EXECUTIVE INFORMATION SYSTEMS USING SAS/AF SOFTWARE

INTRODUCTION

The problem of presenting current and useful information to executives is a long-standing one. Executives require almost immediate data on the performance of the company in all major areas, as well as status information on the progress being made in carrying out corporate strategic plans.

At GTE Data Services (GTEDS), the data-gathering problem is further complicated by the necessity of collecting information from ten data centers around the country. The data is collected daily through our “backbone” network, and must then be arranged in a format that will provide top-level management with easy access to the information they need.

The Executive Work Station (EWS) is the solution to these business needs. It is based on an umbrella concept, creating a framework into which any number of applications can be plugged. In addition, access to existing systems, such as Telemail and PROFS®, can be easily added. Within EWS, we have developed the Strategic Program Management system (SPM), and are now developing a replacement for a turnkey stand alone Executive Information System.

The use of SAS/AF® software for new mainframe applications, and SAS/GRAPH® for graphic capabilities, gives the executive easy access to required information, without generating masses of irrelevant data to wade through. The use of an index file and macros provides the method for changing viewing requests as the data changes. This paper covers the requirements of an Executive Work Station, the basic architecture and design that we chose, the PC-based EWS driver program, and a more detailed discussion of the first application, the Strategic Program Management (SPM) system. It also covers some specific problems that came up during the development of the system, and how I used the capabilities of the SAS® System to solve them.

REQUIREMENTS OF AN EXECUTIVE WORK

Several requirements for an Executive Work Station stand out. The system must be “executive-friendly”, that is, the use of the system cannot require knowledge of data processing details. Such items as logons for a given application, user IDs, and CLIST names must all be handled by the EWS driver. Within any application developed for EWS, single-key selection is the method of choice, and any other keying must be kept to an absolute minimum.

Any application included in the EWS must also be “executive-proof”, i.e., designed so that the user cannot accidentally exit the application. PF keys must be definable by the developer to prevent problems. The EWS driver program must be able to take over in the event of accidental (or purposeful) exit from the application.

Another requirement for applications developed under EWS is flexibility. The system must be flexible enough to change the presentation as executive information needs change.

The majority of the data provided to the executives is best seen in graphic representation. Graphics capability is therefore a heavily weighted requirement in the selection of tools.

ENVIRONMENT

Since many of the executives at GTEDS are already using IBM PC/XT® s, any machine chosen for EWS had to be capable of running all standard IBM PC software. The IBM 3270 PC®, our eventual choice, will handle any IBM-compatible software that is well-behaved. This category includes most standard business applications, though some special versions may be required.

Mainframe communications were a primary requirement for the equipment. From any mainframe in the GTEDS backbone network, any other mainframe in the network can be accessed, as well as a number of Telnet applications, including Telemail. The IBM 3270 PC provides this communications capability.

Given the earlier requirement of “executive-friendly”, we needed to use the EWS driver program as a complete session controller. With the IBM 3270 PC, we were able to write an application program using IBM's High-Level Language Application Program Interface® (API), thus taking advantage of the 3270 PC Control Program® facilities.

The IBM 3270 PC package that makes up EWS includes a Program Symbols board for host graphics, as well as an all-points-addressable board for PC graphics.

EWS DRIVER DESIGN—PC PROGRAM

The EWS driver program chooses the mainframe depending on the application selected, and completes the logon procedure using the appropriate user ID. There is a rotation system of passwords for security, and each copy of the program is tailored to the executive's needs. A sample EWS main menu is shown in Figure 1.

Using API, the EWS driver can send all required keystrokes to the mainframe to allocate files and invoke the application while the user is seeing a “Please Wait” panel. Once the application is ready, the user is switched to that screen. The “Please Wait” message is shown in Figure 2.

The EWS driver program continually monitors the use of the application system. When the user exits the application, the screen is changed back to the EWS menu while mainframe cleanup and logoff are proceeding. During this time, the program locks the keyboard for the mainframe session, preventing any interference with the procedures (accidental or otherwise). The wrap-up panel is shown on Figure 3.

After observing several executives trying out a prototype system, I also added a “mother hen” feature to the monitoring. The greatest difficulty with our prototype system was the user's tendency to select another option with a PF key before the previous option had completed, thus locking the keyboard. Instruction on use of the “reset” key did not go well. The “mother hen” feature now constantly monitors the mainframe session for the locked keyboard conditions, and immediately sends the keystroke for a reset to the mainframe.

