

The SAS System Enters the Third Dimension

Using DATA Step Views for Real-time Quality Control

Jeffrey A. Polzin
Robert N. Rodriguez
James F. Sattler

ABSTRACT

The SAS System is developing rapidly and in many ways as it approaches the 21st century. This paper deals with one set of quantitative changes and shows how these changes, in combination, usher in a new qualitative change. As a result of new developments at the supervisor, procedure, and host levels, the SAS System is moving beyond batch and interactive processing capabilities and into a new, third dimension of real-time and event-driven processing. This opens up entirely new vistas for applications developers.

INTRODUCTION

The specific application discussed in this paper is real-time statistical process control, which is used to detect and reduce excessive variation in manufacturing processes. For example, consider a process that fills tubes of prescription skin cream. Overfilling the tubes results in a profit loss, while underfilling is prohibited by government regulations. In general, failure to maintain statistical control leads to poor product quality, waste of material, and customer dissatisfaction, as well as loss of profit.

In modern manufacturing locations, automated systems for inspection and data collection make it possible to measure multiple characteristics on every unit that is produced. In principle, a wealth of data are available for detecting out-of-control conditions and adjusting the process on a real-time basis. In practice, however, the software for analyzing process data is not as sophisticated as the data-collection facilities, and process control at many sites continues to be based on relatively infrequent sample measurements and primitive statistical methods.² In the case of the tube-filling process, tubes are sampled only every two hours, whereas they are filled at the approximate rate of one per second. Consequently, the ability to detect excessive variation in real time represents significant potential savings.

The procedures in SAS/QC software provide a wealth of tools for process data analysis. However, their application to real-time process control is limited by the fact that SAS procedures, in general, operate on fixed data sets rather than continuous streams of data (of course, this limitation depends on the definition of "real-time".) Consequently, support for real-time processing in SAS/QC software requires fundamental changes in the SAS System.

In order to understand how this evolution is occurring, it is necessary to look at individual components in the three levels of the SAS System.

CHANGE I - SUPERVISOR LEVEL - DATA STEP VIEWS

Views first evolved within relational database system software products, as part of structured query language (SQL). SQL is a language used to access and query data stored in tables. SQL is also used to create new tables formed from the syntax specified while accessing or querying the data.

Data in tables must be stored physically, whereas SQL views store only SQL access and query statements. Views do not contain data but rather only the description of how the data are to be accessed and queried. SQL views improve resource management.

SQL processing was incorporated into Version 6 of the SAS System to provide applications developers with new tools. However, data descriptions based on SQL syntax are somewhat limited. SQL was never intended to be a full programming language. At most it can be used as an embedded language. Therefore, although the SAS System benefits greatly from the use of SQL-based views, such views have inherent functional limitations.

Fortunately for applications developers, as SQL was being incorporated into the SAS System, the Institute began to develop another kind of view based on the syntax of the SAS DATA step. DATA step views were first released in 1990, in an experimental form.³

Today, with the 6.07 release of the SAS System, a production version of DATA step views is available, albeit with some features that are still considered experimental.⁴

Conceptually, a DATA step view is a SAS data set of the type VIEW. Like other SAS data views, such as those created with the SQL procedure, it does not actually contain data values. Instead, it contains a stored DATA step program that defines data or describes data stored elsewhere. SAS DATA step views preserve the resource management functionality of SQL views, while adding considerable power to descriptive view definition logic.

The development of views means that the DATA step has several new roles to play in the SAS System. First, as has just been pointed out, the DATA step now has the added power of being able to create a view, not just an intermediate SAS data file. Second, since an intermediate data file is no longer a prerequisite for processing, DATA step views play a new role of improving resource management by reducing the need to store SAS data files. The third role of the DATA step view is to act as a conduit, sending a stream of data directly from an input source to a procedure or other DATA step. The fourth role follows logically from the conduit role. If data can be sent directly to a process in the form of a stream, why not also send signals to the procedure at the same time?

Asking that question meant answering it. SAS Institute developers enhanced DATA step views to send special signals, in the form of return codes, to a procedure. Thus the DATA step's fourth role is to communicate directly with view-compatible procedures. A view-compatible procedure can be a modified existing procedure or a new procedure.

CHANGE II - PROCEDURE LEVEL

While core developers were hard at work making these changes at the supervisor level, procedure developers began to make corresponding changes and enhancements at the procedure level. Let us turn our attention now to this subject.

