Tech Note 527
Generating an ActiveFactory Workbook Report within InTouch®

Topic#: 002263
Created: March 2008

Introduction

This *Tech Note* provides step by step directions and sample code to run an ActiveFactory Workbook using InTouch.

Application Versions

- Wonderware ActiveFactory 9.2 and later. ActiveFactory supports Microsoft® Excel 2000, XP, 2003, and 2007.
- Wonderware InTouch 10.0 and later

Create an ActiveFactory Workbook (Example)

1. Open ActiveFactory Workbook by clicking **Start / All Programs / Wonderware / ActiveFactory / Workbook**.
2. Create a new blank sheet.
3. Go to **ActiveFactory / Connection Management...**
4. Add your IndustrialSQL Server (InSQL) to the **Server list**.

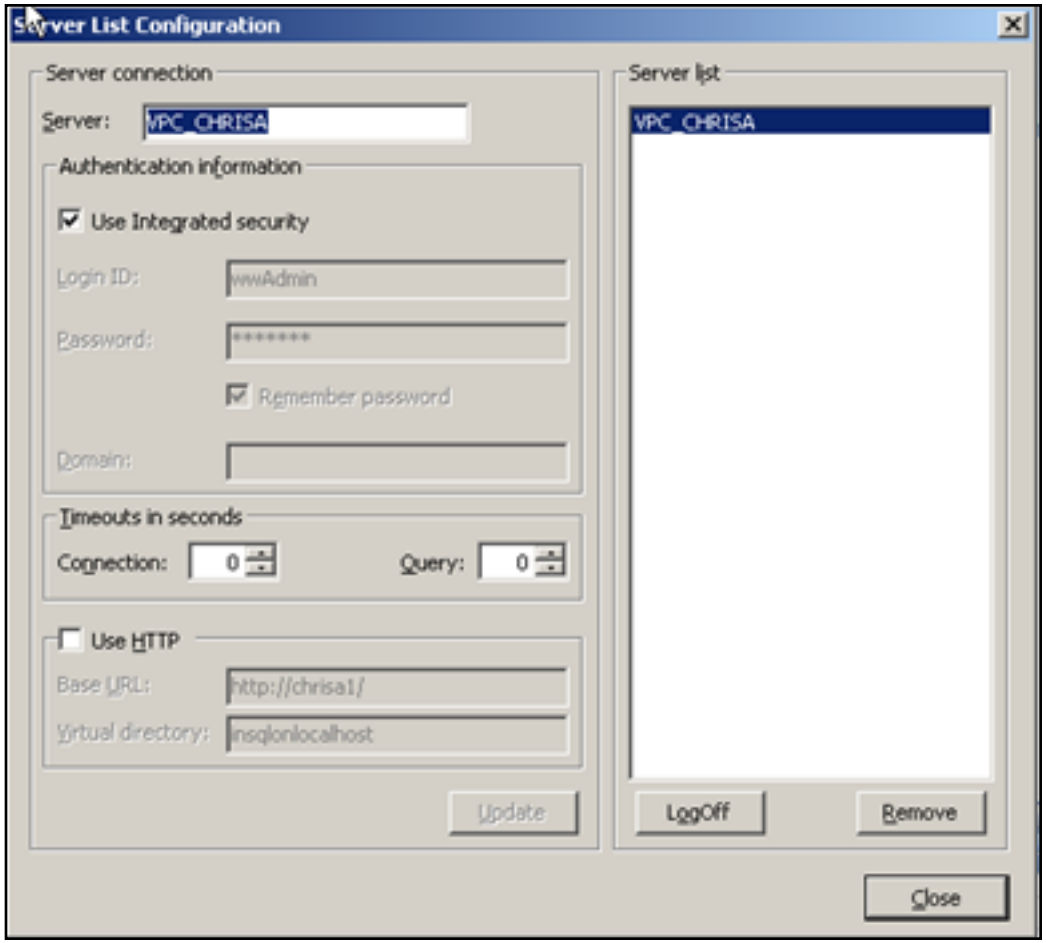

Figure 1: ActiveFactory Server List Configuration

5. As an example, create a **History Values** report by clicking **ActiveFactory / Tag Values / History Values**.

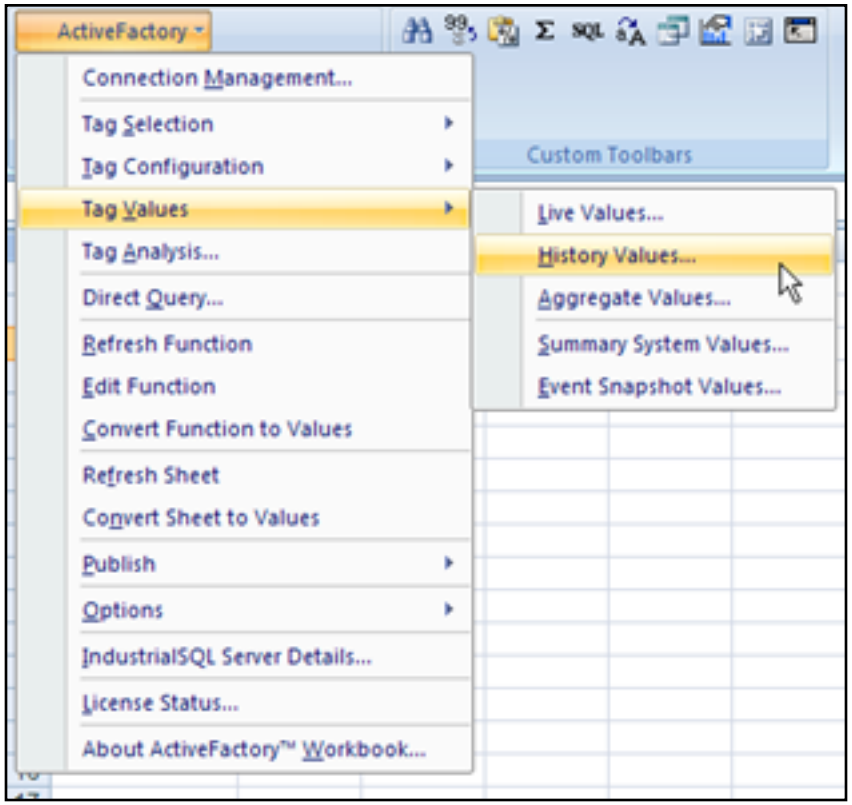

Figure 2: Create History Values Report

6. In **Step 1** of the **History Values** wizard, click the **Binding Options** button.
7. Click the **Use bounds tags in the range named 'AFTagBinding' of type:** option, then **Next**.

