Kodak

Smart Review System 1.0

Getting Started Guide

Document Number
719-00925A-EN-RevA

19 December 2007

Copyright 2007 Kodak. All rights reserved.

Kodak

Smart Review System 1.0
Getting Started Guide

CONTENTS

1. PartnerAPI.wsdl and PartnerAPICallback.wsdl 1

2. Populating Document Roots 1

3. Using the PartnerTestDriver Utility 1

4. Viewing an Image 6

5. Comparing Images 15

6. Retrieving an Annotation 18

7. Implementing Approval 20

8. Integrating With the Client Software 22
   - 8.1 Integrating With the JNLP Smart Review Client 22
   - 8.2 Integrating With the Installed Smart Review Client 24

9. Using the Callback Sample Code 25

10. Pre-Caching Images 27

1. PARTNERAPI.WSDL AND PARTNERAPICALLBACK.WSDL

Web Services Description Language (WSDL) is an XML-based language that provides a model for describing the functionality and location of a web service. Kodak® Smart Review System, the next generation product replacing Kodak's legacy RealTimeProof™ Partner, provides two WSDL files to describe its programming interface—**PartnerAPI.wsdl** and **PartnerAPICallback.wsdl**—which can be found at the root level of the SRS DVD. The main interface to SRS, the **PartnerAPI.wsdl**, has been exposed in the **PartnerTestDriver** utility and **PartnerAPI.asmx** Web page, which will be covered in later sections in this guide. These two tools provide you the convenience of familiarizing yourself with Smart Review System without having to write a line of code.

IMPORTANT: Former RTP Partner WSDL consumers should note that a few alterations have been made. The **PartnerAPI.wsdl** and **PartnerAPICallback.wsdl** of SRS differ in that:

- The WSDLs have changed from an RPC Encoded form to a Document Literal form, making SOAP messages less complex and integration simpler.
- **NoteID** has changed from an `int` type to a `string` type.
- GetMetaData and SetMetaData APIs are no longer compatible and use entirely new data structures.

An existing integration must be revisited to address these differences.

2. POPULATING DOCUMENT ROOTS

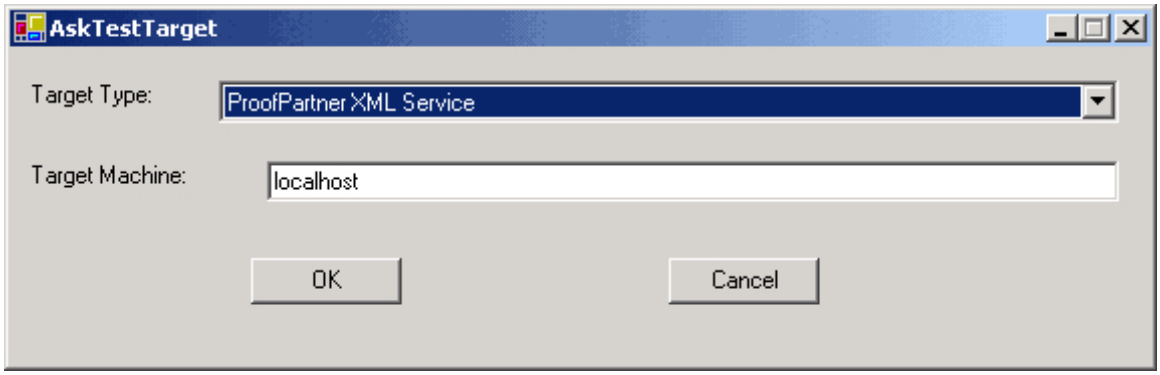
To use images in SRS, you must first place them in a Document Root (DocRoot) directory. In preparation for later sections in this guide, copy the SRS sample images from **C:\Kodak\TWAM\Samples\Images** to the **Default DocRoot** directory. By default, this is on the **SRS System Data** drive, in the folder **\ProofPartnerSourceDocRoot**. If in doubt, the location can be verified (and modified) using Smart Review System Administration. For more information, see the *Smart Review System 1.0 Installation Guide*.

3. USING THE PARTNERTESTDRIVER UTILITY

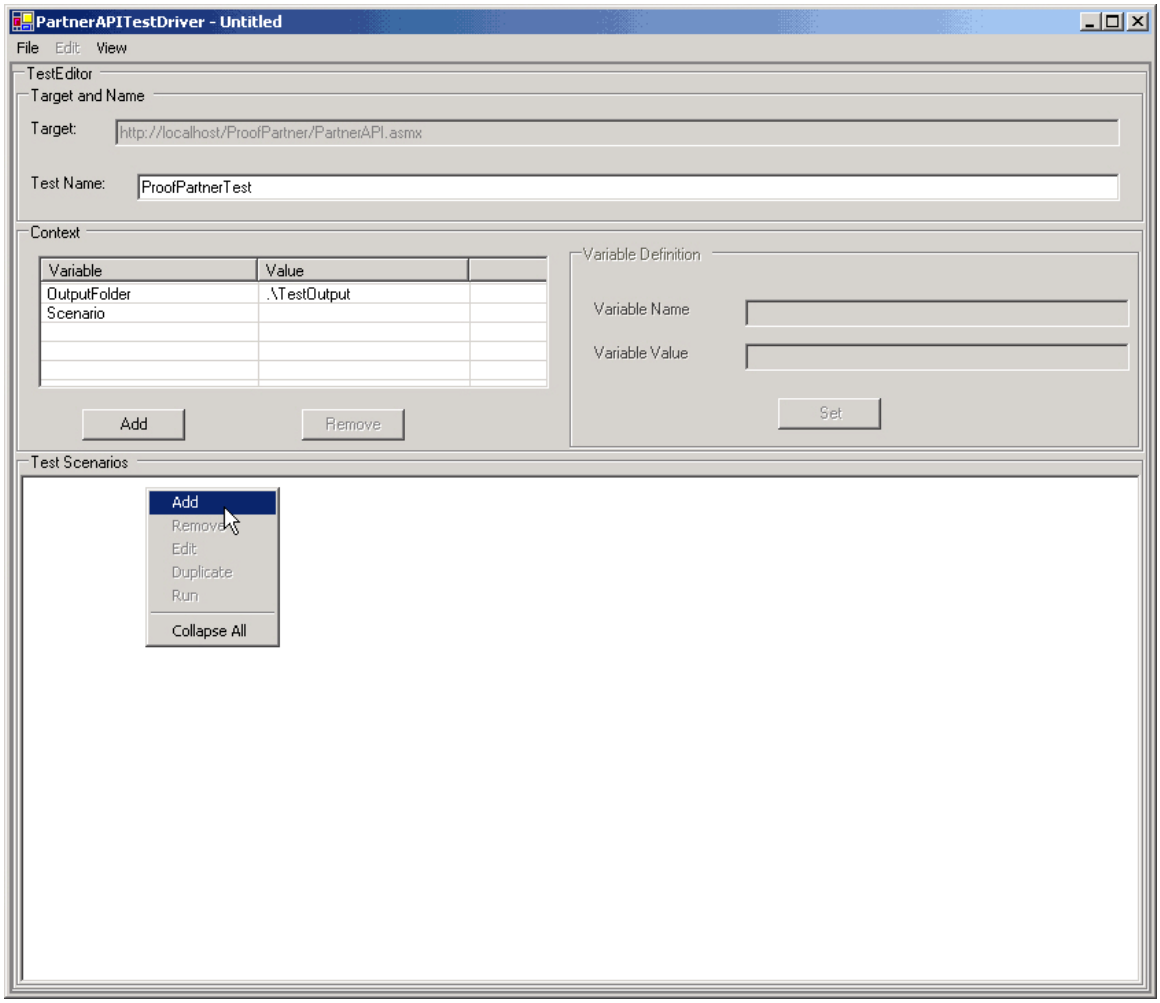
The PartnerTestDriver application is a scriptable test utility for SRS Server. This application is supported on the Microsoft® Windows® platform only.

1. After installing SRS, the PartnerTestDriver utility **PartnerTestDriver.exe** is available in **C:\Kodak\TWAM\Samples\TestDriver**. (The drive letter for the location of the test driver utility depends on where SRS was installed.)
2. Double-click **PartnerTestDriver.exe** to start the utility.
3. From the **File** menu, select **New Test**.
4. In the AskTestTarget dialog box, in the **Target Type** list, select **ProofPartner XML Service.**

