Paper 234-29

Guiding your Enterprise with Enterprise Guide

David Johnson FRSA, DKV-J Consultancies, Melbourne Australia

ABSTRACT

The analysis of business processes, and delivering an evolving set of reports can provide a real challenge if we are to avoid revisiting and redoing our analyses each time an enhancement is requested.

As our SAS trained people develop and move on, we can find that not only are we dealing with existing work being repeated, but we also have a challenge in training and orienting our new SAS users.

If we could bundle our SAS code in a package that retained our analysis code and notes with our current production code, then our new SAS users, and all those who need to change the report will benefit from the initial design work.

We are going to explore the use of Enterprise Guide for three sets of users: the experienced SAS programmer, the new SAS programmer, and the person who has very little SAS knowledge. However, the last person knows how to use a GUI interface to select data tables, apply filters and drop table elements onto a reporting interface, but no other knowledge of SAS software is assumed. This paper is intended to provide information targeted for each group. The products discussed are Base SAS and Enterprise Guide.

THE CHALLENGE

- The business asks for performance measurement reports
- Our analysis of the data highlights rules we need to apply to make those measurements.
- We also find data that doesn't match their understanding of the business process and rules.
- We model reporting tables, performing a retrospective capture of data.
- We design the reports they request
- We add reports to handle the exceptions our analysis identified
- We bundle a process that updates our tables with their reports and have it scheduled.
- Some time later, the business identifies additional reports that it requires.
- The person who creates the new reports didn't create the original reports and can't find the research notes.
- The new person goes through the whole process of data exploration and validation again before modelling the new reports.

How can we avoid the wasted effort of repeating work that has already been adequately completed, and provide our new user with a valuable lesson in our analysis methodology and rigour?

CHARTING A COURSE TO OUR REPORTS

The business brief for my report was simple enough. Our website maintenance team wanted to look at some of the data from the web site access logs and get a view of the sites audience and performance. The logs were already being retrieved from three off-site Unix servers on a daily basis, and were about to become available on an in-house IIS server for a new site. Since pages on the sites were linked they wanted to follow some of the visitors between the sites, as well as analyse each sites performance.

Delivery of the data required a series of steps as follows:

- Introduce them to Enterprise Guide.
- Make available the tables containing the log data to the Enterprise Guide client. (The data is assembled in a daily batch SAS process.)
- Review the data structures with the support team, and identify the type of queries that will be required.
- Discuss modifications to the queries or process to better utilise the available data.
- Assemble recurrent queries or analyses into a project that can be scheduled for routine reporting.
- Deliver to the data security team who became interested in the project when analysis suggested attempts were being made to hack the site.
- Discuss with and deliver to the IT Management team who wanted reassurance that the expenditure generated business value.

EXPLORING THE DATA

We started by reviewing the data with our primary users. This gave us some idea of the queries that will be run. For the experienced SAS programmer who may model data changes, understanding the queries to be run would influence the way we build the model. We started them in a blank Enterprise Guide client window (figure 1).

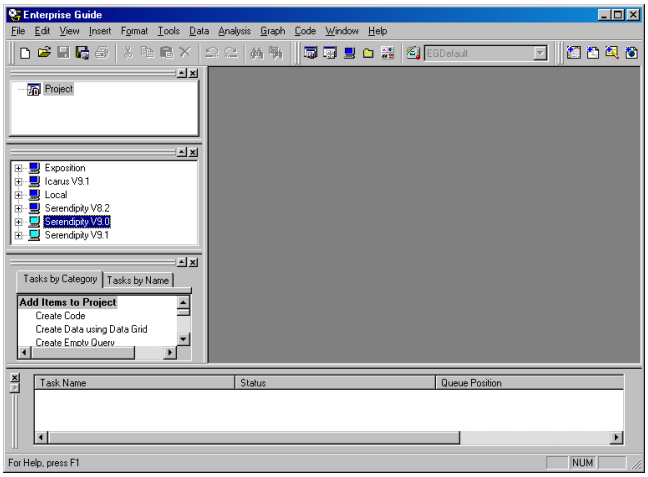

Figure 1
In the middle pane of the left hand side of the window are a series of icons that look like computer monitors. This is our server pane, and each icon identifies a SAS server available to us.

The servers include a Windows workstation, a Solaris server and a Windows 2000 server called “Serendipity”.

“Serendipity” has three entries that refer to three “object servers” started on that machine. Each object server is running a different version of SAS. They allow us to test any of our code on any of the three versions of SAS we have available.

This is a powerful tool since it allows us to validate existing code against a new version of SAS before we migrate versions.

Two of the “Serendipity” windows are a pale blue, which indicates they have been activated. This means we can now use either of two SAS versions to run our analysis jobs.

If we click on the ‘plus’ symbol beside one of these servers, we will see three icons displayed. The first is a link to the libraries we have available to that SAS session. A screen capture with the library list shown, as well as the tables available in the ‘WEBLOGS” library appears at figure 2.

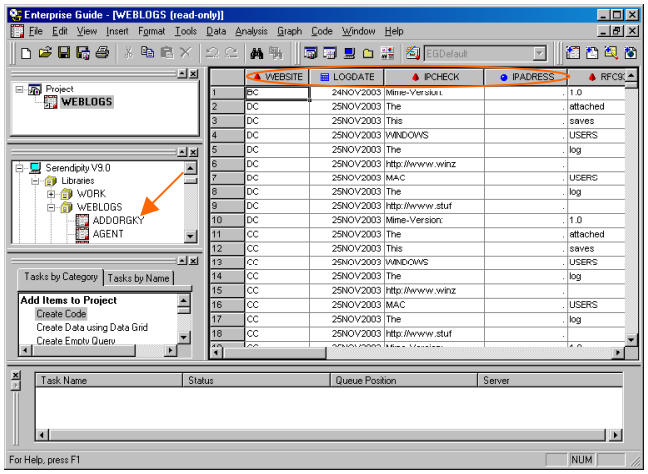

Figure 2
Under the library name “WEBLOGS”; there is a list of tables, two of which are shown beneath the **orange arrow** in the middle pane.

We selected one of the tables towards the bottom of that list, and now the header line of our window states “WEBLOGS (read-only)”.

This refers to the table we selected. It is now displayed for browsing in the right hand pane of our window. It is also shown as an object in the upper left hand pane, where it is shown as a member of a folder called “Project”.

The initial review we perform in our browse window is very important. The names of the columns in our table are displayed at the top of the pane; we can see four of these identified within an **orange oval**.

They include a coded column called “WEBSITE”, and what appear to be valid dates in the column called “LOGDATE”. However, the contents of the IPCHECK and IPADRESS columns don’t seem to be correct. The IPCHECK should contain a numeric octet like “127.0.0.1” which represents the Internet address of the web site visitor. The IPADRESS value should also be a number that is derived from this octet.

As we discussed the data with our users, we have immediately identified an issue that we will now need to resolve by going back to our original batch processes. We’ll use this as an object lesson for our users, and perform an analysis on the data error within our Enterprise Guide project. If this data is visited again in the future, then notes and existing code to quantify a data error are extremely valuable

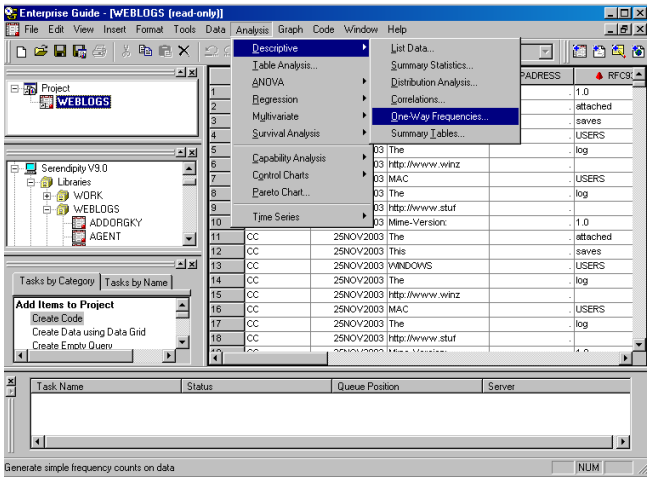

Figure 3
From our program menu, we select “Analysis”, then “Descriptive” and finally “One-Way frequencies…”. This will give us a windows interface from which we can test our data. For those familiar with the Frequency or Freq Procedure in Base SAS, this window is providing an interface to that procedure.

As we will see though, it also provides access to some basic SAS/Graph functionality that can also be very informative.

What we intend to do is summarise the IP addresses recorded in IPCHECK and find the proportion of the records that seem to be the textual values we can still see in the screen shot at figure 3.

