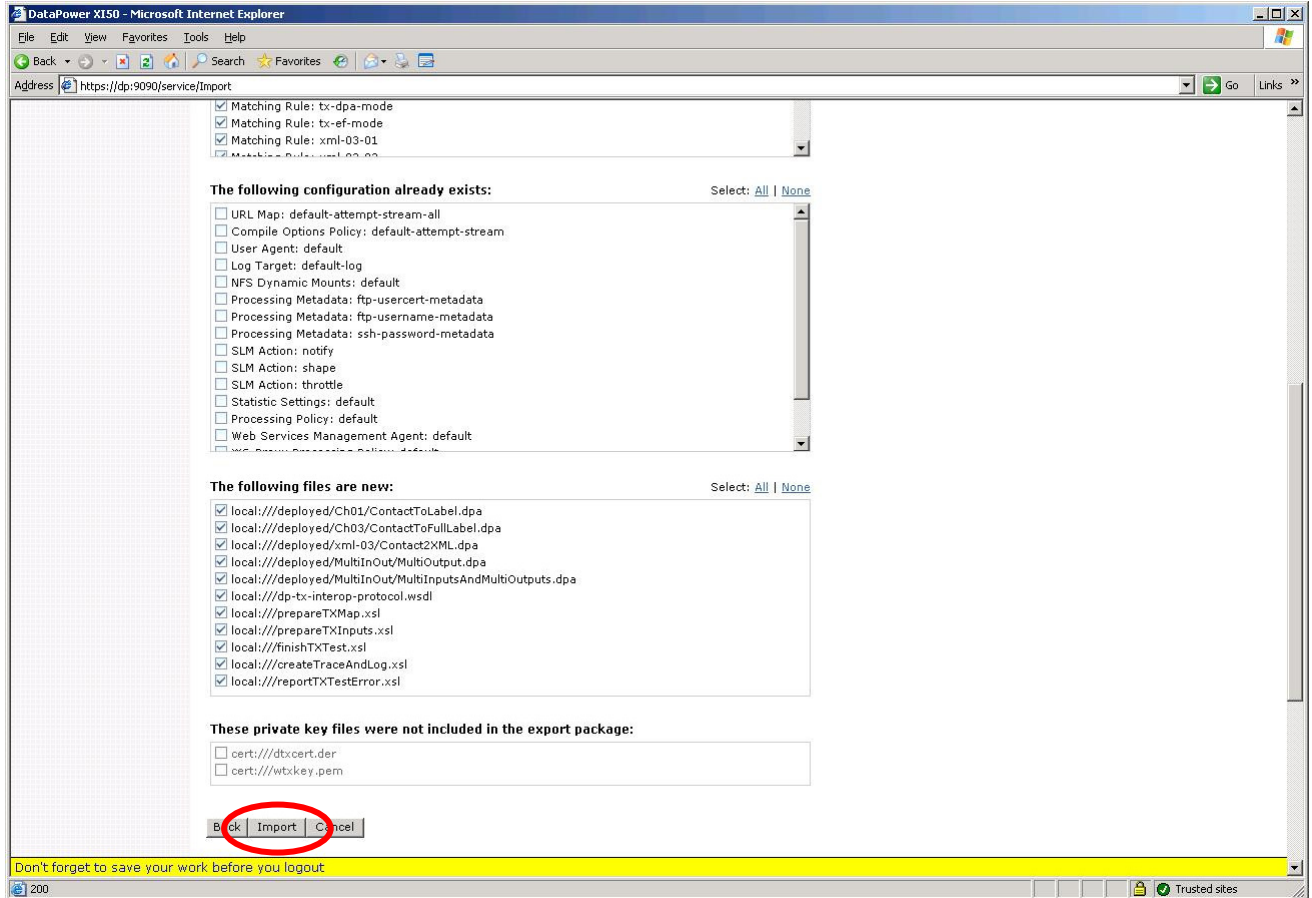
device which would serve the needs of any number of developers. For this lab exercise, you will be importing a slightly modified version. The zip file you will import permits each person working through the lab to setup and configure these services.

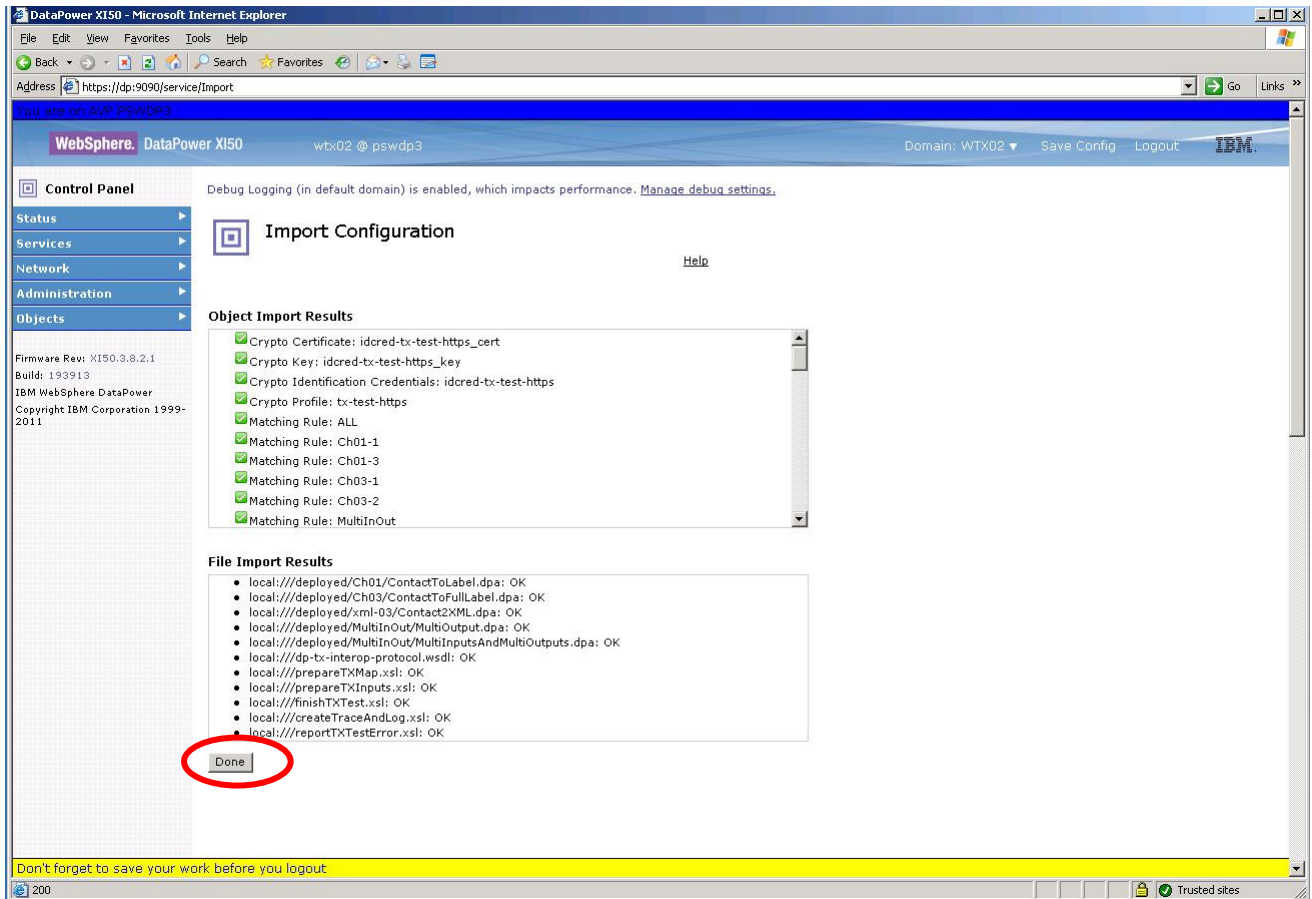


- Then click Next on the following screen:

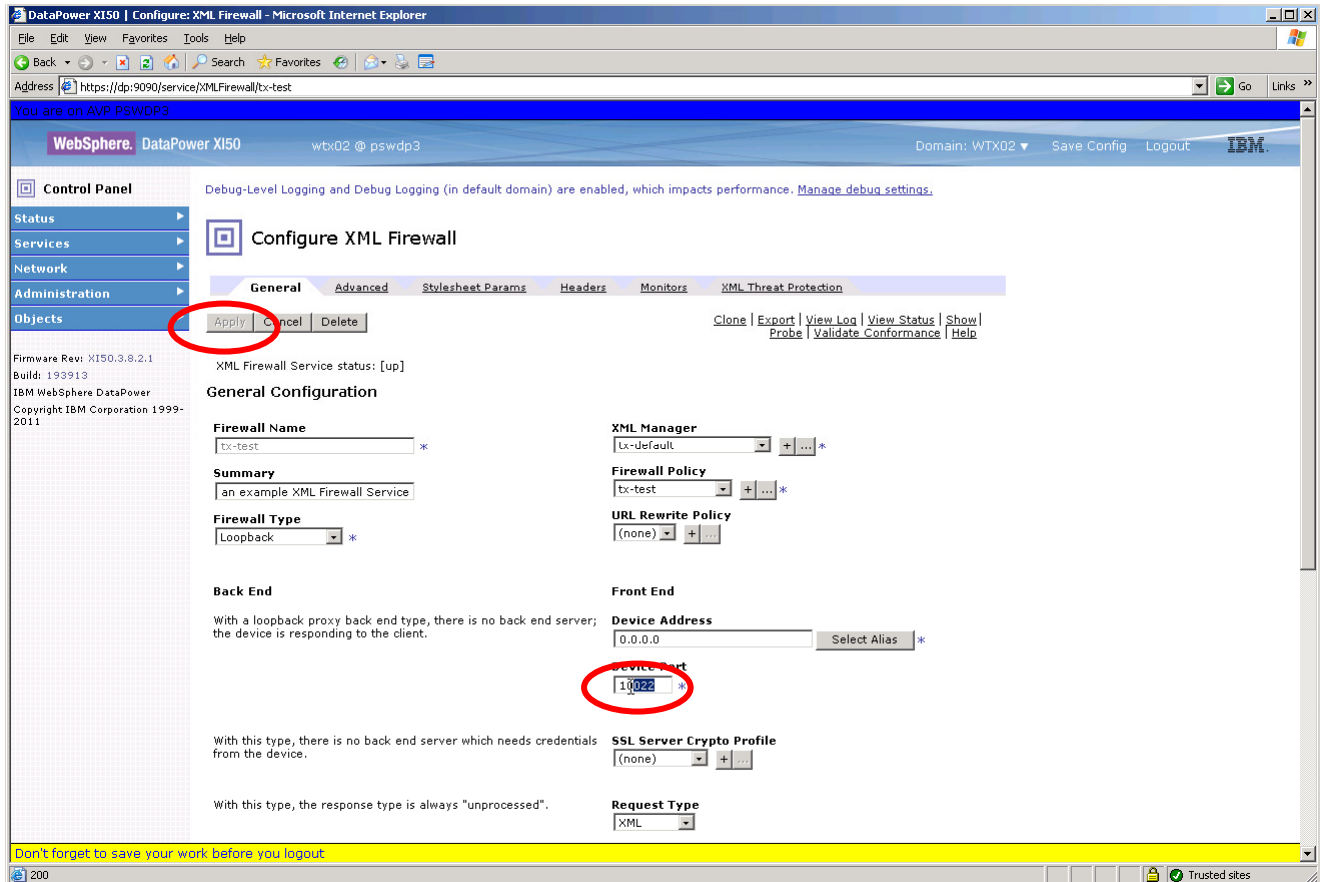


- The following screen shows all the objects selected from the zip file for import, with all the new objects and artifacts pre-selected. Select the 'Import' button, and then 'Done' on the following screen:





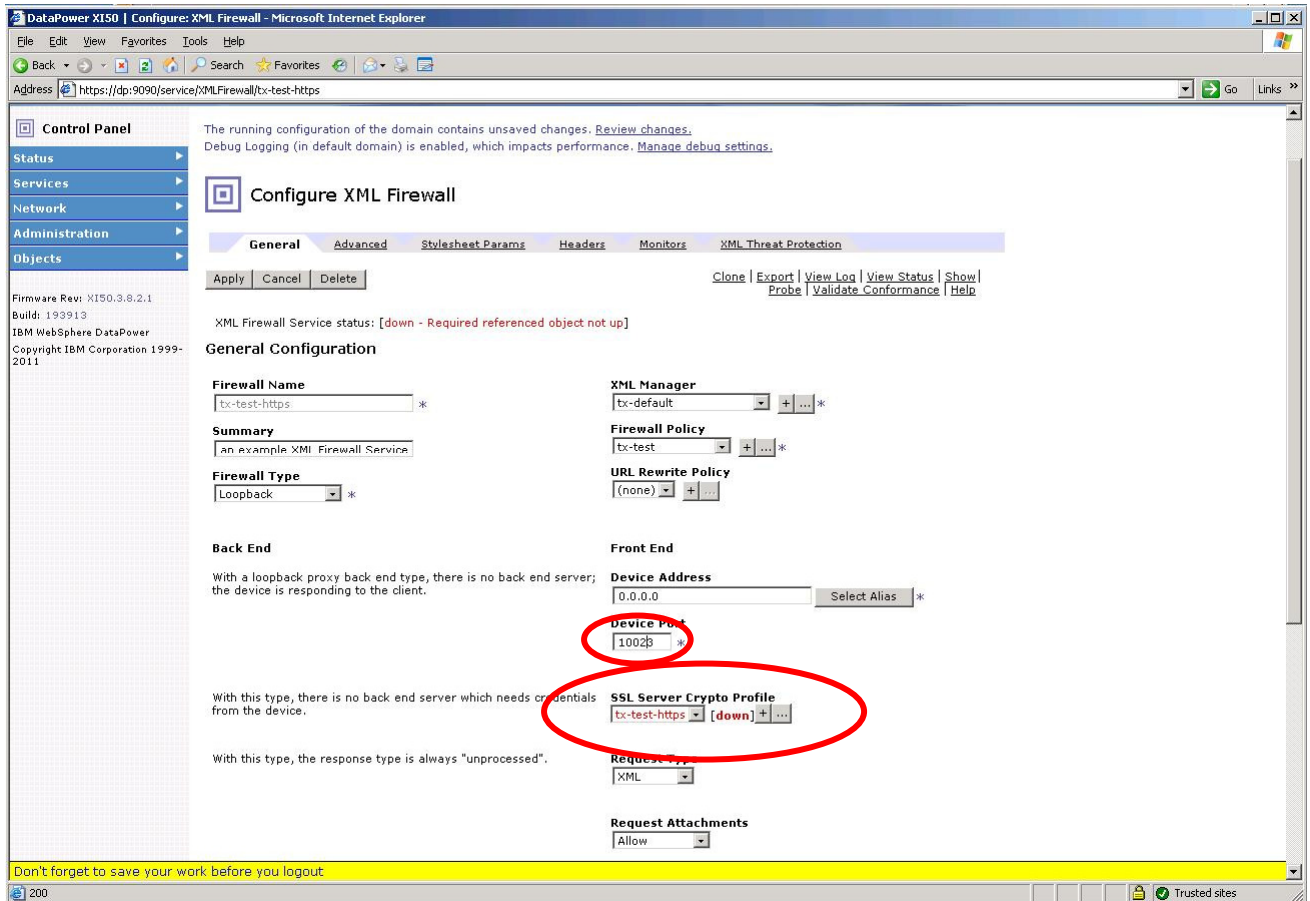
- After you have imported this configuration file, the main Control Panel should display. Click on the *XML Firewall Wizard* icon. The next screen to appear lists the three imported XML Firewall services from the zip file. However, you will see that all three services are in a **down** state, since there are other services on the device that have already allocated the same ports on which the new services that you have just added are configured to listen. So let's fix that up! You will correct all three services in turn while we're here. First, click on the *tx-test* service to bring up the configuration screen for that service.
- Change the *Device Port* to **8xx2** (where xx == your workstation ID):
 - **NOTE:** This value must match the value you configured in Design Studio in Part 1!



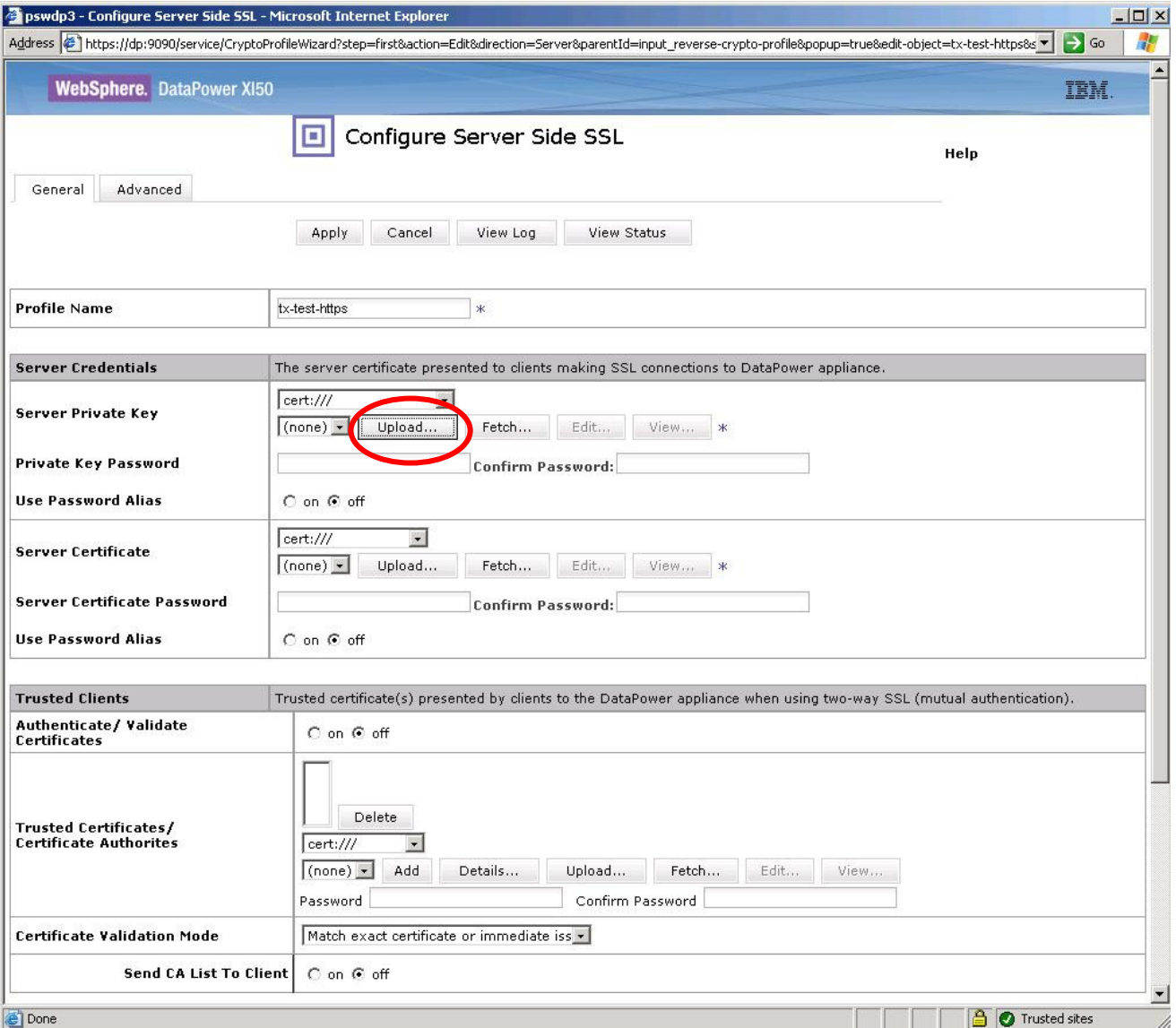
- Click the *Apply* Button (which will be enabled once you change the port number).

THE REST OF PART 2 IS OPTIONAL -- SKIP THE REST OF THIS SECTION FOR NOW AND PROCEED TO NEXT PART OF THE LAB (PART 3 ON PAGE 25). YOU MAY RETURN TO THIS SECTION AFTER COMPLETING THE LAB IF TIME PERMITS.

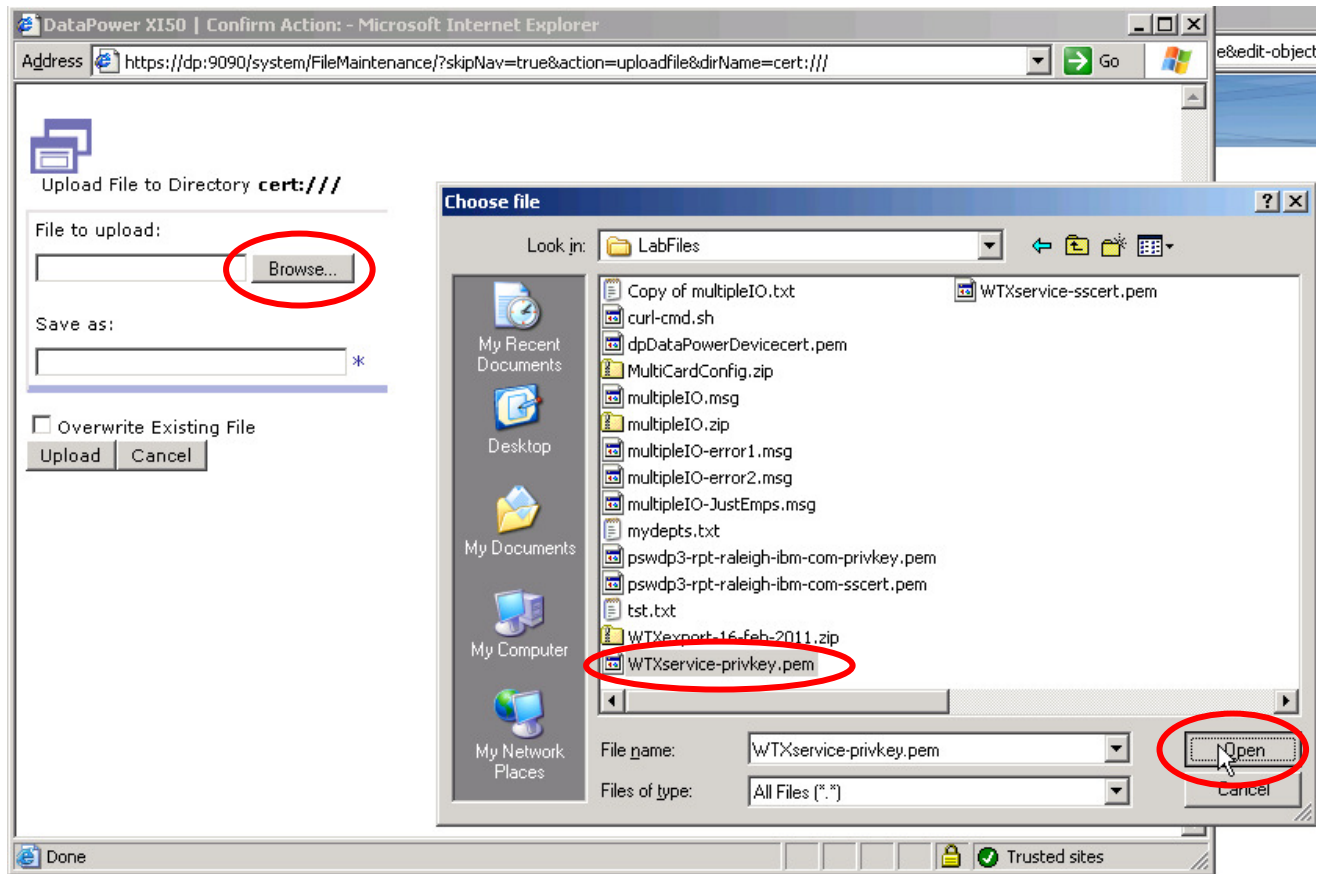
- Now, return to the list of *XML Firewall* services (from the main page click *Control Panel*, then *XML Firewall*). Repeat these steps for the *deployed-tx-test* XML Firewall, but this time specify port **8xx4**.
- Next, you will set up the XML Firewall service which accepts https connections from Design Studio. This service is named *tx-test-https*. Return to the list of *XML Firewall* services as you did for the previous two services, then click on the *tx-test-https* service. The first configuration change you will make is similar to the changes we made to the first two firewalls – change the *Device Port* to **8xx3**:

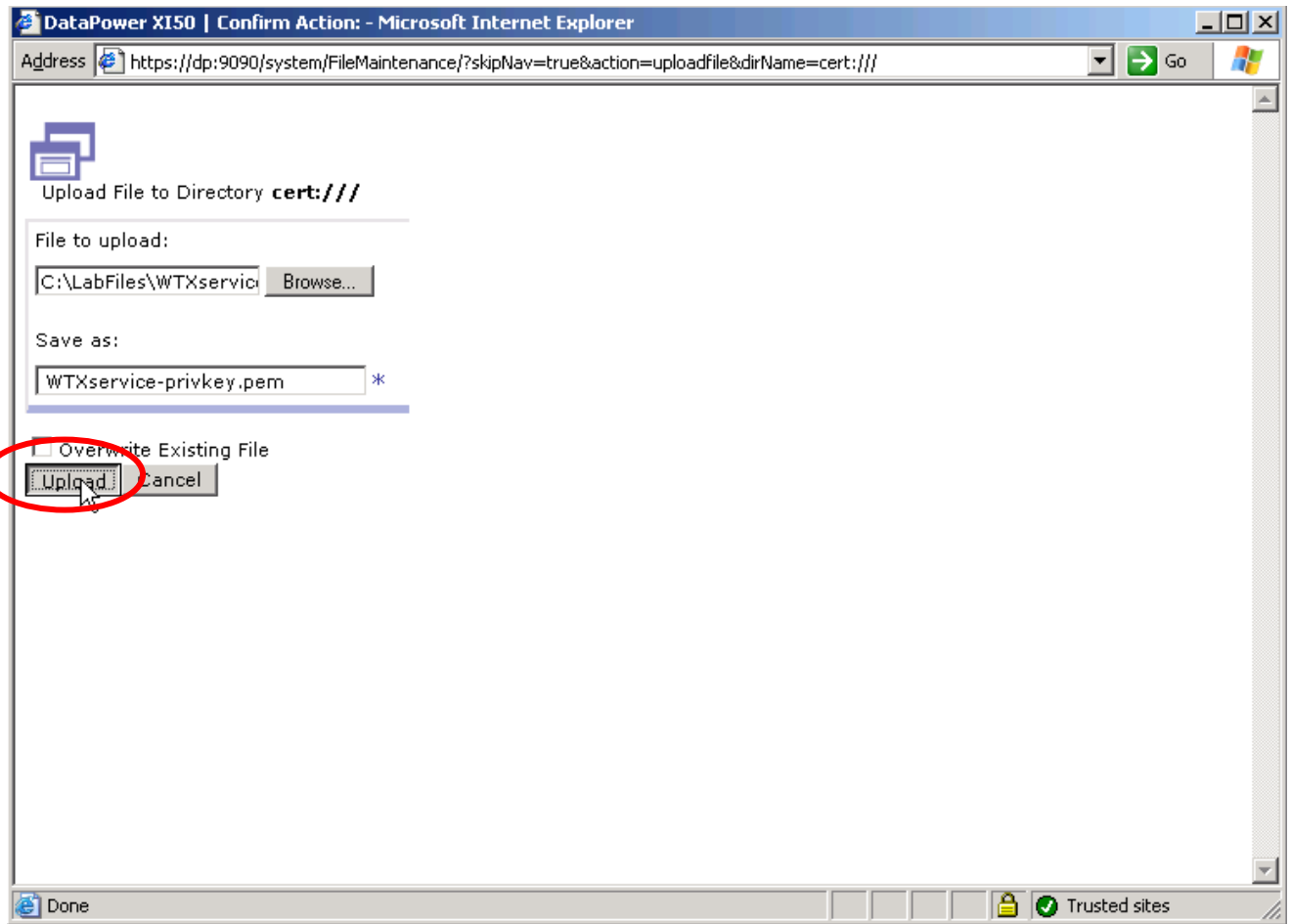


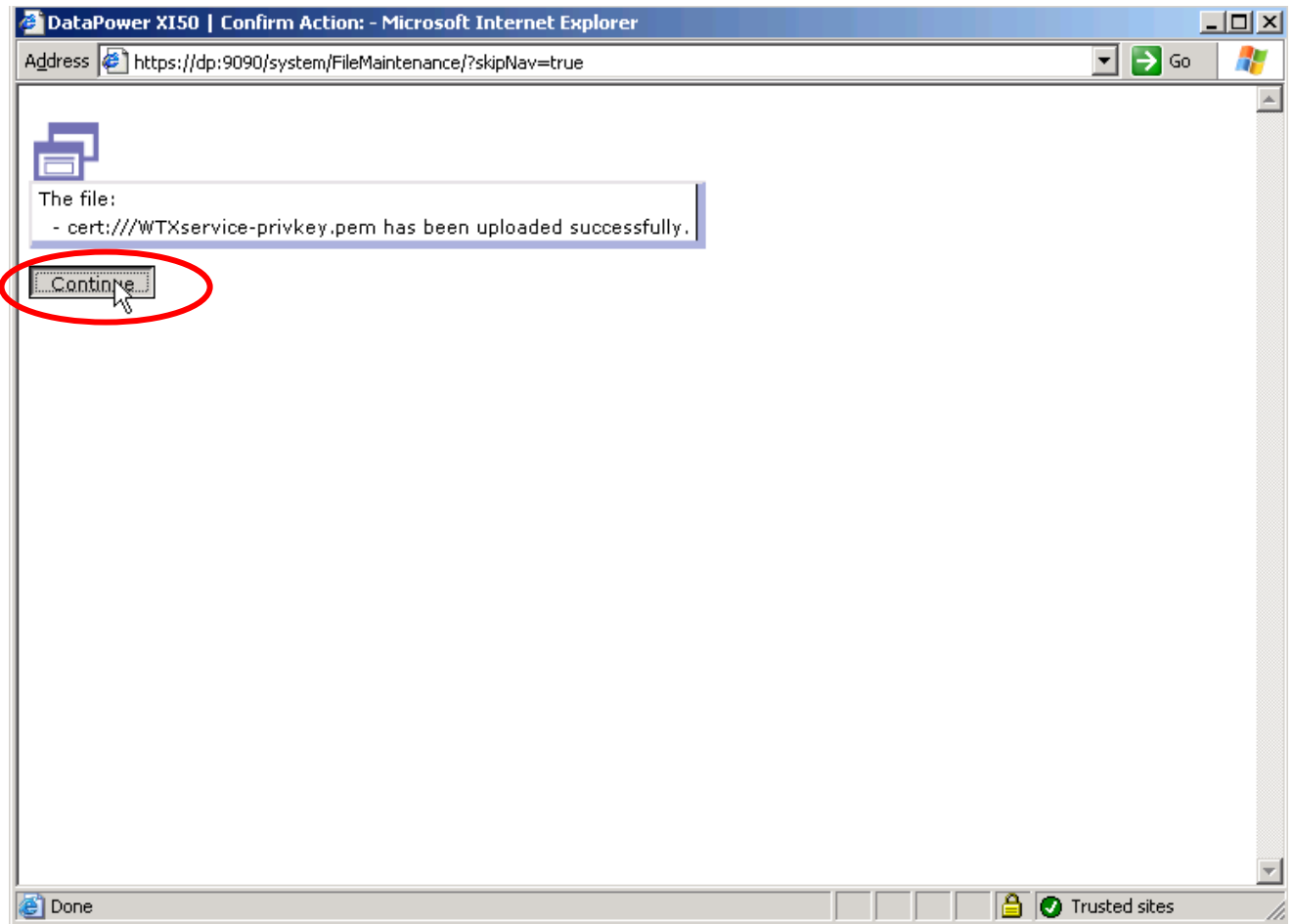
- Note, however, that the object *SSL Server Crypto Profile* is also **down**. You will now correct the configuration for that object as well. Click on the edit button ('...') to the right of the *SSL Server Crypto Profile* object *tx-test-https*. The configuration screen for the *Crypto Profile* object displays:



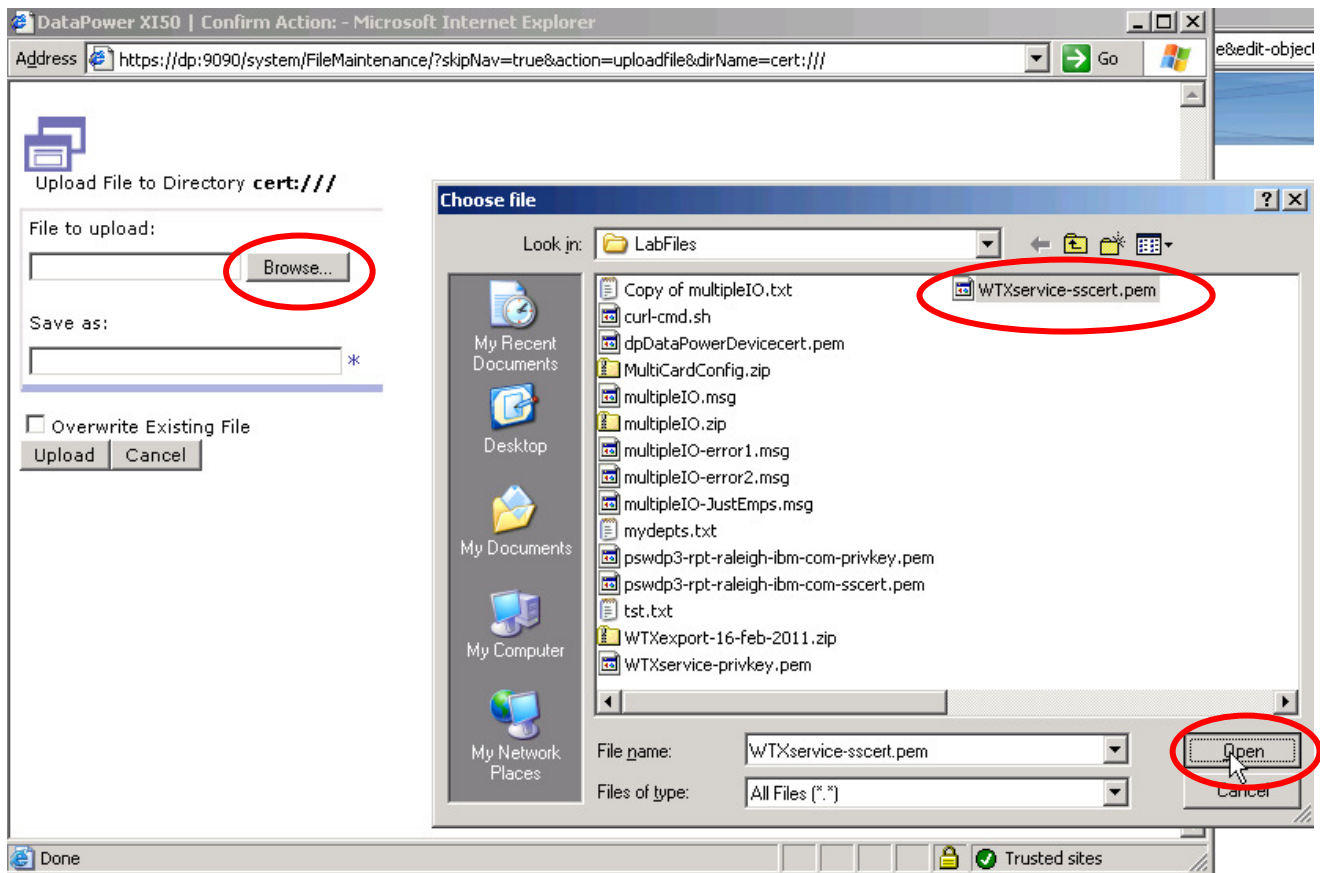
- Click on the *Upload...* button to the right of the drop-down list to the right of *Server Private Key*, and then browse to select the file *c:\LabFiles\WTXservice-privkey.pem*. Open the file, then press the *Upload* button on the next page to complete the upload. On the confirmation dialog that appears, press *Continue*.



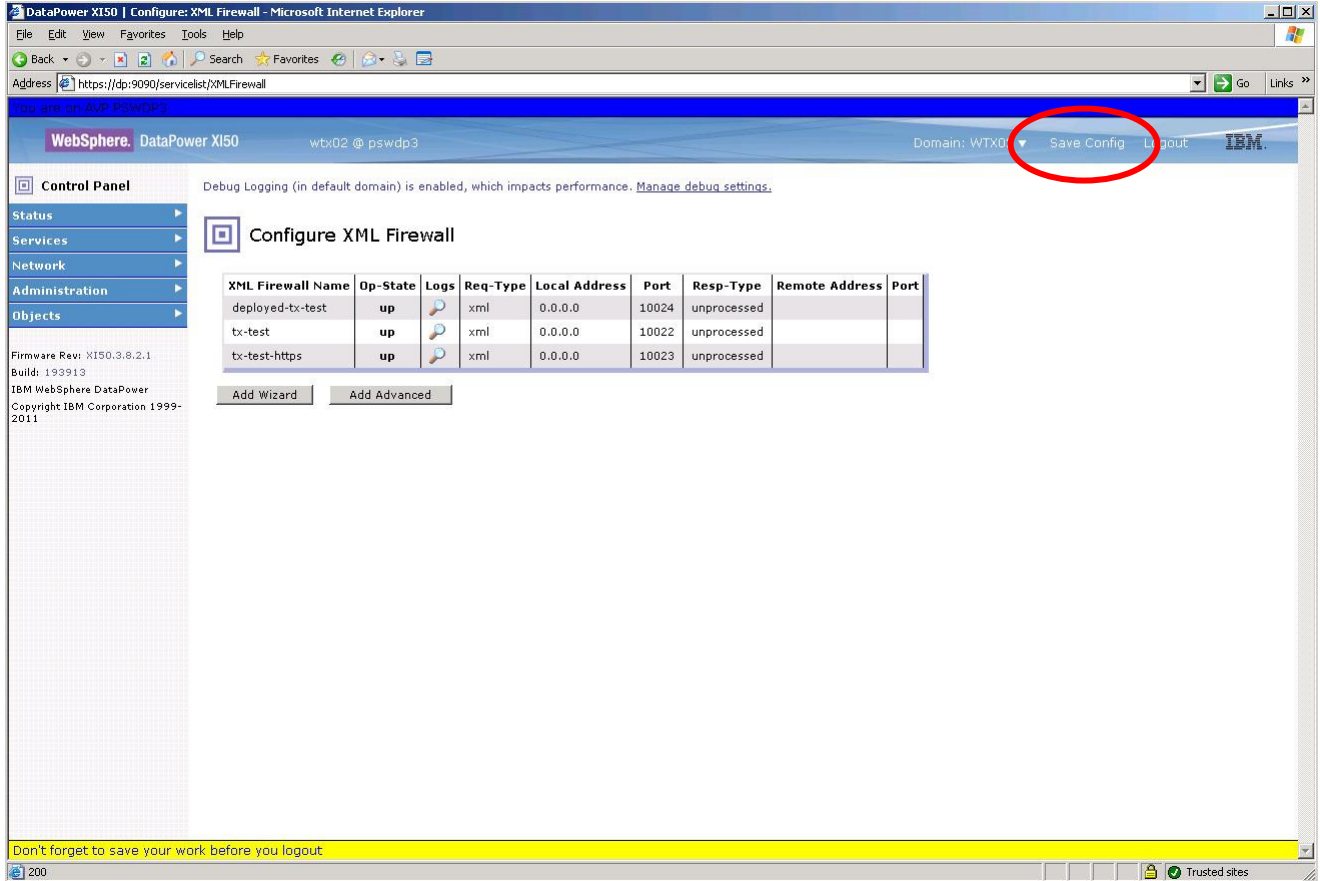




- Repeat the steps for the *Server Certificate*, selecting the file *c:\Labfiles\WTXservice-sscert.pem*:



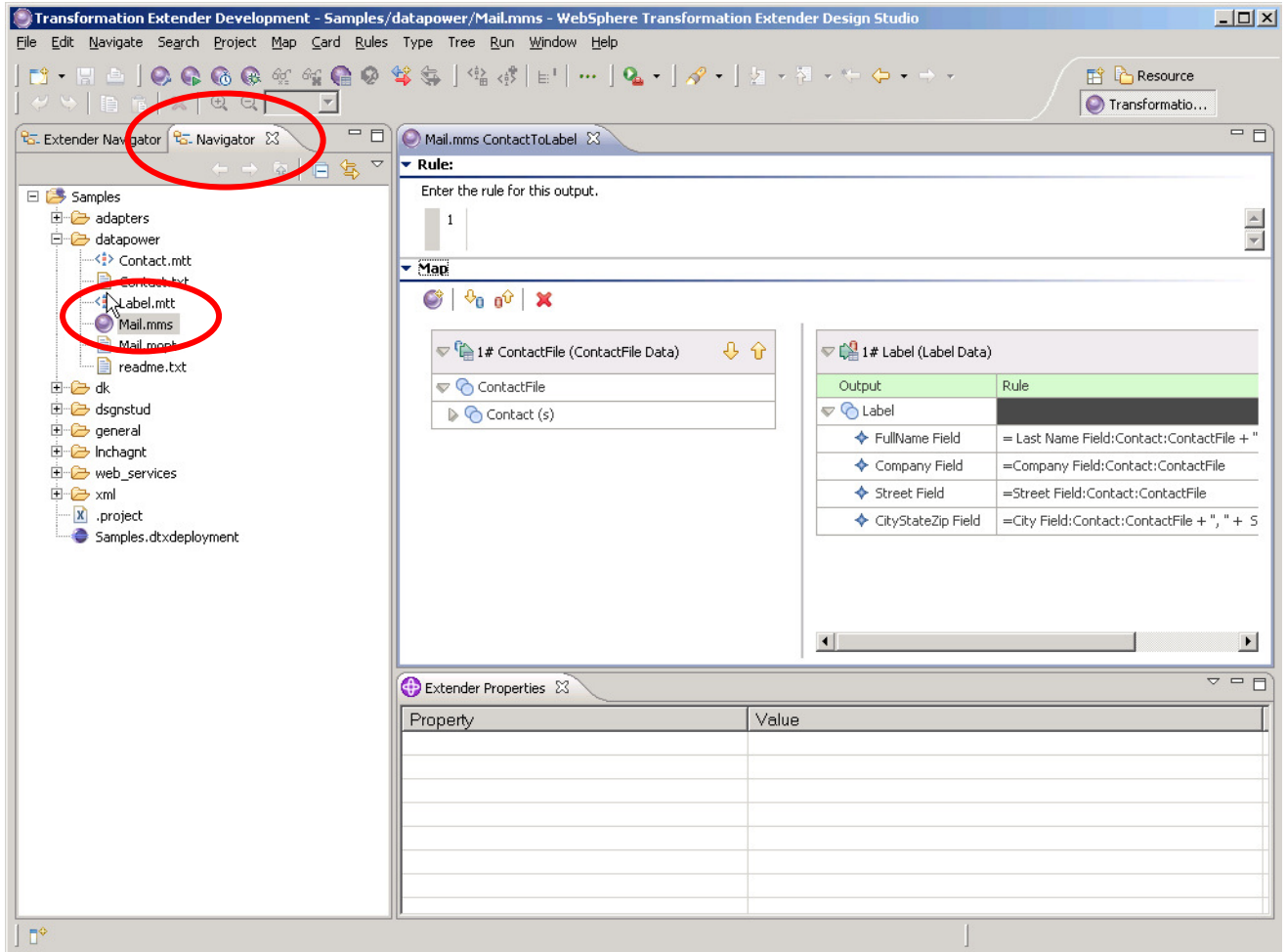
- Save your changes to the Crypto Profile object by clicking the *Apply* button, then click *Apply* again on the XML Firewall configuration screen. The list of the three XML Firewalls should reappear, and all three services should now be in Op-State *up*.



