Paper 196

Rocky Mountain Web: Telecom Network Interconnection Quality Measurements

Faith Reneé Sloan
FRS Associates, LLC San Francisco, California
Larry Bramblett
Datawarehouse Solutions, LLC San Ramon, California

Introduction

The Telecommunications Industry is in a race. A race to merge, takeover, open new markets and gain control of both local and long distance service. Those companies that wish to win can only offer two things: *price* and *service*. To gain the competitive advantage requires complex statistical quality measurements. The SAS ® system delivers the platform that makes it all possible.

USWest ® Telecommunications, as a regulated local telephone service provider, is required to open its market to local competition. In turn USWest's goal is to be given access to the lucrative long distance market. The key to gaining access to the long distance market is demonstrating to State and Federal utility authorities that USWest's business practices allow Competitive Local Exchange Carriers (CLEC) a level playing field. No small task when USWest territory covers 14 states and operational systems from several different regional companies.

Project Scope

In 1997 USWest embarked on the task of developing a system that would provide service level measurements to utility authorities and under contract CLECs. The two business units that had direct interest in the success of the new system were Marketing and Network Operations. Business goals of the two groups where defined and the project was divided between contract measurements and network service measurements.

USWest Network Global Measurement Group is chartered as the staff support group responsible for reporting on network-wide service levels. The new Interconnection reporting platform fell within their area of responsibility. The group has a long history of analytical reporting using the SAS ® system. What the team did not have was experience in the development of an Intranet-based, Data Mart containing large numbers of measurements. An external search was conducted to pull together a team that could provide the expertise and vision needed for success.

Three areas of needed expertise were identified:

- A Data Warehouse/Data Mart Architect
- An Intranet Architect
- A Statistician

Trilogy Consulting, Data Warehouse Solutions, LLC and FRS Associates, LLC combined with the USWest Network Global Measurement Group to form the Co-Provider Analysis System(CPAS) project team.

Emerging Technology Supports Emerging Requirements

The Measurements Group had a general idea of what was needed but did not have detailed system requirements. Using the SAS® systems capability to support rapid application development, several prototypes where processed and a design emerged that supported long term corporate measurement goals.

CPAS would support long term storage, 12 months at first and expandable. The system had to be "data source independent" and process multiple data sources from existing operational systems and Data Warehouse storage. The system had to process service data with varied attributes and statistical measurement rules.

CPAS had to control measurement reporting by preventing modification by end users. In addition to CPAS controlled measurements the platform had to provide data mining capabilities.

CPAS had to be supported in a secure Intranet environment with access given to a pre-defined list of management personnel, analysts and attorneys on a need-to-know basis.

The application must be browser independent (Netscape and Microsoft 3.0 and higher) without the need for plug-ins.

Cross platform data retrieval and deliver would require CPAS to process on one system and deliver information to another.

CPAS had to be automated to allow production level processing without manual intervention.

The platform had to allow for expanding and changing network measurements without re-coding and be completely table driven.

The SAS® System Delivers

Primary data retrieval came form a SAS Data Warehouse located on a Sun Enterprise 6000. Initially 200 measurements where identified as input into CPAS. The number quickly grew to over 600 product level measurements.

