ISSN 0961-9348

casg Computer Audit Specialist Group

JOURNAL

VOLUME 9 NUMBER 3 SPRING 1999

The British Computer Society

Technical Briefings for 1998/1999 Season

For more details of all Technical Briefings, and details of costs and registration, contact Jean Brown, on 01803 872775

Technical Briefing (Chargeable Attendance)

Tuesday 20 April 1999 Benchmarking IT, Systems development, Data centre

---

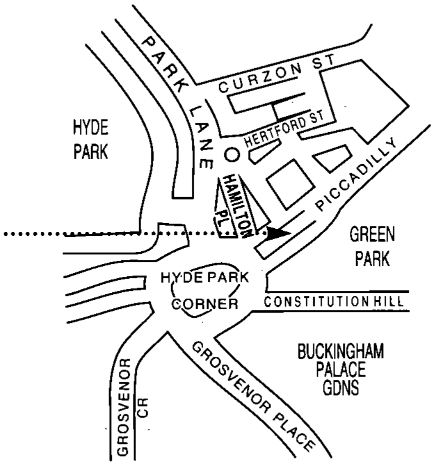
Late Afternoon Meeting (Free Attendance)

Tuesday 18 May 1999 Forensic Computing

Followed by the Annual General Meeting.

Venue for Technical Briefings

Royal Aeronautical Society,
4 Hamilton Place
London W1V 0BQ



Contents of the Journal

GUIDELINES FOR POTENTIAL AUTHORS

The *Journal* publishes various types of article.

Refereed articles are academic in nature and reflect the Group's links with the BCS, which is a learned institute governed by the rules of the Privy Council. Articles of this nature will be reviewed by our academic editor prior to publication and may undergo several iterations before publication. Lengthy dissertations may be serialised.

Technical articles on any IS audit, security, or control issue are welcome. Articles of this nature will be reviewed by the editor and will usually receive minimal suggestions for change prior to publication. News and comment articles, dealing with areas of topical interest, will generally be accepted as provided, with the proviso of being edited for brevity. Book and product reviews should be discussed with the appropriate member of the editorial panel prior to submission. All submissions should either be on double spaced, single-sided A4 paper, e-mail, or on PC format diskette in Microsoft Word, Ami-Pro, or ASCII format. Electronic submission is preferred.

Submissions should be accompanied by a short biography of the author(s) and a good quality monochrome photograph, or electronic image.

Submission Deadlines

Spring Edition	7th February
Summer Edition	7th May
Autumn Edition	7th August
Winter Edition	7th November

The ***Journal*** is the official publication of the Computer Audit Specialist Group of the British Computer Society. It is published quarterly and is free to members.

Letters to the editor are welcome as are any other contributions. Please contact the appropriate person on the editorial panel.

Editorial address:
47 Grangewood,
Potters Bar
Herts, EN6 1SL

Designed and set by Carliam Artwork,
Potters Bar, Herts
Printed in Great Britain by PostScript,
Berkhamsted, Herts.

EDITORIAL

Corporate governance is back in the news again as a result of the latest requirements of the London Stock Exchange. If you are a listed company your main board will soon be required to state that they have reviewed the effectiveness of the company's financial, operational, compliance and risk management controls. Now this goes far beyond the 'Cadbury' requirement of evaluating financial control and places risk management and operational controls firmly on the agenda. Information technology is both risky and often poorly controlled, so the onus is now on us computer auditors to make management take note of the new requirements and to provide them with a framework for measuring their control over all aspects of it. What if your company is not listed? Well, past experience has shown that where the commercial world goes, the rest is soon to follow, so it will just be a matter of time before local and central government, not-for-profit organisations and private limited companies will have to fall into line.

How best to approach the task? As companies will be required to report annually, we may have to move away from our current strategic planning approach of covering the major risks over a number of years, to one of assessing our risks and controls annually. What better use for control self assessment based around risk mitigation? What better excuse, if one were needed, for getting involved in system developments at the requirements stage? Remember, internal and external control is a requirement of any system and internal audit is, by default, a secondary user of all the company's systems. By helping to design quality systems we help with the corporate governance requirement too.

John Mitchell

The views expressed in the Journal are not necessarily shared by CASG. Articles are published without responsibility on the part of the publishers or authors for loss occasioned in any person acting, or refraining from acting as a result of any view expressed therein.