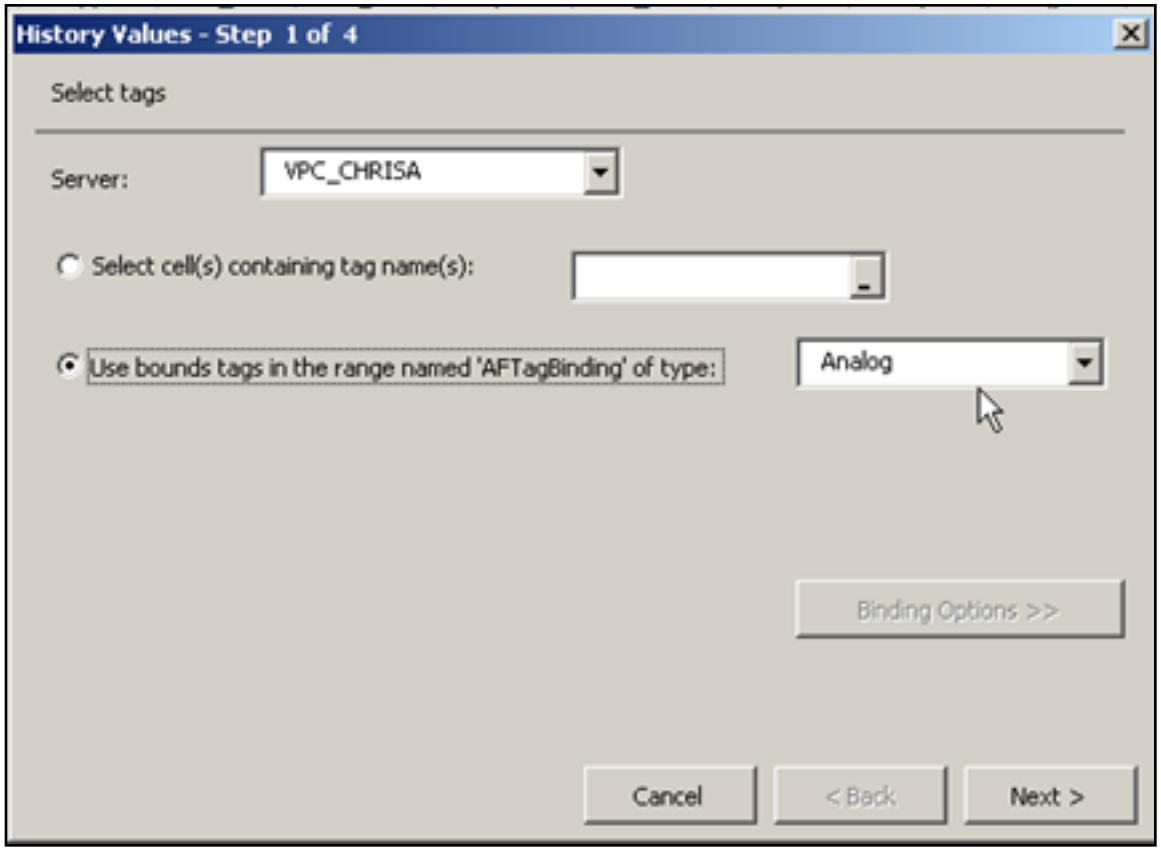

Figure 3: History Values Step 1 of 4

8. In **Step 2** of the report, select the output cell in which the tag values appear, then click **Next**.

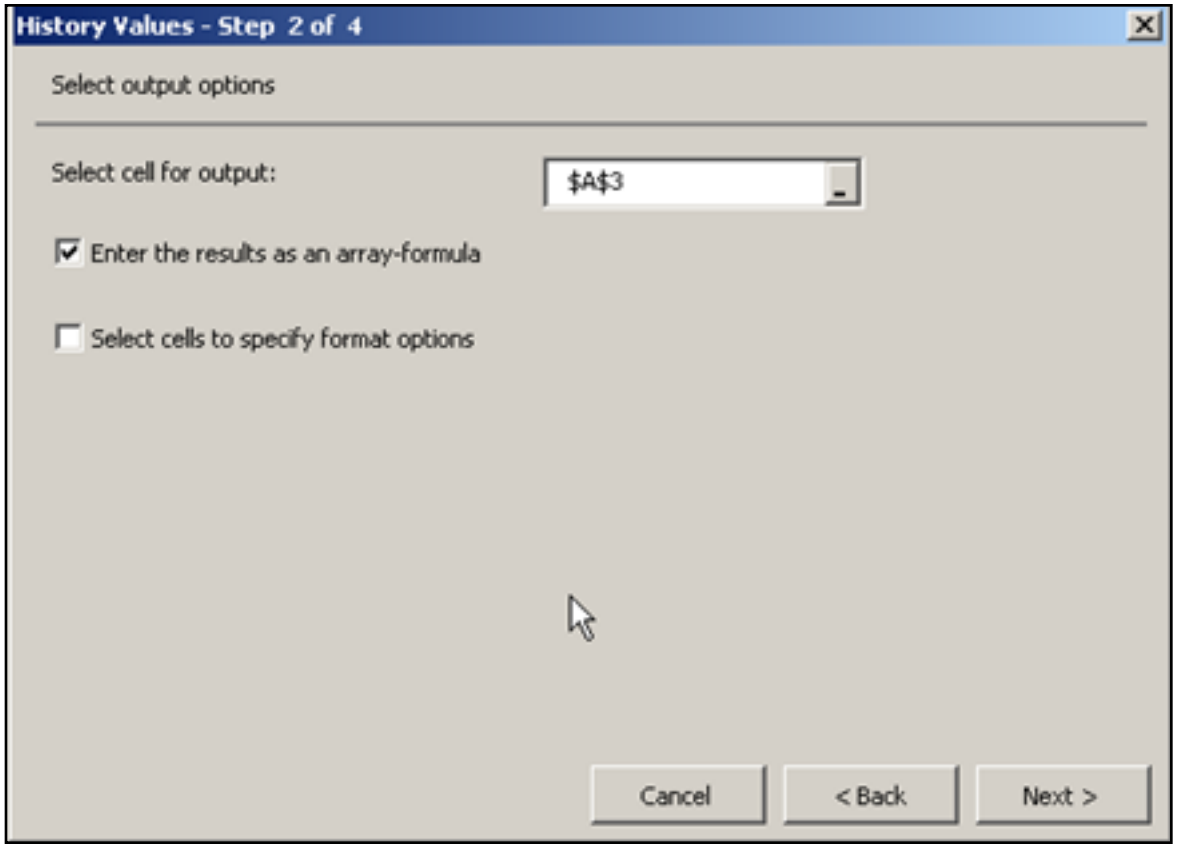

Figure 4: History Values Step 2 of 4

9. In **Step 3** of the **History Values** wizard, apply any format to the report or click **Next** to use the default settings.