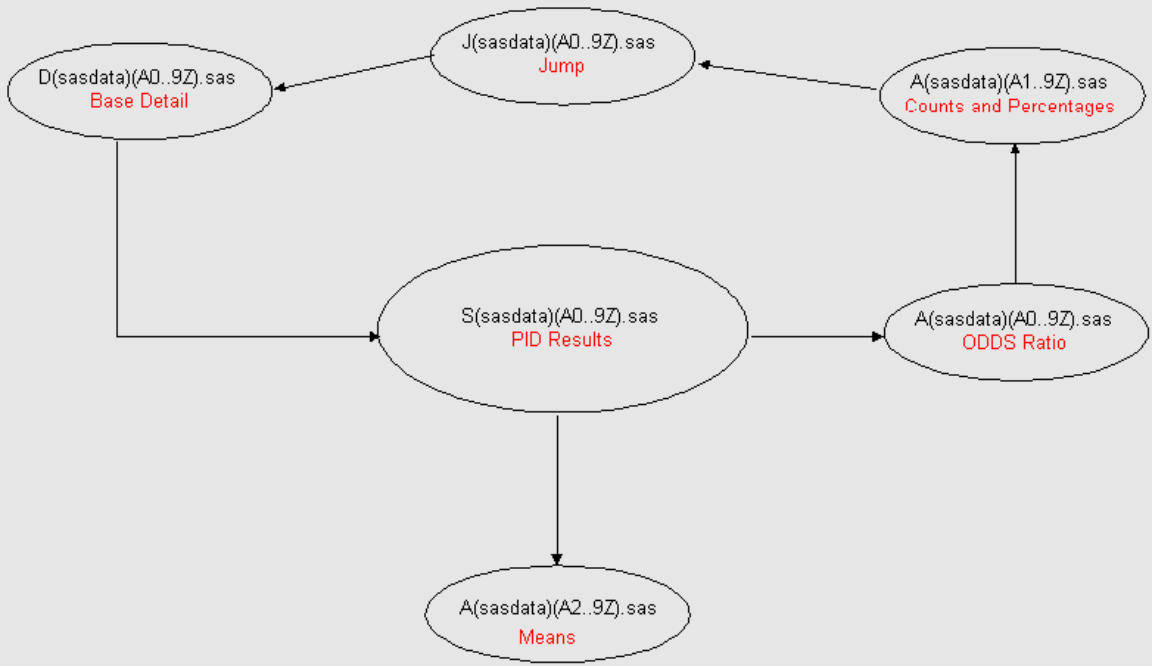
Evaluation of repair requests, service orders, and network outages required advanced statistical routines. All measurements where conducted at both wholesale and retail levels, by each CLEC and for USWest as a company. Measuring quality and service required comparing each group against every other group. Models where developed for each measurement and grouped at appropriate levels.

Grouping input data sources and measurement business rules focused the design into a Data Mart star schema that would support each measurement group. Each group is independent yet accessible by the others. Each group supports it's own SAS/MDDB ™ linked to the detail use to create the controlled statistical measurements.
In total, the data groups form the CPAS Data Mart.

Each data group consists of a detail data table, a statistical data table, measurement performance indicator (PID) results table(s), Indexed drill down (Jump) table and MDDB data store.

Data store naming convention support year 2000 by using SAS date as part of naming convention. Each table data group supports up to 999 sub groups per storage date. Primary tables start at A0 with sub-groups A1, A2, and so on.

Sub tables contains pre-conditioned measurements which provide high-speed access by SAS/IntrNet™ and eliminate modification of measurement results by end users. Navigation starts at the Odd Ratio table or the Means table, drills to Counts and Percentage and drills to detail by way of indexed jump tables. See the schema below.



SAS/MDDB data stores are created from base detail table and accessed via the Intranet using the SAS Institute-supplied SAS/IntrNet MDDB Report Viewer.

Additional data group levels for new analysis are added as needed to the data group ring . Each new data group maps back to the original data store.