Chairman's Corner

John Bevan

We all develop our skills and knowledge in a number of ways: by work experience, attending training courses, private study, going to CASG meetings, and so on. I have been toying with ideas on how I could record and present my own development in way that would allow others to verify my claims of expertise. I had briefly settled on something similar to, but in some ways not as good as, the British Computer Society's Continuing Professional Development (CPD) scheme. For BCS members continuing personal development is obligatory, and CPD is one way of planning, evaluating, and recording it. If you do not already follow another professional society's scheme, then I recommend that you take a good look at the BCS CPD. Call BCS HQ (01793-417417) for details, or see http://www.bcs.org.uk/cpd/index.html.

I also recommend that you assess the qualifications in IS and telecommunications provided by the BCS Information Systems Examination Board (ISEB). They cover topics that are important to many computer auditors, and can be obtained by attending a short course (of one or two weeks say), and taking an exam at the end. Enclosed with this edition of the Journal are details of the Project Management and Information Security Management qualifications. For more on these and other topics call BCS HQ or see the BCS Web site: http://www.bcs.org.uk/iseb. Although I have no personal experience of associated courses, they seem particularly attractive because they cover relatively self-contained, limited scope, topics, and can be taken when needed. Because students' learning is measured by the final exam such a course will always score higher in CPD terms than a similar course without an exam!

Although the BCS does not provide broad brush computer audit qualifica-tions, such as those on offer from ISACA and IIA-UK, in some ways it is providing a relevant and distinct alternative. Should it be extended?

Welcome to our new Treasurer, Mike Demetriou. Bill Barton resigned after the AGM, so the management committee followed the constitutional path and elected Mike at the January 20th committee meeting. Many thanks to Bill for his past work as CASG Treasurer and committee member.

Meet the Treasurer

Mike Demetriou is an Information Systems Auditor currently working for NatWest Audit at Lombard North Central in Redhill.

Mike has worked for Lombard (who are a subsidiary of NatWest Bank) since 1979 in a variety of IT related roles including Operations Support and Contingency Planning. He has much experience in mainframe, and more recently, desktop computing. He moved into an Internal Audit role in 1993 but continued his association with IT looking at IT issues from a risk and control perspective rather than in a day to day line role (poacher turned gamekeeper some might say).

It was in this most recent role that Mike became involved with CASG, initially attending Technical Briefings before joining the committee in 1997. He recently agreed to have his name put forward for the role of CASG Treasurer and was elected to this post by the committee in January 1999.

Mike is also a member of the Institute of Internal Auditors (IIA) and is studying for their professional qualification MIIA, having already achieved the IT QiCA level last year.

He is married with two children and lives in Surrey within easy commuting distance from his office at Redhill. Interests include travel, squash, jogging and gardening.

CASG Adminstrator Wanted

The management committee are looking for someone to carry out this vital part time job. It includes:

- Membership: mailing renewal reminders, answering membership enquiries, banking fee cheques, and keeping the list of members up to date
- Meetings: compiling and sending out meeting publicity, fielding enquiries, taking bookings, making final arrangements with speakers caterers and venues, compiling and copying delegate packs

At present we have several hundred members, and arrange three full day and three evening meetings a year. The committee sets strategy, finds speakers, and provides guidance on how to do the administration. We need someone who is efficient, well organised, and reliable, with a permanent address, normal office PC skills, and telephone! It is a very important job, with scope to work cooperatively with the committee to find the best way of organising and promoting the specialist group's activities.

We are looking for someone to start in June 1999 or soon afterwards, because our present administrator has too many other commitments. We have some flexibility in exactly how the work is done.

If you know someone who may be interested, then encourage them to phone me to find out more, answer questions, and perhaps agree on how to take matters further. Alternatively they can drop me a note and I'll call them back at a convenient time.

John D Bevan 46 Queens Road Hertford SG13 8AZ 01992-503509 evenings and weekends

System Development Control

(To aid the achievement of quality systems)

John Mitchell

INTRODUCTION

I have subtitled this article "to aid the achievement of quality systems", because I believe that internal audit can do a great deal to achieve that goal and also because it implies a positive, added-value step for audit to take. The increasing importance of quality in the airline industry, as represented by IS9000, makes it imperative that we do our part in helping to achieve that objective.

THE CHARACTERISTICS OF REAL-TIME COMPUTING

Real-time systems tend to be complex, and often operate across national and cultural boundaries. They often have the characteristics of newspapers in that they are time-critical and sometimes subject to public scrutiny. Often, these computer systems provide the competitive edge; the difference between success and failure.