The data we are going to analyse should, in the most part, start with a numeric value. So we can expect that all the invalid values will appear at the end of the frequency table. Our result set should summarise, in one screen, the incorrect values and the proportion of the table that is at fault.

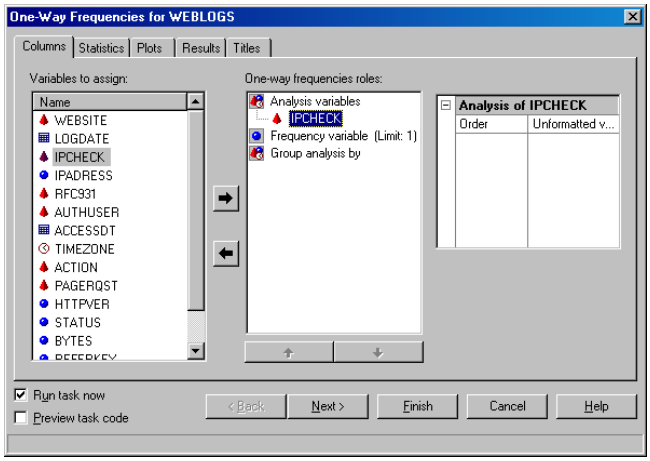

Figure 4
The user interface in Enterprise Guide opens at the column selection tab first. We can click on the required column and then drag it onto the "Analysis variables" role in the middle pane. Alternatively, we can also select the variable and then click on the arrow that points from the "variables to assign" pane to the "One-way frequencies roles".

Then we click on the selector tab called "statistics" at the top of the interface. From this tab we select the two options that refer to "missing values". We select to "Show frequencies" and "Include in calculations".

For those more familiar with writing Frequency Procedure code, this is the same as setting the option "Missing" in the tables statement, and will give us a clearer picture of the proportion of bad data.

When we have selected these options, click the "Finish" button at the bottom of the window. Since the "Run task now" option is checked (bottom left of window), the frequency analysis will be produced immediately.

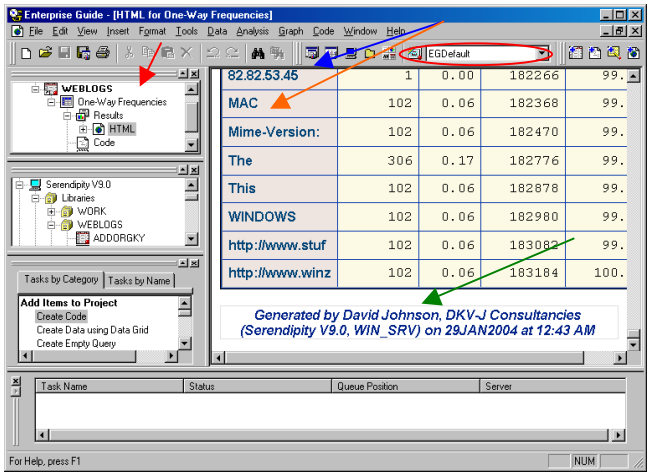

Figure 5
In our project window, there is now a task associated with the data table we selected. The red arrow points to the task "One-Way Frequencies", under which is a "Results" object that has an object called "HTML" associated with it. We can see that "HTML" is highlighted, and it is this object that is displayed in the right hand pane.

A blue arrow points to a valid IP address in the frequency table, and we can see that the invalid values, indicated with the orange arrow, are all quite a small proportion of the data.

At the end of the output is a footnote, indicated with a green arrow that includes the name of the SAS server that produced the output.

In this one summary then we can see the validity of the data, and we have a report that tells us not only when it was run, but who ran it and against which server resource. This is good documentation of the analysis process, and could be especially useful to others in the future.

The report generated is already attached to the project, and we will give it a better name shortly to preserve the analysis. However, if it is HTML, then this report will be readable on any machine with a web browser installed. So let's look at a way we can share this report immediately with other people who may not have SAS/Enterprise Guide.

SHARING OUR FIRST ANALYSIS, AND SAVING OUR PROJECT

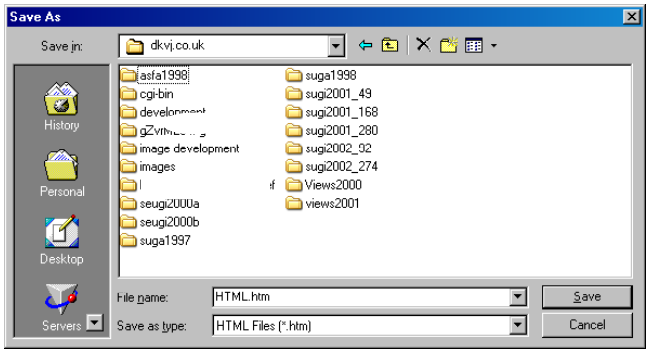

Figure 6
If the "HTML" object is highlighted in the project pane, then when we click on "File" on the program menu we will have an option called "Save HTML As/Export". Clicking this option brings up the dialog box we see here.

We can then select a folder, and save the object with any name we choose.

In this case, the folder is the mirror folder for one of our websites. By saving it here we can use our website synchronisation routines to send the output up to the live website.

When the output is saved, the HTML includes a copy of the style definition we used for the display of the report. Enterprise Guide uses styles that are really Cascading Style Sheets (CSS) that define the appearance of the results.

If you look back at figure 5, you'll see a red oval has been drawn around a selection box displaying the name "EGDefault". This is a selector box that allows us to choose a style for our output. We can customise and share these styles with other users if we wish to create a Corporate standard for all our reports.

Let's look at the other elements of our project now. In Figure 7 we see our application window with an expanded project area to better display the elements of the project.

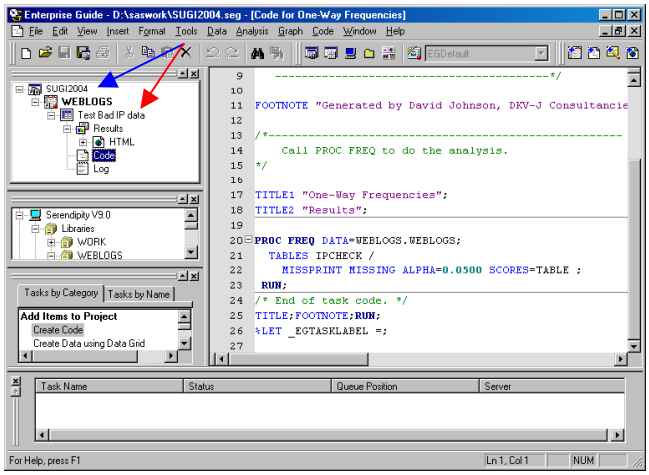

Figure 7

We'll look at some other things we can do to save a permanent project. Notice the blue arrow points to a word "SUGI2004" where previously we saw the word "Project". By saving the project we have given it a permanent name. The name now also appears in the Window title bar.

It is likely that we may do more frequency analyses, so the object "One-way frequency analysis" has been renamed (red arrow) to something that indicates what we were actually testing.

Notice too that since the project window has been stretched a little, we can now see that the analysis has two other objects called "code" and "log".

"Code" is highlighted and appears in the main window. The code is standard SAS code, with titling, footnoting and a call to a procedure that is not much different to the code you may write in base SAS. Aside from saving your output to an external file, and saving your project to an Enterprise Guide Project file, you can also access the code that was run and copy and paste this anywhere you wish.

When I came to this point the first time I used Enterprise Guide, I realised how easy it would be to share pieces of work with other SAS users of all levels of experience. If you are an experienced SAS user, it is likely you will recognise the colour coding of the code in the project window.

In fact, this code window is a full-blown SAS program editor, and you can make any changes you wish directly to the code. While generating code can be a point and click exercise, as the code becomes more complex than the interface allows, you can then edit the code directly and submit it to the same SAS servers.

In the case of some more difficult procedures such as you would use in complex statistical analyses, and to a lesser extent complex tabulations or graphical displays, the interface writes code for us that might have been tedious to develop from scratch ourselves.

THE FIRST USER REPORT: PAGE FAILURES

The first analysis requested by the team is the names of the pages that were requested from the web server, and were not found. Experienced web users will recognise this as a status 404 message. They then want a simple summary of the number of times each missing resource was requested and then tabulation by month of date requested.

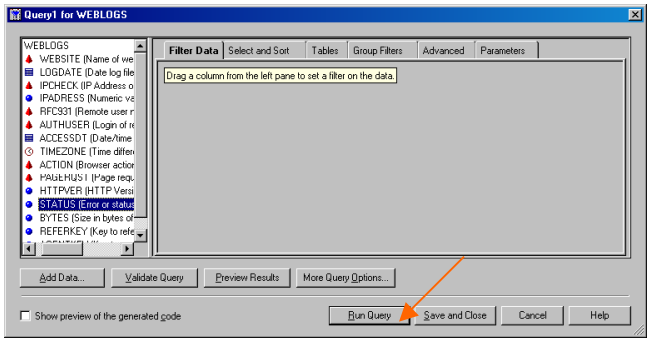

Figure 8

Since we only want to see records where the status value is 404, we will apply a filter to the table before we analyse it.

