Creating Statistical Web Services Using ASP.NET

Neil W. Polhemus Chief Technology Officer StatPoint, Inc. 2325 Dulles Corner Blvd., Suite 500 Herndon, VA 20171 <u>neil@statpoint.com</u>

May 10, 2007

<u>Abstract</u>

ASP .NET XML Web Services provide a useful mechanism for exposing statistical calculations and graphics to web application developers. Using this approach, the capabilities of statistical software packages can be tapped by developers without invoking the software's GUI. Data and instructions are passed to the package using XML files. Output (including graphics) is returned as HTML with imbedded images, while numerical results are made available to the calling program in XML files. The same scripts that control the web services can also be accessed by the software package's user interface to reproduce the same analyses, or the GUI can be used to help create the XML scripts. This talk examines some of the issues involved in implementing such an approach, using the STATGRAPHICS statistical software package as an example.

Introduction

Some commonly used statistical packages began as a command language, with a graphical user interface added at a later date. Others were developed from the outset using only menus and dialog boxes, with no underlying language. For the latter, exposing the capabilities of the package for use by other programs presents unique challenges. This paper describes an approach based on ASP.NET Web Services, using the STATGRAPHICS statistical software package as an example.

Access by External Applications

Programs such as STATGRAPHICS which were designed to run exclusively in an interactive environment are often somewhat awkward to control from an external process. For example, suppose a real-time SPC application wished to use the program to fit a distribution and perform a capability analysis. The external application would have to:

- 1. Place the desired data in a file that the program could read.
- 2. Instruct the program to read the data and perform the desired calculations.
- 3. Retrieve the results (both numerical and graphical) and display the output.

If the SPC application was web-based, the output would have to be in a format that could be displayed in a standard HTML page.

Solutions to this problem include:

Method 1: **Interfacing with the program's GUI**. Using this method, a standard project would be created which reads a data file with a predefined structure, performs the requested analyses, and saves output in HTML format. This is the method used in current versions of STATGRAPHICS, where users may create and launch predefined StatFolios and use StatPublish to create HTML output. The disadvantage of such an approach is that the program's GUI must be launched on the client machine.

Method 2: **Exposing the program's methods for direct access by the external program.** This enables the calculations to be performed invisibly in the background. Although this approach requires substantial development effort, it allows users to integrate the routines into their applications in a seamless manner.

Exposing Statistical Procedures Using JavaBeans

A previous approach taken to expose the statistical and graphical procedures in STATGRAPHICS was the creation of a library of Java Beans called STATBEANS. STATBEANS consists of three types of beans:

- 1. **DataSource StatBeans** these beans maintain a rectangular data table which other StatBeans access to retrieve data for analysis. DataSource StatBeans are provided for reading data from local text files, for reading data over the Internet or local intranets, for accessing databases via JDBC, and for maintaining data generated by user programs.
- 2. **Calculation StatBeans** these are non-visible beans which perform statistical calculations. They may be called by user programs to calculate statistics. They are also accessed by the tabular and graphical StatBeans.
- 3. **Tabular StatBeans** these StatBeans perform statistical calculations and display them in the form of tables.
- 4. **Graphical StatBeans** these StatBeans perform statistical calculations and display them in the form of graphs. Users create applications by first adding one or more datasource StatBeans to their project, and then linking the other StatBeans to the datasource.

StatBeans can be called by user applications on a server or by applets embedded in a web page. A typical program takes the following form:

//create a datasource bean FileDataSource fileDataSource1 = new STATBEANS.FileDataSource(): //set the file name to be read fileDataSource1.setFileName("c:\\statbeans\\samples\\spc.txt"); //create a calculation bean ControlCharts controlCharts1 = new STATBEANS.ControlCharts(); //set the chart type controlCharts1.setChartType("X-bar and R"); //set the name of the column containing the data controlCharts1.setXVariableName("strength"); //set the subgroup size (if not 1) controlCharts1.setSubgroupSize(5); //set specification limits controlCharts1.setUpperSpecificationLimit(300.0); controlCharts1.setNominal(250.0); controlCharts1.setLowerSpecificationLimit(200.0); //create a table bean with 19 rows ControlChartsTable controlChartsTable1 = new STATBEANS.ControlChartsTable(); controlChartsTable1.setNumberOfRowsInDisplay(19); //set the number of decimal places for displaying the limits

controlChartsTable1.setDecimalPlaces(2);

