Paper 46-26

STDINFO: From SAS/AF to SAS/IntrNet
Reshma Kakkar and Ray L. Ransom, Centers for Disease Control and Prevention

Introduction

Internet/Web based applications are becoming increasingly popular not only as a means of enabling more users to have access to data sources but also as a way of integrating them with emerging technologies. The decision, in this case, to move STDINFO, a SAS/AF application, to the Web, is based on a need to reach a larger audience and increase usage. The use of Web-based technology allows us to remove platform-dependence from the application and capitalize on users' increasing familiarity with Web browsers. From a developer's perspective, web-based applications development offers opportunities for skills development using more relevant and state of the art tools.

Background

The Division of Sexually Transmitted Diseases Prevention (DSTD) at the Centers for Disease Control and Prevention (CDC) in Atlanta collects data on the occurrence of STD's and disseminates this information to fellow program directors, service providers, and researchers in the public health community. STDINFO was developed as a decision support system for the analysis of these data. At present, the system is available on the DSTD LAN only. However, the greater potential of this application would be realized if the data were made available to the state and local health departments responsible for submitting the raw data to CDC. Finally, an organization-wide initiative at CDC towards the development of standards for data collection and storage and for the integration of surveillance systems with a special emphasis on the Web as a tool, is guiding applications development at CDC to the Web.

The STDINFO Project has two major components. The first is a data warehouse for organized storage and documentation of existing and future data files at DSTD and the establishment of data marts that are disease and population specific. The second is a decision support system (DSS) providing data analysis, querying and reporting tools specific to STD data, providing access based on individual access rights to data files stored in the data warehouse.

Our objective is to develop STDINFO as a Web-based application. This paper discusses the challenge involved in retaining the original functionality when moving a SAS/AF application to the Web and the need to strike a balance between the SAS system ® and open systems design. In this scenario, developing a Web-based system is being considered the equivalent of an open systems design.

Current Status

Detailed descriptions of the current design including coding examples were presented at SUGI 23 (Ransom, et al, Paper 258). Below is a brief overview of various aspects of STDINFO/LAN.

Users

STDINFO/LAN is currently installed on approximately 40 machines on the DSTD LAN. Users include medical epidemiologists, statisticians, behavioral scientists, public health advisors, economists, administrators, data analysts and programmers. A sample of STDINFO users belongs to the STDINFO Users Group which serves as a resource for planning application enhancements. All users can provide feedback to the developers through a button on the application toolbar that stores their requests or comments in a table for reference by the developers. Several potential users from state and local health departments have been identified as beta-testers of STDINFO/Web.

A typical scenario of how STDINFO is used: A public health adviser, prior to visiting a county's health department, might wish to print reports stratified by age and gender on the occurrence of syphilis, prevalent and incident, in the county, to facilitate the assessment of STD services provided and planning for improvements. STDINFO/LAN has been used by program staff on site visits to counties that are involved in syphilis elimination activities. The use of the application provides them with information on trends in syphilis, number of cases for the last 5 years according to standard demographics as well as rates and graphics. High quality graphics, such as tables and various types of graphs, can be generated by users from the current week's surveillance data within 24 hours of the submission deadline.

Systems architecture

STDINFO is a client/server application distributed across a Novell LAN with NT clients, a SUN ES3000 UNIX server and an IBM mainframe. The structure of the present system consists of the SAS system on NT clients, the SAS/AF application on a Novell Network/LAN, the data warehouse/metadata on a SUN ES3000 Unix server, and the larger source disease report files residing on the IBM mainframe. Several configurations were tested from an entire client installation to a full server application. The specifics of these tests and the resulting decisions were presented by Humphrey, et al at SUGI 22 (paper 301). STDINFO currently uses SAS version 6.12 and can work on Windows NT as well as Windows 9x operating systems. SAS/Connect and SAS/Access, along with Base SAS Software facilitate the distributed design.

Security

Security for STDINFO is maintained by assigning access rights to users of STDINFO. Novell is the current network software and Novell rights data are piped to ASCII text files and finally SAS data tables as metadata. Assignment of rights is the responsibility of the Database Administrator (DBA) based on guidance provided by appropriate Division management. The metadata-driven design of STDINFO insures that users access always reflects current file access rights as defined by the DBAs. For example, users of STDINFO will only see files to which they have read/write access at the time they access the application.