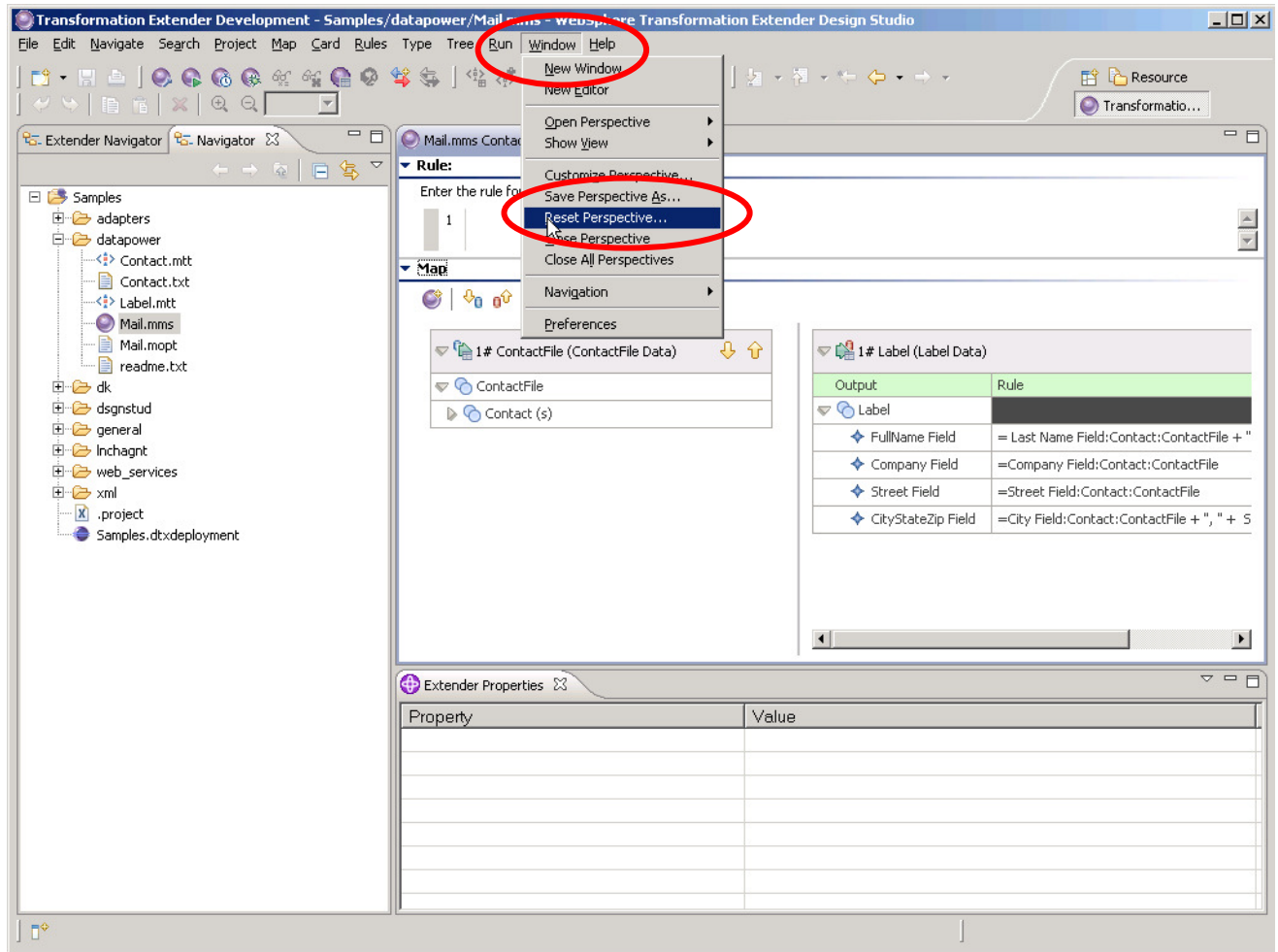
- Save your configuration by clicking on the *Save Config* menu choice (in the blue header section). You have now completed the configuration steps on the DataPower device. Minimize Internet Explorer, and return to your WTX Design Studio session.

Part 3: Test Design Studio Integration with DataPower

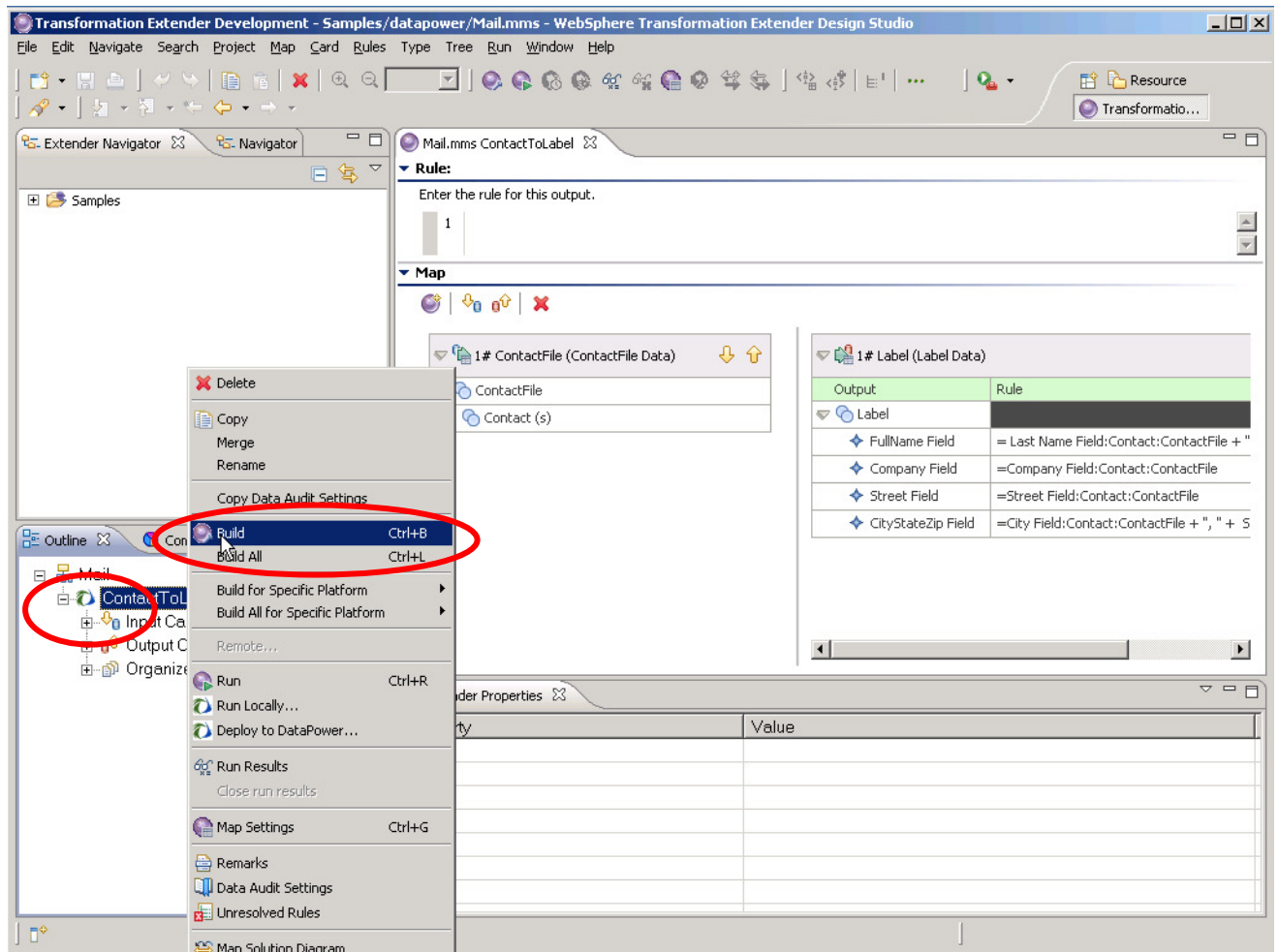
From the *Navigator* tab, click the '+' menus to open *Samples / datapower*, then double-click on the map *Mail.mms*:



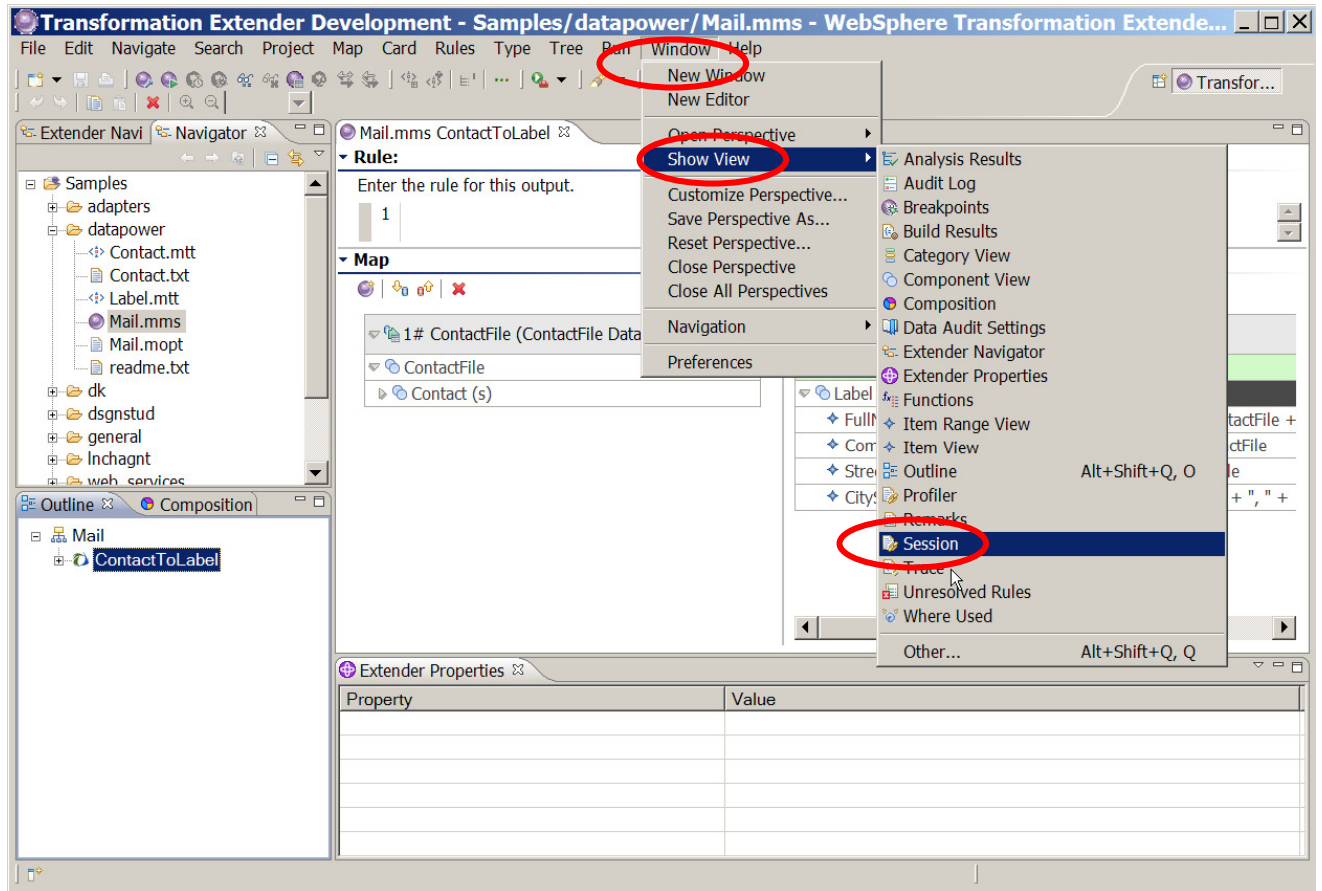
- Now, Use the *Reset Perspective* menu option under the *Window* menu to open the map pane:



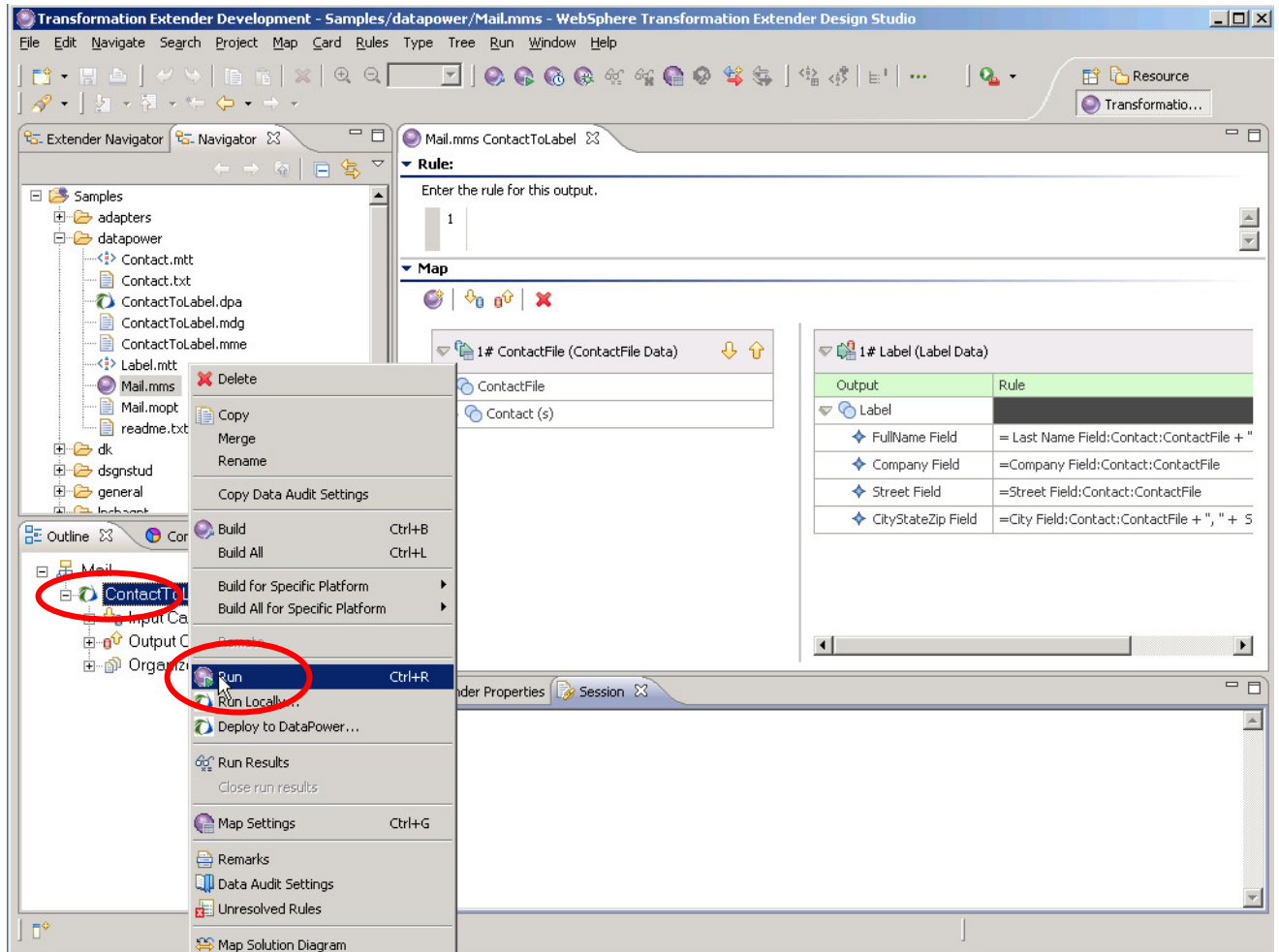
- From the *Outline* tab in the left-hand that appears as a result, expand *Mail* and then right-click on the *ContactToLabel* map. Select *Build* to build the map targeted for DataPower:



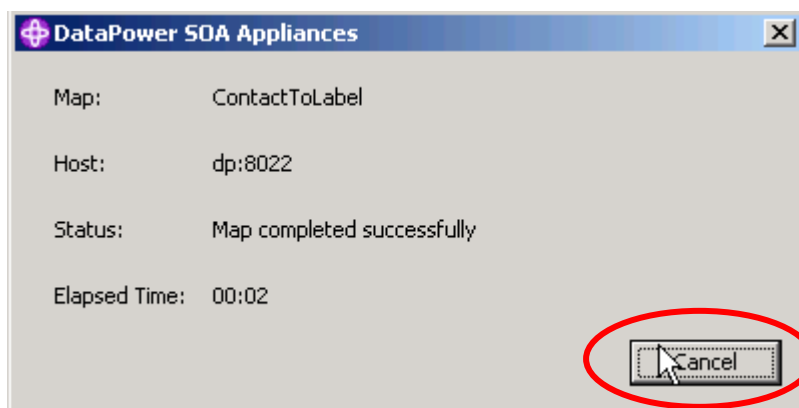
- Note that the icon representing the map gives you a visual indication that this map has been compiled for a DataPower runtime.
- Now, you will use the integrated facilities of Design Studio to run the map on DataPower. First, you will open the session view so you can see the results of the test that you will run shortly. Choose *Window / Show View / Session* from the menu:



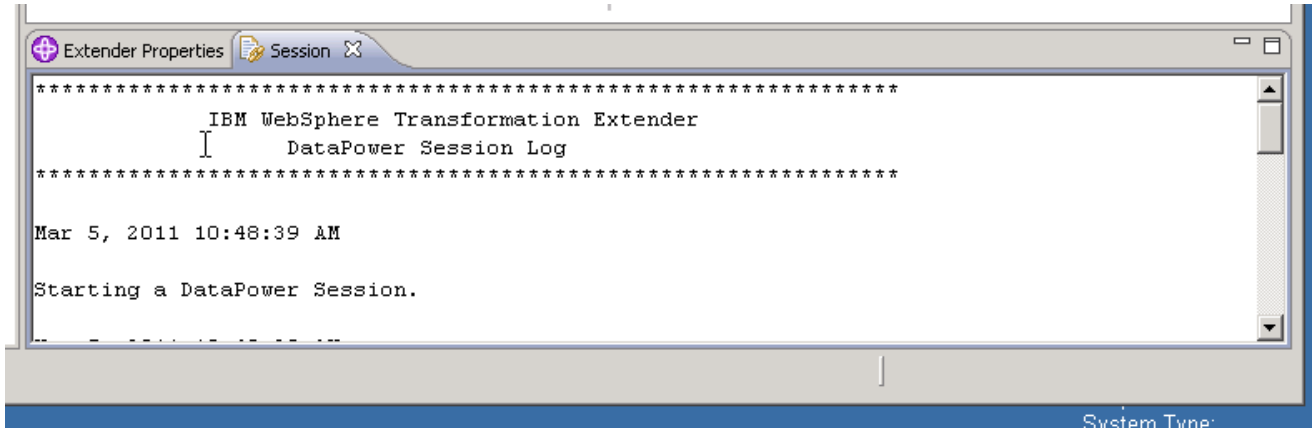
- Now, right-click on the map *ContactToLabel* and select *Run*:



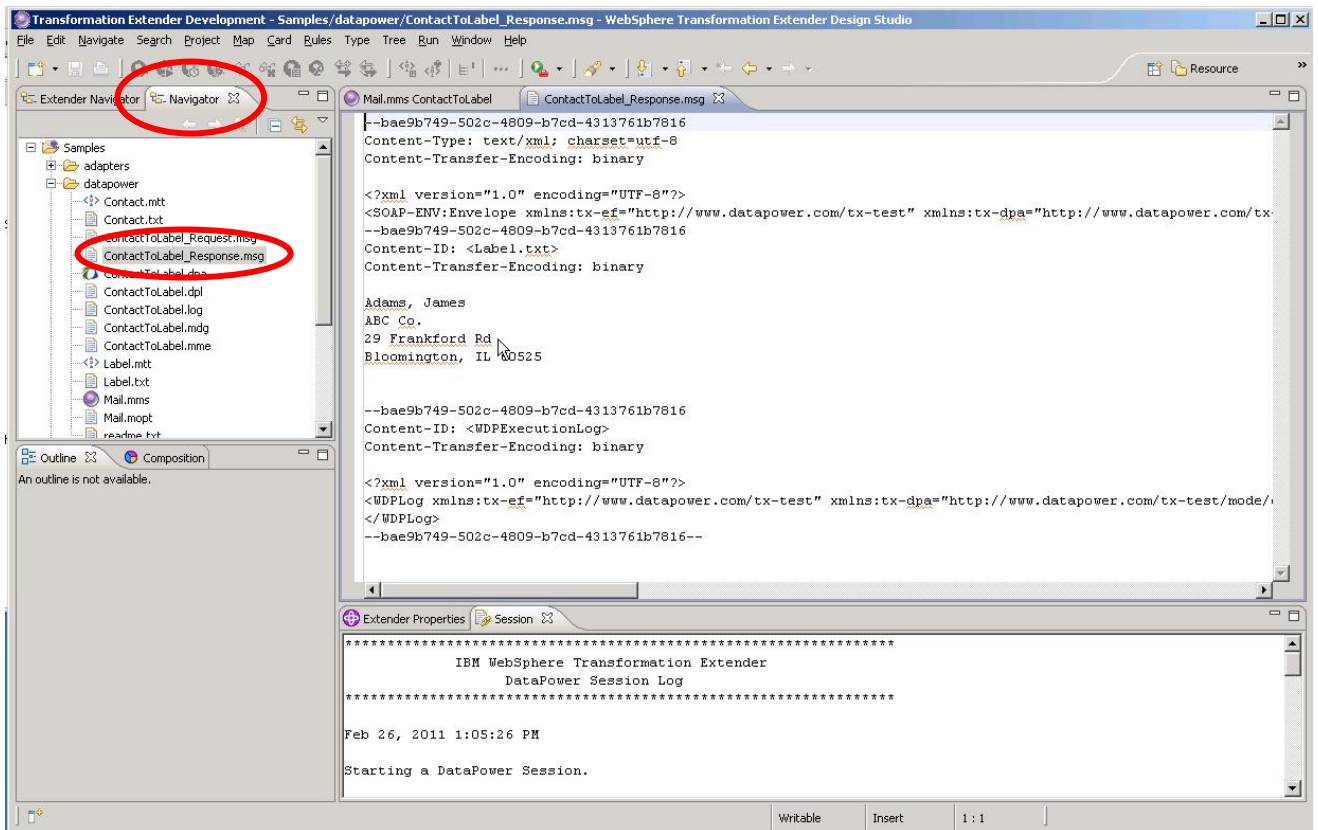
- A dialog box appears showing the results of the run on the DataPower device. Note that the port indicated should match the port you configured earlier for the DataPower service *tx-test*.



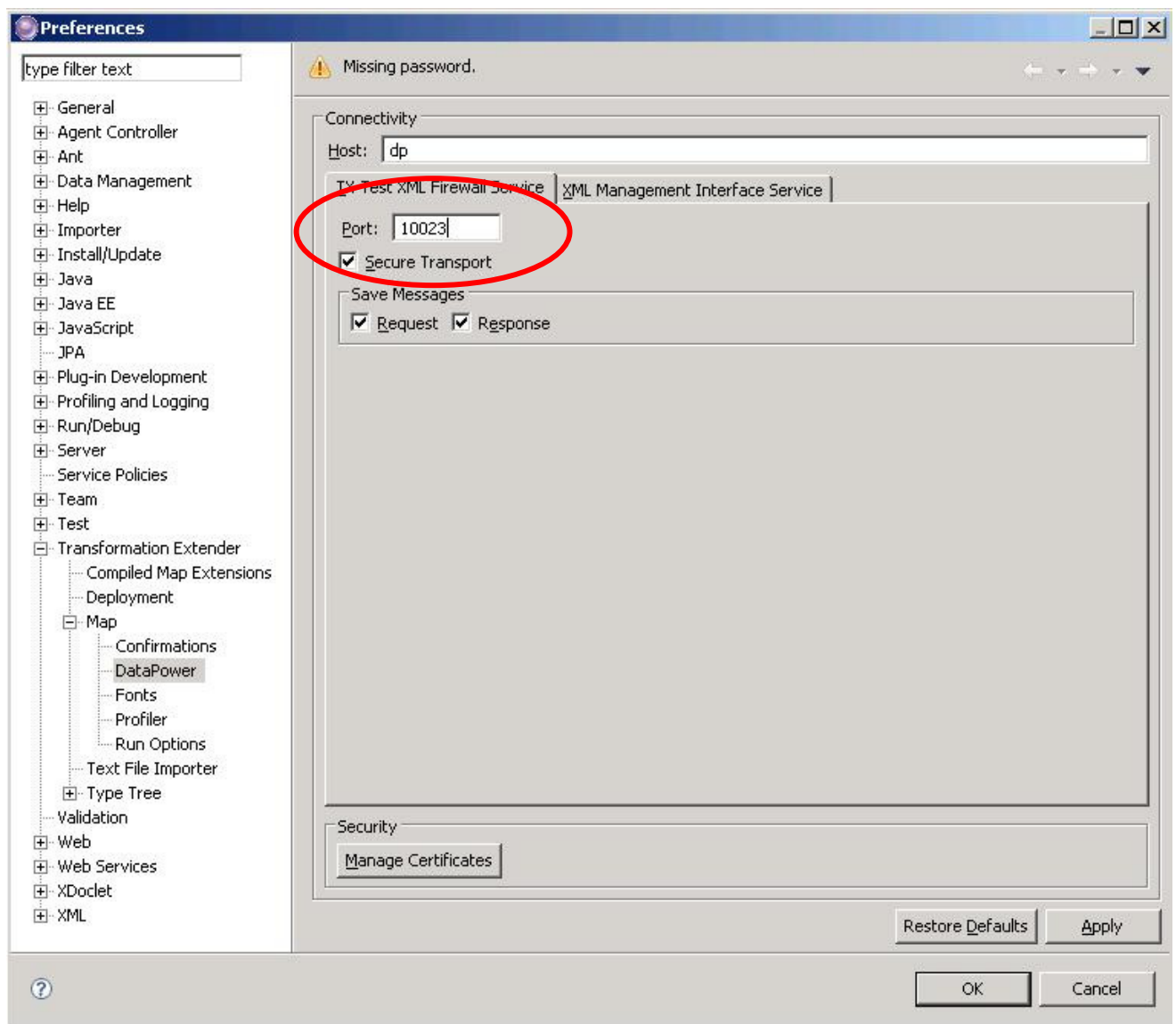
- After you press *Cancel* to dismiss the dialog box, the session view contains a log of the run:



- If you encounter any errors, review the session log, recheck your configuration steps to this point, correct any errors, and rerun the test to verify successful configuration. Assuming you have done all the configuration steps correctly, at this point you have successfully completed the setup and testing of the simplest integration of Design Studio with DataPower.
- Because we configured Design Studio to save both Request and Response messages, you should see both the “packaged” request and response messages. It is worth a quick look at the response message (*ContactToLabel_Response.msg*). **NOTE:** if you try to open the request message, one of the MIME sections is the binary map itself in encoding *application/octet-stream* and may give you some *funny* behavior when displayed on the screen... so best to avoid that unless you have a binary editor! Switch to the *Navigator* tab on the left-hand navigation pane. Open the response message (double-click the message from the *Navigator* window):



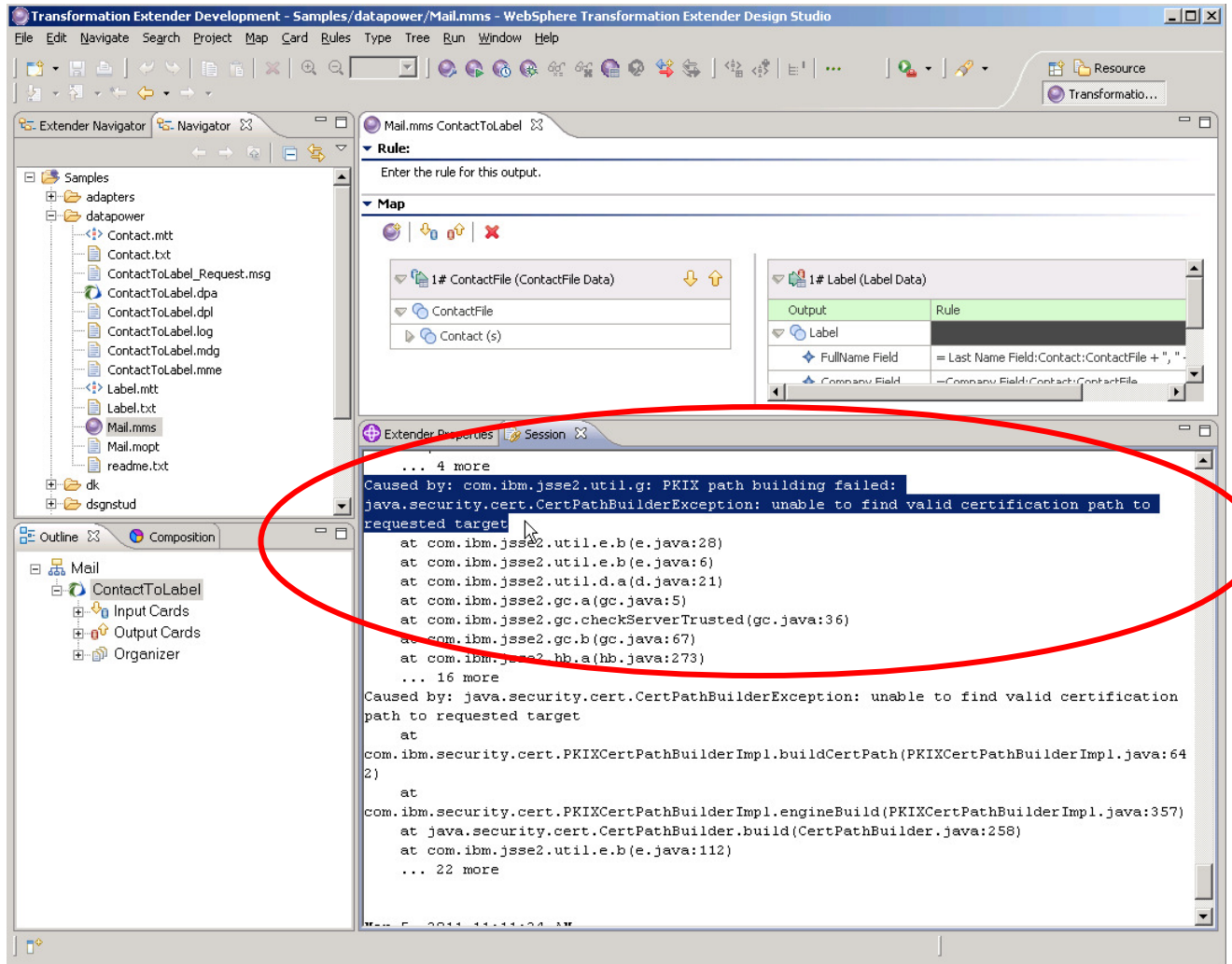
- These integration services package their requests and responses as MIME messages for handling by DataPower.
 - **NOTE:** The DataPower services that implement (handle) these requests are also very instructive to examine, but this lab doesn't have time to delve into them.
- **BEFORE CONTINUING WITH THIS SECTION, YOU MUST COMPLETE THE SKIPPED SECTION IN PART 2 (STARTING ON PAGE 17) WHICH CONFIGURES THE HTTPS SERVICE (tx-test-https) ON DATAPOWER. FOR NOW, CONTINUE TO PART 4 (PAGE 39). YOU CAN RETURN TO THIS PART OF THE LAB LATER IF TIME PERMITS.**
- Now, we'll move on to configure Design Studio and DataPower to use https for communication.
- From the Design Studio menu *Window*, open the *Preferences* dialog again. Change the port for the TX-Test XML Firewall Service to **8xx3** (to match the port value you configured for the DataPower service *tx-test-https*), and **check** the *Secure Transport* checkbox:



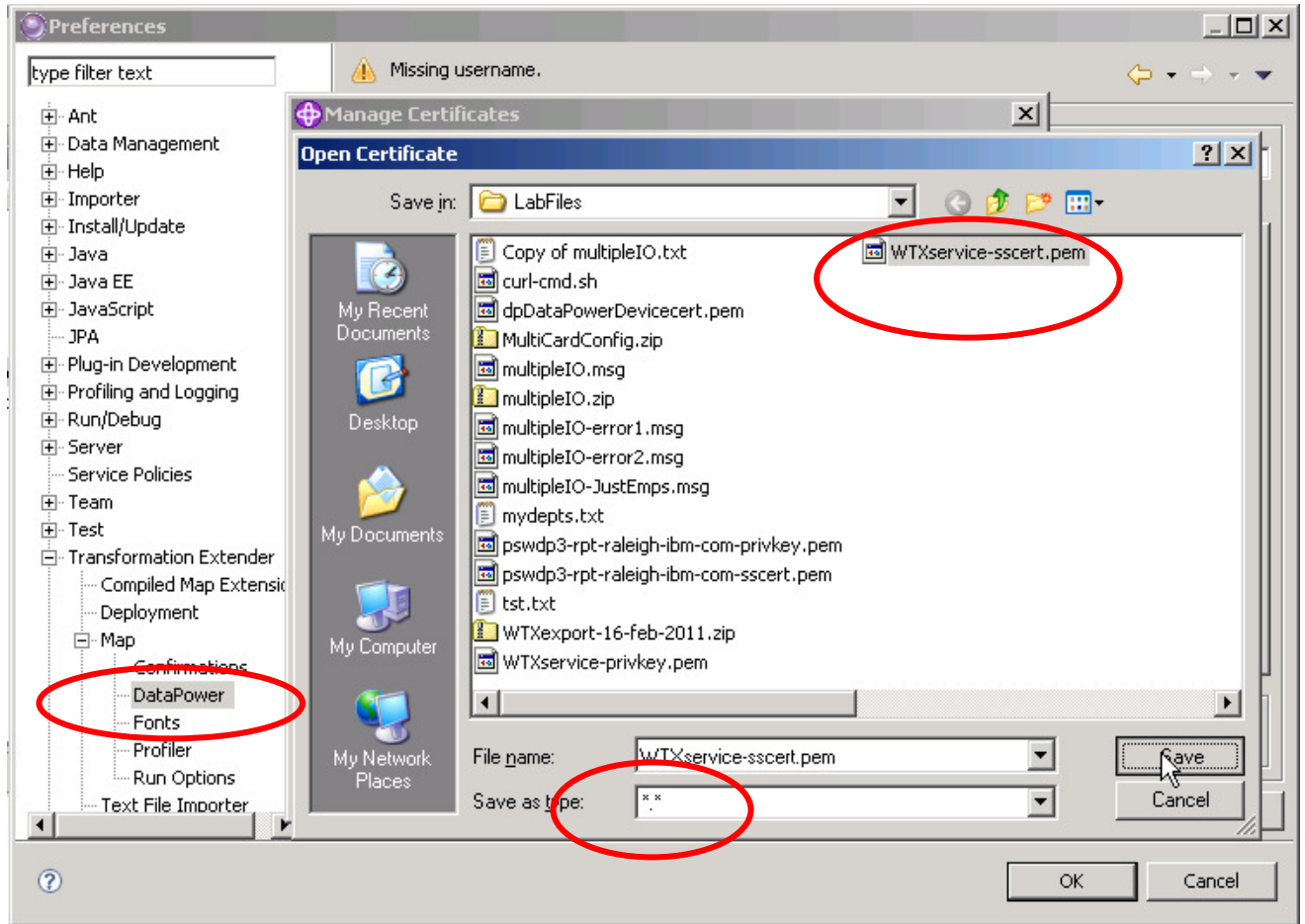
- **NOTE:** Double-check that the port number you specify here matches the port number you configured for the XML Firewall service *tx-test-https* on the DataPower device. They must match for the configuration to succeed.
- Click *OK*, then re-run the map (right-click on the map and choose *Run*):



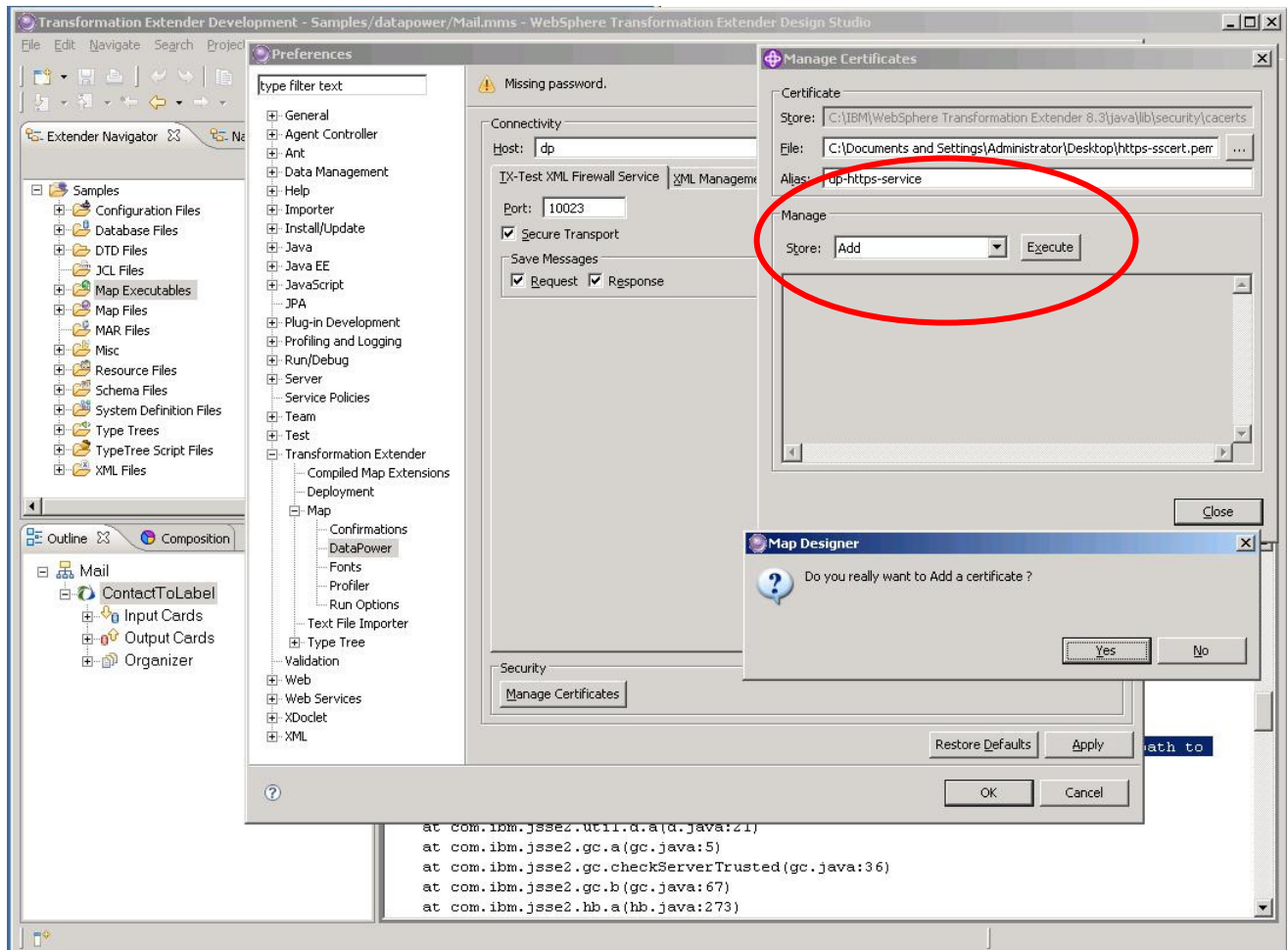
- The test fails. Diagnose the problem by examining the session log and reviewing the error.
- There is no foolproof set of steps for locating the critical piece of information in the logs that points out the error, but it is often productive to start at the end of the log and work backwards, since the errors are from a Java stack trace and therefore consist of a stack of nested exceptions.
- In this case, you can see that the error being reported is that the certificate (presented to Design Studio by DataPower as part of the SSL initiation protocol) can not be validated.



