PAPER 130-27

USE YOUR WEBSITE TO DOCUMENT YOUR WEBSITE

Karen L. Wyland, Decision Support Systems, Sodexho, San Bruno, CA

ABSTRACT

You've developed and put into production a great website using the SAS/IntrNet technologies. You may now be considering tackling the job of documenting your website applications and processes. Have you considered using the very same tools you're using to produce your website to create a website for your development/maintenance personnel?

Decision Support Systems (DSS) at Sodexho makes extensive use of the SAS/IntrNet technologies to keep their development team informed. Several reasons led us to create a developer's website. We're a small team and work on individual tasks. We need to keep current on the status of our projects or "done-bys". We also need to document the procedural tasks we each perform so we have a "turn-over" document ready at all times. It's not always possible for our team to connect to the remote server since we require secure shell access. A website using only a browser interface is perfect for our "anytime, anywhere" needs.

With our development website, we've also decreased much of the static documentation that we all love to maintain! Although we'll always have manual documentation, we've enhanced our knowledge base and clearly met the needs of the administrators and developers in using the SAS/IntrNet tools.

INTRODUCTION

It's often difficult to decide just what you're going to need to include in a knowledge base for the development team until the site is already well into production. Conversely, it's easy to see that you continue to create processes and applications, learn new tricks and tips, and attempt to keep it all up to date -- somewhere, somehow. However, time is passing and you desperately need to create and maintain a knowledge store so your team stays informed. In addition, they need easy access to this information from anywhere using only a web browser.

Let Me At It!

Use SAS! Although this may not seem like a revelation, putting it to effective use may be a daunting task. DSS has been hosting a decision support Intranet at Sodexho for several years now. The development website began almost immediately. It started very simply with links to various installation and procedural documents stored on the server. It now includes scheduled monitoring of processes and logs, administrator messaging on the status of various servers and applications, dynamic analysis of the data warehouse, and a viewable and searchable repository of programs and processes.

Now we want to help you get started in dynamically administering your website. We won't instruct you how to write SAS code, JavaScript, HTML, etc. What we will do is:

- Show you our development website.
- Describe each section.
- Provide some sample code.
- Explore several scheduled SAS batch jobs.

Before we do that, let's look at that huge pile of information and data you've accumulated and think about how you're going to organize it.

VOLUMES OF STUFF - OH MY!

You've probably created a lot of "how-to" documents, tables, or lists of various contacts and resources, collections of manuals, setup and configuration notes, or even flowcharts and network diagrams. Of course, there are also many SAS programs and processes you run, either manually or on a scheduled basis, to keep your website information current. Then there's a list somewhere of the latest changes to the operating system or SAS software and the effects of those changes. Lastly, you surely keep your team members up-to-date on the status of your projects and responsibilities. Although some information will always remain static and manually updated, you can often enhance access by dynamic processes. Moreover, if you hadn't even thought of using SAS/IntrNet to create a development knowledge store, read on.

GOT ANY IDEAS? YOU BET!

For static information, add dynamic lists and links.

1. Schedule a SAS batch job to read the contents of specified directories each night to generate a new listing of various file types such as SAS programs, Word, Excel, html, etc.
2. Create web pages that include the file name, hyper linked to each file for viewing and printing. (see Programs / Data Warehouse, page 3)

For interactive/selective analysis, use the SAS/IntrNet tools to:

1. View datasets in the data warehouse (see Programs / Data Warehouse, page 3).
2. Analyze website and SAS logs.
3. Perform complex searches (file aid) of any file such as but not limited to SAS programs and HTML code.

For critical processes, schedule SAS batch jobs:

1. Validate all registered website users against the company payroll and mail a list of all invalid users to the administrator each night (see Verification of Users, page 4).
2. Send a birthday greeting to web users.
3. Send a page and/or email to the team whenever a server or process goes down, stamped with the date and time (see Server and Process Status, page 4).

Great -- we've peaked you're interest.