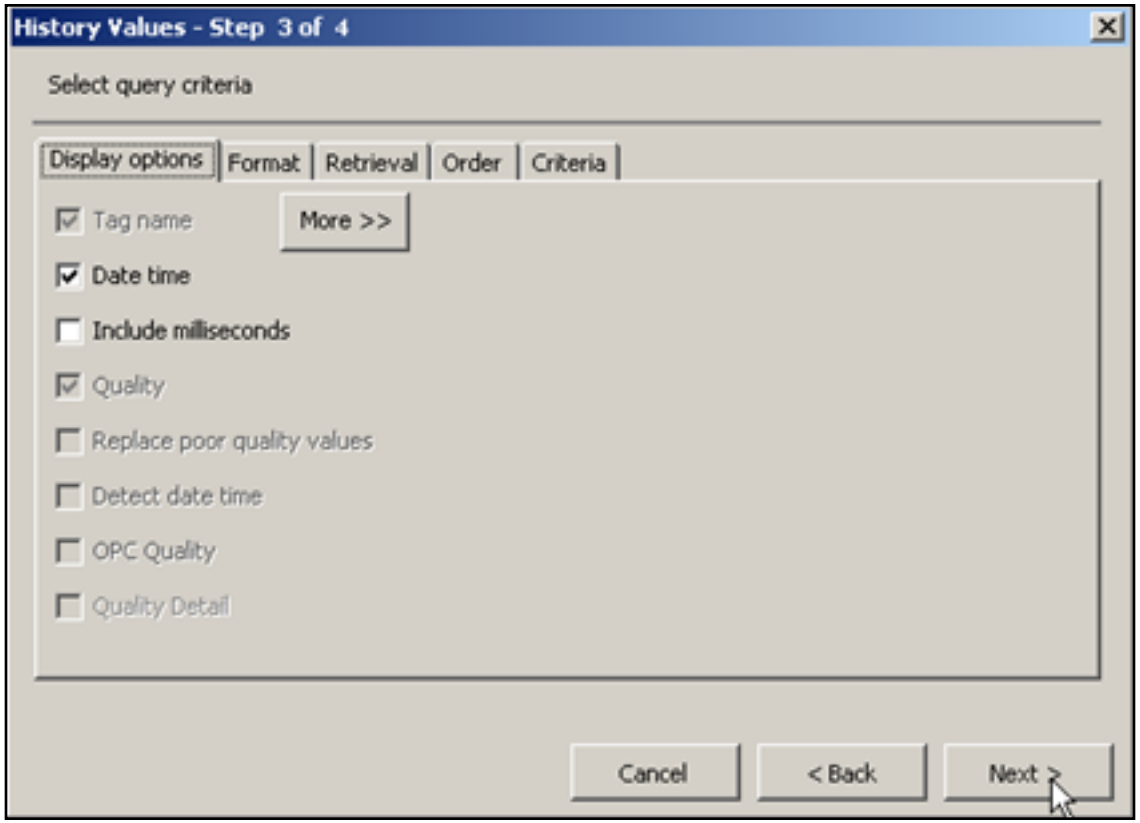

Figure 5: History Values Step 3 of 4

10. In Step 4 of the **History Values** wizard, select the **Bound times** option and click **Finish**.

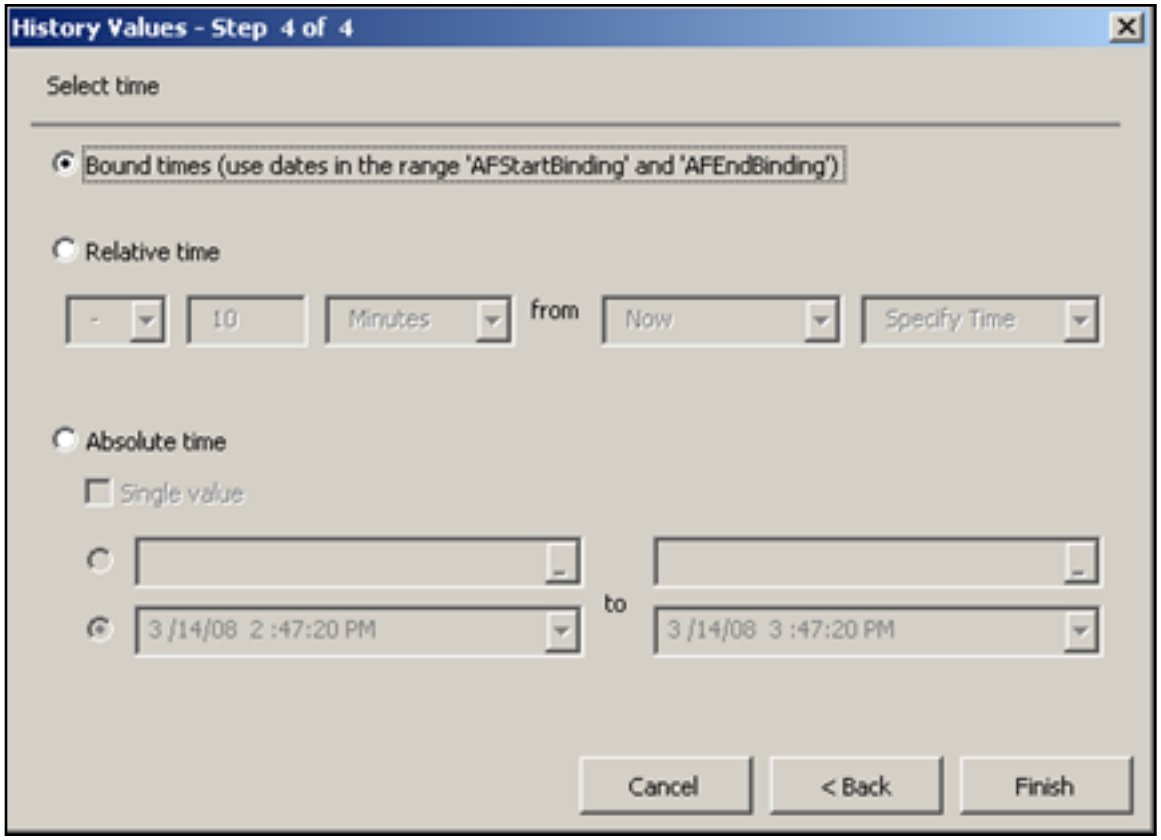

Figure 6: History Values Step 4 of 4

11. Clicking **Finish** creates the binding Function, and the **AFBindings** sheet appears. AFBinding gives you the option to customize your report's start time, end time, and tag selection via scripting.