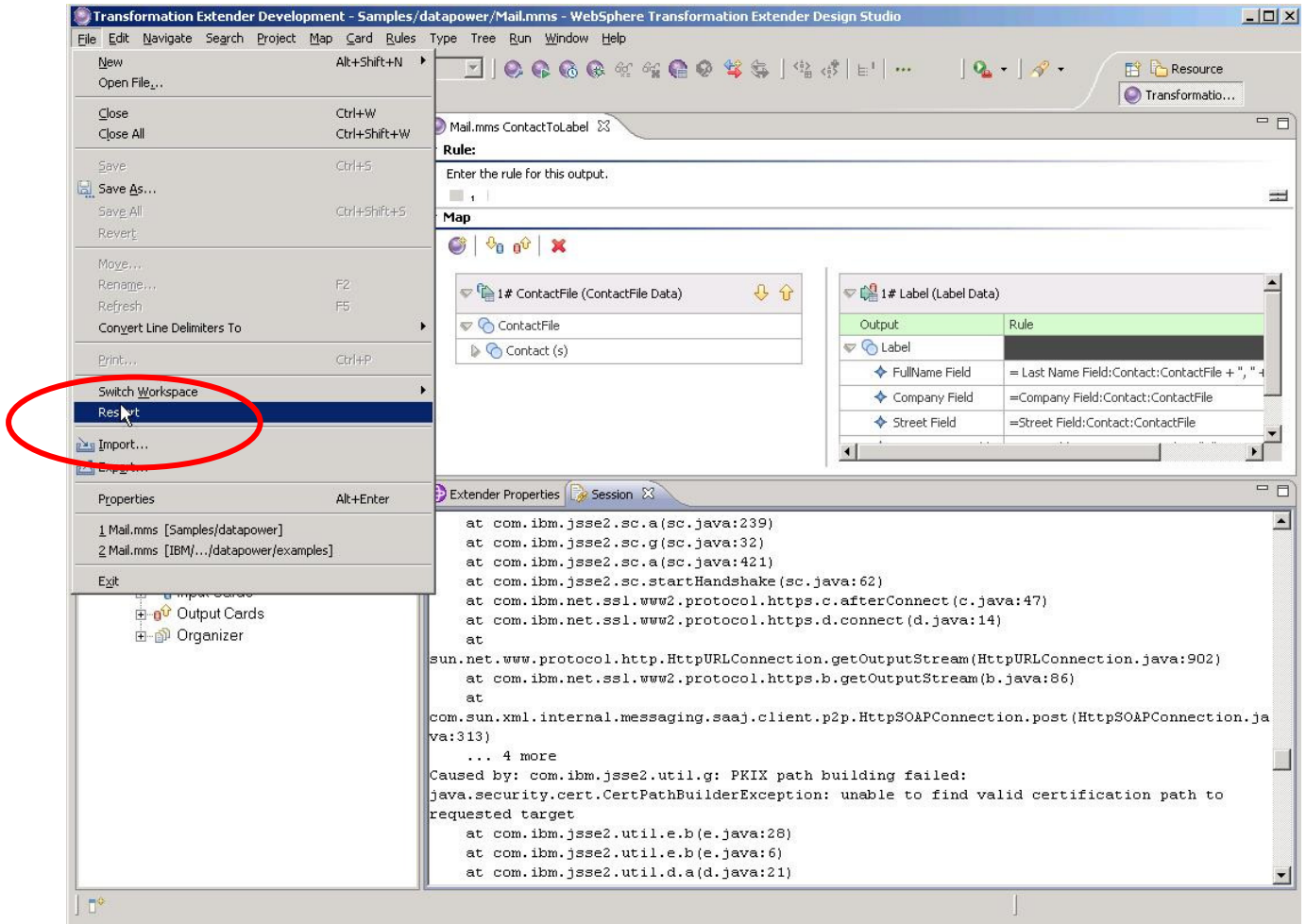
- In order for Design Studio to successfully connect to DataPower over HTTPS, you will need to configure Design Studio with the public certificate that the DataPower device presents to SSL clients for the service `tx-test-https`. Open the *Preferences* Dialog again (*Window/Preferences*), open to *Transformation Extender/Map/DataPower*, and then click on the 'Manage Certificates' button. From the resulting dialog, click on the ... button to the right of the *File:* field, and select the **same public** certificate (`c:\Labfiles\WTXservice-sscert.pem`) that you used when you configured the DataPower service `tx-test-https`. Note that you will have to change the *Save as type:* filter to `*.*` because this certificate has a `.pem` extension, not the default `.der` extension. The WTX online documentation suggests that only DER format certificates will work, but (as you will see), PEM format certificates will work as well:



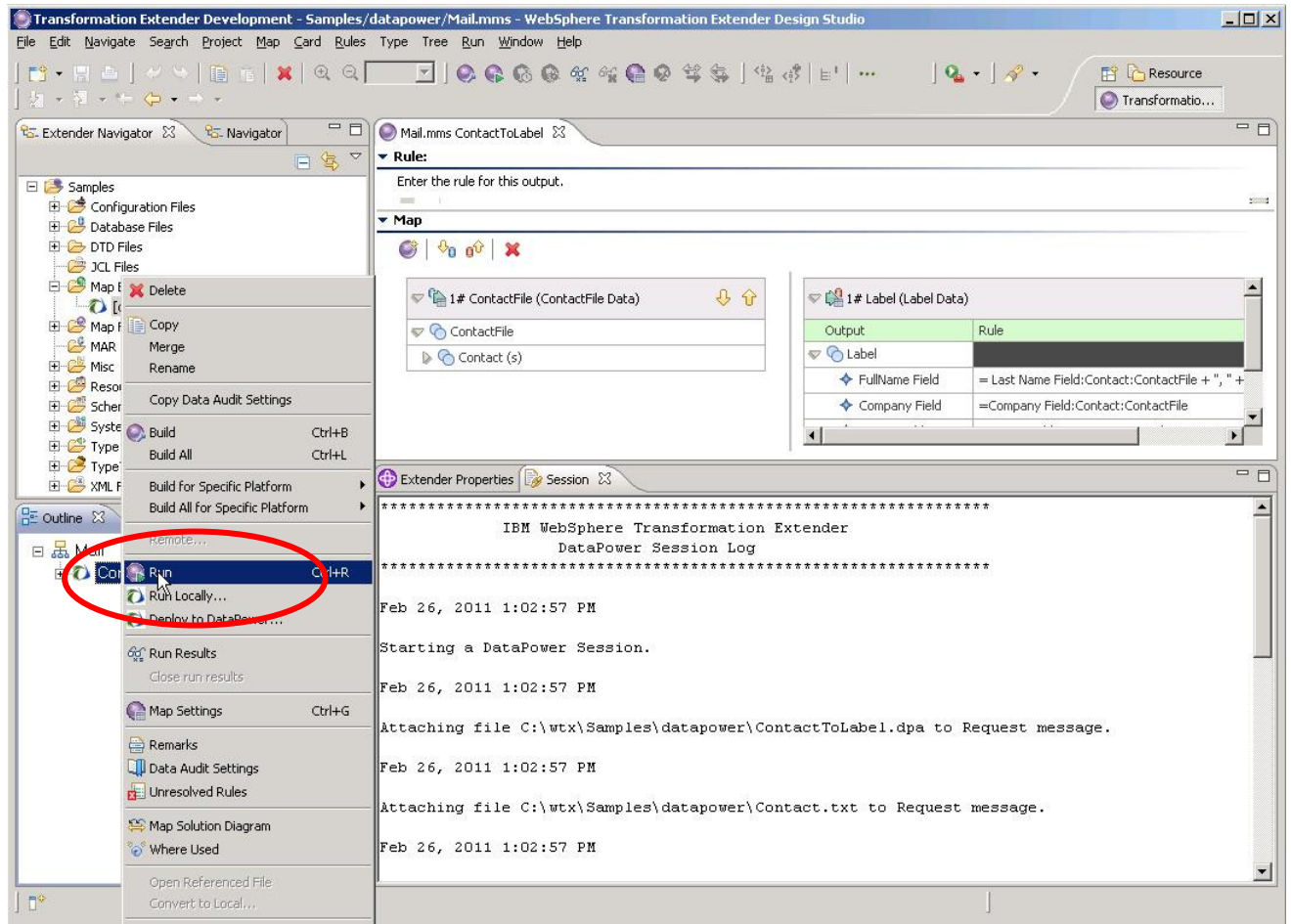
- After returning to the *Manage Certificates* dialog, type in the value *dp-https-service* into the alias field. [NOTE: Any unique alias would work, but it makes sense to have the alias reflect the use for which it is intended in some way.] Then select *Add* from the *Store:* dropdown list:



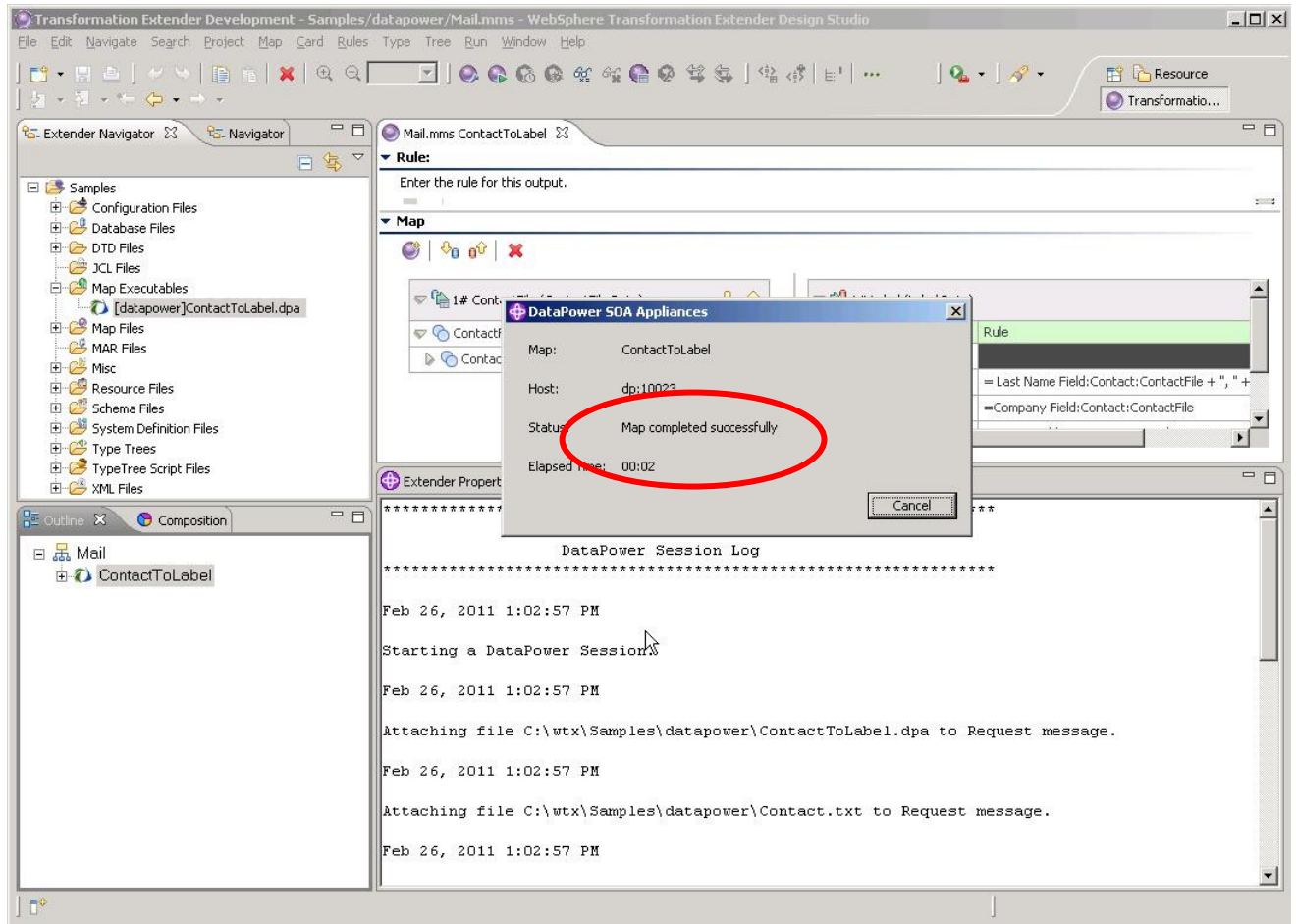
- Answer *yes* to the dialog asking *Do you really want to Add a certificate?*. Then close the *Manage Certificates* dialog, and then close the *Preferences* dialog by clicking *OK*. Repeat the test (right-click on the map, select *Run*).
- What happened? You should encounter the same error as before (in the session log: “*unable to find valid certification path...*”). Unfortunately, there is no way to force Design Studio to re-read (and therefore pick up changes to) the keystore and truststore files it uses, short of restarting Design Studio. So, restart Design Studio [and open the same workspace (clwtx) after restart]:



- And then try to *Run* the test again:

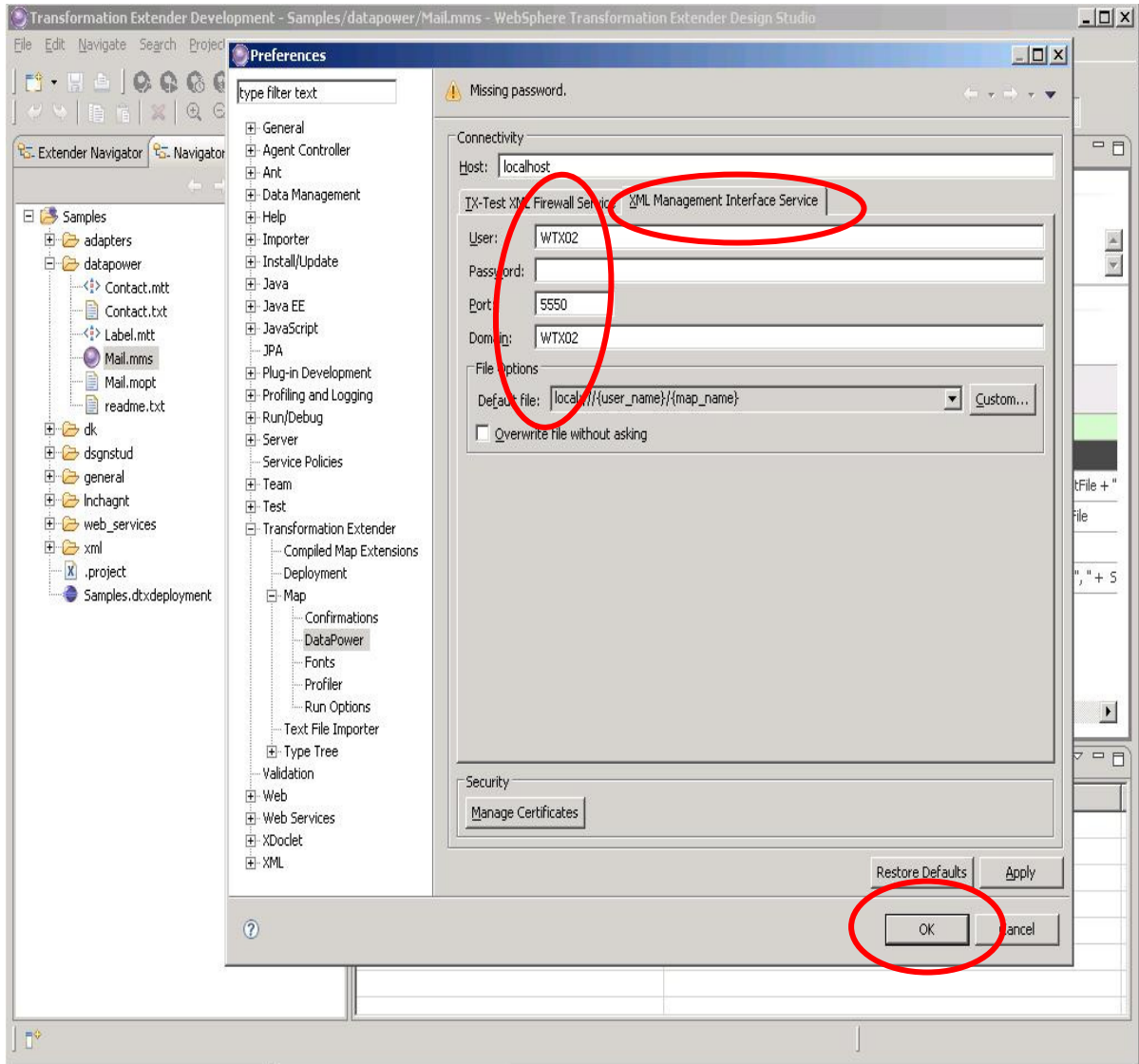


- This time, the map executes successfully!

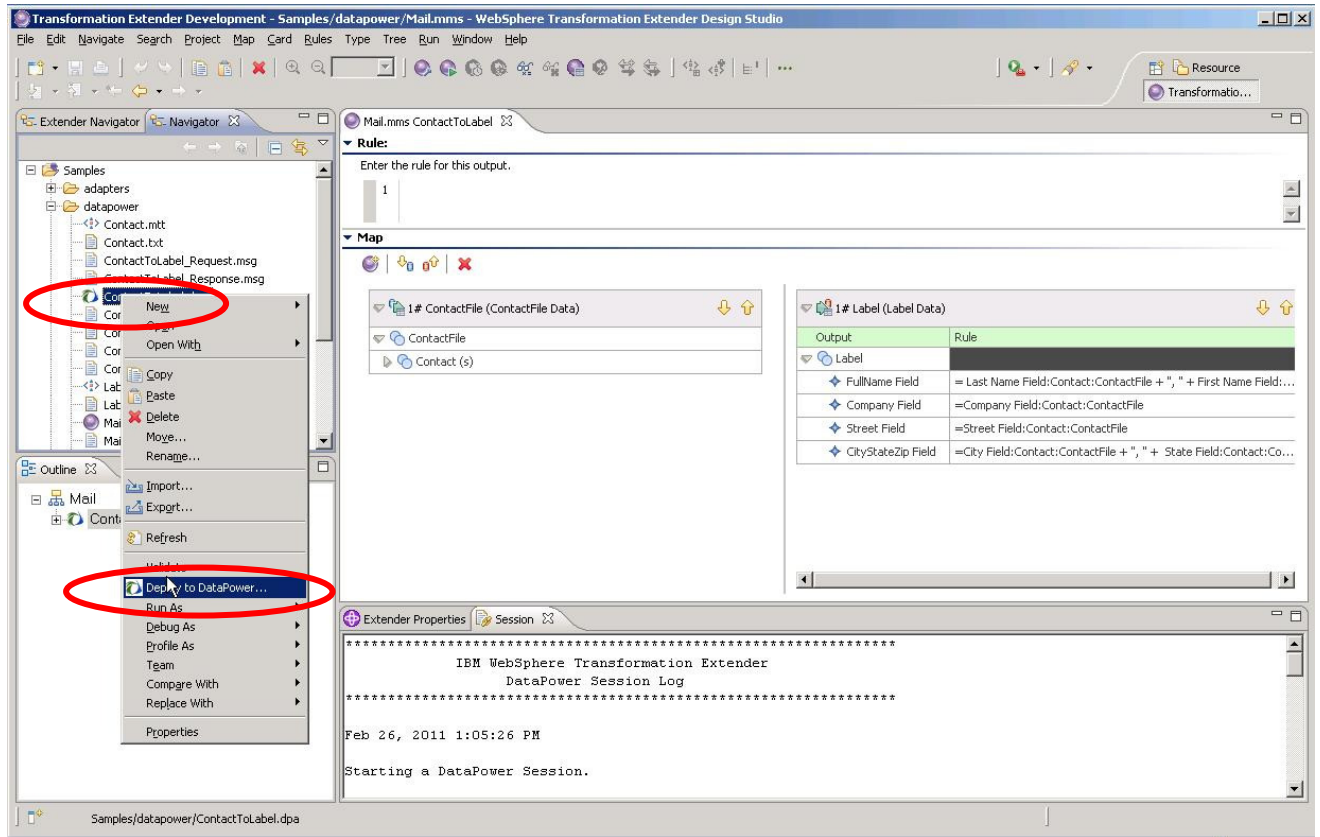


Part 4: Configuring Design Studio to Upload Maps to DataPower

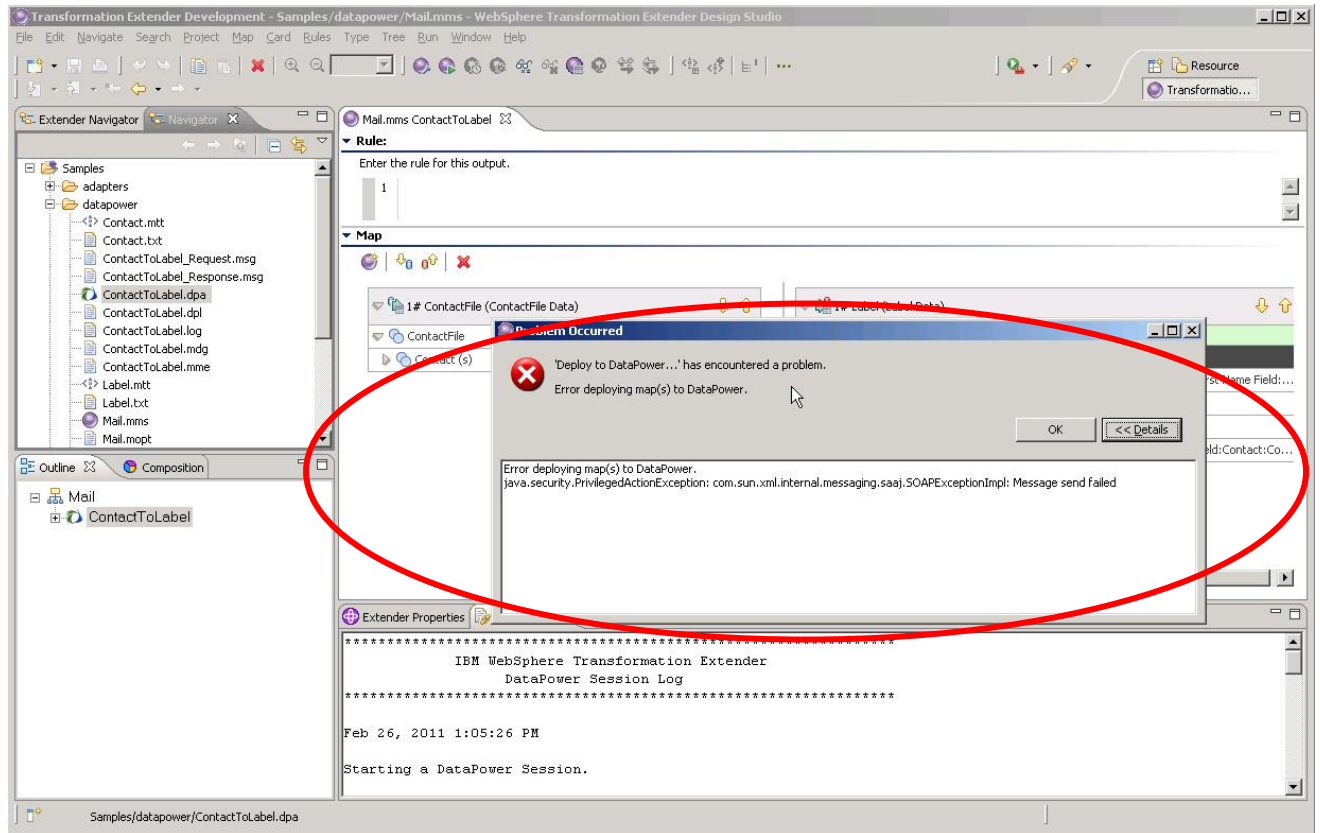
- Another useful feature of Design Studio is the ability to upload compiled maps directly to a DataPower device. This functionality would typically be used when a developer is implementing a DataPower service which integrates a map into that service. While iteratively developing (changing) the map in Design Studio, it is handy to be able to upload the map to the DataPower filesystem from within Design Studio so that the new map can be tested.
- To begin configuring this feature, open (once again) the *Preferences* Dialog to *Transformation Extender/ Map / DataPower*. Select the *XML Management Interface Service* tab. Now, enter the User, Password, and Domain specific to this lab workstation. The user and domain will be **WTXxx**, where **xx** is the two-digit number you are to use for this workstation, and the password is **WTXxxxx**. (These are the very same user, password, and domain that you used to log into the DataPower GUI.) Leave the port setting at the default value (5550) – this is the DataPower XML Management service port; there is only one port setting for the entire device.
 - **NOTE:** For this functionality to be available, the DataPower administrator must enable the XML Management service on the device. That step has already been done for this lab's device.



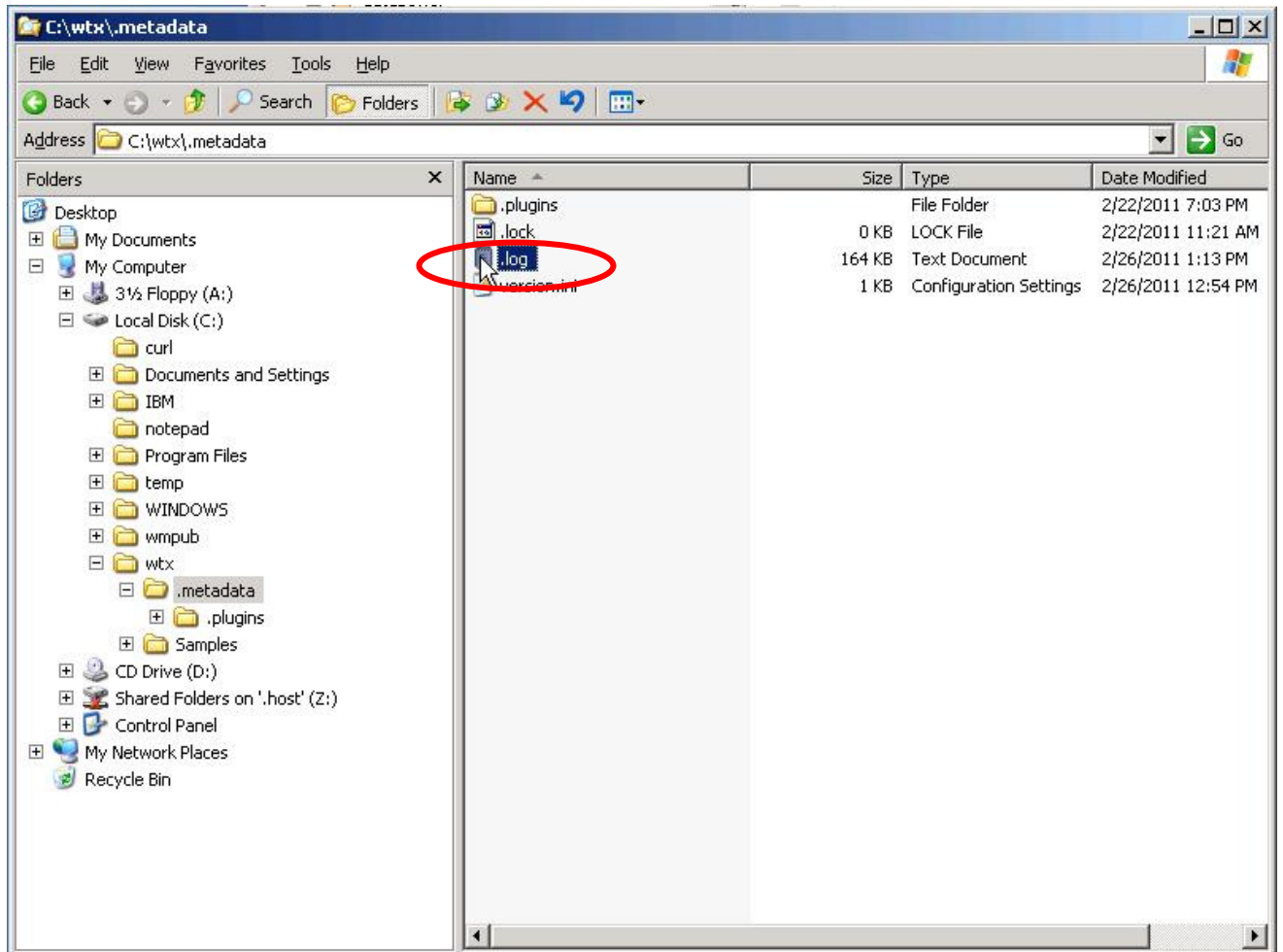
- Click **OK** to save your new settings, then test that you can upload a map from Design Studio to the DataPower device by right-clicking on the map and selecting *Deploy to DataPower*.



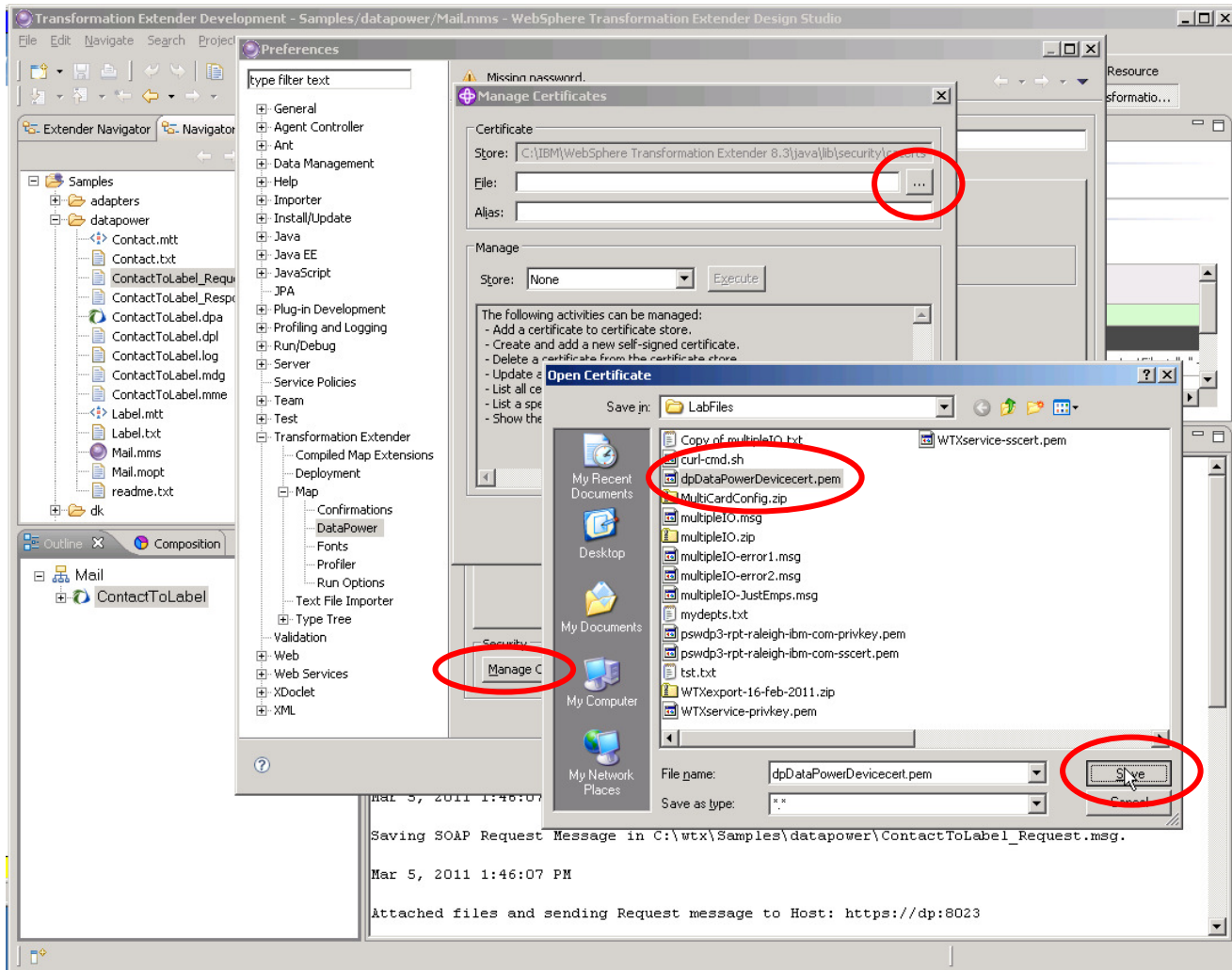
- Did it succeed? You should get an error similar to this:



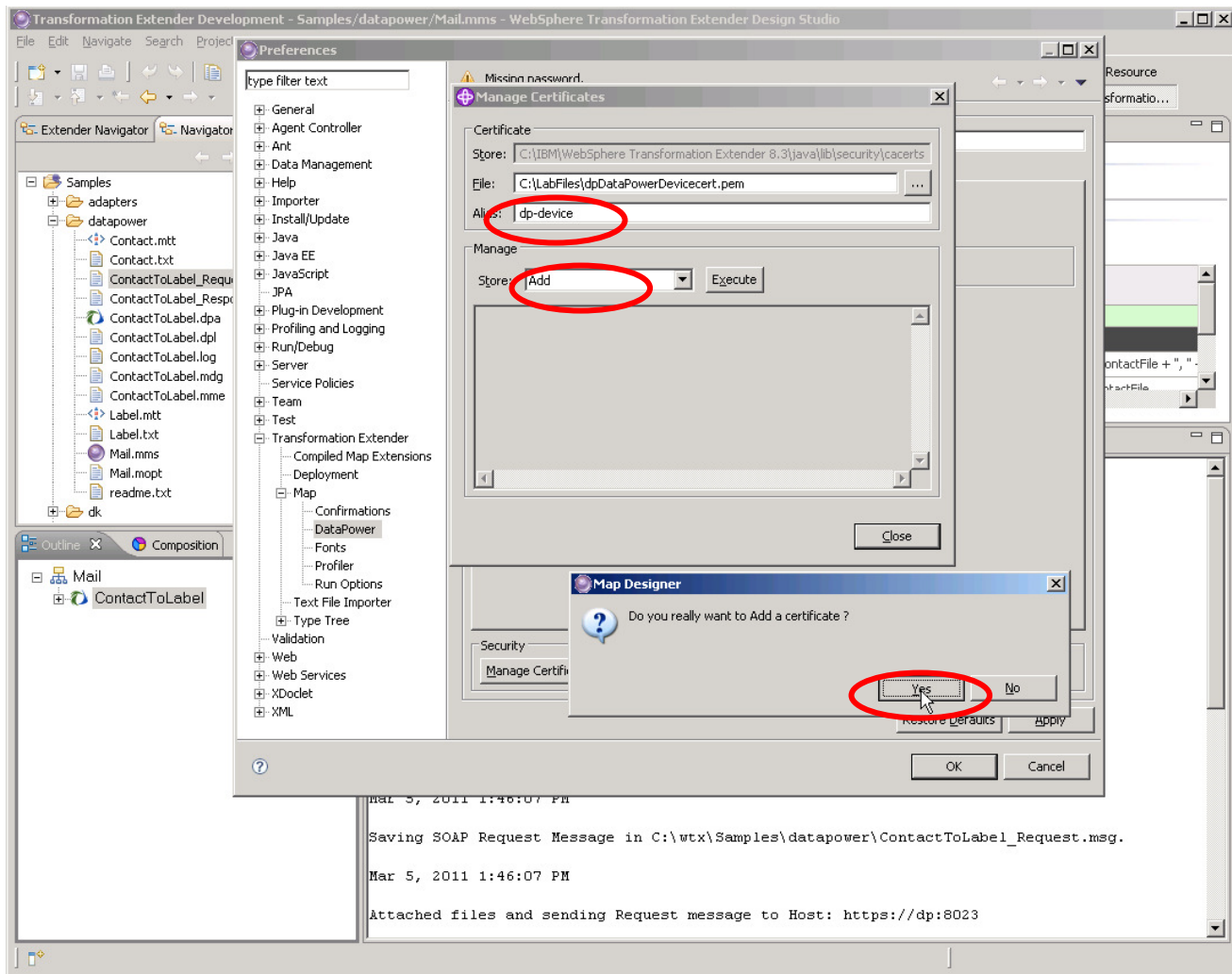
- You will now try to discern the problem by taking a look at Design Studio's log file. This file can be found in the *.metadata* folder of the workspace directory. Open Windows Explorer and open the *.log* file (*c:\wtx\metadata\log*):



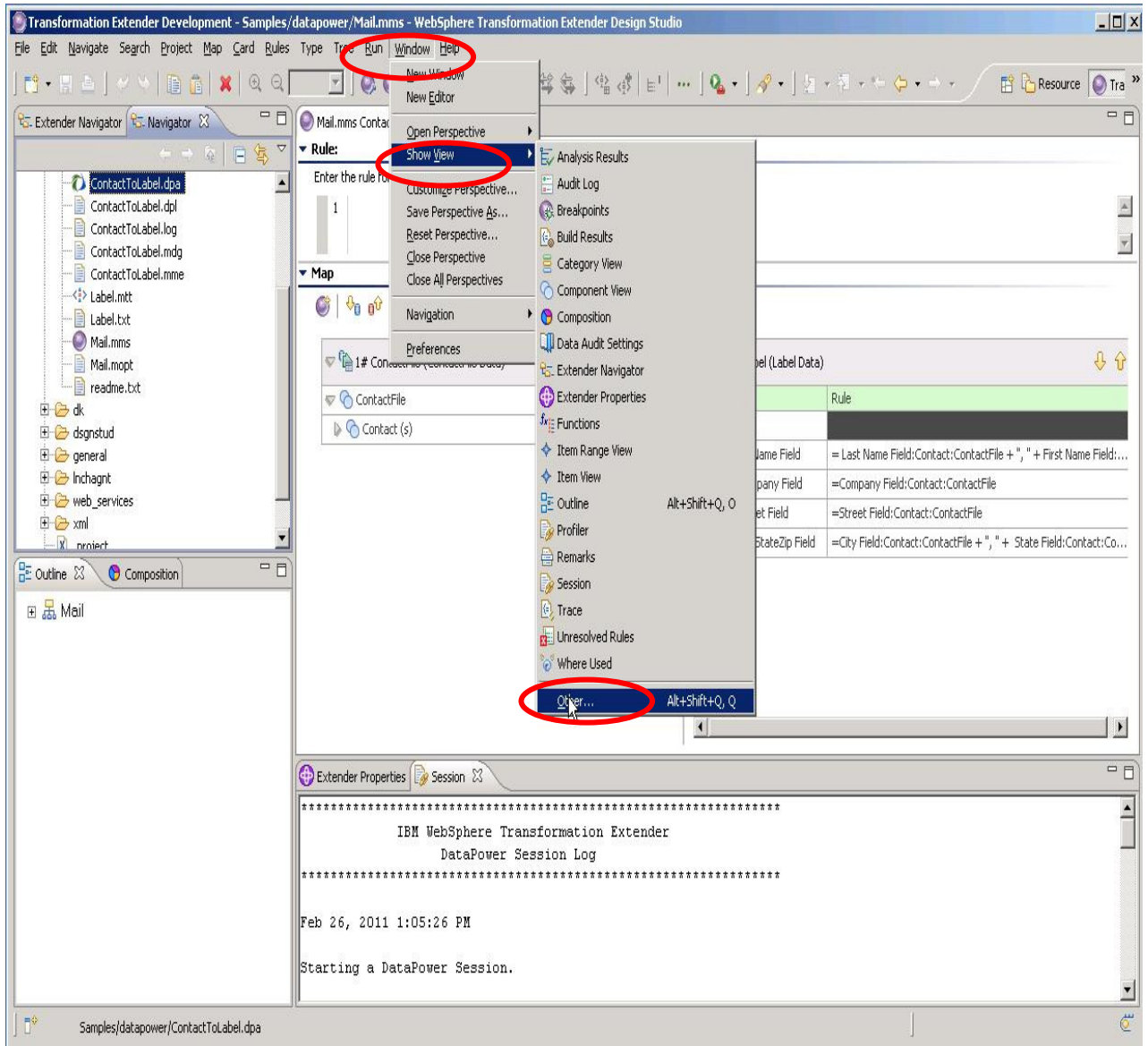
- Near the end of the .log file, you can see the problem.
 - As we indicated in *Part 3* (page 32) -- when we were configuring https communication for testing maps -- there is no foolproof set of steps for locating the critical piece of information in the logs that points out the error, but it is often productive to start at the end of the log and work backwards, since the errors are from a Java stack trace and therefore consist of a stack of nested exceptions.

