

The Role of SAS/IntrNet* in a Web-Enabled Database System

John R. Copeland, David W. King, Paul C. Gangarosa; CDC, Atlanta, GA

ABSTRACT

The Centers for Disease Control and Prevention (CDC) has implemented an application, known as the New Vaccine Surveillance Network (NVSN) data system, to study the impact of new vaccines, specifically influenza and pneumococcal. The system allows investigators in Rochester, NY and Nashville, TN to enter patient data, generate reports and graphs, and download data, using only a web browser.

The NVSN data system allows users to store and retrieve data from a SQL Server 7 database. SAS/IntrNet®, along with HTML, JavaScript, Java applets, and Java Database Connectivity, are used for data extraction and analysis. The SAS Output Delivery System is used to send SAS output to users' web browsers. The system employs a Compaq 7000 for data storage and a Sun E5500 for SAS/IntrNet processing.

The NVSN data system demonstrates the ability of the SAS/IntrNet product, combined with other Web and networking technologies, to provide remote users with a myriad of reports, graphs, and statistical analyses - any task that the SAS System can perform - as well as to extract their own data to perform independent analyses.

INTRODUCTION

In November 1999, the National Immunization Program (NIP) of the CDC implemented the NVSN. The initial purpose of NVSN was to measure incidence of acute respiratory infections (ARI) in children less than five years of age, with plans to study how the introduction of new vaccines (such as influenza and pneumococcal) affect ARI incidence. It was decided that

first a system would be put in place to record ARI's from hospital admissions, and later a study involving outpatient ARI's would be added.

NIP enlisted Vanderbilt University in Nashville, Tennessee and The University of Rochester in Rochester, New York to recruit hospital patients to consent to having certain information submitted anonymously to this study. Vanderbilt University has access to three Nashville area hospitals and The University of Rochester has access to two Rochester area hospitals. Medical personnel and epidemiologists from NIP, Vanderbilt, and Rochester devised several forms, to be completed by Vanderbilt and Rochester personnel, to provide the information needed for this study. The forms were designed to collect demographic information, as well as information from hospital charts and laboratory tests.

Once the content of the forms was determined, the method by which the data would be delivered to NIP and stored had to be decided. Three methods were suggested. One suggestion was that Vanderbilt and Rochester would each enter their form data into Epi Info screens. (Epi Info is a thick-client application for data entry and some data analysis. It is widely used in the Epidemiology field.) The resulting data tables would then have to be sent to NIP by FTP, or by storing the data on compact disk and sending the disk by mail. Another suggestion was that NIP would develop software for entry and storage of the form data. This method would also require a means of transporting the data to Atlanta. The other suggestion was that NIP would develop a database and enable Vanderbilt and Rochester

personnel to, using only a Web browser, enter the data over a secure data network.

This presentation is part of the "Internet, Intranet, and Web" section of the conference, so obviously the secure data network is the method that was chosen. Several advantages were noted for using a secure data network over the other two methods:

- Ensures data consistency between the two sites
- Provides most rapid delivery of data to NIP
- Requires no on-site installation of software
- All upgrades can be made at NIP without having to ship subsequent versions of software

NVSN DATABASE

A relational database was developed using Microsoft® SQL Server. The NVSN database contains eight tables (one for each form), and the tables are linked by a unique ID variable. Staff at Vanderbilt and Rochester enter the data from the forms using Active Server Pages, also designed by NIP. SAS/IntrNet is used to allow NIP, Vanderbilt, and Rochester personnel to view reports and graphs; and to allow Vanderbilt and Rochester to retrieve their own data in comma-separated-values format, so that they can generate their own analyses.

SAS/ACCESS

SAS® Version 8.1, which is the version currently being used by NIP, allows tables of relational databases to be read as SAS datasets. Reading MS SQL Server tables requires first creating an ODBC connection from the machine running the SAS session to the database. Once this is accomplished, the SQL Server tables may be accessed as if they were SAS datasets by simply submitting a `libname` statement

designating ODBC as the engine. The statement is of the form:

```
libname sqldata odbc
        noprompt="uid=userid;pwd=password
;dsn=odbcsourcename";;
```

where *userid* is an account with read access to the SQL Server database, *password* is the corresponding password for *userid*, and *odbcsourcename* is the name of the ODBC data source chosen when the ODBC connection was created. The process of accessing the SQL Server tables with SAS is slightly more complicated in the NVSN data system because the SAS/IntrNet product runs on a Sun® E5500 server with the Solaris 2.6 operating system, and ODBC is a Microsoft feature. This problem was solved by scheduling a SAS batch job to run every night on the MS NT Compaq® 7000 server which houses the SQL Server database, utilizing SAS/CONNECT® to copy all of the NVSN tables to the SAS/IntrNet server as SAS datasets.