We filter the data by clicking on "Data" and "Filter…" on the program menu.

The dialog pictured here appears, and we select "Status", which is highlighted in the left hand pane.

We then drag it onto the working area under the tab called "Filter data".

This brings up the query builder window.

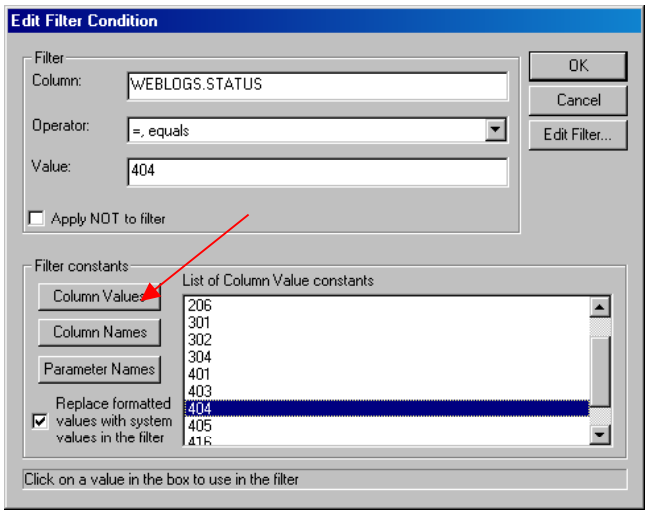

Figure 9

The name of the column is automatically populated with the table and column name from the previous window. By clicking on the button "Column values", indicated with the red arrow, we get a list of all values found for this column.

For large tables this will be preceded by a warning since Enterprise Guide will request the SAS server produce a list of distinct values.

From what we see in other parts of the application, it seems that it transparently runs a Query in SQL on the SAS server and surfaces the results to the query window.

In this case the table is not too big, and by clicking on the required value, it is placed in the "Value" entry field. If we wanted to exclude certain values, we could set the check box called "Apply NOT to filter", and then select only those values we did not want to see.

When we click the OK button, we are returned to the previous

window, where we can click the button called “Run Query”, which is marked with an orange arrow.

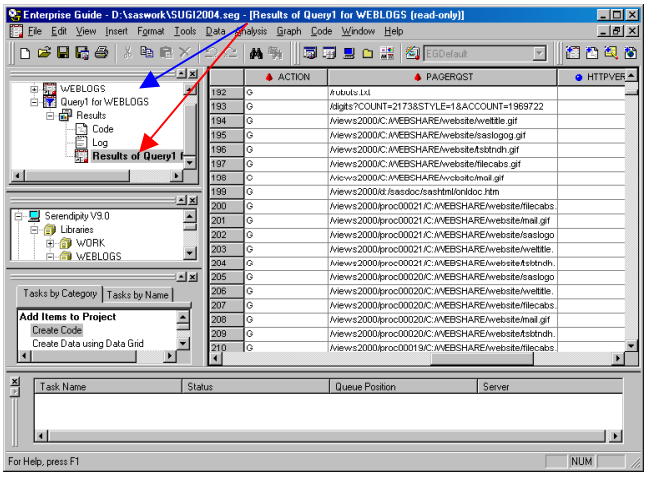

Figure 10

Our project now includes an object called “Query1 for WEBLOGS”, marked with a blue arrow, which we will rename shortly to identify the selection we performed.

Just as our frequency analysis had a section called “Results”, so this query does as well. The first two entries are the code generated for the filter we have applied to the data, and the log from that process.

The third element of the “Results” set is the outcome of the query, which has been named “Results of Query1 for WEBLOGS”. As the icon indicates, this is a table, which has been opened in browse mode in the right hand pane.

To keep our project informative, we then rename the “Query1 for WEBLOGS” object to “Select 404 records”. The table we created in the results set is also renamed to “Result_404”

With this object highlighted, we will now define some analysis of the data.

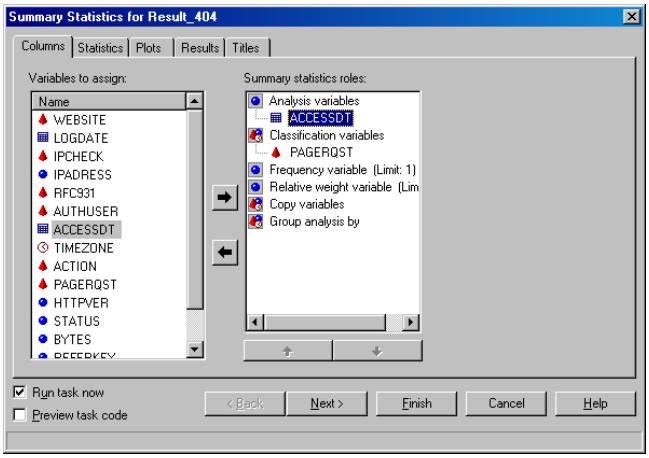

Figure 11

The results we want are for each individual resource that was requested. We will define the column “PAGERQST” to be a classification variable.

We want to know the earliest and latest use of the page, and the number of times it was used.

For an analysis we need a numeric variable, and the date of the request “ACCESSDT” will provide that information.

We then click on the “Statistics” tab to define the required analyses.

Since this is a general review of the data, we probably would not define any other parameters. However, if we wanted to see results collated for each website, then we would add “WEBSITE” as a grouping variable in the role “Group analyses by”.

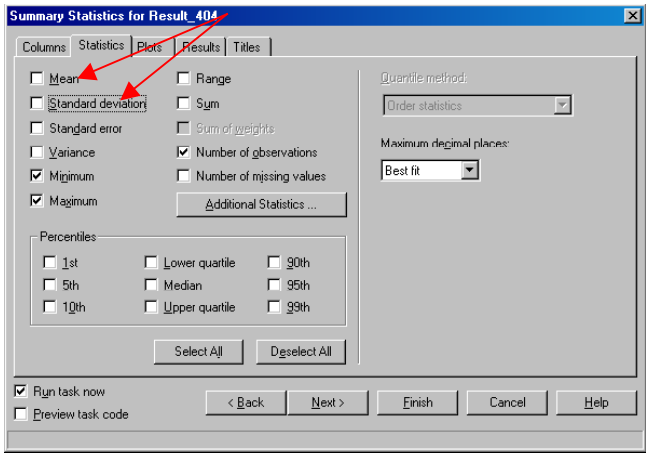

Figure 12

By default, the “Mean” and “Standard deviation” boxes are ticked, indicated by the red arrows. We remove these selections since they will not produce any meaningful result.

The “Minimum”, “Maximum” and “Number of observations” statistics are also selected by default. These will satisfy our query.

Since these are results required by our main users, we will modify the titles and footnotes on the pages to ensure they are appropriately descriptive.

To do this we click on the tab marked “Titles”.

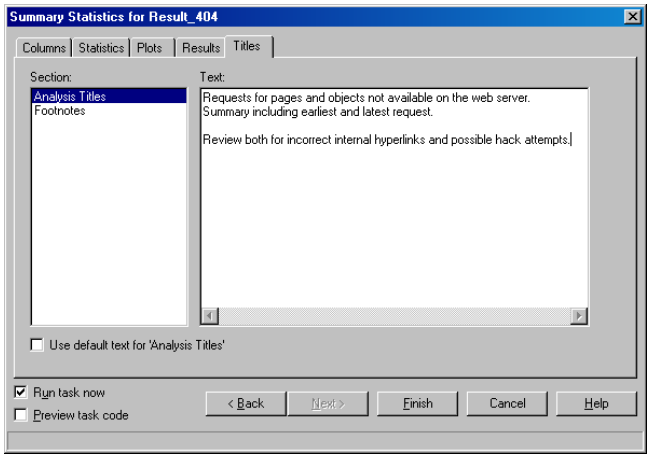

Figure 13
The title window provides titles based on the output requested. Since we have requested one analysis, we have one analysis title set, as well as a footnote.

The titles we define replace the titles we might have defined in base SAS as “Title1”, “Title2” and so on. In this case we have provided two titles followed by a blank line and another title.

This will result in a two-line header describing the general project. After a break there will be a specific message relating to the report. We would previously have done this with title statements 1, 2 and 4.

This approach also allows me to create similar reports based on the same data or summary, but with certain parts selected. All reports may then have common first and second titles, and a definitive fourth statement.