Going up a Level

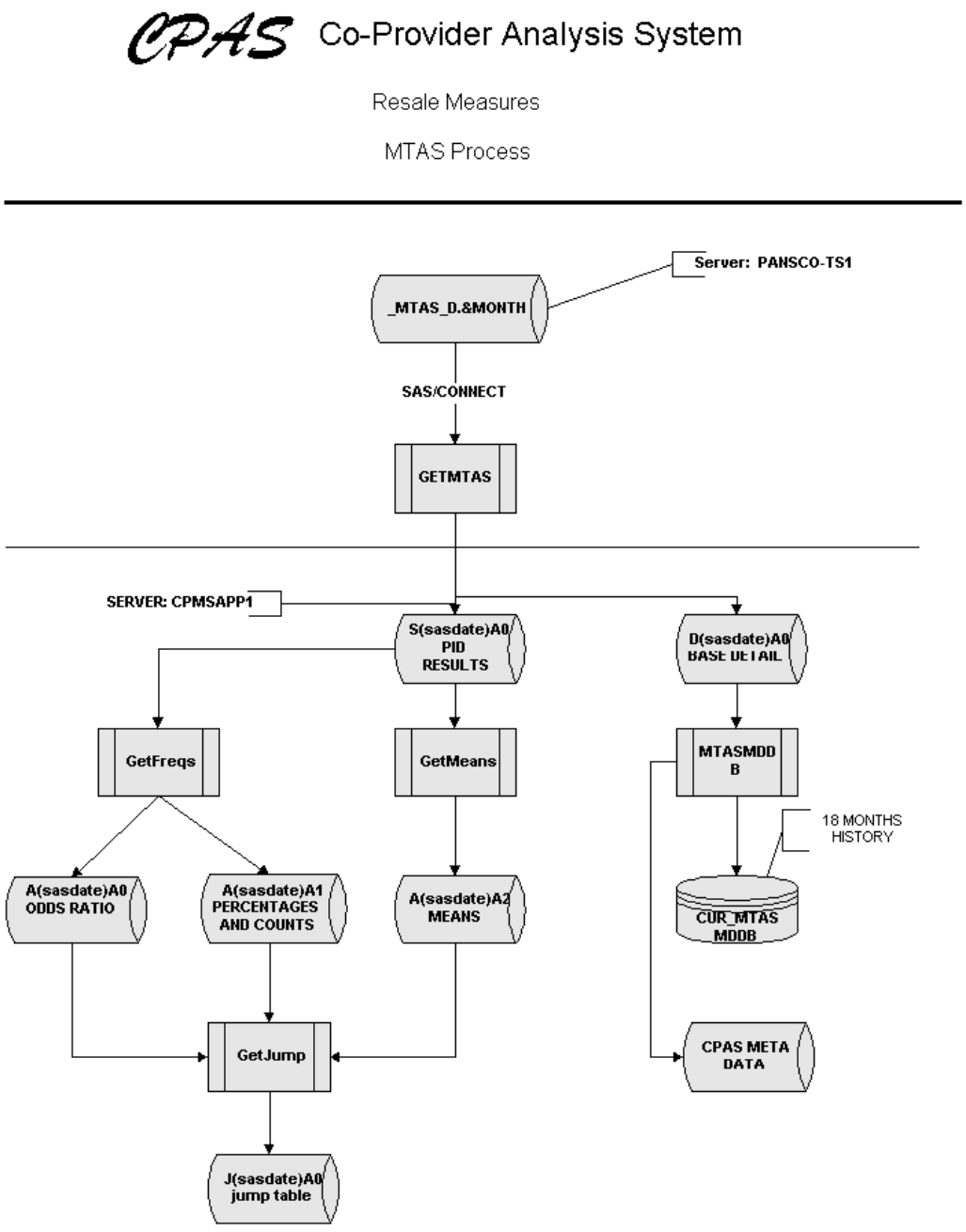
As measurement groups became known, CPAS matured to the next level of Data Mart development. SAS macro source code replaced redundant measurement code to simplify production processing. Next an internal data dictionary was developed to provide information to end users about

complex Performance Indicator (PID) or measurements rules. By bringing PID rules on-line confusion about how measurements are processed, data origination and system code was minimized.

Opening up Documentation

On-line visual documentation provides end users graphical representation of the system and all data stores. Using Visio ® 4.0 for Microsoft Windows, data flow diagram charts were created and exported as images suitable for the Intranet. With SAS/IntrNet providing the gateway, the CPAS platform opened online exploration of SAS data sets and Metadata definitions to the end user community. See Diagram below.

CHART: MTASPDF
MODIFIED: Tuesday, September 29, 1998



Stacking Multi-Dimensional Databases

While the main focus for CPAS is controlled statistical measurements, multiple SAS/MDDBs provided the ability to perform data mining by focusing on ad-hoc requests. With each data input source a paired MDDB provides access to historical data trending and detail level analysis.

Supported by SAS MDDB Report Viewer downloaded from SAS Web site, each primary SAS data set is uploaded monthly into the appropriate SAS/ MDDB. The MDDB groups are linked with the Multi Dimensional HTML page. Both one and two-dimensional reporting along with graphics is available to CPAS visitors.

CPAS Intranet Details

SAS/IntrNet

The SAS/IntrNet software allows developers to create interactive web-based applications which gives users the ability to not only access data but to also analyze and display data dynamically. One of the advantages of SAS/IntrNet is that these web-enabled applications allow us to globally distribute complex applications without the need to install the SAS Software on every user's machine!