Using airline systems as an example. The reservation system books the customer onto the aeroplane; the scheduling and replenishment system ensures that the aeroplane is in the right place, at the right time and ready to go; the maintenance systems keep it in the air and helps to ensure compliance with safety legislation. The interaction between the customers, the travel agents, the airline staff and third-party suppliers is reflected in the airline's computer systems. This, coupled in many cases with a reliance on third party telecommunication agencies, requires the development of well controlled and resilient computer systems.

Are we auditors up to the job?

WHAT IS THE JOB?

Our job is to help management in the discharge of their responsibilities in controlling the organisation in a cost effective and economical manner. We usually achive this by conducting independent appraisals and analysis of operations. Ideally, we choose our "targets" with care; based on some form of risk assessment. Change is one of the most risky things that an organisation can do and the development/enhancement of a computer application usually means that a big change is envisaged. In view of the importance of computing to airline survival it is almost a pre-requisite that we are involved with developing systems. If we are not, then we are not doing our job.

DEBUNKING THE INDEPENDENCE ARGUMENT

Poor audit departments often try to cloak their ignorance of I.T. by saying that their independence will be compromised if they get involved with the system development process. Their argument is usually "how can we fairly audit the live application if we helped design it in the first place?". What a falacious argument! Internal audit can never be considered totally independent when one considers who pays its wages. Secondly, involvement does not mean we build the system. The patient involved in surgery does not wield the knife!

So let us stake this particular Dracular once and for all. We are the control experts, we are paid to help management, we cross departmental, functional and skill boundaries. Let us use our skills to help build well controlled, quality systems.

TWO AUDITS IN ONE

The SDLC process has two main components; the project process and the product (computer application) produced by that process. We can audit both aspects simultaneously.

For the project, management would like to see it completed on time and to budget. This implies good planning and monitoring.

For the product, management would like it to meet specification and be well controlled. In short they wish to see a quality product. But what exactly is quality?

WHAT IS QUALITY?

The dictionary defines quality as "degree of excellence". and I suspect that no one would disagree that quality systems are what everyone wants. The problem is achieving it.

Quality has also been defined as either fitness for purpose, or meeting the specified requirements. Either way, there can be no such thing as good, or poor, quality. If something does not meet its purpose, then it is useless, not just poor. Likewise, something that exceeds the requirements is just a waste of money.

However, there can be little doubt that different people view quality differently - for example:

- a user wants a system to do what he wants and only what he wants, consistently and without problems;
- operations staff require systems that do not fall over and are easy to operate;
- designers wish to develop elegant systems that are easy to maintain;
- senior management want the development to be ready early and under budget and for the system to cost effectively support the business objectives.

Each of these views is valid, but it can be difficult to achieve them all simultaneously.

Computerised applications present special problems to us auditors. A major consideration is that the cost of inserting a control once the system has gone live, is about one hundred times the cost of doing so at the design stage. Because of this we need to identify important systems before they are too far down the design road, so that the necessary control procedures can be considered as part of the quality design process.

SO WHAT NEEDS TO BE CONSIDERED?

Today, in general, hardware is reliable and of good quality and telecommunications facilities basically work in the developed world. System software (operating systems, DBMS and the like) is of functional quality, but remains subject to fixes (upgrades) with alarming regularity. To the application system developer these are bought-in components and therefore are taken as "given" in the development process.

Because of this I will concentrate on the development process relating to application systems. I do this for three reasons:

- your own development staff have to accept the "given" supporting software;
- the application is what the users deal with in their day-to-day affairs;
- these are what most people are concerned about.

THE DEVELOPMENT PROCESS

Over the years there has evolved a general system development and maintenance methodology called the System Development Life Cycle (SDLC). This provides an orderly framework within which systems development work can be performed. This framework was created to assist developers to:

- ensure the developed system will meet requirements;
- deliver the system within the agreed budget;
- deliver the system on time;
- create a maintainable system.

It also provides a useful control framework for audit purposes.

The System Development Life Cycle

The conventional SDLC framework is as follows:

- Definition of requirements;
- Translation of requirements into technical specifications;
- Development, or acquisition, of equipment and/or software to meet the technical specifications;
- Development of procedures for IT staff and users to use the system;
- Testing;
- Training of staff to use the new system;
- Conversion of previous data to the new system;
- Implementation;
- Ongoing changes to the system to meet changed circumstances;
- Decommissioning of the system.