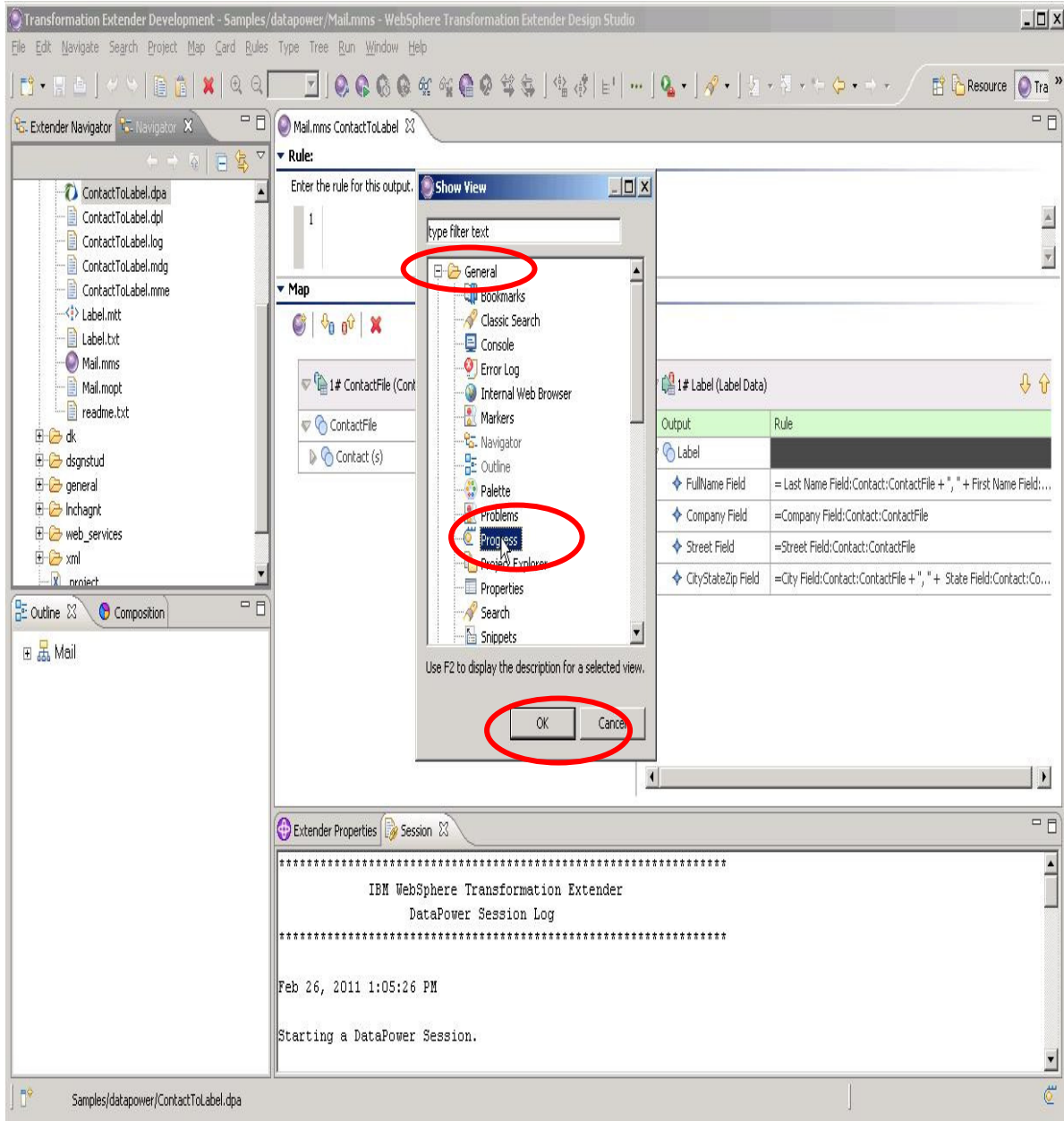


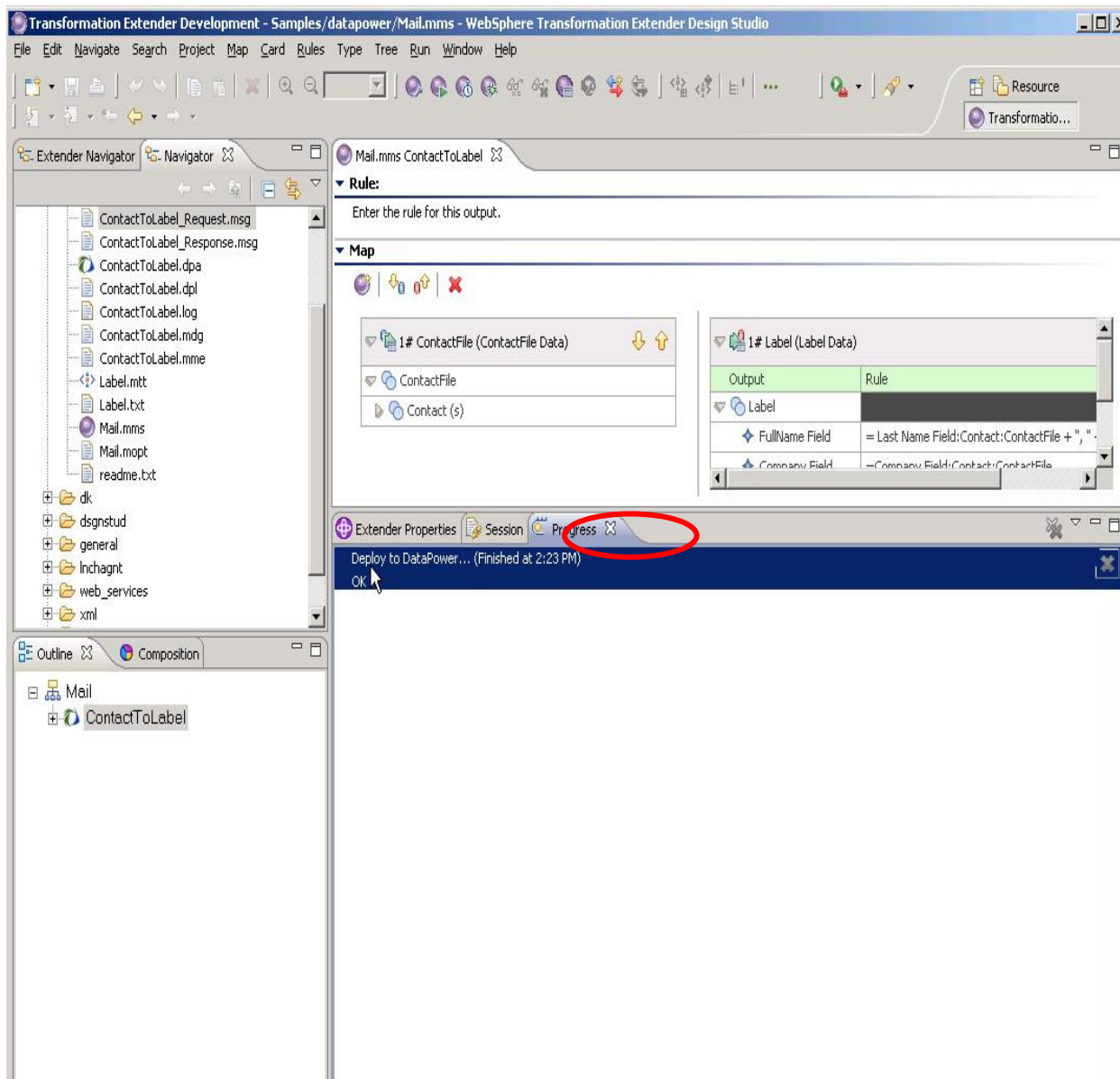
- Then (similarly to what you did before), give the certificate an alias (*dp-device*), then select 'Add' from the pull-down list to the right of 'Store:' to add the certificate to Design Studio's trust-store:



- As before, you must restart Design Studio so that the new certificate can be used. After restart, attempt another Deploy; this one *should* succeed. We can view the progress of this deployment by opening the *Progress View*:







- So, where did we actually deploy the map file? [That is, where is the file on the DataPower device?] We can view the DataPower filesystem through Internet Explorer. From the left-hand navigation menu, click *Administration*, then *File Management*. Expand the *local* directory, then the directory *WTXxx* under *local*:

Available Space: 86 MBytes (encrypted), 215 MBytes (temporary)

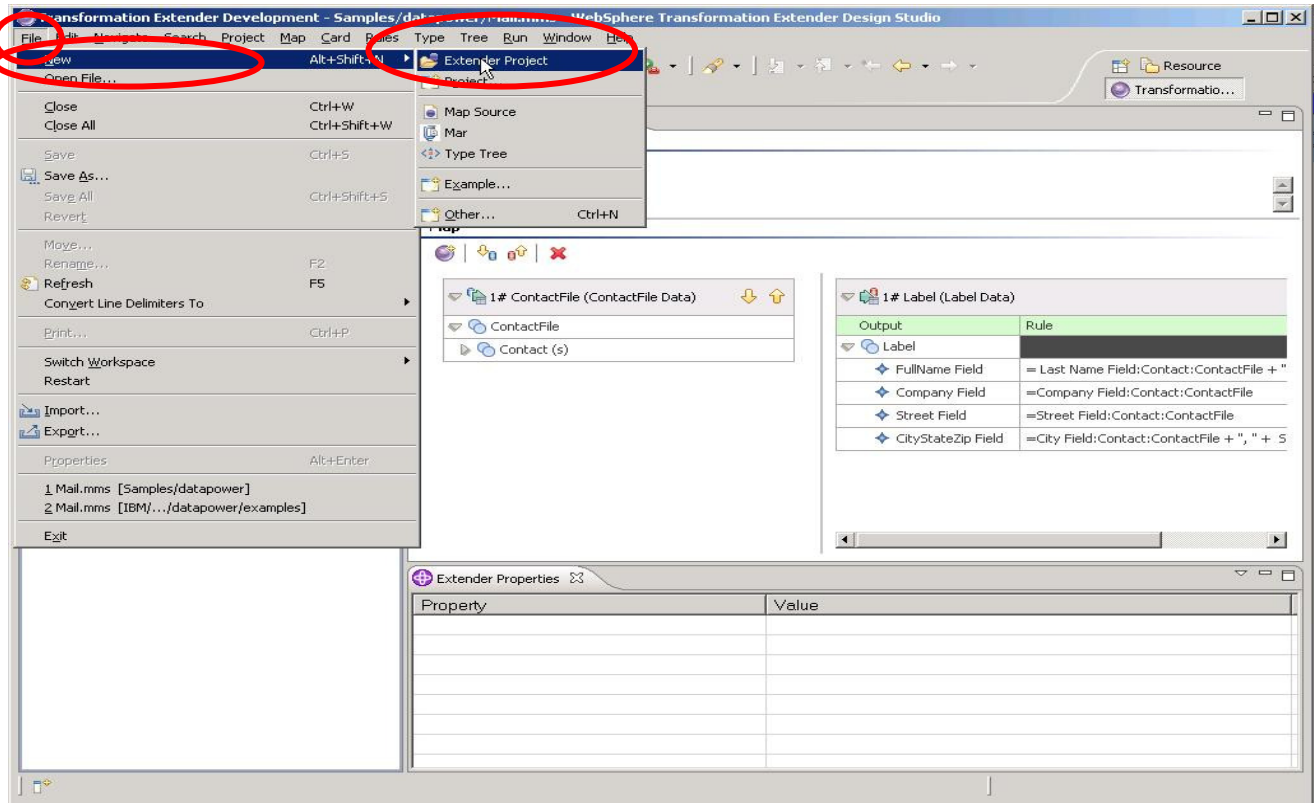
Name	Action	Size	Modified
cert:	Actions...		
chkpoints:	Actions...		
config:	Actions...		
export:	Actions...		
local:	Actions...		
deployed	Actions...		
WTX02	Actions...		
ContactToLabel.dpa	Edit	8238	2011-03-05 18:03:00
createTraceAndLog.xml	Edit	6233	2011-02-22 13:32:45
dp-tx-interop-protocol.wsdl	Edit	14605	2011-02-22 13:32:45
finishTXTest.xml	Edit	4371	2011-02-22 13:32:45
prepareTXInputs.xml	Edit	1947	2011-02-22 13:32:45
prepareTXMap.xml	Edit	5122	2011-02-22 13:32:45
reportTXTestError.xml	Edit	6735	2011-02-22 13:32:45
transform.xml	Edit	488	2011-02-27 17:27:34
logstore:	Actions...		
logtemp:	Actions...		
pubcert:	Actions...		
sharedcert:	Actions...		

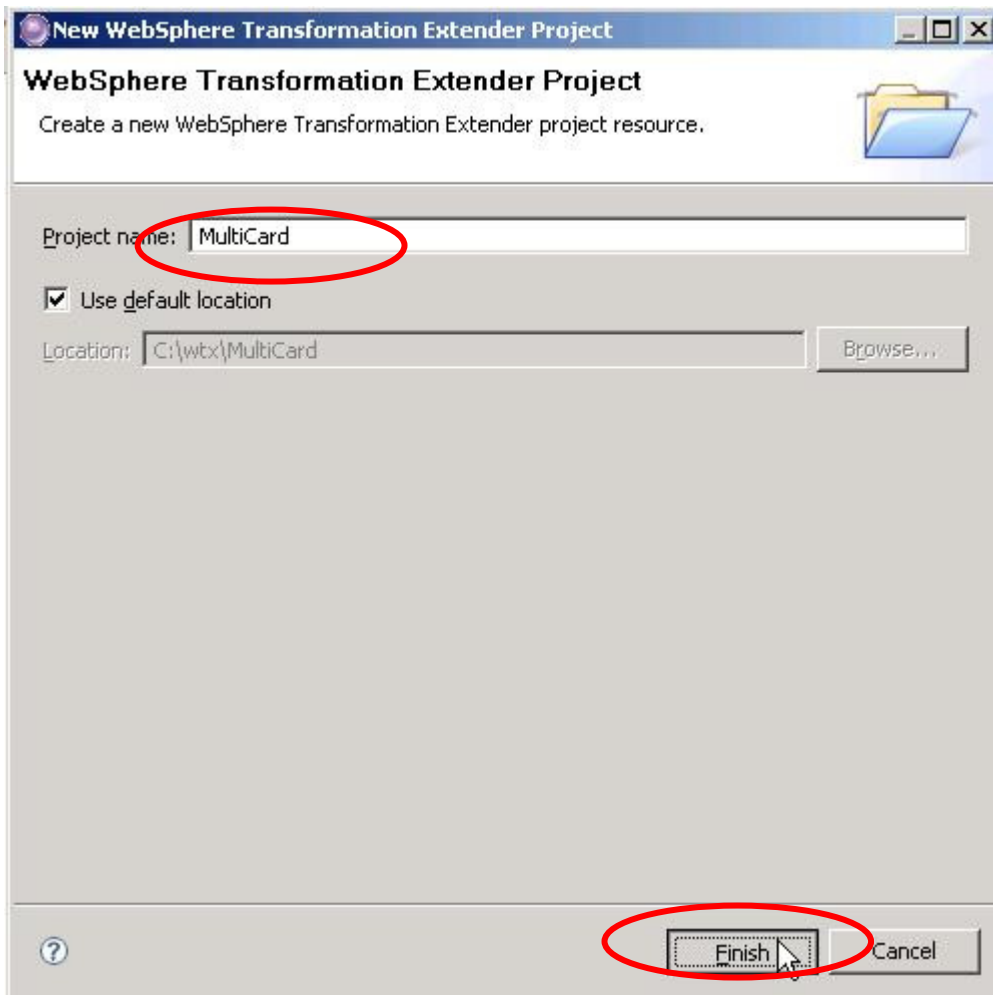
Don't forget to save your work before you logout

- This location is controlled by the settings under Preferences in Design Studio. You can review your preferences setting; the [default] location is `local:///user_name/map_name`. This location can be modified to suit your preferences and the requirements of any services that you might implement which refer to the map location. We'll see how we use this uploaded map in the next exercise.

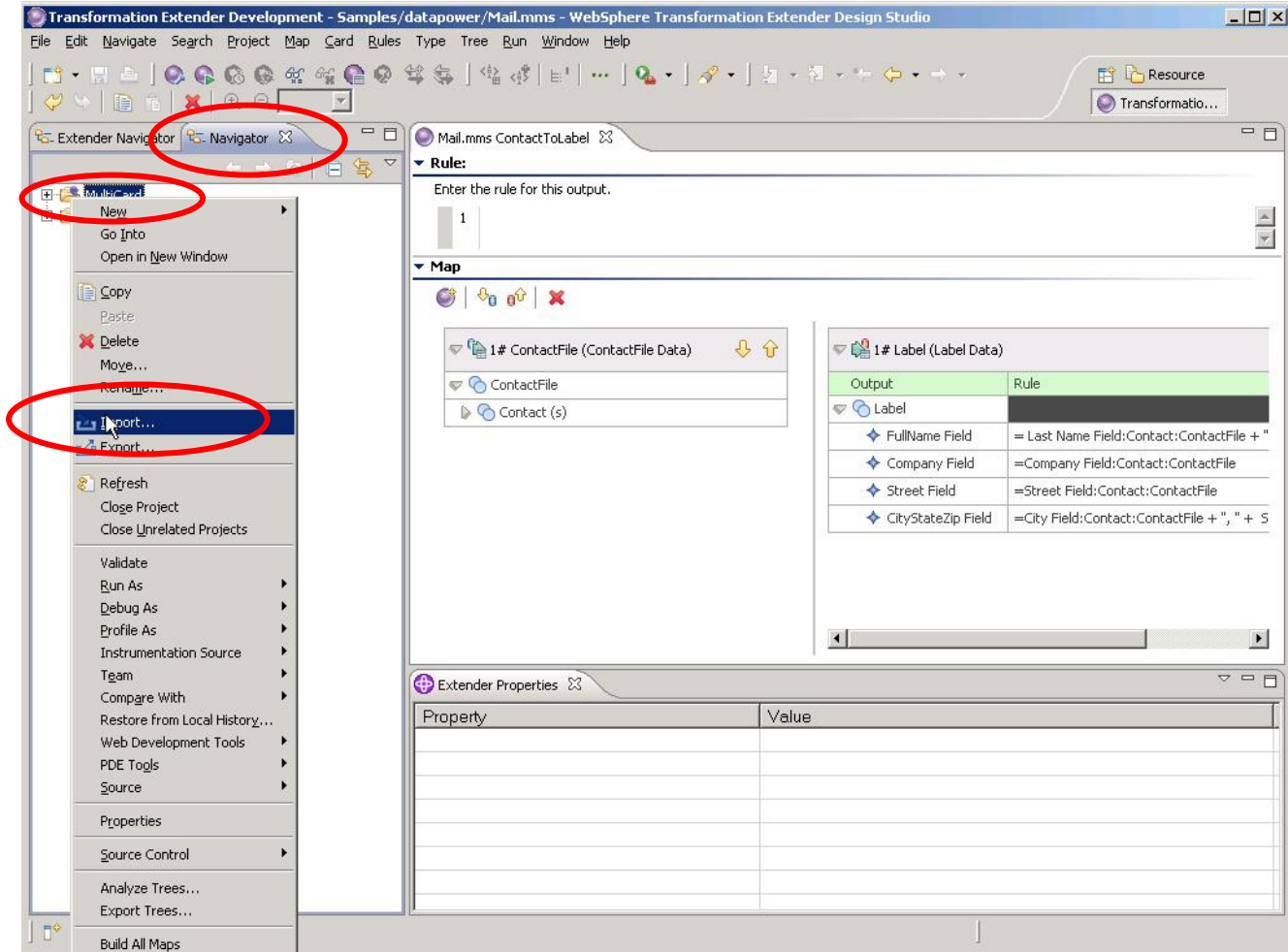
Part 5: Setting Up a MultiCard Project in WTX Design Studio

- To save time, you will import an existing multi-card map into Design Studio. First, create a new *Extender Project* named MultiCard

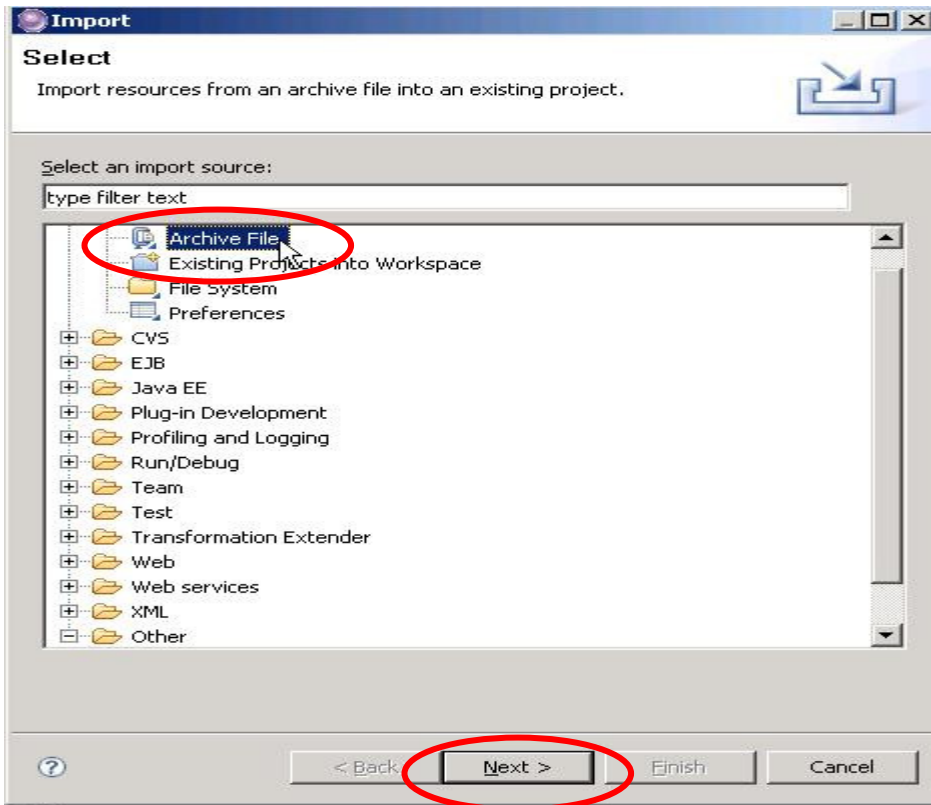




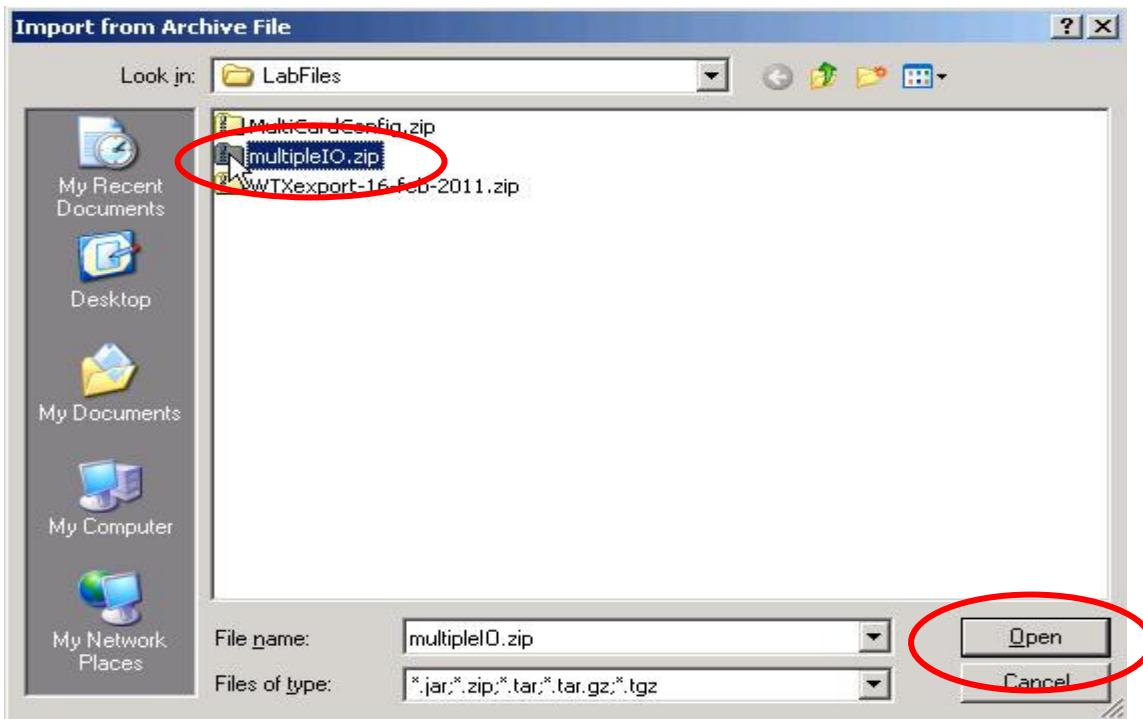
- From the navigation pane (on the left), switch to the *Navigator* tab. Next, import the project from the archive file *c:\LabFiles\multipleIO.zip* by right-clicking on the new *Multicard* project name, and selecting *Import* from the pop-up menu:



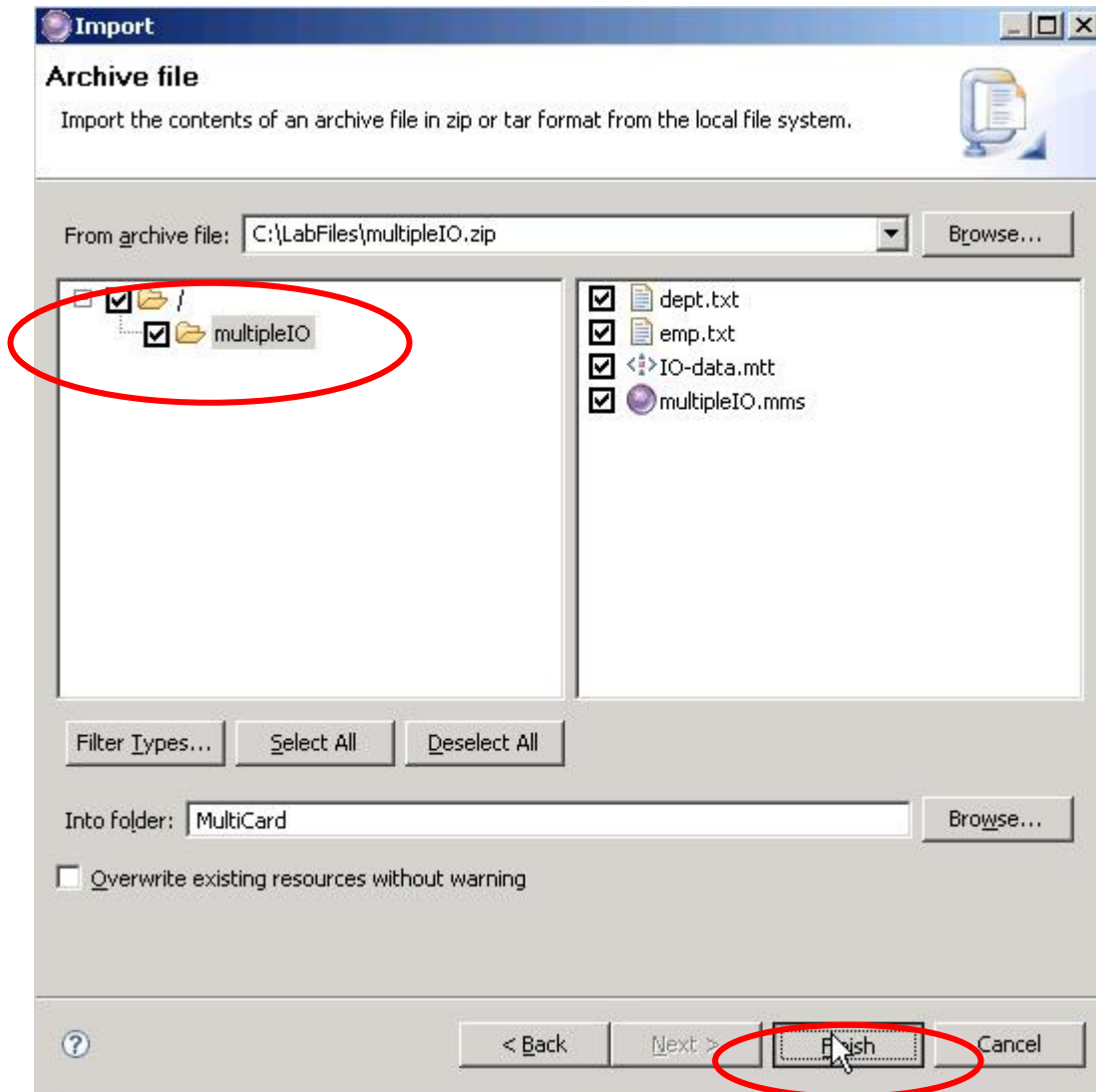
- Select *archive file* from the *General* folder on the next dialog window:



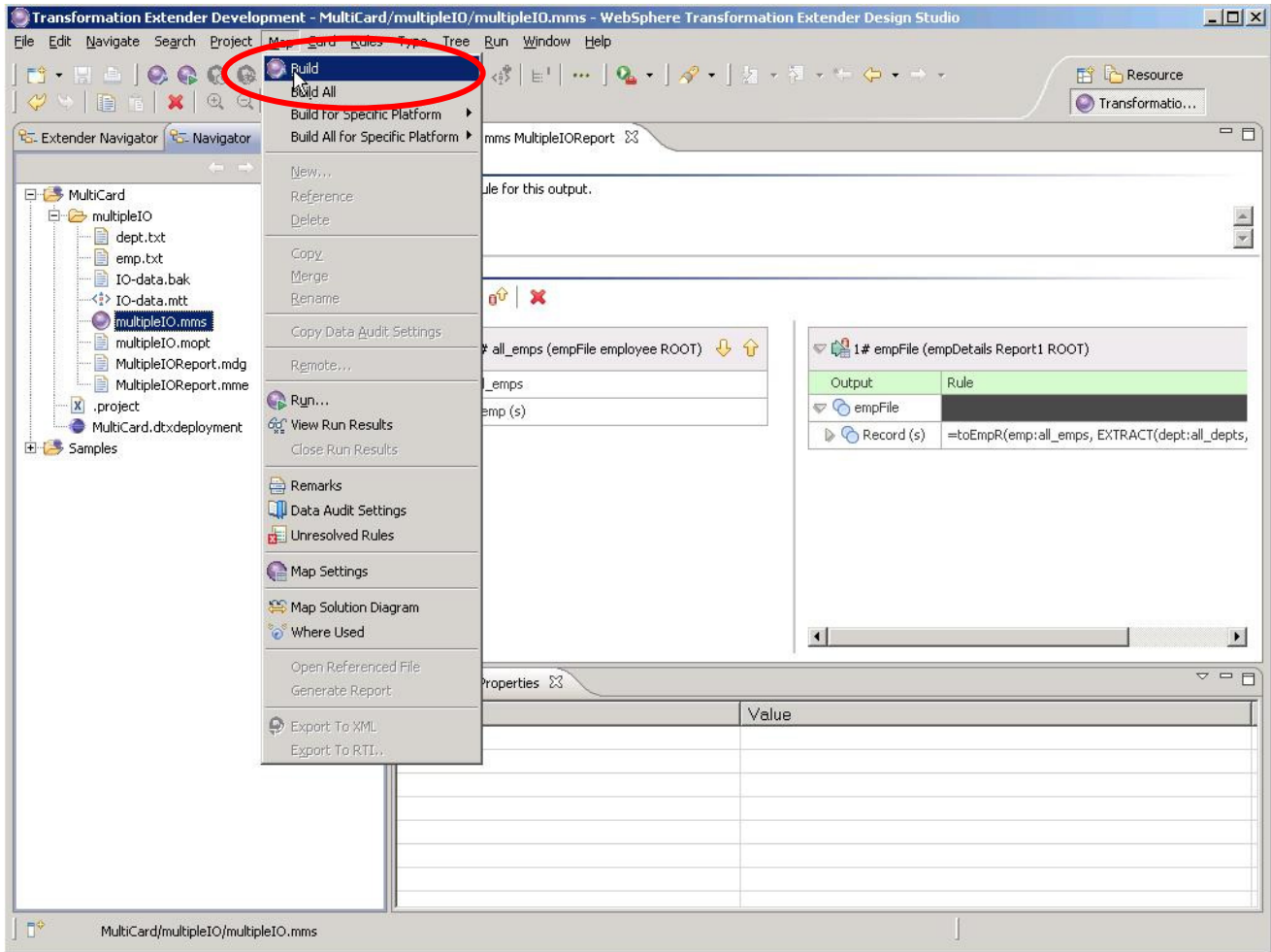
- and select `c:\LabFiles\multipleIO.zip`:



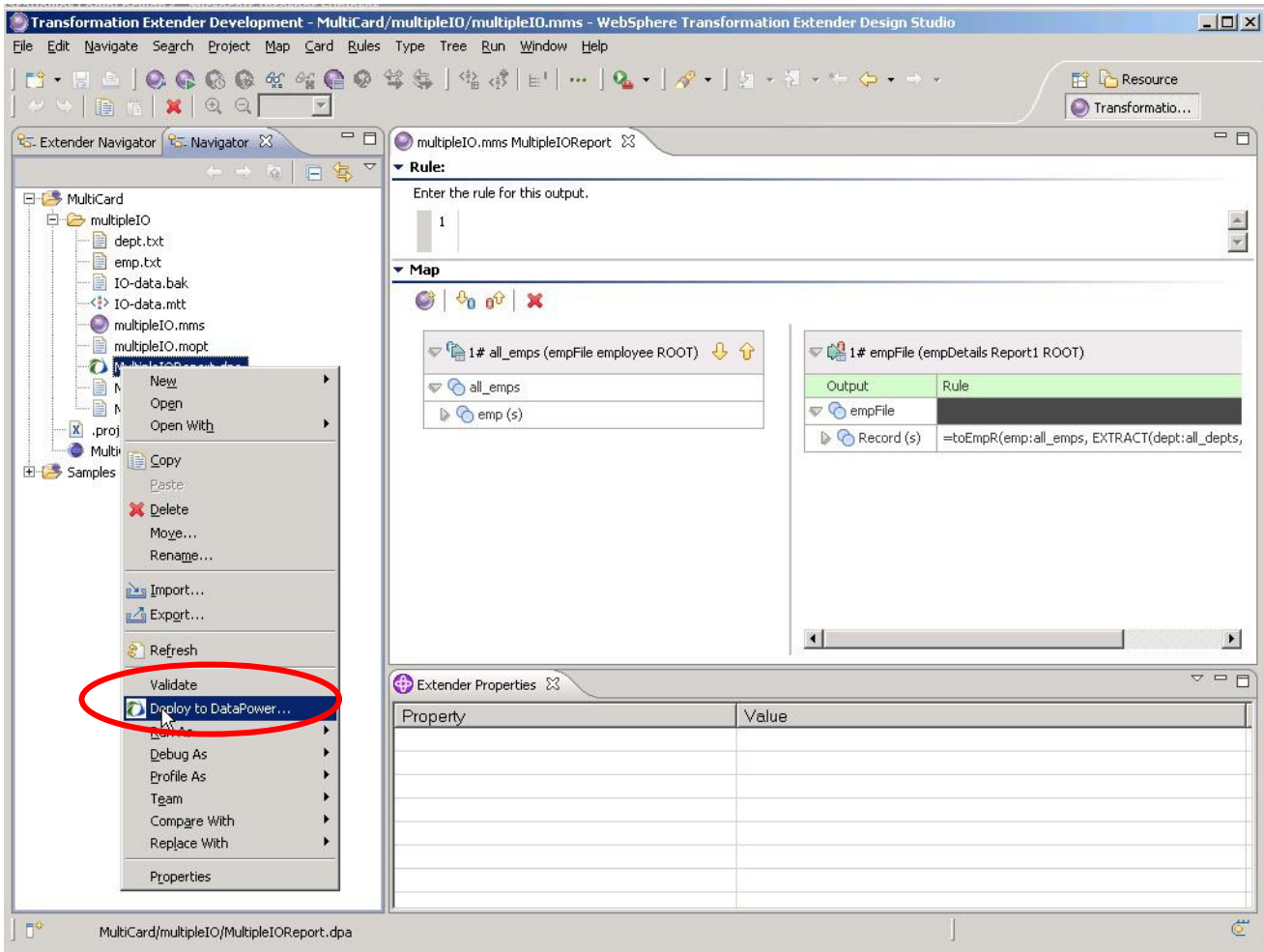
- You can expand the contents to see what you are actually importing:



- Now, open the project, open the map (*multipleIO.mms*), and then build the map:



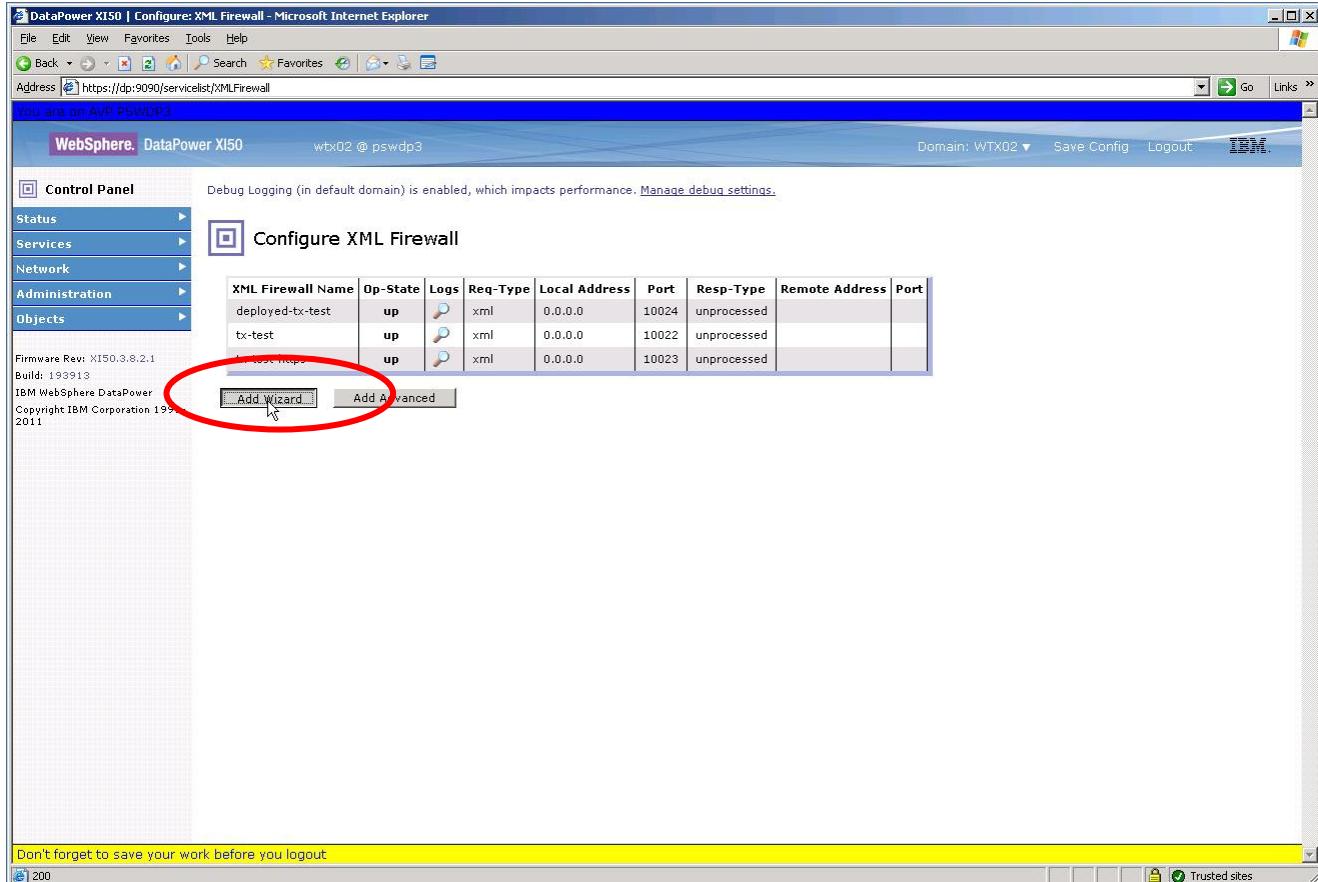
- then deploy the compiled DataPower map (*multipleIOReport.dpa*) to DataPower (right-click on the *.dpa* file):



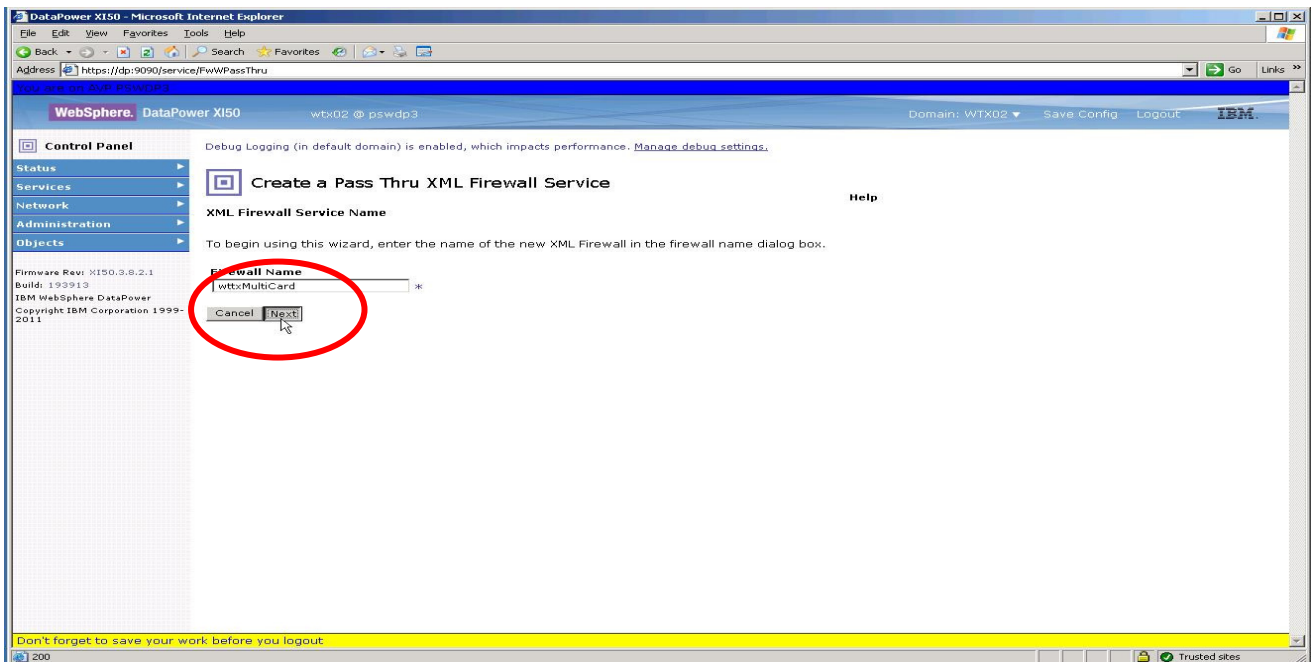
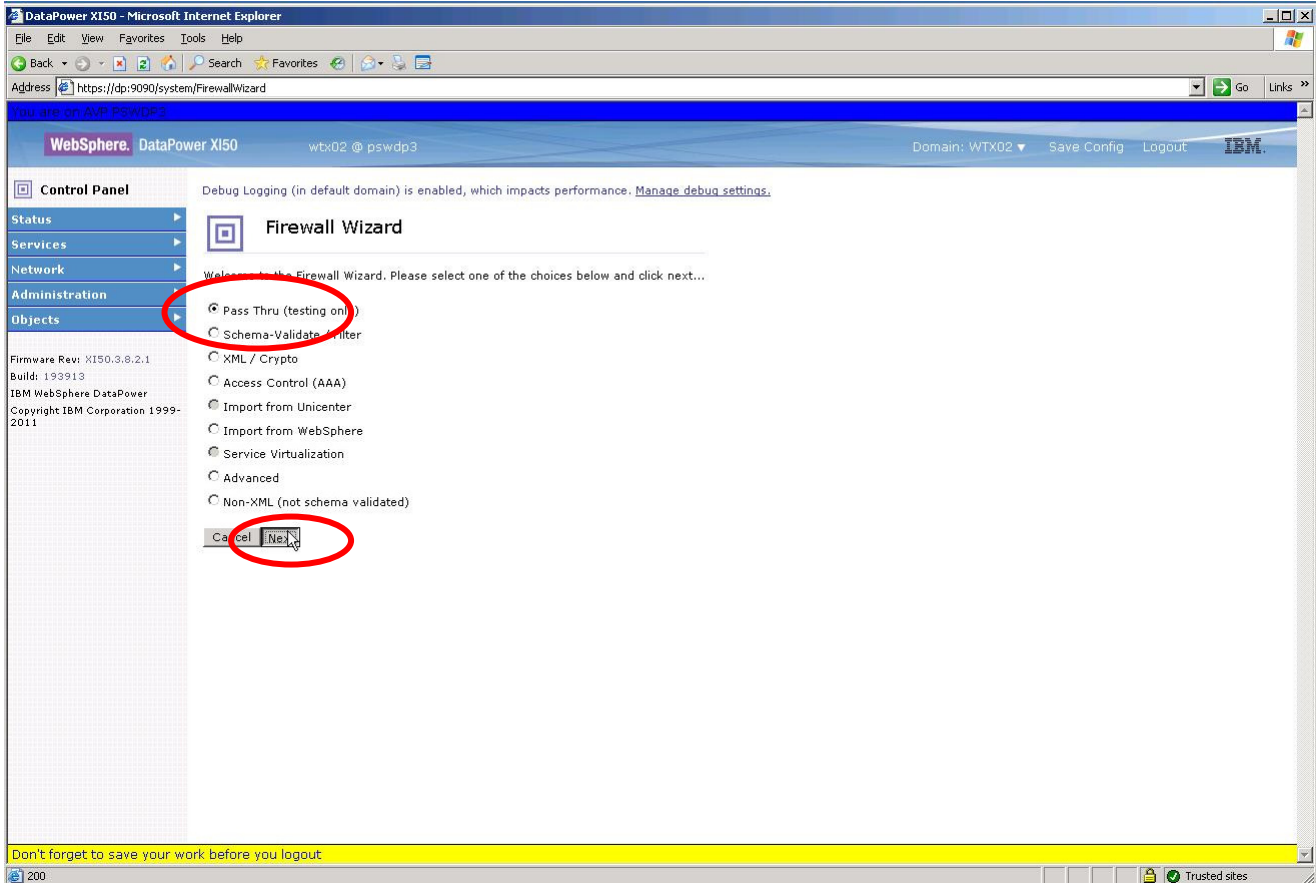
- You will now use Internet Explorer to take a look at the filesystem on the DataPower device to see what happened, and to configure a real service that uses the map. Minimize Design Studio for the next activity.