SAS/IntrNet consists of many components. In the interest of time, this discussion will not elaborate upon the Multi-Dimensional Database or the MDDB Viewer. The rest of this presentation will focus upon one very powerful and practical component – **The Application Dispatcher.** This component forms the backbone of the CPAS Intranet.

First we will provide some insight into the Common Gateway Interface or CGI.

Common Gateway Interface (CGI)

CGI is ***NOT*** a programming language. It a protocol or way developers can integrate external gateway scripts and programs with the Web. Before CGI came along, we were only able to render plain, *static* HTML documents via the browser. However, a CGI script or program is executed in real-time, so that it can output dynamic information.

The most common languages used by Web developers are Perl and C++. CGI Scripts are also written in UNIX shell, Tcl, VBScript, Visual Basic and a host of other programming and scripting languages.

The Application Dispatcher component is a Web gateway written by SAS Institute using the Common Gateway Interface (CGI). The power of SAS/IntrNet is that you do not need CGI programming experience in order to use the Application Dispatcher. All you need to do is create the Web user interface and develop the back-end SAS programs! This allows Web developers to minimize development time by leveraging their existing SAS programming experience.

Application Dispatcher

The Application Dispatcher is the component of SAS/IntrNet software that lets you send information from a web browser to a SAS session for processing and returns the results to your browser. The Application Dispatcher has two components:

1. the Application Broker (SAS Institute-supplied CGI program)
2. the Application Server (SAS session)

Here's how they work together.

First, a user submits a request through a Web browser. The Web server receives the request and invokes the Application Broker.

The Application Broker is a CGI program written by the SAS Institute which resides on your web server. It interprets the user request and sends a message to the Application Server via TCP/IP (the Internet Protocol) passing parameters such as which program to execute, which data to use, etc.

A Dispatcher *program* is the actual SAS, SCL, or macro code you write which processes the data. A Dispatcher *application* is the combination of a Dispatcher program (or programs) and the HTML pages that serve as the user interface.

The Application Server is a SAS session that calls the appropriate Dispatcher program from the Application Library defined by the developer. The Dispatcher program may be a SAS program, a macro, or a compiled SCL program. The Dispatcher program runs, and then sends data to the Web server. The data may be an HTML file, plain text, graphics, some other format, or some combination of formats. The Web server sends this data to the user's Web browser, and the transaction is complete.

The CPAS Dispatcher application consists of a few HTML pages and many SAS programs.

CPAS Application

The Cpas_Opts.sas program was created to contain all of the global parameters such as libnames, filenames, and macro variables. It is called or 'included' in every SAS program. For example, /WEB4/CPAS/DATA is where all application data is stored. There are also defaults associated with the HTML pages such as background color (bgcolor), visited link color (vlink), font size (fsize), font face (fface), etc.

```
/*=======================================*/
/* SET LIBNAMES/FILENAMES as well as MACRO */
/* VARIABLES -- REQUIRED                 */
/*=======================================*/
%LET DATALIB=CPASDATA;
%LET TEMPLIB=TEMPLIB;
LIBNAME &DATALIB "/WEB4/CPAS/DATA";  *--
REQUIRED;
LIBNAME &TEMPLIB "/WEB4/CPAS/DATA/TEMP"; *--
REQUIRED;
LIBNAME LIBRARY "/WEB4/CPAS/FORMATS";  /*
FORMAT LIBRARY */    *-- REQUIRED;
%LET DATA=/WEB4/CPAS/DATA/;
%LET debugst=0;
%LET service=newsas;
%LET bgcolor=ccffff;
%LET vlink=RED;
%LET link=BLUE;
%LET text=BLACK;
%LET fface=Arial,Helvetica;
%LET fsize=4;
```

Next the home page, index.html was created. HTML Frames were used so that we can always have the month/state/compliance selection criteria available from the top frame. The bottom frame will be used for results generated from the query. Initially it displays the welcome page which also have a link to the MDDB Viewer. See below.

This page allows the visitors to select compliance and state parameters which are then passed to the Application Broker, /cgi-bin/broker, which in turn communicates with the Application Server to call a Dispatcher program, GetRpt.sas, which resides in an Application Library pre-defined on the server as *complian*. GetRpt.sas sends the HTML results back to the Web server, which renders the results to the user's Web browser.