5. In the **Target Machine** box, type the server name or URL. Click **OK**.

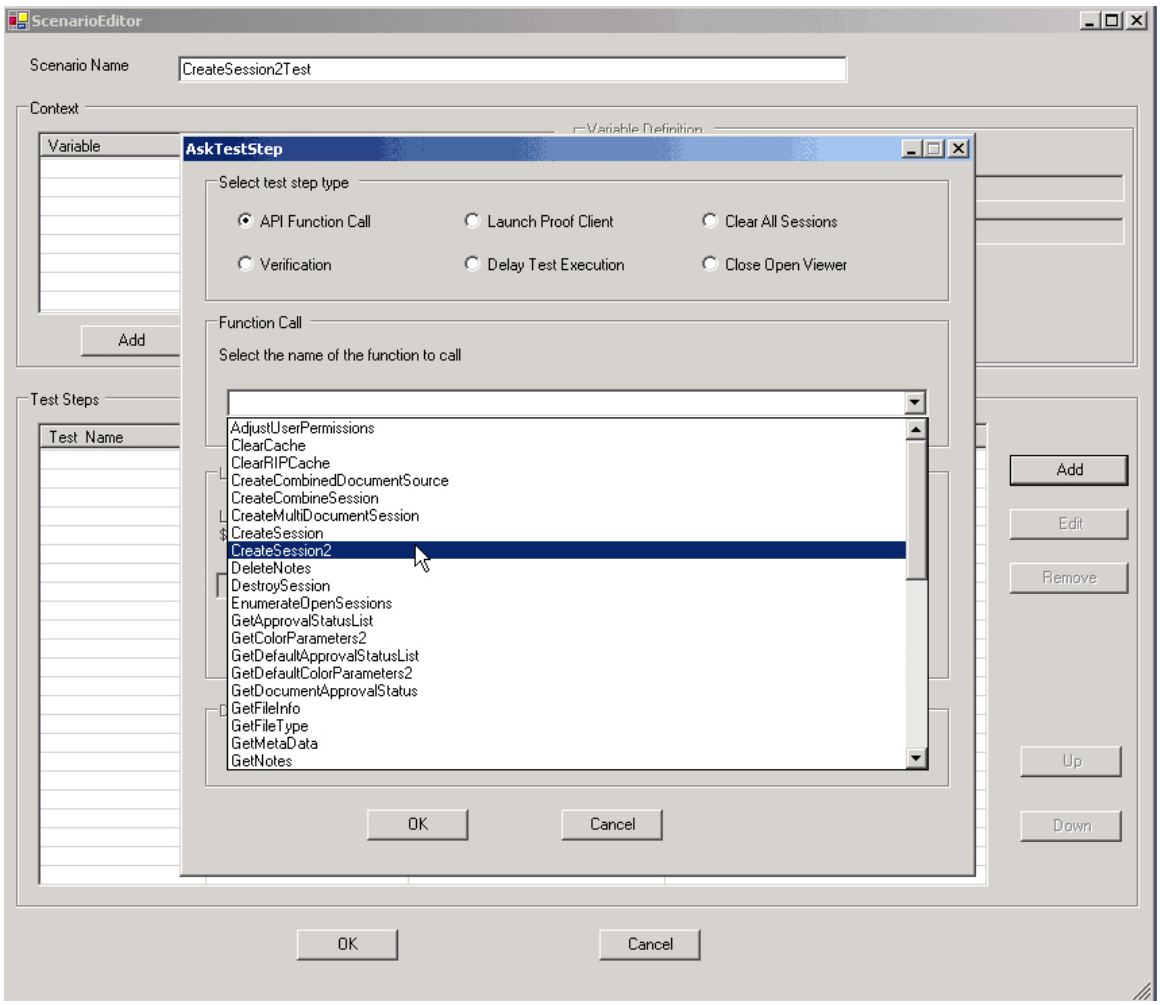


6. Right-click in the **Test Scenarios** box, and select **Add**.

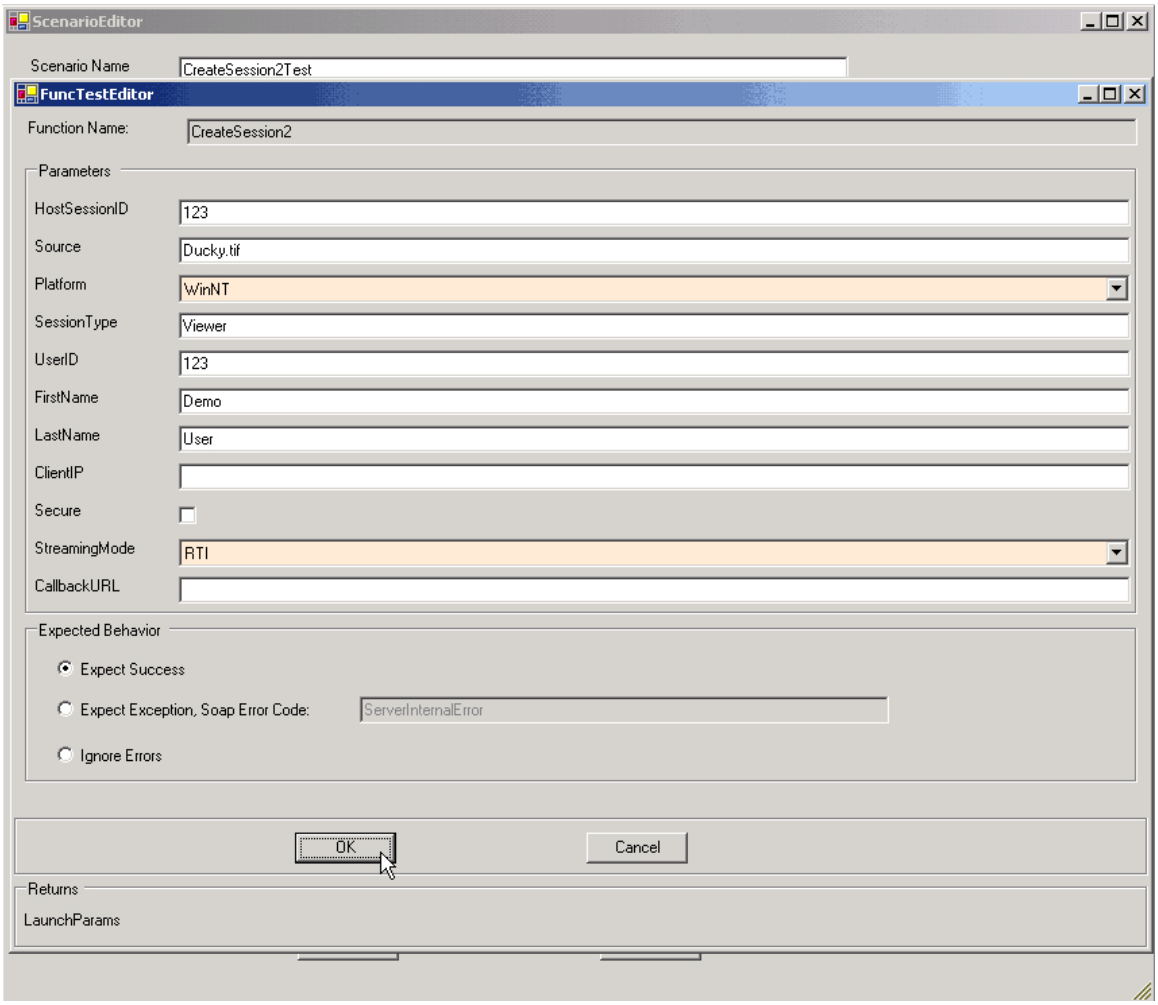


7. In the AskTestStep dialog box, type a **Scenario Name**, and click **Add**.
8. In the **Select test step type** area, select the desired test step type.

9. In the **Function Call** area, select the desired function. Click **OK.**

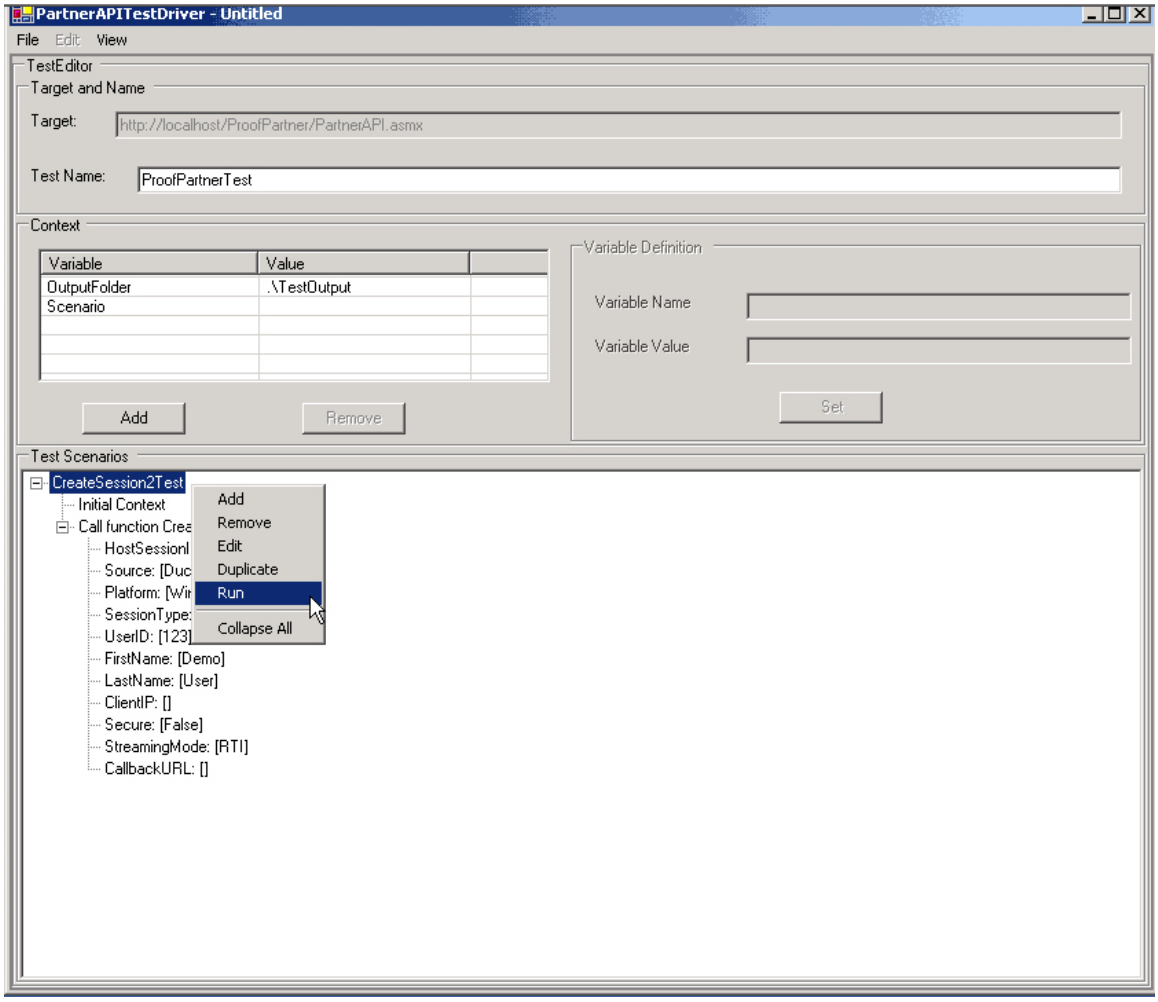


10. Fill in the **Parameters** area with test values, and click **OK**.

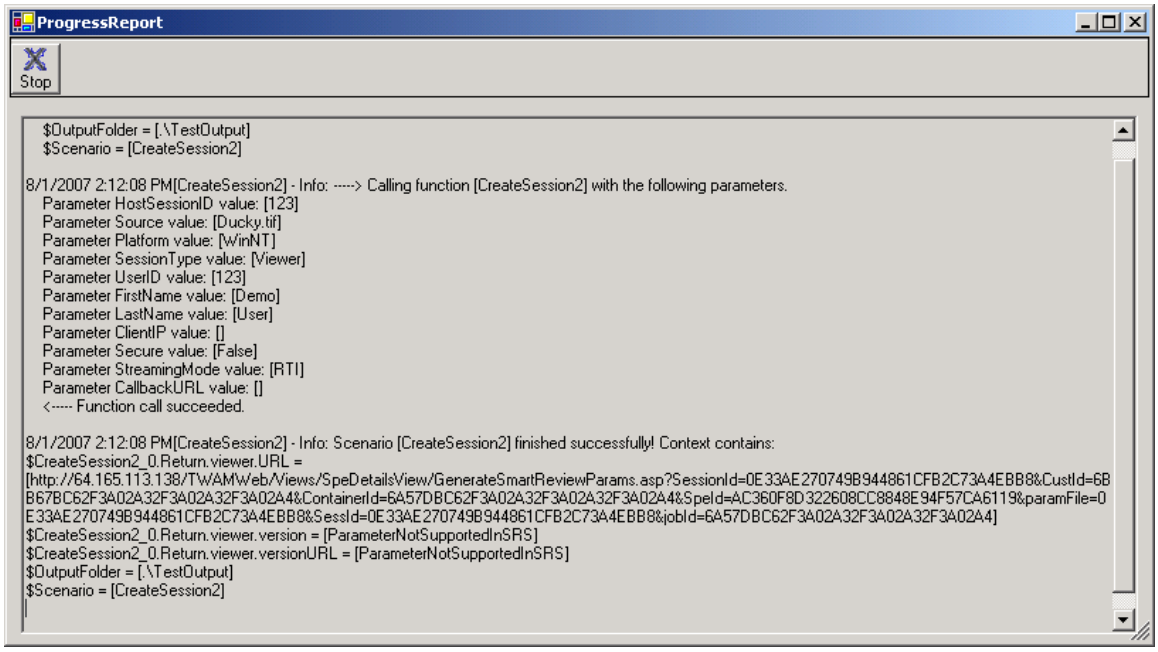


11. Click **OK** again to close ScenarioEditor.

12. In the **Test Scenarios** box, right-click the scenario that you just created, and select **Run**.



13. The ProgressReport window displays the results of the test scenario.



Note: In the PartnerTestDriver utility, if a function returns information, the variable from the utility that contains the returned information is as follows:

```
$<functionname>_<index>
```

In the preceding scenario, the variable is:

```
$CreateSession2_0.Return.viewer.URL
```

If you call a function more than once in a test scenario, to access the return information from subsequent calls, you must reference these returns with incremented indexes. Indexes start at 0, so, for example, if you make three calls to CreateSession2, the results for each call are stored in the following variables:

```
$CreateSession2_0

$CreateSession2_1

$CreateSession2_2
```

14. Minimize or close the ProgressReport window. In preparation for a later section, in the main PartnerAPITestDriver window, select **File > Save As** and browse to **C:\Kodak\TWAM\Samples\TestDriver\SampleTests**. Type the file name **CreateSession2Test**, and click **OK**.

4. VIEWING AN IMAGE

There are two strategies for deploying and starting the Smart Review client. At any one time, the SRS server can support both strategies.

- **JNLP Smart Review client:** The client computer uses the Java Network Launch Protocol to automatically download the Smart Review client from the SRS server the first time a user launches a session, and checks for updates to the client on each subsequent launch. Java is required on the client machine to run the JNLP Smart Review client.
- **Installed Smart Review client:** You deploy the Smart Review client executable and the user installs the application on the client computer (similar to how the legacy RTP Proofing client was deployed). The installed PC client installs its own copy of the required Java Runtime Environment. The installed Mac client relies on the Java Runtime Environment that exists on the computer as a part of the Mac operating system.

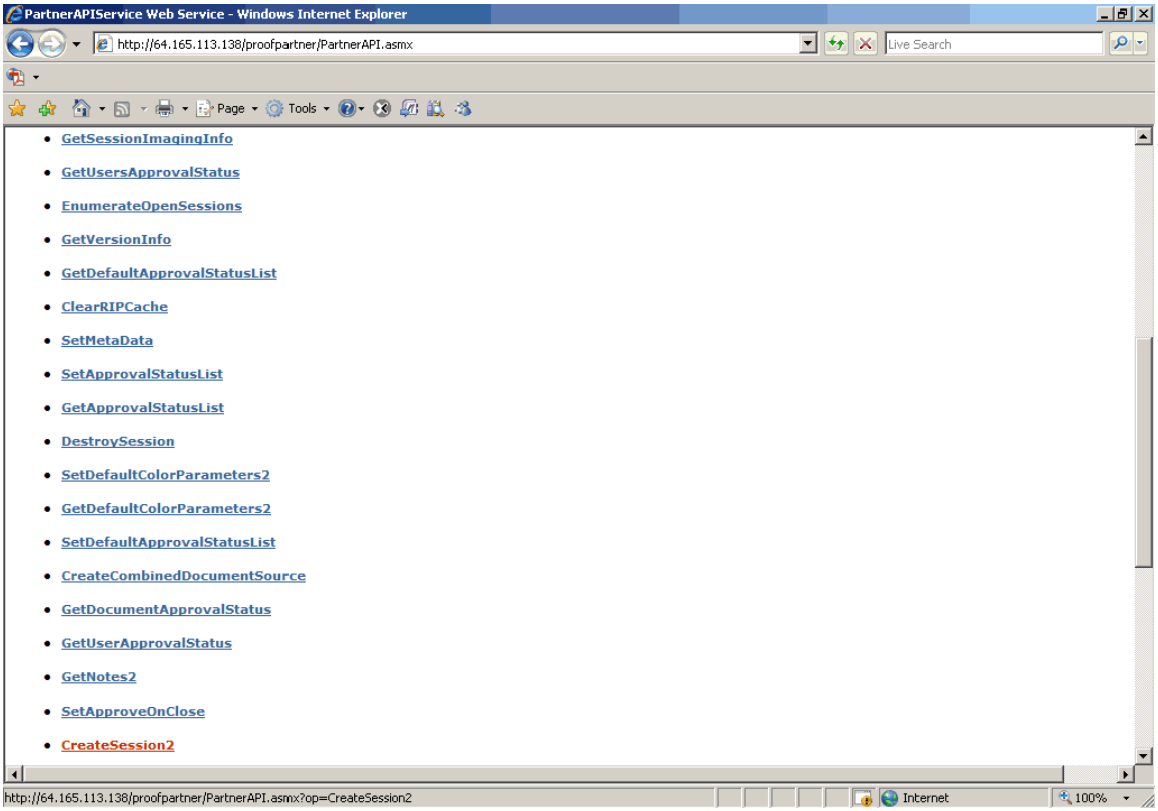
To view an image using the JNLP Smart Review client (PC and Mac):

For simplicity, we illustrate testing the JNLP Smart Review client using the test page automatically generated by Microsoft .NET Framework® at **PartnerAPI.asmx**.

