Paper 1

OUT OF THE MAINFRAME & INTO THE WEB: THE TRIALS AND TRIBULATIONS OF PORTING SAS MAINFRAME APPLICATIONS TO THE WEB

Thomas E. Link, U.S. Environmental Protection Agency, RTP, NC
Thomas A. Dessent, Lockheed Martin Corporation, Contractor to US EPA
Pranav K. Patel, Lockheed Martin Corporation, Contractor to US EPA.
M. Arthur Alexander III, SAS Institute, Cary, NC

INTRODUCTION

Our Journey Begins with a Database Named "AIRS"

The U.S. Environmental Protection Agency (EPA) depends on information – lots of it – to carry out its mission to protect and improve the nation's air quality. EPA's main collection of computerized information about air pollution is ***AIRS***, the Aerometric Information Retrieval System. AIRS is the "mother of all air pollution databases" and the most respected source for this information in the world.

Values in the AIRS database tell how much pollution is released into the air by industry, commerce, and just plain folks. Other values tell the ambient concentration of pollutants in the air we breathe, and still more values tell what administrative and legal actions have been taken to fix pollution problems. EPA gives the 50 states and many local environmental agencies direct access to all the AIRS data, and direct control of their own data. By analyzing the values in AIRS, EPA and the states can determine the "health" of our air, where the problems are, and how well remedial actions are working. All of these are good things.

In the late 1980's more and more people began talking about "visualizing" data temporally and spatially – GIS and stuff like that. Yes, even back then, the need to understand and visualize trends in large databases REALLY QUICKLY, was beginning to creep into our ordinary daily lives! Our team looked at GIS and knew that was too much horsepower for the majority of our data owners and users. We needed something better than a pie-chart, but not as complex as three-dimensional underground toxic waste flow visualization.

Our users also needed something that they could access on our central mainframe from anywhere in the world, and we needed it to be something we already had available and knew something about. Federal agencies like the USEPA usually don't have resources to re-engineer software systems at the drop of a hat, either. We decided upon SAS/GRAPH® software – and lo, and behold, we were on our way! We called our new creation *AIRS Graphics – A New Way to* *See Air Pollution Data.*.

And Why Do We Need Graphics?

It's simple: a picture is worth a thousand words or – in our case – a thousand pages of printout! AIRS Graphics can help both novice and experienced AIRS database users find the proverbial needle in the haystack. What about trends analyses – that's where the "money" of the data is, and much more quickly spotted with visual representations.

AIRS Graphics lets an inexperienced database user browse the data visually, using maps and charts see categories and trends. An experienced user can employ more complex features of AIRS Graphics to quickly pinpoint a certain type of data. If more detail is required, the small list of subjects identified in AIRS Graphics permits a narrowly targeted retrieval of all the gory details from the full AIRS database.

Where Will It Lead?

Throughout this decade our team has built and extended AIRS Graphics, keeping up with new releases of the SAS® software along the way, and incorporating user friendly features into a mainframe environment – like our Point 'n' Click – that most programmers thought to be impossible.

Now, more and more, Federal Agencies like EPA are taking advantage of new hardware and software innovations to make databases open and available to the General Public. Why? Because the General Public has a right to know about the air quality where they live, and the technology to deliver this resolution of data, to an individual, at a reasonable cost, now exists. And, as banks found out years ago, letting a customer "help himself" with secure "ports" such as automatic teller machines, saves corporate dollars and promotes customer convenience and happiness. Everyone wins!

Our journey depicted here, chronicles where we've come from, where we are, and the somewhat scary unknown we're headed into with Public Access via the Internet. Just a few short years ago, it was not possible to make a TSO-based system like AIRS Graphics available to the world at large. Now it seems, not only possible, but inevitable. Our saga begins, long ago, when IBM Graphics terminals were just starting to appear on a few select desktops, and PCs with real computing power still seemed far away in the distant future. And long ago, was only eight years ago!

YESTERDAY - 1991

Remember what computing was like in 1991?

On the desktop, a PC with a 486-50 CPU was top of the line; "a screamer," one ad said. Windows™ 3.0 was available, but most of us were using DOS 5. Electronic bulletin boards were the way to distribute free technical information. We read glossy magazines about the "automatic data processing" industry, and wondered what kind of real-world computer projects would use this "Unix" and "client/server" stuff.

At EPA, the mainframe was king. We used PCs, of course, and some departments preferred to work on a cluster of VAX mini-computers. But only the mainframe provided industrial strength computing power, acres of disk and tape drives, a nationwide communications network, and help desk technical support. Every EPA desktop had access to the mainframe through 3270 terminal emulation software on the PC network. And there too, waiting for us to put it to good use, was the newest version of the SAS System, Release 6.06.

We Need a Better Way to See Air Pollution Data!

In 1970 Congress passed the Clean Air Act and created EPA as an independent agency. EPA established a national network of air pollution monitoring stations and began compiling a database of air pollution measurements. By 1991, that AIRS database contained millions of values, and the EPA mainframe was its home. EPA staff used the AIRS database to get a picture of the nation's air quality, and report it to Congress.

But it wasn't really a picture they got, it was text – lots and lots of text, columns of numbers, rows of numbers, page after page. The information was there, but the reports were not very, um, stimulating. Fortunately, a few people close to the AIRS database were familiar with the capabilities of the SAS System. They realized that SAS/GRAPH maps and charts could show AIRS air pollution data in new and interesting ways, and SAS/AF® software could make it accessible through a simple menu interface. Soon a new “viewer” for air pollution data was born – AIRS Graphics.

What Did We Set out to Accomplish with AIRS Graphics?

AIRS was a typical mainframe data system of the mid-1980's. Using “menu” screens displayed on a monochrome 3270 terminal, you chose a report and specified data selection criteria. Some time later, when the batch report job finished, you got a printed report or viewed a print file on your terminal. We thought this situation could be improved in several ways.

• **Pictures instead of words** Maps and charts are more effective than text reports for some kinds of data. We wanted to focus on geographic aspects of the data, and on “my neighborhood” more than the nation.

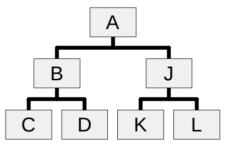
• **Immediate feedback** Batch reporting did not lend itself to data explorations. We wanted an interactive, responsive system that would provide results right away.

• **Ease of use** Instead of 300-page manuals, we wanted users to have pick lists and on-line help. We wanted attractive, intuitive screens.

• **Data integration** Each AIRS report contained a single type of data; to learn about air pollution monitoring stations and air pollution emitters located in your county, you needed two separate reports. We thought it would be informative to combine these kinds of information in one map.

So, How Did We Implement the Vision?