Part 6: Configuring DataPower XML Firewall To Use A MultiCard Map

- From the main DataPower window, click on the *XML Firewall* icon (wizard). On the page listing the existing XML Firewalls, click the *Add Wizard* button:



- You will create a new XML Firewall named *wtxMultiCard* with the following properties:
 - Pass Thru
 - Loopback-proxy
 - Port == **8xx5** [NOTE: xx == *your* workstation id!]
- The screenshots on the next pages capture the sequence of pages and steps:



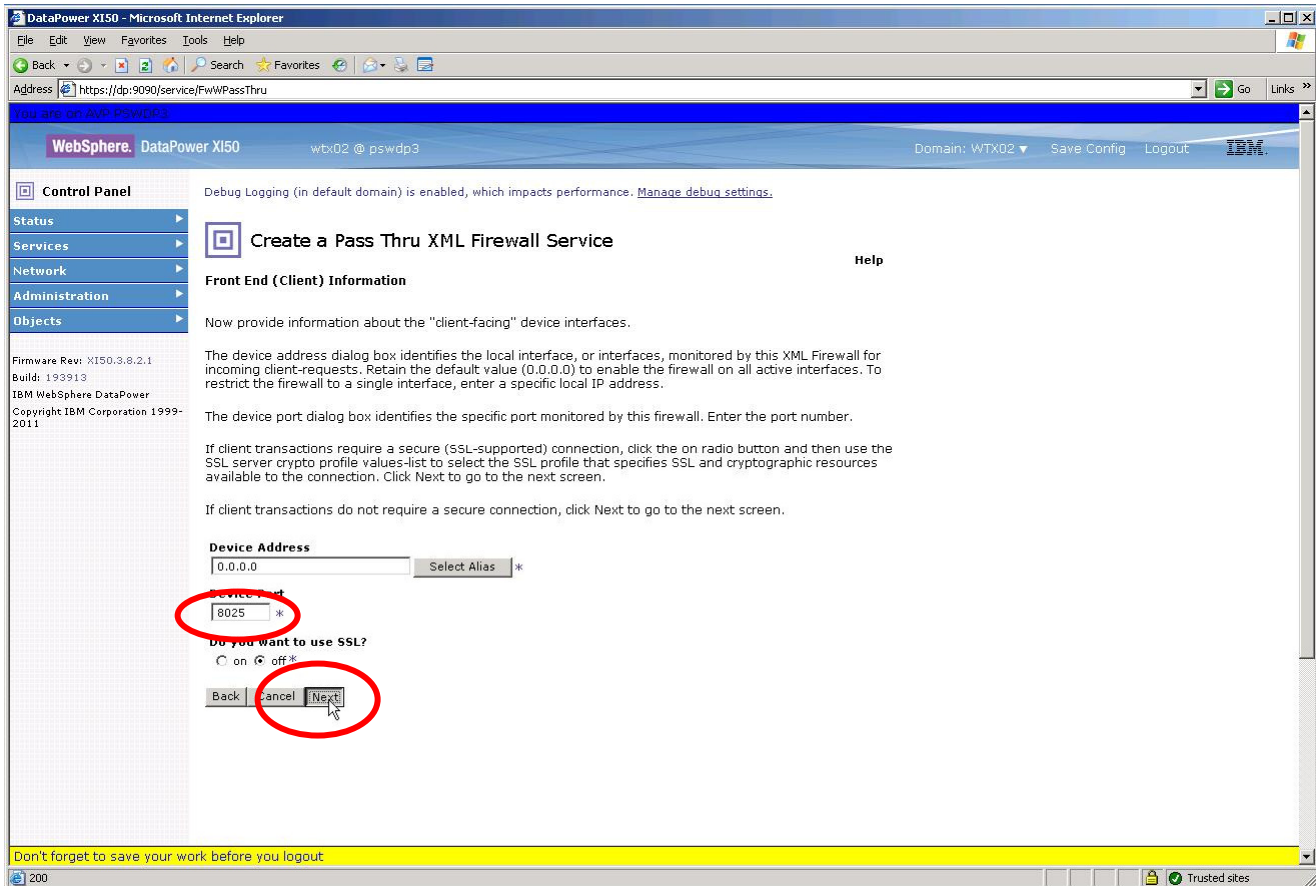
The screenshot shows the IBM DataPower XI50 administration console in a Microsoft Internet Explorer browser. The page title is "DataPower XI50 - Microsoft Internet Explorer". The address bar shows "https://dp:9090/service/FwPassThru". The console header includes "WebSphere. DataPower XI50" and "wtx02 @ pswdp3".

The main content area is titled "Create a Pass Thru XML Firewall Service" and includes a "Help" link. Under the heading "Select Service Type", there is a "Firewall Type" dropdown menu currently set to "loopback-proxy". Below the dropdown are "Back", "Cancel", and "Next" buttons. The "Next" button is circled in red. Text instructions explain that a static firewall supports a single web or application server, while a loopback firewall offers no server support.

On the left side, there is a "Control Panel" menu with options for Status, Services, Network, Administration, and Objects. Below this menu, the following information is displayed:

- Firmware Rev: XI50.3.8.2.1
- Build: 193913
- IBM WebSphere DataPower
- Copyright: IBM Corporation 1999-2011

A yellow banner at the bottom of the console reads "Don't forget to save your work before you logout". The browser's status bar at the bottom shows "200" and "Trusted sites".



- **NOTE:** Use your assigned port number! (8xx5)

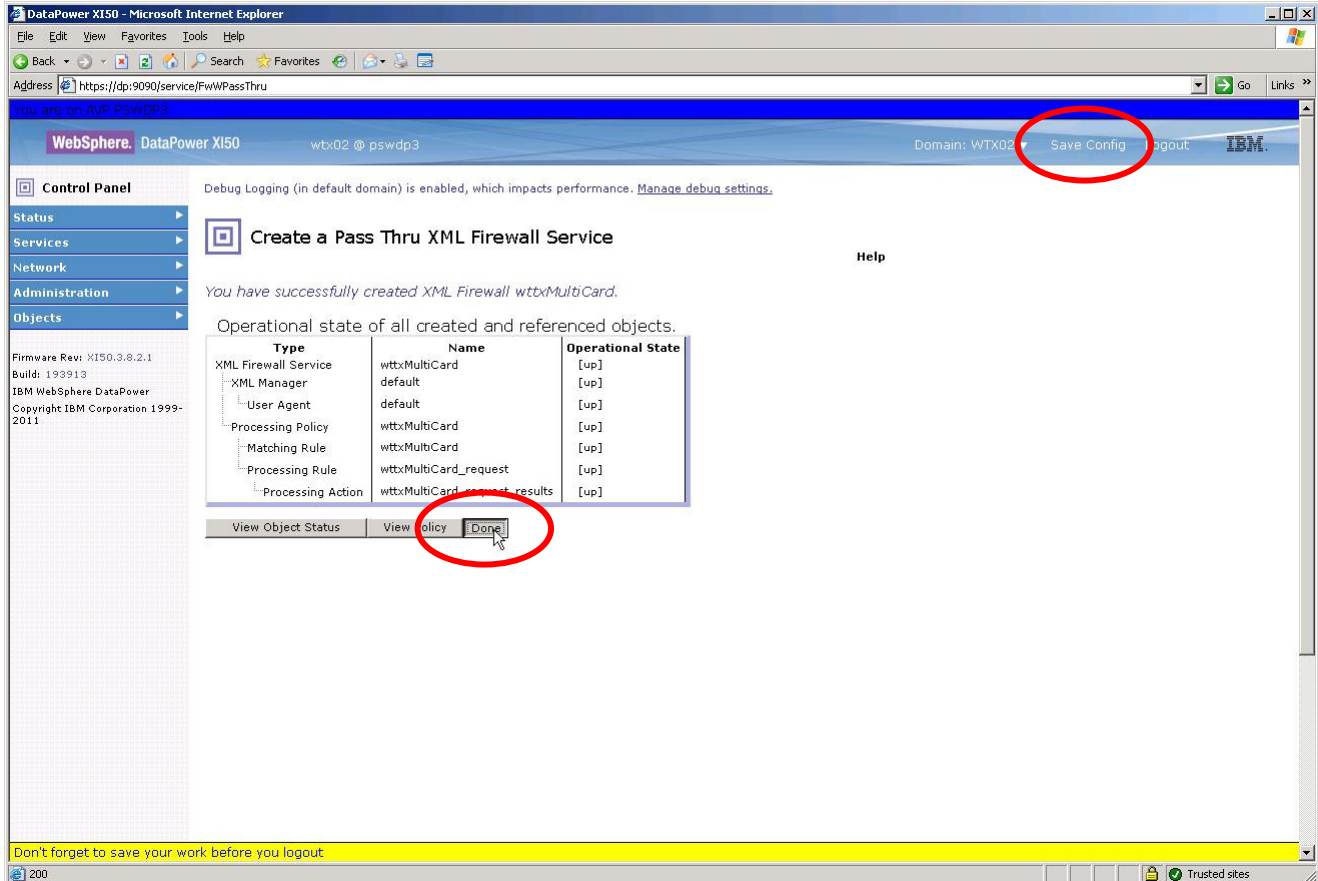
The screenshot shows the IBM DataPower XI50 administration console in a Microsoft Internet Explorer browser window. The address bar shows the URL <https://dp:9090/service/FwPassThru>. The page title is "WebSphere. DataPower XI50" and the user is logged in as "wtb02 @ pswdp3".

The main content area is titled "Create a Pass Thru XML Firewall Service" and includes a "Confirm Your Changes and Commit" section. The configuration details are as follows:

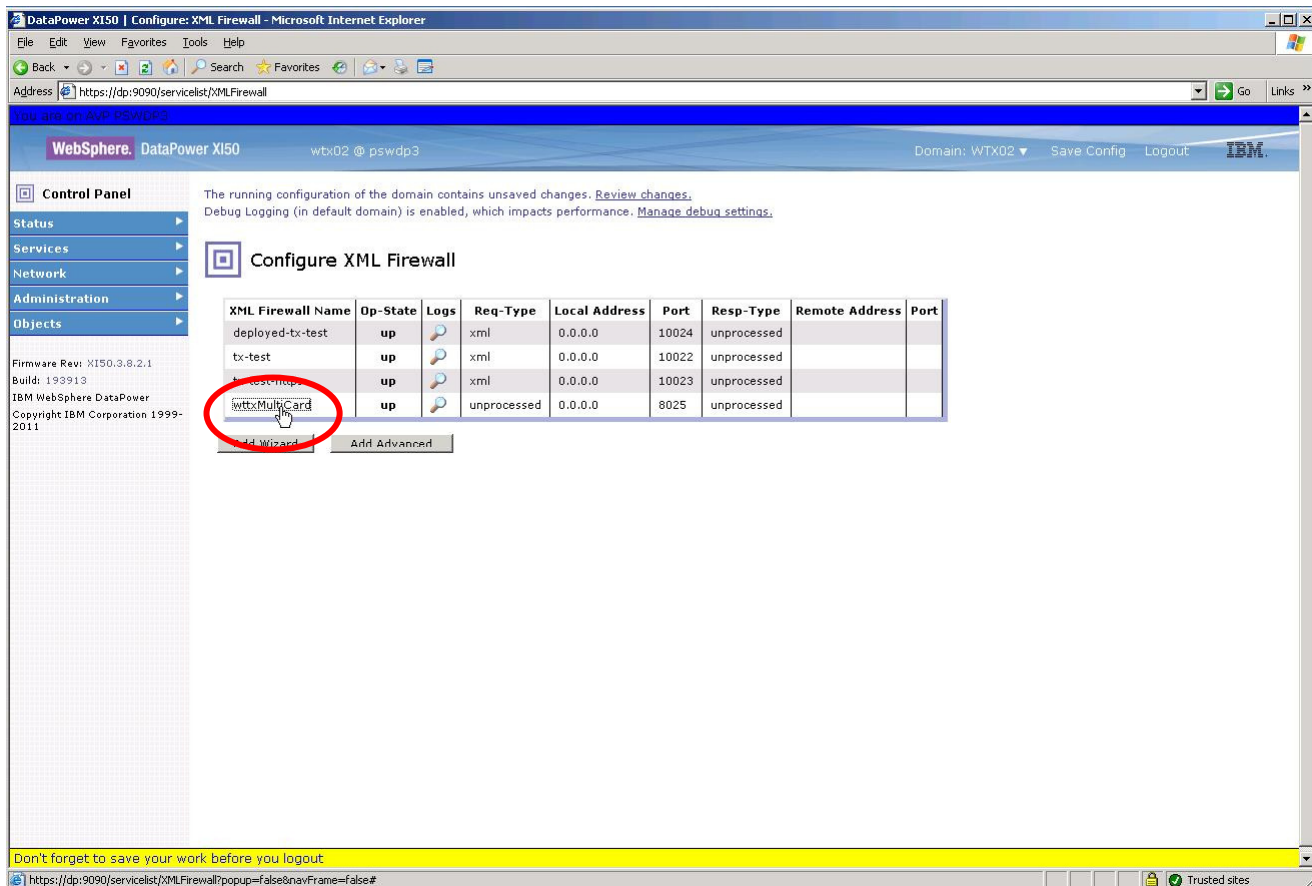
Firewall Name:	wtb:MultiCard
Firewall Type:	loopback-proxy
Server Address:	
Server Port:	
SSL Server Crypto Profile:	
Device Address:	0.0.0.0
Device Port:	8025
SSL Client Crypto Profile:	
Max. Message Size:	0
Override XML Manager parser limits:	off
Enable MMxDoS Protection:	off
Enable Message Tampering Protection:	off
Enable SQL Injection Protection:	off
Enable X-Virus Scanning:	off
Enable Dictionary Attack Protection:	off

At the bottom of the configuration area, there are buttons for "Back", "Cancel", "XML Threat Protection", and "Commit". The "Commit" button is circled in red.

A yellow banner at the bottom of the page reads: "Don't forget to save your work before you logout".

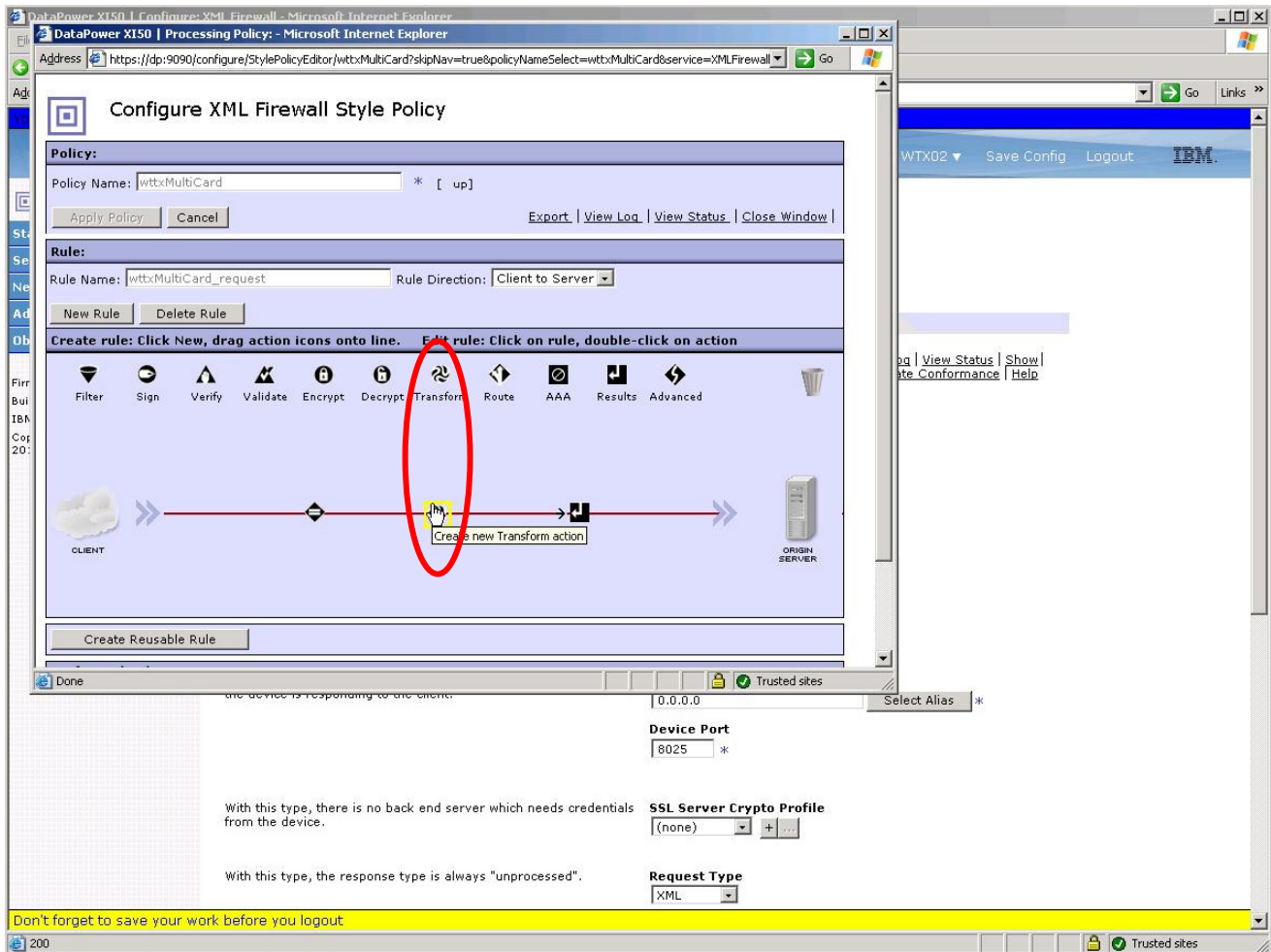


- After clicking Done, save your configuration (always a good idea when you've done some significant activities using the GUI). You have now created the basic service using the wizard, but you have additional configuration changes to implement. Return to the list of XML Firewalls by clicking on the *XML Firewall* icon, and then click on *wtxMultiCard* to edit the xml-firewall and continue with configuration:

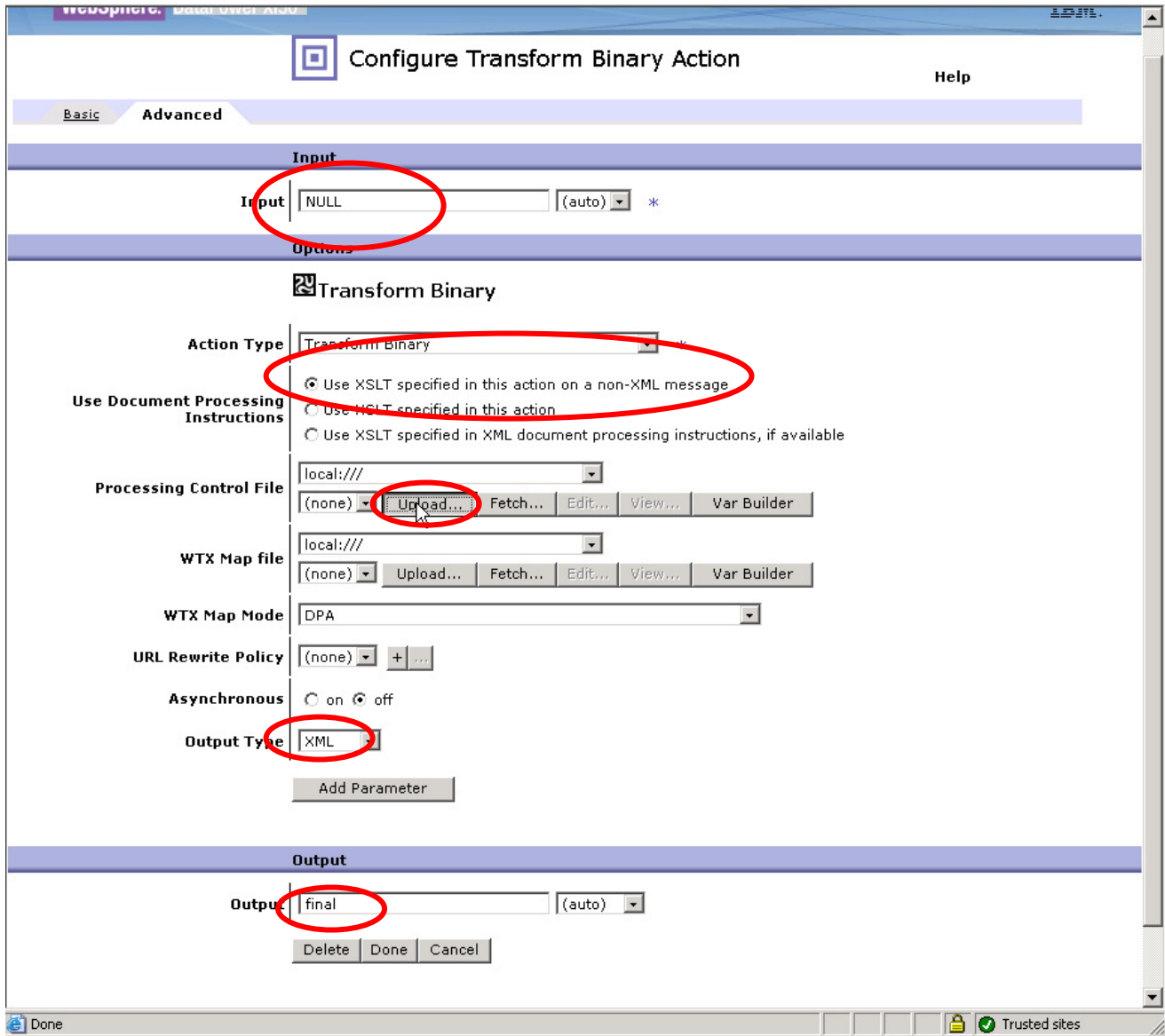


- The configuration page for the firewall service displays:

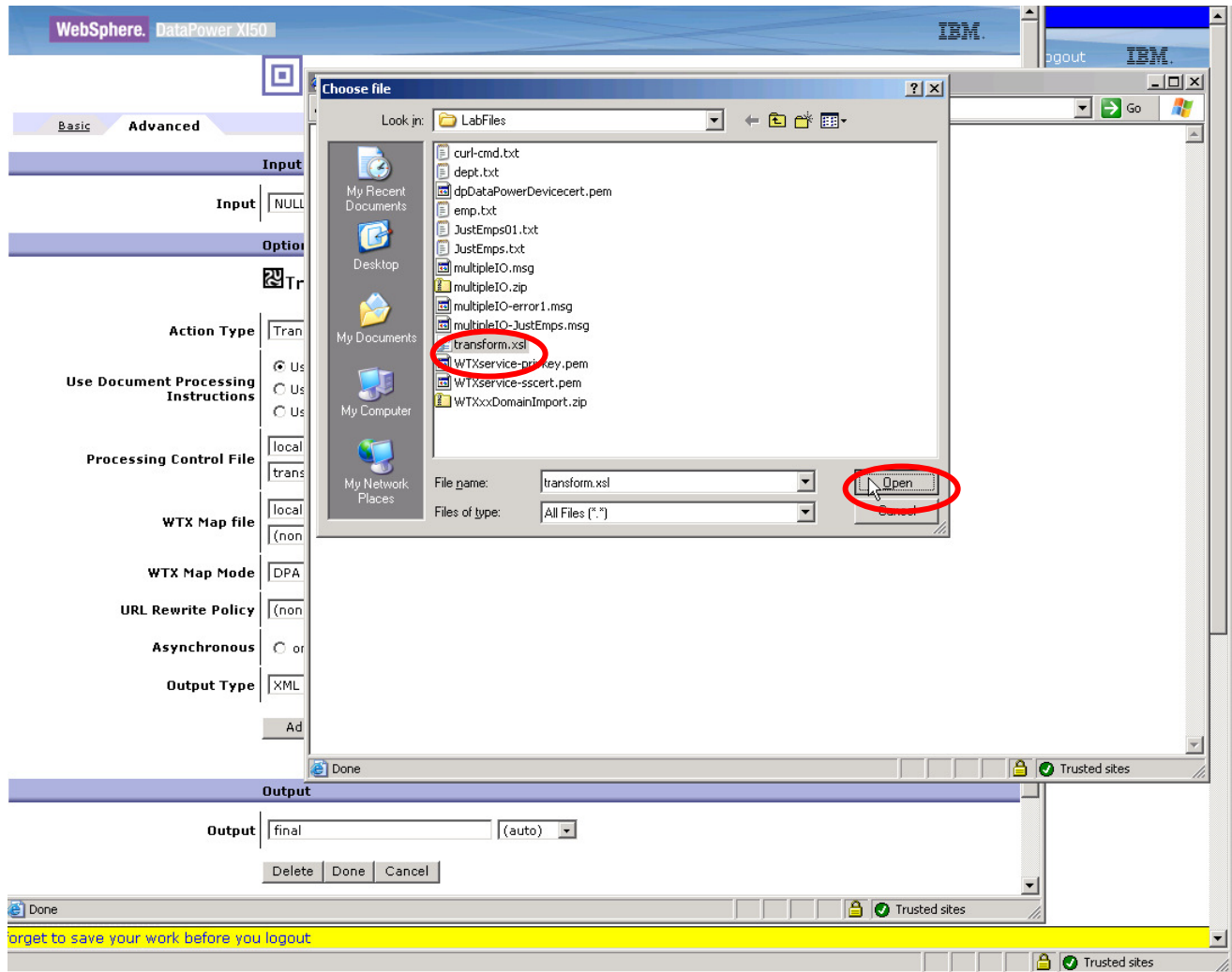
- First, change the *Request Type* to *XML* using the drop-down menu.
- Next, change *Request Attachments* to *Allow* using the drop-down menu. [This option only appears after you have changed the *Request Type* to *XML*.]
- You will now configure the processing policy for the service. Click on the *'...'* (*edit existing*) button to the right of the selected Firewall Policy (*wtxMultiCard*) that was created automatically for you during the initial XML Firewall creation steps. The initial processing policy created by the wizard is a simple set of actions, consistent with a loopback firewall: a *Match Action* which matches any URL [***], and a *Results Action* which returns the INPUT context.
 - **TIP:** If you hover the mouse over an action, the details of the action will display in a popup. This will come in handy...
- You will now augment this processing policy with a series of actions so that the service can execute the multcard WTX map against an input request arriving as a MIME message.
- Drag a *Transform* action onto the policy between the *Match* and *Results* actions. This transform will base64 encode a simple message as the first part of our output (MIME) message:

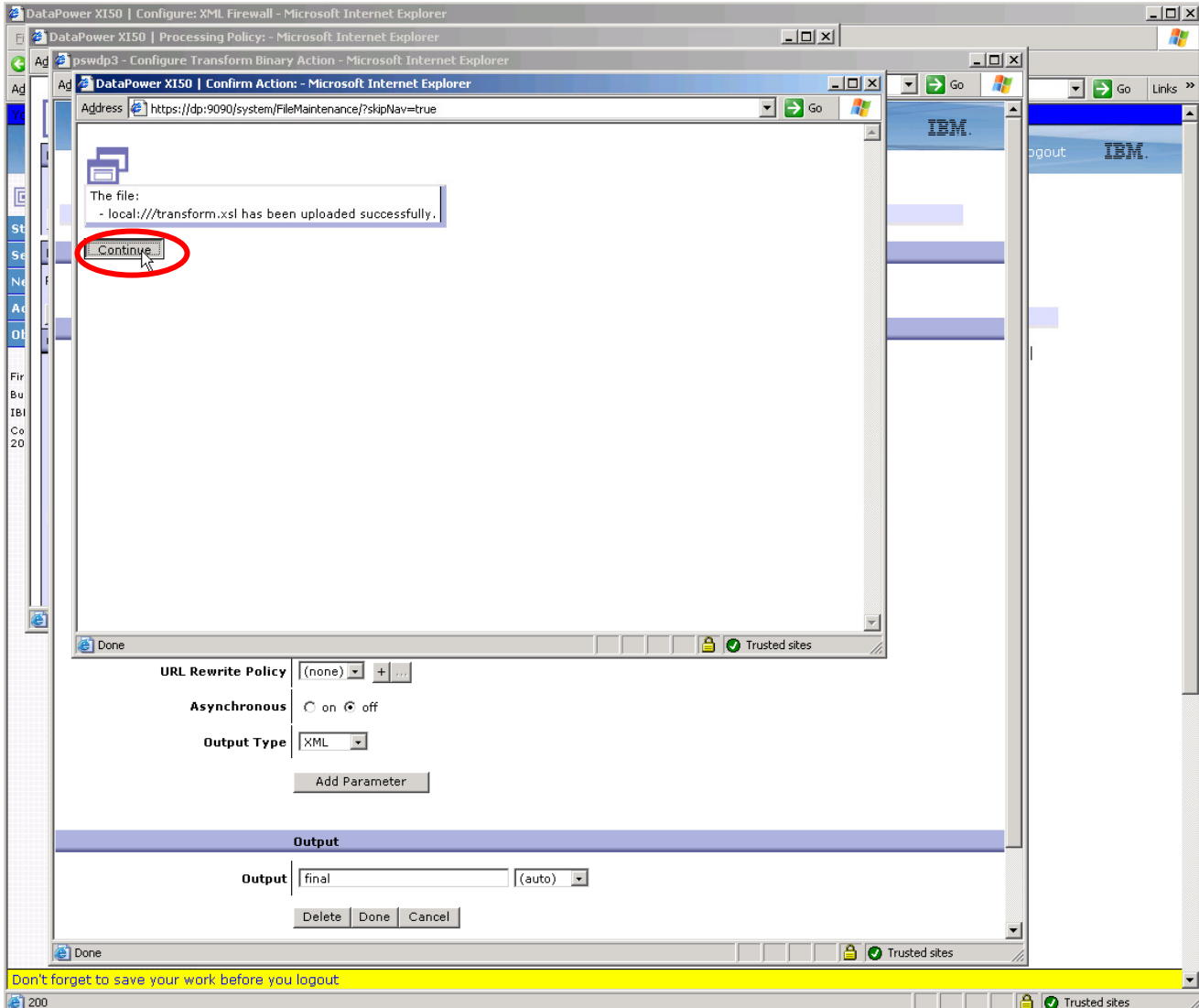


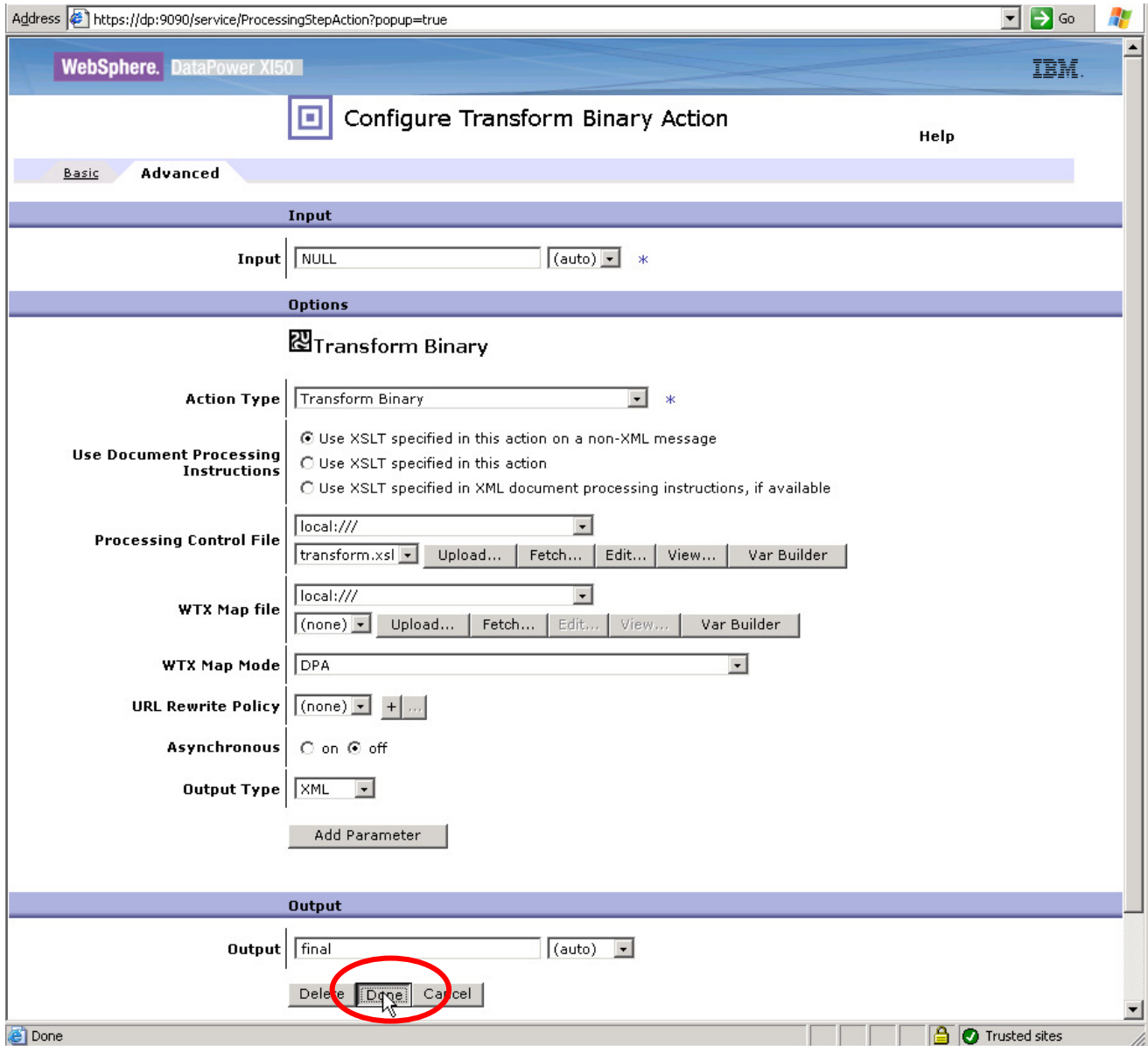
- Double-click on the *Transform* to open the configuration window for this action. Click on the *Advanced* tab. Click the radio button *Use XSLT specified in this action on a non-XML message* to the right of *Processing Control File*. The window will reconfigure itself to accommodate the new set of options available. Now, change the *Input* context to NULL (you will have to type that in as it is not available from the drop-down list), and the *Output* context to *final* (again, you will have to type that value into the field). Change *Output Type* to XML. Then click the *Upload* button to the right of "(none)" in the *Processing Control File*:



- A new dialog window appears. Use the Browse button of the subsequent dialog to upload the file *c:\LabFiles\transform.xml* to the *local:///* directory on DataPower. Click the *Upload*, *Continue*, and *Done* buttons on the following screens to apply the changes:







- Next, drag an *Advanced Action* icon onto the processing policy after the *Transform Action*. Double-click this action to configure a *Fetch* action. This action will retrieve the attachment from the **INPUT** context with content-id *dept.txt*, and place the results into a context named **dept_ctx** (you will once again have to **type** this value into the field). See the following sequence of screenshots:

Policy:
Policy Name: * [up]
Apply Policy Cancel Export View Log View Status Close Window

Rule:
Rule Name: Rule Direction: Client to Server
New Rule Delete Rule

Create rule: Click New, drag action icons onto line. Edit rule: Click on rule, double-click on action

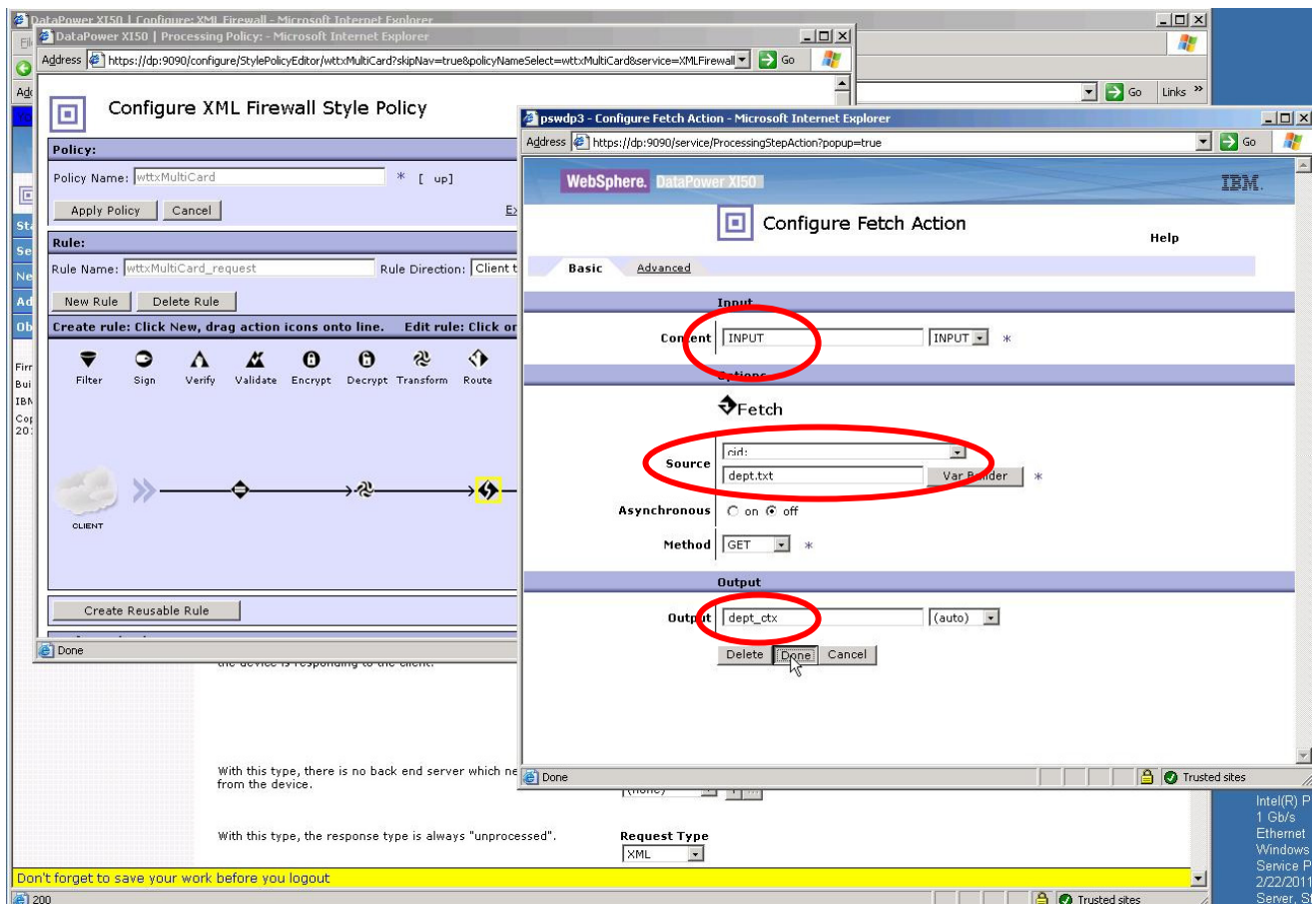
Filter Sign Verify Validate Encrypt Decrypt Transform Route AAA Results Advanced

CLIENT ORIGIN SERVER

Create new Advanced action

Create Reusable Rule

The screenshot displays two overlapping web browser windows from the IBM DataPower XI50 management console. The background window, titled "Configure XML Firewall Style Policy", shows a configuration page for a policy named "wtt:MultiCard". It includes fields for "Policy Name" and "Rule Name" (wtt:MultiCard_request), and a toolbar with icons for Filter, Sign, Verify, Validate, Encrypt, Decrypt, Transform, and Route. A flow diagram below the toolbar shows a sequence of actions starting from a "CLIENT" icon. The foreground window, titled "Configure Action", is a modal dialog for selecting an action type. The "Fetch" option is selected and circled in red. Other options include Event-sink, Extract Using XPath, For-each, Header Rewrite, and Log. The "Fetch" description states: "The Fetch action retrieves resources (such as an XML file) from specified URL for use within a processing rule. The result of the Fetch is placed in the Output context." At the bottom of the foreground window, there is a "Request Type" dropdown menu set to "XML". A yellow banner at the bottom of the screen reads "Don't forget to save your work before you logout".