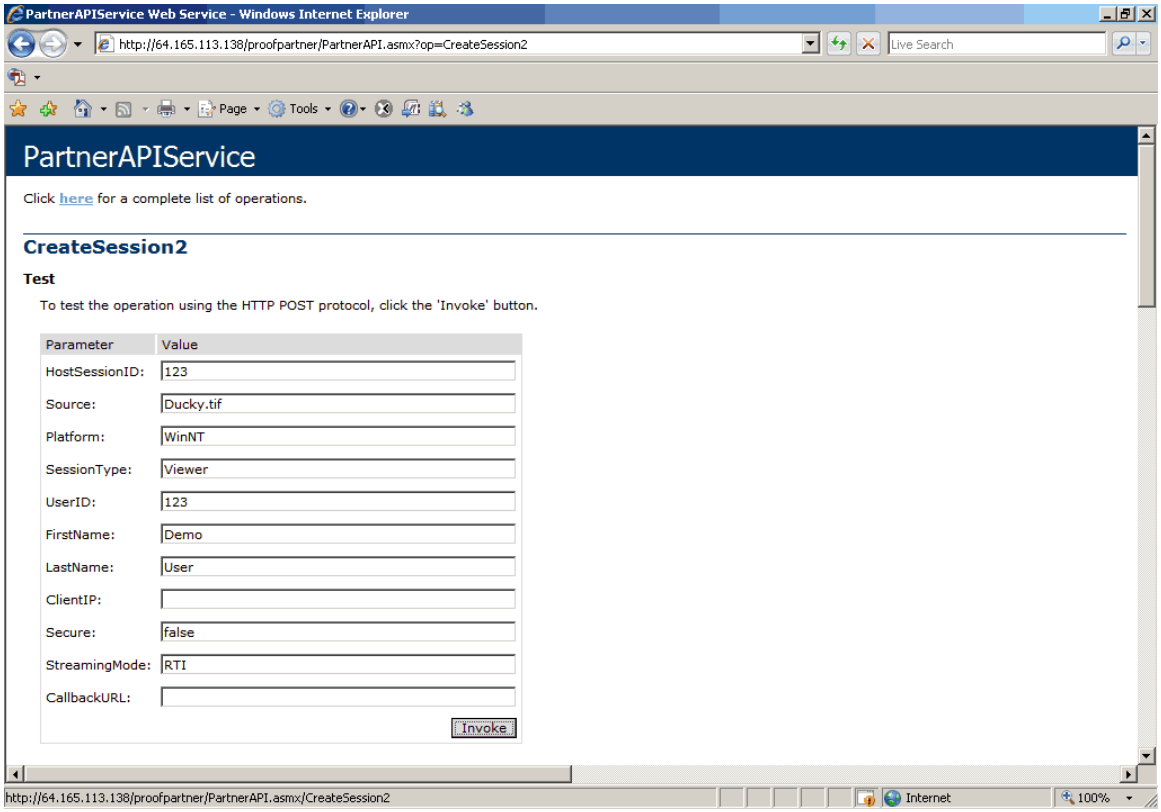
1. Java must be installed on the client computer. The recommended version for SRS is 1.4 or 1.5. You can check the Java version on a PC from **Control Panel > Java > General > About**. On a Mac, select **Finder > Utilities > System Profiler > Software > Applications > Java**. If necessary, you can download and install the latest Java version from http://www.java.com. Generally, the Smart Review client is compatible with the latest available Java version.

2. On the client computer, open a Web browser window and type **http://[serverurl]/proofpartner/PartnerAPI.asmx** for the URL address—for example, **http://64.165.113.138/proofpartner/PartnerAPI.asmx**.

3. Click **CreateSession2**.



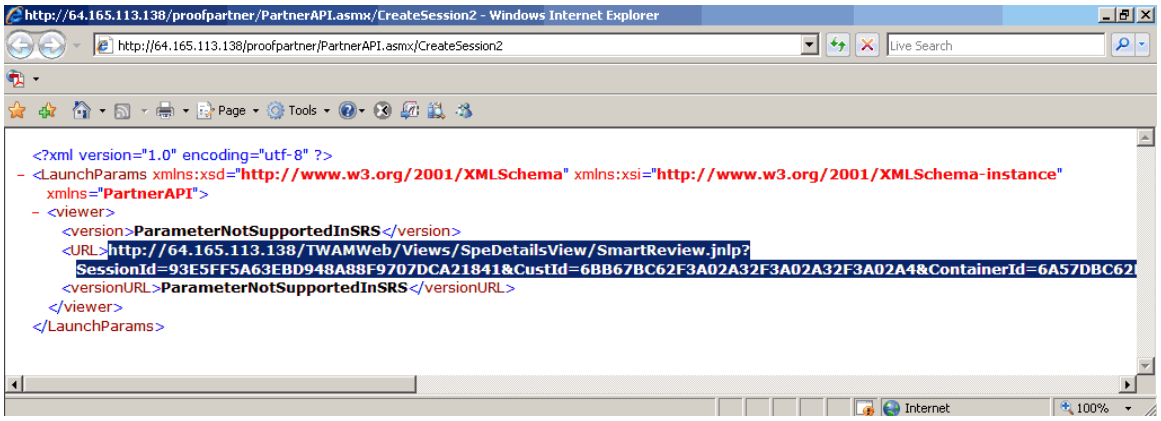
4. Fill in the Parameter table with test values, and click **Invoke**.



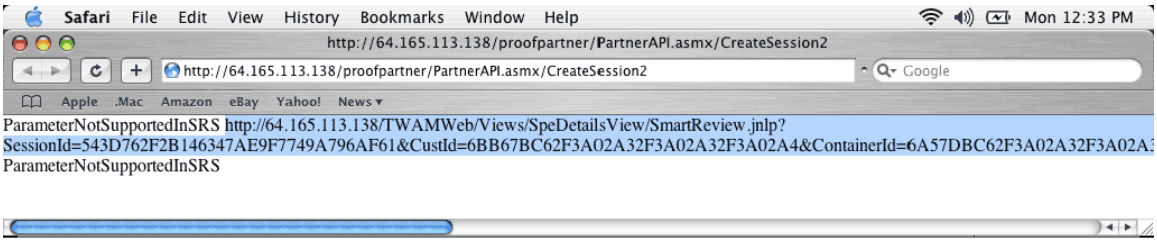
The `URL` tag contains the address that you will use to launch the client.

Note: The Apple® Safari® Web browser may not display XML on screen, but the XML is available from **View > View Source**.

On a PC:

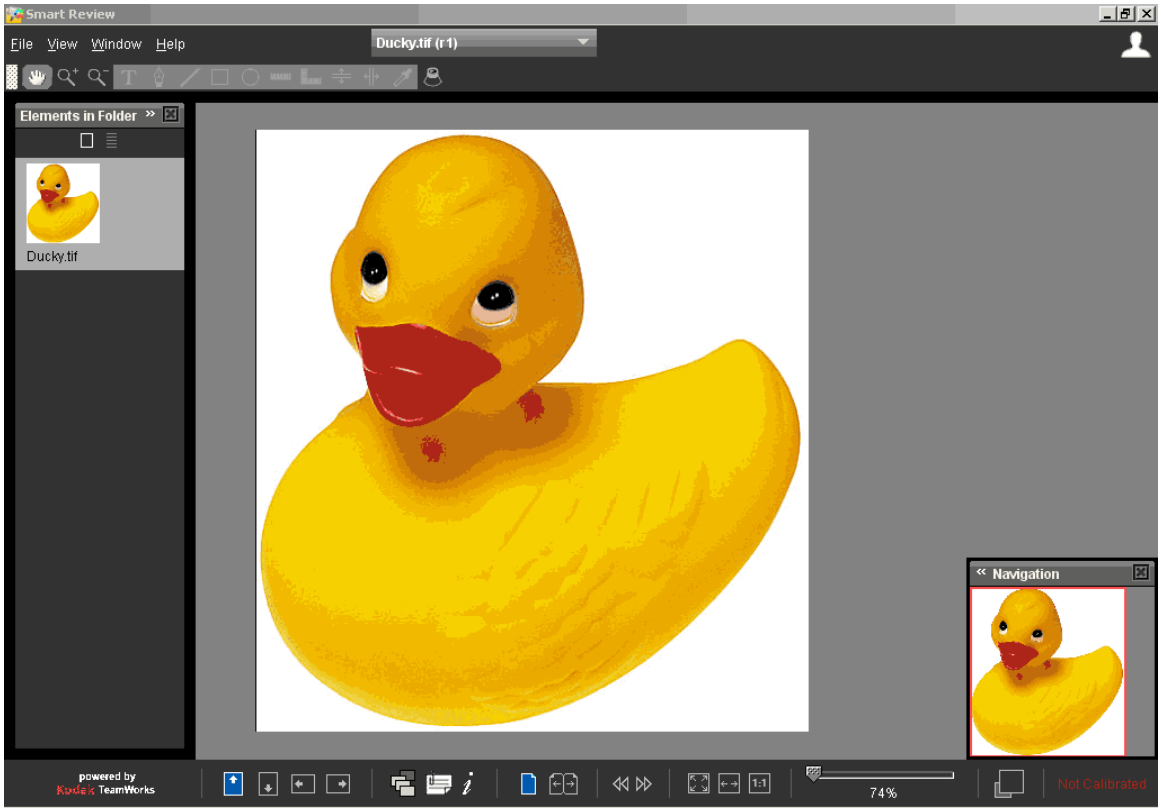


On a Mac:

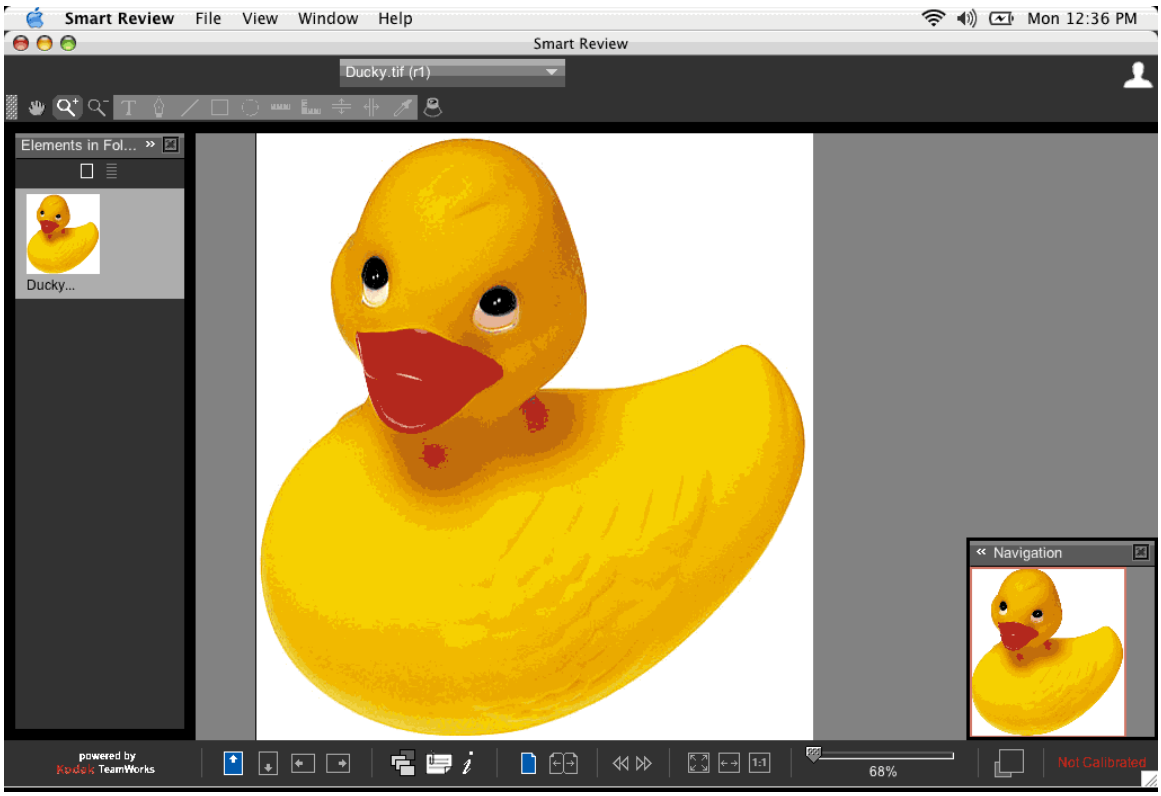


5. Copy the complete URL into the URL address of the Web browser. The JNLP Smart Review client starts.

On a PC:



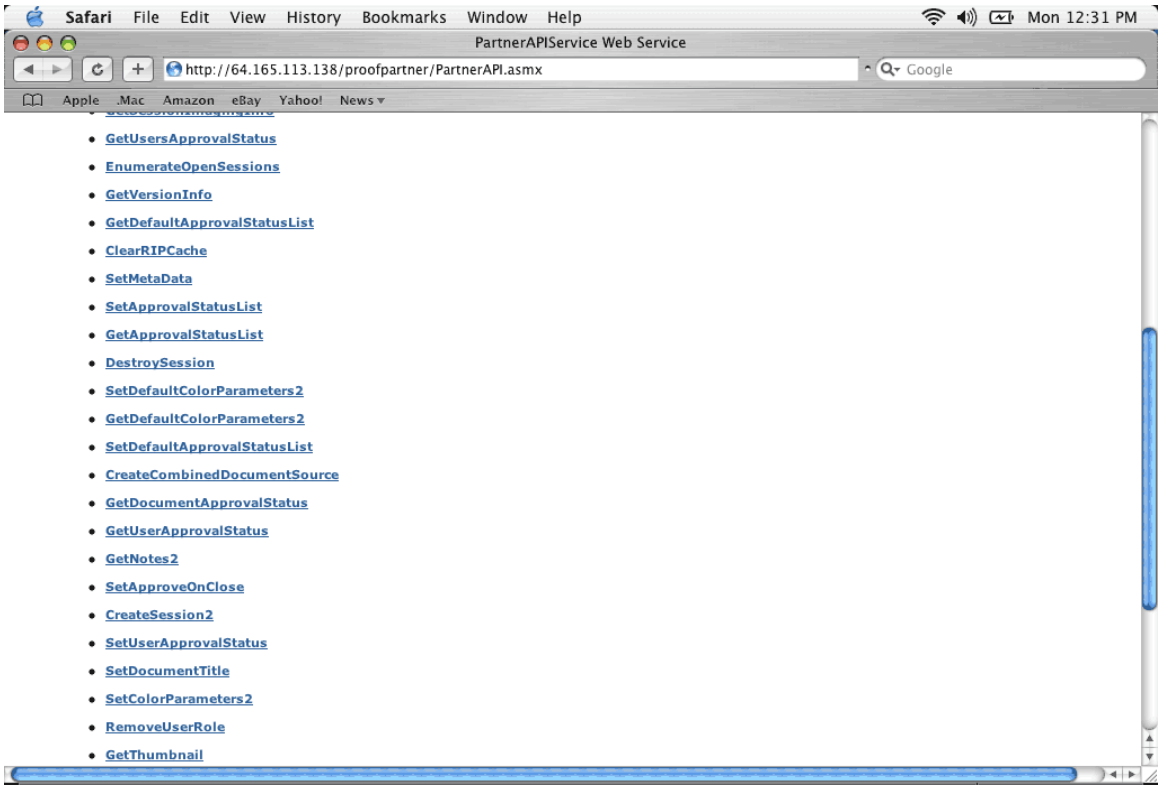
On a Mac:



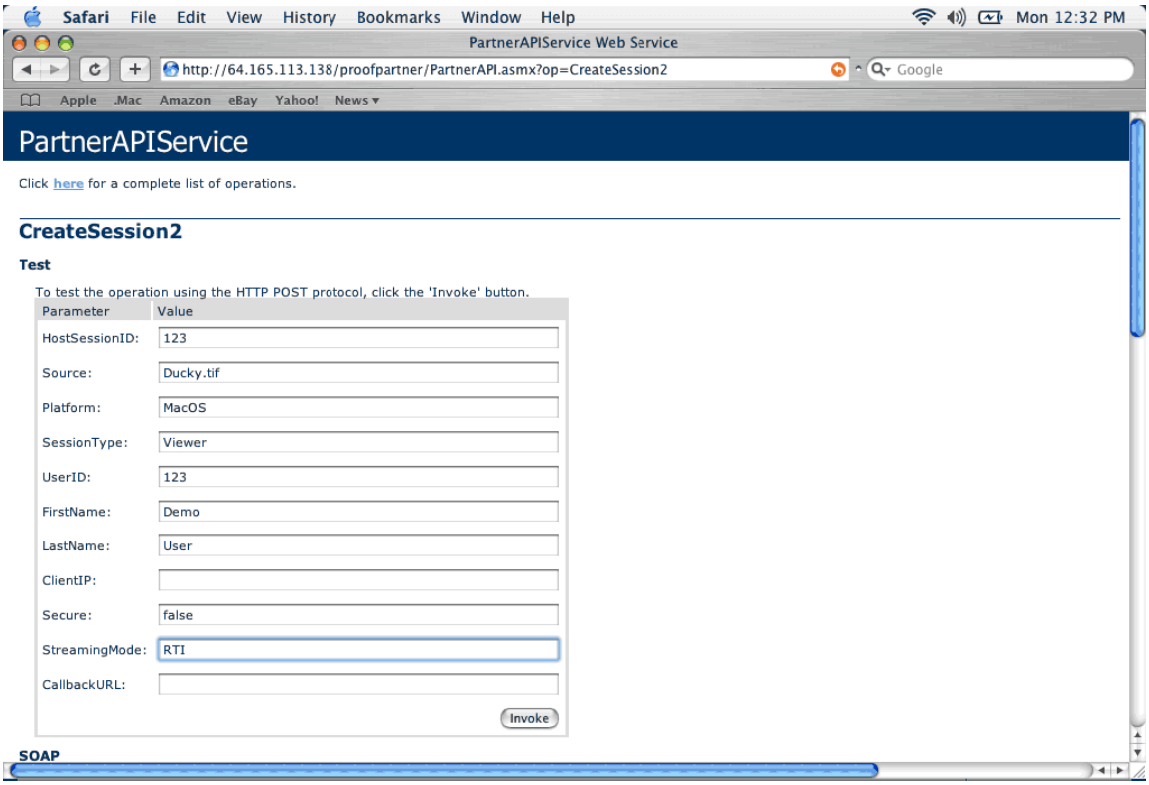
To view an image using the installed Smart Review client (Mac):

For simplicity, we illustrate testing the installed Mac Smart Review client using the test page automatically generated by Microsoft .NET Framework at **PartnerAPI.asmx**.

1. On the SRS DVD, navigate to the folder **SRS\SRSClient_Mac** and double-click **SmartReview 1.0 (XXX).dmg** to mount a disk on the Mac. Drag the Smart Review application to the computer's **Applications** folder.
2. On the client computer, open a Web browser window and type **http://[serverurl]/proofpartner/PartnerAPI.asmx** for the URL address—for example, http://64.165.113.138/proofpartner/PartnerAPI.asmx.
3. Click **CreateSession2**.


Safari File Edit View History Bookmarks Window Help
Mon 12:31 PM
PartnerAPIService Web Service
http://64.165.113.138/proofpartner/PartnerAPI.asmx
Google
Apple .Mac Amazon eBay Yahoo! News
GetUsersApprovalStatus
EnumerateOpenSessions
GetVersionInfo
GetDefaultApprovalStatusList
ClearRIPCache
SetMetaData
SetApprovalStatusList
GetApprovalStatusList
DestroySession
SetDefaultColorParameters2
GetDefaultColorParameters2
SetDefaultApprovalStatusList
CreateCombinedDocumentSource
GetDocumentApprovalStatus
GetUserApprovalStatus
GetNotes2
SetApproveOnClose
CreateSession2
SetUserApprovalStatus
SetDocumentTitle
SetColorParameters2
RemoveUserRole
GetThumbnail

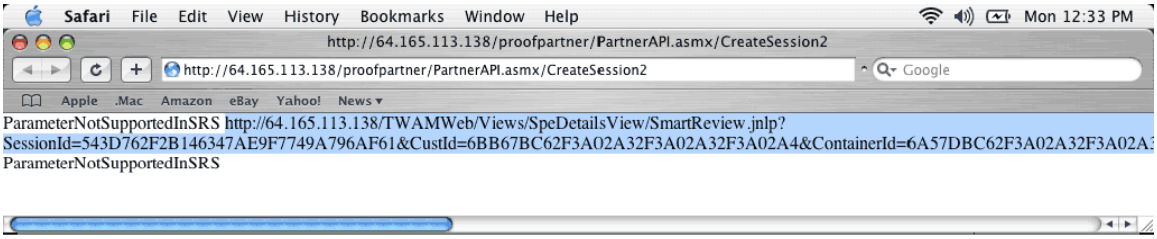
4. Fill in the Parameter table with test values, and click **Invoke**.



The `URL` tag contains the address that you will use to start the client.

Note: Safari may not display XML on screen, but the XML is available from **View > View Source**.

Copy this complete URL.

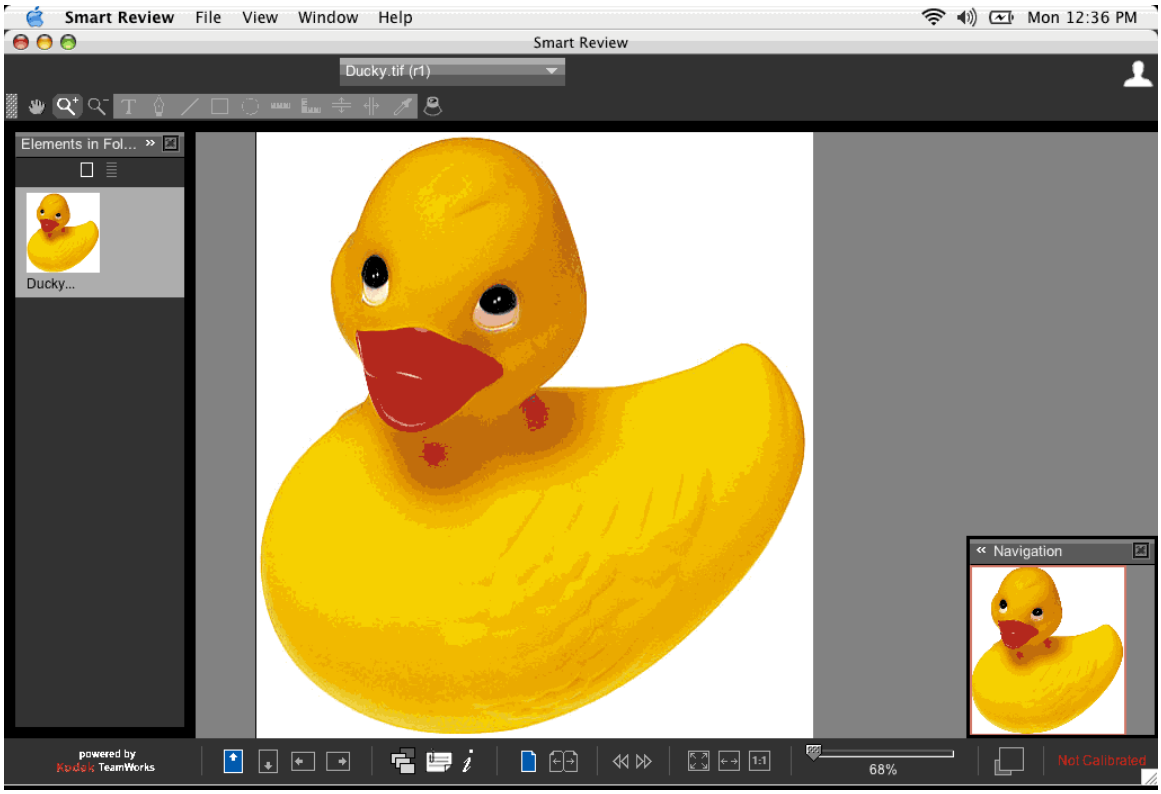


5. Open a command prompt window and navigate to where the Smart Review client is installed.

6. At the command prompt type `open "smartreview:<URL>"`

   where `<URL>` is this complete URL that you copied in step 5.

   The installed Smart Review client starts.

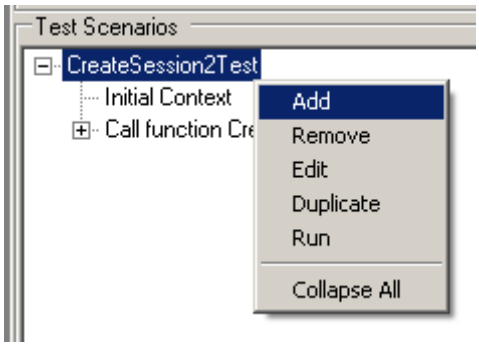


To view an image using the installed Smart Review client (PC):

For simplicity, we illustrate testing the installed Smart Review client using the PartnerTestDriver utility to generate function calls and perform launch.