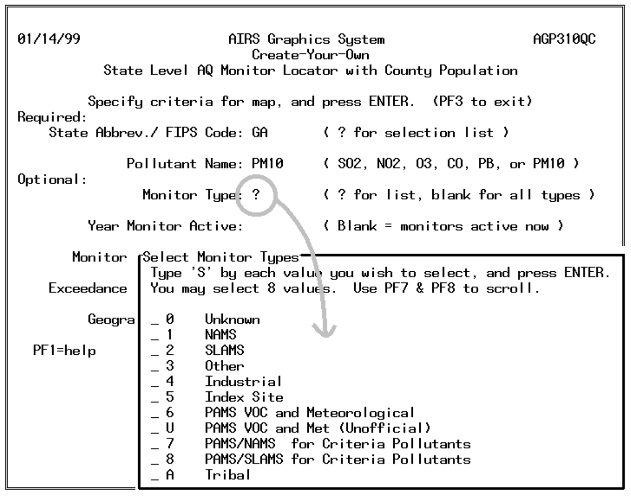
Using the SAS System, of course! We designed SAS/AF screens to provide a “menu” interface similar to the traditional AIRS screens. We designed “graphical reports” that we produced with GMAP, GPLOT, and GCHART procedures. We developed SCL programs to validate user choices, extract data subsets, run the procedures, and more.



AIRS Graphics used a “hierarchical” interface, like AIRS (and most DOS and mainframe applications of that time). By making choices on a series of menus, you got to one of several end points, and then you went back to the beginning. A➜B➜C, or A➜J➜K, but not B➜K or C➜D.

How Does AIRS Graphics Work?

Let’s go through the steps to create a map or chart. First, you choose which type of “graphical report” to produce. It might be a line plot of daily air pollution values, or a map showing the locations of air pollution monitoring sites. Twenty-six types of graphics are available.



```
01/14/99                    AIRS Graphics System                    AGP310QC
                               Create-Your-Own
           State Level AQ Monitor Locator with County Population

         Specify criteria for map, and press ENTER.  (PF3 to exit)
Required:
    State Abbrev./ FIPS Code: GA       ( ? for selection list )

              Pollutant Name: PM10     ( SO2, NO2, O3, CO, PB, or PM10 )
Optional:
                Monitor Type: ?        ( ? for list, blank for all types )

         Year Monitor Active:          ( Blank = monitors active now )

       Monitor  Select Monitor Types
                 Type 'S' by each value you wish to select, and press ENTER.
    Exceedance   You may select 8 values.  Use PF7 & PF8 to scroll.

         Geogra  _ 0     Unknown
                 _ 1     NAMS
   PF1=help      _ 2     SLAMS
                 _ 3     Other
                 _ 4     Industrial
                 _ 5     Index Site
                 _ 6     PAMS VOC and Meteorological
                 _ U     PAMS VOC and Met (Unofficial)
                 _ 7     PAMS/NAMS  for Criteria Pollutants
                 _ 8     PAMS/SLAMS for Criteria Pollutants
                 _ A     Tribal
```

Your choice activates a SAS/AF program specifically for that type of graphic. It displays a screen with the choices and options available for the graphic – geographic area, pollutant, date range, and so on. Because each type of graphic has its own screen, there are no irrelevant options. This helps you avoid inappropriate choices.

To begin making a graphic, you type values representing your choices in the screen fields. If you have generated this type of graphic before, the SAS/AF program restores your previous choices and displays them. You may enter a question mark in most screen fields and get a list of the available choices in a pop-up window.

When all the choices are made, you press the Enter key. The SAS/AF program validates each screen field, comparing each one with a list of valid values and checking for conflicts among the fields. If there is a problem, the program highlights the field(s) involved and

```
                    AIRS Graphics System
                       Create-Your-Own
                 Plant/Monitor Locator Map
             49 plants and 26 monitors selected.
  You may change criteria, or press ENTER to draw the map.
```

displays a message describing the problem. You alter the value(s) and press Enter. This sequence repeats until the program detects no problem with your choices.

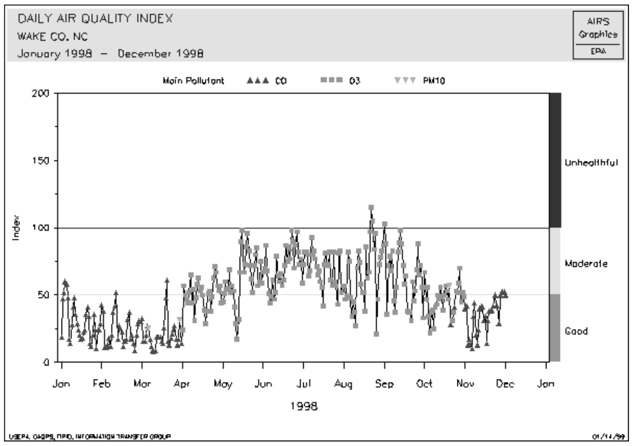
Using criteria based on your choices, the SAS/AF program extracts a data subset from SAS files or operating system files, and creates a new SAS data file in the WORK library. At this point, the program has enough information to produce the graphic you requested. But it’s possible you got something you didn’t expect – the entire state rather than one county you meant to choose. Or perhaps you got nothing – the county has no air pollution monitors of the type you chose. So the program “pauses” to let you know how much data it extracted. It displays a brief message on the screen and returns control to you.

At this juncture, you may alter your choices to select different data, and get a new summary message. Each time you alter a screen field and press Enter, the SAS/AF program repeats the validation, data extraction, pause sequence. (The program checks the modified attribute of each screen field to determine if you changed anything that would affect data extraction.) If you are satisfied with the data extraction results, you just press Enter again, leaving the screen fields unchanged. When the program detects I have data and no criteria changed, it goes to the next step – producing a graphic. The I have data condition prevents SAS errors that would occur if the program attempted to produce a map or chart with an empty data file.

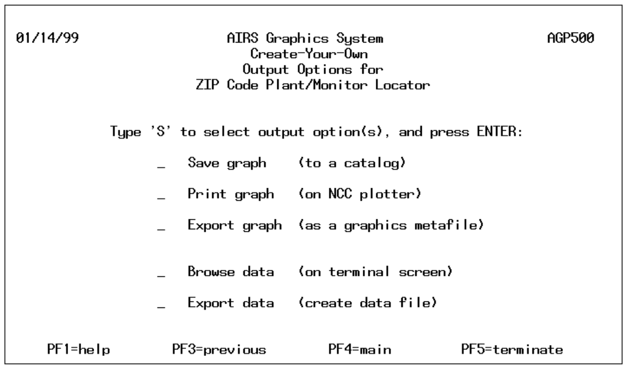
Now the SAS/AF program has reached the main event, its *raison d’être* – producing a graphic. Preparing for that requires some additional data extraction and manipulation. To make a map of air pollution monitoring sites, state or county outlines need to be combined with the site location data (extracted above), and projected with Proc GPROJECT. The extracted data may need to be summarized using Proc FREQ or reshaped with Proc TRANSPOSE. The SAS/AF program associated with each type of graphic carries out the appropriate sequence of steps required to produce a particular graphic of that type. Some steps are done entirely in SCL and some in submit blocks, using data steps and SAS procedures.