//create 2 plot beans ControlChartsPlot controlChartsPlot1 = new STATBEANS.ControlChartsPlot(); ControlChartsPlot controlChartsPlot2 = new STATBEANS.ControlChartsPlot(); //turn off the legends controlChartsPlot1.setDisplayLimitValues(false); controlChartsPlot2.setDisplayLimitValues(false); //set the chart numbers for each plot controlChartsPlot1.setChartNumber(0); controlChartsPlot2.setChartNumber(1); //ask for warning limits on the first chart controlChartsPlot1.setShowOuterWarningLimits(true); controlChartsPlot1.setShowInnerWarningLimits(true); //add a smoother to the second chart controlChartsPlot2.setSmootherType("EWMA"); //make the calculation bean a listener for changes in the FileDataSource bean fileDataSource1.addDataChangeListener(controlCharts1.listenerForDataChange); //make the table and plot beans listeners for changes in the calculation bean controlCharts1.addDataChangeListener(controlChartsTable1.listenerForDataChange); controlCharts1.addDataChangeListener(controlChartsPlot1.listenerForDataChange); controlCharts1.addDataChangeListener(controlChartsPlot2.listenerForDataChange); //instruct the fileDataSource bean to read the file fileDataSource1.readData();

The beans can be added to an applet on a web page, or they may be instructed to save their output in image files. Typical output is shown below:



Although this approach is convenient for skilled programmers, it is not easy to implement in a client-server environment.

Exposing Statistical Procedures ASP.NET Web Services

ASP.NET is a powerful and widely used way to create web applications. It provides tools for creating the client side of web applications, and tools for developing web services that run on a server. XML is used to communicate between an application and a web service.

In developing the STATGRAPHICS .NET Web services, we first constructed an XML scripting language. A typical script is shown below:

```
<?xml version="1.0"?>
<statgraphics>
 <globals>
   <OutputLanguage>English</OutputLanguage>
   <SiqDigits>6</SiqDigits>
   <StatAdvisor>Both</StatAdvisor>
   <ConfLevel>95</ConfLevel>
   <TableRows>1000</TableRows>
   <GraphWidth>4.0</GraphWidth>
   <GraphHeight>4.0</GraphHeight>
   <FunctionResolution>101</FunctionResolution>
   <ContourResolution>51</ContourResolution>
   <htmlfileName>temp/myoutput.htm</htmlFileName>
 </globals>
 <data>
   <FileName>temp/nonlin.xml</FileName>
   <DecimalSeparator>.</DecimalSeparator>
   <DateFormat>MM/DD/YYYY</DateFormat>
   <MissingValue>NA</MissingValue>
 </data>
 <proc name="SREG">
   <input>
     <Y>Chlorine</Y>
     <X>Weeks</X>
   </input>
   <options>
     <Model>Linear</Model>
     <AltFit>None</AltFit>
   </options>
   <output>
     Summary
     LackOfFit
     <X>20</X>
       <X>30</X>
       <X>40</X>
       <ConfLevel>95</ConfLevel>
       <Limits>TwoSided</Limits>
     Comparisons
     Residuals
     Influential
     <graph name="Model">
       <Plot>All</Plot>
       <Resolution>101</Resolution>
       <PredLimits>Yes</PredLimits>
       <ConfLimits>Yes</ConfLimits>
       <ConfLevel>95</ConfLevel>
       <Limits>TwoSided</Limits>
     </graph>
     <graph>Observed</graph>
     <graph name="ResidsVsX">
       <Plot>Studentized</Plot>
     </graph>
     <graph name="ResidsVsPred">
```

```
<Plot>Studentized</Plot>
</graph>
<graph name="ResidsVsRow">
<Plot>Studentized</Plot>
</graph>
</output>
<results>
<Predicted>PREDICTED</Predicted>
<Residual>RESIDUALS</Residual>
</results>
</proc>
</statgraphics>
```