To take advantage of the DATA step's new capabilities, SAS Institute developers recognized that certain procedures have to become event-driven. The SHEWHART procedure in SAS/QC software was selected as an initial candidate for this modification. Many SAS/QC users have requested a real-time version of this procedure, so it was a natural starting point to test the new functions of DATA step views.

The SHEWHART procedure creates control charts which are used to detect unusual patterns of variation in output from manufacturing and other processes. A control chart displays summary statistics (such as means and ranges) plotted over time, and the display is superimposed with control limits that quantify the natural variation in the process. A point outside the control limits or an unusual pattern of points signals that the process is out of control and that corrective action may be needed.

In real-time process control applications, timely notification of problems is critical. For instance, in order to react effectively to an out-of-control condition, it may be necessary to take a new measurement sample (referred to as a subgroup sample) every five seconds and update the control chart at this rate.

This requires basic modifications in the SHEWHART procedure. First, the procedure needs to be able to receive and process a return code from a view specifying a particular update mode. In other words, the procedure has to be able to act on a continuous stream of data—as opposed to a fixed SAS data set—in one of several ways (modes) indicated by the return code. For example, one particular return code might instruct the procedure to accumulate the data, whereas another return code might instruct the procedure to process a moving window of the data.

Second, the SHEWHART procedure has to provide new options for the specifics of a particular processing mode. For example, to support the *moving window mode* an option is needed to define the size (number of samples) of the window. Likewise, the procedure has to provide additional flexibility for terminating or resetting the processing of the data. In the *accumulation mode* the optional alternatives might be to continue processing until there are no more data, until an out-of-control condition is detected, or until (say) 200 samples are accumulated (in which case accumulation starts anew).

Finally, it is important that the procedure's graphics capabilities be improved so that it can keep up with the speed of the incoming data stream. This requires augmenting the current, batch-oriented graphical routines with dynamic graphics support so that only those features of a chart that change are updated. Greater speed is not the only objective of this enhancement; a related goal is to enable the operator to click on an out-of-control point and open a window, either to add a comment or to obtain related information.

A new experimental procedure, PROC TSHEWHART, that incorporates the first two types of enhancements was developed in the spring of 1992 for demonstration at this conference.

CHANGE III - HOST LEVEL

In addition to changes at the supervisor and procedure levels, host-level changes are needed to facilitate real-time process control. More likely than not the data stream will be collected directly from a device on the shop floor. To process this stream new engines and data-collection device-drivers are needed.

An interface to DEC's BASEstar product for VMS is currently being developed to allow the SAS System to receive and analyze real-time data. Access to BASEstar is provided by utilizing the SAS BASEstar database server, initially provided as part of the SAS/BASE product. The database server acts as a data repository for information

that BASEstar has collected based upon user specifications of device configurations, information concerning the format of the data samples from devices, designation of repository placement, etc. SAS System users can gain access to the data held by the SAS BASEstar database server through the use of the SAS I/O BASEstar engine, initially provided as part of the SAS/BASE product. This engine is not much different from other SAS I/O engines except that it is specifically designed to communicate with the SAS BASEstar server to service requests to provide data or information from specifically named BASEstar server entries.

Interaction between the BASEstar product and the SAS BASEstar server can be accomplished using the BASEstar DCL (DEC Command Line) or DECWindows interfaces to describe data to be processed by the SAS BASEstar server or by a specific user application using the BASEstar system call library. To describe the data to the SAS BASEstar server, one must first describe all the devices from which information is to be sampled. An extensive device support library is available as part of the BASEstar product. Once the devices have been selected, data specifications must be defined to describe the structure of the data to be sampled. In BASEstar terminology these are called physical points, actual device samples, and logical points, references to other physical and logical points or expressions. Once the points have been defined they are combined to form an external map or record definition. When the external map is defined the SAS BASEstar server destination can be specified as the repository site. This procedure is analogous to a SAS user defining a library and engine association along with the variable definitions of a SAS data set member in the library.