Another attempted method of solving this access problem was to write a Java application that, when invoked by a SAS program, would read the appropriate SQL Server table and write its contents to the SAS/IntrNet server as a delimited text file. This text file could then be read by SAS with an `infile` statement. This method had the advantage of allowing SAS/IntrNet to capture the most recently recorded data, whereas the SAS/CONNECT method only allows SAS/IntrNet to report data that was entered as recently as the previous day. However, for simplicity's sake, the SAS/CONNECT method is the one that is being used.

SETTING UP SAS/INTRNET

To take advantage of the SAS/IntrNet product, we first had to install SAS version 8 on the SAS/IntrNet server. We followed

the recommendation of the installation program and installed SAS in the OPT directory on our Sun server. Then the IntrNet product itself was installed. A Pearl script file (/opt/sas8/utilities/bin/inetcfg.pl) comes with SAS/IntrNet and performs initial setup. When executed, this program creates subdirectories, sets up a port for default service, and builds the start.pl file. The start.pl file starts the SAS/IntrNet session and names the appropriate Appstart file. We customized the Appstart file to create the libraries to be later used in SAS programs written for SAS/IntrNet.

After the SAS/IntrNet server was configured, our Web server had to be set up to work properly with the SAS/IntrNet server. The same Sun E5500 is used for both the SAS/IntrNet server and the Web server, but this is not a requirement. We use Apache Web server which, after a separate setup program shipped with SAS/IntrNet is run, houses the Broker executable and the broker.cfg file in its cgi-bin directory. The broker.cfg file contains information about our Web server's directory structure and had to be customized for SAS/IntrNet to function properly.

SAS/INTRNET TASKS

The NVSN data system employs a Sun E5500, which serves as a Web server, a SAS/IntrNet server, and houses SAS datasets that contain the same data as the SQL Server tables. What do we want to do with it? What can we do? It was soon discovered that there is little that anyone wants that we cannot provide using SAS/IntrNet, especially when we combine SAS/IntrNet with Java and JavaScript. One of the earliest concerns about the NVSN data was, "is the data being entered logically?". For instance, is the patient's admission date before the discharge date? Or, is the difference between the patient's admission date and birth date within a month of what was entered as the patient's

age? The SAS/IntrNet product provides a convenient way for project scientists to check questions such as these with the click of a mouse. A long list of such logic checks was devised for each NVSN form (some checks involve items from more than one form) and SAS programs were written to subset the NVSN datasets to include records where errors were found, and display the erroneous records in the requestor's Web browser.

Other items of interest on a daily basis are questions about enrollment and parental consent, and the age of patients participating in the study. These are issues that can be effectively addressed with charts and plots. The second NVSN SAS/Intrnet task is to graphically display simple statistics such as frequencies, percents, means, and quartiles.

The third task of SAS/IntrNet within the NVSN data system is to allow the remote sites (Vanderbilt and Rochester) to retrieve their own data, so that data analysts at each site can perform more complex and site-specific analyses of interest to them.

LOGIC CHECKS

As was stated before, a long list of logic checks was compiled for the seven main NVSN forms. To perform the checks, SAS data steps were used to subset the data, retaining records that do not pass the logic checks. Then proc print, along with the Output Delivery System (ODS) directing HTML output to the _webout location is used to print the results to the web browser. For instance, to check for records showing a discharge date before the admission date, code similar to the following is used:

```
ods listing close;
ods html body=_webout (dynamic nobot)
      rs=none;
data screening;
      set nvsnd.screening;
```

```

keep caseid admitdate whoisit
timestamp;
proc sort;
  by caseid;
data chart;
  set nvsnd.chartrev;
  keep caseid dischargedate;
proc sort;
  by caseid;
data screening;
  merge screening (in=a) chartrev
(in=b);
  by caseid;
  if a and b;
data screening2;
  set screening;
  if input(admitdate,mmddyy10.) gt
input(dischargedate,mmddyy10.);
proc print;
  var caseid admitdate dischargedate
whoisit timestamp;
  title 'Admit Date After Discharge
Date';
run;