Division policies for requesting data access, data use, and dissemination are clearly defined and the process is supported through the design of STDINFO and the incorporation of the policies in viewable formats in the application.

Since All source data and warehouse loading programs are stored on the LAN and IBM mainframe, which are backed up daily, STDINFO's fault tolerance is quite high. In the event of a complete loss of the application, the entire application can be restored and all data tables reloaded inside of three hours. Since the application is behind multiple firewalls at CDC and is accessible only to users defined to the DSTD Domain, encryption is not currently employed.

Metadata

The metadata used by STD INFO currently consists of data that lists and describes users, rights, data sources, data tables, variables, and valid variable values. For instance, in order to determine what data sources a particular user should be able to view in a STDINFO session, a table of userids and access rights is referenced upon entry into the system. Data table and variable names presented in list boxes in the application are uses of metadata. The list boxes are dynamically populated so the data does not have to be hard-coded or physically updated each time. By making the application metadata-driven, we can grow and enhance the design with minimal changes to existing code. These metadata structures support various functionality such as the generation of queries and sub-setting of data in addition to populating list boxes.

The only occasion on which manual data entry is required is when new data sources are first registered with the application. The name of the data source, a description of the data, their storage location and the names of those responsible for the data are entered into a single table which is the parent of all subsequent metadata tables which are dynamically generated using the CONTENTS and FORMATS procedures and various Novell commands. Once this is done the data source is available for use depending on user rights.

Interfaces

At present there are two forms of interfaces. The first is a data source specific interface that is customized for the data source that it is related to. That is, the interface contains control objects and screen components specifically designed to facilitate access to a particular data source. This is the interface where users can only see those data sources for which they have been assigned rights except for a couple of data sources that are for common use. The second interface is a generic one with a common design format allowing query access to all other weekly and aggregate project data. This interface allows users to build custom analysis files and store them in various formats for use with their preferred analysis software.

Analyses

The system enables the user to carry out analyses and view the results as trend lines, basic graphs, tables and rates. If the user builds a data set, they have the option to store it as a SAS/SPSS/dBase/Epi Info 6 data set. Analysis and reporting tools have been developed to support specific components of the DSTD mission, such as evaluation of health department clinical services in preparation for syphilis elimination in the United States. In the near future, this will be the aspect of the application deserving the most resources and providing the greatest benefit.

Output

The user can view the output in a graphical format, or use a provided screen capture tool to cut and paste the output to any software package that supports this Windows functionality. The output, in a tabular or graphical format, can be saved on the user's hard disk. The Output Delivery System in Version 8 will be utilized to improve on the current output formats.

Maintenance/System administration

All data warehouse loads are scheduled. Cron tables are used for the UNIX-based data tables and Novell scheduling is used for metadata creation. All metadata are recreated in its entirety nightly and warehouse loads are scheduled based on data reporting cycles. Results logs are stored for all scheduled processes and routinely monitored. This helps determine if the processes are running smoothly and facilitates troubleshooting in the event of a load failure. Currently, the most common cause for failure is system changes on platforms not under our jurisdiction such as the CDC mainframe.

Web-based design

In moving the current SAS/AF application to the Web there are several points to be considered. We will discuss these under two main categories – the tools that will be utilized and the methods.

Tools

HTML and SAS/IntrNet provide us with all of the tools necessary to present the application on the web and retain

current functionality. Insightful planning on the part of SAS/IntrNet developers at The Institute also allow us to capitalize on existing SCL code where we have properly followed the recommended Model-View design concept. That is, our data model is separate from the view allowing us to utilize the old SCL model in the new Web-based view. Desired enhancements might require the exploration of additional tools and methodologies such as XML, JAVA, and other evolving Internet technologies.