The implementation of the SAS BASEstar server and I/O engine is based upon a CLIENT-SERVER model. Communication between the two components is accomplished using three access methods: DEC MAILBOX for intramachine access, DECNET for intermachine access, or TCP/IP utilizing SOCKETS for intermachine access across a heterogeneous machine network. The TCP/IP interface allows users in a heterogeneous machine network to process data collected on a DEC system and process that data on a non-DEC machine. The SAS BASEstar server and I/O engine provide READ-ONLY access to BASEstar data supporting the BOOLEAN (8 bits), WORD (16 bits), LONG (32 bits), FLOAT, DOUBLE, CHARACTER (maximum character length of 200), and PACKED DECIMAL BASESTAR data types.

EXAMPLES

In the following examples we demonstrate the evolution of the current usage for the TSHEWHART procedure in relation to real-time quality control capabilities. The first example begins shows how real-time process control is currently implemented for most environments using SAS/QC software. The second and subsequent examples demonstrate extensions for real-time process control using DATA step views to control and interact with the TSHEWHART procedure.

Prototype Example - I

In this example, the SAS DATA Step creates an intermediate SAS data file called FILLWTS. After the file has been completely built, the procedure reads it and outputs a control chart for averages.

```
/*          SUGI EXAMPLE I          */

filename shew 'fillwt.dat';

/* The mylim data set provides a set of pre-established */
/* control limits to be displayed on the control charts */
/* that are created by the examples                      */

data mylim;
    length _var_ $ 8 _subgrp_ $ 8 _type_ $ 8;
```

```

_var_      = 'fillwt';
_subgrp_   = 'sample';
_type_     = 'standard';
_limitn_   = 5;
_sigmas_   = 3;
_lclx_     = 240.175;
_mean_     = 257.133;
_uclx_     = 270.092;
_lclr_     = 0;
_r_        = 29.4;
_uclr_     = 62.166;
_stddev_   = 12.64;

output;
run;

data fillwts;
  infile shew;
  sample+1;
  do i=1 to 5;
    input fillwt @;
    output;
  end;
  label fillwt='Fill Weight'
        sample='Sample Index';
  drop i;
run;

title1 '_';
title2 'X Chart for Fill Weights';

proc tshewhart data=fillwts limits=mylim;
  xchart fillwt*sample='*';
  outtable=warning(keep=_exlim_ sample) readlimits;
run;

```

Prototype Example - II

In this example, the SAS DATA Step builds an input view called FILLWTS. The procedure reads data as a stream but waits for the entire stream to finish before processing.

```

/*          SUGI EXAMPLE II          */

filename shew 'fillwt.dat';

/* data mylim as in Example I; */

data fillwts/view=fillwts;          /*INPUT VIEW*/
  infile shew;
  sample+1;
  do i=1 to 5;
    input fillwt @;
    output;
  end;
  label fillwt='Fill Weight'
        sample='Sample Index';
  drop i;
run;

title1 '_';
title2 'X Chart for Fill Weights';

proc tshewhart data=fillwts limits=mylim;
  xchart fillwt*sample='*';
  outtable=warning(keep=_exlim_ sample) readlimits;
run;

```

Prototype Example - III

In this example, the SAS DATA Step input view called FILLWTS now has the additional capability of defining a return code signal. The procedure also has additional capabilities of specifying a real-time processing mode.

```

/*          SUGI EXAMPLE III          */

filename shew 'fillwt.dat';

/* data mylim as in Example I; */

data fillwts/view=fillwts;

```

```

  retain count 0;
  infile shew;
  sample+1;
  do i=1 to 5;

    input fillwt @;
    output;
  end;
  label fillwt='Fill Weight'
        sample='Sample Index';
  count+1;
  if count = 15 then do;
    output fillwts rc=$SYSRC(_SWPAUSL);    /*RETURN CODE*/
  end;
  drop i;
run;

/* Titles as in Examples I and II */

proc tshewhart data=fillwts limits=mylim
  rtmode=window(15,1);    /*REAL-TIME MODE*/
  xchart fillwt*sample='*';
  outtable=warning(keep=_exlim_ sample) readlimits;
run;