    The following figure shows the default tag selection. Customization options for InTouch tags are included in the **commented script** below.

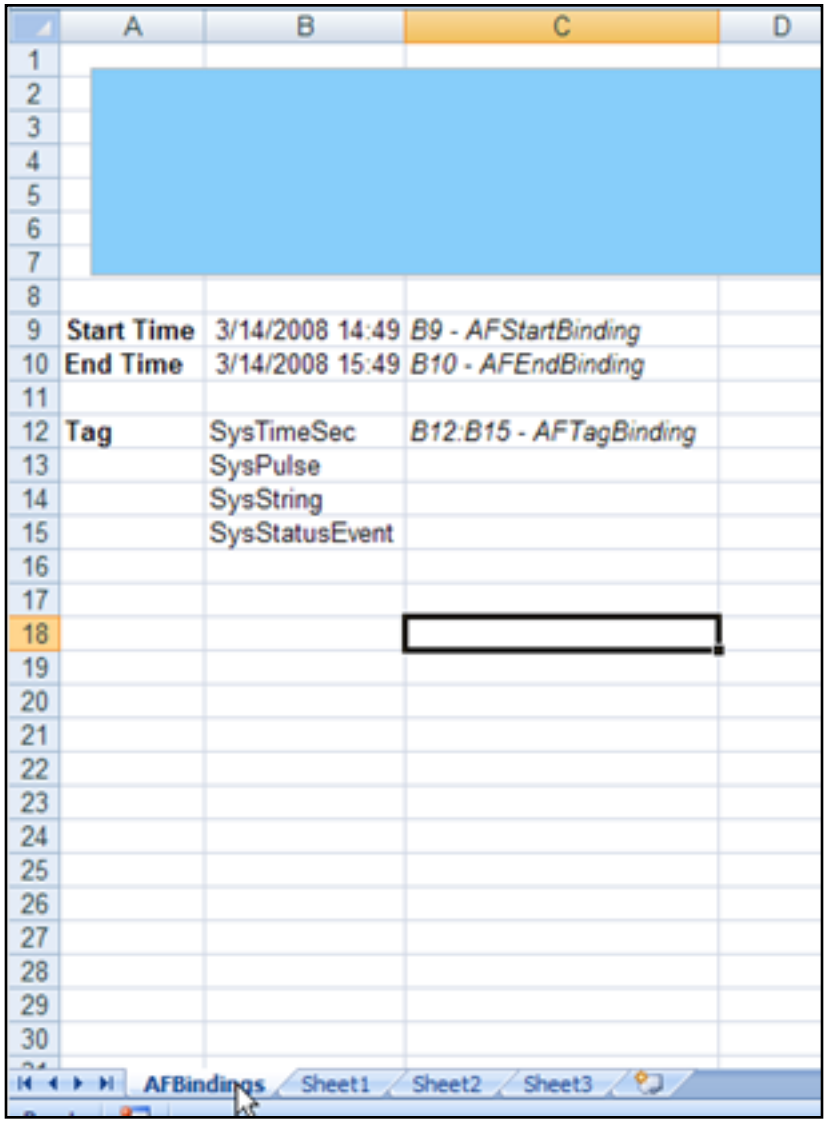

Figure 7: AFBindings Sheet

12. Save the report in the folder of your choice. This path will be used later in the script. In this example, the path is **C:\Technote\WorkbookRunner.xls**