I haven’t changed the footnote because this was defined administratively and captures the data we saw above identifying the person and resources used to create the report. From my experience, I think footnotes should be standardised to reflect the data validity and ownership of the report. The footnote should not be changed without good reason, although it may be enhanced if specific data issues need to be highlighted.

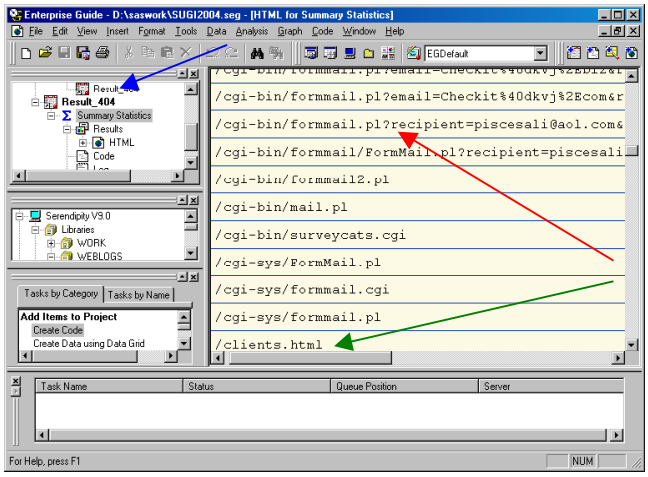

Figure 14

When we ran the report it produced a table, part of which we can see here in the results pane. The execution process meant that we were using the outcome of one process to run another, so Enterprise Guide created a new reference to our data set at the top level of the project, marked with a blue arrow.

The process run is identified with the name “Summary Statistics”, and the usual elements of output code and log follow. There is a problem with the output that we shall look at in a moment, but there are some interesting results available already.

On our website we have a problem with a referenced page that is not available, which is the “clients.html” page marked with a green arrow.

Also of interest is that someone has attempted to crack the website to create an open mail relay. The entries referring to “formmail.pl” are all cracks that would send an email to the perpetrator.

Based on this email they would have found a vulnerability in the website which they could use to generate spam that would have appeared to have come from my website. Needless to say, the crack attempt has failed. The people concerned also no longer have those email addresses. My primary log system generated emails I sent to their webmasters asking their accounts be closed. We could all benefit from a little less spam.

DATE TIME ANALYSIS IN ENTERPRISE GUIDE

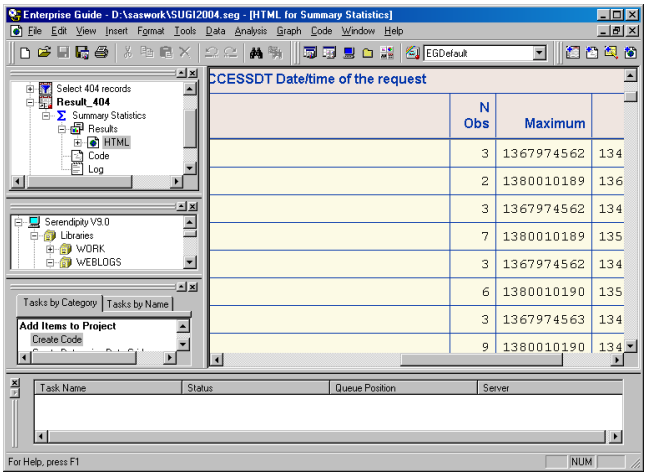

Figure 15

The problem with the output is clear when you look at the minimum and maximum access date values.

These numbers are SAS date times, but the format is being expressed in seconds since 1 January 1960.

The problem is documented in SAS Note 006167 and is a vulnerability in Enterprise Guide. The note makes the suggestion to save the results to a table and alter the properties of the table to include the correct formats.

While the limitation in the current release of Enterprise Guide regarding formatting of dates is an issue, it gives us the excuse to use a conventional method for the next and last analysis we will review.

The website team want to know the busy and quiet times on the website by time of day and also by day of week. To perform this analysis we will create a copy of the original data with two additional columns containing the required classification variables.

While we could use the SAS format Tod. to display the time of day from a SAS date time, for a day of week we need to apply the function WeekDay() to the function DatePart() on the date time. Once the date time format issue is resolved, we will still have to

transform the data to find the busy days of the week.

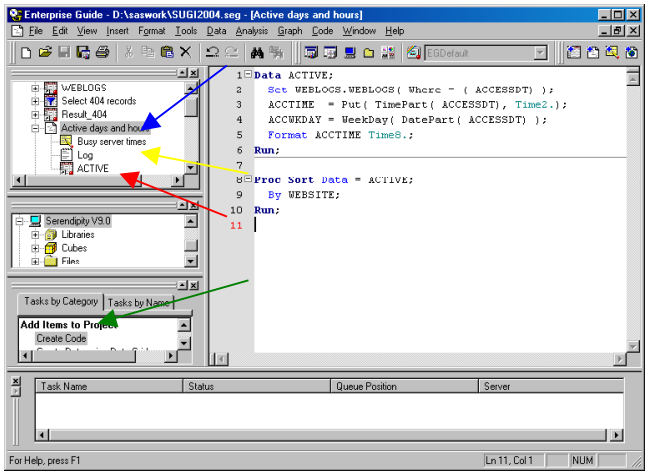

Figure 16

To create a new translated table we will create a code object, which is simplest to do by double clicking the “Create Code” task we can see highlighted with the green arrow. We then write code to create a temporary table that includes the derived values. We have included a sort step as well since we will probably analyse the data individually against each website.

For clarity of the project we have renamed the code object, marked with the blue arrow, to reflect the type of data we are trying to retrieve.

When we have finished writing the code, it can be run by clicking “Code” and “Run” on the program menu, or by right clicking the code task and selecting “Run”. I prefer the right click method because I can also select the server to run the code from the same right click menu.

If we now highlight the new table “ACTIVE”, indicated by the red arrow, then we can perform analyses on the new columns.

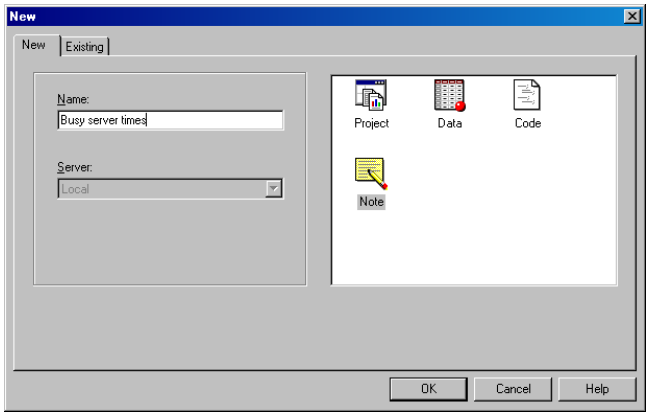

Figure 17

Before we do that however, we will add a notation to the code to discuss the purpose of the analysis. Those of us who are SAS programmers are used to adding comment blocks to our code, and that option is still open to us.

However, for statements regarding the purpose of the code object, or the nature of the data, these comments may impede the readability of the code object.

Since we design the structure of the project to assist users of all levels of SAS experience, the readability of the code is important, so I tend to use another documentation method as well in Enterprise Guide.

We can add a “Note” to most objects in the project, so we start by selecting the “Active days and hours” code task as shown in the previous screen capture. From the program menu we select “File” and “New” to get this dialog box. We then select “Note” and give it a more informative name than the default name that appears.

A note is then created against my code task in which we can add comments about the code. The note is coloured yellow like a “sticky note”, but you can only write simple text: that is, there is no facility to highlight words or phrases with other colours of text, or by emphasising the font. Despite this limitation, this is a very useful way to provide assistance to the next person who will use or review the project. You can see the note in the preceding screen capture, identified with the yellow arrow.

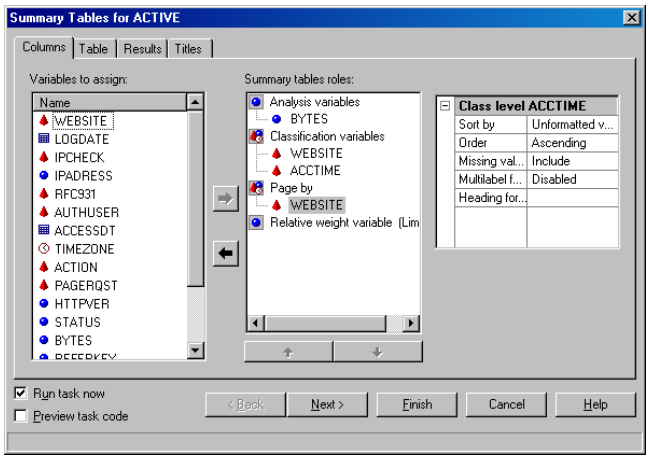

Figure 18

To complete our analysis we will generate a table of the time of day or the day of week against the number and quantum of data transferred. We will produce one table for each website.