The first line indicates that the file contains XML (Extended Markup Language), which is a popular grammatical system for constructing custom markup languages. The second line contains the opening root element <statgraphics> tag, while the last line contains the closing </statgraphics> tag. Between these tags are several major blocks of code:

- 1. **Globals** Elements with a set of "global" tags set system options that apply to all procedures that follow. There may be more than one *Globals* section if options need to be changed midway through a script.
- 2. **Data** Elements within a set of "data" tags specify the source or sources of the data to be analyzed. STATGRAPHICS may access up to 10 sources of data simultaneously, which it loads into 10 virtual datasheets. More than one *Data* section is allowed. Procedures will always use the most recent data specified.
- 3. **Profile** Defines a graphics profile, which is a set of attributes that control the appearance of a graph. Once defined, a profile may be applied to any graph or used to set the default attributes for all graphs.
- 4. **Proc** The "proc" tag invokes a specific statistical procedure. Within these tags are 4 subsections:
 - a. **Input** specifies the names of the data columns or STATGRAPHICS expressions based on those columns that contain the data input for the procedure.
 - b. Options specifies the values of any Analysis Options.
 - c. **Output** specifies the tables and graphs to be created, together with their *Pane Options*.
 - d. **Results** specifies the calculated results that should be saved, together with names for each item.

A typical application utilizes the Web Services in the following manner:

Step 1: The data to be analyzed is defined. This may be done interactively on a web page as shown below:

C Ente	r Data - \	Windows In	nternet Explore	e r								×
Ge)- 🙋	http://localh	ost:1160/SGNETDe	emo/DataEntry.	aspx	• +	×	Google			2	•
🚖 🏟	🟉 Ent	er Data					🏠 •	N - 1	🖶 🔻 🔂 P	age 🕶 🌾	🐊 Tools 🤜	, »
	stat	grap	hics [®] .Net web	services	demo							^
Welcom ENTER Set Glo Select F Specify Calculat	<u>ne</u> DATA bal Optio Procedur Procedu te	ons 'e ire Inputs							Previou	JS	Next	
Enter yo press N	our data i ext to ru Datatabl	in the data t in the proce	table below an edures using au	d assign each tomatically s	n column a n elected data	neaningfu isets.	l name.	. Or leav	e the datas	sheet er	mpty and	1
Row	weeks	chlorine]	Chiptenet								
1	8	0.49										
2	8	0.49										
3	10	0.48										
4	10	0.47										
5	10	0.48										
6	10	0.47										
7	12	0.46										
R <	12	0.46] 3	>

A query could instead be issued against a database on a server. The resulting data must be placed in an XML file for access by the web services.

Step 2: Global options are set.

🖉 System Opt	ions - Windows Internet I	Explorer								
00-	http://localhost:1160/SGNET	Demo/DataEntry.aspx 🛛 🔽	😽 🗙 Google	P -						
🚖 🚓 🌈s	iystem Options		🗄 • 🔊 - 🖶 • 🛙	🍦 Page 👻 🍈 Tools 👻 🂙						
■ statoranhics										
	.INET We									
<u>Welcome</u> Enter Data										
SET GLOBAL	OPTIONS		Pre	vious Next						
Select Proced	<u>ure</u> Jure Inputs									
<u>Calculate</u>										
The following o	options apply to all statistic	cal procedures:								
The role wing c	spaons appry to an statistic	in procedures.								
System Optio	ns			-						
Significant	Rows to display in tables	: Output language:	Date separator							
	1000	● English	⊙ slash (/)							
01	100	• O French	O period (.)							
05	Graph width:	O German	O dash (-)							
06	600 pixels	OSpanish	Date format							
07	Graph height:	StatAdvisor:	Om d yy							
08	400 pixels	O Tables	⊙mdyyyy							
09	Function resolution:	O Graphs	Odmyy							
0 10	Contour plot resolution:	O Both	Odmyyyy							
	101	○ None	⊖yymd							
Confidence	Graph background color		O yyyy m d							
	White 👻	Missing value indicator:	Omm dd yy	~						
<	C		· · ·	<u>></u>						

Global options control things such as the number of significant digits in the output, the size and resolution of the graphics, and the output language. These settings are passed to the web services in the GLOBALS section of the XML script.