- We will now add a second *Fetch* action to retrieve the list of Employees – the attachment contained in the input message with content-id *emp.txt*. See the screenshots for the sequence, again paying particular attention to ensure that the Input and Output contexts, and the attachment URIs are specified as shown:

Address <https://dp:9090/configure/StylePolicyEditor/wtxMultiCard?skipNav=true&policyNameSelect=wtxMultiCard&service=XML> Go

Configure XML Firewall Style Policy

Policy:
Policy Name: * [up]
Apply Policy Cancel Export View Log View Status Close Window

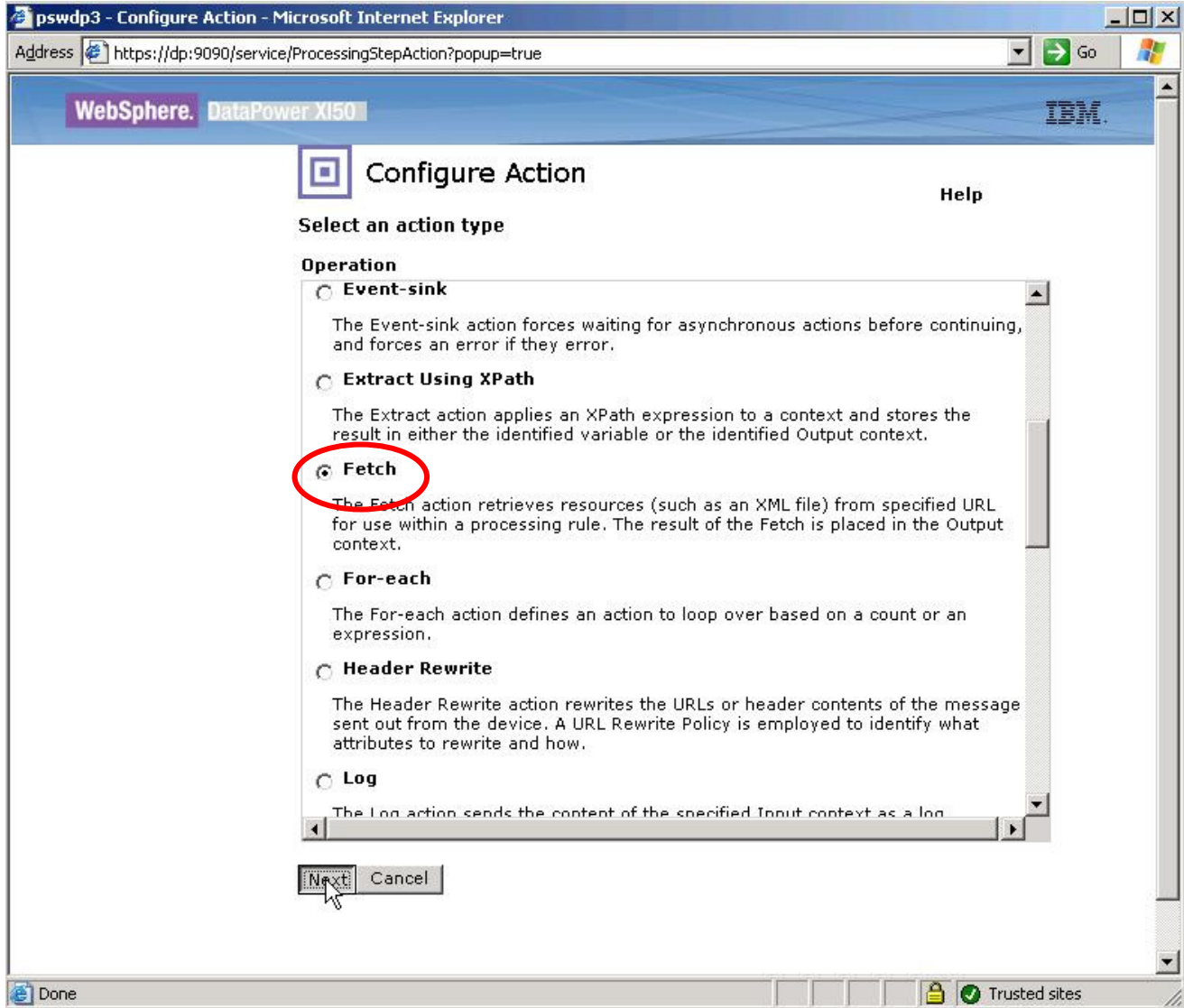
Rule:
Rule Name: Rule Direction:
New Rule Delete Rule

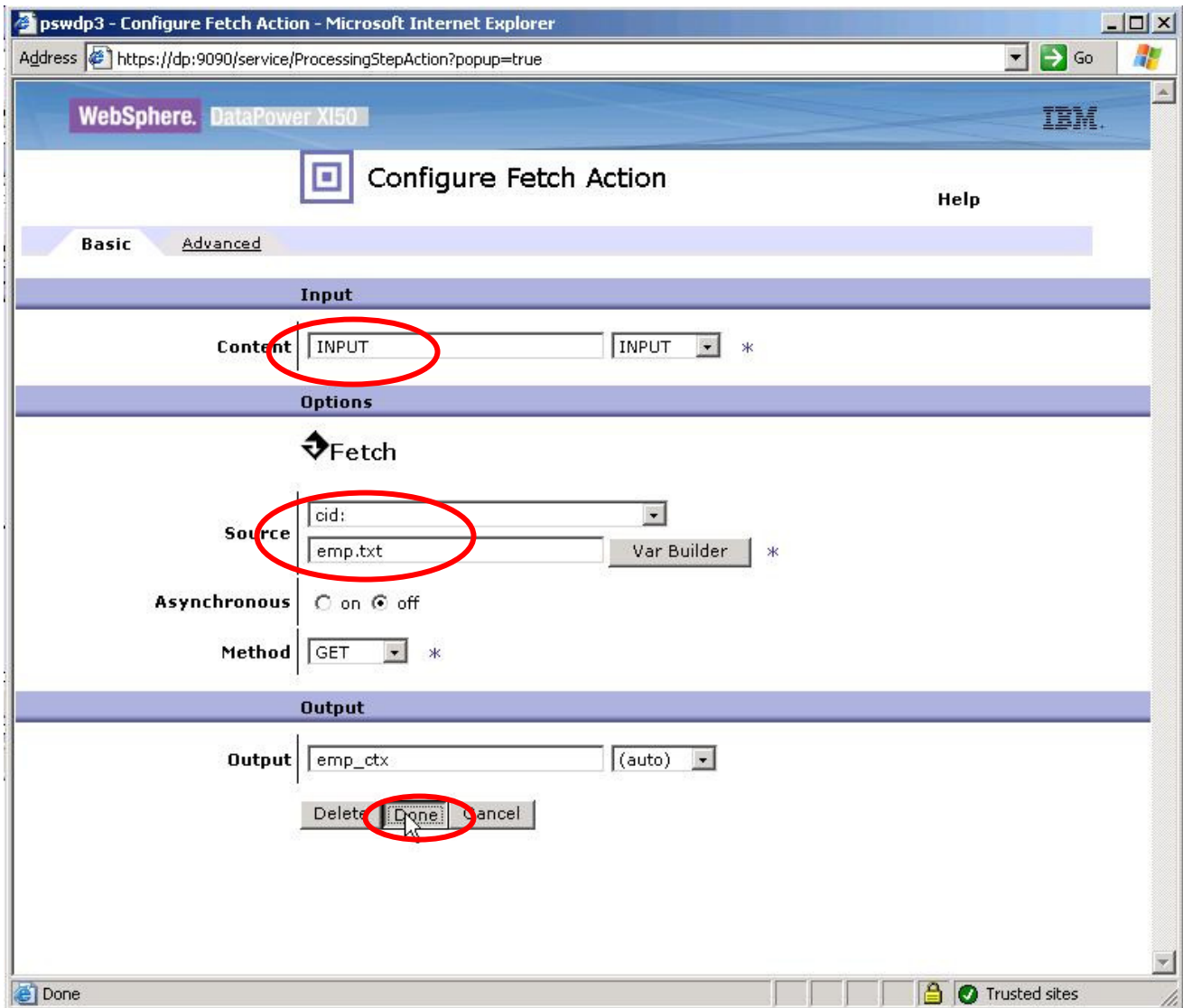
Create rule: Click New, drag action icons onto line. Edit rule: Click on rule, double-click on action

Filter Sign Verify Validate Encrypt Decrypt Transform Route AAA Results Advanced

CLIENT → [Filter] → [Sign] → [Verify] → [Validate] → [Encrypt] → [Decrypt] → [Transform] → [Route] → [AAA] → [Results] → [Advanced] → ORIGIN SERVER

Done Trusted sites





- Now, you will add a *Binary Transform Action* after the two *Fetch* actions. This action will invoke the execution of our WTX map. Drag another *Advanced Action* onto the processing policy just before the *Results Action*, and then configure the action as shown in these screenshots.
 - **NOTE:** The *initial* configuration of this action from the processing policy wizard does **not** permit you to configure the sources of the input and output cards; you will make those changes in a subsequent step.

DataPower XI50 | Processing Policy: - Microsoft Internet Explorer

Address <https://dp:9090/configure/StylePolicyEditor/wtxMultiCard?skipNav=true&policyNameSelect=wtxMultiCard&service=XMLFirewallServ> Go

Configure XML Firewall Style Policy

Policy:

Policy Name: * [up]

[Export](#) | [View Log](#) | [View Status](#) | [Close Window](#)

Rule:

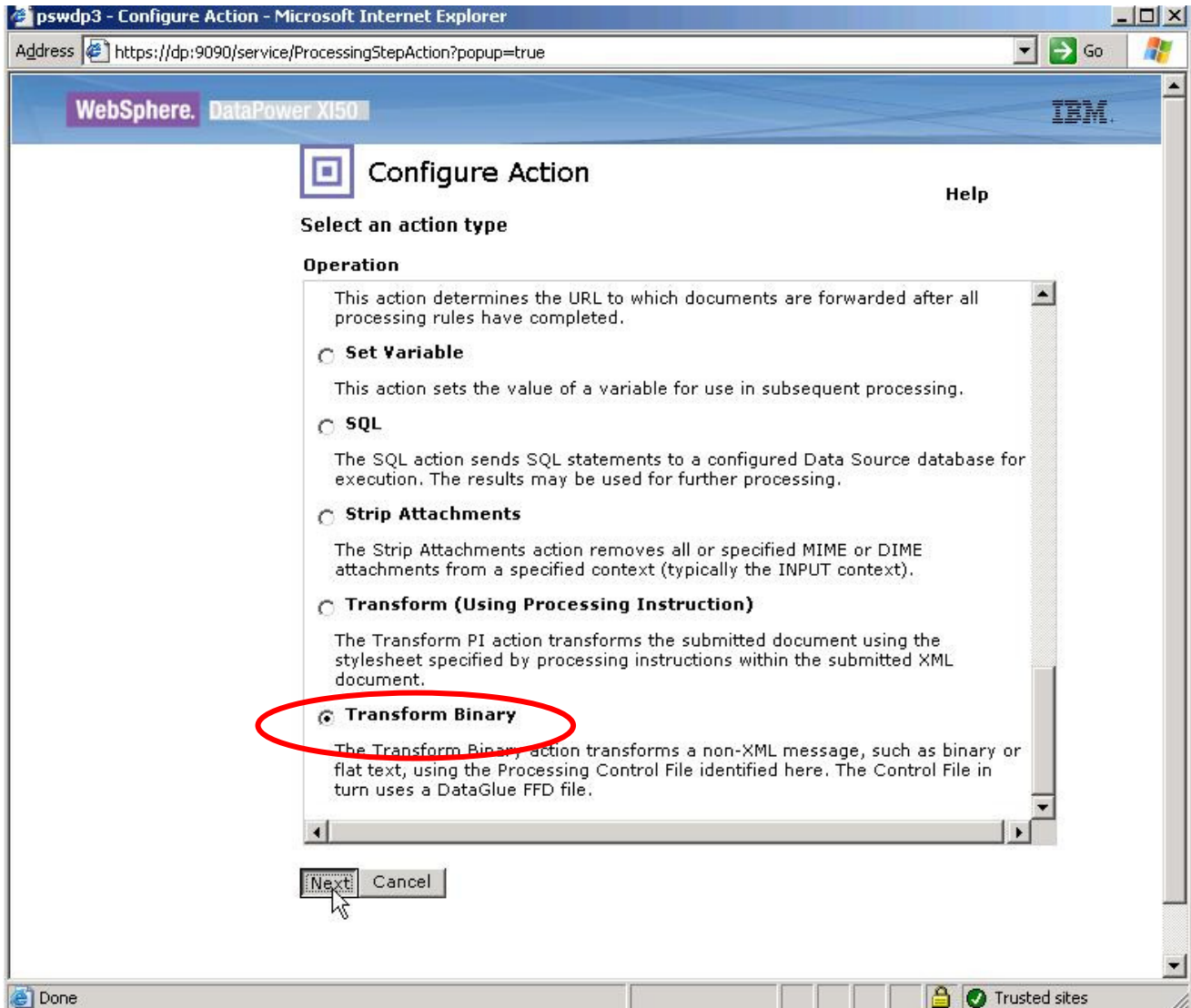
Rule Name: Rule Direction:

Create rule: Click New, drag action icons onto line. Edit rule: Click on rule, double-click on action

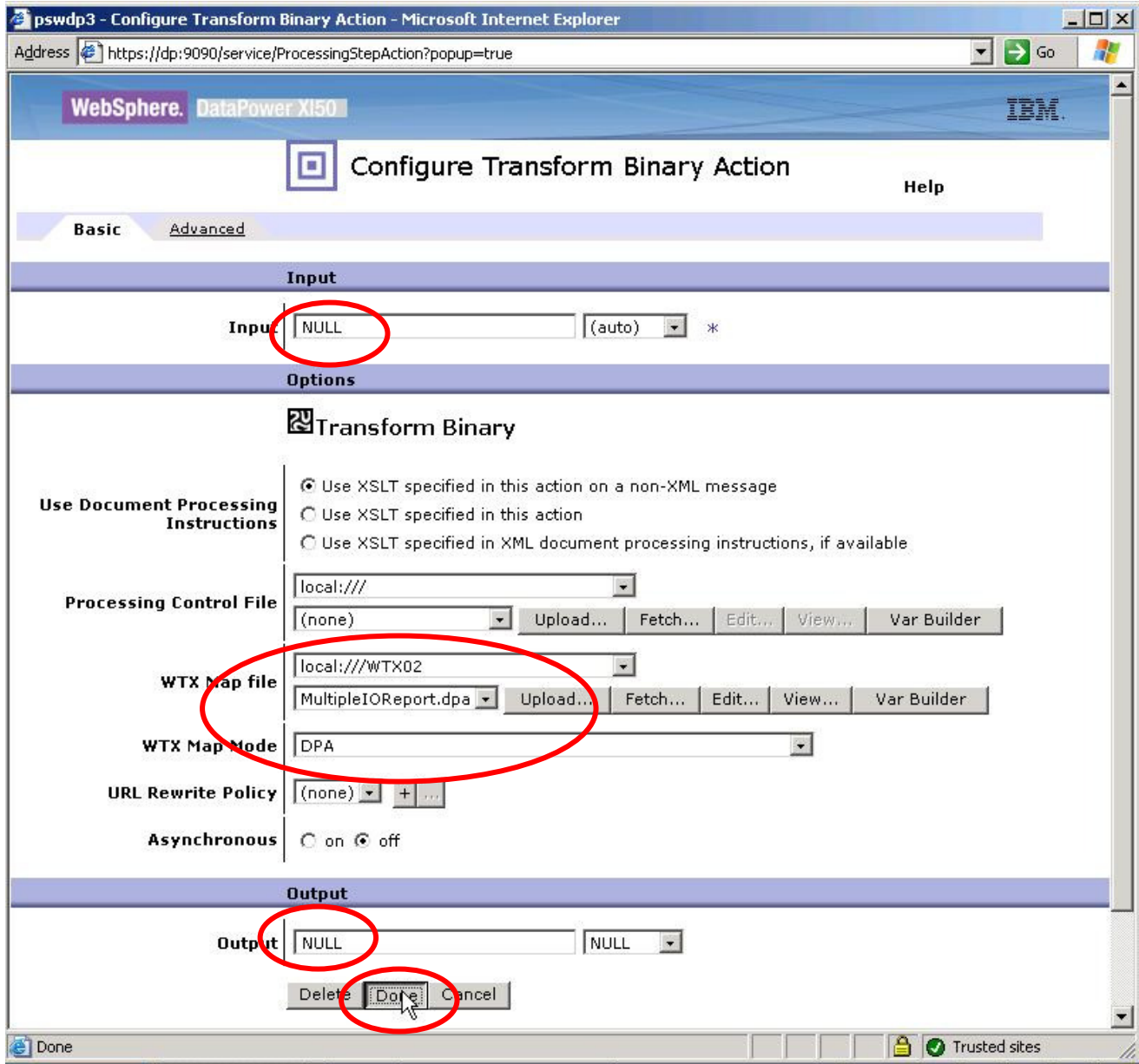
Filter Sign Verify Validate Encrypt Decrypt Transform Route AAA Results Advanced

CLIENT → [Filter] → [Sign] → [Verify] → [Validate] → [Encrypt] → [Decrypt] → [Transform] → [Route] → [AAA] → [Results] → [Advanced] → ORIGIN SERVER

Done Trusted sites



- Select the map file (*multipleIOReport.dpa*) from *local:///WTXxx* (this is the location to which you deployed the map file from Design Studio earlier in the lab). Ensure that the Input and Output contexts are both *NULL*, and that the map mode is *DPA*. You will finish configuring the map in a short while – after we finish adding and configuring the other required policy actions.



- Finally, you need to add two *Results* actions to complete the assembly of all the required parts of our response into the context (*final*) that you will return to the client application. Notice that you will set the destination for these *Results* actions to be *attachments* of that output context, and that you will have to **type** the names of the *Input* contexts in each case. [These two contexts will be *created* as the location of the outputs of our WTX map, but the configuration wizard doesn't know that yet...]

DataPower XI50 | Processing Policy: - Microsoft Internet Explorer

Address <https://dp:9090/configure/StylePolicyEditor/wtxMultiCard?skipNav=true&policyNameSelect=wtxMultiCard&service=XMLFirewallServ> Go

Configure XML Firewall Style Policy

Policy:

Policy Name: * [up]

[Export](#) | [View Log](#) | [View Status](#) | [Close Window](#)

Rule:

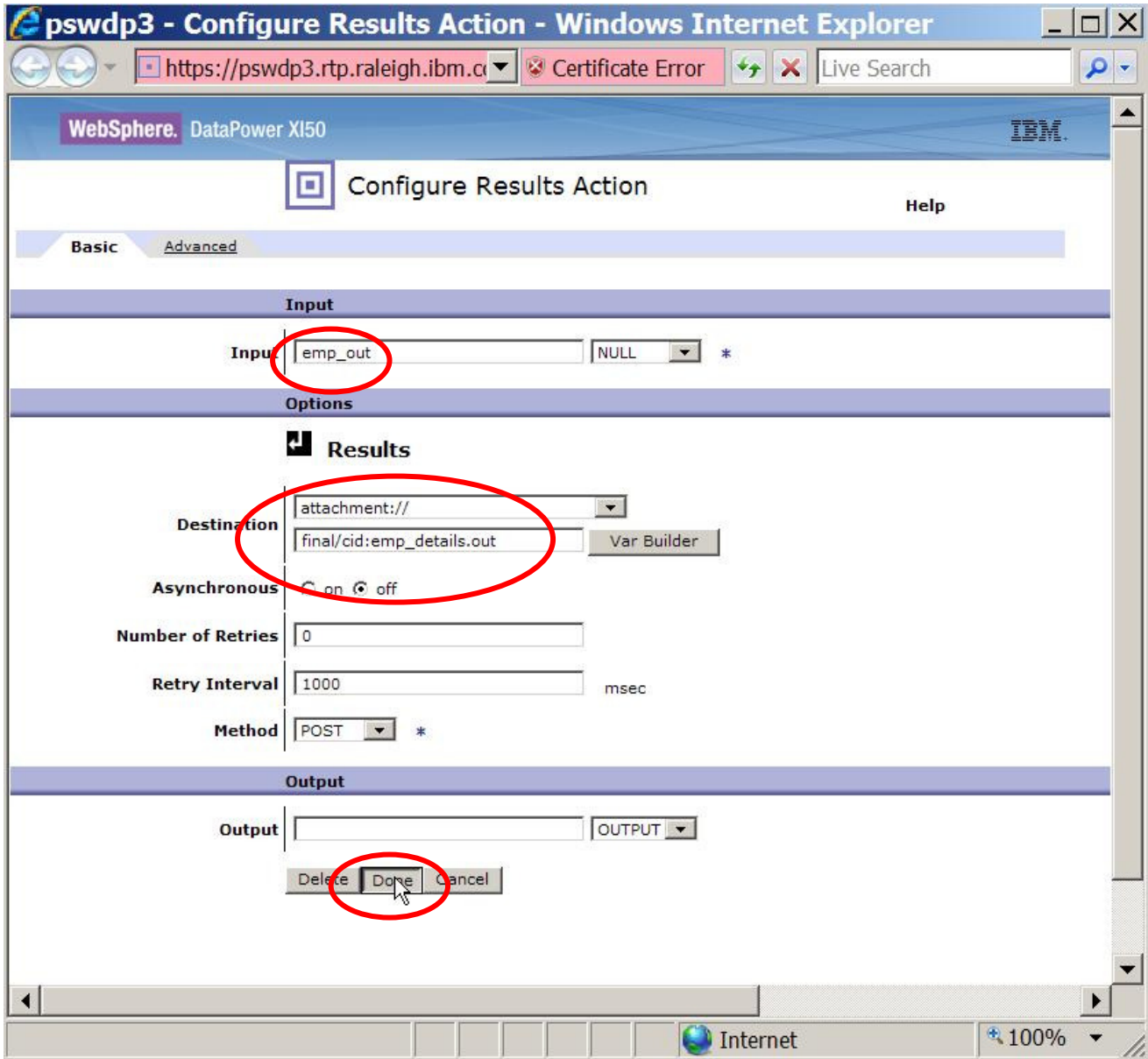
Rule Name: Rule Direction:

Create rule: Click New, drag action icons onto line. Edit rule: Click on rule, double-click on action

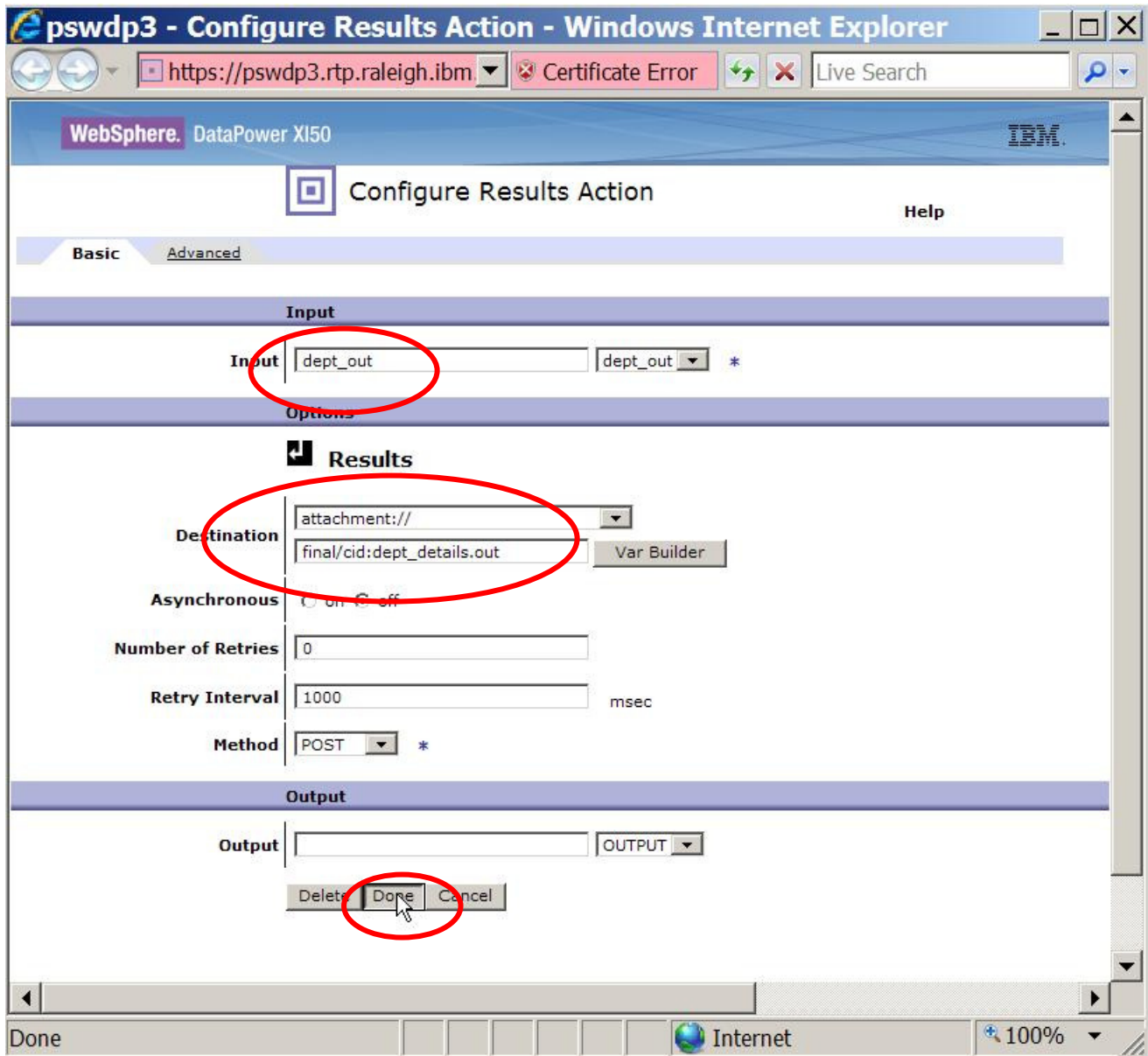
Filter Sign Verify Validate Encrypt Decrypt Transform Route AAA **Results** Advanced

CLIENT → [Filter] → [Sign] → [Verify] → [Validate] → [Encrypt] → [Decrypt] → [Transform] → [Route] → [AAA] → [Results] → [Advanced] → ORIGIN SERVER

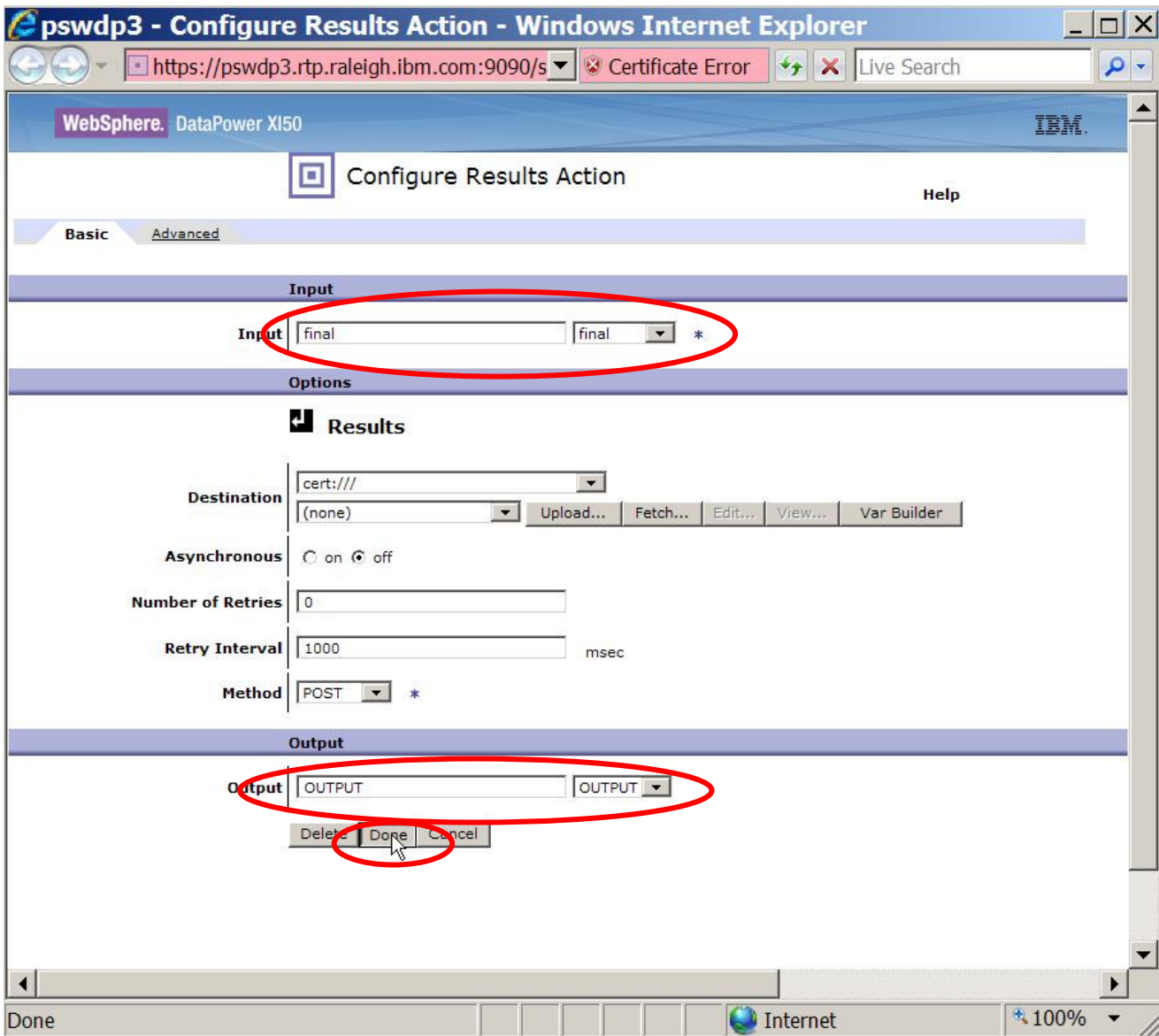
Done Trusted sites



- Drag a second *Results* action onto the policy after the one you just finished adding, and configure it:



- Finally, reconfigure the **last** *Results* action (the existing one added by the wizard) to return the context *final* to **OUTPUT**:



- The completed policy should look similar to this:

Policy:
Policy Name: wtxMultiCard *
Apply Policy Cancel Export | View Log | View Status | Close Window

Rule:
Rule Name: wtxMultiCard_request Rule Direction: Client to Server
New Rule Delete Rule

Create rule: Click New, drag action icons onto line. Edit rule: Click on rule, double-click on action

Filter Sign Verify Validate Encrypt Decrypt Transform Route AAA Results Advanced

CLIENT

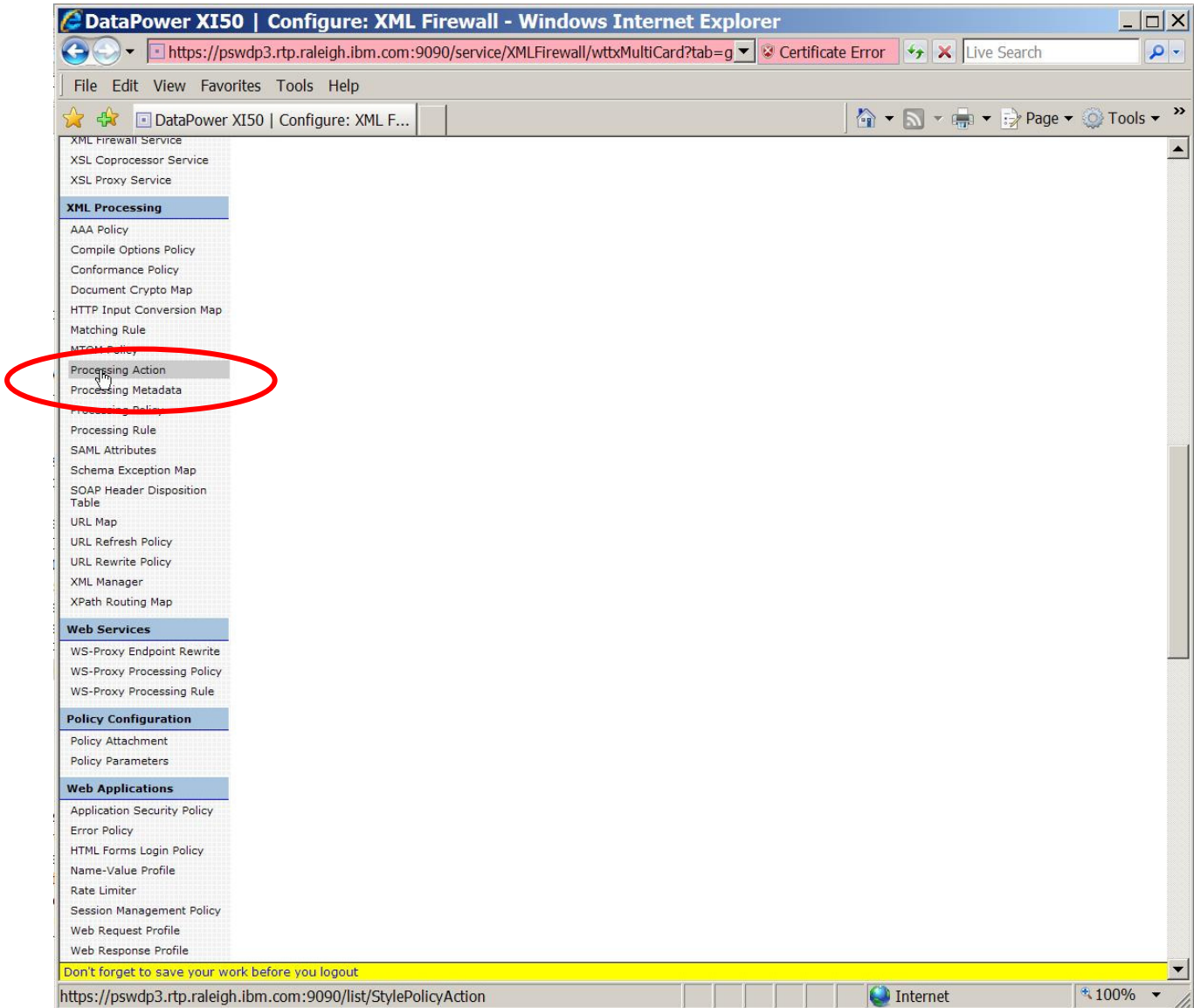
Action: Transform Binary
wtxMultiCard_request_xformbin_0
Action Type: xformbin
Input : NULL
WTX Map file : local:///WTX02 /MultipleIOReport.dpa
WTX Map Mode : dpa
Output : NULL
Locate Named Inputs and Outputs : default
Output Type : default
Asynchronous : off

Order	Rule Name	Direction	Actions	
↑ ↓	wtxMultiCard_request	Client to Server	Filter Sign Verify Validate Encrypt Decrypt Transform Route AAA Results Advanced	delete rule