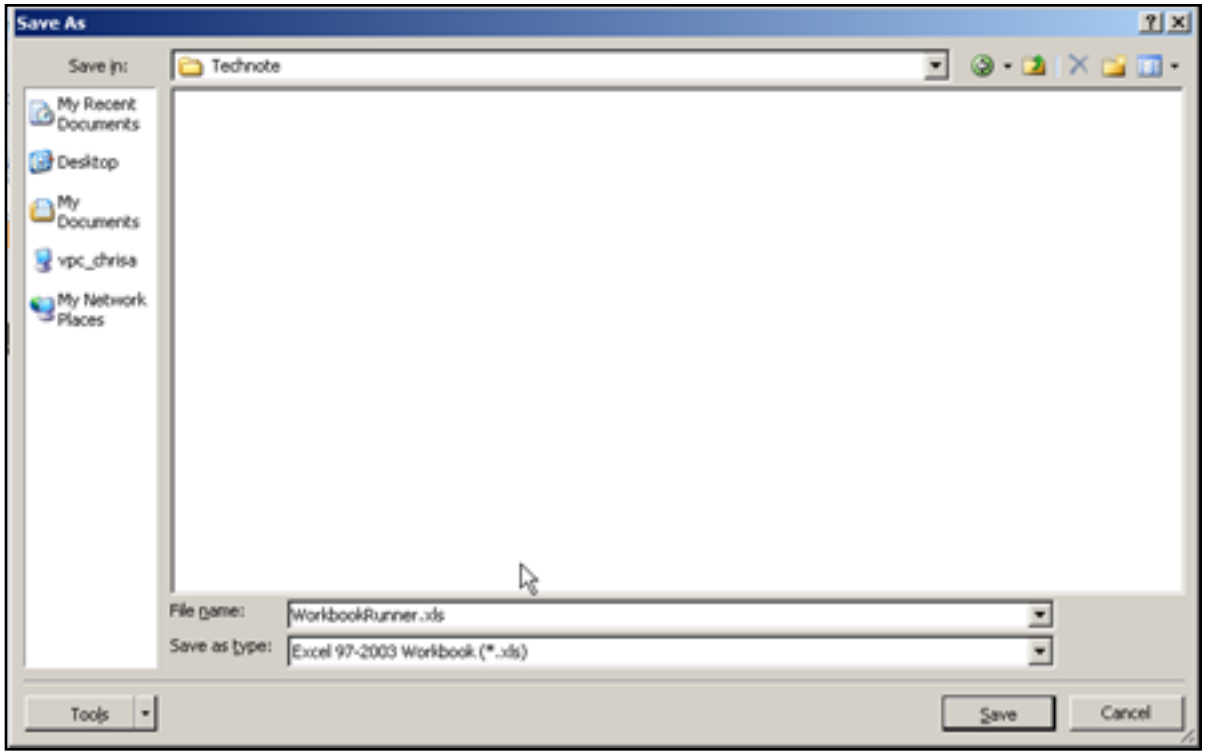

Figure 8: Save Report to a Designated Folder

Create the InTouch Workbook Runner Window

1. Within **InTouch**, create a button and type a caption. In this example the caption is **Workbook Runner**.

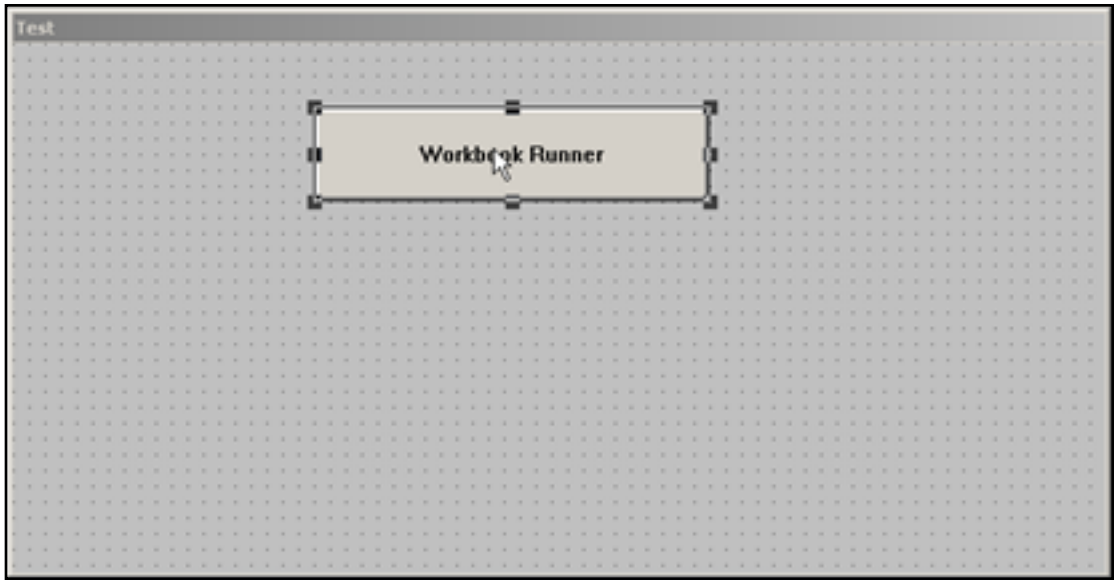

Figure 9: Configure InTouch for Workbook Runner

2. Double-click the button.
3. Click **Touch Pushbuttons / Action**

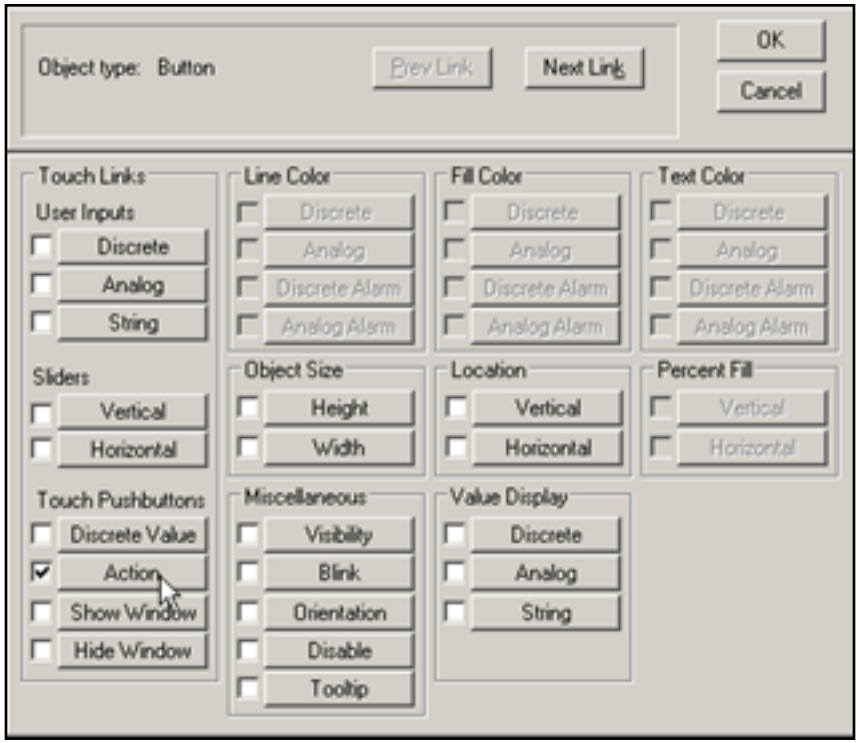

Figure 10: Configure Pushbutton Action

Script the InTouch Pushbutton Action

1. Insert the following script to run an ActiveFactory Workbook report.

   This script is the minimum required to return a basic result to InTouch and is linked in this document to customization options such as output times, tag selections, etc.:

```
OLE_CreateObject(%oRunner,"ArchestrA.HistClient.UI.aaHistClientWorkbookRunner");

InputFile = "C:\Technote\WorkbookRunner.xls";

OutputFile="";
OutputPrefix = "_";

OutputFormat = 1;
```

```
TagString="Systimesec,Systimemin,Systimehour";
NSFolderKey = 0;
NameSpace="";

DateMode=0;
StartDate = StringFromIntg( $Month, 10 ) + "/"+ StringFromIntg( $Day-1, 10 ) + "/"+
StringFromIntg( $Year, 10 ) + " 00:00:00";
EndDate= StringFromIntg( $Month, 10 ) + "/"+ StringFromIntg( $Day, 10 ) + "/"+
StringFromIntg( $Year, 10 ) + " 00:00:00";
Duration=0;

CustomFilters="";

%oRunner.ExcelVisible =1;

ResultString = %oRunner.RunReport2( InputFile, OutputFile, OutputPrefix, OutputFormat,
TagString, NSFolderKey, NameSpace, DateMode, StartDate, EndDate, Duration, CustomFilters);

OLE_ReleaseObject(%oRunner);
```

2. Skip to the **following section** to assign the tags and test the script and the output.

3. Customize the script and output using the following comments. Script **in bold** is from the above example.

```
{
Workbook Runner2 Method for ActiveFactory 9.2:

[Result=] aaHistClientWorkbookRunner.RunReport2(
message inputFile,
message outputFile,
message outputPrefix,
Integer outputFormat,
message tagString,
integer NSFolderKey,
message nameSpace,
integer dateMode,
```

```
message startDate,
message endDate,
integer duration,
message customFilters);
}

OLE_CreateObject(%oRunner,"ArchestrA.HistClient.UI.aaHistClientWorkbookRunner");

{
inputFile:
The name of the source file for the report generation, including the full path. Valid file
types are .htm, .xls, and .xlt.
}

InputFile = "C:\Technote\WorkbookRunner.xls";

{
outputFile:
The name of the output file generated, including the full path. If this parameter is set to an
empty string ( "" ), then a file name is generated automatically according to the following
formula:
OutputFile = OutputPrefix + InputFile + _ + year + month + day + hour + minute + second
}

OutputFile="";
OutputPrefix = "_";

{
outputFormat:
The file type for the output file. Valid values are:
0 = Native. That is, if the source file is an .htm file, the output file is an .htm file. If
the source file is an .xls or .xlt file, the output file is an .xls file.
1 = .htm
2 = .xls
3 = .xlt
}

OutputFormat = 1;
```

{