We select “Analysis”, “Descriptive”, “Summary Tables…” from the program menu and this dialog box appears. Since we will produce a page for each website, we drag the “WEBSITE” variable to the “Page by” role. Enterprise Guide will then also insert the variable as a classification variable.

The classification we want to define for each page will be the access time, so we drag “ACCTIME” into the “classification variables” role. Finally, the variable we want to analyse, “BYTES” is dragged into the “Analysis variables” role.

Then we select the “Table” tab to begin the design of our output table.

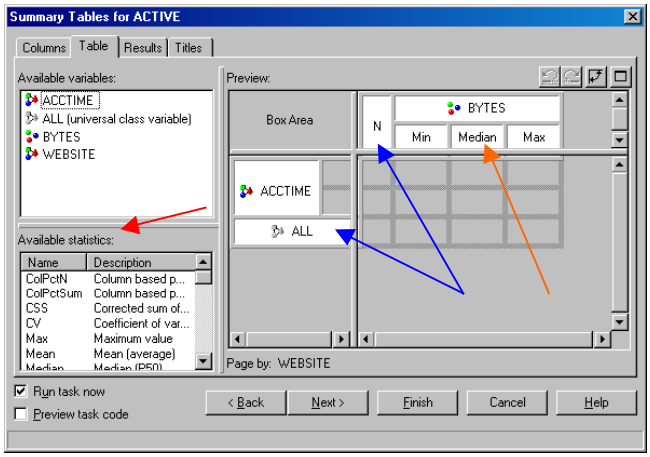

Figure 19

By default, Enterprise Guide defines "N" as a column variable. It also defines "All" as a row variable where missing values might be encountered. The two defaults are marked with blue arrows.

Either definition can be removed, but they are valuable in this case so they were retained.

The layout we will use has the time defining the rows of the table, so we drag "ACCTIME" from the "Available variables" pane into the "Preview" pane and place it above the "All" definition.

Similarly, we associate "Bytes" with the column definition, which Enterprise Guide does leaving a blank row in the column definition area.

Into this we can drag the required statistics from the area marked with the red arrow, and place them in the area marked with the orange arrow.

Then we select the "Results" tab and check the box marked "Save results to a dataset". A name will be provided by default. This table will allow us to generate other displays of the data later. Finally we can, and should modify the default titles to reflect the purpose of the analysis, and the nature of the data.

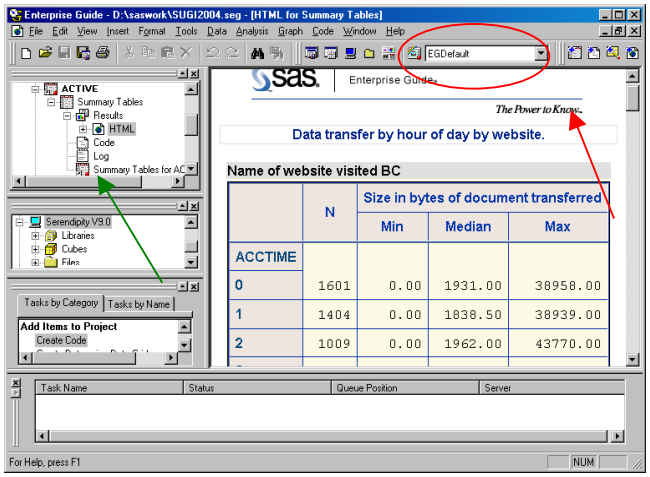

Figure 20

A section of the resultant table is shown here. The task has also generated a summary table, indicated by the green arrow. We will use this in the next step to translate the tabular display to a graphical one.

This is also our first screen shot to show the top of HTML output created with the "EGDefault" output style. (Remember, that is the style in the selection box inside the red ellipse?)

At the top of the page is a graphic object containing the SAS logo and some text. We can replace this object with our own corporate or group logo if we wish by using the style manager functionality of Enterprise Guide. We will look at that later in this paper.

First though, let's generate a graphic representation of the site activity with the output table, which has been renamed to "ACTIVSUMM". (Renaming of objects is most easily done by selecting the object, right clicking it, and selecting "rename" from the menu that appears.

A PICTURE IS WORTH…

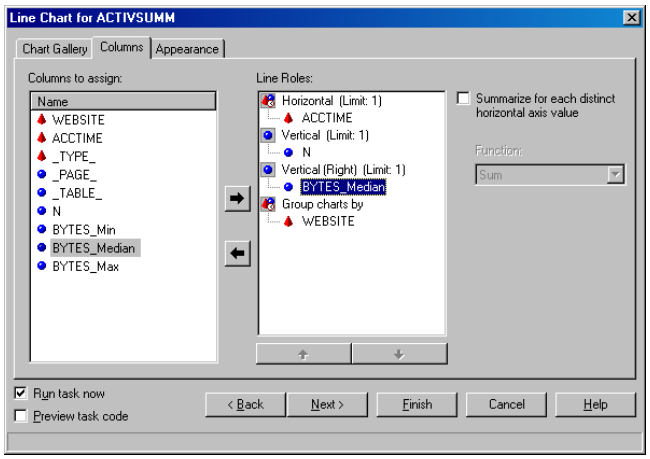

Figure 21

With the table highlighted in the above results, we select "Graph" and "Line…" from the program menu. This screen appears and we select the "Columns" tab to begin the definition process.

We want one chart for each website, so we drag the "WEBSITE" variable over to the "Group charts by" role. We want our horizontal graph vector to show the time of day, so we drag "ACCTIME" onto the "Horizontal" role.

Into the first "vertical" role we drag the "N" statistic since this will give us the number of file transfers in each hour. The SAS/Graph engine provides the functionality to overlay a second line, so we then drag the "BYTES_median" statistic onto the second vertical role.

You should then explore the "Appearance" tab, which is actually a collection of other tabs. No default title is defined, so this can be added here, as well as the usual attributes of colour, symbol, reference marks, legend and so on.

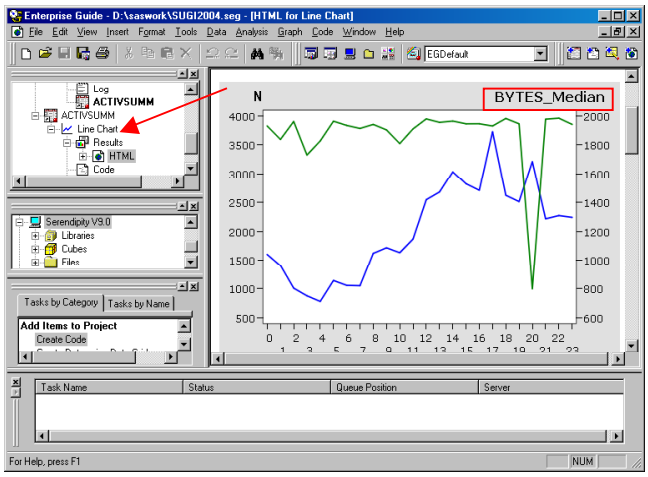

Figure 22

The output graph for one of the sites immediately demonstrates that the site is busiest in the afternoon and evening, as demonstrated by the number of requests shown in the blue line.

The size of the requests, as demonstrated in the green line, is reasonably consistent, suggesting there is little change in the type of request made by time of day, especially in the busiest time.

However, there is a substantial variation at around 8pm that looks like a high number of requests, with a low median request size.

We might form the suspicion that the site has been subjected to a denial of service attack where a small object on the site was repeatedly and frequently requested. Based on what we see here, we would now review the data for the timing, request and source of the request.

To clarify this graph, we would double click on the definition of the task. That is the entry called “Line chart” and indicated by a red arrow.

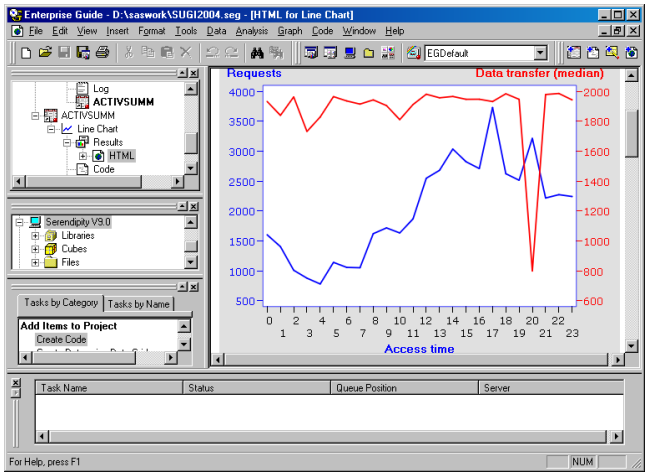

Figure 23

Next we will access the “Appearance” tab on the graph.