DEVELOPMENT TEAM HOME PAGE

Take a moment now and scrutinize a detailed sample of the team home page at the end of this paper. We've organized it based on collections of information and functional processes. We update about half of the sections manually since the data is not readily available and changes at any time. The other half is dynamic

applications that query the warehouse or information created by a SAS process that runs at specifically scheduled times.

Now it's time to roll up our sleeves and look at each section's contents. We'll include a brief description of the type of information or tasks. For those sections with static content, the description may be brief as the intent here is to give you insight into the dynamic processes. For the dynamic sections, we'll include selected sample program code and output for some as the length of this paper does not allow more. Please note: All code is available to anyone who requests it – just contact me.

A. Team Tasks

Keeping up to date with each other

Done-by Lists:
Each month, the team members upload via FTP their "done-by" lists to a specified directory. Done-by lists show all tasks completed by the team member grouped by months. Later this also serves as a planning tool and review for the department. The webpage has links to each team member's page.

Website Tasks / UNIX Tasks
The Website/UNIX administrators maintain these html pages manually and include an ongoing list of completed and planned website additions, deletions, and updates.

B. Contacts / Resources / Escalation Procedures

Where to go and who to call for help.

The information here is critical and maintained manually by the project leader as we cannot load it automatically. If one of us is unavailable, another team member can quickly see whom to contact if necessary.

C. Getting Around the Server

Connections, IP's, Directory Structure.

We manually maintain these tables and matrices. As you know, IP address and connectivity methods constantly change. The availability of this key information has "saved" us more than once.

D. The SAS System

Installation, Setup, Manuals, Tutorials

Don't kid me – you all know where to get this stuff! However, don't forget to include notes on your specific challenges and configurations used on your server as nothing is ever just "out of the box".

E. UNIX Operating System

Not familiar with UNIX…Me Neither.

UNIX Commands and Tasks:
What if a team member is well versed in SAS but has never seen UNIX. There are many helpful resources on the Internet. We decided to narrow those sources by preparing a detailed table of UNIX commands specific to our tasks and our server. We include examples of syntax and expected results.

Cron Jobs Overview:
The UNIX manual pages can be "overkill" at times. That is how it was when we looked at crontab. When our hosting company helped us through the usage of crontab, we documented the steps and description to make it simple.

Shell Scripts:
We use shell scripts for most of our processes. When we schedule a cron job, it executes a script that launches SAS in batch, providing logs for each job run.

Zip and Unzip
There are many versions of ZIP out there. Often getting the correct one for your operating system is confusing. This documents sources and lists the basic commands of our version.

F. Netscape Server Administration

Setting up users, monitoring logs, Getting Started

New User Set-Up:
This process is manual. Unfortunately, we do not have an LDAP (lightweight directory access protocol) server to maintain user and directory mapping. In order to set up a new user, there are three processes: 1) Add user to SAS dataset, 2) Add user to Netscape Web Server, and 3) Send user logon and password.

G. Administrator Tasks

The "How-To" of creating and maintaining the site

Cheat Sheet:
We wanted to give all team members this "cheat sheet" to help in checking out the server and applications. It's actually a type of decision tree. For example:

What If's

1. **User cannot logon to website.**
    1. Is the Netscape Web Server up and running?
    2. Is user inputting the userid and password correctly? Did user forget their password? Look up user in the current Passwords list on the development site.
    3. Test userid and password yourself.
2. **User is unable to retrieve the MDDB application.**
    1. Is the SAS Application Broker up and running?
3. **User is unable to retrieve data from the SQL application.**
    1. Is the SAS Share Server up and running?

Each hyperlink runs a SAS/IntrNet program to test the various applications. The real nicety is there are also examples of expected results that give the team member a comparison for confidence.

Running Process:
Each month, we refresh our data warehouse. We call this the "PROCESS". This section has a five-page document that takes a team member systematically through the three-hour process. There are numerous links to documents that help proof for correct results and links to other helpful resources.

H. Website Users and Usage

Interactive Requests.

