From the Neolithic to the Nineties
The Evolution of Capacity & Performance Reporting
Douglas Clarke and Mark Hayden, General Accident (UK)

Abstract

We all collect vast amounts of data for CPE, but how to turn this into "information"? Over the past three years the Capacity Planning team at General Accident have evolved this from an almost prehistoric technological level into a coherent, online reference system using a SAS/AF® GUI application, whilst remaining firmly rooted in SAS® and MXG®.

"The way we were" centred upon voluminous (b&w) paper reports which were both unreadable and unread, suffered from inconsistent "workload" definitions, and took forever to produce. We are now able to present meaningful (and usable) information to managers, online at their own desks. This MI is readily available and quickly updated, and is presented through a "manager-friendly" PC-based system.

The evolutionary path involved structuring workloads, combining data from multiple monitors, standardizing reports, and (eventually) building the Capacity/MI FRAME system for PC/LAN. The needs of our customers were kept firmly in mind, and the structure was designed for an evolving service management infrastructure with its own demands for information.

This presentation focuses on how a bunch of prehistoric cavepeople have taken these evolutionary steps, and the lessons we learned in the process.

Introduction

General Accident (UK) is the British general insurance arm of the General Accident group, based at the world headquarters in Perth, Scotland. Our computing is mainly based on IBM® 3090-J technology in a predominantly MVS® environment, with two "production" machines, two lesser (VM®-based) mainframes, and some 28 major "workloads" (plus development and test). We have a Capacity Planning team of 3, dealing with performance analysis, service reporting and capacity planning. Our data sources include MVS monitors (SMF®, RMF®) and The Monitor® for CICS®, processed by Merrill's MXG software, with an ever-increasing array of locally-developed SAS programs to analyze and present the information.

Our Problems

Capacity Planning reports were a major monthly ordeal. A large battery of batch jobs had to be run to produce a huge volume of laser-printed graphs in black & white, which, being on paper, were then split, collated, and bound in a bulky maroon A4 folder for circulation around IT management. Our problem was that the reports were too bulky and hard to understand; we were providing our managers with data, not information - data which they had not the skills, time or inclination to interpret. It was very easy to miss key points of information. The reports were often out of date, as certain managers tended to "sit" on the maroon folder without reading it. It was not unknown for us to search it out only to find that nobody had read it that month!

The suite of SAS programs which produced all these treesworth of unloved reports had grown organically, and was now very cumbersome. Different programming styles and archaic data structures made them awkward to maintain, and extremely difficult to learn. An example is the monthly CPU file, having one observation per month and one variable per resource per "workload" - cf. Foil 04.

Adding a new workload to this data meant adding a new variable to the data, to the program which created it, and to every program which referenced it. The scope for error was huge. The "workloads" were not clearly defined or kept standard. Documentation was negligible, largely confined to a few cryptic comments inside the SAS programs. As we had no online mainframe graphics capability, any new or revised graph had to undergo a painful period of development, especially waiting for prints to come back (usually the next day).
Why change?

Re-read the above!

What we needed was a low-maintenance reporting suite, easily updated for new application systems, and to improve the quality of the reports we wanted to change from b&w to colour graphics, but what we really wanted to be able to do was to put the graphs out online, at the IT managers' own desks. After a bit of thought we came down to four possible ways forward:

1) Continue as we were.
   That is, just go on as before, putting up with the overheads, making little improvements when and where we could. The upside of this was that it would cost nothing extra and had no change issues. The downside was self-evident: we wouldn't be getting away from any of our existing problems.

2) Buy in a package.
   This option would have the advantages of a quick delivery, minimal maintenance and probably colour graphics. Against it was a bit of "rightsizing", generating all the MXG PDBs on the mainframe, then extracting workload-oriented PDBs which could be downloaded monthly to SAS on PC. From LAN-based SAS data sets we could then produce colour graphs to illustrate all our major metrics, and store these in SAS catalogs. PC printer technology would enable us to print colour graphs for IT management until such time as we could deliver the output at their own terminals. The benefits of this approach would be the availability of on-line and colour graphics, integration of forecast data, and ease of developing new graphs. The problems would be that we were unfamiliar with PCs, and would have the burden of downloading the data.