We might change the colours chosen by default to better differentiate between the two plots.

We could change the colour of the right hand axis title to the same colour as the plot line, and also the axis titles

We can programmatically do this with SAS/Graph procedures, but the GUI is easier to work with for testing and making changes. Once we are satisfied with the appearance, we might then copy the generated code and use it elsewhere.

For the advanced SAS programmer, it is an easy way to create code for less familiar SAS products. As a long time fan of the SAS/Graph product, I have spent a lot of time writing and tuning Graph code. Now the results can be shared in less time, with other programmers who are less experienced in the product.

LOOKING FOR THE EXCEPTIONS

Earlier in this project we included a block of code to perform some data preparation. We will finish looking at our users requirements by adding another block of custom code. We’ll consider how we might handle exceptional situations, such as the unusual drop in data transfer above. In a batch process we would need to take these steps to generate an exception report:

- Analyse the summary data for usual performance characteristics.
- Identify any period of exceptional performance.
- Select records matching the period.
- Summarise those records in a meaningful manner.
- Match that summary to a “usual” performance.
- Display the comparison.

We will use a very simple analysis here; we’ll look for the mean and standard deviation of the transfer rates, and then identify any value that is more than two standard deviations away from the mean. This analysis makes a number of assumptions that are poor in a purist statistical sense, but since we only want indicators of what might be unusual behaviour, and we can examine the data ourselves later, this approach is reasonable. You’ll note also that we are seeking “usual” rather than “normal” performance; “normal” has a particular meaning in statistical analysis.

We’ll use the Means Procedure to extract the statistics on our data. Here is the summary code we will use.

```
Proc Means Data = SASUSER.TAB19058
    FW = 12  VarDef = DF NoPrint;
  By WEBSITE;
  Var BYTES_Median;
  Output Out = TESTACTIV  Min = MIN
         Max = MAX   Mean = MEAN   StdDev = STDDEV;
Run;
```

Then we’ll calculate the upper and lower bounds of “usual” data based on these analyses and store these to macro symbols.

```
Proc Sql _Method  STimer  NoPrint;

  Select Put( Sum( MEAN, ( STD * 2) ), 6.),
         Put( Sum( MEAN, ( STD * 2) * -1 ), 6.)
         Into :MMaxVal,
              :MMinVal
  From TRANSTEST
  Where WEBSITE = "&MSite";

Quit;
```

We select the data for the site we are analysing and flag those hours outside the usual limits with the label “Extreme”, and the other hours with the label “Usual”. In this way we have also built the control table for a format that we can use to label the data in our next summary.

```
Data EXCESS;
  Set SASUSER.TAB19058( Where = ( WEBSITE = "&MSite") );
  FMTNAME = 'FTestFm';
  TYPE    = 'C';
  If &MMinVal <= BYTES_Median <= &MMaxVal  Then
        LABEL = 'Usual  ';
  Else  LABEL = 'Extreme';
  START = Put( ACCTIME, 2.);
  END   = Put( ACCTIME, 2.);
Run;

Proc Format CntlIn = EXCESS;
Run;
```

Then we summarise the source data using the formats we have defined to group the hours according to their being “usual” or “extreme”.

```
Proc Means Data = ACTIVE  FW = 12  VarDef = DF
    NoPrint  NWay;
  Where WEBSITE = "&MSite";
  Class ACCTIME;
  Var BYTES;
  Output Out = STATACTIV N = NUM   Min = MIN
       Max = MAX   Mean = MEAN   Median = MEDIAN
       Q1  = Q1    Q3   = Q3     StdDev = STD;
  Format ACCTIME $FTestFm.;
Run;
```

Our last step is then to tabulate the data or graph it to display the performance of the web site.

APPEARANCE MATTERS…

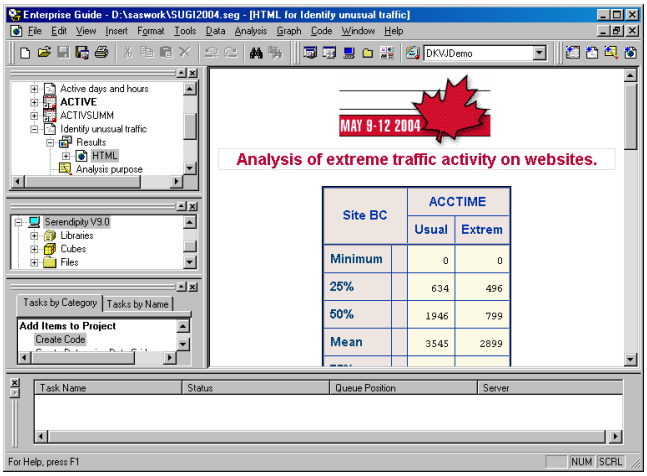

Figure 24

Here then is the table for one of our sites showing the first few comparative statistics. This sort of analysis is usually quite specific to a corporate need, so we won’t be concerned too much with the content now, but look at the appearance of the output.

The image at the top is specified as part of a style sheet we have called DKVJDemo, the name is shown immediately above the maple leaf image.

The colour of the first title has also been changed to match the header, and indeed all elements of the page could have been changed as well.

The key point about this screen capture is that the output has been customised for its audience; in this case the SUGI conference attendees. Changing the appearance is simply a matter of selecting a new style from the drop down list immediately above the output pane.

If we click on “Tools” and “Style Manager…” on the program menu then the following screen appears.

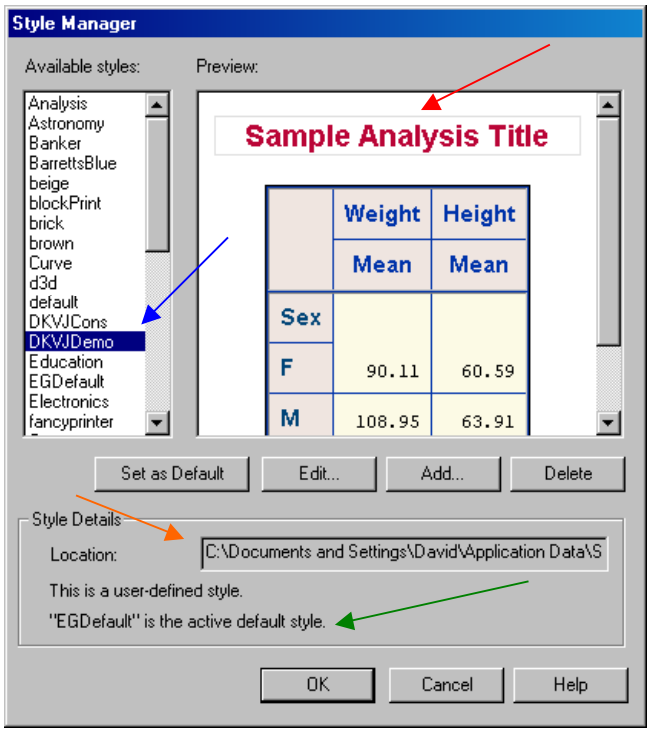

Figure 25

The style we are reviewing is identified with the blue arrow. We can recognise the scarlet title indicated with the red arrow.

The style we are using is saved in a user area as defined by the "Location" text entry indicated with the orange arrow. By default the Enterprise Guide standard styles are held in the installation directory, and the edited styles in the user directory. We can point to styles in shared network areas that would allow us to share new styles with a team.

Note the message about the default style, indicated with a green arrow. If we click on the button "Set as default" we can change the default to the DKVJDemo style for this user's installation of Enterprise Guide.

To modify the style we click the "Edit…" button that takes us to another screen.

Based on the style we started with we can now modify that style, or save it as a new style.

We can change most aspects of font, text size, style, colour and background as demonstrated in the next screen capture.

We can also redefine the lines and shading used for tables, which as we see here has shaded areas in the header areas.

The process can be time consuming, so the best solution is to select the style nearest to the result we want and save a modified version of that style.

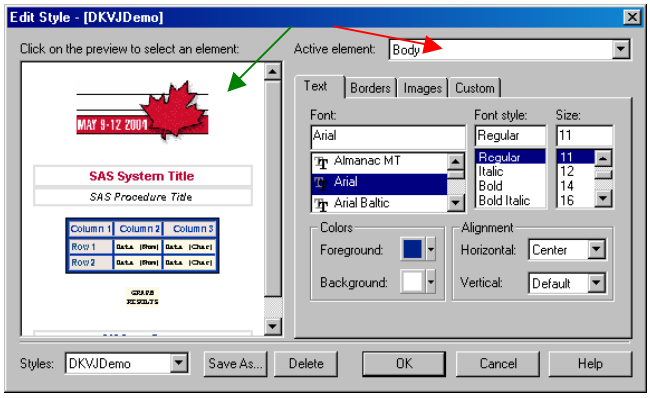

Figure 26

When we enter the edit screen, we see a preview of the final style, and the selectors for our font, colour and alignment settings.