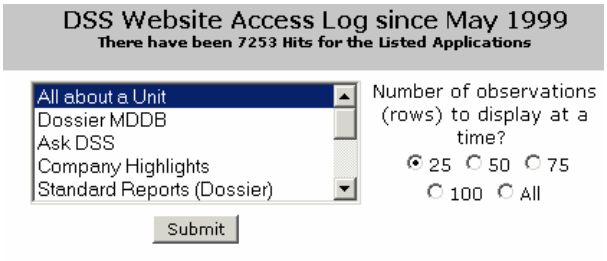
Passwords:
A SAS/IntrNet application for viewing the current web user roster with user demographics and passwords that lets us quickly help users with logon issues and produces a total user roster.

Web Stats (All):
A scheduled SAS batch job runs each night that reads our Netscape Web Server logs and creates a text file listing all accesses by user, sorted by date and time, and summed to show a user's total time online.

Web Stats (by App):
This is a SAS/IntrNet application that let's us query the

Netscape Web Server logs by application. We can customize the output by the number of observations to display.

Sample Selection Page:



I. Programs / Data Warehouse

Viewing and Printing Program and Data Repositories.

Program Repository:

Each night a scheduled SAS batch job is run that reads all program directories and creates a hyper linked web page of the all SAS programs used in production as well as those called directly from the SAS/Broker. We segregated the links in this section by type of program, such as development, production, and web applications.

Sample SAS program:

```
/*-----------------------------------------------
PROGRAM:        PGMS/WEB/DEVEL_LIST.SAS
PURPOSE:        Create HTML pages with links to
                view all SAS programs
AUTHOR:         Karen Wyland, AUG2001
Notes:          Cronjob runs each night
-----------------------------------------------*/
filename stuff pipe 'ls /usr/web/htdocs/webpgm';

data _null_; file
'/usr/web/htdocs/development/webpgms.htm' ls=190
lrecl=190;
     infile stuff pad missover;
     input aline $char200.;
     put "<a href=""webpgms/" @;
     put aline @; put '">' @;
     put aline @; put '</a><br>';
     run;

filename stuff pipe 'ls /usr/dss/pgms/';

data _null_; file
'/usr/web/htdocs/development/pgms.htm' ls=190
lrecl=190;
     infile stuff pad missover;
     input aline $char200.;
     put "<a href=""pgms/" @;
     put aline @; put '">' @;
     put aline @; put '</a><br>';
     run;

filename stuff pipe 'ls /usr/dss/pgms/web';
```

Data Warehouse:

Using the sashelp.vtable dataset and the SAS/IntrNet websamp.dataset.scl code, we've created a SAS program, called from the SAS/Broker, to produce a table of all datasets applicable to the DSS Website. Each dataset is hyper linked to the websamp.dataset.scl code to return a view of the variables and opportunity to download it.

Sample SAS program:

```
/*-----------------------------------------------
PROGRAM:        /PGMS/WEB/DATADICT2.SAS
PURPOSE:        Allow browsing data warehouse
AUTHOR:         Karen Wyland, APR2001
NOTES:          Uses Cascading Style Sheet for
                Output and dataset.scl by SAS
-----------------------------------------------*/
data _null;
 call symput('titdate',put(today(),worddate.));
 call symput('browtit','Data Dictionary');
 call symput('titwords','Data Dictionary from
sashelp.vtable');
run;

proc sql;  create table work.fluff as
     select libname, memname, memtype, memlabel,
nobs label='# OBS ',
             nvar label='# VARS ', crdate,
modate, indxtype
     from sashelp.vtable
     where libname
in('SASDL','BIP','EUP','HKP','CNP','BNP','SFP','
FXP')
     order by libname, memname;
     quit;

proc sql noprint;
     select count(*) into :num
     from work.fluff;
     quit;

data _null_;
     file _webout;
     set work.fluff nobs=nobs end=thatisit;
     if _n_ = 1 then
     do;
       put 'Content-type: text/html';
       put ;
       *--- Report Title --;
       put '<font color="#800080">  ';
       put '<table border="0" width="100%"
style="margin-top: 0; margin-bottom: 0">';
       put '<tr><td width="100%"
bgcolor="#C0C0C0"';*bgcolor="#9585CB"> ';
       put '<p align="center"><font
face="Verdana" size="2"><b>';
       put "&titwords (&num)</b><br>";
       put "<font size=""1""> Click on a Dataset
Name to View and Download the Contents";
       put "<br>as of Period &period &pd_mo_L FY
&fy_t"<br>';
       put '</td></tr></table></b><div
align="center"> ';
       put '<table border="0" cellpadding="0"
cellspacing="0" width="65%"';
       put 'style="font-family: Verdana; font-
size: 9pt"></tr>';
       put '<td
width="30%"><b>Libname.Dataset</td>';
       put '<td><b>Nobs</td>';
       put '<td><b>Nvar</td>';
       put '<td><b>IndxType</td>';
       put '<td><b>CrDate</td></tr>';
     end;

put '<td><a href="../cgi-
bin/broker?_service=default&_program=sample.webs
&dataset.scl &dataset=';
     put libname +(-1) '.'  memname;
     put '">' libname +(-1)'.' memname '</td> ';
     put '<td>' nobs' </td>';
     put '<td>' nvar' </td>';
     put '<td>'indxtype' </td>';
     put '<td>'crdate' </td>';
     put '</tr>';

     if thatisit then put '</table></div>';
   run;
```