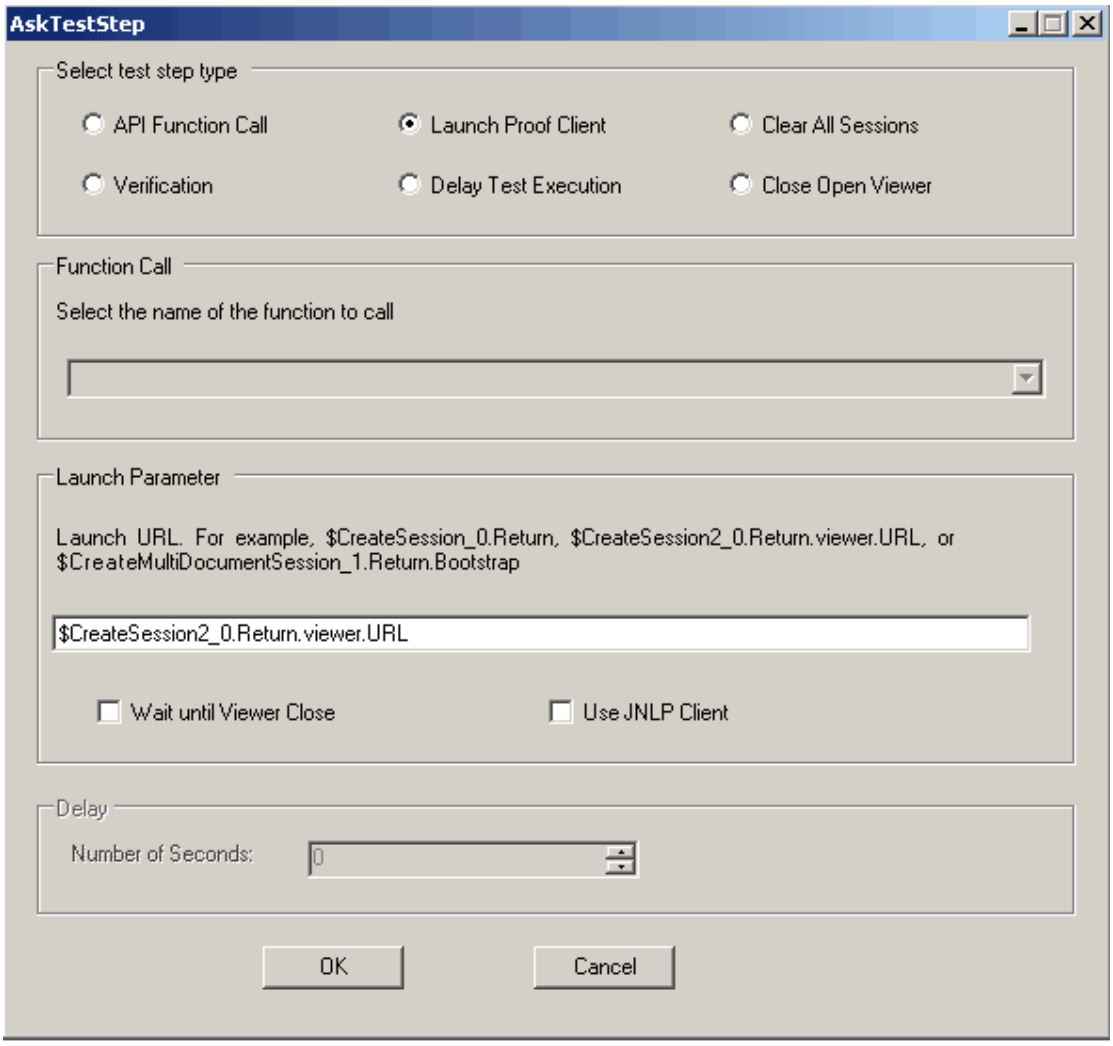
Note: The PartnerTestDriver utility is supported on the Windows platform only.

1. Confirm that the complete **TestDriver** folder is placed on the client computer.
2. On the SRS DVD, navigate to the folder **SRS\SRSClient_Windows** and double-click **Setup.exe**.
3. Follow the prompts to install the client. The executable also installs a local private copy of the Java run-time environment, which is used only by the Smart Review installed client.
4. Having completed all the steps in Section 3, *Using the PartnerTestDriver Utility*, from the PartnerTestDriver utility, select **File > Open**, and browse to **C:\Kodak\TWAM\Samples\TestDriver\SampleTests**. Select **CreateSession2Test.xml**, and click **OK**.
5. In the **Test Scenarios** box, right-click **CreateSession2Test** and click **Add**.

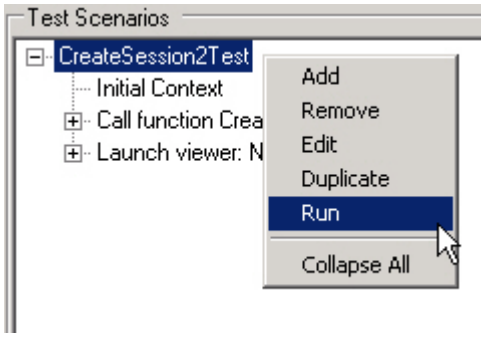


6. In the **Select test step type** area, select **Launch Proof Client**.

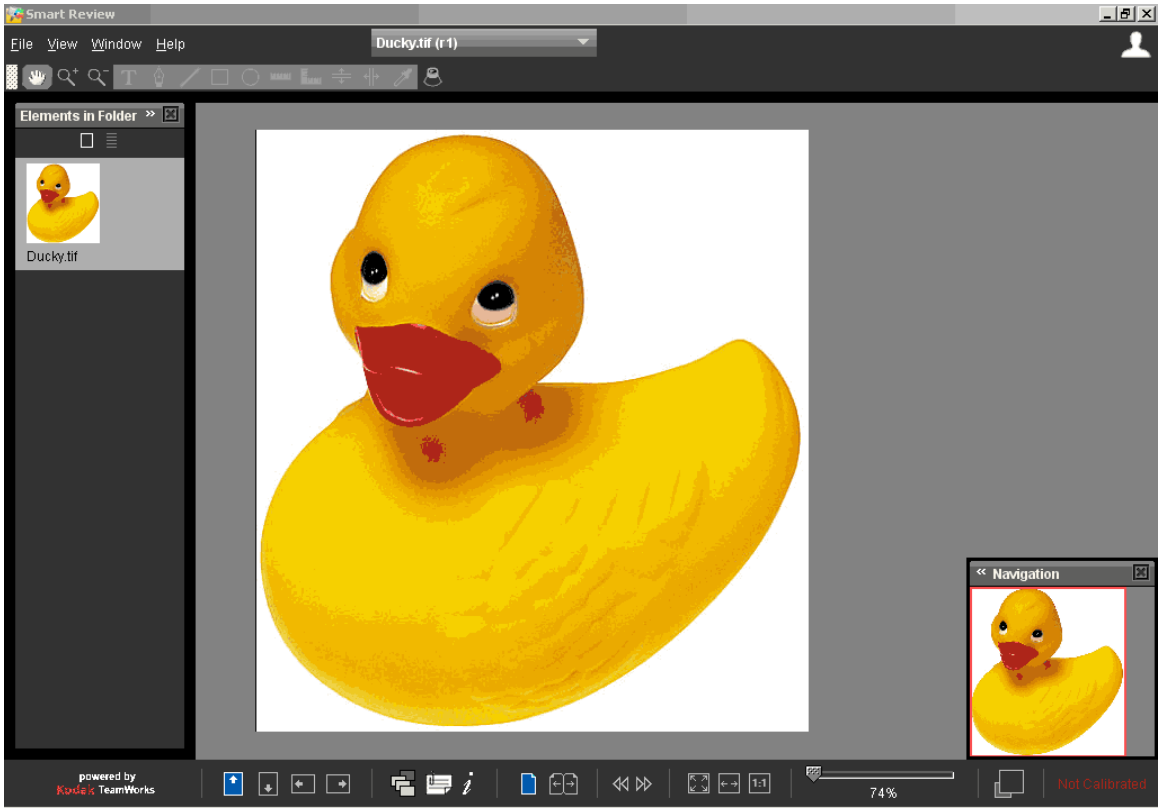
7. In the **Launch Parameter** area, type `$CreateSession2_0.Return.viewer.URL` (the returned information from the function call to CreateSession2). Click **OK**.



8. Right-click **CreateSession2Test** again and select **Run**.



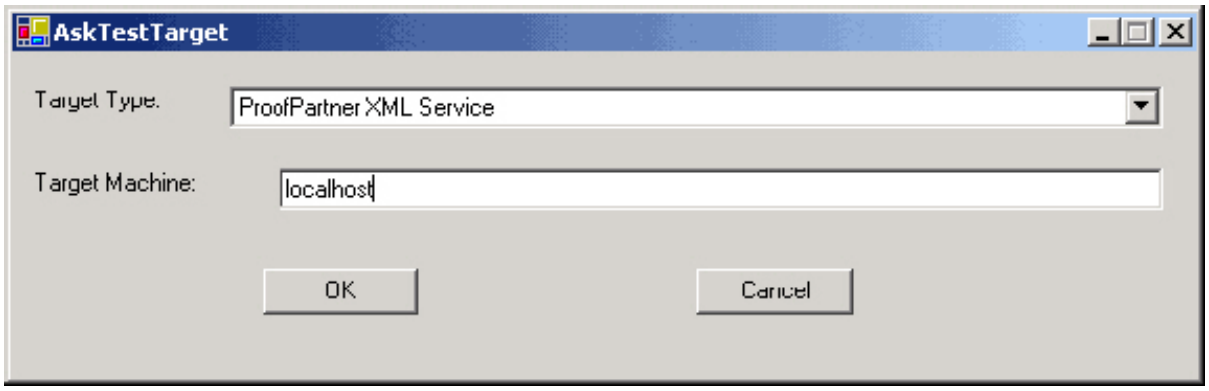
The installed Smart Review client starts.



5. COMPARING IMAGES

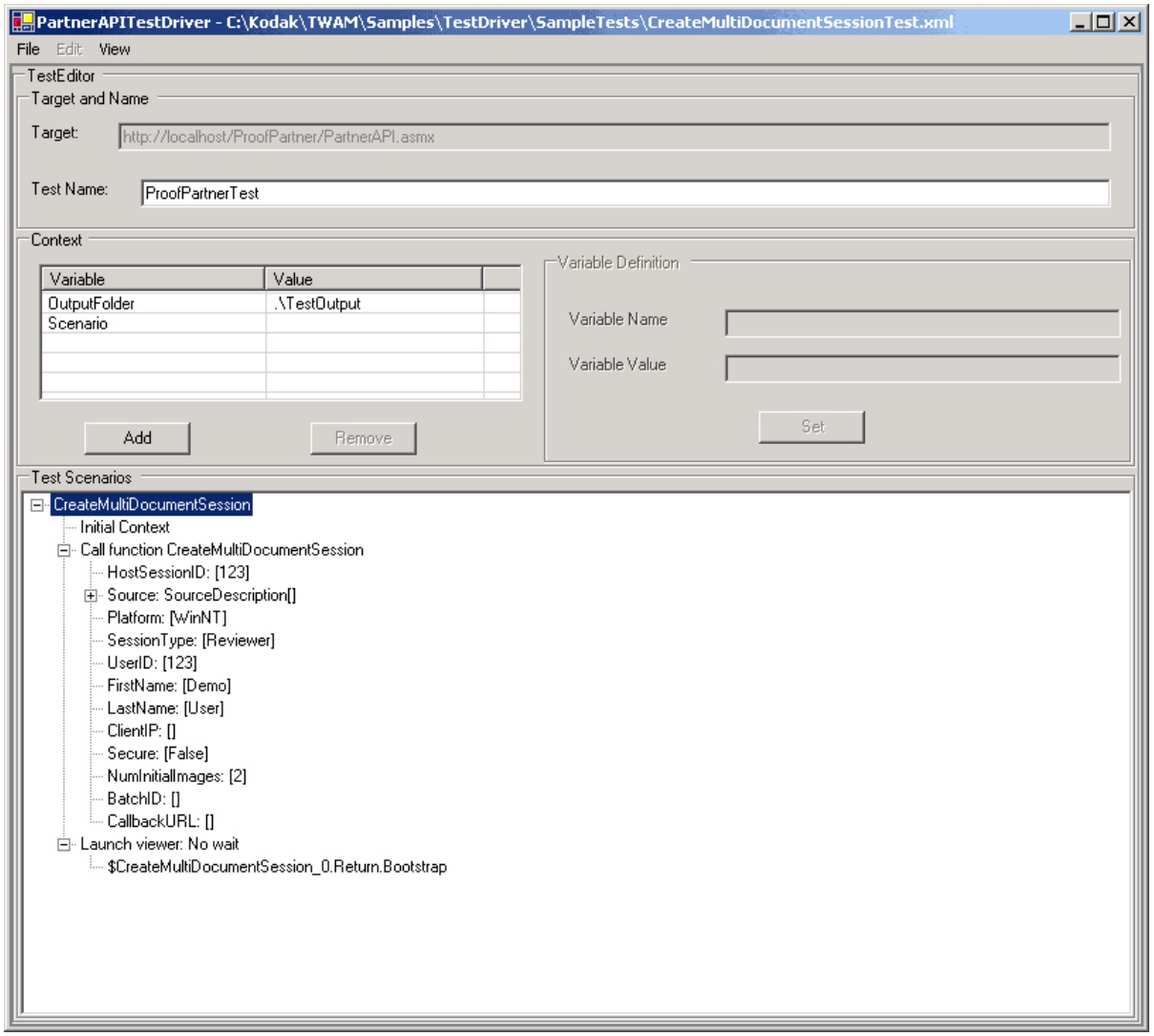
For simplicity, we illustrate comparing images using a pre-existing sample script for the PartnerAPITestDriver utility.