Not all steps are necessarily performed for each system. The formality of the performance of any, or all, of the steps will vary from very formal (i.e. for a system that is to be subjected to evaluation under the ITSEC/ITSEM programme, or ISO9000) to totally informal (i.e. for a spreadsheet to prepare expenses claims).

Various techniques and tools exist to assist developers during development (for example SSADM, 4GLs, prototyping, CASE). The use of these aids may vary the nature of the development process, but they do not substitute for any of the steps that must be taken.

THE FIVE LEVELS OF SDLC OF MATURITY

Research indicates that there are five levels of maturity to the SDLC in organisations.

A company at the first level will be working in an ad-hoc way and although it may have formal standards and procedures there will be no management mechanisms in place to ensure that they are used.

The second level, the repeatable process level, is distinguished by being able to make correct estimates and detailed plans and being able to keep to them. It does not necessarily imply that a quality product will appear as a result.

The third level, the defined process level, is still only qualitative. Well organised projects are tracked against plans and they work to a set of well-defined procedures and standards, but staff are unable to make major quantitative statements about the quality of the software, such as the amount of trouble it will give during maintenance.

Staff working for a company at the fourth level, are able to make such statements. Typically, a manager working on a project is able to accurately answer questions about the effectiveness of program walkthoughs and design reviews in terms of saving money and meeting the specified (quality) requirements.

The final level, the optimising process level, is one which supports the automatic gathering of process and product metrics. Staff working on projects at this level are able to use this data to modify system design and testing in order to ameliorate problems and improve efficiency.

WHAT SYSTEMS SHOULD WE LOOK AT?

I suggest the use of categories to determine our level of involvement in a development. The categories reflect our assessment of the likely impact of the system on the business once it is implemented, rather than the cost of its development.

Category A developments are defined as those systems which are of such importance to the business that Audit attention on a regular basis is considered to be essential, if it is to effectively discharge its responsibilities to management. This may involve Audit attendance at Steering Group and other meetings, but will certainly require Audit to be on the circulation list of all correspondence relating to the development.

Category B systems are those where Audit participation at selected points (requirements specification, detailed design, testing, etc.) is believed to be necessary.

Category C developments only require a watching brief at the regular meetings between Audit and the IT Department.

Category D systems are usually ignored for Audit review purposes.

Every new system, or important enhancement, should be put into one of the above categories as soon as we are aware of its likely development. The category may well change as we gather more information about the system, but the important issue is that we are informed as to the likelihood of a development at an early stage, so that a decision can be taken on the likely resource implications.

HOW SHOULD WE BE INVOLVED ?

The degree of our involvement will depend on which category the development has been assigned to.

Category A developments will usually require our involvement on a continuous basis and a particular auditor will be allocated to the system. This does not mean that s(he) will be part of the Project Team, but s(he) will expect to receive all correspondence, documents, plans and reports relating to the development, as they are produced and also to attend meetings of the Project Steering Group and other relevant bodies.

We should not expect to "sign off" stages of the development, as that is the responsibility of the user, but we should check that the appropriate output from each stage is approved by the responsible person before the next stage is commenced.

We may well wish to do some testing of our own, but as a minimum we should review the system and user testing at the appropriate time.

Category B developments only require us to receive the outputs from each stage of the development. Typically these will be the:

- Development Plan;
- Requirements Specification;
- Detailed Logical Design;
- Detailed Physical Design;
- Operations Manual;
- User Manual.

We should be circulated with both the draft and final issues of the above. We should always respond, even if we have no constructive comments to make. The reason for receiving the development plan is to allow us to plan our own work.

Category C developments will form agenda items for the regular meeting between us and the I.T. Department. We may well request specific documents as a result of these meetings.

Category D developments will normally not receive any attention unless a specific request is made and then only after examining the implications to the coverage of those systems in a higher category.

HOW TO ACHIEVE A QUALITY PRODUCT

I have assumed that the organisation already has a series of procedures governing IT and systems development, therefore the ideas set out below cover those items which in the past have been less well performed despite all the good intentions by those concerned.

Defining the process

- Select a process, tools and techniques appropriate to the system being developed.
- Whatever process is selected for a particular development it should be sufficiently defined, clearly understood by everyone involved and capable of management (and possibly internal audit) review.
- Select the tools and techniques that the people involved understand; if new tools and techniques are being used then try them out on a small project first, ***so that staff learn how to get the most out of them.***
- Do not be fooled that a super process inevitably produces a quality product. A super process helps but people are fallible, including those involved in the development.
- On completion of the development, confirm that the documentation written during the development really reflects the implemented system.