To change a style element we can either select it by name from the "Active element" dialog indicated with a red arrow, or we can click on the element directly in the preview frame.

The "Borders" tab gives us access to table controls including style, weight and colour for each element.

On the "Images" tab we can select the banner image, indicated with a green arrow, and also set a background image for the whole output.

Be aware if using a custom image that it must be available to all readers of reports from Enterprise Guide. For this reason, it is best to use an image stored on a network drive, and it should be referenced with a UNC definition rather than a drive map. (A UNC, or Universal Naming Convention path might read like \\Serendipity\EntGuide rather than D:\)

BUILDING A REPORT DOCUMENT

In our previous steps we have created a series of reports that are all neatly held in an Enterprise Guide project. In the project any user of Enterprise Guide can review the tasks and output. If they have appropriate permissions for the project, then they can also rerun the tasks at any stage to refresh it for current data.

One of the most powerful aspects of Enterprise Guide is to deliver output to people who do not have the software. We have already looked at exporting one task report to HTML, now let's look at exporting more of the project. Naturally, we could export each individual element to it's own file and give a person the list of file names. Or we can take the following approach.

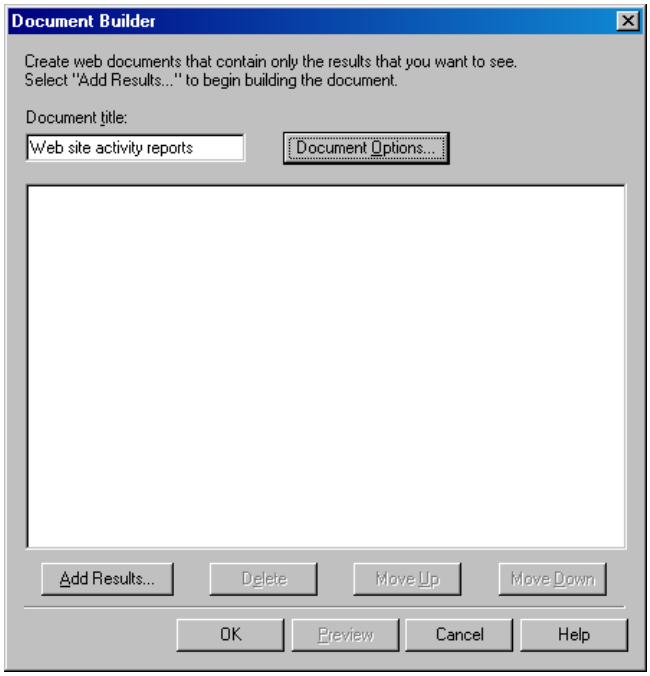

Figure 27

By clicking “Tools” and “Document builder…” on the program menu we bring up this dialog.

We will start by giving it a meaningful name, and then click on the “Document Options…” button.

That brings up another screen from which we can select the default style sheet for the whole report.

We can also request a table of contents to be generated. The table can be at the top of the document like an index page, or at the side of the document like a frame on a web site.

We accept the options we have chosen and then select the “Add results…” button.

A window like the following will then appear, and we select the appropriate pieces of our project to add to the document.

Note the “Move Up” and “Move Down” buttons that we can use to reorder the output if we aren’t happy with the sequence.

The “Preview” button will also allow us to review the appearance before finalising the document. All three buttons are currently greyed out because we have no output to select yet.

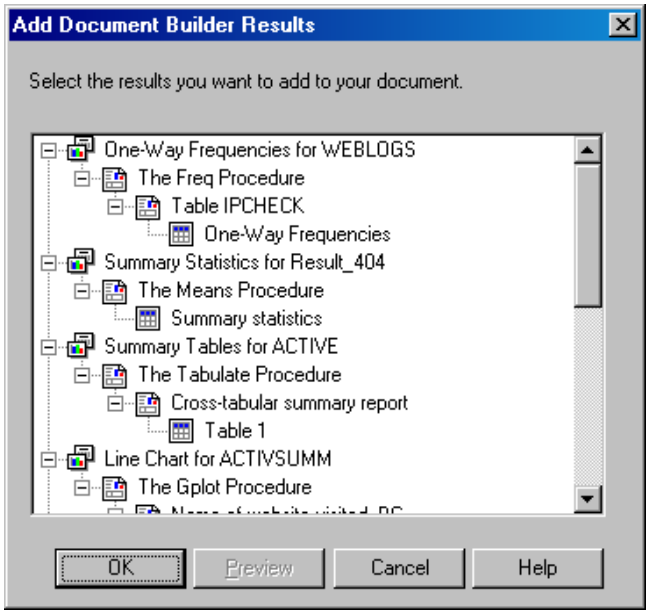

Figure 28

Each of the pieces of output is then indexed on a window.

We can select them individually, pressing OK after each one, or hold down the “Ctrl” button and select multiple elements.

When we click the “OK” button we are returned to the previous screen, with our output elements shown.

As we select elements we can also select the “Preview” button that will bring up our web browser with the requested output.

This will let us review again the content of individual elements.

Since the previous window allows us to reorder the elements, it is best to just select the elements here and modify the final document in the previous screen.

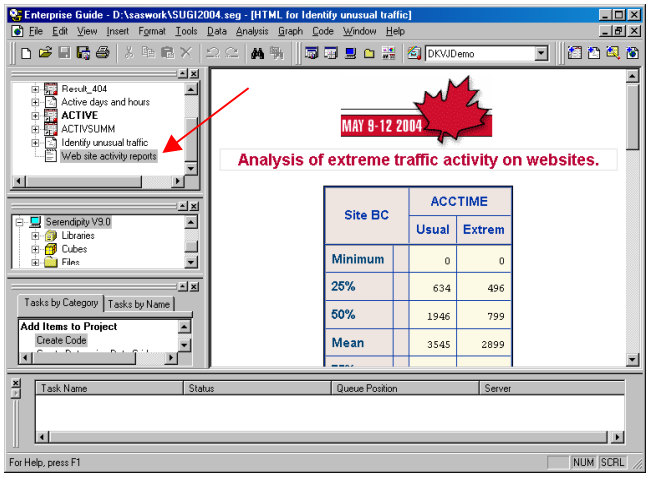

Figure 29

Our document is now part of our project and can be exported as an HTML object by right clicking the task and defining the path and file name.

As a process then, we have a series of tasks to review, test and report on data that result in a published report.

Any user now may open the project and run each of the tasks.

Clearly, for stable reports we would prefer that the project could be opened and all the tasks run in sequence.

Enterprise Guide does provide that functionality, and we’ll explore that next.

BUILDING A PROCESS

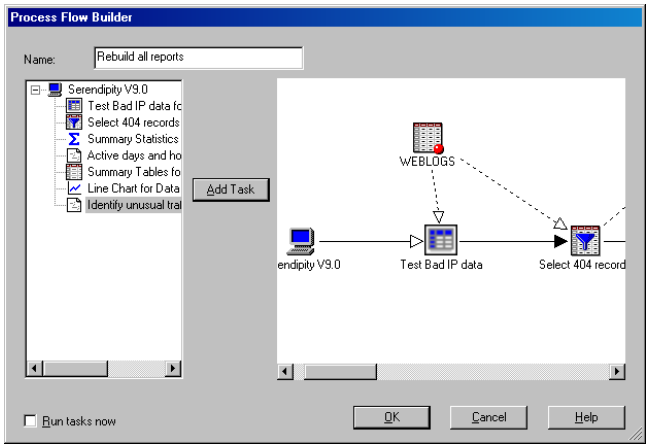

Figure 30

From the "Tasks" option on the program menu, we can select "Process Flow Builder…" which will bring up this screen.

All our tasks are shown on the left, and by selecting each in turn and clicking "Add task" we add them to the process flow diagram on the right.

Notice that the server that will run the job is defined first, followed by each task in order.

Where a task reads a table, it is shown above as an input, as is the table "WEBLOGS" in this diagram.

Where a task builds a table, it is shown in the same place, but the linking arrow is reversed. We can then clearly see which tasks depend on data from previous tasks.

Unfortunately, there is not yet any way to export this process flow to an external document. One can hope a later release of Enterprise Guide will address this shortfall.

Before leaving this screen, it is also a good idea to give it a meaningful name.