tagString
A comma separated list of strings to be used for the AFTagBinding named range. If the AFTagBinding range does not exist, and this parameter is set to any value other than an empty string (""), an error is raised. Valid formats are:
"<tagname1>,<tagname2>"
"'<tagname1>','<tagname2>'"
For example:
"ReactLevel,ReactTemp"
"'ReactLevel','ReactTemp'"
}

TagString="Systimesec,Systimemin,Systimehour";

NSFolderKey = 0;
NameSpace="";

{
dateMode
Determines the values used for the AFStartBinding and AFEndBinding named ranges. An error is raised if the binding ranges do not exist or if this parameter is blank. Valid values are:
0 = Use specific start and end times.
1 = Use a duration relative to the current time.
2 = Use a duration relative to the specified start time.
3 = Use a duration relative to the specified end time.
Use the startDate, endDate, and Duration parameters to specify the dates.
}

DateMode=0;

{
startDate
A date string that can be converted to a date by the Visual Basic CDate() function. A good format to use is one that reflects the standard short date and short time format on the local system.
If the dateMode parameter is set to 1 or 3, this parameter is ignored.
If the dateMode parameter is set to 0, this value indicates the specific date/time to be used for the AFStartBinding range.
If the dateMode parameter is set to 2, then "rel" is used for the AFStartBinding range and '+Duration(StartDate)' is used for the AFEndBinding range.

}

StartDate = StringFromIntg($Month, 10) + "/"+ StringFromIntg($Day-1, 10) + "/"+ StringFromIntg($Year, 10) + " 00:00:00";

{
endDate
A date string that can be converted to a date by the Visual Basic CDate() function. A good format to use is one that reflects the standard short date and short time format on the local system.
If the dateMode parameter is set to 1 or 2, this parameter is ignored.
If the dateMode parameter is set to 0, this value indicates the specific date/time to be used for the AFEndBinding range.
If the dateMode parameter is set to 3, then "rel" is used for the AFStartBinding range and '+Duration(EndDate)' is used for the AFEndBinding range.
}

EndDate= StringFromIntg($Month, 10) + "/"+ StringFromIntg($Day, 10) + "/"+ StringFromIntg ($Year, 10) + " 00:00:00";

{
Duration
The time span, in seconds, used for date/time calculations. This value cannot be a negative number.
If the dateMode parameter is set to 0, this value is ignored.
If the dateMode parameter is set to 1, "rel" is used for the AFStartBinding range and '-Duration()' is used for the AFEndBinding range.
If the dateMode parameter is set to 2, "rel" is used for the AFStartBinding range and '+Duration(StartDate)' is used for the AFEndBinding range.
If the dateMode parameter is set to 3, "rel" is used for the AFStartBinding range and '-Duration(EndDate)' is used for the AFEndBinding range.
}

Duration=0;

{
customFilters
A string of name-value pairs used to pass information from the ActiveFactory Reporting Website to the workbook file before the report is run.