Sample output:

Data Dictionary from sashelp.vtable (285)
Click on a Dataset Name to View
and Download the Contents
as of Period 3 November FY 2002

Libname.Dataset	Nobs	Nvar	IndxType	CrDate
BIP.ACCOUNT	1577	2		12DEC01:11:44:48
BIP.BALBASE	43424	66	SIMPLE	12DEC01:11:41:58
BIP.BALSHEET	43424	59		12DEC01:11:12:57
BIP.BAL_ACT	395	2		12DEC01:11:44:53

CRITICAL PROCESSES – SCHEDULED JOBS

Apart from the Development Team Home Page, some of our most important processes are "behind the scenes". They are so important in helping us maintain the site, we want to share several of those with you.

A. Server and Process Status Notification

Instant Notification of Problems with Server and/or Processes.

We have processes that must be up and running at all times. This includes SAS/Broker1, SAS/Broker2, SAS/Share, Netscape Web Server, and Netscape Admin Server. If any process stops, the website will not function properly. Our solution was to schedule (crontab on UNIX) a SAS batch job that runs every 10 minutes of every day to check and see if everything is running. When any exception occurs, we send a cell phone page and email immediately to the administrators, stamped with system date, time and the exceptions.

Manually, we'd have to do the following:

- Connect to server and log on as administrator.
- Check to see that all processes are running.
- Perform this task every 10 minutes.

Sample SAS program:

```
/*----------------------------------------
PROGRAM:  PGMS/SERVERS_STATUS.SAS
PURPOSE:  Are servers up and running?
AUTHOR:   Karen Wyland, JUL2000
Notes:    Mails to Admin cellphone and emails
if one or more servers are NOT running.  Set
in crontab for running every 10 minutes.
----------------------------------------*/

*--- Create text file with list of processes
that involve the website servers running --;

data _null_;
%sysexec(ps -ef|grep 'suitespot'>pp.txt);
%sysexec(ps -ef|grep './sas_share'>>pp.txt);
%sysexec(ps -ef|grep './sas_broker'>>pp.txt);
run;

*--- Initialize Macro variables --;
%let flag=;
%let name=;

*--- Read in list and check each process --;
data _null_; file
'/usr/dss/servers_status.txt';
   attrib name format=$10.;
   infile 'pp.txt' end=thatisit pad missover
ls=199;
      if thatisit then
      do;
        flag = sum(of t1-t5);
        call symput ('flag',compress(flag));
        name = '';
        if not t1 then name='dog';
        if not t2 then name='httpd';
        if not t3 then name='admin';
        if not t4 then name='share1';
        if not t5 then name='broker1';
        call symput
('name',compress(upcase(name)));
        file log;
        put '----> ' flag= name=;
      end;

      input aline $char199.;
      x = index(aline,'grep');
      if not x;
      put aline;
      if index(aline,'./uxwdog')    then t1+1;
      if index(aline,'ns-httpd')    then t2+1;
      if index(aline,'ns-admin')    then t3+1;
      if index(aline,'share1.sas') then t4+1;
      if index(aline,'broker1')     then t5+1;
      run;

*--- Add message on status of servers --;
data _null_; file
'/usr/dss/servers_status.txt' mod;
      put;
      if &flag lt 5 then put "On &sysdate at
&systime, the &name Server is DOWN!";
      else put "On &sysdate at &systime, All
Servers are UP and RUNNING!";
      run;

*--- Check condition for Down Servers and
send messages to Administrators --;
data _null_;
if &flag < 5 then
   do;
        command1= "mailx -s ""On &sysdate
&systime the &name Process is DOWN on Minny""
admin1@xyz.com < servers_status.txt";
        call system(command1);

        command2= "mailx -s ""On &sysdate
&systime the &name Process is DOWN on Minny""
555-111-2222@mobile.att.net <
servers_status.txt";
        call system(command2);

   end;  stop;
   run;
```