3) Redevelop the mainframe suite.
   Rewrite the analysis and reporting programs around a "workload-oriented" data base on the mainframe, under MVS SAS, making it more easy to maintain and use. The attractive features of this were that we were already very familiar with SAS, and that there would be no additional costs. There were also some less appealing features: we had no mainframe colour graphics capability, nor colour printing, we couldn't integrate PC-based forecast data, and mainframe response times already deteriorated rapidly as the systems got busy.

4) Build a new approach, using mainframes and PCs.
   This would involve a bit of "rightsizing", generating all the MXG PDBs on the mainframe, then extracting workload-oriented PDBs which could be downloaded monthly to SAS on PC. From LAN-based SAS data sets we could then produce colour graphs to illustrate all our major metrics, and store these in SAS catalogs. PC printer technology would enable us to print colour graphs for IT management until such time as we could deliver the output at their own terminals. The benefits of this approach would be the availability of on-line and colour graphics, integration of forecast data, and ease of developing new graphs. The problems would be that we were unfamiliar with PCs, and would have the burden of downloading the data.

We eventually plumped for the fourth option. Why? Well, a thoroughly revised workload-oriented reporting system would make maintenance a lot easier; PCs were just emerging within GA as the forward direction for technology; the presentation clarity of the graphs would be significantly improved with the addition of colour; development/revision of graphs would be immensely easier if it could be done interactively through PC SAS; and eventually IT management would be able to access the outputs with ease. It was at this stage that we concluded that we would have to look seriously at SAS/AF with a view to building a "point and click" type application to direct non-Capacity Planners around the large array of graphs we wanted to produce. It would be untrue to say that we had a development strategy at this point, but a kind of a plan had emerged.

1) Develop the concept of workload-oriented reporting and PDBs
2) Develop PC-based graphs to illustrate our major metrics
3) Publish a monthly capacity report in colour to replace the maroon folder
4) Develop a PC-based SAS/AF application to guide users to the right graphs
5) Build-in some interrogative tools, to aid the extraction of information

Having decided to go for PC-based SAS for this (amongst other reasons), we were provided with IBM PCs and OS/2®. The SAS System which we received was at V6.08, though we have since moved up to 6.10.

Workloads or Loadsawork?

As we have mentioned, there was a lot of inconsistency in the definition of a "workload". One of the first tasks was to look hard at what we were running and how we categorized it. RMF reporting performance groups formed the basis of our workload definitions, and a fairly effective pattern had been established (cf. Foil 15).
The reporting, however, failed to make full use of this structure, as it was mainly based on the CICS regions. There was little account made of the database regions, and very little interest in TSO, batch, the terminal session manager, etc. Many CICS regions were single-application systems (usually tied to a database region), but several ran multiple applications (dozens) and some applications were spread across multiple CICS regions. The picture was, to say the least, confusing.

A pragmatic definition of a "workload" was built into a new kind of PDB. The various regions were defined as "workload parts" and the combination of CICS and database was the basic "workload". A flag identified the nature of each part: CICS, database type, TSO, Batch. Having a system-id to identify the mainframe on which a workload ran simplified matters if a workload had to be moved between systems. These variables could be used with WHERE clauses to combine applications very effectively (e.g., extract all CICS).

Fresh SAS programs were written to take the information we needed from our MXG PDBs and build it up over each month. We agreed a basic set of variables, such as peak hour CPU utilization, several transaction counts, central and expanded storage occupancy, session counts, etc. CPU figures were adjusted for capture ratios, and the data coming from an MVS guest under VM were likewise adjusted for VM. We merged the various kinds of data together as a single observation. This would put all the data in one place, rather than in a variety of different files. There would be a common pattern for every workload, making this very easy both to maintain and to learn. It would also make it very easy to add new measures (disk space, for example, was added late on). The pattern would be the same for monthly figures as for hourly figures (just add the date and hour). These were built on the mainframe, then downloaded on a monthly basis to LAN.

Workload-oriented data would not cover everything, and some system-wide measures and reports would also be needed (for instance, the UIC values, overall CPU, paging rates).

Workload-Oriented Reporting

Once we had the workload PDBs we could produce SAS programs to report on these. There would be a common data structure for each workload, so SAS macros presented themselves as a good way of developing the new reports. One macro was developed for each "resource" - e.g., peak hour CPU - taking a workload-id as a parameter. These contained SAS programs to extract and process data from the workload PDBs into SAS/Graph® colour charts and plots, and to store these into one SAS catalog per month. We called this system "ART" (Application Resource Trends). An unique naming convention was devised; e.g., the ASU application has a response-time graph called libref.monyy.ASURESP.GRSEG. The workloads were all defined; the resources were all defined; the graphs for any month were stored in a single catalog. All of this lent itself to the kind of menu-driven SAS/AF application we wanted to build to access the information more quickly.