MAINFRAME APPLICATIONS—TOOLS

After reviewing the requirements for the Executive Work Station, we decided to use the SAS System as our major application development tool.

The ability to specify a user profile for PF keys solved a large part of the “executive-proof” requirement. If there are any functions that should not be allowed, the key definition is removed. This ability (and even more, the use of PF keys in SAS/AF screens) also helped solve the “executive-friendly” requirement.

The use of program screens in SAS/AF helped to give the system the required flexibility. The executive can key in selection criteria to limit the data presented.

413
SAS/GRAPH provided the required graphics capability.

**STRATEGIC PROGRAM MANAGEMENT SYSTEM**

The first application to be included in the Executive Work Station, the Strategic Program Management system (SPM), is a menu-driven system built with SAS/AF Software. The data for the system are a set of plans, consisting of a hierarchical structure of strategies, programs, milestones, and activities. All levels have an owner or person responsible at that level. Activities have a status code, an estimated completion date, a revised completion date, and comment text fields. A list of the hierarchy and associated data elements is in Figure 4.

At the beginning of the year, when the new strategic plans are issued, the menus are built for the detailed display portions of the system. These provide an easy method for paging through the hierarchy of the plans. An example of the menus is in Figure 5.

One of the first problems we came across was the lack in the SAS/AF software of the ability to update text screens in batch. We needed this ability because the status of an activity, its date and its comment fields can be changed daily. We were faced with the prospect of manually updating the status information on up to 300 activity screens per day.

We solved this problem by using a SAS program to write a SAS program that executed PROC GSLIDE. This program incorporated the new data from a file that was maintained with PROC FSEDIT under SAS/FSP®. The individuals responsible can enter the information for their activities during the day using PROC FSEDIT, and a batch job run overnight replaces the GSLIDE screens in the graphics catalog. The GSLIDE procedure does not have a provision for assigning unique names to the entries in the graphics catalog, so we included a PROC GREPLAY at the end of the job to assign names. The screen used for updating status information is shown in Figure 6, and an example of the code used for creating the slides is in Figure 7.

Our second problem: in order to access the graphs that we had created with PROC GSLIDE, we needed a method of identifying and accessing them without the user having to know the names we used, or being presented with a very long list to choose from.

The solution to this problem was the creation of "index" files on a SAS library, with variables for the screen name, the ID of the Strategy, Program, Milestone, and Activity represented, and the owner, status code, and estimated completion date—all of which are used as selection criteria for viewing. By using these files, the executive can restrict the screens that are displayed (e.g., only those in his/her area of responsibility, or only those that have been given a status code that indicates a problem). An example of the code used with this index file is shown in Figure 8.

The third area addressed was the problem of utilizing text fields in a variety of screen formats. This problem was solved by having the data entry people enter only whole words (no hyphenation) in the original 50-byte long fields, and writing a textflow macro that would reformat this text into whatever length was required in the screen being created. An example of this code is shown in Figure 9.

**CONCLUSIONS**

The EWS system is an easy-to-use, menu-driven system that provides the executive user with essential information on a timely basis.

The use of SAS/AF Software menus and SAS/GRAPH screens gives us the ability to provide our executives with an easy way to access data needed to better perform their jobs.

The system is flexible enough to allow for individual tailoring and future growth, and will become the gateway for the majority of the executive information systems in the future.

SAS, SAS/GRAPH, and SAS/FSP® are registered trademarks of SAS Institute Inc., Cary, NC, USA.

SAS/AF is a trademark of SAS Institute Inc., Cary, NC, USA.

IBM PC/XT and IBM 3270 PC are registered trademarks of International Business Machines Corporation.