Step 3: The statistical procedure is selected.

C Untitled Page - Windows Internet Exp	lorer		X
💽 🗸 🖉 http://localhost:1160/SGNETD	emo/SystemOptions.aspx 🛛 🖌 Google	P	•
😭 🍄 🌈 Untitled Page		🔹 🖶 👻 🔂 Page 👻 🍥 Tools 👻	»
statgraphics	services demo		~
<u>Welcome</u> Enter Data Set Global Options SELECT PROCEDURE Specify Procedure Inputs Calculate		Previous Next	
Select any of the following STATGRAPH	ICS procedures:		
○Data - Display Data Table	○ Compare - Two Sample Comparison		
○ Plot - X Plot	○ Compare - Paired Sample Comparison		
○Plot - X-Y Plot	O Compare - Multiple Sample Comparison		
○Plot - X-Y-Z Plot	○ Compare - Oneway ANOVA		
○Plot - Matrix Plot	○ Compare - Multifactor ANOVA		
○ Plot - Multiple X-Y Plot	 Relate - Simple Regression 		
○ Plot - Multiple X-Y-Z Plot	○ Relate - Multiple Regression		
○ Plot - Frequency Table	○ SPC - X Chart		
○ Plot - Contingency Table	○ SPC - X-bar or Median Chart		
○Describe - One-Variable Analysis	○ SPC - P or NP Chart		
ODescribe - Multiple Variable Analysis	○ SPC - U Chart		
ODescribe - Frequency Tabulation	○ SPC - C Chart		
ODescribe - Crosstabulation	○ SPC - Capability Analysis		
<		>	~

This selection is passed to the web services using a PROC statement in the XML script.

Step 4: Procedure inputs are specified.

🖉 Simple Regression - Windows Inter	net Explorer				×	
🚱 🗸 🖉 http://localhost:1160/SGN	ETDemo/SelectProcedure.aspx	✓ +→ ×	Google	٩	•	
🚖 🕸 🌈 Simple Regression		6	• 📾 • 🖶 •	🔂 Page 👻 🎯 Tools 👻	»	
statgraphics) veb services demo				 Image: A state of the state of	
<u>Welcome</u> Enter Data Set Global Options Select Procedure SPECIFY PROCEDURE INPUTS Calculate			Pi	revious Next		
Simple Regression Procedure	r model seleting a depender	t variable V	to an independent	variable		
X. <u>Documentation</u>	inodel relating a dependen		to an independent	Variable		
Dependent variable (Y): chlorine	3	Edit				
Independent variable (X): weeks		✓ Edit				
(Select)	Help					
Model:						
◯Linear	○ Squared-Y Square Ro	ot-X	🔿 Double Recij	procal		
○ Square-Root Y	O Logarithmic-X		⊙ Squared-Y F	Reciprocal-X		
○Exponential	○ Square Root-Y Logar	ithmic-X	○ Squared-X			
○Reciprocal-Y	○ Multiplicative		○ Square Root	Root-Y Squared-X		
○ Squared-Y	O Reciprocal-Y Log-X		○Log-Y Squa	red-X	~	
<						

The input includes the names of the fields containing the data to be analyzed, names of output fields to be saved, and procedure specific options. These settings are passed to the web services in the corresponding PROC section of the XML script.

Step 5: Calculations are performed and the output returned.