The macros were all compiled and stored in SASUSER.SASMACR for the capacity planners who would be using them. They were executed simply by submitting a program statement like %ARTRESP(appl-id); where appl-id would be a short identifier for the workload, and would allow the program to correctly title and name graphs, as well as perform workload-specific processes. The programs themselves contained sufficient awareness to "decide" in which monthly catalog to store the graphs, and used old-style SAS macros for control of conditional processing for certain applications.

System-Wide Reporting

To complement the workload-oriented reporting we wrote some SAS programs for the PC which dealt with system-wide capacity and performance measures - total CPU trends, UIC values, ES-CS page movement, etc. These produced graphs which went straight into the same monthly catalog as the workload graphs. We called this "SRU" (System Resource Utilization).

Like the workload programs, these new programs were encapsulated within SAS macros, compiled and stored in SASUSER.SASMACR. This meant that we could generate a whole batch of graphs simply by a program statement like %SRUMCPU(system-id); with system-id being the only variable. It allowed for the handling of minor idiosyncrasies of some systems, and for conditional execution of sections of code, directly within the macros (whereas ART used old-style SAS macros for similar control).
Practicalities

Consider the point we have reached. We have an array of programs on the mainframe platform to take data from various MXG data sets and merge this to produce a workload-oriented PDB. We have a battery of macro-driven programs on a PC SAS platform which take workload-oriented data and turn it into pretty and meaningful graphs. So what's missing? The missing link is getting the stuff from mainframe to PC. Ideally we would have liked to use SAS/Connect®, but budget constraints told against us. In the event, we used a combination of PROC CPORTr on the mainframe, a file transfer mechanism and PROC CIMPORT on the LAN.

Downloads were done on a monthly basis, and the graphs then produced by running the relevant macros. The resulting colour reports were templated four-up (2 by 2) to reduce paper. The graphs were bound with a summary text to form the "Capacity Status Report". This was much smaller than the maroon folder and easier to read, but still suffered from one of its basic problems: there was only one copy being passed around the managers. The next step had to be designing a way of presenting all this information to IT management online, at their own desks, and in a managed way. It was time to take a look at the FRAME entry of SAS/AF.

SAS/AF - First Steps

We opted to take SAS/AF for the PC from day one. Bad luck in course schedules, holidays and unplanned events meant that we never managed to get anyone onto a SAS/AF development course. Eventually, with the aid of the SAS manuals below, we started "playing" with the FRAME entry. They proved very helpful.

- "Getting Started with the FRAME Entry : Developing Object-oriented Applications" (56570)
- "SAS/AF Software : FRAME Entry Usage & Reference" (56012)
- "SAS Screen Control Language" (Usage 56031, Reference 56030)

Our faltering steps were concentrated on the selection of pre-built graphs from catalogs - the principal task of our proposed SAS/AF application. With no prior knowledge we managed to put together a few list boxes, push buttons and SAS/Graph Output objects, and suddenly we were getting places: it was actually working! Soon we were adding all the features of the FRAME entry. We weren't used to sales people telling the truth. "This", we thought, "is easy." And to be honest, it was.

Capacity/MI

After a very little prototyping we rolled up our sleeves and got on with the real job of putting together a working system. Our objective was to build a menu-driven GUI application to guide the user around our SAS/Graph catalogs of pre-built and stored graphs. To this aim we started from a simple top menu (FRAME) with a number of icons, each of which took the user to a subsequent FRAME. In this way we were able to "stream" the system: we had the workload graphs, system-wide graphs, data on our branch network, and others. As a quick entry point for IT management we decided also to provide a "key points" FRAME. All the FRAMEs had a common "feel": a blue background, a common graphic text title, a black message line at the top, standard push buttons for "Help" and "Go Back", and easy-to-use objects such as icons and list boxes.

The "ART" (workload) element was simply based on selection lists; pick a month (for the catalog), then a workload, then a "characteristic" (e.g., response time, CPU utilization) and there's your graph within seconds. The data used to populate the lists was held in SAS data sets, for ease of maintenance. The facility for accessing branch network data was a bit more sophisticated, relying on having the actual data in place rather than pre-built graphs, allowing the user to "home-in" on particular features, build graphs or view the data (using SAS/FSP®). The "key points" service for managers gave them a number of "bullet points" which we thought they should know about, combining on one FRAME a graph, some figures and a bit of explanatory text.