Sample Email Message: (if a process is down)

From: dssadmin@sodexho.mn.uswest.net
To: kwyland@sfo.com
Cc:
Subject: On 09AUG01 22:23 the BROKER1 Process is DOWN on Minny

B. Verification of Users

Instant Notification of Users no longer on company payroll.

Security is extremely important. One security check is to know when a web user has left the company so we can cancel their logon privileges. Twice a week we receive via FTP an updated dataset called "tolist" which contains up-to-date employee rosters. We have a scheduled cron job that runs bi-weekly to check for this new dataset, import it, and update the warehouse. That takes care of updating our "tolist".

The next step is to run a bi-weekly SAS job that checks the users database against the updated "tolist" to remove any users no longer on the payroll, stamp their entry in the user dataset as removed with the current system date and send an email to the administrators as notification of any removed employees.

Sample SAS program:

```
/*-------------------------------------------
PROGRAM:  USERS_CHECK_PAYROLL.SAS
PURPOSE:  Check for users no Longer on Payroll.
          Runs against bi-weekly tolist.
          Stamp emps removed with sysdate.
          Send Message to Administrator.
AUTHOR:   Karen Wyland, SEP2001
NOTES:    Scheduled cron job runs bi-weekly.
-------------------------------------------*/
%MACRO DOIT(DSET=DSET);

proc sort data=sasdl.&dset out=work.&dset;
     by name; run;
proc sort data=sasdl.tolist out=work.tolist;
     by name; run;

data work.stuff ;
     merge work.&dset (in=m)
           work.tolist   (in=t);
     name = upcase(name);
     by name;
     if (m) and (not t) and (not removed);
     removed = "&sysdate"d;
    run;
%MACRO FOUNDIT;

*--- If Deleted Entry exists ---;
data work.merged;
     merge sasdl.&dset
           work.stuff;
     by name;
     run;

*--- Mail List of Users Removed ---;
x 'rm external/oops.txt';
title2 "&dset Stamped as Removed from &dset
Database";
data work.oops; file 'external/oops.txt' new;
     set work.stuff;
     do;
      output;
      put "&dset Checked ToList on &sysdate";
        put svp name assigned dassign removed;
     end; run;

data _null_;
     set work.oops;
     if _n_ = 1 then
     do;
         command =  'mailx -s "Users NOT on
TOLIST" admin1@xyz.net < external/oops.txt';
          call system(command);
     end;
     run;
%MEND; *FOUNDIT ;

data _null_;
     rc = exist('work.stuff','data');
     if rc = 1 then %foundit;
     run;

%MEND; *DOIT;

%DOIT(DSET=ENDUSERS);
*DOIT(DSET=WEBUSERS);
```

CONCLUSION

The Decision Support Systems team needed an up-to-date knowledge base to keep on track with the DSS Intranet site. In addition, we needed to access this information from anywhere, anytime. It's not always possible to connect to our remote server since we require secure shell access. A website using only a browser interface is perfect. If an administrator or developer is traveling, they can now quickly check out the website. Whether it's updating team project management, searching for and printing programs, viewing and downloading data from the website warehouse, helping a user with logon procedures, performing another member's procedural tasks, or performing website trends, it's all there.