The format for the string is as follows:
<name>=<value>
To pass more than one name-value pair, join them with ampersands. For example:
<name>=<value>&<name>=<value>
The parameter name that you use must correspond to an existing named range in the workbook that starts with "AFBinding."
The value you specify in the name-value pair is used for the corresponding named range in the workbook. You can specify multiple values if you separate them with commas.
For example, you workbook contains the AFBindingReportValue and AFBindingReportText named ranges. You want to pass a value of 5 for the report value and Line1 and Line2 for the ReportText. The customFilters parameter is:
ReportValue=5&ReportText=Line2,Line2
}

```
CustomFilters="";

%oRunner.ExcelVisible =1;

ResultString = %oRunner.RunReport2( InputFile, OutputFile, OutputPrefix, OutputFormat,
TagString, NSFolderKey, NameSpace, DateMode, StartDate, EndDate, Duration, CustomFilters);

OLE_ReleaseObject(%oRunner);
```

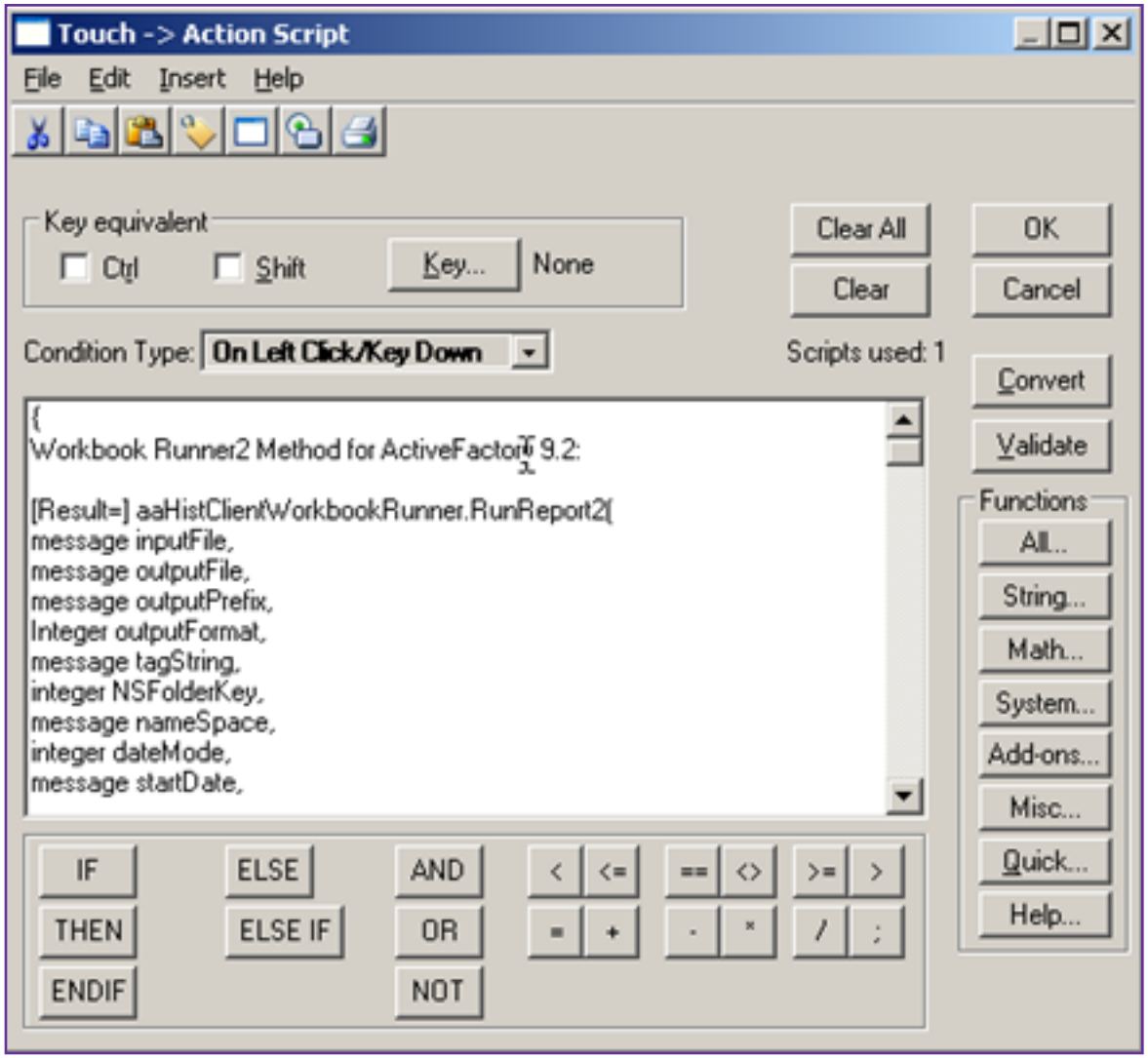

Figure 11: InTouch Script Editor

5. After pasting the code, click **Validate** to identity which tags have not been defined. The following table lists the correct data type for each tag defined in the script.

Tagname	Type
InputFile	Memory Message
OutputFile	Memory Message
OutputPrefix	Memory Message
OutputFormat	Memory Integer

TagString	Memory Message
NSFolderKey	Memory Integer
NameSpace	Memory Message
DateMode	Memory Integer
StartDate	Memory Message
EndDate	Memory Message
Duration	Memory Integer
CustomFilters	Memory Message
ResultString	MemoryMessage

6. Click **OK** twice to save changes.

7. Switch to **Runtime** to test the script.

Figure 12: Switch to Runtime

8. If necessary, customize the script by adding the comments included in the **previous example**. Test in runtime as necessary.