IBM High-Level Language Application Program Interface, 3270 PC Control Program and PROFS® are trademarks of International Business Machines Corporation.
<table>
<thead>
<tr>
<th>SPM HIERARCHY LEVEL</th>
<th>DATA ELEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy</td>
<td>Strategy ID</td>
</tr>
<tr>
<td></td>
<td>Strategy Owner</td>
</tr>
<tr>
<td></td>
<td>Strategy Text</td>
</tr>
<tr>
<td>Program</td>
<td>Strategy ID</td>
</tr>
<tr>
<td></td>
<td>Program ID</td>
</tr>
<tr>
<td></td>
<td>Program Owner</td>
</tr>
<tr>
<td></td>
<td>Program Text</td>
</tr>
<tr>
<td>Milestone</td>
<td>Strategy ID</td>
</tr>
<tr>
<td></td>
<td>Milestone ID</td>
</tr>
<tr>
<td></td>
<td>Milestone Owner</td>
</tr>
<tr>
<td></td>
<td>Milestone Text</td>
</tr>
<tr>
<td>Activity</td>
<td>Strategy ID</td>
</tr>
<tr>
<td></td>
<td>Milestone ID</td>
</tr>
<tr>
<td></td>
<td>Milestone Owner</td>
</tr>
<tr>
<td></td>
<td>Milestone Text</td>
</tr>
</tbody>
</table>

![Figure 4. SPM Data Elements](image)

Select Option ***

<table>
<thead>
<tr>
<th>Opt</th>
<th>Strategy</th>
<th>Owner</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PF1</td>
<td>A</td>
<td>Smith</td>
<td>Text of Strategy A</td>
</tr>
<tr>
<td>PF2</td>
<td>B</td>
<td>Jones</td>
<td>Text of Strategy B</td>
</tr>
<tr>
<td>PF3</td>
<td>C</td>
<td>Scott</td>
<td>Text of Strategy C</td>
</tr>
<tr>
<td>PF4</td>
<td>D</td>
<td>Smith</td>
<td>Text of Strategy D</td>
</tr>
<tr>
<td>PF5</td>
<td>E</td>
<td>Williams</td>
<td>Text of Strategy E</td>
</tr>
<tr>
<td>PF6</td>
<td>F</td>
<td>Brown</td>
<td>Text of Strategy F</td>
</tr>
<tr>
<td>PF7</td>
<td>G</td>
<td>Greene</td>
<td>Text of Strategy G</td>
</tr>
</tbody>
</table>

PF11 Help

PF12 Return to Main Menu

![Figure 5. Sample SPM Menu](image)

Command ***

<table>
<thead>
<tr>
<th>Command</th>
<th>Edit SRQ data set: SPM.STATFILE</th>
<th>Screen</th>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Strategies: __________ Program: __________ Milestone: __________
Activities: __________
Status Code: __________
Revised Completion Date: __________
Comments: __________

PF7-Prev  PF8-Next  PF9-Add Another  PF12-Edit

![Figure 6. Status Update Screen](image)
**Figure 7. Code For Generating PROC GSLIDE Code**

```
PROC DISPSON:
 LENGTH NAME # 0:
 LENGTH PROS # 1:
 DO UNTIL (STRA = "S" AND PROS = "P" AND MILE = "M"):
   SET LUNB.ACTIV: END:
   IF STRA = "S" AND PROS = "P" AND MILE = "M" THEN DO:
     PROS = ":1
     OUTPUT:
   END:
   IF PROS = " " THEN DO:
     NAME = "SPINDLES":
     OUTPUT:
   END:
   STOP:
DATA XULL:
SET TEST1;
FILE KEYFD-JD;
CLEAR:
PARTO "DEVP-ED:BM32702 DISPLAY;":
PART1 = "PROC GREPLAY MDUT" SCR.SPMSCRAA NOPR":
PART2 = "";
PART3 = ":1
PUT 0.
PUT 0.
PUT 0.
PUT 0.
STOP:
DATA XULL:
"EXCLUDE KEYFD:
SPEND DISPSON;":
```

**Figure 8. Code Used With Index File**

```
%macro textflow(p��eria):
   li = li + 1;
   ctr = ctr + 1;
   do while(substr(totalstr.1.pos.1) = " ");
   pos = pos + 1;
   if pos = li then do;
     put 'word too long ' totalstr.1.pos.1+1;
     li = li + 1;
     pos = li + li + 1;
   end;
   end:
   li = li + 1;
   line = substr(totalstr.1.pos.1);
   substr(totalstr.1.pos.1) = ";
   newstr = left(totalstr.1-pos);
   totalstr = newstr;
   %endtextflow:
   FORMAT activitiy description:
   "|\textbf{TextFlow}()|" |"||\textbf{TextFlow}()||"|\textbf{TextFlow}()|"
   FORMAT measure of achievement description:
   totalstr = measure;
   \textbf{TextFlow}() line04 = line04;
   \textbf{TextFlow}() line05 = line05;
```

**Figure 9. Text Flow Macro And Its Use**