Methods

a) System architecture

We will continue to use the SUN ES3000 UNIX server for the metadata and data warehousing. In addition we will use SAS/IntrNet and develop a browser-based interface. At the time of this paper's publishing, an HTML interface has been developed. However this has not yet been done on a CDC web application server as originally planned. Access to this Web server will be provided through the CDC Secure Data Network(SDN) and user requests will be sent to the SAS Application server through a SAS/IntrNet server. Since the present STDINFO/LAN application is not accessed outside CDC's firewall, the SDN's digital certificate-based security has not played a role so far. The ultimate SAS application server will be a high-end Sun server capable of fast queries on large index data tables. The Scalable Performance Data Server will also be used to enhance performance and the Multiple Dimension Database Server will be used to support drill-down functionality.

b) Security

Access to STDINFO/Web will be provided through the CDC Secure Data Network (SDN). The SDN is a collaboration between CDC and Verisign where digital certificates are administered to CDC partners. User profiles collected at time of certificate assignment will be used to build security metadata tables for application guidance.

c) Metadata

As mentioned earlier we plan to continue using the existing metadata structure and make enhancements as additional information or functionality is required. We will use htmSQL to dynamically populate HTML list boxes from the metadata.

For STDINFO/Web we have used data source, data tables, and variable type metadata tables to increase functionality. Metadata are used to dynamically populate the drop down boxes visible to the user containing the various data sources available for use. Once the user selects a data source, htmSQL code allows the generation of data tables using metadata (Figure 1). It is important to note here that the use of existing metadata structures allows us to capitalize on years of SAS data management expertise at CDC to provide metadata to support applications development. SAS/IntrNet is the key link between the old SAS code and the new development

environments without us changing the way we do data management.

As an example, the first html program, which is created using html tags but is saved with a .hsql extension, queries the metadata to populate the list box that displays data sources (Figure 2). In STDINFO the metadata are regenerated everyday; therefore the list boxes are dynamically populated every time the html program is loaded. When the user makes their choice of data source by clicking on it, the 'form' function in html submits the page with htmsql functions to the SAS/SHARE server while executing the htmsql.exe file (Figure 3). At present we are using the Personal Web Server that is a feature of Microsoft FrontPage 98/2000. The cgi program, that is a feature of the Application Dispatcher in SAS/IntrNet, is saved on the Personal Web Server which also runs when the user submits their selection from the browser and passes the information to the application server that is constantly running. The sql program performs updates and retrieves the resulting data set. The results are returned to the file specified in the 'form' function, which is opened in the browser. The user then makes their next selection and the same steps are executed. The only difference is that the user's previous selection is passed in the form function each time in a 'hidden' format so that the user does not see it displayed. This continues till the last .hsql file is executed in the browser and the next step returns the required output to the user. This program passes the information selected by the user to the application broker that runs the SAS program stored on the UNIX server to generate the required output. The output is returned to the web browser in a separate window.

Figure 2

```
<HTML>
<HEAD><TITLE>htmSQL: STD INFO Metadata
</TITLE>
</HEAD>
<BODY BGCOLOR="FFFFFF" leftmargin="0"
topmargin="0">
{query datasrc="DSTDUNIX"}
{sql}
   select libname, liblabel
   from metadata.library
   where server="U" and (libname>="Y1" and
libname <="Y2")
   order by liblabel
{/sql}
```

Figure 3

```
<font face="verdana, arial, helvetica, sans-serif"
size="2"><b>SELECT DATA SOURCE FOR
ANALYSIS</font>
<br>
<FORM name = "mainform" METHOD="get"
target=dataset ACTION="http://std109935/cgi-
bin/htmSQL.exe/sasintrnet/htmSQL/vars2.hsql? "
>
```

```
  <INPUT TYPE="HIDDEN" NAME="_SERVICE"
VALUE="default">
  <INPUT TYPE="HIDDEN" NAME="_DEBUG"
VALUE="2">
  <p><select size="1" name="DS" onchange =
submit() >
          <option value="empty"> - Select from
the list -</option>
{eachrow}
          <option value="{&libname}">
{&liblabel}</option>
{/eachrow}

  </select>
```

d) Interfaces

The interface that the user will interact with in order to access the data will be developed using HTML.

Navigation: STDINFO/Web attempts to avoid the linear navigation style employed in STDINFO/LAN to promote ease of use and a consistent navigational approach in the Web version. Instead tab objects have been created in HTML that allow the developer to display several different options or questions on one screen without having the user go through several screens. If the user changes their mind about an option, they can select the appropriate tab instead of going back through several screens to find the item. This also allows the designers of STDINFO/Web to use the same screens and code to accomplish similar tasks.