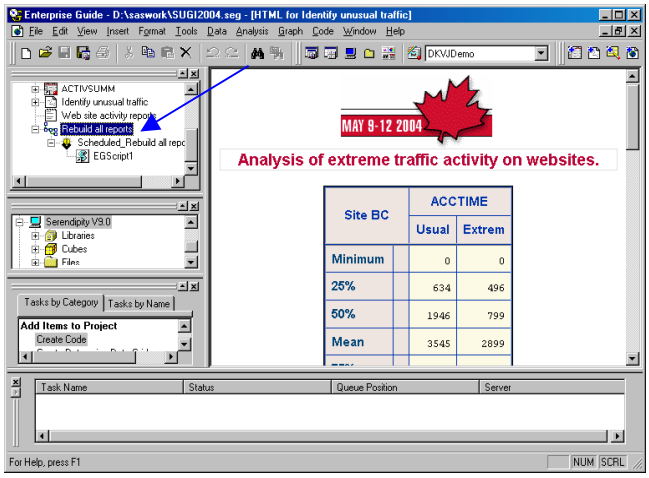

Figure 31

Once the process flow has been defined it is saved to the project as a task. Rerunning all the tasks in the process flow now only requires selecting the process flow.

If we right click the process flow object, we will also be offered the option to schedule the process. This allows us to automatically update the project regularly.

Selecting this option brings up the Windows scheduler. The Windows scheduler is largely self-explanatory, and can vary in appearance between different versions of Windows. If the scheduler isn't available then make sure you have a minimum of "Power User" status on the machine.

The frequency and time of running can then be set. On completion of scheduling a Visual Basic script is created in the same folder as the project. This script is run by the Windows scheduler to update the project.

The project will then be as current as the last scheduled run time when it is next opened.

EXPORTING FROM A BATCH PROJECT

Updating the exported report document will take a little more work. The schedule process has only updated the project, and while it is current to Enterprise Guide users, the exported HTML pages will not be. The reason is that the export process has been run manually. You can however modify the schedule to update the exported HTML as well. This is documented in SAS Note SN-002696. The process involves modifying the Visual Basic script to perform a copy of the project object to an external file.

I have modified the Enterprise Guide generated script to copy the Document object from the project to a new file location. The following excerpt shows the originals code elements (in black), comments I have added (in green) and new code (in blue).

```
    If not (PFDObject Is Nothing) Then
        '----
        ' Run the PFD
        '----
        PFDObject.Run
        If Checkerror("ProcessFlow.Run") = True Then
            Exit Sub
        End If
    End If


    '---- Manual changes to get HTML export.  DHJ 1/2/03
```

```
If Err.Number <> 0 Then MsgBox "Error prior to objResults definition"
 '---- Get the results
 Dim objResults
 Set objResults = prjObject.ProjectItems("Web site activity reports")
If Err.Number <> 0 Then MsgBox "Error on definition of objResults"

 '---- Save the results to a file on the local server
 objResults.SaveAs ( "D:\Saswork\Weblogs\SUGI2004.htm")

If Err.Number <> 0 Then MsgBox "Error on save of objResults"
 '---- End manual changes to export results.


 '-----
 ' Save the new project
 '-----
 prjObject.Save
 If Checkerror("Project.Save") = True Then
     Exit Sub
 End If
```

The statements "`If Err.Number <> 0 Then MsgBox …`" are for debugging purposes. They trap any errors caused by incorrect definition of the project object or the output file destination. They should be removed or commented out once the code is working satisfactorily.

You may note as well that the code provided in the SAS Note refers to a class called "objProject". However, Enterprise Guide Version 2 creates a script where the primary class is named "prjObject". Blindly applying the modifications in the SAS Note will result in code that won't work. For that reason I always add the diagnostic messages to trap errors that may be caused by my changes.

While knowledge of Visual Basic language and scripting would assist in making the changes to the script, the changes are moderate and could easily be made by most users.

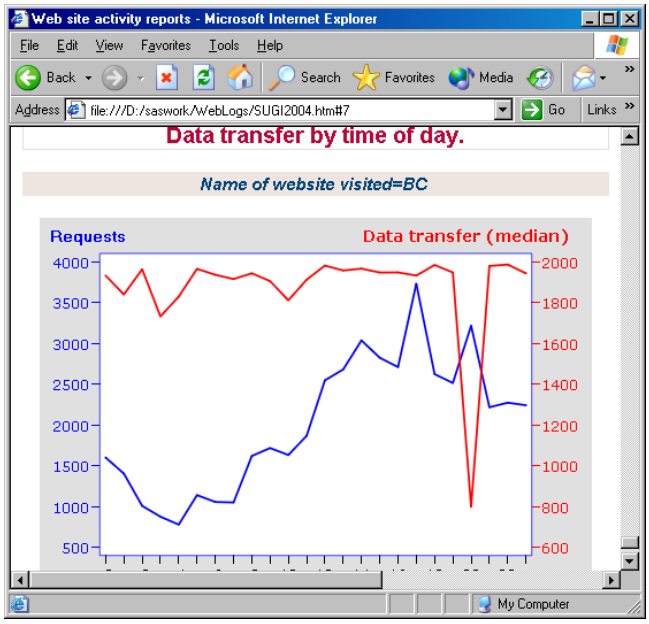

Figure 32

Here is the demonstration of the script changes working correctly: the output document viewed in a web browser.

Note that the title is still in the shade of crimson we defined for the DKVJDemo style sheet. When Enterprise Guide exports the document to HTML, it embeds the style sheet elements in the HTML.

This means that you can develop your project and share it with other Enterprise Guide users on a network where you all share access to the project file, data and style sheets, and then bundle a single file for other interested parties to view.

If you choose to use a banner image in your style sheet however, then you will need to provide a copy of the banner image in the appropriate directory on your web server. You will also need to modify one line of the HTML to point to the new location.

From the original HTML, here is the definition of the banner image.

```
.EGBanner {
    Background-image:url("file://C:/Documents
and Settings/David/My
Documents/sugi2004.gif");
```

The modified line on my web server would now read:

```
.EGBanner {
    Background-image:url("../images/sugi2004.gif");
```

SECURING THE DATA

Assembling analyses and data tests with reports isn't just valuable to our primary users. In the sample analyses we saw, we encountered possible evidence of attempts to crack and attack our site. These issues immediately caught the attention of our Business Information Security Officer, who wanted his own view of the data and its issues.

Once the process of selecting data and building reports was demonstrated, he was able to perform his own queries with a copy of the project. However, the strength of Enterprise Guide lies in co-operative analysis and information sharing. So the reports of interest to him were added to this project and another document built for the knowledge he sought.

Changes to the batch process exported his reports as regularly as the batch was run, and one project now met the requirements of two functional areas.

ANALYSIS BY MANAGEMENT

The Information Systems Manager also took an interest in the project when he saw the potential for:

- Data sharing
- Data analysis by users (saving his limited analyst team)
- Co-operative and collaborative reporting
- Information delivery from thin clients which freed budgetary resources for more powerful SAS servers
- Knowledge surfaced to his desktop that didn't require SAS programming skills

She produced some analyses on the data exploring the use of the sites and the traffic generated as well as checking the popularity of our service reports and pages. Her findings contributed to decisions she made on his staff and resource allocation.

At no time did she write a single line of SAS code. Similarly, the BISO (who despite being well versed in DCOM and valuable in assisting with configuration of SAS/Integration technologies), was wholly unversed in SAS.

SUMMARY

We have looked at the following functionality in Enterprise Guide:

- Creating a project where we read a simple table and produce an HTML page we saved for our non-SAS colleagues.
- Creating a more complex analysis project where we have filtered out a sample of data and performed some simple analysis before we save or publish our results.
- Assembling that code into a Process Flow that we then scheduled to run automatically.
- Delivering a set of reports to a web server.
- Sharing data findings with other disparate functional areas.
- Contributing to the informed decisions made by management on resources.
- Encapsulating the work of a multi disciplinary team of people for the benefit of other interested parties.

TRADEMARK CITATION

ACKNOWLEDGEMENTS

The client who saw the potential for Enterprise Guide and set some great challenges to overcome.

SAS Institute for providing two fine training resources to get clients started.

- http://support.sas.com/training/elearn/tutorials/v8/eg/ is the self paced e-learning guide
- http://support.sas.com/sassamples/bitsandbytes/03jul/usingeg1.html, my favourite starters guide.

My family for their support, patience and late night coffee pots.

CONTACT INFORMATION

Your comments, suggestions and questions are valued and encouraged. Please contact the author:

David Johnson FRSA
DKV-J Consultancies
4 Bataba St
Moorabbin Vic
Australia 3189
Work Phone: +61 (0) 3 9553 0301
Fax: +44 (0) 7005 98 0829
Email: sugi234@dkvj.biz
Web: http://www.dkvj.com