ABSTRACT

(This paper is based on a presentation given in March 1998 at the SAS Executive Conference in Cary, NC)

Using SAS/Intrnet™ tools, members of the INFOMAN® development team at Bell Atlantic have developed a SAS reporting application, accessible to our clients from the web. Now data center operations managers at Bell Atlantic can access real-time mission critical data from their desktop, and instantly view static or dynamically driven reports that display graphics, spreadsheet style summary reports, or both, to provide a visually powerful effect. This paper focuses on the development efforts of a sampling of the reports and details the major SAS elements and programming techniques used to create the reports for the web environment.

INTRODUCTION

With the release of the SAS/Intrnet™ tools in 1997 and other SAS software enhancements, new solutions became available, answering the question how to provide business managers rapid access to the data they need most to manage their day to day activities and analyze the internal problems within their organizations. Graphics, business reports, and even visually enhanced drill down methods can be produced automatically via the web in the time it takes to point and click on a few buttons. You can even click on a hyperlink and download your selections to an Excel spreadsheet if you’d like for further analysis. Today the possibilities to produce visually interesting graphs and other customized reports via the web with SAS software are a ‘can do’ operation with a win-win message for SAS programmers and the clients they support.

PROJECT BACKGROUND

As a software engineer with a background in application design and development, I was offered an opportunity to join the INFOMAN® development team at Bell Atlantic. One objective of the team was to provide detailed business reports to Bell Atlantic executives. The data resided on a mainframe platform with no easy way for managers to access the data quickly. In addition, many managers would need to make specific requests to create the types of reports they wanted to help them assess business results on a day to day level.

In some cases, a staff person would spend hours every day manually creating various Excel books containing different reports for distribution to a wide group of managers. Creating the reports in this manner was very time consuming and tedious. There was also a margin of error due to the manual intensity involved.

In an increasingly competitive marketplace, Bell Atlantic managers wanted new reports developed. They wanted quick access to their data and the flexibility to develop different groupings and summaries. They also asked to see data represented graphically in either bar or pie charts in the reports.

The SAS/Intrnet™ tools provided the answer for delivering a new mechanized reporting application. The new process would eliminate the need for staff to spend hours every day manually creating and distributing reports. Using SAS/Intrnet™ software, combined with SAS publishing tools for the web, HTML web pages, SAS macro logic, SAS/GRAPH software and other SAS reporting procedures, as well as basic SAS software, the development of a SAS reporting application from the web was established. It would replace previous attempts at report generation and distribution and was a welcome change by Bell Atlantic managers.

This paper focuses on some of the programs that have been developed, the key elements behind their development, and the methodology using the SAS/Intrnet™ tools. Each program offers examples of resolving different programming needs and outlines the uses and capabilities you can gain with the SAS/Intrnet™ tools. Our data represents a Problem and Change Data file.

HOW DOES THE SAS/INTRNET™ SOFTWARE OPERATE?

With the SAS/Intrnet™ software installed, you have the ability to run SAS programs directly from your personal web browser and have the output displayed back to your browser. The user begins by making selection criteria choices from a web page or HTML form, which has been developed as part of the web application. The web page or HTML form acts as the starting point to invoke the Application Dispatcher. Either by clicking on a submit
button or on a hyperlink, the user submits their request, and the information is passed to an Application Broker component. The Application Broker software accepts the user’s request and passes it to the Application Server, which executes the named Dispatcher Application program included in the software instructions. Results are routed back through these components to your browser. The whole process works so quickly, that in literally seconds, your results can be displayed back to your browser for your review and analysis.

For the SAS programmer, there is virtually no maintenance effort needed for these components beyond installing them. The work effort is mainly behind creating the HTML forms or web pages and the SAS programs they interact with.

As for the users, they may not even know that they’re executing SAS at all. Having a web application gives the users a point and click environment that needs little technical expertise or training to use. Users also benefit cost wise from a thin web client platform without the added expense of extra software licenses. In addition, we’ve found that the SAS/Intrnet™ technology works effortlessly and seamlessly to the eye of the end user.

The following programs were developed specifically for execution from a web browser. They are part of the INFOMAN® Intranet reporting application created for Bell Atlantic managers.