Each week, we think of new ways we can be more productive. Now we have a framework upon which to grow.

ACKNOWLEDGMENTS

The Technical Support staff at SAS Institute, as always.

The Sun Hosting Engineering team at Qwest who helped us learn the world of UNIX.

Mr. Nick Thaler, my mentor, friend and boss for his day-to-day support and enthusiasm.

REFERENCES

SAS Institute's Web Site - Web Technology, Communities.

Gilly, Daniel (1986). Unix in a Nutshell,. Cambridge: O'Reilly & Associates, Inc.

SAS Institute, Inc., *SAS Companion for UNIX Environments: Language, Version 6, First Edition*, Cary, NC: SAS Institute, Inc., 1993. 256 pp.

Lots of trial and error. ☺

CONTACT INFORMATION

Your comments and questions are valued and encouraged.

Please get in touch if you'd like additional program code which we were unable to include here.

Contact the author at:

Karen L. Wyland
Sodexho
Decision Support Systems
883 Sneath Lane, Suite 213
San Bruno, CA 94066
Work Phone: 650-742-7630
E-mail Address: kwyland@pacbell.net

Development Team Home

Team Tasks Contacts/Resources/Escalation Procedures Getting Around the Server The SAS System Unix OS Netscape Server Administrator Tasks Website Users/Usage Program Repository/Data Warehouse

A. **Team Tasks**
   1. Doneby Lists
   2. DSS Website General Tasks Ongoing list of open and completed tasks.
   3. SAS on UNIX Tasks Specific tasks for SAS Software.

B. **Contacts/Resources/Escalation Procedures**
   1. Contacts Who our team contacts for help.
   2. Escalation List How US West contacts our team.
   3. Software / Hardware Specs Versions, installations, and configurations.

C. **Getting Over, Under, Around and Through**
   1. Access logons and IP Addresses Needs monitoring for updates.
   2. FTP Matrix (From-to) Roy's cross-referenced IP addresses for FTP
   3. Minny Directory/File Structure Detailed listing of types of files stored on Minny.

D. **The SAS System, v6.12, TS055**
   1. SAS 6.12 IntrNet Web Tools on SAS Website Very complete documentation, html ready.
   2. SAS 6.12 Installation Notes on DSS Install on Solaris.
   3. Search SAS-L Archives Type in a keyword to search the Deja SAS-L newsgroup.

E. **Unix Operating System**
   1. UNIX Commands and Tasks Frequently used on Minny, with samples and explanations.
   2. Cron Jobs Overview US West put together a great document on getting started.
   3. Shell Scripts Samples of beginning MAIL scripts.
   4. Zip and Unzip Usage examples.

F. **Netscape Server Administration**
   1. New User Set-Up What steps to take when setting up a new user with access?
   2. Website Access Log Analyzer Output of Netscape Web server Administrative reports.
   3. Big Block.com Explanation of HTTP Status Codes in Access and Error Logs.

G. **Administrator Tasks**
   1. Cheat_Sheet Instructions for managing Minny.
   2. Running Process Step-by-Step instructions on creating "fsm$fast", the data warehouse on Minny.
   3. Process Flow Matrix of programs, sub-pgms, datasets called, datasets created, reports created.

H. **Website Users/Usage**
   1. Passwords Generated dynamically whenever password database, sasdl.passwords is edited.
   2. Web Stats (All) Web Access Log Statistics by User since started collecting logs May 1999.
   3. Web Stats (by App) Web Access by Application Accessed.

I. **Program Repository/Data Warehouse**
   1. Web Programs Called from the Broker Copy of /usr/web/htdocs/webpgms/ (broker programs).
   2. Production Programs Copy of /usr/dss/pgms (both production and development)
   3. Web Production Programs Copy of /usr/dss/pgms/web (both production and development)
   4. Display Website Datasets Outputs an html table of the sashelp.vtable for various datasets and allows user to click on a dataset to display it and download it into excel.