Scroll to top

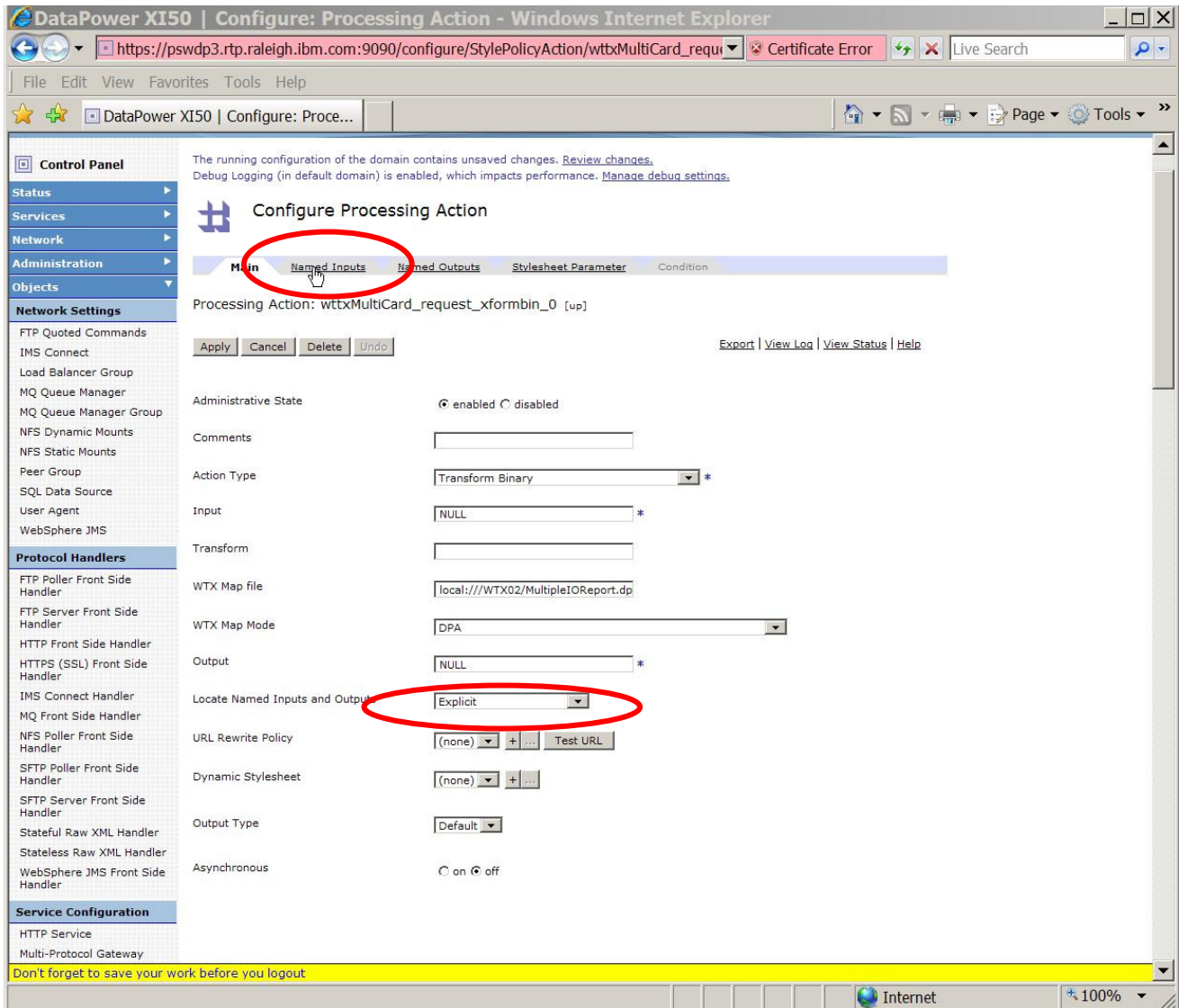
- **TAKE NOTE** of the name of the *Binary Transform Action*, since you will need to know the name to finish configuring that action (which you will do next).
 1. Recall that you can discover the action's name by hovering the mouse over the *Binary Transform Action* in the 'Configured Rules' section of the page.
- Click the *Apply Policy* button.
- 'Before you leave this screen and finish configuring the *Binary Transform* action that will run the map, you may want to spend a minute to understand the processing logic for this service. The processing policy consists of eight actions for traffic originating from a client:
 1. A *Match* action decides which message are processed by this rule
 2. A (*Binary*) *Transform* action produces a context (*final*) where we will assemble our output 'pieces'
 3. Two *Fetch* actions which (in turn):
 - a. retrieve *departments* and
 - b. retrieve *employees* from the corresponding attachment in the incoming message

4. A *Binary Transform* action which runs the WTX map
 5. Two *Results* actions which (in turn):
 - a. Move the output from the map's first output card to an attachment of the *final* context
 - b. Move the output from the map's second output card a second attachment of the *final* context
 6. A Final *Results* action which sends the *final* context (now completely 'assembled') to the context named *OUTPUT*, which is returned to the client application
- Now, click *Close Window*, and then *Apply* on the main configuration screen for the XML Firewall. Save your configuration.
 - Next, you need to make further configuration changes to the *Binary Transform Action* that executes the multcard WTX map. As mentioned earlier, the input and output card mapping can **not** be done from the XML Firewall wizard; these changes must be configured from the object itself.
 - **Reminder:** you will need the name of the action to select the proper action from the list.
 - From the left-hand navigation menu, click on (open) the *Objects* menu, then select *Processing Action* (under *XML Processing* section). Click on the Binary Transform Action's name to edit:



Service	Name	Status	Mode	Input/Output
Service Configuration	tx->xform-Ch01-1	saved	up	xform
	tx->xform-Ch01-3	saved	up	xformbin
	tx->xform-Ch03-1	saved	up	xformbin
	tx->xform-Ch03-2	saved	up	xformbin
	tx->xform-dpa	saved	up	xformbin
	tx->xform-ef	saved	up	xformbin
	tx->xform-MultiInOut	saved	up	xformbin
	tx->xform-MultiOut	saved	up	xformbin
	tx->xform-trace	saved	up	xformbin
	tx->xform-xml-03-01	saved	up	xformbin
	tx->xform-xml-03-02	saved	up	xformbin
	validate-input-against-wsdl	saved	up	validate
	validate-result-against-wsdl	saved	up	validate
	wtbMultiCard_request_fetch_0	saved	up	fetch
	wtbMultiCard_request_fetch_1	saved	up	fetch
wtbMultiCard_request_results_1	modified	up	results	
wtbMultiCard_request_results_2	new	up	results	
wtbMultiCard_request_results_2	new	up	results	
wtbMultiCard_request_strip-attachments_0	saved	up	strip-attachments	
wtbMultiCard_request_strip-attachments_1	saved	up	strip-attachments	
wtbMultiCard_request_xformbin_0	saved	up	xform	
wtbMultiCard_request_xformbin_0	saved	up	xformbin	

- Change the *Locate Named Inputs and Outputs* field to *Explicit* using the drop-down list:



- Navigate to the Named Inputs tab, and then add two inputs as shown:

DataPower XI50 | Configure: Processing Action - Windows Internet Explorer

https://pswdp3.rtp.raleigh.ibm.com:9090/configure/StylePolicyAction/wtxMultiCard_requi

Control Panel

The running configuration of the domain contains unsaved changes. [Review changes.](#)
Debug Logging (in default domain) is enabled, which impacts performance. [Manage debug settings.](#)

Configure Processing Action

Main | **Named Inputs** | Named Outputs | Stylesheet Parameter | Condition

Processing Action: wtxMultiCard_request_xformbin_0 [up]

Apply | Cancel | Delete | Undo

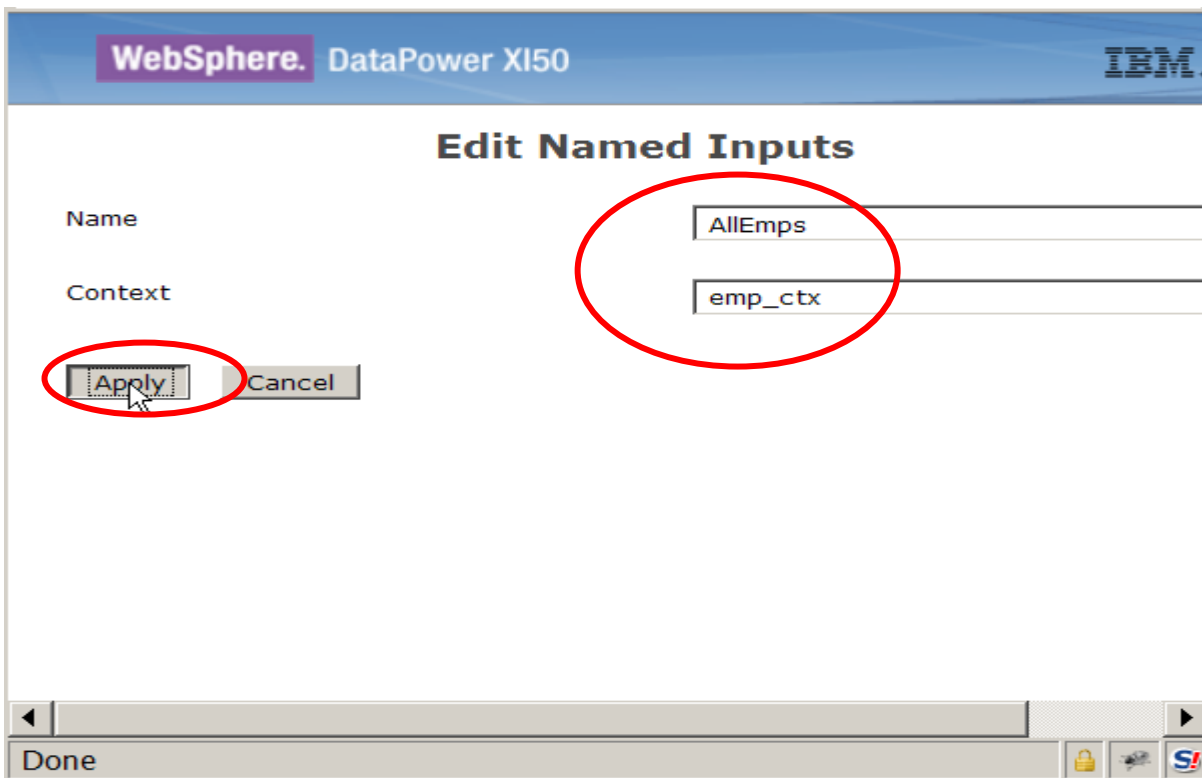
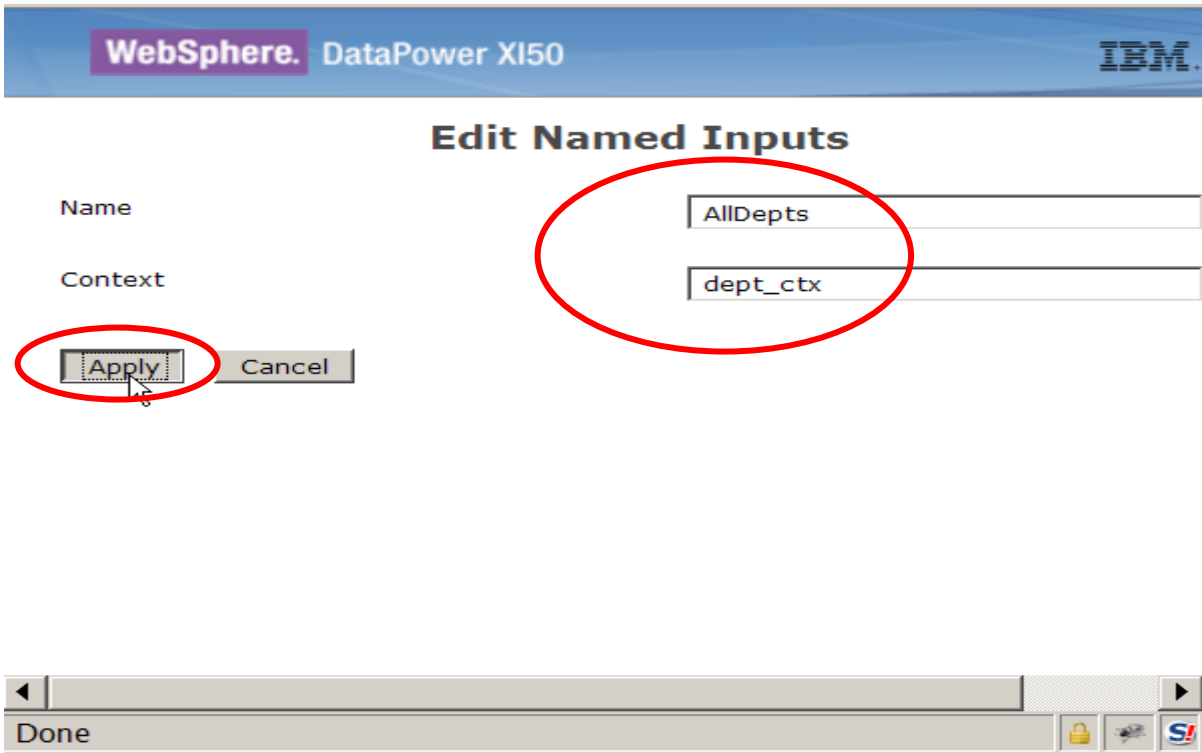
Export | View Log | View Status | Help

Name	Context
(empty)	

Add

Don't forget to save your work before you logout

Done



- From the *Named Outputs* tab, configure two output cards:

DataPower XI50 | Configure: Processing Action - Windows Internet Explorer

https://pswdp3.rtp.raleigh.ibm.com:9090/configure/StylePolicyAction/wtxMultiCard_requi

File Edit View Favorites Tools Help

DataPower XI50 | Configure: Proce...

Control Panel

The running configuration of the domain contains unsaved changes. [Review changes.](#)
Debug Logging (in default domain) is enabled, which impacts performance. [Manage debug settings.](#)

Configure Processing Action

Main Named Input **Named Outputs** Stylesheet Parameter Condition

Processing Action: wtxMultiCard_request_xformbin_0 [wp]

Apply Cancel Delete Undo Export View Log View Status Help

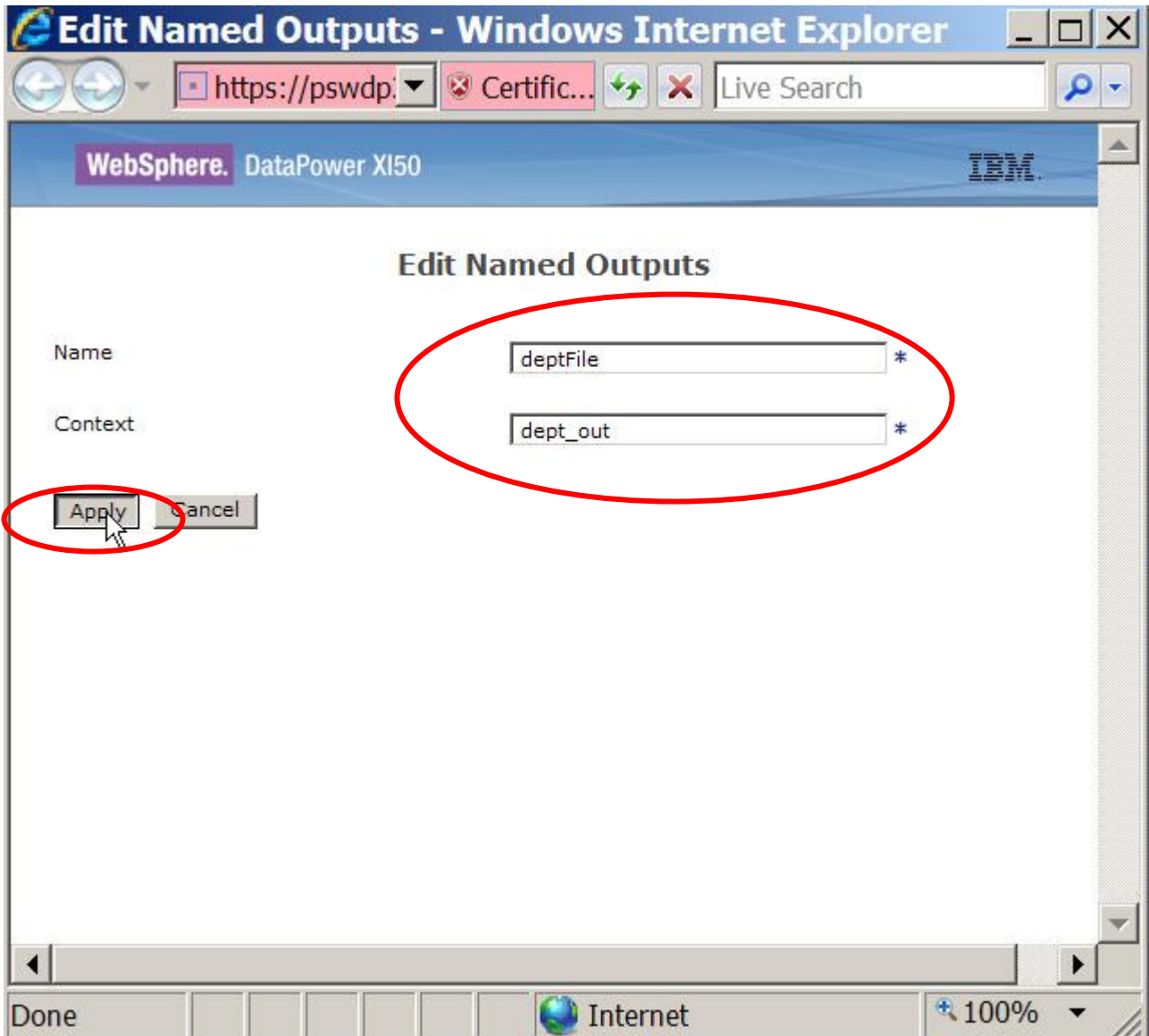
Named Outputs

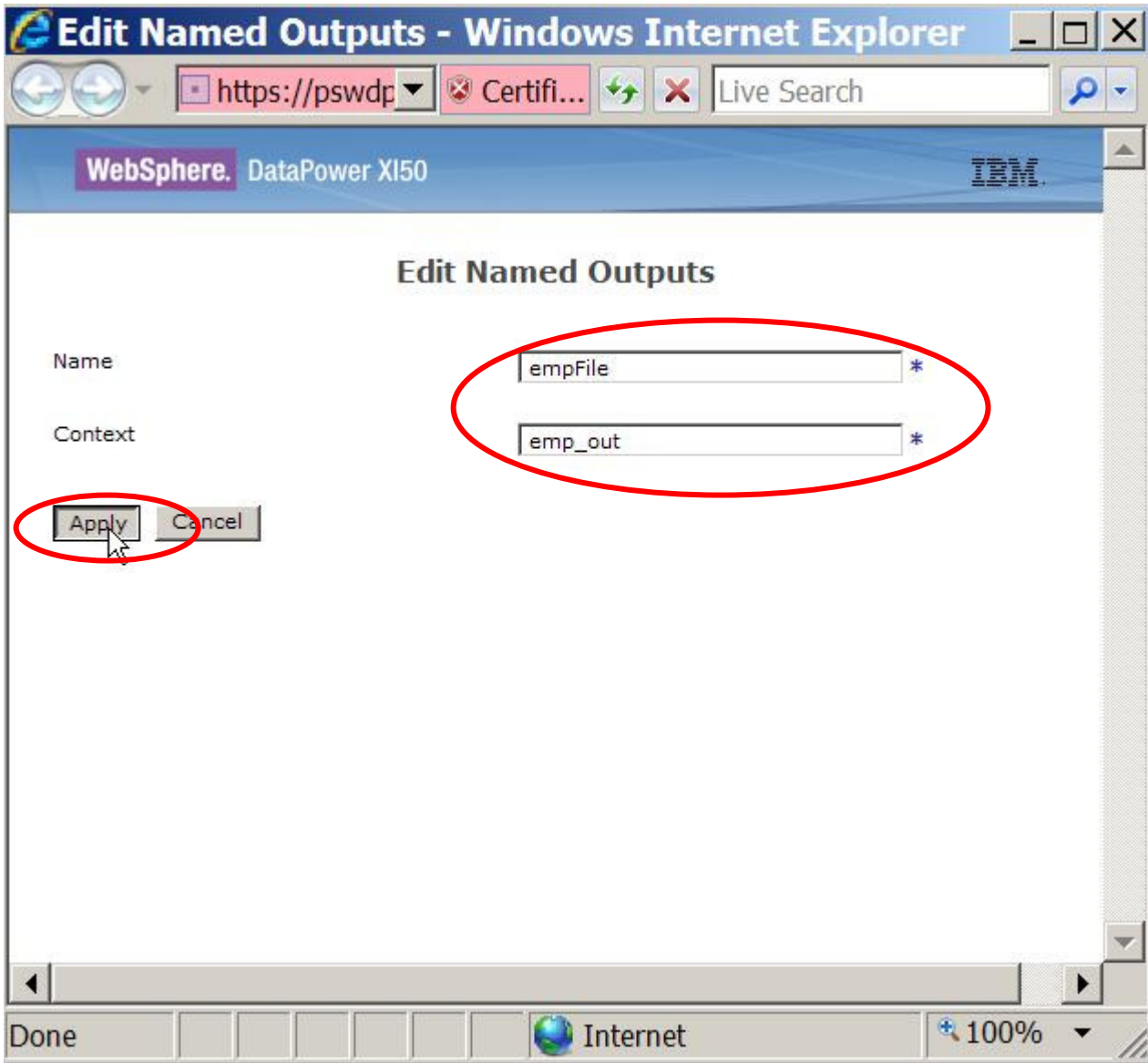
Name	Context
(empty)	

Add

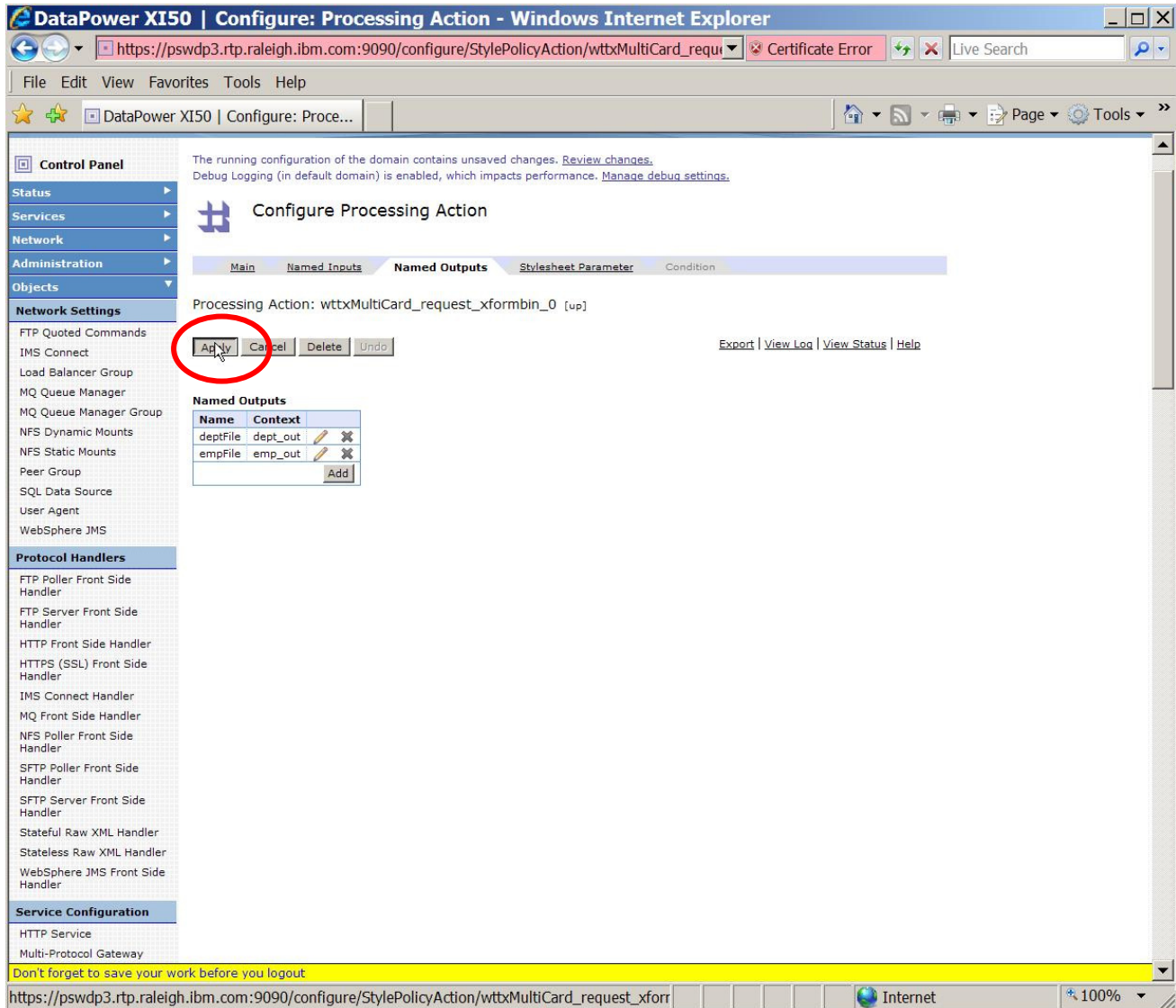
Don't forget to save your work before you logout

Internet 100%





- Apply the changes you've just completed, and then save your configuration.

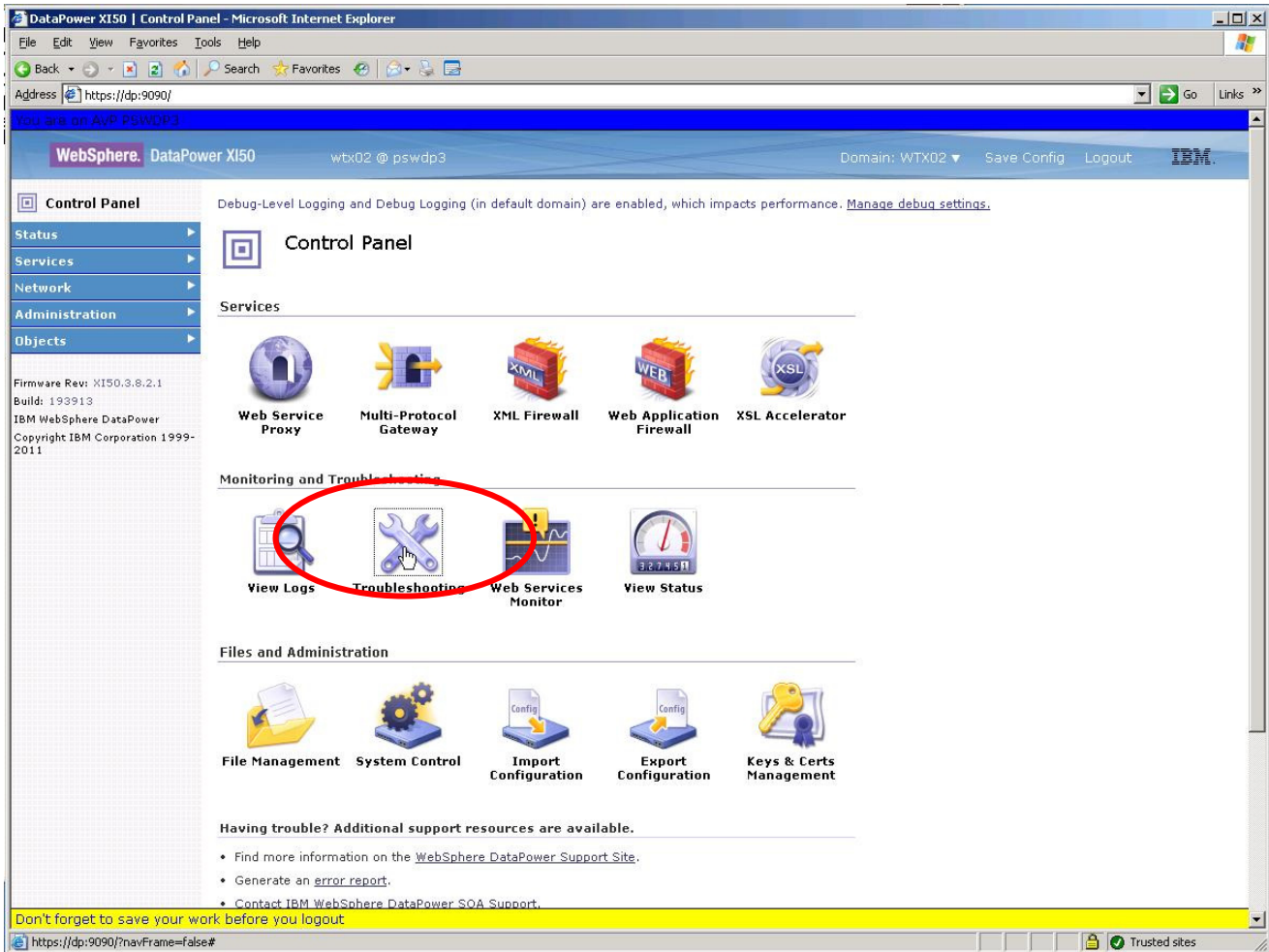


- You are now ready to test your service using cURL, a freely available tool that can (among many **many** other capabilities) submit http requests from a command-line. Open a Windows command window by clicking on the Command Window icon on the Desktop. You will now type in the cURL command to send an http request containing a MIME-format input message to your service.
 - A *template* containing the command you need to enter is contained in the file `c:\LabFiles\curl-cmd.txt`. You may want to use that template to copy the command for execution into your command window, but you will have to adjust the port number in the command to direct the request to your service port on the DataPower device:

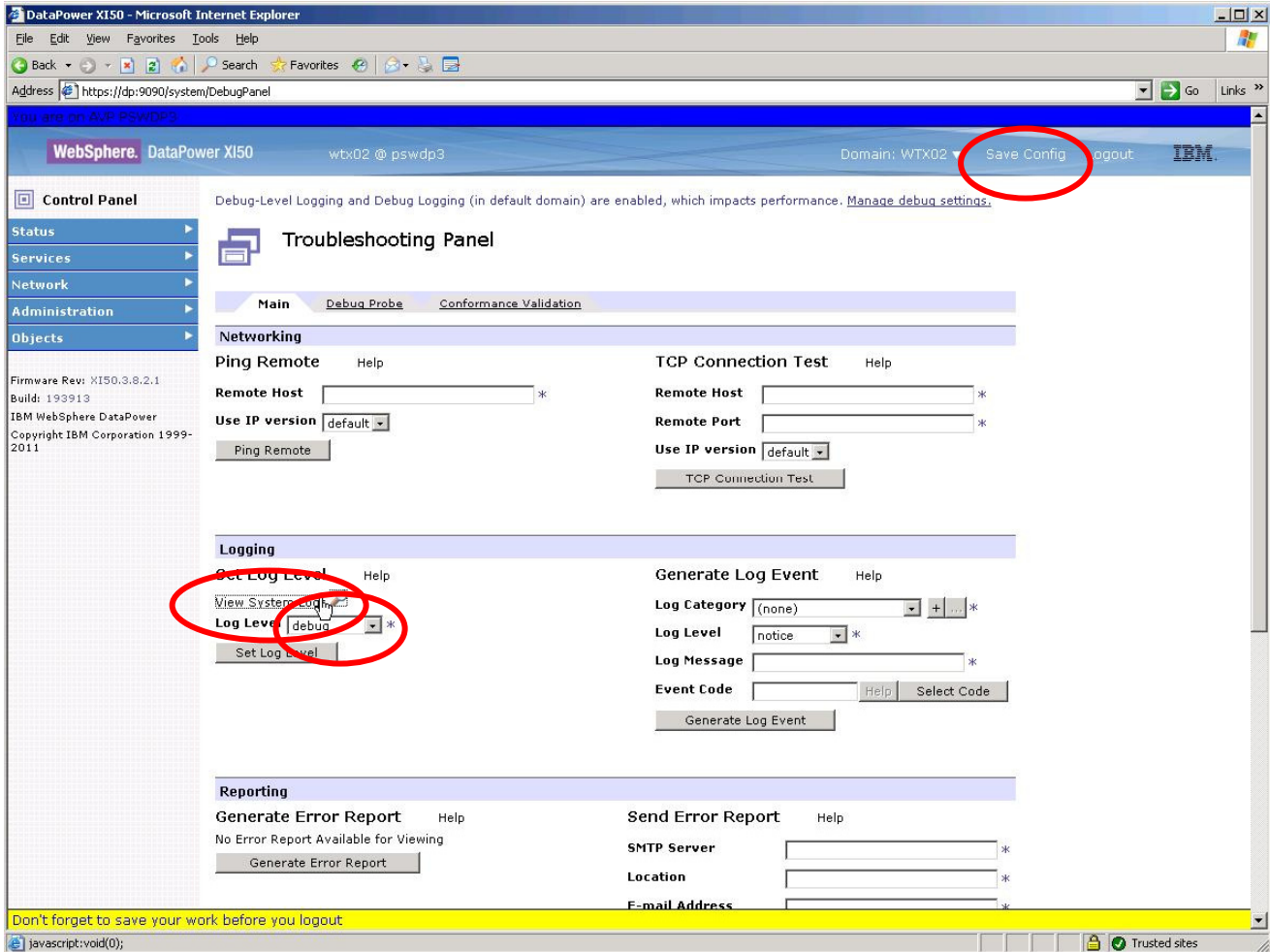


```
Cmd Prompt in curl directory
c:\curl>curl.exe --data-binary @c:\labfiles\multipleIO.msg -H "Content-Type: multipart/related; type=\"text/xml\"; boundary=\"mime_boundary\"" http://dp:8025
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"><env:Body><env:Fault><faultcode>env:Client</faultcode><faultstring>Internal Error (from client)</faultstring></env:Fault></env:Body></env:Envelope>
c:\curl>
```

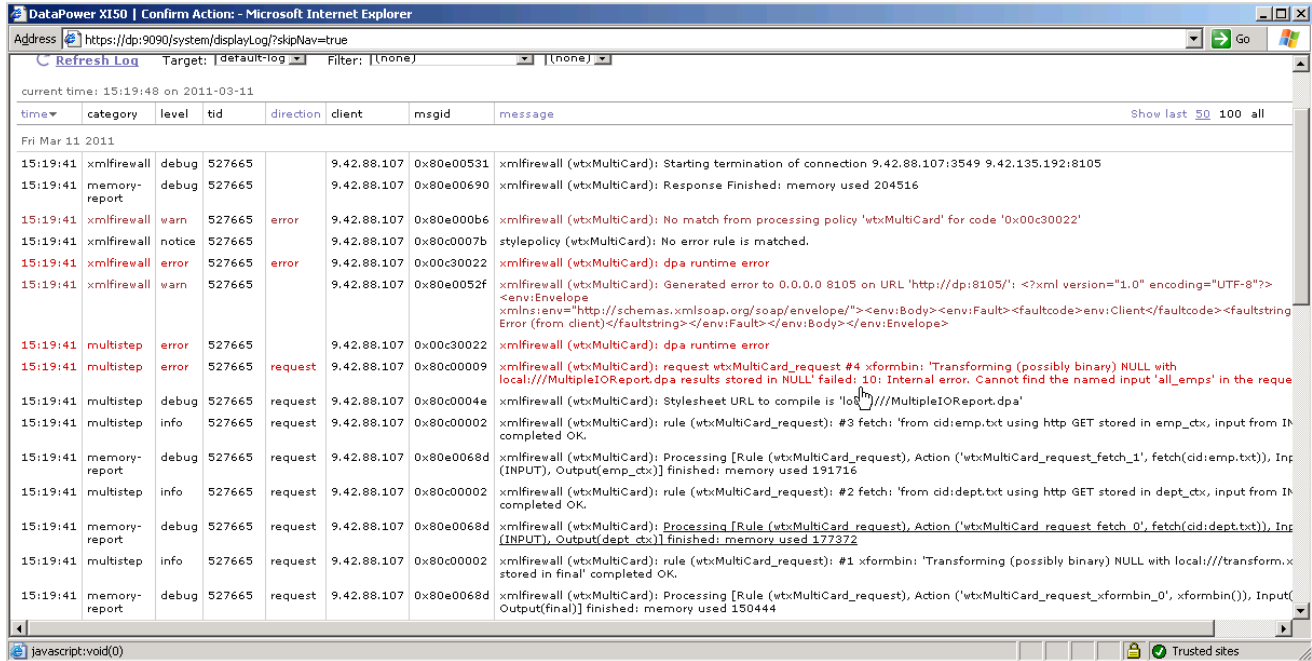
- *What happened?* DataPower (in the absence of specific customization to the service, and by design – you don't want to divulge information that would assist hackers in accessing the service) returns a very general error back to the client. To troubleshoot the problem, you will need to start by looking at the DataPower logs.
- Using Internet Explorer, navigate to the DataPower home page, then click on Troubleshooting Icon:



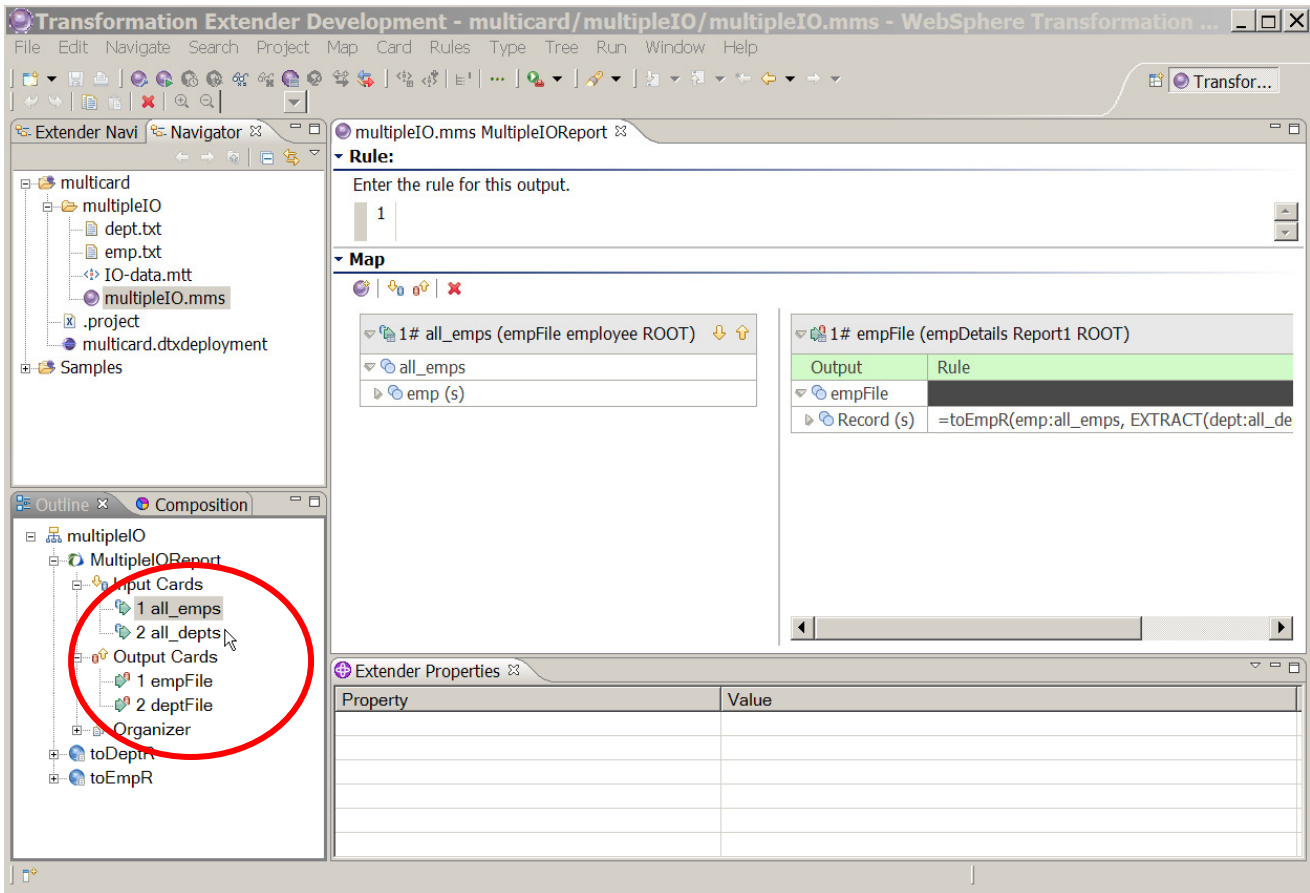
- Ensure that your logging level is set to *debug*. Save your configuration if you changed it. Now, click on the *View System Logs* link – this will bring up a separate window displaying the DataPower log:



- Send another request to your service from the command window, then return to the DataPower log window and use the *Refresh Log* link to update the logs. You can now read the log to figure out what went wrong:



- DataPower logs are read from bottom to top in the GUI view (in other words, the log records in this display are from newest to oldest, top to bottom). You can see that the DataPower runtime could not locate any context mapped to the input card *all_emps*.
- Return (navigate) to the DataPower configuration screen for the Binary Transform Action, and switch to the *Named Inputs* tab. The two input cards were [deliberately] misnamed for both “departments” and “employees” – we specified *AllDepts* and *AllEmps* as the name of the input cards, but the actual names are *all_depts* and *all_emps*.
 - This is very likely the single **most** common problem in configuring maps with multiple inputs or outputs. Although you wouldn’t do this deliberately under normal circumstances, this sort of misconfiguration is particularly likely if there are different individuals implementing the DataPower service from the ones developing the map in Design Studio. There is no easy way to examine the deployed file (the compiled map file ending in *dpa*) to display the named inputs and outputs. So the service implementer on DataPower may be relying on written (or even verbal) instructions from the map designer, and mistakes are easy to make! Note that the card names are case sensitive as well!
 - However, the card names are **easy** to see in Design Studio – the card names are displayed in the map detail pane. Switch back to the Design Studio window and take a look:



- Now, return to your Internet Explorer window and correct the DataPower configuration issue. Navigate to the Processing Action as you did previously (from the left-hand navigation menu -- *Objects/XML Processing/Processing Action*), select the correct action (by name), and then edit the two *Named*

Input entries (click on the *Pencil* icon to the right of each entry to edit):

Main **Named Inputs** Named Outputs Stylesheet Parameter Condition

Processing Action: wttxMultiCard_request_xformbin_0 [up]

Apply Cancel Delete Undo

Named Inputs

Name	Context		
AllDepts	dept_ctx		
AllEmps	emp_ctx		
<input type="button" value="Add"/>			

- After you have fixed the Named Inputs, the configuration should look like this:



Configure Processing Action

Main **Named Inputs** Named Outputs Stylesheet Parameter Condition

Processing Action: wtxMultiCard_request_xformbin_0 [up]