```

The *nvsnd* data library is assigned in the Appstart program. A *label* statement is also included in the last data step, but was omitted here for brevity. The variable, *whoisit* indicates the person who entered the record and *timestamp* is the date and time that the record was entered. The *caseid* variable is a unique identifier by which datasets may be merged. All other logic checks, such as making sure fields are the correct length and ensuring that age is consistent with birth and admission dates, are performed using this simple method of subsetting and printing with ODS.

GRAPHS

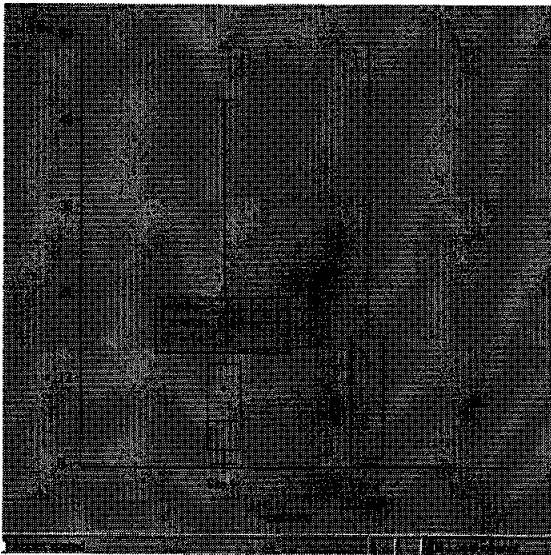
SAS/IntrNet provides a choice of ways to display graph output in a Web browser. ODS can convert output from SAS/Graph to graphics interchange format (GIF) and include the image in a web page. SAS/IntrNet also offers the use of the

%ds2graf(data=dataset, other parameters) macro. This macro sends an applet that displays a histogram, pie chart, or scatter plot, to the Web browser. The variables of interest, as well as information about chart type and chart size, are passed as macro parameters. The applet is interactive: placing the mouse over a component of the graph displays information (variable levels and statistics) about that component; and a right mouse click displays popup menus to allow the user to change items such as graph type, 3D/2D, horizontal/vertical, and colors.

When these two methods, GIF and applet, were demonstrated to NIP, the applet was a big hit. It was decided that applets should be used whenever possible. (Loading times associated with dial-up connections may require us to rethink this decision.) The demonstrations to NIP staff elicited suggestions for types of graphs to show. One suggestion was a box plot showing age statistics for Vanderbilt and Rochester. SAS proc *boxplot* can fulfill this request, but since we are applet fans here at NIP, another method was needed to generate an applet displaying the requested box plot. With *%ds2graf* as inspiration, the *%ds2box* macro was written.

Developing this macro was a long process for a beginning Java programmer. The first step was to write an applet to display box plots with specific values for minimum, maximum, first and third quartiles, median, mean, and axis labels. Then, these values were replaced with the *getParameter*(String name) method; and the HTML that displayed the applet was changed to pass the necessary parameters. (This is the technique used by the *%ds2graf* macro.) Finally, *proc univariate*, *proc means*, and data steps were used to calculate these parameters and, using *put* statements, to write the HTML that displays the applet and passes the needed parameters. These

data steps were encased in the %ds2box macro, which only requires three parameters: a dataset name, a dependent variable, and an independent variable. So a SAS/IntrNet program that calls the %ds2box macro produces an applet, displaying box plots determined by the parameters passed to the macro. The applet is interactive, displaying precise values when the mouse pointer is placed over a portion of a box plot that represents a statistic. A right mouse-click opens a pop-up menu, allowing the user to change the applet colors.



DATA RETRIEVAL

SAS/IntrNet provides an easy way to download data to a client PC. The %ds2csv macro opens a dialog box that displays the user's directory tree. It allows the user to choose the name and location of the data file to be downloaded, and then writes the data to a comma-delimited text file. A file in this comma-separated-values format can be easily opened with Microsoft® Excel and/or converted to a SAS dataset.