```

Prototype Example - IV

In this example, an output DATA Step view has been created to read the WARNING data set that the TSHEWHART procedure outputs. The log shows the results of the real-time processing.

```

/*          SUGI EXAMPLE IV          */

filename shew 'fillwt.dat';

/* data mylim as in Example I; */

/* data fillwts/view=fillwts as in Examples II and III; */

data test/view=warning;    /* OUTPUT VIEW */
  set warning;
  when=time();
  if _exlim_='UPPER' then
  do;
    put 'UPPER CONTROL LIMIT EXCEEDED AT ' when time!2.8
      ' FOR SAMPLE ' sample;
  end;
run;

title1 '_';
title2 'X Chart for Fill Weights';

proc tshewhart data=fillwts limits=mylim
  rtmode=window(15,1);
  xchart fillwt*sample='*';
  outtable=warning(keep=_exlim_ sample) readlimits;
run;

NOTE: The infile SHEW is:
      File=$DUS$00:|BIO_MK1.RJPS.QCSAMP|FILLWT.DAT
NOTE: This is experimental version 1.0 of the SHEWHART procedure
      for demonstrating real-time processing.

UPPER CONTROL LIMIT EXCEEDED AT 8:31:21.9800 FOR SAMPLE 5
UPPER CONTROL LIMIT EXCEEDED AT 8:31:22.2100 FOR SAMPLE 13
UPPER CONTROL LIMIT EXCEEDED AT 8:31:22.2200 FOR SAMPLE 14

NOTE: 21 records were read from the infile SHEW.

```

Prototype Example - V

In this example, the input DATA Step view reads directly from a device using the SAS BASEstar I/O engine and SAS BASEstar database server.

```

/*          SUGI EXAMPLE V          */

/* data mylim as in Example I */

libname device BASESTAR 'host name';    /*BASESTAR ENGINE*/

```

```

data fillwts/view=fillwts;
  retain count 0;
  set device.show;          /*DATA READ WITH SET STATEMENT*/
  sample+1;
  do i=1 to 5;
    output;
  end;
  label fillwt='Fill Weight'
        sample='Sample Index';
  count+1;
  if count = 15 then do;
    output fillwts rc=$SYSRC(_SNPAUSL);
  end;
  drop i;
run;

data test/view=warning;
set warning;
when=time();
if _exlim='UPPER' then
do;
  put 'UPPER CONTROL LIMIT EXCEEDED AT '   when time12.8
    ' FOR SAMPLE ' sample;
end;
run;

title1 ' ';
title2 'X Chart for Fill Weights';

proc tshewhart data=fillwts limits=mylim
  rtmode=window(15,1);
  xchart fillwt*sample=**/
  outtable=warning(keep=_exlim sample) readlimits;
run;

```

DATA STEP VIEW PROCESSING ADVANTAGES

The prototype examples just discussed illustrate how a number of multi-level, and initially disparate, changes and enhancements to the SAS System combine to produce a qualitatively new kind of application.

These techniques offer distinct advantages that need to be made explicit.

First, the syntax of DATA step views is richer than the syntax of SQL views. Even without the addition of return codes, a SAS DATA step view can use the vast power of the SAS DATA step language to manipulate and manage input data. SQL, in contrast, was never intended to be a full-fledged programming language, but only a data access tool. Second, the logic of a DATA step view is stored by the SAS Stored Program Facility in compiled form. That means that the SAS System offers developers the ability to build views that are a compiled organic hybrid of access statements and procedural logic. This is a major software advance.

FUTURE DEVELOPMENTS

Running the TSHEWHART procedure against a DATA step view marks the first step in a new, evolutionary path for the SAS System. Further developments are likely at all three levels discussed above.

At the supervisor level, new return code signals will be introduced to communicate with other procedures.

At the application level, more procedures will become view-compatible. Among current SAS/QC procedures, the following are the most likely candidates for view-compatibility:

- the CAPABILITY procedure for analyzing the capability of an in-control process. Here a typical real-time display might consist of a histogram, superimposed with specification limits, whose bins and bars change as data are accumulated. A typical outcome might be a mail message

indicating that the process is no longer capable of meeting the specifications.

- the CUSUM procedure which creates cumulative sum control charts
- the MACONTROL procedure which creates uniformly weighted and exponentially weighted moving average control charts
- the PARETO procedure which creates Pareto charts

In addition to providing view-compatibility, SAS/QC procedures will need to implement algorithms that are more appropriate for real-time process control. The Shewhart approach, which was introduced in the 1920's, assumes that process measurements are sampled rather than collected exhaustively; that the process mean is stable and impacted only rarely by special causes of variation; and that it is expensive to tune the process continually. To the extent that these assumptions are reasonable in real-time applications, the Shewhart chart will continue to be useful.