The web services then execute the script and return the output as an HTML page. Numerical output is returned in standard HTML tables so that it may be easily copied to applications such as Microsoft Excel:

6 STATGRA	PHICS Output -	Windows Inte	rnet Explo	rer										X
00-	🙋 http://localhos	t:1160/SGNETDe	mo/SimpleRe	gression.asp	x 🔽	4 9	×	Google					٩	•
🚖 🏟 🍾	STATGRAPHICS O	utput				1	<u>م</u>	5	-	r 🔹 🔂 Pa	age ▼	🚫 Tool:	5 -	»
														~
sta	atoran	hics												
	rigiup	.NET web	service	s demo										
Welcome							_		_		_		_	
Enter Data											_		_	
Set Global C Select Proce	<u>ptions</u> dure									Previous		Finish		
Specify Proc	<u>edure Inputs</u> :													
														_
STATGRA	PHICS Web Se 127/2007 9-55-3	ervices 5 AM												
Generated /	12112001 9.55.5	5 AM												
Data file: n	onlin.xml													
Simple Reg	ression - chlori	ine vs. week	S											
Dependent v	ariable: chlorine		-											
Independent Squared-V r	variable: weeks eciprocal-X mo	del·V = sart(:	a + h/X											
Squarce-11	cciproca-74 mo	uci i – sqri	a + 0/2x)											
Coefficients	5		_											
-	Least Squares	Standard	<i>T</i>											
Parameter	Estimate	Error	Statistic	P-Value										
Slope	0.151785	0.00319479	41.2495	0.0000										
Slope	0.095725	0.0010424	17.5440	0.0000										
Analysis of														
Source	Sum of Squar	es Df Mean	Square I	F-Ratio P	-Value									~
<				Ш									>	

Graphs are returned as imbedded images so that they can be imbedded in other HTML pages if desired:



Any calculated results are returned as XML:

🤌 http://localhost:1160/sgnetdemo/temp/p1lwvbuxsznqsl55kzzfi245_\$sg\$_results.xml - Windows Int 🗐	
🕞 🕞 👻 http://localhost:1160/sgnetdemo/temp/p1lwvbuxsznqsl55kzzfiź 🕶 🍫 🗙 Google	<mark>ہ</mark> ج
🚖 🛠 🌈 http://localhost:1160/sgnetdemo/temp/p1lwvbuxsznq 👘 🔹 🗟 🔹 🖶 🗣 🔂 Page 🔹 🎯 Tool	; • »
xml version="1.0" standalone="yes" ?	^
- <dataset xmlns="NetFrameWork"></dataset>	=
-	
<row>1</row>	
<predicted>0.493709</predicted>	
<lowerplims>0.513946</lowerplims>	
<upperplims>0.472606</upperplims>	
<residuals>-0.00370928</residuals>	
<sresiduals>-0.424636</sresiduals>	
<leverages>0.170244</leverages>	
-	
<row>2</row>	
<predicted>0.493709</predicted>	
<lowerplims>0.513946</lowerplims>	
<upperplims>0.472606</upperplims>	
<residuals>-0.00370928</residuals>	
<sresiduals>-0.424636</sresiduals>	
<leverages>0.170244</leverages>	
-	
<row>3</row>	
<predicted>0.470485</predicted>	
<lowerplims>0.490892</lowerplims>	
<pre><upperplims>0.449151</upperplims></pre>	
<residuals>0.00951544</residuals>	
<sresiduals>I.0115</sresiduals>	
<leverages>0.0831/51</leverages>	
-	
	_
<upperplim5>U.449131</upperplim5>	×

Creating XML Scripts

To ease the development of XML scripts, the program's GUI can be modified to interpret and create such scripts. In STATGRAPHICS, we have added a "Show XML" option to various windows and dialog boxes, which displays the XML script required to create the output in that window. The script can be copied and pasted into other documents. We have also added two lines to the *File* menu:

Execute XML Script: read san XML script and creates the requested output within the GUI.

Create XML Script: saves an XML script that creates the contents of the current StatFolio when submitted to the Web Services.

Conclusion

ASP.NET Web Services provide a convenient mechanism for exposing statistical calculations to external web applications. Simple XML scripts can be created and posted to the server, which will return the output in a web-friendly format. The approach makes it possible for web application developers to access the statistical calculations of a program while bypassing its normal GUI. If desired, the GUI can be modified to interpret those same XML scripts, so that users can reproduce the same analyses within the menu-driven application.