Apply Cancel Delete Undo

Named Inputs

Name	Context		
all_depts	dept_ctx		
all_emps	emp_ctx		
<input type="button" value="Add"/>			

- Apply your changes and save your configuration. Now, re-run your test from the command window by reissuing the cURL command as before. The response should look like this:

```
Cmd Prompt in curl directory
c:\curl>curl.exe --data-binary @c:\labfiles\multipleIO.msg -H "Content-Type: multipart/related; type=\"text/xml\"; boundary=\"mime_boundary\"" http://dp:8025

--mime_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: binary

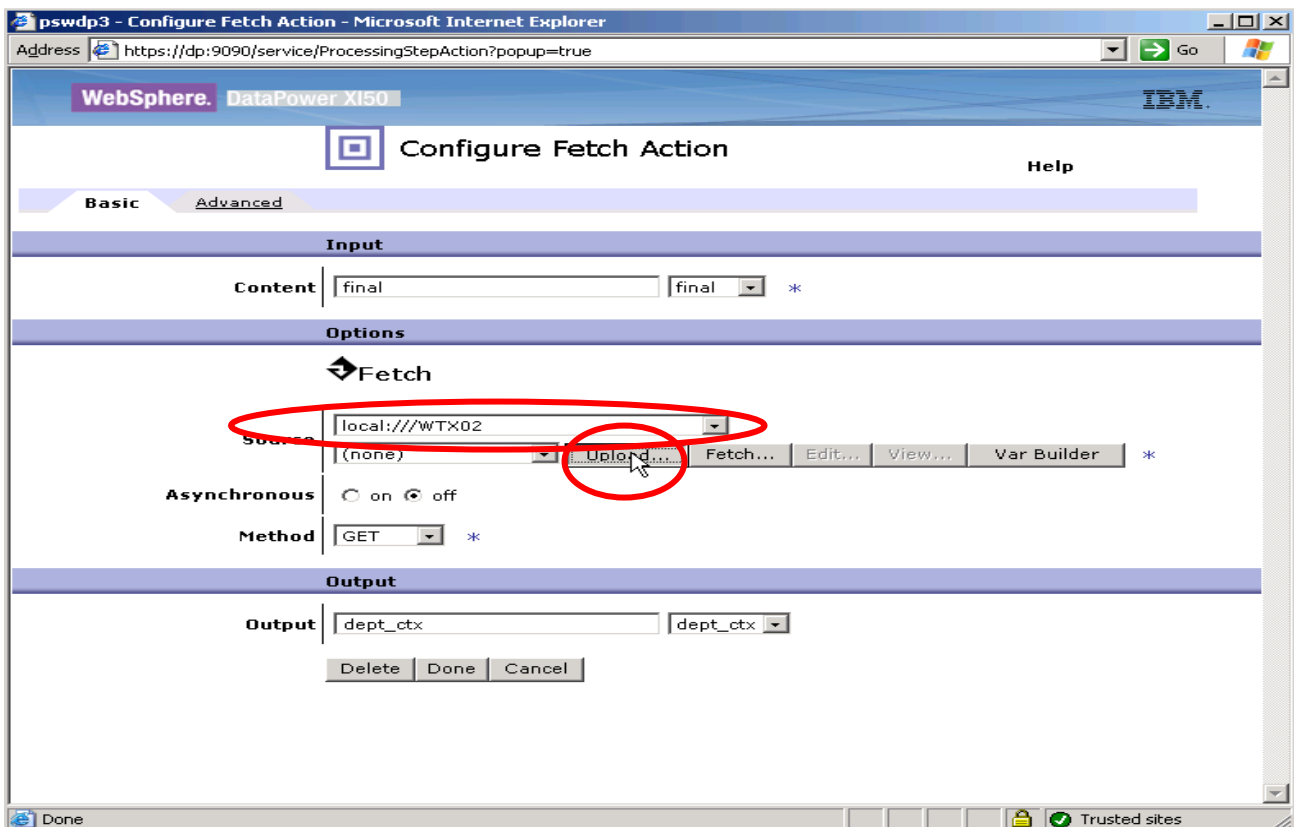
<?xml version="1.0" encoding="UTF-8"?>
<encoded>dGhpcyBpcyBtYWluIGRvY3VtZW50</encoded>
--mime_boundary
Content-ID: <emp_details.out>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream

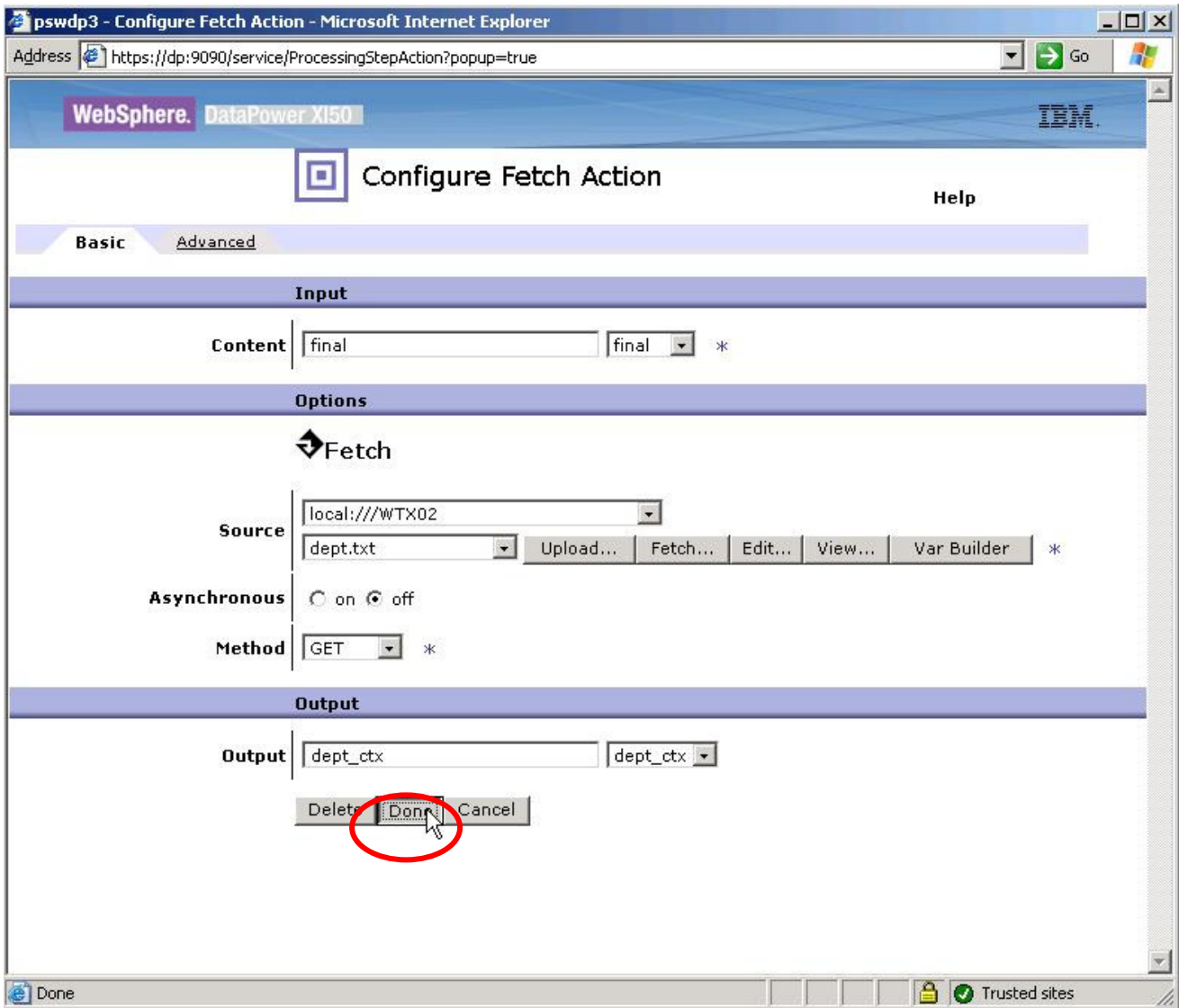
Sales,Benedykt,assistant
Admin,Kazimierz,Manager
Sales,Magdalena,SW_Engineer
Sales,Walburga,HW_Engineer
Sales,Dawid,Technician
Admin,Otto,Sales
Admin,Jozefat,Sales
Sales,Teresa,assistant
Admin,Jan,Manager
Admin,Zygmunt,SW_Engineer
RD,Leonard,HW_Engineer
Admin,Justina,Technician
Sales,Wincenty,Sales
RD,Lukasz,Sales
HR,Romuald,assistant
Admin,Alfons,Manager
RD,Lucjan,SW_Engineer
Admin,Wanda,HW_Engineer
RD,Monika,Technician
HR,Waclawa,Sales
RD,Hiacynt,Sales
Sales,Maciej,assistant
Admin,Maksymilian,Manager
Admin,Ignacy,SW_Engineer
Admin,Ludmila,HW_Engineer
Admin,Pelagia,Technician
Admin,Saturnin,Sales
RD,Leon,Sales
Admin,Jacinta,assistant
RD,Klementyna,Manager
RD,Wawrzyniec,SW_Engineer
Sales,Bernard,HW_Engineer
RD,Bozena,Technician
Admin,Adalbert,Sales
RD,Michalina,Sales
Admin,Prakseda,assistant
```

- You have successfully configured and tested the service!

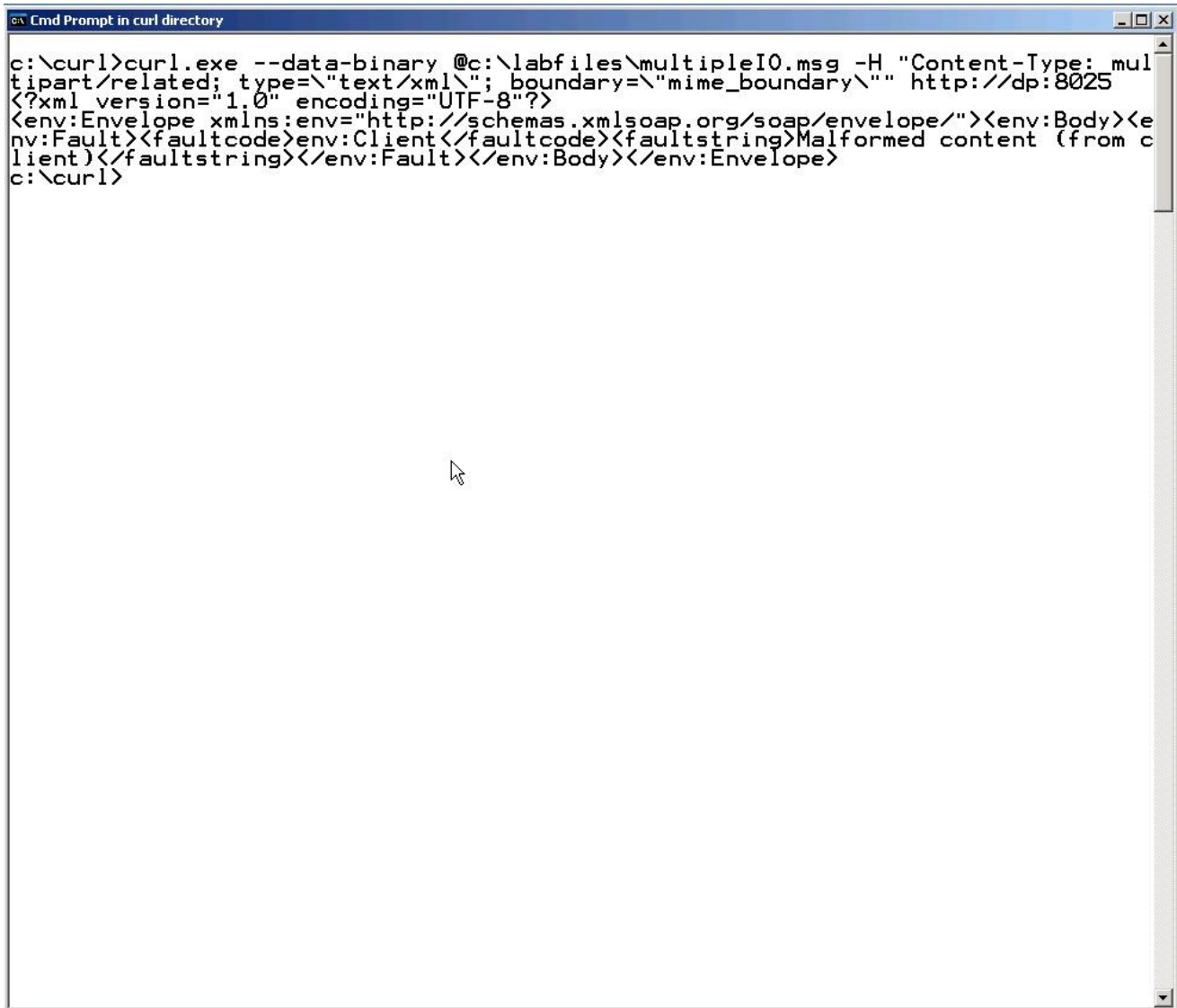
Part 7: Converting One Input Source to a Local Document

- Rather than construct and supply an input message containing both the list of employees and the department information on every request, we can convert the service to only require a list of employees. To do this, you will make a simple change to the service configuration – we’ll change the *Fetch Action* which retrieves the “department file” so that we get that information from a local file on the DataPower device.
- Return to the processing policy by navigating through the XML Firewall wizard on the main (login) screen.
 - [click XML Firewall, select the *wtxMultiCard* firewall, click the edit button (“...”) to the right of the selected processing policy.]
- Edit the first *Fetch Action* (which currently retrieves the *dept.txt* attachment from the incoming message).
 - Remember – you can *hover* the mouse over the actions on a processing policy to have a popup appear containing the details of the action. This will help you identify the correct action to edit.
- Change the Source to *local:///WTXxx*, and then upload the file *c:\LabFiles\dept.txt*.





- Apply the changes and save your configuration.
- Retest your service, this time using the input file *c:\LabFiles\multipleIO-JustEmps.msg*:

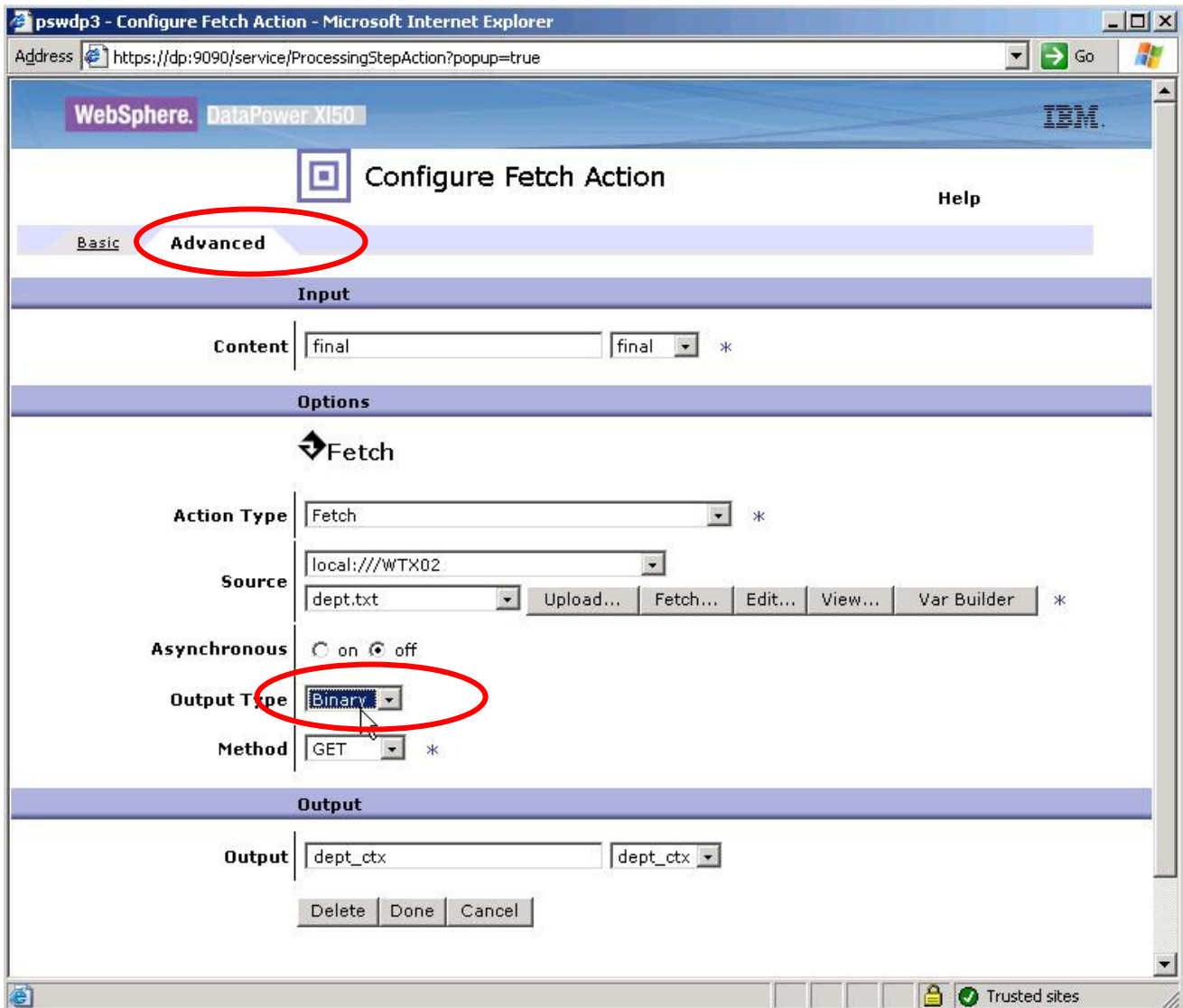


```
c:\curl>curl.exe --data-binary @c:\labfiles\multipleIO.msg -H "Content-Type: multipart/related; type=\"text/xml\"; boundary=\"mime_boundary\"" http://dp:8025
<?xml version="1.0" encoding="UTF-8"?>
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"><env:Body><env:Fault><faultcode>env:Client</faultcode><faultstring>Malformed content (from client)</faultstring></env:Fault></env:Body></env:Envelope>
c:\curl>
```

- The request failed, so you will need to troubleshoot again. Look through the DataPower logs (you may need to send in another request once you have opened the Log window; refresh the log!). There should be a log entry something like this:

time	category	level	tid	direction	client	msgid	message
Tue Mar 01 2011							
20:53:56	xmldatafirewall	debug	633344		9.42.88.107	0x80e00531	xmldatafirewall (wtbxMultiCard): Starting termination of connection 9.42.88.107:2450 9.42.135.192:8025
20:53:56	memory-report	debug	633344		9.42.88.107	0x80e00690	xmldatafirewall (wtbxMultiCard): Response Finished: memory used 157828
20:53:56	xmldatafirewall	warn	633344	error	9.42.88.107	0x80e000b6	xmldatafirewall (wtbxMultiCard): No match from processing policy 'wtbxMultiCard' for code '0x00030001'
20:53:56	xmldatafirewall	error	633344	error	9.42.88.107	0x00030001	xmldatafirewall (wtbxMultiCard): Parse error
20:53:56	xmldatafirewall	warn	633344		9.42.88.107	0x80e0052f	xmldatafirewall (wtbxMultiCard): Generated error to 0.0.0.0 8025 on URL 'http://dp:8025/'; <?xml version="1.0" encoding="UTF-8"?><env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"><env:Body><env:Fault><faultcode>env:Client</faultcode><faultstring>Malformed content (from client)</faultstring></env:Fault></env:Body></env:Envelope>
20:53:56	multistep	error	633344		9.42.88.107	0x00030001	xmldatafirewall (wtbxMultiCard): Parse error
20:53:56	multistep	error	633344	request	9.42.88.107	0x80c00009	xmldatafirewall (wtbxMultiCard): request wtbxMultiCard_request #2 fetch: 'from local:///WTX02/dept.txt using http GET stored in dept_txt_input from final' failed: illegal character '1' at offset 0 of local:///WTX02/dept.txt
20:53:56	xmldataparse	error	633344	request	9.42.88.107	0x80e003aa	xmldatafirewall (wtbxMultiCard): illegal character '1' at offset 0 of local:///WTX02/dept.txt
20:53:56	xmldataparse	debug	633344	request	9.42.88.107	0x80e003a6	xmldatafirewall (wtbxMultiCard): Parsing document: 'local:///WTX02/dept.txt'
20:53:56	multistep	info	633344	request	9.42.88.107	0x80c00002	xmldatafirewall (wtbxMultiCard): rule (wtbxMultiCard_request): #1 xform: 'Transforming INPUT with local:///transform.xml results stored in final completed OK.
20:53:56	memory-report	debug	633344	request	9.42.88.107	0x80e0068d	xmldatafirewall (wtbxMultiCard): Processing [Rule (wtbxMultiCard_request), Action ('wtbxMultiCard_request_xform_0', xform(local:///transform.xml, Input[INPUT], Output[final])] finished: memory used 144252
20:53:56	multistep	debug	633344	request	9.42.88.107	0x80c0004e	xmldatafirewall (wtbxMultiCard): Stylesheet URL to compile is 'local:///transform.xml'
20:53:56	xmldataparse	debug	633344	request	9.42.88.107	0x80e003ab	xmldatafirewall (wtbxMultiCard): Finished parsing: http://dp:8025/
20:53:56	xmldataparse	debug	633344	request	9.42.88.107	0x80e003a6	xmldatafirewall (wtbxMultiCard): Parsing document: 'http://dp:8025/'
20:53:56	multistep	debug	633344	request	9.42.88.107	0x80c0005c	xmldatafirewall (wtbxMultiCard): Stylesheet URL to compile is 'local:///transform.xml'
20:53:56	memory-report	debug	633344		9.42.88.107	0x80e0068c	xmldatafirewall (wtbxMultiCard): Request Started: memory used 59204

- You can see the problem here – when we changed the Fetch action to get from the local file, the format of the file wasn't what DataPower expects.
 - By default, DataPower expects XML documents.
- Let's correct that; edit the *Fetch Action*, switch to the *Advanced* tab, and change the *Output Type* to *binary*:



- Apply your changes and save. Retest:


```
Select Cmd Prompt in curl directory

c:\curl>curl.exe --data-binary @c:\labfiles\multiple10-JustEmps.msg -H "Content-
Type: multipart/related; type=\"text/xml\"; boundary=\"mime_boundary\"" http://d
p:8025

--mime_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: binary

<?xml version="1.0" encoding="UTF-8"?>
<encoded>dGhpcyBpcyBtYWluIGRvY3VtZW50yBub3cgd2l0aCBqdXN0IG9uZSBhdHRhY2htZW50ICh
lbXBsb3llZXMp</encoded>
--mime_boundary
Content-ID: <emp_details.out>
Content-Transfer-Encoding: binary
Content-Type: application/octet-stream

Sales,Benedykt,assistant
Admin,Kazimierz,Manager
Sales,Magdalena,SW Engineer
Sales,Walburga,HW Engineer
Sales,Dawid,Technician
Admin,Otto,Sales
Admin,Jozefat,Sales
Sales,Teresa,assistant
Admin,Jan,Manager
Admin,Zygmunt,SW Engineer
RD,Leonard,HW Engineer
Admin,Justina,Technician
Sales,Wincenty,Sales
RD,Lukasz,Sales
HR,Romuald,assistant
Admin,Alfons,Manager
RD,Lucjan,SW Engineer
Admin,Wanda,HW Engineer
RD,Monika,Technician
HR,Waclawa,Sales
RD,Hiacynt,Sales
Sales,Maciej,assistant
Admin,Maksymilian,Manager
Admin,Ignacy,SW Engineer
Admin,Ludmila,HW Engineer
Admin,Pelagia,Technician
Admin,Saturnin,Sales
RD,Leon,Sales
Admin,Jacinta,assistant
RD,Klementyna,Manager
RD,Wawrzyniec,SW Engineer
Sales,Bernard,HW Engineer
RD,Bozena,Technician
```

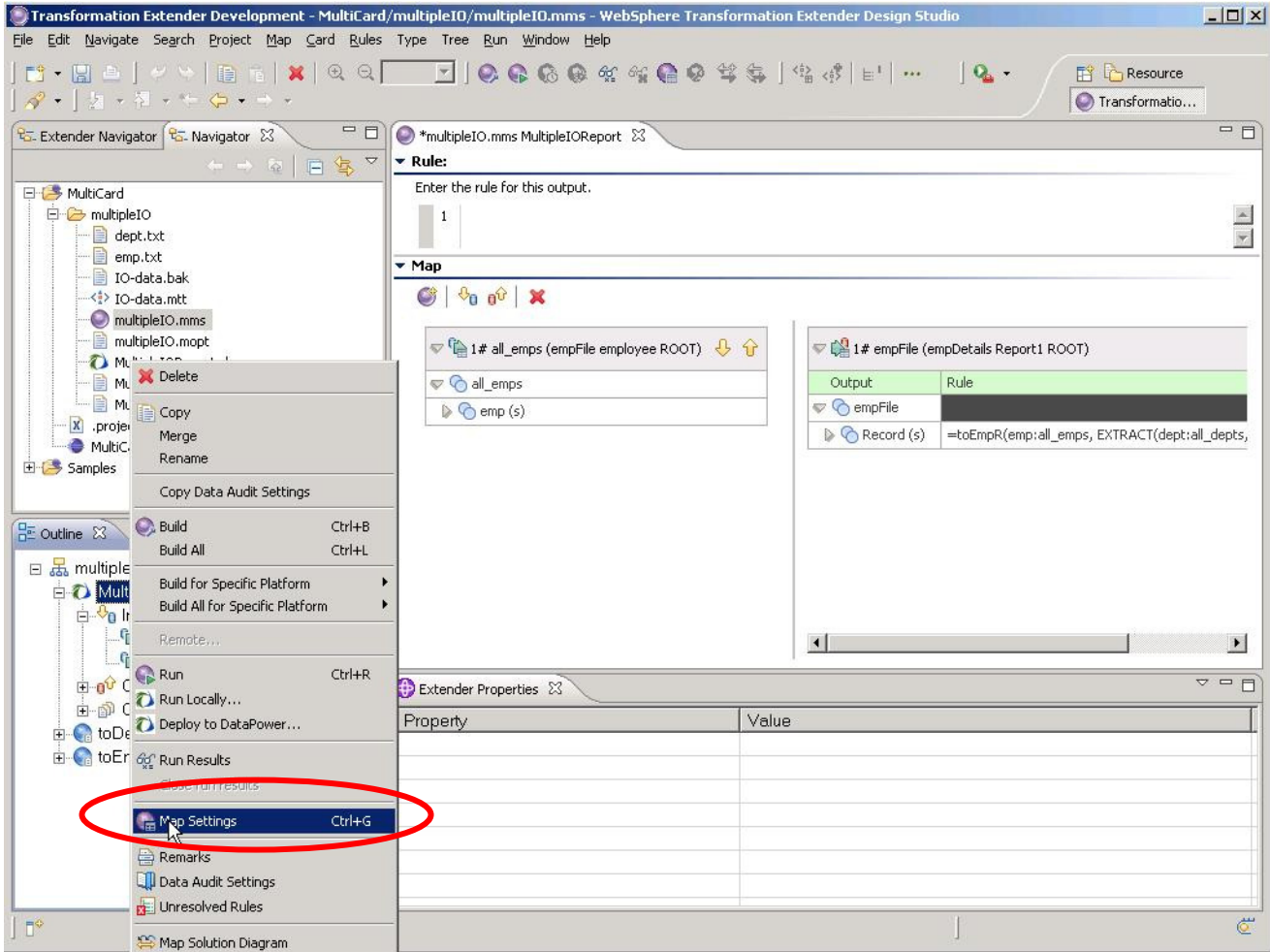
- Now the service works.

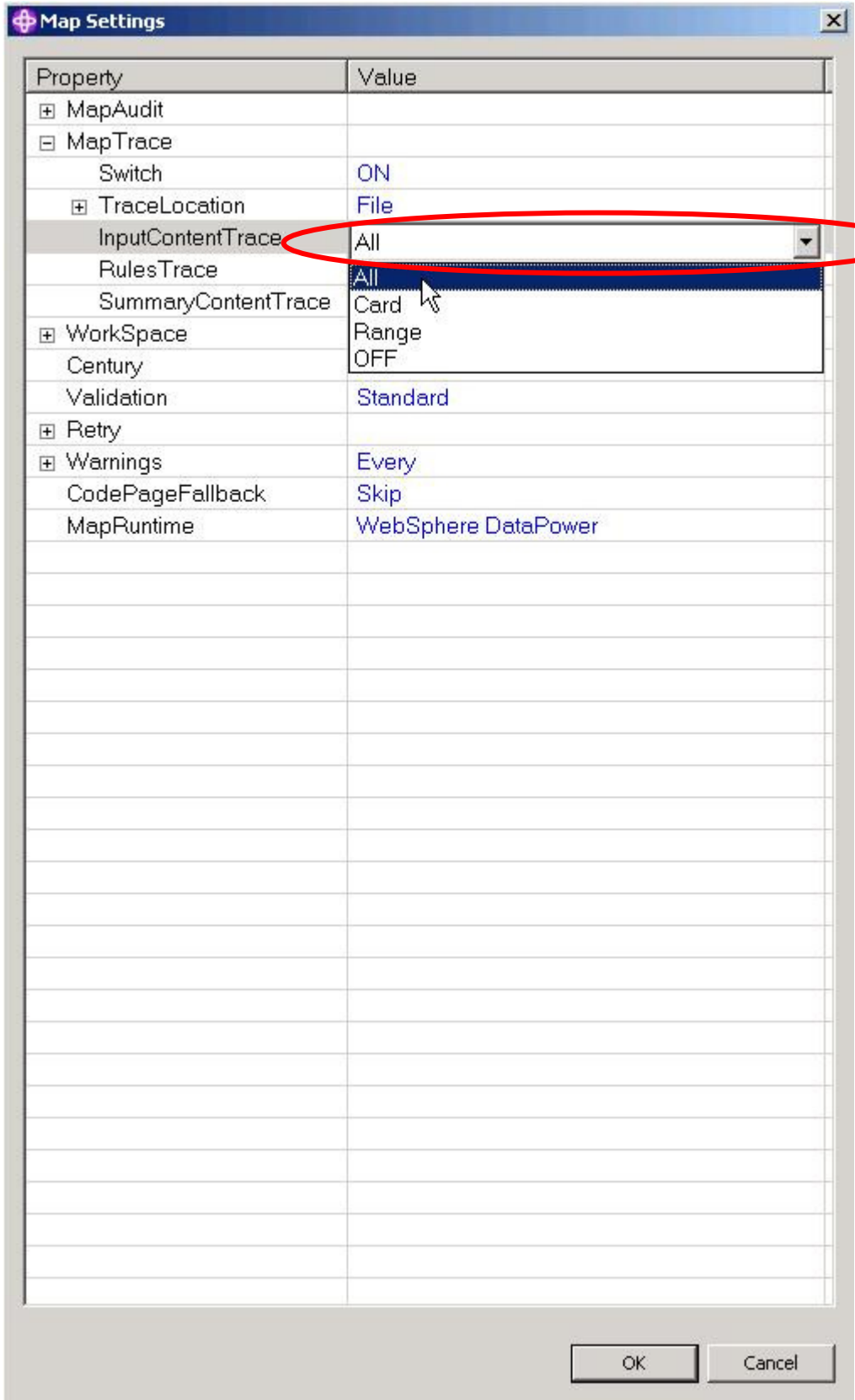
Part 8: Troubleshooting Invalid Input

- Lastly, we'll work through another common problem. Use cURL to send the request `c:\LabFiles\multipleIO-error1.msg` to your service. What happened? Once again, let's start with the DataPower logs:

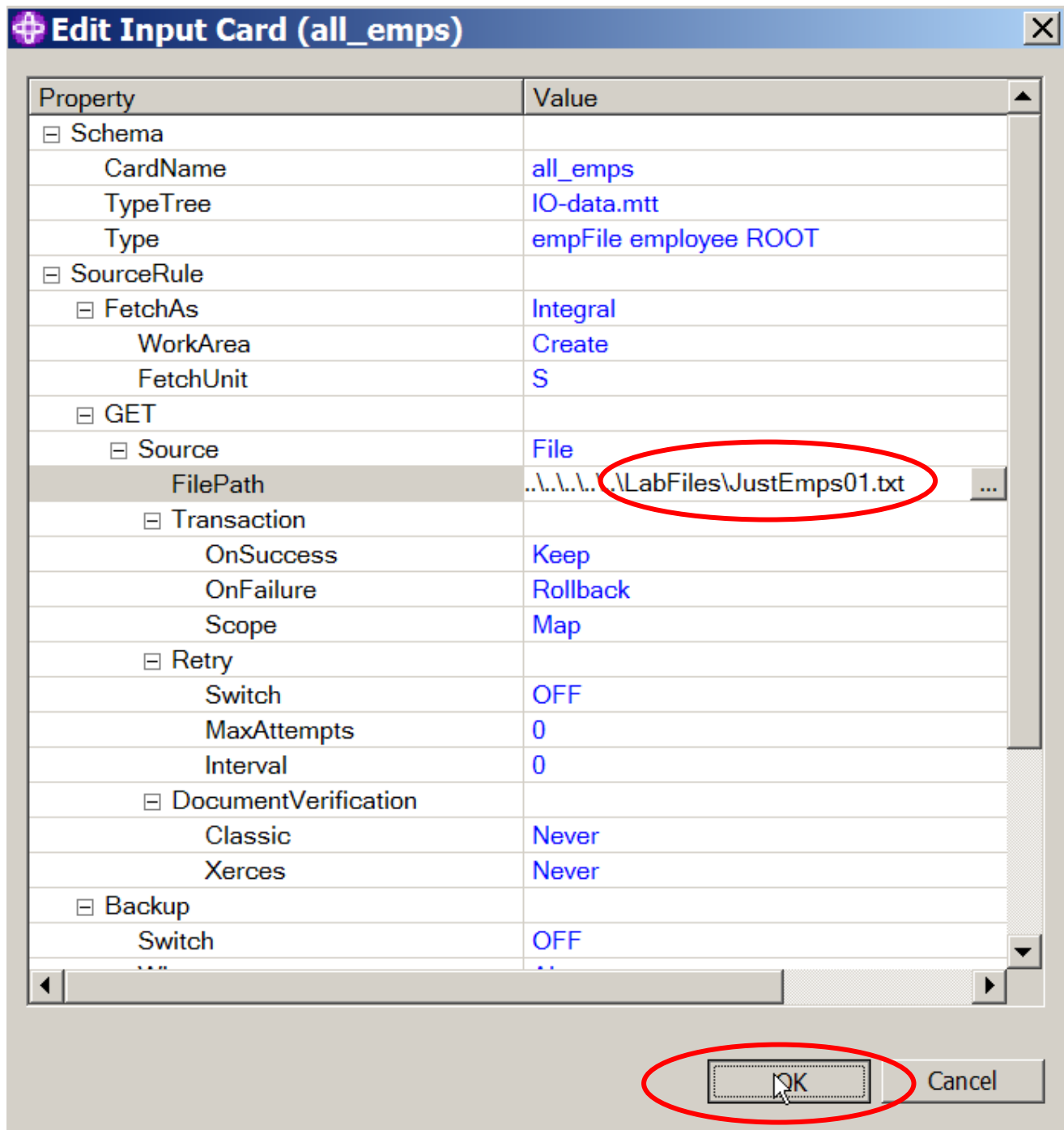
The screenshot shows the DataPower XI50 System Log for XML Firewall Service. The log is displayed in a table with columns for time, category, level, tid, direction, client, msgid, and message. The log shows a sequence of events starting with a connection termination, followed by several warnings and errors. A key error entry at 21:14:29 shows a 'dps runtime error' with a message indicating a generated error to 0.0.0.0:8025 on URL 'http://dp:8025/'. The error message includes an XML snippet: `<?xml version="1.0" encoding="UTF-8"?><env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/"><env:Body><env:Fault><faultcode>env:Client</faultcode><faultstring>Internal Error (from client)</faultstring></env:Fault></env:Body></env:Envelope>`. This is followed by a 'dps runtime error' at 21:14:29 with the message: 'request wtbxMultiCard_request #6 xformbin: 'Transforming (possibly binary) NULL with local:///WTX02/MultipleIOReport.dpa results stored in NULL failed: 21: Input valid but unknown data found''. The log continues with several 'request' entries and 'completed OK' messages, ending with a 'Request Started' message at 21:14:29.

- We can see that the DataPower runtime doesn't like something about our input, but it is difficult to discern exactly what the matter is – we get a fairly generic “21: Input valid but unknown data found”. To work through this issue, you will Design Studio and turn on tracing.
 - NOTE: There is no tracing available when maps are executed on the DataPower device.
- Return to your Design Studio session and edit the Map Settings (right-click on the compiled map, select *Map Settings*). Switch Trace to *ON* and set *InputContentTrace* to *All*:

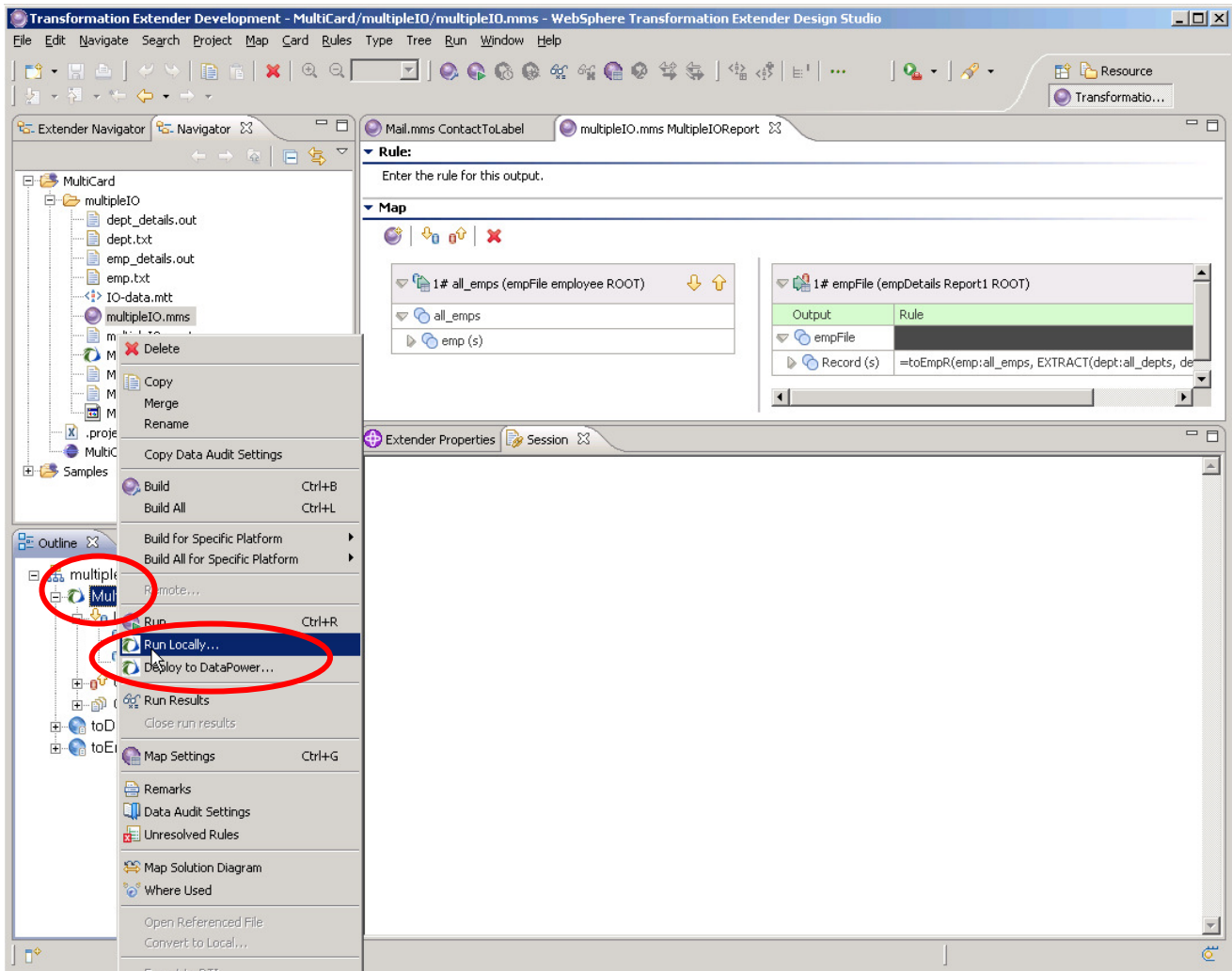




- In the outline view, select the *all_emps* input card, right-click and Edit the card. Change the input file to *c:\LabFiles\JustEmps01.txt*. [This is a stand-alone copy of the section *emps.txt* from the MIME-format file *c:\LabFiles\multipleIO-error1.msg* used as input to your service.]



Save and then rebuild the map, then right-click the map and choose *Run Locally*:



- Now, open the Trace View window (*Window/Show View/Trace*). Throughout the trace, you can see what's happened – somehow (deliberately in this case) the input terminators for each line have been changed from <CR><LF> to <CR>.
- Tracing the problem map by running locally within Design Studio is often an effective means for tracking down issues with the input data.

This concludes the WTX / DataPower Lab