Sample Program A. The Application Incident Report

Key features of this program include:

- Point and Click execution from a hypertext link.
- Techniques for highlighting text on key observations.
- Drill down effect.
- Macro processing functionality.
- Download to Microsoft Excel option.

Executing a SAS program from a hyperlink is easily accomplished with the SAS Application Dispatcher software. The hyperlink, coded in your web page, appears as underlined text to the user. By clicking on the hyperlink for the desired report, the user invokes the web reporting process, and the Application Incident Report is dynamically created. This is an easy method for end users to obtain their reports.

Figure 1 illustrates part of a web page containing hyperlinks for different reports, including one for the Application Incident Report.

The hyperlink for the Application Incident Report once pointed to the original Excel version of the report. When the report was rewritten in SAS, there was no need to change anything about the web page from the client’s perspective. We were able to provide a smooth transition to the new process, just by having the hyperlink interact instead with our SAS program using the SAS Application Dispatcher software.

The Excel version of the Application Incident report that we were asked to replace with SAS code had certain rows highlighted in red. If an application was in a ‘Top 20’ ranking, the person creating the report in Excel would manually change the color of the row as needed. The report was at a detail level, and our first direction was to code the report using SAS PROC REPORT with SAS web publishing tools. But we wanted to change the color of the text in an individual row of data in the report, if the value in a field met certain criteria.

To resolve this problem, we coded a SAS DATA step instead, using If-Then logic on the particular data field to test if highlighting was needed. If highlighting were needed, HTML code would be executed that specified the color red for the text. Otherwise the report rows would be printed in plain text by executing a different block of HTML without the color specified. Embedding HTML code in a SAS DATA step is feasible with the SAS/Intrnet™ software and provides an added advantage for web enabled applications.

Here is a section of SAS code illustrating this technique.

```sas
Data _null_;
set work.inappl;
file _webout;
Put "Content-type: Text/html’;
put ;
put ‘<html’;
```
The following ‘if statement’ checks the value of the field ‘catasort’ to determine if red highlighting will be applied.

    if catasort = '1' then do;
    put '<td><strong><font face="Arial" size=-4 color="#FF0000">comk'</strong></td>;
    put '<td><strong><font face="Arial" size=-4 color="#FF0000">region'</td>;
    put '<td no wrap><strong><font face="Arial" size=-4 color="#FF0000"><a href="http://your.system.com/xxxweb/PRO BVIEW.REXX?RNID=' RNID1'">rnid'</a></td>;
    put '<td align="center"><strong><font face="Arial" size=-4 color="#FF0000">avoid'</td>;
    put '<td nowrap><strong><font face="Arial" size=-4 color="#FF0000">dato'</td>;
    put '<td><strong><font face="Arial" size=-4 color="#FF0000">dayofwk'</td>;
    put '<td nowrap><strong><font face="Arial" size=-4 color="#FF0000">timo'</timeampm9.></td>;
    put '<td nowrap><strong><font face="Arial" size=-4 color="#FF0000">timf'</td>;
    put '<td><strong><font face="Arial" size=-4 color="#FF0000">sysdm Mins</td>;
    put '<td nowrap><strong><font face="Arial" size=-4 color="#FF0000"> typedeg'</td>;
    put '<td align=center><strong><font face="Arial" size=-4 color="#FF0000">audt'</td>;
    put '<td><strong><font face="Arial" size=-4 color="#FF0000">cause'</td>;
    end;

    else do;
    put '<td><strong><font face="Arial" size=-4 color="#FF0000">comk'</strong></td>;
    put '<td><font face="Arial" size=-4 color="#FF0000">region'</td>;
    put '<td><a href="http://your.system.com/xxxweb/PRO BVIEW.REXX?RNID=' RNID1'">rnid'</a></td>;
    put '<td><font face="Arial" size=-4 color="#FF0000">avoid'</td>;
    put '<td><font face="Arial" size=-4 color="#FF0000">dato'</td>;
    put '<td><font face="Arial" size=-4 color="#FF0000">dayofwk'</td>;
    put '<td><font face="Arial" size=-4 color="#FF0000">timo'</timeampm9.></td>;
    put '<td><font face="Arial" size=-4 color="#FF0000">timf'</td>;
    put '<td><font face="Arial" size=-4 color="#FF0000">sysdm Mins</td>;
    put '<td><font face="Arial" size=-4 color="#FF0000"> typedeg'</td>;
    put '<td align=center><font face="Arial" size=-4 color="#FF0000">audt'</td>;
    put '<td><font face="Arial" size=-4 color="#FF0000">cause'</td>;
    end;