Although the specific processing for each type of graphic is unique, many graphics include common elements. An example is the title block across the top of every map and chart, which includes the AIRS Graphics “logo,” the name of the graphic, and a summary of data selection criteria. Since all graphics have a title block, we developed a common routine – an SCL method – that creates SAS Annotate Facility commands to draw the title block. A SAS/AF program invokes the method and sends it the data selection criteria for the graphic. The SCL method takes care of the details of composing the title and generating Annotate commands.

When all the preparations are complete, the ultimate step is to run a graphics procedure, GMAP, GCHART, or GPLOT, to create and display the image on your terminal screen. The default SAS graphics device driver for AIRS Graphics, *GDDMPCG*, includes a software zoom and pan capability, so it is possible to enlarge parts of a graphic. Since the graphic is a vector image, zooming in really does show more detail.



After admiring your graphical creation for a while, it's time to move on. Pressing the Enter key clears the graphic and takes you to a "disposition" menu screen. Here you can choose to save the graphic in a SAS catalog, send it to color or grayscale printers, or export it in several image formats, including CGM and PostScript. You also can view the data used to produce the graphic, or export it as text, database, or spreadsheet.



```
01/14/99                  AIRS Graphics System                  AGP500
                             Create-Your-Own
                           Output Options for
                       ZIP Code Plant/Monitor Locator


          Type 'S' to select output option(s), and press ENTER:

                _   Save graph      (to a catalog)

                _   Print graph     (on NCC plotter)

                _   Export graph    (as a graphics metafile)


                _   Browse data     (on terminal screen)

                _   Export data     (create data file)


    PF1=help        PF3=previous        PF4=main        PF5=terminate
```

Now you have reached the end of the line, the final node in the hierarchy of menu screens. Going back one level takes you again to the menu screen for the graphic you just produced. Your menu choices are still there, and you can easily change a field or two and produce another variant of the same graphic. Going back further takes you to the choice of graphics types, and then out of the AIRS Graphics application.

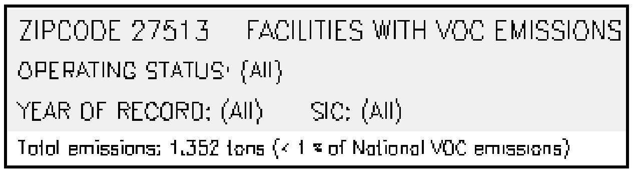
I Can See My Data!

There was much to like about AIRS Graphics. For the first time, regular people – not just GIS wizards and SAS jocks – could produce maps and charts of air pollution data. Certainly, the graphics lacked detailed information available in some AIRS printed reports. On the other hand, the graphics provided a much better, understandable view of geographically related data.

AIRS Graphics screens were more attractive and easier to use than many of the "regular AIRS" screens, where pop-up selection lists and help screens were nowhere to be found. Even without a graphics terminal (or graphics terminal emulation software) you could get maps and charts, thanks to the graphics export feature of AIRS Graphics. As you started a session, AIRS Graphics displayed a "news" screen with information about new features or temporary system problems. A "feedback" screen let you send us your suggestions, complaints, and kudos.

What is wrong with this picture?

But there were weaknesses, too. Mainframe graphics terminals and software emulations usually had seven colors and black background. Seven, bright, strong, primary colors. And SAS/GRAPH's fill patterns were.... Well, we used *solid* and *empty* whenever possible. Fonts also were a big problem. In graphics mode, mainframe terminals do not have hardware fonts, and even the simplest SAS/GRAPH soft font, *Simplex*, is rendered badly. Notice the uneven, "gnarled" look of the map title shown below. It was difficult to make text elements – title, legend, footer – large enough to read without robbing too much space from graphical elements.



ZIPCODE 27513   FACILITIES WITH VOC EMISSIONS
OPERATING STATUS: (All)
YEAR OF RECORD: (All)   SIC: (All)
Total emissions: 1,352 tons (< 1 % of National VOC emissions)

There also were weaknesses in the design of AIRS Graphics. Our maps showed where air pollution sources and monitors were located, their geographic distribution. But you could not associate a particular "dot" on a map with a particular factory or power plant or air monitoring site. If a map showed a site with obviously incorrect latitude-longitude coordinates, there was no way to identify which specific site that was.

Our maps did not provide enough "analysis" of the data. You could see that county A had one pollution source and county B had three. But you could not tell that the one source in county A released twice as much pollution than all three source in county B.

TIME PASSES . . . 1995

FRAME Entries Anyone?

Few legacy applications are rewritten simply for the sake of making them prettier. The old axiom of "if it ain't broke, don't fix it" is rarely more appropriate than here. So when SAS/AF FRAME entries became available, we began to consider whether there was anything in the FRAME entry environment that would make AIRS Graphics not just flashier, but better. Flashy graphics are fun to look at the first time you see them, but they can soon become a nuisance, particularly if they require time to download and don't improve the ease of use of an application. SAS/AF Program entries had already given us a great deal of flexibility in designing and creating screens for text based data entry and display of the data. We already had pop-up selection list capability via the *datalistc/n* functions. Extended tables allowed us to display large tables of textual data by creating scrollable areas at the bottom of any program screen. Screen Control Language was rich with functions to allow us extensive data manipulation, error checking, and access to all of Base SAS software through submit blocks. Why did we need FRAME Entries?

EPA air pollution data consist primarily of the annual emissions of seven "criteria pollutants" from thousands of plants nationwide, and ambient concentrations of those pollutants measured hourly by monitoring stations nationwide. All the data are integrally related to the geographical position of the plants, the locations of the monitors that track their emissions, and country population distribution within that network of sites. Monitoring stations are placed in selected areas to track the effect of pollutants as they leave the stacks and are carried by winds across the face of the earth. Since the data is so intensively geographic, most of the output from AIRS Graphics is in the form of maps of various geographic coverage areas selected by a user. The selection of coverage areas and information related to the seven pollutants is the primary function of the menus. The display of the resulting data at varying coverage areas is the primary output.

Since a SAS/AF program screen could only display text, there could be no graphical “assists” for the user on the program screen. Decisions about the various parameters had to be made “in the dark.” A map was then created and displayed by leaving the program screen and branching to the graph output window. The first time a user actually saw a graphic visualization of his selections, was when this resulting map was produced. After one look at the output it might be evident that the inclusion of some other parameter or a different level of a given parameter would have been closer to what the user really wanted. Back to the text screen to begin the trial and error process of text entry to graphic output and back – over and over. It was apparent that being able to combine text and graphics on the same screen would be of some benefit. Such a screen would allow a more graphical approach for choosing coverages.

Most people who spend any amount of time with maps have a pretty good knowledge of the geography of the United States, down to the state level. After that most would recognize a map of their own county and perhaps even neighboring counties. But what if they needed to select a cluster of counties, perhaps one that straddled state lines? Few individuals could accomplish this from an alphabetical selection list of county names. Remember, even though a state's counties are listed in a data base in alphabetical order, the geographical positions of that neatly ordered list will be scattered all over the state.