1. Run the PartnerAPITestDriver utility as described in Section 3, *Using the PartnerTestDriver Utility*. Select **File > Open**. In the **Target Machine** box, type the URL or server name of the SRS server, and click **OK**.



2. Browse to **C:\Kodak\TWAM\Samples\TestDriver\SampleTests,** select **CreateMultiDocumentSessionTest.xml**, and click **OK**.

   The PartnerAPITestDriver utility opens the script and populates the functions and test parameters into the Test Scenarios box. You can right-click and/or expand the trees to view and edit these parameters.

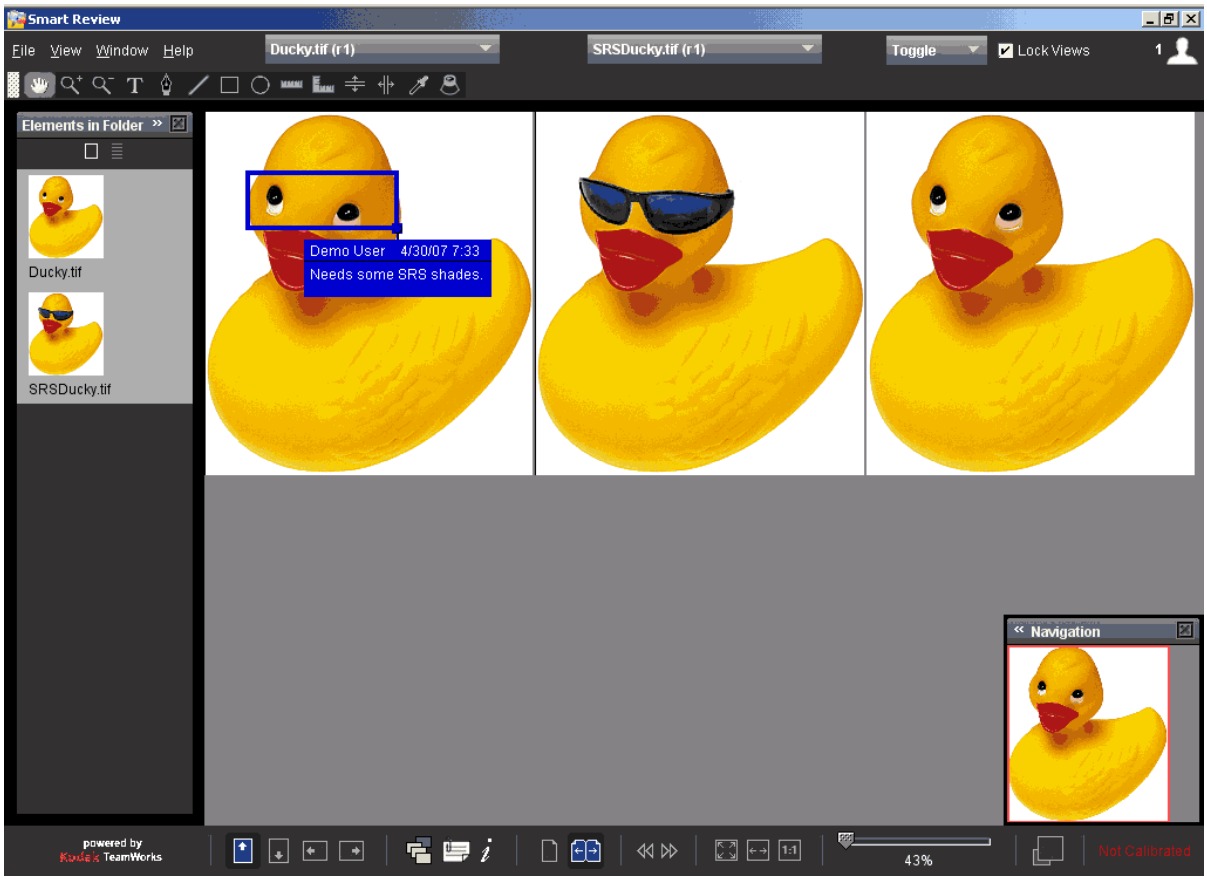


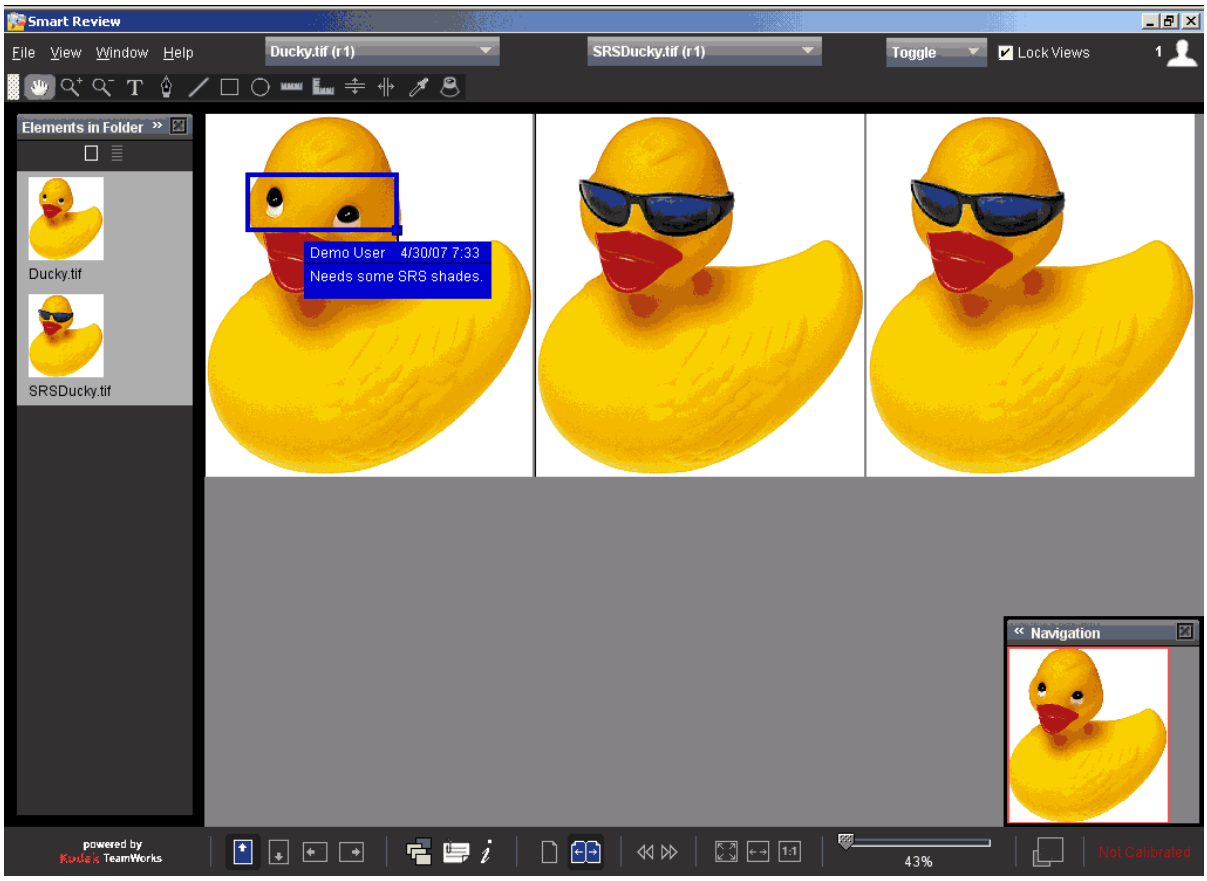
3. Right-click **CreateMultiDocumentSession** and select **Run**. A Smart Review session starts for the images **Ducky.tif** and **SRSDucky.tif**.

4. In preparation for Section 6, *Retrieving an Annotation*, add an annotation to **Ducky.tif** by selecting one of the text, pen, line, rectangle, or oval tools on the Smart Review toolbar, and clicking on the image.

5. Select **View** > **Compare**.

   SRSDucky.tif is loaded and the two images are compared.





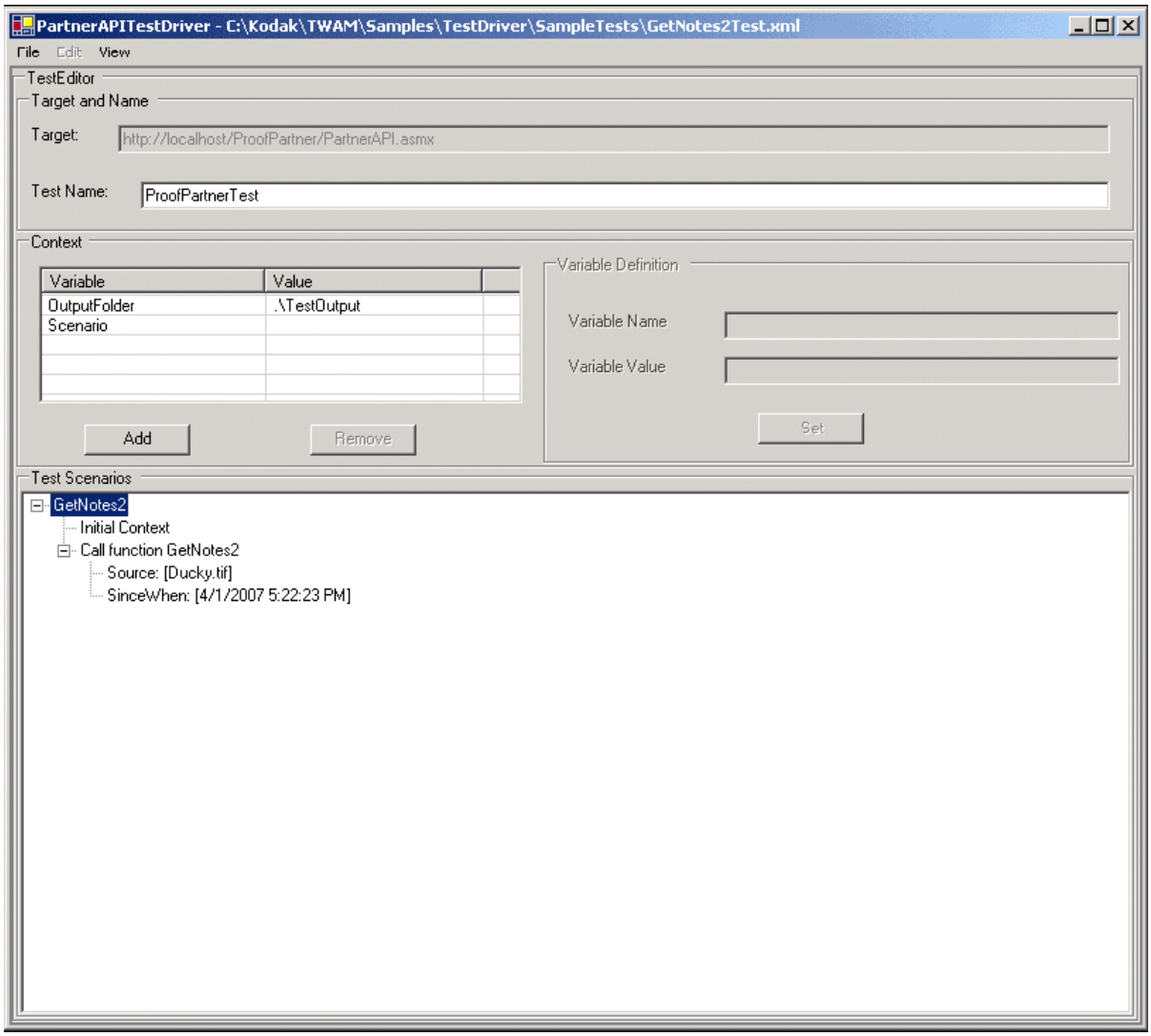
6. RETRIEVING AN ANNOTATION

For simplicity, we illustrate retrieving an annotation using a pre-existing sample script for the PartnerAPITestDriver utility.