However, better strategies are available for situations that require feedback control. For instance, if autocorrelation is observed in the process mean, then time-series (ARIMA) models for the process dynamics provide the basis for feedback controllers that minimize deviation from a specified target.¹ Many other strategies have been proposed for real-time applications.² Although a survey of these methods is beyond the scope of our discussion, the point to be made here is that the evolution of real-time process control in the SAS System will mandate changes in analysis as well as processing.

Finally, at the host level, new "engines" will become available for data-collection devices. Development will continue to enhance the capabilities of the experimental SAS BASEstar data base server and I/O engine as requested from the experimental test sites. A future SAS product specifically designed to deal with real-time data collection and analysis is currently in early design stages. This product will be supported across all machines platforms for which it is technically feasible. This product will be able to provide device data definition and management along with extensive analysis and visualization capabilities. If you have suggestions for product coverage or capabilities, please contact Jeffrey A. Polzin at SAS Institute, Inc.

CONCLUSIONS

The SAS System is evolving at three different levels in a way that has opened the door to a new kind of processing. In the 1970's, the SAS System was used almost entirely for batch applications. In the 1980's, with the advent of SAS/FSP and other products, the SAS System facilitated the development of sophisticated interactive applications. Now, with the advent of DATA step views, view-compatible procedures, and new engines, the SAS System has entered a third dimension of processing. Today, only the dim contours of this new three-dimensional world are discernible in ways that affect the product itself and applications. In the product, for example, the lines of demarcation between the DATA step and PROC step will become blurred.

The SAS System's event-driven capabilities will make it possible to build an entirely new range of applications in the years ahead. In particular the role of the SAS System in manufacturing automation applications should increase⁵, and one can speculate that SAS/QC procedures will be able to reset input parameter settings automatically, or shut down an out of control process.

Other application areas where event-driven processing could be employed include forecasting using SAS/ETS procedures, laboratory automation, and data-entry/validation/reporting systems. These new vistas will become clearer as soon as more event-driven products become available for experimentation.

One thing is clear. The real-time example using the experimental TSHEWHART procedure shows that the SAS System has evolved beyond batch and interactive processing dimensions. The SAS System has entered the third dimension of real-time, event-driven processing.

The TSHEWHART procedure, SAS BASEstar database server and I/O engine will become available for limited experimental use beginning June 1992. If your site is interested in becoming an experimental site for these features please contact Jeffrey A. Polzin at SAS Institute Inc, Cary NC.

AUTHORS

The authors of this paper may be contacted at :

Jeffrey A. Polzin
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
Phone (919) 677-8000
FAX (919) 677-8123

Dr. Robert N. Rodriguez
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
Phone (919) 677-8000
FAX (919) 677-8123

Dr. James F. Sattler
Syntex Research
MS A4-200
3401 Hillview Ave.
Palo Alto, CA 94303
Phone (415) 852-3016
FAX (415) 496-3616

TRADEMARKS

SAS, SAS/BASE, and SAS/QC are registered trademarks of SAS Institute Inc. in the USA and other countries. * indicates USA registration.

IBM is a registered trademark of International Business Machines Corporation.

DEC is a trademark of Digital Equipment Corporation.

VMS is a trademark of Digital Equipment Corporation.

BASEstar is a trademark of Digital Equipment Corporation.

Other brand and product names are registered trademarks or trademarks of their respective companies.

REFERENCES

1. Harris, T. J. (1985). "Optimal stochastic control" in Encyclopedia of Statistical Sciences, Volume 6, edited by S. Kotz and N. L. Johnson. New York: John Wiley & Sons. pp. 459-464.
2. Keats, J. B. (1989). "Process Control in Automated Manufacturing: Some Key Issues" in Statistical Process Control in Automated Manufacturing, edited by J. B. Keats and N. F. Hubele. New York: Marcel Dekker. pp. 17-31.
3. Polzin, J.A. (1991). "An Insider's Sneak Preview of the Next DATA Step Release", in SUGI 16 Proceedings, pp. 1041-1047.
4. Polzin, J.A. (1992). "DATA Step Views", in SUGI 17 Proceedings.
5. Sattler J. F. & Sharlin J. (1985) "The Current and Potential Role of the SAS System in Industrial Applications", in SUGI 10 Proceedings, pp. 407-412.