This report also has the ability to drill down through the data to individual rows or tickets. When the results of the report are displayed back to your browser, the column labeled TKT # demonstrates an embedded hyperlink.

Figure 2 displays results from the Application Incident Report.

Figure 2

The user can click on any ticket number and interact with the mainframe host database, through a REXX API used by the host application. The REXX API retrieves and displays more details about the ticket number, enabling the user prompt access to additional detailed information about that individual observation.

Another feature of this program was to use macro processing statements to control which month of data would be extracted for the report from the input data set. The user clicks on the hyperlink from the report web page, either under the Current month or Previous month heading. That’s all they need to do to make their choice.

The hyperlink contains a parameter named REPORT, which is equal to either CURMONTH or PREMONTH, depending on which hyperlink is selected. When the user clicks on their report selection, invoking the Application Dispatcher, the value of the REPORT parameter is passed through the SAS/Intrnet™ components to the SAS program as a global macro variable. Macro logic in the SAS program tests the value of REPORT and controls which lines of SAS code are executed to extract the appropriate month of data for the user’s request.
Macro processing enables the SAS developer to code one program to handle multiple options. This technique adds immense flexibility to the SAS programs and saves time on future maintenance requirements.

We also provided the users with a SAS software tool to export SAS data sets, either permanent or temporary, to Microsoft Excel. The SAS program to do this was designed specifically for web applications and needed only a minor adjustment to meet our needs. Our area SAS Account Manager helped us locate this particular SAS tool, after attending a demonstration with our team where the users had mentioned they would like this ability.

The download to Excel option is easily requested from a hyperlink at the bottom of the results from the initial SAS request. The user saves the output from the download process as a .csv file and can then open the file in Excel. This gives the users even more flexibility to examine the data and create specialized reports they tailor to meet their own personal needs.

The next program dynamically produces both graphical output and a summarized data report in one web report.

**Sample Program B. Problem Management Dynamic Graphs**

Key features of this program include:

- Point and Click execution from an HTML form.
- Macro processing tips for enhancing program output.
- Creating graphs and reports together in one web page.
- Using SAS web publishing tools.

You can create an HTML form to allow your clients the flexibility to create unique reports based on their own selection criteria. The HTML form can be designed with radio buttons, drop down lists, data entry fields, and more. These options provide different methods for collecting data choices from your user.

Figure 3 represents a portion of the Problem Management Dynamic Graphs HTML web page. It uses radio buttons to collect the user’s input request.

In this report, the user requests a time frame, a priority, either a bar or a pie chart, and then the locations they want the criteria applied to.

Once the user has made their selections, they click on a submit button provided at the end of the form, and the choices they made are passed as parameters through the SAS/Intrnet™ components. The parameters are then processed as global macro variables by the named SAS program.

This report, like the others that have been developed, evolves around creating output based on specific data criteria entered by the user. Most times there are observations that meet the user’s request, but at times, you may find that there are no observations to construct a report from, and that can cause additional processing needs.

You can create and use a macro variable to check whether there are any observations that met the user’s input requirements, and avoid unwanted results, or error messages, from your SAS processing. Testing for observations will add value to and enhance your reporting process. With a simple SAS DATA step, you can issue a user-friendly message to your clients, if their request produces no observations from the input data set.

Figure 4 is an example of providing an informative message when there is no data to construct a report from.
The Dynamic PCMS Graphs Report produces two outputs combined together into one web page. The graphical output is displayed first, followed by the report output, the result of a SAS PROC TABULATE. By combining graphical and report output in one page, you can achieve a highly visual and detailed presentation of your data. To accomplish this presentation style, we used the following methodology.