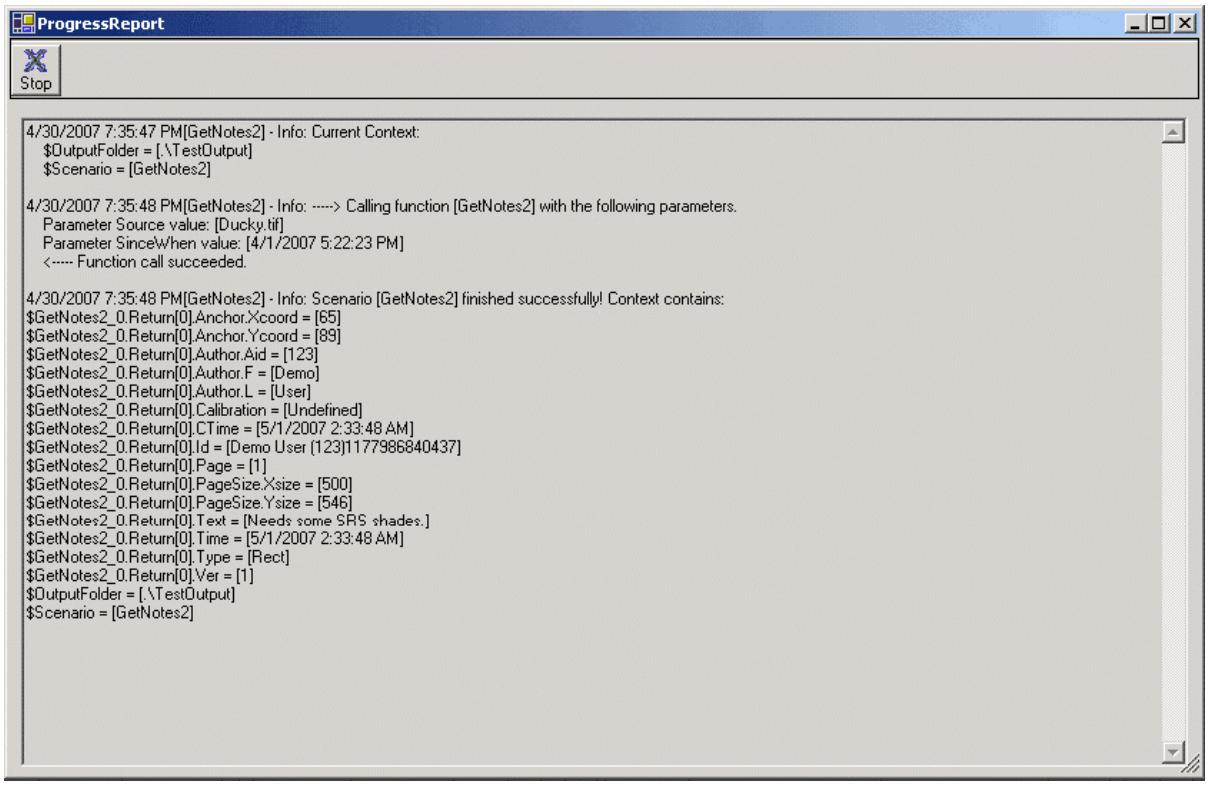
1. Run the PartnerAPITestDriver utility as described in Section 3, *Using the PartnerTestDriver Utility*. Select **File > Open**. In the **Target Machine** box, type the URL or server name of the SRS server, and click **OK**.

2. Browse to **C:\Kodak\TWAM\Samples\TestDriver\SampleTests**, select **GetNotes2Test.xml**, and click **OK**.

   The PartnerAPITestDriver utility opens the script and populates the functions and test parameters into the Test Scenarios box. You can right-click and/or expand the trees to view and edit these parameters.



3. In the **Test Scenarios** box, right-click **GetNotes2** and select **Run**. The ProgressReport should demonstrate the call to GetNotes2 and collect the existing notes on **Ducky.tif**.



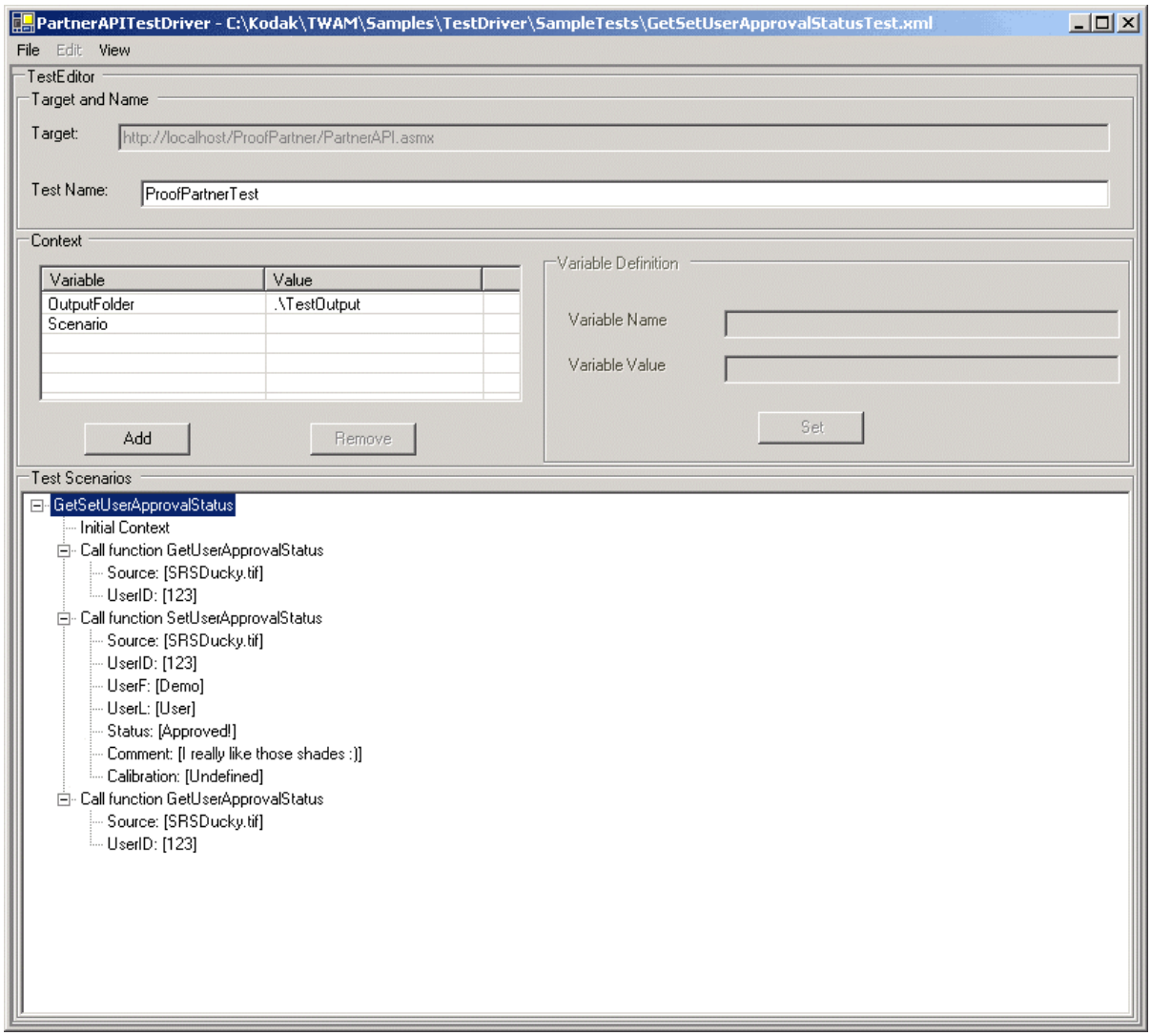
7. IMPLEMENTING APPROVAL

For simplicity, we illustrate implementing an approval cycle using a pre-existing sample script for the PartnerAPITestDriver utility.

1. Run the PartnerAPITestDriver utility as described in Section 3, *Using the PartnerTestDriver Utility*. Select **File > Open**. In the **Target Machine** box, type the URL or server name of the SRS server, and click **OK**.

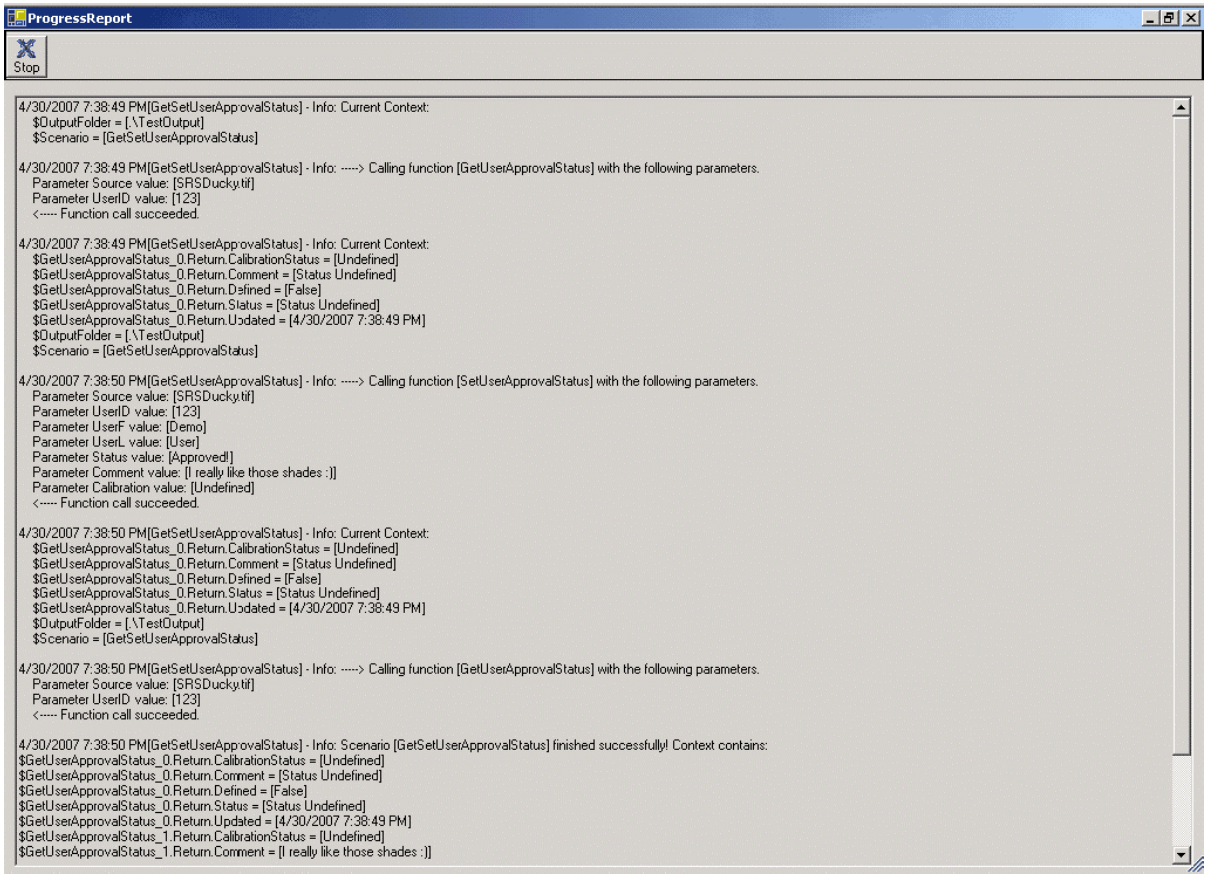
2. Browse to **C:\Kodak\TWAM\Samples\TestDriver\SampleTests**, select **GetSetUserApprovalStatusTest.xml**, and click **OK**.

   The PartnerAPITestDriver utility opens the script and populates the functions and test parameters into the Test Scenarios box. You can right-click and/or expand the trees to view and edit these parameters.



3. In the **Test Scenarios** box, right-click **GetSetUserApprovalStatus** and select **Run**.

   The ProgressReport should demonstrate the calls to GetUserApprovalStatus, SetUserApprovalStatus, and again GetUserApprovalStatus, and collect the newly created user approval status on **SRSDucky.tif**.



8. INTEGRATING WITH THE CLIENT SOFTWARE

The Smart Review client launch procedure differs from RTP Partner in that you will no longer deliver a bootstrap file with the extensions **ppc** or **ppp**.

Note: An existing integration must be revisited to address this difference.

There are two types of Smart Review clients to launch: the JNLP Smart Review client and the installed Smart Review client.

8.1 Integrating With the JNLP Smart Review Client

To integrate the JNLP Smart Review client, you must redirect client browsers to the SessionReference URL returned by a session creation function. Below is an example of how to do this in ASP:

```
<%
Option Explicit
Response.Buffer = True

'Make a SOAP message
dim BaseURL, SoapAction, Req
BaseURL = "http://localhost/ProofPartner/PartnerAPI.asmx"
SoapAction = "urn:PartnerAPICreateSession2Action"
Req = "<?xml version=""1.0"" encoding=""utf-8""?>" & vbCrLf
Req = Req & "<soap:Envelope xmlns:xsi=""http://www.w3.org/2001/XMLSchema-"
Req = Req & "instance"" xmlns:xsd=""http://www.w3.org/2001/XMLSchema"" "
Req = Req & "xmlns:soap=""http://schemas.xmlsoap.org/soap/envelope/"">"
Req = Req & vbCrLf
```

```
Req = Req & "<soap:Body>" & vbCrLf
Req = Req & "<CreateSession2 xmlns=""urn:PartnerAPI"">" & vbCrLf
Req = Req & "<HostSessionID>1</HostSessionID>" & vbCrLf
Req = Req & "<Source>ducky.tif</Source>" & vbCrLf
Req = Req & "<Platform>WinNT</Platform>" & vbCrLf
Req = Req & "<SessionType>Reviewer</SessionType>" & vbCrLf
Req = Req & "<UserID>1</UserID>" & vbCrLf
Req = Req & "<FirstName>Demo</FirstName>" & vbCrLf
Req = Req & "<LastName>User</LastName>" & vbCrLf
Req = Req & "<ClientIP></ClientIP>" & vbCrLf
Req = Req & "<Secure>false</Secure>" & vbCrLf
Req = Req & "<StreamingMode>RTI</StreamingMode>" & vbCrLf
Req = Req & "<CallbackURL></CallbackURL>" & vbCrLf
Req = Req & "</CreateSession2>" & vbCrLf
Req = Req & "</soap:Body>" & vbCrLf
Req = Req & "</soap:Envelope>" & vbCrLf

'Post SOAP message to SRS and catch response
dim xml, returnValue
Set xml = Server.CreateObject("Microsoft.XMLHTTP")
xml.Open "POST", BaseURL, False
xml.setRequestHeader "Host", "localhost"
xml.setRequestHeader "Content-Type", "text/xml; charset=utf-8"
xml.setRequestHeader "Content-Length", len(Req)
xml.setRequestHeader "SOAPAction", SoapAction
xml.Send Req
returnValue = xml.Responsetext
Set xml = nothing

'Extract SessionReference from response
dim SessionReference
SessionReference = getBetween(returnValue, "<URL>", "</URL>")

'Make HTML page string
dim RedirectURL
RedirectURL = " <script language=JavaScript>" & vbCrLf
RedirectURL = RedirectURL & " <!--" & vbCrLf
RedirectURL = RedirectURL & " if (window.navigate) " & vbCrLf
RedirectURL = RedirectURL & "     window.navigate('"
RedirectURL = RedirectURL & Cstr(SessionReference)
RedirectURL = RedirectURL & "');" & vbCrLf
RedirectURL = RedirectURL & " else " & vbCrLf
RedirectURL = RedirectURL & " window.location='" & Cstr(SessionReference)
RedirectURL = RedirectURL & "';" & vbCrLf
RedirectURL = RedirectURL & " //-->" & vbCrLf
RedirectURL = RedirectURL & " </script>" & vbCrLf

'This is the first write sent to user. It redirects to SessionReference
Response.Write(RedirectURL)

Function getBetween(strSource, strBegin, strEnd )
'Simple XML parser, assumes tags exist and first instance is what you want
    dim beginFound, endFound, getStart, getCount
    beginFound = Instr(1, strSource, strBegin)
    endFound = Instr(1, strSource, strEnd)
    getStart = beginFound + len(strBegin)
    getCount = endFound - beginFound - len(strBegin)
    getBetween=mid(strSource, getStart, getCount)
End Function

%>
```