Not everything was perfect about this application. We had developed it on fairly high-quality PCs, with good graphics resolution. When we tried to run it on other machines we hit our first real problems. We'd paid little attention to the attachments on the FRAMES; more fool us. We found that our screens did not fit well: list boxes got truncated, graphic colours changed, icons turned into blocks, etc. We went back to the manuals, and added arrows and springy things into the FRAMES until they would fit all our PCs. Eventually we did overcome our problems; the system was ready. We called it "Capacity/MI". It had taken only two months from opening the manuals to the completion of testing. Finally, we released it onto an unsuspecting world.
"More, more, more ..."

And the world liked it. Or, at least, our IT management did. It meant that, for the first time in GA, they could sit at their own desks and call up information about the capacity and performance of our computer systems in full colour on their own PCs. The only concern which we got back from our customers was that we'd built the system on OS/2, and a lot of them were using Windows™ (a strategic issue which remains unresolved). The main reception we got was one of "great system; but can it do ...?". We quickly received "wish-lists" of new features; for instance, service level reporting and disaster recovery planning facilities. A major beauty of the system is that, away from paper-based reporting, we can now add different measures at will.

**Capacity/MI 2.0**

Almost as soon as Capacity/MI 1.0 hit the managers' desks, we began work on designing V2.0, building on V1.0. At the time of 1.0 our customers had no idea of what to expect and no experience on which to base informed judgement; having delivered it to their desks, we have raised expectations, and allowed them to think about how such a powerful and versatile toolkit can be extended. It also brings in more sophisticated SCL programming (having finally understood SCL lists!). Screen clarity has been improved, and 2.0 makes some use of image objects and SAS/Graph Output objects to "prettify" the FRAMES - first impressions do count.

The new system is initially "streamed" into Management, Service and Technical. It has involved us in a lot more background work than we'd imagined, looking at a whole new generation of reporting which has to tell us what the real customers of our computer services are getting out there in the branches, in business terms as well as technical ones. A "service register" has been designed into the heart of it. We are aiming to develop a decision support system which is far more interactive than 1.0, and which incorporates more tools to allow our customers to formulate their own questions rather than the ones which we'd prefer they asked.

What next? Well, addressing the PC platform question (OS/2 vs. Windows) is high on our agenda. Though we're dependent on a corporate decision on this issue, we are optimistic. The SAS Institute have done their best to reassure us of the portability of SAS/AF applications, so if the GA decision goes in favour of Microsoft® we expect to make a quick and painless transition from OS/2. Similarly, if we have to support both PC platforms, having our Capacity/MI system on SAS should help minimize the maintenance overheads.

**In Conclusion**

GA now has information on computer capacity and performance online at the desks of our IT managers when they want it. We have an easily-maintained system, which can adapt quickly to changes. Capacity Planning is providing more and better information than ever before. And maybe we've saved a few trees too.

How has the SAS System helped us? In one word, integration. We have the MXG software; we have our SAS programs extracting workload-oriented data from the PDBs; we have a GUI system on SAS/AF which delivers the right information to the right people at the right time. We designed and wrote this system, so we can add whatever we want to it, at any time. And we did it without any major expenditure.

What has been the key to our success (apart from the SAS System, that is)? A good data structure, which we designed from a blank sheet of paper to underpin all our efforts.

---

*SAS, SAS/AF, SAS/CONNECT, SAS/FSP and SAS/GRAPH are registered trademarks of SAS Institute Inc., Cary, NC.*

*MXG is a registered trademark of Merrill Consultants.*

*CICS, IBM, MVS,OS/2, RMF, SMF and VM are registered trademarks of International Business Machines Corporation.*

*The Monitor for CICS is a registered trademark of Landmark Systems Corporation.*

*Microsoft is a registered trademark of Microsoft Corporation.*

*Windows is a trademark of Microsoft Corporation.*

---

Douglas Clarke and Mark Hayden
Capacity & Performance
Technical Services, SD
General Accident
Pitheavlis, Perth
Scotland, UK

Phone (UK) : +44 (0)1738 895861
Fax (UK) : +44 (0)1738 895495