Consistent icons and graphics conventions have been used in the application design. For example, when a user clicks on a particular tab to view a specific screen, that tab is a different color than the other tabs visible to the user, thus indicating which tab is in use.

Since the list boxes are dynamically populated, the user does not see the second list box till they have made a selection in the first list box. For example, when the user select a disease for which they want to see data, the next box that pops up only contains the state abbreviations for those states that report data on the disease selected earlier. Similarly when the user selects a state, the third box that appears contains counties for only that state previously selected. At any time the user can go back to the first box and change their selection and the second box is again populated according to the new choice.

d) Analyses

The new Web based system will have similar functionality as the old application in terms of the analyses it will be able to perform. The new interface techniques described above allows the user to repeat multiple analyses with slight modifications to the data and variables selection without having to repeat the entire process.

For example, the final list box allows the user to choose between a table and a type of graph. When the user makes their selection, this runs a program to generate frequencies for the variables chosen. The output is displayed using the 'webout' command. At present the output is displayed in a separate browser window to facilitate printing of the results.

e) Output

We plan to use the Output Delivery System or ODS in SAS for generating output from STDINFO/WEB.

At present we are still using SAS 6.12 and have not been able to bring ODS that is unique to version 8 into use. We do generate an output through 'webout' and have a certain degree of control over the display as we can customize the background color, text font and size. Currently we prefer to invest our precious person-time resources into adding more data and analysis tools rather than enhancing the appearance of the output.

Benefits

Some of the benefits of moving a SAS/AF application to the Web are:

- It will give the application access to a larger audience – in this case state and local health departments throughout the US will be able to use data on STDs. STDINFO/Web will furnish a single source for both data and results. This improved access to data will result in increased awareness of data resulting eventually in improved data quality and more decisions being data-driven as findings are reported by DSTD to state and local health departments.
- It will adhere to recently developed CDC standards for applications development and deployment. STDINFO/Web is also being considered as a possible prototype of a standard for a data analysis and reporting application at CDC.
- It will also help in the development of standards for the purpose of providing epidemiological tools. It has already helped in the standardization and automation of DSTD reports and publications. The use of a map software incorporated into STDINFO has helped in automating the generation of the Division's annual surveillance report to a great extent.
- It will allow us to focus skill development on Internet technology within the Division of STD at CDC. STDINFO/Web is user-friendly and has the look and feel of other Web-based applications. As a result researchers can independently use it to analyze data files and this leaves programmers with increased time for more sophisticated analysis tasks and further skill development such as applications development and

4

object-oriented methodology to support applications development.
- It will facilitate providing an application to users which is easy to use and contains familiar controls and design concepts. The interface for STDINFO/Web looks similar to other Web-based applications so that for users who are familiar with Web browsers, the application proves to be fairly intuitive.

Ultimately, using standardized tools on the Web will allow the public health community as a whole to share tools for disease prevention. These may be tools that are both unique to STD prevention and common to all public health, whether the aim is to improve preventive behaviors, detect and control outbreaks, develop new or better treatments and cures all through efficient use of increasingly over-utilized information technology resources.

References

Humphrey K, Ransom R (1997), "Systems Architecture Solutions for an STD Information System", Proceedings of the 22 Annual SAS Users Group International Conference, pp 1425-1430

Ransom R, Hixon S (1998), **"Client/Server Application Design Strategies for Small Development Teams"**, Proceedings of the 23rd Annual SAS Users Group International Conference, pp 1431-1439

Acknowledgments

SAS is a registered trademark or trademark of SAS Institute Inc., in the USA and other countries.

Contact Information

Reshma Kakkar
Epidemiology and Surveillance Branch
Division of STD Prevention
Centers for Disease Control and Prevention
1600 Clifton Road
Mail Stop E-02
Atlanta, GA 30333
e-mail: rmk5@cdc.gov
phone: (001) 404-639-8362
fax: (001) 404-639-8610

Ray L. Ransom
DOS/CDC
2190 Kampala Place
Washington, DC 20521-2190
e-mail: rlr1@cdc.gov

Figure 1