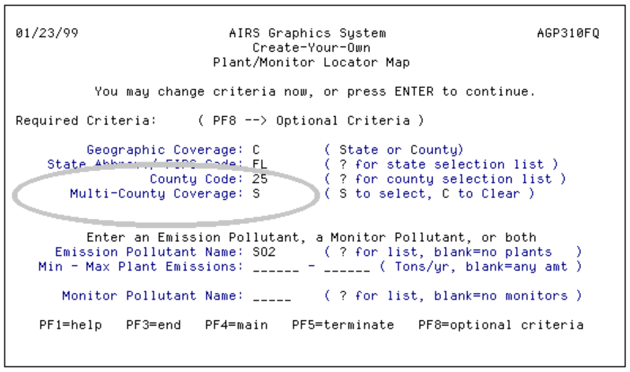
We realized that we could make life a lot easier for AIRS Graphics users if we gave them a screen combining a text selection list of counties with a map of those counties. This was our first “bright idea” for a FRAME entry that would improve AIRS Graphics.

FRAME One: The Multi-County Selection Window

For several AIRS Graphics maps, users could select a single state or a single county as the geographic coverage. Users asked us for the ability to include multiple counties in a one map, and we thought this looked like a good place to try out the new FRAME technology. A new “toy” for us, a better interface for users!

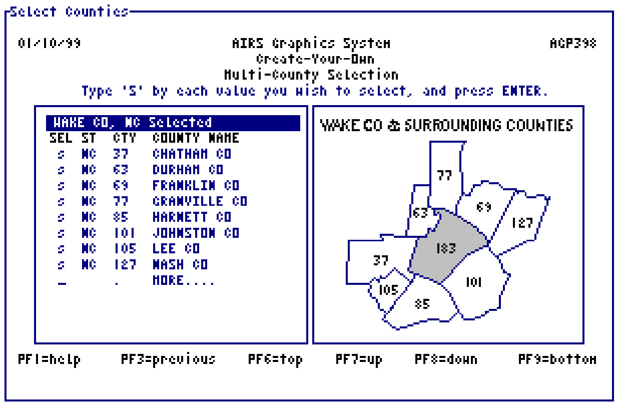
With the existing menu screens, a user selected a state and a county within the state to produce a county scale map. We decided to extend this process, rather than implement something radically different. Users would continue to select a primary county; we would add the ability to select additional counties, particularly those surrounding the primary county.

To implement multiple-county selection, we added a field to the menu screens of affected maps. Selecting the new field signaled that a multi-county map was desired.



```
01/23/99                  AIRS Graphics System                  AGP310FQ
                            Create-Your-Own
                        Plant/Monitor Locator Map

          You may change criteria now, or press ENTER to continue.

Required Criteria:     ( PF8 --> Optional Criteria )

         Geographic Coverage: C        ( State or County)
     State Abbrev./FIPS Code: FL       ( ? for state selection list )
                 County Code: 25       ( ? for county selection list )
       Multi-County Coverage: S        ( S to select, C to Clear )

         Enter an Emission Pollutant, a Monitor Pollutant, or both
      Emission Pollutant Name: SO2      ( ? for list, blank=no plants   )
    Min - Max Plant Emissions: ______ - ______ ( Tons/yr, blank=any amt )

       Monitor Pollutant Name: _____    ( ? for list, blank=no monitors )

   PF1=help   PF3=end   PF4=main   PF5=terminate   PF8=optional criteria
```

When a SAS/AF program saw the multi-county signal, it identified counties surrounding the selected, “primary” county. The program used the maximum and minimum latitude and longitude points for the selected county to define a bounding rectangle. Then it scanned a file of maximum and minimum points of all US counties, and selected any county whose bounding rectangle overlapped or touched the bounding rectangle of the center county. Finally, it created a small map of the primary and surrounding counties, and displayed the map beside a selection list of the county names in a new FRAME window. County FIPS codes shown on the map and in the list allowed a user to associate a county’s name with its location. The FRAME allowed us to put the map and list in a side-by-side arrangement the seems “natural” and is amenable with the shape of a terminal screen.



```
Select Counties
01/10/99                  AIRS Graphics System                  AGP398
                            Create-Your-Own
                        Multi-County Selection
        Type 'S' by each value you wish to select, and press ENTER.

   WAKE CO, NC Selected
   SEL  ST   CTY   COUNTY NAME
    s   NC   37    CHATHAM CO
    s   NC   63    DURHAM CO
    s   NC   69    FRANKLIN CO
    s   NC   77    GRANVILLE CO
    s   NC   85    HARNETT CO
    s   NC   101   JOHNSTON CO
    s   NC   105   LEE CO
    s   NC   127   NASH CO
    _         .    MORE....

PF1=help    PF3=previous    PF6=top    PF7=up    PF8=down    PF9=bottom
```

To allow maximum flexibility, additional counties could be added to the cluster by selecting a “more” entry on the selection list. Since we couldn't anticipate everything, this “more” selection would have to be accomplished by entering the state and county into a second extended table that was superimposed on the county cluster map. With this addition, a user could select virtually any number of counties, even counties from multiple states. A lot of flexibility compared to selecting a single county from a text list!

FRAMES PROLIFERATE! A New Way To View Data Emerges

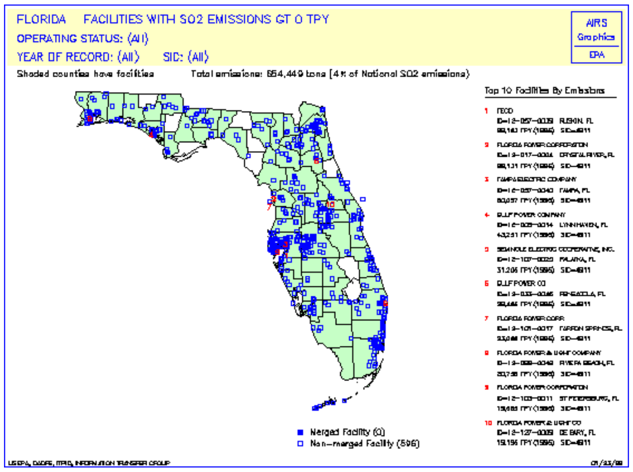
The multi-county selection window showed us how useful FRAMES could be. But after an AIRS Graphics user made county selections with our slick new window, he or she went back to the standard interface provided by SAS/AF program screens. Exploring air pollution data with AIRS Graphics involved many iterations of:

visualize it ➜ make selections ➜ generate it ➜ display it ➜ decide what to do with it ➜ go back and do it all over again

What we really wanted to do was scrap text selection lists and make geographic selections by clicking on a map. Even better, we'd like to start out with the data already in visual form. We'd like to decide on a new subset of the data based on the current graphical view of the data. We'd like more flexibility in data selection and display, more paths through the data, without having to specify parameters each time we changed from one view to another. And dream of all dreams, we'd like to be able to select a specific plant or monitor on a map and get immediate information about it, instead of having to guess about what polluter was represented by that little blue marker in a sea of blue markers. (See map at top of next column.)