Index.html is coded as follow:

```
<HTML>
<HEAD>
</HEAD>
<frameset rows="40%,*">
  <frame src="/cgi-
bin/broker?_service=newsas&_prognam=complian.Se
lect.sas&_debug=0 name="select" autoscroll
resize="NO">
   <frame src="welcome.html" name="main"
autoscroll resize="NO">
</frameset>
</html>
```

Simply put, index.html uses the application dispatcher program, Select.sas. Select.sas dynamically populates the month and state pull down lists and finally displays the top frame including the USWest logo.

```
%MACRO Select;
  %include '/WEB4/CPAS/SOURCE/Cpas_Opts.sas';
  %getmonth
  DATA _NULL_;
    file _webout;
    PUT 'Content-type: text/html';
    PUT;
    PUT '<HEAD>';
PUT '<TITLE>Choose Top-Level
Parameters</TITLE>';
    PUT '</HEAD>';
    PUT '<BODY bgcolor="white" TEXT="'"&text"'"
LINK="'"&link"'" VLINK="'
"&vlink"'">';
    PUT '<FONT FACE="'"&fface"'"
size="'"&fsize"'">';
    PUT '<FORM ACTION="/cgi-bin/broker"
METHOD="POST" TARGET="main">';
    PUT '<INPUT TYPE="HIDDEN" NAME="_service"
value="'"&service"'">';
    PUT '<INPUT TYPE="HIDDEN" NAME="_program"
value="complian.GetRpt.sas">';
    PUT '<INPUT TYPE="HIDDEN" NAME="_debug"
value="'"&debugst"'">';
    PUT '<INPUT TYPE="HIDDEN" NAME="cnttype"
value="html">';
    PUT '<CENTER>';
    PUT '<TABLE size="500" border="0"
cellpadding="0" cellspacing="0">';
    PUT '<tr>';
    PUT '<td>';
PUT '<img src="/CPAS/images/uswestlogo.gif"
height=57 width=131 alt="USWESTLOGO"
 align="middle">';
    PUT '</td>';
    PUT '    <td width="100"
valign="top"><br>';
    PUT '      <b><font
color="purple">COMPLIANCE</b></font>';
    PUT '    </td>';
    PUT '    <td width="200" valign="top">';
    PUT '      <font
color="purple"><center><b>Select one<br>
MONTH</center></b></font>';
PUT '</td>';
    PUT '    <td width="200" valign="top">';
    PUT '      <font
color="purple"><center><b>Select at least
one<br> STATE</center></b></font
>';
PUT '</td>';
```

```
PUT '<tr>';
    PUT '<td>';
PUT '<font face="&fface" size="+1">Co-Provider
Analysis System (CPAS)</FONT><br><a
href="/CPAS/DOCS"
target="main"><b>DOCUMENTATION</b></a>';
    PUT '</td>';
    PUT '    <td valign="top">';
    PUT '      <INPUT type="radio" name="compl"
value="1">Compliant';
    PUT '         <br>';
    PUT '      <INPUT type="radio" name="compl"
value="0" CHECKED>Non-Compliant';
    PUT '         <br>';
    PUT '      <INPUT type="radio" name="compl"
value="2">UnResolved';
    PUT '    </td>';
    PUT '    <td width="200" valign="top">';
    PUT '       <center>';
    PUT '       <SELECT name="month"
size="3">';
  RUN;
  DATA _NULL_;
    SET availmo; /* AVAILMO is created from
getmonth.sas autocall macro */
    file _webout;
    BY MONTH;
    LENGTH SEL $8;

/*=======================================*/
/* Make current month the default selection */
/*=======================================*/
    IF LAST.MONTH THEN SEL="SELECTED";
    ELSE SEL="";
    PUT '         <OPTION value="'month'" '
SEL'>'LONGMO;
  RUN;
  DATA _NULL_;
    file _webout;
    PUT '       </SELECT>';
    PUT '       <center>';
    PUT '    </td>';
    PUT '    <td width="200" valign="top">';
    PUT '    <center>';
    PUT '  <SELECT MULTIPLE name="statel"
size="3">';
    PUT '   <OPTION value="ALL">ALL STATES';
  RUN;
  /*==================================*/
  /* Populate the state pull down list  */
  /* from the STATES SAS datafile.      */
  /*==================================*/
  DATA _NULL_;
    SET &DATALIB..states;
    file _webout;
    PUT '   <OPTION value="'STATE'">'LONGNM;
  RUN;
  DATA _NULL_;
    file _webout;
    PUT '  </SELECT>';
    PUT '  </center>';
    PUT '</td>';
    PUT '</tr>';
    PUT '<tr>';
    PUT '  <td> </td>';
PUT '<td colspan="2"><INPUT type="submit"
value="Get Compliance Report">';
    PUT '  </td>';
    PUT '</tr>';
    PUT '</TABLE>';
    PUT '</CENTER>';
    PUT '</FORM>';
    PUT '</FONT>';
    PUT '</BODY>';
    PUT '</HTML>';
 RUN;
%MEND;
%Select
```