9. Click the **Workbook Runner** button. When the report is complete, the ResultString will contain the filename of the report. The screenshot below displays the value of some tags we defined in step 11 (**above**).

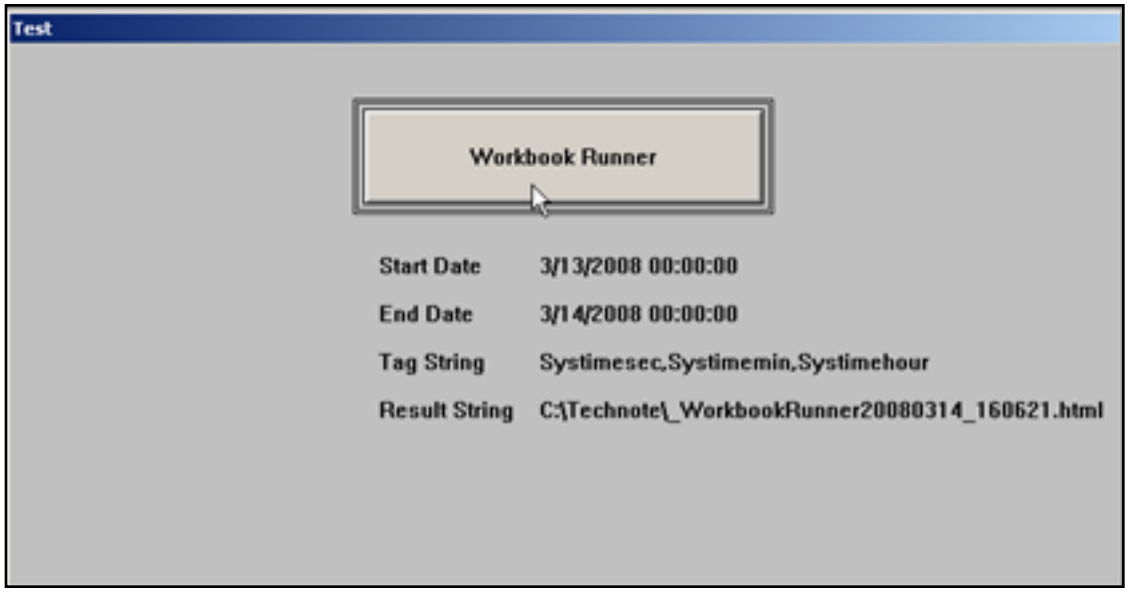

Figure 13: Runtime Workbook Runner Testing

10. Open the report with Internet Explorer to see the results.

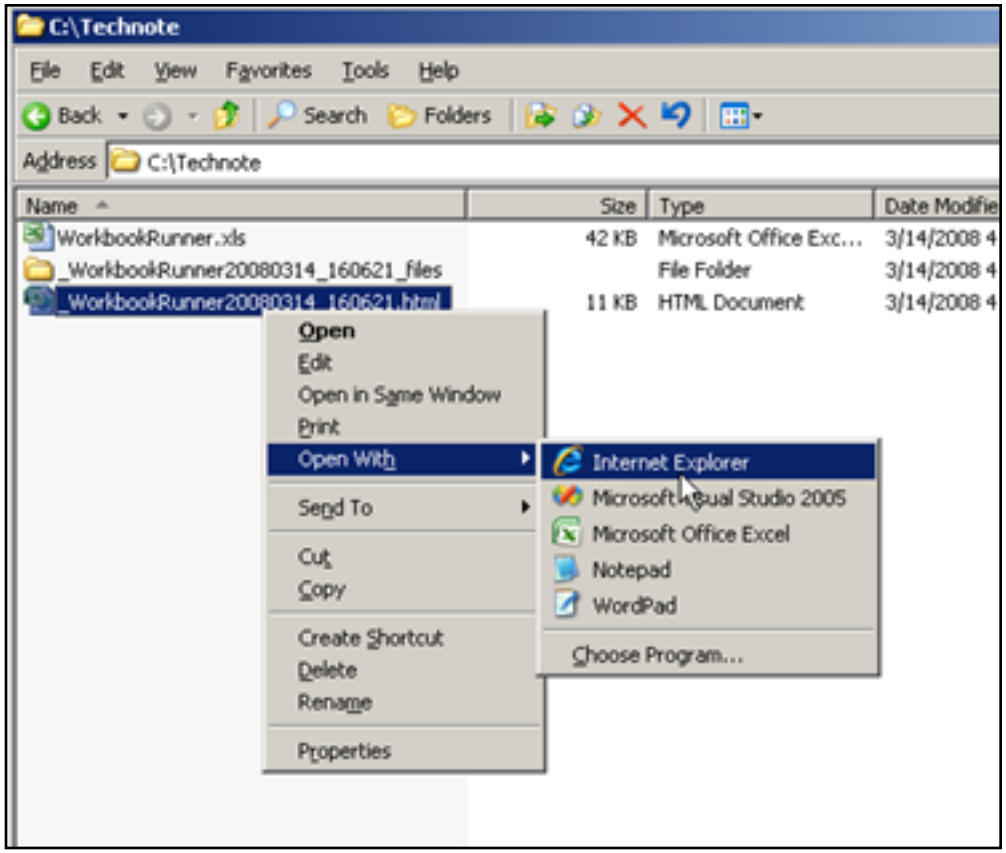

Figure 14: Open Report File Using Internet Explorer

C. Azer

Tech Notes are published occasionally by Wonderware Technical Support. Publisher: Invensys Systems, Inc., 26561 Rancho Parkway South, Lake Forest, CA 92630. There is also technical information on our software products at **www.wonderware.com/support/mmi**

For technical support questions, send an e-mail to **support@wonderware.com**.

Back to top