First, generate the Content-type header inside a SAS DATA step. The first and second PUT statements are required. Then use a third PUT statement, to code the HTML to point to your hyperlink to the SAS/GRAPH program. The third PUT statement processes the separate graph program and displays the results back to your browser.

```sas
%else %if &graph='B' and &num ne 0
  %then %do;
  data _null_;
  file _webout;
  put 'Content-type: text/html';
  put;  
  put '<center><img src="http://your.system.com/cgi-bin/broker.exe?
  _service=newsas&
  _program=programs.latest-
  bar.sas&locat=' '&locat"'></center>'';
  run;
  %end;
```

Now you are ready to add the SAS code for the PROC TABULATE to produce the second piece of the report.

```sas
%else %if &graph='B' and &num ne 0
  %then %do;
  data _null_;
  file _webout;
  put 'Content-type: text/html';
  put;  
  put '<center><img src="http://your.system.com/cgi-bin/broker.exe?
  _service=newsas&
  _program=programs.latest-
  bar.sas&locat=' '&locat"'></center>'';
  run;
```

To display the results of the PROC TABULATE back to the browser, we used the SAS web publishing macro %TAB2HTM. The SAS web publishing tools enable SAS output to be displayed back to your browser and provide options to help format the data.

This statement, `%tab2htm(capture=on), begins the HTML Tabulate Formatter. The statement, `%tab2htm(capture=off), ends the HTML Tabulate Formatter. Many arguments are available to designate formatting characteristics. For example, we used rlsize=-1 to reduce the size of the row label text by 1 and clsize=-1 to reduce the size of the caption text by 1 to improve the overall appearance of the report as it appears in the browser.
STANDARDIZING REPORT APPEARANCES

As more and more reports were being developed for the web platform, we began to notice differences among the reports, which, in terms of presentation, had an uncomplimentary effect. For example, one report had a submit button that read, “click here”. Another submit button read “create report”. Other small differences relating to the verbiage on the web page or the font that was used were beginning to take on an unstructured disordered appearance. We decided to set some guidelines for the growing web application to establish a uniform appearance that would be unique to our web application.

For a well-organized appearance, we set standards for wording and font sizes on the web pages. For example, all submit buttons now read “Create Dynamic Report”. We paid attention to verbiage and had it agree across similar web pages. These small changes helped bring about a uniform appearance that added quality and a more professional look.

A message, “Produced by AD HOC Reporting System” was tagged onto the SAS reports to distinguish the results produced by the new SAS process from any of the non SAS produced reports. This tag line made it easy for our clients to know the source of any reports they printed.

For a small cost, we invested in Microsoft FrontPage 98, a web site creation and management tool. We used the software to give our web application a brand new look that was interesting, fun, and professional.

We made on-line documentation for the reports as another standard enhancement. The documentation HTML document can be accessed from a hyperlink at the bottom of each HTML form or web page. By clicking on the link, specific details of the criteria that defined the report are displayed to the user, along with a support contact list to reach for questions about the report.

All combined, these changes added up to a well-organized, professional look that complimented our work efforts for creating the graphs and reports while developing the web application for our clients.

CONCLUSION

In today’s global market, the growth of the World Wide Web and Internet/Intranet access has created a new mode of communication for corporations and their executives. With the development of SAS/Intrnet™ software, complex business information that once took hours or longer to prepare into useful reports can now be retrieved in seconds.

A key component of any successful business is the ability to make information readily available to sales, marketing, and production managers. Using SAS/Intrnet™ software with other web technology helps provide businesses with new ways to communicate important data in a real-time format for better decision making and business planning.

REFERENCES

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. Other brand and product names are registered trademarks or trademarks of their respective companies.


ACKNOWLEDGEMENTS

Special thanks to my colleague, Rob Watters, for his encouragement, suggestions, and critique of this paper. His knowledge and creativity have guided this project from the beginning.

AUTHOR CONTACT INFORMATION:

Mary Singelais
Bell Atlantic
237 Daniel Webster Hwy.
Merrimack, NH 03054
(603) 595-1074
Mary.E.Singelais@BellAtlantic.com