The bottom frame is initially populated by the simple Welcome.html HTML code.

When the user selects a compliance level of Non-Compliant, September as the month, and ALL STATES as the state criteria, and finally clicks on the 'Get Compliance Report' submit button, the HTML Form tag, <FORM ACTION="/cgi-bin/broker" METHOD="POST"> instructs the web server to call the Application Broker, /cgi-bin/broker while passing all of the HTML FORM fields (variables) to this CGI program.

The location of the Application Broker is defined in the ACTION attribute of the HTML FORM tag as /cgi-bin/broker. The data contained in the required field, _PROGRAM, identifies the name of the Dispatcher program, GetRpt.sas. The parameters and their associated values are contained in INPUT and SELECT fields. These fields include compl, month, and statel from the <INPUT type="radio" name="compl"...>, <SELECT name="month"...>, and <SELECT MULTIPLE name="statel" ...> HTML tags.

The radio button variable, compl, is 0, the month variable is "SEP", and the statel variable is "ALL". These are not the only parameters being passed to the broker. There are 'hidden' HTML form fields such as _prognam that was mentioned before and _debug. _prognam is used by /cgi-bin/broker to determine which Application Dispatcher program should be executed. In this case, it is GetRpt.sas in the complian Application Library.

Here is GetRpt.sas and an explanation of how it works:

```
%macro GetRpt;
  %include '/WEB4/CPAS/SOURCE/Cpas_Opts.sas';
  %curmon
  DATA _NULL_;
    call
symput('amtas0',trim(left("A"))||trim(left("&ds
"))||trim(left('A0')));
  RUN;
  %LOCAL states i j hstates;
  %GLOBAL statel0 statel1 statel;
  %IF %superq(statel)= %THEN %DO; /* NO STATEs
were selected! */
    %LOCAL statel0; /* Since the dispatcher did
not create this */
    %LET statel0=0;
  %END;
  %ELSE %IF %superq(statel0)= %THEN %DO;/* Only
one STATE  was chosen */
     %LOCAL statel0 statel1; /* Since the
dispatcher did not create these */
     %LET
statel1=%left(%trim(%superq(statel)));
     %LET statel0=1;

/*=======================================*/
/* If ALL has been selected, then create  */
/* my list of STATEs and my total number  */
/* of STATEs macro variables from the     */
/* detail SAS datafile.                   */
/*=======================================*/
     %IF "&statel"="ALL" %THEN %DO;
```

```
       /* Just use current month's AMTAS0 since
it contains ALL states */
       PROC SORT DATA=&DATALIB..&amtas0
OUT=statelis nodupkey;
         BY STATE;
       RUN;
       DATA STATELIS(DROP=MEASURE);
         SET STATELIS(KEEP=STATE MEASURE)
END=FINI;
          call
symput('STATEL'||LEFT(_N_),TRIM(LEFT(STATE)));
          IF FINI THEN call
symput('STATEL0',LEFT(_N_));
       RUN;
     %END;
  %END;
  %IF &statel0=0 %THEN %DO;
     %errstate
  %END;
  %ELSE %DO;
     %DO j=1 %to &statel0;
        %IF &j=&statel0 %THEN %DO;
           %LET
states=&states"%left(%trim(%superq(statel&j)))"
;
           %LET
hstates=&hstates%22%left(%trim(&&statel&j))%22;
        %END;
        %ELSE %DO;
           %LET
states=&states"%left(%trim(%superq(statel&j)))"
,;
           %LET
hstates=&hstates%22%left(%trim(&&statel&j))%22%
2C;
        %END;
     %END;

/*=========================================*/
/* Create a macro var for all states to be  */
/* used if  the user selects the option to  */
/* view stats related to all of her selected*/
/* states.This macro var will be url encoded*/
/*=========================================*/
     %LET WHERE= (compl=&compl AND state in
(&states));

     %analmeas(A)

     DATA FINAL(KEEP=cat state newstate compl
prod_cd);
       MERGE ANALF(IN=A) MEAS(KEEP=CAT
MEASURE);
       BY MEASURE;
       IF A;

/*====================================*/
/* Here I will put HTML anchor tags    */