The capability to click on a map polygon to select a particular state or county represented by that polygon was later implemented in V6.11 of the SAS System with the FRAME Map object. But in V6.08 on MVS, all we had was the SAS/GRAPH Output object. It was not designed to give this kind of feedback, but we dreamed big dreams, and we experimented.

Our experimentation paid off. Through careful control of the data, SAS/GRAPH options, and Proc GMAP output, we were able to predictably link elements in a GRSEG output with the corresponding input data. We could select a polygon on a SAS/GRAPH Output


FLORIDA FACILITIES WITH SO2 EMISSIONS GT 0 TPY
OPERATING STATUS: (All)
YEAR OF RECORD: (All) SIC: (All)
AIRS Graphics
EPA
Shaded counties have facilities
Total emissions: 654,449 tons [4% of National SO2 emissions]
Top 10 Facilities By Emissions
1 TECO
ID=12-057-0039 RUSKIN, FL
2 FLORIDA POWER CORPORATION
ID=12-017-0004 CRYSTAL RIVER, FL
3 TAMPA ELECTRIC COMPANY
ID=12-057-0040 TAMPA, FL
4 GULF POWER COMPANY
ID=12-005-0014 LYNN HAVEN, FL
5 SEMINOLE ELECTRIC COOPERATIVE, INC.
ID=12-107-0029 PALATKA, FL
31,206 TPY (1995) SIC=4911
6 GULF POWER CO
ID=12-033-0045 PENSACOLA, FL
7 FLORIDA POWER CORP.
ID=12-101-0017 TARPON SPRINGS, FL
8 FLORIDA POWER & LIGHT COMPANY
ID=12-099-0042 RIVIERA BEACH, FL
9 FLORIDA POWER CORPORATION
ID=12-103-0011 ST PETERSBURG, FL
10 FLORIDA POWER & LIGHT CO
ID=12-127-0009 DE BARY, FL
19,196 TPY (1995) SIC=4911
■ Merged Facility (0)
□ Non-merged Facility (596)
USEPA, OAQPS, ITPG, INFORMATION TRANSFER GROUP
01/23/98

object and link it to corresponding information in our SAS data files, using a value returned by the FRAME *_get_info_* method.

Through additional experimentation, we learned how to use the same techniques with markers annotated on the maps. We could determine what monitoring site or factory an individual map marker represented – a capability still not available in the SAS System outside of SAS/GIS™.

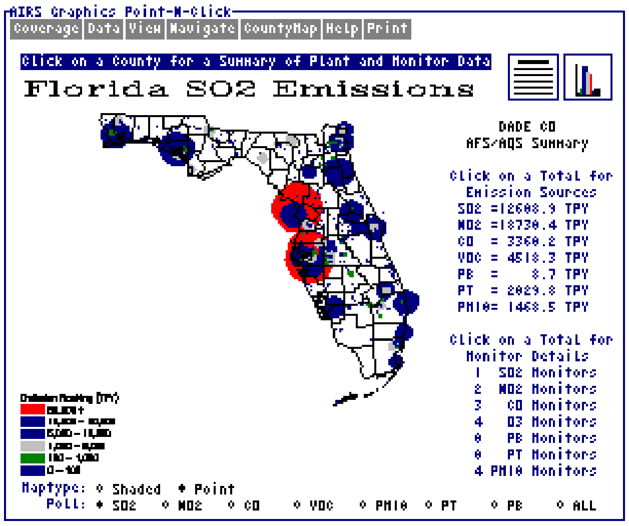
The result of this experimentation was *Point-N-Click*, a new component of AIRS Graphics that gave users a new approach to interaction with the data. The flexibility of FRAMES would give us as many paths from Point A to Point B as there were users of the system. Whereas AIRS Graphics had considered the graphical representation of the data to be the finished product of a series of text selections, Point-N-Click would be a visual data browser, combing data selection and the visual results in a single package.

The standard components of an AIRS Graphics map were geographic coverage and pollutant selection, spread over various types of graphical output. Point-N-Click was designed with three basic data views: maps, charts, and tables. We also included an *ad hoc* report generation tool. We wanted a user to be able to change the view, the pollutant or the geographical coverage from any screen, with a simple click. The FRAME entry environment provided us radio boxes, list boxes and check boxes, in addition to text entry fields. Since V6.08 had no customizable icons or toolbars, we simulated graphical pushbuttons with miniature SAS/GRAPH Output objects. The FRAME Graphics object had dynamic hotspotting capability, and the extended table object could identify an individual column or row selection. With the added functionality of our hybrid map "hot spots", we had the components necessary to implement the vision.

In any Point-N-Click data view, you could choose the geographical coverage by selecting an element of the displayed data – a map polygon, a bar on a graph, or a row in an extended table. Thus, you could start out at the national scale and drill down to county level with a couple of clicks. To choose a different pollutant, you simply clicked another radio button. To jump between data views, you clicked on one of our graphical "simulated" icons.

The Florida map shown at the top of the next column is an example of the dynamic new data screens that comprise AIRS Graphics Point-N-Click. Plant locations are depicted by colored circles, scaled to indicate the relative size of the emission. These circles make it obvious which locations are the larger emitters.

The tables on the right side of the screen summarize county totals for both plant emissions for each of the seven criteria pollutants and the monitor count for each pollutant. The user can click on any county polygon to view this tabular data for the selected county.



DADE CO
AFS/AQS Summary

Click on a Total for Emission Sources

SO2	=	12608.9 TPY
NO2	=	18730.4 TPY
CO	=	3360.2 TPY
VOC	=	4518.3 TPY
PB	=	8.7 TPY
PT	=	2029.8 TPY
PM10	=	1468.5 TPY

Click on a Total for Monitor Details

1	SO2 Monitors
2	NO2 Monitors
3	CO Monitors
4	O3 Monitors
0	PB Monitors
0	PT Monitors
4	PM10 Monitors

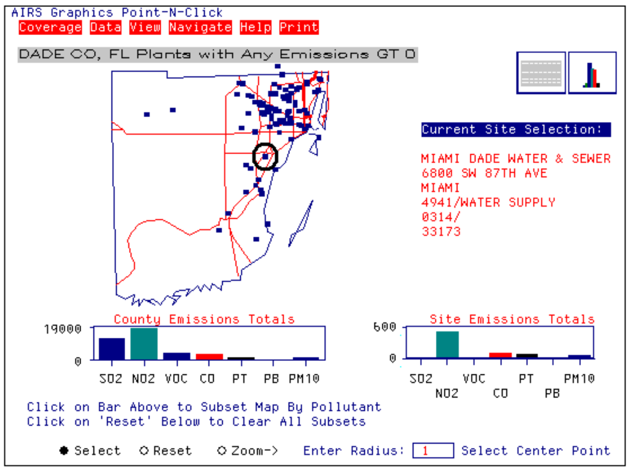
Maptype: ○ Shaded ● Point
Poll: ● SO2 ○ NO2 ○ CO ○ VOC ○ PM10 ○ PT ○ PB ○ ALL

Each line of the table is also selectable. Clicking on one of the table values will display a popup list of the sites included in the table entry. Likewise, a row in the popup can be selected to view even more detail about the site selected from the popup.