8.2 Integrating With the Installed Smart Review Client

To integrate with the installed Smart Review client, you must make a command line system call to the executable **SmartReview.exe** with an altered SessionReference URL as the command line argument. Specifically, you must put "`smartreview:`" at the beginning of the string and enclose the string in quotes.

This is most practical when you have a client application of your own from where you can shell to the client operating system to call **SmartReview.exe**.

Below is an example of how to do this in ASP:

```
<%
Option Explicit
Response.Buffer = True

'Make a SOAP message
dim BaseURL, SoapAction, Req
BaseURL = "http://localhost/ProofPartner/PartnerAPI.asmx"
SoapAction = "urn:PartnerAPICreateSession2Action"
Req = "<?xml version=""1.0"" encoding=""utf-8""?>" & vbCrLf
Req = Req & "<soap:Envelope xmlns:xsi=""http://www.w3.org/2001/XMLSchema-"
Req = Req & "instance"" xmlns:xsd=""http://www.w3.org/2001/XMLSchema"" "
Req = Req & "xmlns:soap=""http://schemas.xmlsoap.org/soap/envelope/"">"
Req = Req & vbCrLf
Req = Req & "<soap:Body>" & vbCrLf
Req = Req & "<CreateSession2 xmlns=""urn:PartnerAPI"">" & vbCrLf
Req = Req & "<HostSessionID>1</HostSessionID>" & vbCrLf
Req = Req & "<Source>ducky.tif</Source>" & vbCrLf
Req = Req & "<Platform>WinNT</Platform>" & vbCrLf
Req = Req & "<SessionType>Reviewer</SessionType>" & vbCrLf
Req = Req & "<UserID>1</UserID>" & vbCrLf
Req = Req & "<FirstName>Demo</FirstName>" & vbCrLf
Req = Req & "<LastName>User</LastName>" & vbCrLf
Req = Req & "<ClientIP></ClientIP>" & vbCrLf
Req = Req & "<Secure>false</Secure>" & vbCrLf
Req = Req & "<StreamingMode>RTI</StreamingMode>" & vbCrLf
Req = Req & "<CallbackURL></CallbackURL>" & vbCrLf
Req = Req & "</CreateSession2>" & vbCrLf
Req = Req & "</soap:Body>" & vbCrLf
Req = Req & "</soap:Envelope>" & vbCrLf

'Post SOAP message to SRS and catch response
dim xml, returnValue
Set xml = Server.CreateObject("Microsoft.XMLHTTP")
xml.Open "POST", BaseURL, False
xml.setRequestHeader "Host", "localhost"
xml.setRequestHeader "Content-Type", "text/xml; charset=utf-8"
xml.setRequestHeader "Content-Length", len(Req)
xml.setRequestHeader "SOAPAction", SoapAction
xml.Send Req

returnValue = xml.Responsetext
Set xml = nothing

'Extract SessionReference from response
dim SessionReference
SessionReference = getBetween(returnValue, "<URL>", "</URL>")

'Replaces several &amp; with &
dim CleanSessionReference
CleanSessionReference=Replace(SessionReference,"&amp;","&",1,6)

'quote, path, exe, quote, space, quote, smartreview:, response, quote
```

```
dim batchText
batchText = Chr(34)
batchText = batchText & "C:\Program Files\Kodak Smart Review\bin\"
batchText = batchText & "SmartReview.exe"
batchText = batchText & Chr(34)
batchText = batchText & " "
batchText = batchText & Chr(34)
batchText = batchText & "smartreview:" & Cstr(CleanSessionReference)
batchText = batchText & Chr(34) & vbCrLf

'This sends batch file
Response.ContentType="text/plain"
Response.AddHeader "Content-Disposition", "attachment;filename=srs.bat"
Response.Write(batchText)

Function getBetween(strSource, strBegin, strEnd )
'Simple XML parser, assumes tags exist and first instance is what you want
    dim beginFound, endFound, getStart, getCount
    beginFound = Instr(1, strSource, strBegin)
    endFound = Instr(1, strSource, strEnd)
    getStart = beginFound + len(strBegin)
    getCount = endFound - beginFound - len(strBegin)
    getBetween=mid(strSource, getStart, getCount)
End Function

%>
```

9. USING THE CALLBACK SAMPLE CODE

The **PartnerAPICallback.wsdl** of SRS differs from RTP Partner in that it has been changed from an RPC Encoded form to a Document Literal form. An existing integration must be revisited to address these differences.

A sample callback service implementation is included in the SRS installation. To configure this service, complete the following steps.

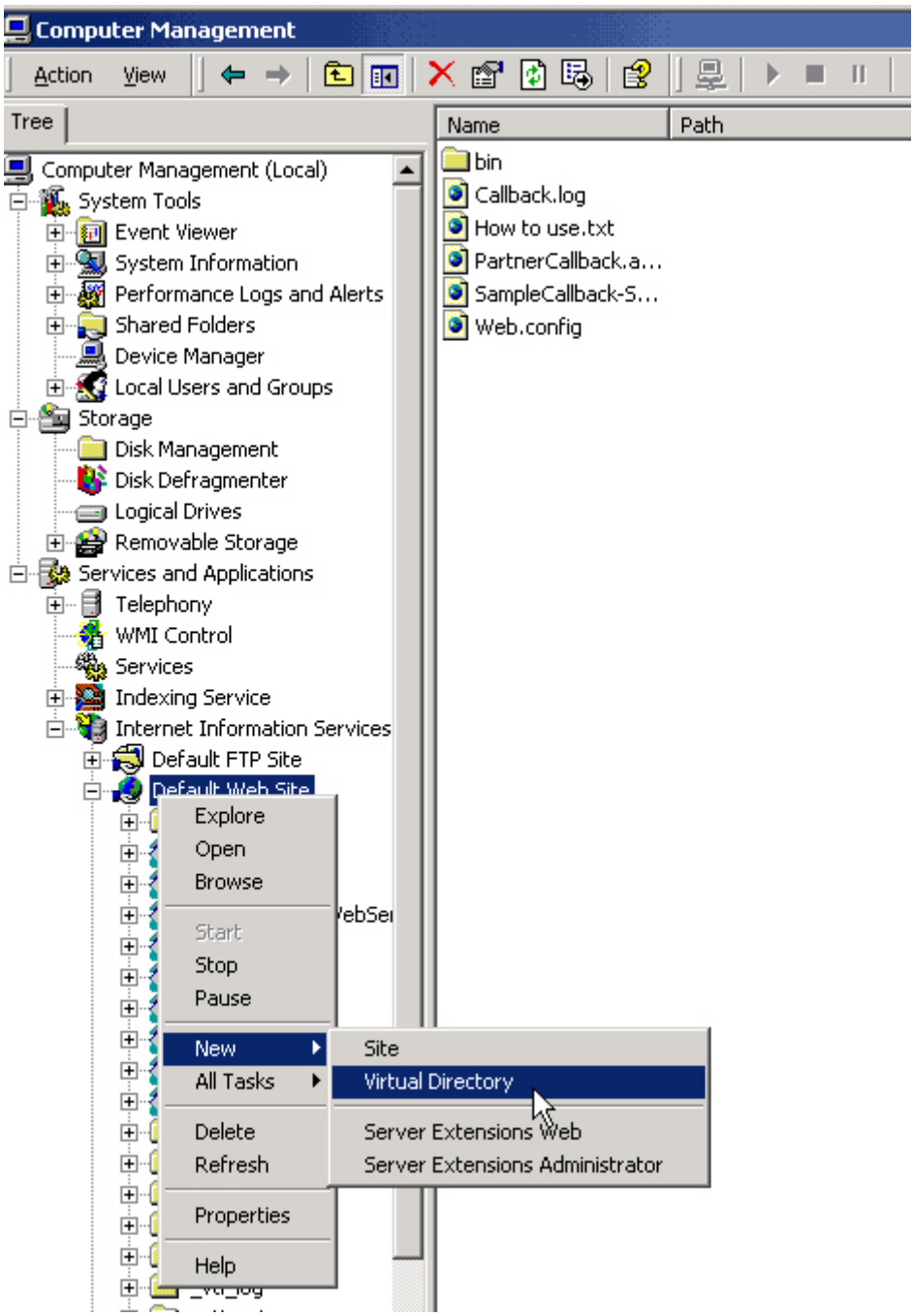
1. Copy the contents of **C:\Kodak\TWAM\Samples\Callback to C:\SrsCallbackTester**.

   Note: The callback tester will not work if the folder **C:\SrsCallbackTester** does not exist on the computer being configured as the callback test server.

   The contents of the **C:\SrsCallbackTester** folder should now contain:

   - **Web.config**
   - **SampleCallback.asmx**
   - **bin** folder containing the following files:
     - **SoapTracing.dll**
     - **SoapTracing.pdb**
     - **SampleCallback.dll**
     - **SampleCallback.pdb**

2. Select **Computer Management > Services and Applications > Internet Information Services**, right-click **Default Web Site**, and select **New > Virtual Directory**.



3. Using the Virtual Directory Creation Wizard, give your **Virtual Directory** an alias and, for your **Web Site Content Directory**, browse to **C:\SrsCallbackTester**. In the Access Permissions dialog box, ensure that, in addition to the default settings, **Run Scripts** is selected. Accept the defaults for the remaining settings and click **Finish**. Configure the SRS server with the callback service at **http://[servername]/SrsCallbackTester[or_your_chosen_Alias]/SampleCallback.asmx** before creating a session (or on creating a session, for the parameter CallbackURL, enter the value **http://[servername]/SrsCallbackTester[or_your_chosen_Alias]/SampleCallback.asmx**).

   Note: If the callback service is on the same server as SRS, **[servername]** is **localhost** or **127.0.0.1**.

If callbacks are expected from activities on the session, you will find log entries in the file **C:\SrsCallbackTester\Callback.log**.

If callbacks are expected from activities on the session, you will also find SOAP log entries in the file **C:\SrsCallbackTester\SampleCallback-SoapTrace.log**.

10. PRE-CACHING IMAGES

The pre-caching procedure differs in SRS from RTP Partner in that you no longer need to QuerySessionImagingStatus (now deprecated) in a loop after creating a session. Existing pre-caching scripts must be revisited to address this difference.

In SRS, when the system is configured to pre-cache, caching is triggered when a source is added to the system. To configure SRS to pre-cache before running your scripts:

1. In Smart Review System Administration, click **Rendering**.
2. Confirm that the rendering options are configured as follows:
   - **Raster Cache Image Size** is set to the appropriate size for your normal monitor (**1600** for smaller monitors; up to **3200**, but not higher, for larger monitors)
   - **Pre-Render Processing Level** is set to **2**
   - **Preserve Intermediate Render Cache** is selected

   For more information on each of these settings, see *Configuring the System* in the *Smart Review System 1.0 Administration Guide*.
3. Click **Update**.

   A source will be “added” to the system, and thus cached, on first call of any of the following APIs:

   CreateSession
   CreateSession2
   CreateMultiDocumentSession
   GetFileInfo
   GetFileType
   SetColorParameters2
   GetColorParameters2
   SetUserApprovalStatus
   SetApprovalStatusList
   SetDocumentApprovalStatus

   For simplicity, GetFileInfo is recommended for regular pre-caching, and SetColorParameters2 is recommended for pre-caching files that require a color target different from the system default or files that have been previously cached and need to have their cache updated after alteration of their associated color target.