/* around the STATE variable. NEWSTATE */
/* will be the final outputted variable*/
/*====================================*/
startit='<a
href="broker?_service='||"&service"||'&_debug='
||"&debugst"||'&cnttype=html&_program=complian.
MeasRpt.sas&state=';
       mid='">';
       catit='&cat=';
       complit='&compl=';
       dsit='&ds=';
       endit='</a>';
newstate=startit||trim(left(state))||catit||tri
m(left(cat))||complit||trim(left(compl))||dsit|
|trim(left(&ds))||mid||put(trim(left(state)),$s
tate.)||endit;
       LABEL
         newstate="State"
       ;
      RUN;
      PROC SORT DATA=FINAL;
        BY CAT NEWSTATE;
      RUN;

      /*============================*/
      /* Get Counts by cat and state */
      /*============================*/
      PROC MEANS NOPRINT DATA=FINAL;
         var compl;
         by cat state;
         class newstate prod_cd; /* _TYPE_ = 2
is at the prod_cd or product level */
         output out=stats (Where=(_type_=2))
                n=numcompl
         ;
      RUN;
      DATA _NULL_;
        CALL symput('c',put(&compl,compl.));
        CALL symput('fn',"t&_RMTHOST..tsv");
      RUN;
      DATA STATS;
        SET STATS;
        LABEL
          NUMCOMPL='Total'
        ;
        file "/opt/nes/docs/CPAS/&fn";
        put cat '09'x state '09'x prod_cd '09'x
numcompl;
      RUN;
      TITLE1 "Top Level Summary for &c
Measures";
      TITLE2 "by Measure Category and State";
      DATA _NULL_;
        FILE _webout;
        PUT "Content-type: text/&cnttype";
        PUT;
        PUT '<HTML>';
        PUT '<HEAD>';
        PUT '<TITLE></TITLE>';
        PUT '<BODY>';
        mid='">';
        catit='&cat=';
        complit='&compl=';
        dsit='&ds=';
        endit='</a>';

all=catit||"ALL"||complit||trim(left(&compl))||
dsit||trim(left(&ds))||mid||"here"||endit;
        PUT '<font size="+2">Click <a
href="broker?_service='"&service"'&_debug='"&de
bugst"'&_program=complian.MeasRpt.sas&cnttype=h
tml&state=' "&hstates" all ' to view ALL
metrics for ALL of your selected states.';
        PUT 'Otherwise click on any specific
state below.</font>';
        PUT '<FORM ACTION="/cgi-bin/broker"
METHOD="POST" TARGET="main">';
        PUT '<INPUT TYPE="HIDDEN"
NAME="_service" value="&service">';
        PUT '<INPUT TYPE="HIDDEN"
NAME="_program"
value="complian.Download.sas">';
        PUT '<INPUT TYPE="HIDDEN" NAME="_debug"
value="&debugst">';
        PUT '<INPUT TYPE="HIDDEN" NAME="ds"
value="&ds">';
        PUT '<INPUT TYPE="HIDDEN" NAME="compl"
value="&compl">';
        PUT '<INPUT TYPE="HIDDEN" NAME="statel"
value="ALL">';
        PUT '<INPUT TYPE="HIDDEN"
NAME="statel0" value="">';
        PUT '<INPUT TYPE="HIDDEN" NAME="month"
value="&month">';
        PUT '<INPUT TYPE="HIDDEN"
NAME="cnttype" value="">';
        PUT '<CENTER>';
        /*PUT '<INPUT type="image"
src="/CPAS/images/download.gif" border=0>';*/
        PUT '</CENTER>';
        PUT '</FORM>';
PUT '<a href="/CPAS/'"&fn"'"><img
src="/CPAS/images/download.gif" border="0"
height="31" width="111"></a>';
        PUT '</FONT>';
```

```
        PUT '</BODY>';
        PUT '</HTML>';
      RUN;
      %footer(p)
      %DS2HTM(data=stats,
              var=newstate,
              id=cat,
              sum=numcompl,
              openmode=append,
              htmlfref=_webout,
              runmode=s,
              bgtype=color,
              bg=white,center=y,
              twidth=50,
              tcolor=#003333,
              tface=arial,
              tsize=+2,
              clbgcolr=yellow,
              clcolor=brown,
              clface=arial,
              clsize=+1,
              ibgcolr=#ccccff,
              encode=n,
              ftag=bold + small);
  %END; /* End of No State Selected ELSE DO
Check */
%MEND;
%GetRpt
```