PUTTING IT ALL TOGETHER

Once the SAS/IntrNet product has been installed on the server, the broker.cfg file has been customized for the server, and

the Appstart file has been edited to create program and data libraries, one needs only to know a little HTML to develop a functional Web-based data analysis and reporting system. For the NVSN system, the Appstart program designates a directory as the SAS program library, called NVSNP. All NVSN SAS programs are stored in this directory. The appstart program designates a directory to be the SAS data library called NVSND. All of the NVSN datasets are stored in this directory and the NVSND library is referenced in each program without submitting a libname statement. The SAS programs are run by executing the broker using common gateway interface (CGI) technology, and stipulating the desired port service and program. The NVSN data system uses two methods for accomplishing this. One is to simply write a hyperlink to call the application broker and designate the program. In each NVSN SAS program, except for one, the program that downloads the comma-separated-values file, this method is used. For example, the SAS program that performs the logic checks on the "screening" form data is run by writing the following HTML anchor tag:

```
<a href="http://directory/cgi-bin/broker?
_service=default&
_program=nvsnp.screening_checks.sas">
Screening
</a>
```

where *directory* is the root URL for our SAS/IntrNet system.

To download a file, the requestor needs to submit more information than for viewing a graph. The user must specify the site whose data are requested. The user must also specify which of the seven tables is wanted, and they also must submit their site's password. The NVSN system uses an applet to collect and submit this information. The site and table are selected from choice menus, and the password is typed into a text field. The applet will not

run the SAS program if any of these three fields are blank. The SAS program that contains the %ds2csv macro also performs the password validation. If the requestor submits the correct password, then the %ds2csv macro is executed. Otherwise, a SAS program is run that returns an HTML file stating that the password is incorrect. This task of selecting the site and table and entering the password could be more easily accomplished with an HTML form; however, the NVSN system is using an applet due to the programmer's short-lived experiment with client-side password validation. When executed, the applet points to the following URL:

```
http://directory/cgi-bin/broker?  
_service=default&  
_program=nvsnp.download.sas&  
psswd=password&  
location=location&  
form=dataset
```

This is the same format as with the other NVSN SAS programs, except that values for the macro variables "psswd", "location", and "form" are passed to the download.sas program.

CONCLUSION

Web technologies provide a convenient and effective way of transmitting and storing data collected by NIP and its research partners. SAS/IntrNet, along with the ability of SAS to easily read data in a variety of commercial databases, provides an excellent mechanism for retrieving data in a Web browser. The data can be retrieved in the form of lists, tables, graphs, or any type of statistical analysis that SAS can perform. SAS/IntrNet also provides an easy way of delivering raw data to analysts who wish to perform their own analyses.

The development of the NVSN data system has been an excellent educational experience for everyone involved. It has also served as a demonstration tool to

show NIP what we can do with relatively new technologies and it has given us a model on which we can base future projects.

RESOURCES

We took advantage of SAS training to learn some of the skills needed for this project. "SAS Web Tools: SAS/IntrNet Administration", and "SAS Web Tools: Static and Dynamic Solutions Using SAS/IntrNet Software" provided the knowledge needed to begin the SAS portion of the NVSN project. "SAS Web Tools: Advanced Dynamic Solutions Using SAS/IntrNet Software" covers the more advanced features such as the applet and comma-separated-values generating macros. The course notes from all three of these courses provide excellent references and the Web page, <http://www.sas.com/rnd/web/intrnet/format/index.html> is also a useful reference.

ACKNOWLEDGEMENTS

We would like to thank Henry Rolka, David Walker, and Gay Allen of NIP for making this project happen, Kimp Walton of NIP for sharing his SAS programming knowledge, and Fran Walker of NIP for helping to determine the reports to be generated.

CONTACT INFORMATION

John Copeland can be reached by phone at 404-639-8866, or by email at Jcopeland@cdc.gov.

SAS is a registered trademark in the USA and other countries. * indicates USA registration.

Microsoft and SQL Server are either registered trademarks or trademarks of the Microsoft Corporation in the USA and/or other countries.

Compaq is a trademark or registered trademark or service mark of Compaq.

Sun is a trademark or registered trademark of Sun Microsystems, Inc.