Requirements

- All systems grow during development as the full ramifications emerge. Always try to contain systems. If this is impossible, then create a dynamic system where it is designed to be changed very frequently.
- Avoid complexity as far as possible. If the nature of the system requirements are complex in themselves:
  - Ask the question "Is the complexity really necessary for the business?" If not save the development cost and simplify the way of doing the business.
  - Make sure the complex parts are specified by users in full detail - for example by getting the users to make a spreadsheet of the formulae and all options.
- Be precise - do not assume anything. Cover all aspects of the system; not only the required business functionality, but also confidentiality, resilience and any imposed limitations on possible solutions.
- Write simply, so that those approving the requirements document really do understand what they are approving. Language is a vague form of communication and therefore the possibility of misunderstandings is high.
- The future is a closed book. It is not too difficult to define clearly understood requirements; it is impossible to define what you do not know. All systems therefore need to be designed to provide flexibility and ease of change because the requirements will never be complete except at a very low level.

Translation of requirements into technical specifications

- Avoid this step if possible. It adds a layer of interpretation which usually introduces inconsistencies. Use prototyping, formal methods, etc., instead.
- Don't design the technical specifications to fit specific equipment. The equipment will change but the application represents a much larger investment for the organisation.
- Isolate all complex operations into discrete parts of the system. Possibly it would be advantageous to design the system so that the complex parts are processed on local machines and the main system only processes simplified data (for example an airline maintenance programme might be processed on a discrete machine which provides the main system with only that data required for the airline as a whole such as cost information).

Development

- It is always true that a better piece of equipment or operating software will exist tomorrow. The selection process can only achieve the best known about today. With hindsight someone is quite capable of identifying a better bought in component tomorrow.
- All bought in components have limitations. It is important to identify them, determine if the limitations matter to the system under development and if so to amend the design (or the proposed clerical procedures) to compensate for the limitations.
- Create standard procedures for coding functions, using languages, etc., and enforce compliance with them. This includes procedures for use by users when developing their own spreadsheets, reports using report writers, etc.
- Try to ensure commonality of presentation of screens, etc., to users.
- Mistakes do happen, corruption does occur, therefore the system should be built to detect any errors that may occur before they cause damage and have a mechanism for putting it right.

Clerical procedures for IT and user staff

- Make them simple. Include all clerical procedures including filing, detection of errors after processing, etc.
- Do write them in parallel with the development, because it is often possible to simplify both the development and the clerical procedures by considering the best method of using these together and not separately, as happens if the procedures are written after development.
- Check that it is possible for appropriate staff/management to be able to confirm that no material error exists in the results of processing.

Testing

- Do this thoroughly. Involve the users actively.
- The final test should encompass tests of the programmes, equipment and clerical procedures together.

Training

- Create training appropriate to the level of staff being trained; directors need a different sort of training from junior staff, but they still nccd training!
- The period immediately after implementation is when a system is most likely to go wrong. It is therefore helpful to combine the training with the testing so that staff can make mistakes on the test system and perhaps, more importantly, see the results of mistakes so that they learn to recognise errors.
- Staff turnover is a fact. Create training courses to teach the new staff to use the system properly and efficiently and ensure that they get the training before starting to use the system.

Conversion

- If additional data has to be assembled, allow plenty of time for this to be done properly.
- Provide facilities to prove the conversion was 100% correct when performed.

Implementation

- Create an orderly implementation and cut-over timetable.
- Avoid implementing a partly complete system; do not for example leave the year-end routines to be done later.

Ongoing maintenance

- Do amend the documentation.
- Do re-test the system from time to time.
- Notify all concerned when there is a change (even to the system software) so that they are on their guard for problems and errors arising from the change.
- Change all affected parts together; often infrequently run programs fall over because changes have not been made to them.

REPORTING IN A TIMELY MANNER

It is no longer acceptable for us to go in after the battle to bayonet the wounded. We must raise an issue as soon as it becomes apparent and in such a way that relevant management are forced to respond. We need to build ourselves into the SDLC methodology so that our role is acknowledged.