the integration of the Web Publishing Tools and SAS/IntrNet comes into play.

HTMLFREF is set to _webout since all text output produced within an Application Dispatcher program should be written to the Web browser via the fileref _webout.

When the user clicks on the 'Get Compliance Report' submit button from index.html, the HTML FORM request is sent to the web server with all of its parameters which then causes the broker CGI program, /cgi-bin/broker, to execute. The broker communicates with the Application Server which knows to execute the SAS program which is passed as_PROGNAM. In this case, _PROGNAM=GetRpt.sas. The results of GetRpt.sas is finally presented back to the user browser as a summary of how many entries are Non-Compliant for the month of September by state in the permanent SAS dataset as illustrated below.

First we merge the temporary SAS datasets, ANALMEAS and MEAS which were generated by the analmeas macro. The curmon macro was called to perform a subset where month="SEP".

Next we invoke the MEANS procedure in order to get summary counts.

The **CALL symput('c',put(&compl,compl.));** statement creates a macro variable, &c which will be "Non-Compliant" in this case since &compl=0. The format value is "Non-Compliant" when compl=0 in the permanent SAS format library. It will be used simply to render a self-explanatory title to the user.

Here you will notice that we are using the Dataset Formatting macro, %ds2htm, of the Web Publishing Tools. Here's where

There are many Application Dispatcher programs contained within CPAS other than GetRpt.sas. If a visitor drills down by State in the "Top Level Summary...", then MeasRpt.sas is executed. MeasRpt.sas displays all measures or PIDs out of compliance for the selected state. The next level is the CLEC level. The drill levels continue all the way to the actual detail data.

Special consideration had to be made to ensure that a visitor would not be presented with 1000's of detail records! There is a limitation to how much a browser can take! The visitor is able to page through the detail records 50 records at a time.

CONCLUSION

USWest® is now able to quickly take corrective action when the data demonstrates that a CLEC is falling outside of the acceptable compliance level for specific measures. In addition, they can provide compliance reports to regulatory authorities as well as to internal staff personnel on a 'need-to-know' basis.

This will contribute tremendously to USWest's goal to gain access to the lucrative long distance market. CPAS is a tool which allows USWest to demonstrate to State and Federal utility authorities that USWest's business practices allow CLECs to fairly compete across the 14 USWest territories.

The Web Publishing Tools in conjunction with SAS/IntrNet are powerful strategic tools. They are robust enough in their complexity to support mission critical systems and simple enough to allow SAS developers to quickly create dynamic, interactive applications throughout an organization without requiring that the viewers have SAS Software installed on their machines.

CONTACT:

FRS Associates, LLC
2750 Market Street, Suite 101
San Francisco, CA 94114-1987
415.626.9796
http://www.frsa.com/, faith@frsa.com

References

SAS Institute Inc,(1998), *"SAS/IntrNet™ Software: Delivering Web Solutions", CD-ROM*

SAS Institute's Web Tools section of their web site - *http://www.sas.com/web/*

Sloan,F.R., (1997), "Developing a PC-SAS® World Wide Web Database System," *Proceedings of the Twenty Second Annual SAS Users Group International Conference.*