At any point, the pollutant displayed on the map can be changed by selecting another radio button. A detailed table of the state data, summarized by county, or a bar chart of the top 10 counties can be displayed by clicking on the table icon or the bar chart icon, respectively, located in the upper right corner of the screen. Finally, an immediate printout of any screen in the system can be generated by clicking in the 'Print' button in the pull down menu at top of the screen.

At the county level, map markers are also selectable. The map of Dade County, FL below, contains a table and a bar chart giving detailed information about a single, selected plant. This data is updated each time a marker is selected from the map. For areas where markers are located so close together that they overlap, a zoom feature has been included. Selection of the zoom radio button, a coverage radius and the subsequent selection of a map marker to serve as the zoom center will generate and display a circular subset around the selected marker. At some zoom level this feature will allow selection of individual markers, even in locations of high site density. Clicking on the site table will display a popup of additional detail for a selected site.

With the addition of Point-N-Click, Airs Graphics users had a completely new way to view air pollution data.



Current Site Selection:
MIAMI DADE WATER & SEWER
6800 SW 87TH AVE
MIAMI
4941/WATER SUPPLY
0314/
33173

Click on Bar Above to Subset Map By Pollutant
Click on 'Reset' Below to Clear All Subsets

● Select ○ Reset ○ Zoom-> Enter Radius: 1 Select Center Point

TOMORROW - 1999

An Opportunity to Inform the Public

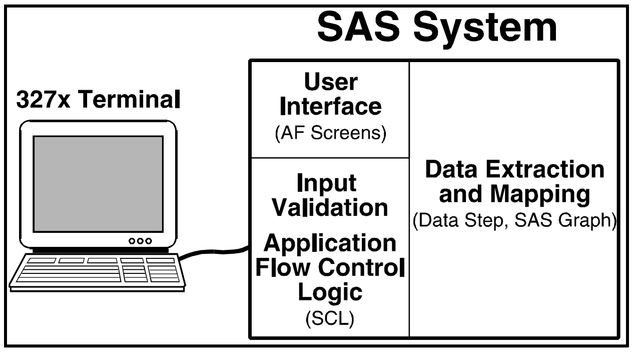
As Airs Graphics on the mainframe became popular with the EPA community and State and local agencies, people outside these agencies – who did not have access to the EPA mainframe – began to request maps and charts, too. We needed a way to disseminate information to the public. The cost, however, for providing the information for both EPA and the public would be high. EPA would need more communication lines with special mainframe communications hardware, dial-up modems, phone lines, etc. The public would need to have mainframe terminal software, high speed modems. How could we put AIRS Graphics in the hands of a wider audience with a small budget and existing equipment?

The most cost effective way to disseminate information to the world was THE WEB. Powerful PC's with Internet connections and modern, graphical browsers sat on most desktops. The Internet was becoming ***the*** Information repository, with an associated low cost.

AIRS Graphics on the Mainframe: We Control the User

Keeping the original design goals of AIRS Graphics, while at the same time adding additional goals of ease of use, interactivity, and universal accessibility, was a challenge. In addition, we needed to reuse existing code to make the transition from mainframe to Web affordable. The language of the Web was HTML, a text markup language for large documents. The HTML was put on the back burner. The question we ran into was, should we use advanced features on the web like GIS servers, Java, and other emerging technologies. We wanted to design for the future and take advantage of new technologies. We realized quickly that the Web world has many emerging technologies, and the software cycles are in months and not years. So we had to devise a plan to keep AIRS Graphics on the leading edge but not the bleeding edge. The basic challenge was to go from a mainframe client/server model to a n-tier application without getting burned.

AIRS Graphics on the Mainframe model can be summarized as below:



The 3270 terminal collected key strokes and sent them to our application. Input checking and error handling took place completely within the AF application. The menu processing and decoding also occurred in the AF application. Data extraction occurred in SCL code, and generation of maps and charts was done by SAS/GRAPH submit blocks. These steps could be divided into four blocks: selection criteria processing, data extraction, graph generation, and graph / data downloading.

The mainframe was a closed environment with mostly well-defined screens, SCL code and a sprinkling of data steps and procs. We were in full control of user capabilities, and we decided where a user could proceed. The mainframe response time was reasonably fast and communications were reliable. Few things could go wrong.

What Is the Best Way to Proceed?

One possible way to move AIRS Graphics to the Web was to generate popular graphs and charts and serve them as static HTML pages. This was put on the AIRS Web site, but, visitors to the web site began requesting dynamically generated graphs. The burden of generating the graphs every week and the maintenance and coordination of this effort was large and expensive.

Another option was to put the data into a database like Oracle and use their Web server. This option was also discounted because it did not leverage any of the existing code and would require a expensive database developer. How the graphs would be generated was not clear.

We considered the option of using Java, but decided against it. At that time the technology was too new, wide applicability was not there, and it precluded reuse of our existing SAS code. We also looked at many GIS solutions and concluded that the investment in resources was too high and the response times for these systems was relatively slow.

There were very few choices left and most of them did not lend themselves to creating a mainframe like application. So we again turned to SAS Institute for a solution for our needs. We concluded that SAS/INTRNET™ would allow us to make use of existing code and formed a base to start from.

AIRS Graphics on the Web (AGWeb): No One Is in Control

Armed with SAS/INTRNET and the HTML language, we set out to design how AGWeb would work. One of the obvious directions that we could head was a "sequential" web dialog method. An initial Web page presents the first choice – geographic coverage, for example. Choosing the "state" option leads to a second Web page that lists state names. Then comes a Web page for choosing pollutant, and so on. This certainly would keep the developers in control, since the paths they define are the only ones available to users. One problem with this method was that the "hierarchy" for AGWeb would get very deep and complicated, even for the developers. This method also could place a heavy load on the servers, since each choice would generate a request to the servers.

Another alternative was the "airplane cockpit screen." A scrollable Web browser window gave us plenty of screen "real estate" to work with. No more 32x80 mainframe screens! The cockpit method would use a single web page to list all the choices and selections possible. This would put more control in the users hand. The load on the servers would be reduced because a single request to the server would include all the selections a user made.

HTML was not designed as a screen layout language, and it lacked a good way to precisely control how elements were placed on the screen. This was in contrast with SAS/AF screens, where we could layout the screen to the precise row and column. In the cockpit model, all the data for every possible choice for a particular map would have to be included in the Web page. Some selection lists would have hundreds of items. Such lengthy item lists would make the page download time quite long. Furthermore, the complexity of a cockpit-style page would be daunting to people unfamiliar with many of the choices offered. Thus, we discarded the cockpit model.

The ideal system would work like the AIRS Graphics on the mainframe, which has gone through a series of usability studies, human factors analysis, and interface designs. AIRS Graphics on the mainframe offers a good compromise between the "sequential" and "cockpit" models. We looked for ways to create a environment similar to AIRS Graphics on the mainframe.