Where we consider that an important issue needs to be resolved we should raise an "Audit Issue" document (see Fig 1). These documents require a clear answer to the question, within a set timescale and help to ensure that control aspects are not overlooked.

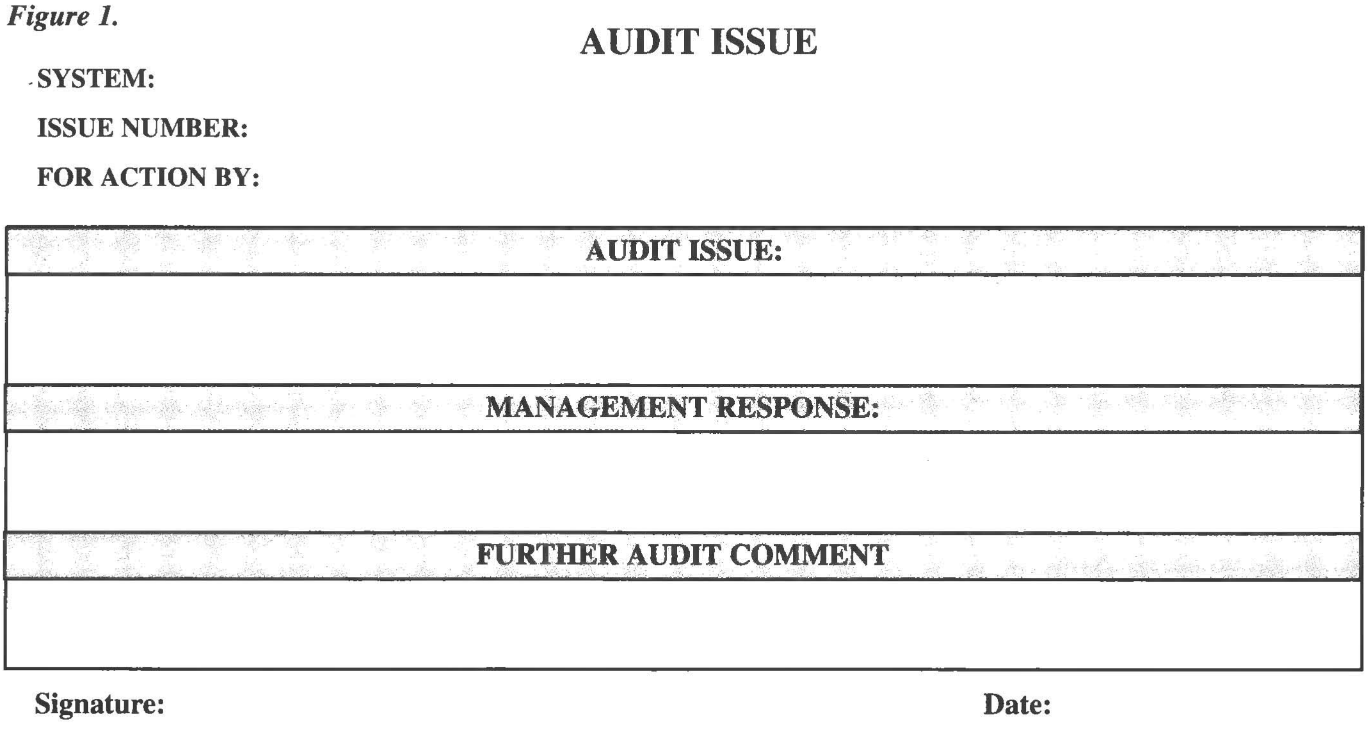
These forms should be used a lot. It shows our constructive involvement, keeps everyone on their toes and enables anyone unfamiliar with the development to quickly see where our main concerns are. It is simple in concept, elegant in operation and it achieves results!

CONCLUSIONS

Quality is a variable attribute. It is in the eye of the beholder and cannot be empirically measured. Conversely, rubbish is obvious to all. A quality process does not guarantee a quality product. Often, when the emphasis is on the process, too much attention is paid to form and not enough to substance. An orderly well run process does however, aid the production of a quality product.

The edge of quality comes from ensuring that everything is done well, nothing is skimped. A positive attitude to achieving quality is necessary by all concerned in developments. The challenge in this changing world to the developers of systems is to master the new techniques and apply them cost effectively to support profitable business ventures. The auditors involved in reviewing developments must help the developers meet this challenge. Finally, so far as achieving quality on a regular basis is concerned, good enough is not enough. It is our job to make this very clear to management by doing a quality audit of developing systems!

Figure 1.



AUDIT ISSUE

SYSTEM:

ISSUE NUMBER:

FOR ACTION BY:

AUDIT ISSUE:

MANAGEMENT RESPONSE:

FURTHER AUDIT COMMENT

Signature: **Date:**

Computer Language Breakthrough

Bell Laboratories has formally announced what it believes is the ultimate computer science language. Described by Iusi Nogoto, the foremost Japanese fourth generation language expert, as "the only truly elegant computer language ever devised," NULL, as it is known, was developed by the same department that originally invented the wrong number, the busy signal, and the phrase, "The number you have reached is not in service."

NULL is the culmination of five years of work by a team of language designers and computer science mathematicians. The final breakthrough occurred when operating system expert Hugh Nicks suggested that if removing GOTOs was good then why not scrap IF statements as well, since they usually required typing too many characters anyway. This brilliant concept was extended through a series of complex mathematical theorems that form the basis of the NULL language. Put in layman's terms by Sally Kahn-Vallee, electrical engineer and PROM reader, "Like we first we tossed out the bath water, then the baby, and like finally the whole tub." The elegance and conciseness of NULL can thus be proven to be a direct consequence of the fact that the language as defined contains no statements at all. While at first glance this may seem a drawback, in fact, it is a major improvement over any other language. A few of the numerous reasons are:

Point 1: Highly structured constructs.

Point 2: Advanced data hiding techniques.

Point 3: A NULL compiler can be written first in NULL without ever needing to be written in a lower level language.

Point 4: Since there are no statements to compile, in fact, no compiler need ever be written in the first place, saving time and money.

Point 5: Since there will be no compilers, no new releases will ever be issued hence maintenance is reduced.

Point 6: NULL programs are highly portable and totally machine independent.

Point 7: NULL programs compile and execute rapidly. An important point to note is that with the addition of a small amount of language dependent code, e.g. PROC/END etc., all NULL programs can be compiled by any other language compiler.

Point 8: Since there will never be new releases of NULL, all programs are upwardly and downwardly compatible.

Point 9: NULL can be parsed top-down, bottom-up, left-right, right-left, inside-out, and over-easy.

Point 10: NULL programs are both self-documenting for clarity and self-concealing for security.

Point 11: NULL programmers are easy to find and once found can be fired since they are not needed.

Point 12: If desired, specialized NULL hardware could be designed, implementing the code in firmware. Of course, such hardware may require years of development. One suggestion from Bell's VLSI experts Nora and Andy Gates was to take an existing available chip and remove all the instructions except NOOP. While this should work in theory, they acknowledged that it is probably not the most efficient implementation.

These are just a few of the many ways NULL is superior to all current computer languages. You can, no doubt, think of more. For further reading consult any of the numerous books and articles by Donald Knuth, David Parnas, and of course, the basis of all modern computer language theory, "The Emperor's New Clothes."

By John R. Andrews, University of Illinois at Chicago.

Unfortunately this work has been overtaken and made redundant by the invention of the VOID (Very Open Interface Device) computer.

This computer has the the advantage of being a one piece open architectury design. Very similar to a lidless shoe box in appearance it allows instant access to the content, which is nothing, thereby saving all access time as you only have to look once.

We have had Write Once Read Many (WORM), now we have Single Look Inform Many (SLIM) concept where one person need only make one access to the device, tell everyone what the saw and save many person hours of access time. It requires no internals and therefore avoids the need for any programmes, programmers, systems designers ... in fact no IT staff at all as all employees have a ready supply of these units from the local branches of Dolcis and the like. The ulltimate lowcost personal device.

Malcolm Dickson - Thos R Miller

(Just shows how the speed of development and change is racking up - Ed).

Web Abuse

California security guard Gary Steven Dellapenta, 50, is the first to be charged under the state's new "cyber-stalking" law.