Cool Technology Helps

HTML - HOW TO MAKE LONG PAGES

HTML had basic user interface objects: text and graphics display, text fields for user input, and buttons and links for navigation. This was fine for a web dialog method, but we needed something that could be dynamic, that would allow us to manipulate the user data and control the flow of interaction. We needed something with basics of programming and ability to manipulate the screen.

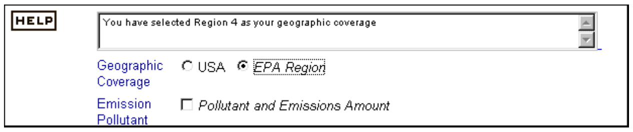
The only thing that came close to meeting our needs was Javascript! With Javascript we could create variables and control some of the flow of interaction. We could also access HTML data in a limited way. The combination of Javascript and HTML did not have all the capabilities needed to port the mainframe application. It did not have the same level of control as a SAS/AF application. But there was enough functionality to create some fancy Web applications.

The first need for AGWeb was to layout the screen. Most commonly used tags gave us very little control of laying out the screens. HTML's table tags were useful in having some control of the layouts, although the tables were not designed to be used as layout tools. In the web world you use what you have! The tables helped keep labels with the corresponding input items.

MESSAGE BOX - THE KEY TO TALKING TO THE USER

We needed a way for AGWeb to give feedback to a user – a way to provide user input error checking, server error messages, information messages, and instructions to guide a user's interaction with AGWeb. We wanted an area of the screen we could change as we needed.

We looked at frames as a way of accomplishing our goal. We could put a single frame at the top of the screen to mimic the message line in SAS/AF and load that frame with static message pages. This did not meet or needs for a dynamic messages. Frames were not widely implemented and took up valuable screen space. Cascading Style Sheets were also available and ideal to implement a message line. At the time this technology was in its infancy and not widely implemented consistently, so did not consider it a viable solution. The best compromise we could make was to put a multiline text input box on the screen, as_shown below. With Javascript we could control the content of the text box as we desired. Javascript to the rescue!


You have selected Region 4 as your geographic coverage

BUTTONS, BUTTONS, BUTTONS

The HTML form feature allowed us to use radio buttons to select from one of many, and check boxes to select many of many. These buttons were good for selecting from at most ten choices. Pop-up lists (selection lists) allowed both one of many and many of many selections. The pop-up lists were good for selecting from a large number of choices. Since all choices do not have to be visible, less screen real estate is consumed by big lists.

JAVASCRIPT TO THE RESCUE_*AGAIN*!

In fact, screen real estate and page (file) size were the next problem we had to solve. Each AIRS Graphics map had some choices that are required and some that are optional. The types of choices included select from a few, select from a large number, select by drilling down, and single selection with range specification. We wanted the user interface to be simple enough so that a novice general public user would not be overwhelmed. At the same time we wanted an EPA or State agency expert user to be able to specify complex criteria and get a map quickly.

Again Javascript saved the day with its ability to attach an action to the press of a button, and its ability to open a new browser window (the *window.open* method). This combination allowed us to pop open another browser window without all the decoration buttons when the user pressed (clicked) a button. The new window created with *window.open* appeared on top of the current browser window. It was a child window of the main parent window. The content of the child window could be controlled with *window.open*. When the child window was opened the parent window did not change (was not reloaded). All the Javascript variables and HTML objects in the parent window retained their information.

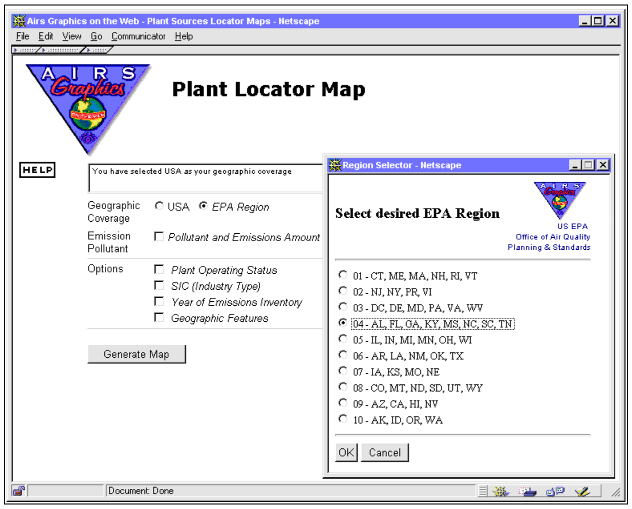
This feature allowed us to pop up a new window where a user could make further selection. When the child window was opened, Javascript also allowed us to communicate the current selection from the parent to the child window. Changes in the child window could be communicated to the parent. This gave us the basis for implementing the equivalents of selection lists and extended tables in SAS/AF.

Putting it All Together

Armed with Javascript we built a prototype that put it together. Let's see how it worked and looked. The following window shot shows a sample of a Plant Locator Map Web page, which accepts a user's choices for producing a map showing the locations of plants emitting particular pollutants.

We needed some buttons to activate pop-up windows for further selections, and some buttons that directly select a particular value. Geographic coverage is an example of a choice that employs both button types. The options are *USA* and *EPA Region*. Clicking on *USA* simply turns on the radio button indicator, to show that option was selected. However, because there are ten EPA regions, selecting the *EPA Region* button will pop up a window containing a list of EPA regions.

We needed a way to indicate which buttons popped up a window. We decided to use italic font style for the label of pop-up buttons. In the figure below, all options are labeled with italic font because they all pop up windows for further selection. The "on" state of the buttons (radio and checkbox) indicate that selections have been made for that button. The selections made are stored in hidden HTML form variables to be processed later.


Airs Graphics on the Web - Plant Sources Locator Maps - Netscape

Plant Locator Map

You have selected USA as your geographic coverage

Region Selector - Netscape

Select desired EPA Region

US EPA
Office of Air Quality
Planning & Standards

- 01 - CT, ME, MA, NH, RI, VT
- 02 - NJ, NY, PR, VI
- 03 - DC, DE, MD, PA, VA, WV
- [✓ 04 - AL, FL, GA, KY, MS, NC, SC, TN]
- 05 - IL, IN, MI, MN, OH, WI
- 06 - AR, LA, NM, OK, TX
- 07 - IA, KS, MO, NE
- 08 - CO, MT, ND, SD, UT, WY
- 09 - AZ, CA, HI, NV
- 10 - AK, ID, OR, WA

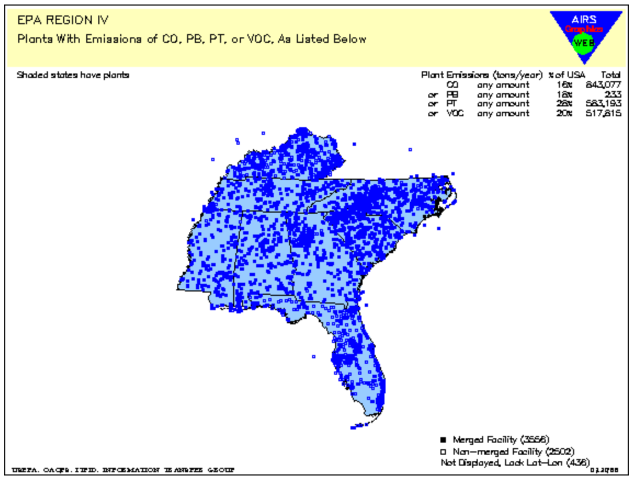
Document: Done

SAS/INTRNET IS COOL!

After a user has selected everything needed to produce the graph, he or she presses the *Generate Map* button and the magic begins. The selection criteria are put into CGI variable form and submitted to the SAS/INTRNET Broker. The Broker then passes the CGI variables to our SAS/AF program. This SAS/AF program is similar to the mainframe version without the AF screens and the error checking of the screen variables. The screen variables now reside in the browser variables. The SCL program takes the values for the variables sent by the broker, processes them to make a graphic, and sends the image back to the browser. SAS/INTRNET nicely handles the CGI interface needed to make the Web application.

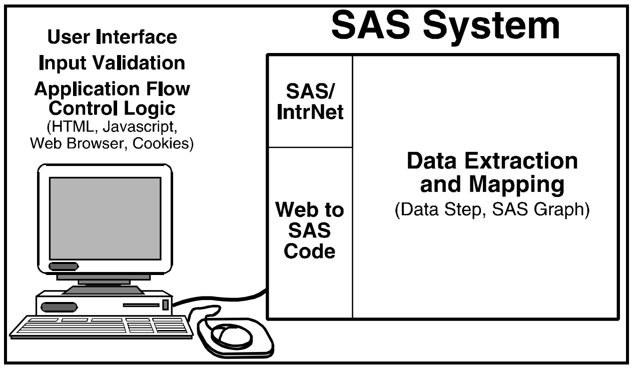
GIFs or JPEGs?

To display the maps and charts that would be generated we would needed to have the maps in either GIF or JPEG formats. We were lucky because SAS Institute in its wisdom had already created graphics drivers for both GIF and JPEG. We chose GIF for its wide spread usage. To adapt mainframe graphics to the Web, we changed some foreground colors for better visibility, and we reversed the background to white. We switched from a mainframe graphics driver to one of the GIFxxx drivers. Otherwise, we kept the graphics generation procedures the same as they were on the mainframe. The “Web maps” looked pretty good, but smaller text was not readable. The figure below shows a sample map.


EPA REGION IV
Plants With Emissions of CO, PB, PT, or VOC, As Listed Below
AIRS
Graphics
WEB
Shaded states have plants
Plant Emissions (tons/year) % of USA Total
CO any amount 16% 843,077
or PB any amount 18% 233
or PT any amount 26% 583,193
or VOC any amount 20% 517,815
Merged Facility (3556)
Non-merged Facility (2502)
Not Displayed, Lack Lat-Lon (436)

COOKIES AREN’T JUST FOR KIDS!

Once a user looks at a map, he or she can decide to refine the criteria. Using the browser’s *back* function returns to the page of selection criteria for the map. But now the values of HTML form hidden variables are gone! Reloading the page has cleared the hidden variables’ values. We overcame this problem by using cookies to save the hidden variables and state information. Some special routines allowed us to package the variables into a single cookie value. With the cookie code in place a user can produce variants of a map easily, since only the changes in selection criteria need be entered. Javascript and cookies retain the Web page choices, just as SAS/AF retains mainframe screen choices. The figure below shows the AGWeb model that we are using.



YES IT’S EASY, BUT YOU CAN *ALWAYS* USE SOME HELP!

With addition of a help button to assist users, we had a full application to work with. The help screens are used to provide quick description of the screens. This application worked like the mainframe version with a prettier interface and wide accessability.

WHAT DID WE GIVE UP?

The AGWeb based on SAS/INTRNET allowed us to leverage existing mainframe code and bring it to the Web world. What did we give up? The AGWEB application is not tightly integrated and has many more bearings where failure can occur. The user is in control so the programmer has to work harder – what else is new?. The small fonts are a big problem because they are not readable.

WHAT’S NEXT FOR AGWEB DEVELOPMENT?

What about the future? As always AGWeb will continue to be on the leading edge and here are some items on our wish list.
• Vector fonts for small font readability.
• Vector based graphics that will provide a zoom capability.
• A drill down capability on the maps and charts
We also look forward to accessing the live AIRS database for some types of data. Even further down the road, we hope to provide FRAME-like capabilities on the Web with Java. Who knows – maybe one day AGWeb will be on a screen in your car’s dashboard, helping you plan trips which avoid routes through high risk air pollution areas. One can dream, can’t they?

EPILOGUE

The Future: Bigger, Better, Faster but Not Free

Ok, so now I have the data, fast, cheap, via the Web – but where’s the expert to help me analyze and interpret the data? What does the data mean? “Smart Guide” software technologies look like the direction we’re all looking to these days. Improved access also means addressing new problems including education and interpretation. Data authenticity becomes more of an issue now than ever before – never have so many people had so much access to so much data with so little trouble, and at such low cost.

As more and more people have access to more and more data, there will be greater scrutiny of the data, more questions about the data, and the need for greater human interaction and contact – people who can talk about the data and make reasonable judgements.

Real-Time air pollution data – like the kind of information you get when you call up your local Time/Temperature phone number – is being developed on a national scale now. Why do I need Real-Time air pollution data? Let’s say that you operate a daycare center for children under the age of six, and some of them have chronic respiratory problems, AND your center is in Los Angeles. Being able to obtain real-time smog concentration data might dictate – for safety and good health – that you keep the kids inside today, or let them romp in the backyard after lunch.

Real-Time air data will also facilitate the development and use of predictive air pollution models which have superior accuracy. Let's imagine that you and your wife – now in your mid-80's – want to visit your new baby nephew in Atlanta next week – and it's the middle of a heatwave in August. If you have access to a website that predicts dangerously high levels of ozone in Atlanta next week – you might change your travel plans, and have a much more healthy visit some other time.

Real Time data projects for Internet public access, such as EPA's award-winning AIR*Now* website (www.epa.gov/airnow), which show ozone formation and transport in near real time, are becoming the order of the day. Projects like AIR*Now* require complex data reporting and transport networks, huge database access and storage requirements, and major computational horsepower. Web-based software of the future must ***have it all*** for real time needs and users.

As the costs of computers, internet access, data storage and analysis continue to decrease, the general public can expect to gain greater access, with greater ease, to more air pollution data at the EPA and all at lower costs than ever imagined. But even at a penny a page, is it reasonable to request a billion pages? Should these costs be borne by all – or just the super users? The debates over who pays for public access – and how much – are just beginning.

CONTACTING THE AUTHOR

Please address questions or comments to:

Mr. Thomas E. Link
U.S. Environmental Protection Agency
Office of Air Quality Planning and Standards (MD-12)
Research Triangle Park, NC 27711

link.tom@epa.gov