Police say that when a North Hollywood woman refused to date him, he posted advertisements on the Internet in her name detailing her supposed rape fantasies. He included her physical description, address, telephone number, and details on how to bypass her security system. She only found out about the ads after six men showed up at her apartment - she does not have access to the Internet. Dellapenta is being held on $300,000 bail, and faces charges of stalking, using a computer to commit fraud, deceive or extort, and solicitation to commit sexual assault.

casg
Computer Audit Specialist Group

British Computer Society

Management Committee

CHAIRMAN	John Bevan	Audit & Computer Security Services	01992 582439 john.bevan@virgin.net
SECRETARY	Raghu Iyer	KPMG	0171 311 6023 raghu.iyer@kpmg.co.uk
TREASURER	Mike Demetriou	Lombard North Central plc	01737 776127 mdemetriou@lombard.co.uk
MEMBERSHIP SECRETARY	Jean Brown		01803 872775 100125.66@compuserve.com
JOURNAL EDITOR	John Mitchell	LHS Business Control	01707 851454 lhs001@aol.com
SECURITY COMMITTEE LIAISON	John Bevan	Audit & Computer Security Services	01992 582439 john.bevan@virgin.net
TECHNICAL BOARD LIAISON	Allan Brown	Consultant	01803 872775 100125.66@compuserve.com
TECHNICAL BRIEFINGS	Jenny Broadbent	Cambridgeshire County Council	01223 317256 jenny.broadbent@finance.cambscnty.gov.uk
	David Cox	Lombard North Central plc	01737 776286 dcox@lombard.co.uk
	Paul Plane	National Westminster Bank plc	0171 726 1000
	Alison Webb	Consultant	01223 461316 amwebbcam@aol.com

Membership Enquiries to:

Jean Brown
Whiddon Lodge
Abbotskerswell
Newton Abbot
Devon
TQ12 5LG

casg
Computer Audit Specialist Group

British
Computer
Society

PLEASE RETURN TO
Jean Brown
CASG Membership Secretary
Whiddon Lodge
Abbotskerswell
Newton Abbot
Devon
TQ12 5LG

Membership Application

(Membership runs from July to the following June each year)

I wish to APPLY FOR membership of the Group in the following category and enclose the appropriate subscription.

CORPORATE MEMBERSHIP (Up to 5 members)* £75

* Corporate members may nominate up to 4 additional recipients for direct mailing of the Journal *(see over)*

INDIVIDUAL MEMBERSHIP *(NOT a member of the BCS)* £25

INDIVIDUAL MEMBERSHIP *(A members of the BCS)* £15

BCS membership number: ______________________

STUDENT MEMBERSHIP (Full-time only and must be supported by a letter from the educational establishment).

Educational Establishment: ______________________________ £10

Please circle the appropriate subscription amount and complete the details below.

INDIVIDUAL NAME:
(Title/Initials/Surname)

POSITION:

ORGANISATION:

ADDRESS:

POST CODE:

TELEPHONE:
(STD Code/Number/Extension)

PROFESSIONAL CATEGORY: (Please circle)

1 = Internal Audit	4 = Academic
2 = External Audit	5 = Full-Time Student
3 = Data Processor	6 = Other (please specify)

SIGNATURE: DATE:

PLEASE MAKE CHEQUES PAYABLE TO "BCS CASG"
AND RETURN WITH THIS FORM TO THE ADDRESS SHOWN ABOVE

ADDITIONAL CORPORATE MEMBERS

INDIVIDUAL NAME:
(Title/Initials/Surname)

POSITION:

ORGANISATION:

ADDRESS:

POST CODE:

TELEPHONE: (STD Code/Number/Extension)

PROFESSIONAL CATEGORY:
1 = Internal Audit   4 = Academic
2 = External Audit   5 = Full-Time Student
3 = Data Processor   6 = Other (please specify)

INDIVIDUAL NAME:
(Title/Initials/Surname)

POSITION:

ORGANISATION:

ADDRESS:

POST CODE:

TELEPHONE: (STD Code/Number/Extension)

PROFESSIONAL CATEGORY:
1 = Internal Audit   4 = Academic
2 = External Audit   5 = Full-Time Student
3 = Data Processor   6 = Other (please specify)

INDIVIDUAL NAME:
(Title/Initials/Surname)

POSITION:

ORGANISATION:

ADDRESS:

POST CODE:

TELEPHONE: (STD Code/Number/Extension)

PROFESSIONAL CATEGORY:
1 = Internal Audit   4 = Academic
2 = External Audit   5 = Full-Time Student
3 = Data Processor   6 = Other (please specify)

INDIVIDUAL NAME:
(Title/Initials/Surname)

POSITION:

ORGANISATION:

ADDRESS:

POST CODE:

TELEPHONE: (STD Code/Number/Extension)

PROFESSIONAL CATEGORY:
1 = Internal Audit   4 = Academic
2 = External Audit   5 = Full-Time Student
3 = Data Processor   6 = Other (please specify)