How To Keep Your Auditor Happy

Robert A. Karlin
Karlins' Korner
7628 Van Noord Ave.
N. Hollywood Ca.
91605

'Twas the night before audit
and all through the shop,
not a creature was stirring,
not the tiniest flip flop.

'The listings were placed
'neath the window with care,
in hopes that the Auditor,
wouldn't look there.

'And I with my coffee cup
clutched in my fist,
searching around for
what could have been missed.

'And as I await there
 sweeping the floor,
 I suddenly hear
 a loud knock on the door.

'I rush to the door,
 and what there attends,
 but a three piece blue suit,
 and eight tiny red pens.

'"On Debits, On Credits,
 "On Dashes, and X's,
 "Is the Hold Account right,
 "Have we paid enough Taxes?."

'He moved through the room
 searching for listings,
 to find for himself
 what controls we were missing.

'And when he was through
 and his notations abed..,
 he turned to my boss,
 with a shake of his head.

'"All of these programs
 "must be rewritten.

"Documentation is bad,
"and the Data's not hidden.

'And as he strode out,
you could hear, loud and clear,
"If you think this was bad,
"just wait 'til next year!"'

with apologies to Clement C. Moore

IN THE BEGINNING

In most cases, it is difficult to involve your auditor in a project from its inception. Usually, he will wait until most of the design is complete before descending from his office upon the project team. A few fairly simple design considerations will go a long way in providing him with comfort and peace of mind, and this can be very important in keeping the project on course and under budget.

COST JUSTIFICATION.

Our first, and primary concern in providing auditability for our systems is cost justification. If we are spending twenty thousand dollars to design and implement a system, we should not spend two hundred thousand to insure its auditability, if the system does not require it. If you are doing bulk mailings, It may not matter if you loose ten percent of your master file, if you can still select enough different names to fill your mailing requirements. On the other hand, if you run a bank, a listing for the IRS of all customers that have earned more than six hundred dollars had better balance to the penny. During the design of the system, the actual worth of the end product (worth, not cost - a three hundred dollar program may be worth millions to the corporation) must be weighed against the cost of any auditability design enhancements.

BACKUP AND RECOVERY

One of the more important areas that an auditor will want to examine is the Backup and Recovery scheme for your project. He will not be concerned with the daily and weekly backups for your overall system (though the scheduling of these jobs in relationship to you system is important), but specifically with the application dependent backups that should be designed into your system. If you nightly processing takes almost your full offshift time, ten minute backups at convenient times during the night could save hours of recovery time. Sometimes, copying one key file to tape could be the key to a timely recovery. If your daytime online operation is critical to the corporation, a lunchtime backup could be very cost effective. Closing and copying your transaction log file at reasonable intervals can also save time and money in the long run. If you chose this route, be certain that you are predictable. If you close down at noon every day for a half hour, don't close down at eleven fifty five one day and twelve fifteen the next. 'When the operator gets hungry' is not a particularly effective lunchtime backup scheme. Keep a reasonable number of generations of your backup, and keep some of them offsite. This is an important part of the original system design, and including the backup procedures will show thoroughness that auditors like. Note that copies of your software should also be kept offsite.

TESTING AND TURNOVER

A second area that an auditor will be interested in is your

testing and turnover procedures. Procedures for both problem resolution and new software releases should be considered as part of the initial system design. Backout procedures should be part of the turnover of any changes to a system. Application system changes should be scheduled at non peak times, and the users should be forewarned that the change is being implemented, so that they may prepare for the possibility of catastrophic failure. All changes should be tested in an environment as close as possible to the production environment in which they will run, and the test results should be part of the change documentation. All changes must be reflected in the system documentation, and no system should be accepted that does not have sufficient documentation. The auditor will probably require the user to sign off to any changes to the application system, whether or not they will be visible to him.

DOCUMENTATION

System documentation is probably the first thing any auditor will ask for, and woe to the project manager that does not have it. The documentation should be written during the project development as an integral part of the system design package, and should be kept current as programming continues. The original program specs should be complete enough to write the program without the need for additional information, though I have yet to see a system where this could be done. At the least, the program specs should be annotated by the programmer when further information becomes available. File contents and usage should be kept in some form of data dictionary,

and the dictionary should be kept current as the project continues. File layouts should be in copybooks, generated, if possible from the data dictionary, and all program should use the same copybooks. Keeping the documentation current is as important as the completeness of its content. System documentation should consist of, at the very least, input and output descriptions, the data transformation and formulae used in the programs, restart and rerun procedures, primary users, other users, system dependencies, balancing instructions, and program and job control listings. Copies of the documentation (either in machine readable form or on microfiche), should be kept offsite. You should also develop some form of change control documentation consisting of a description of the change, the reason for the change, copies of any changed system documentation, and backout procedures in case of a change failure.

SECURITY

Of all areas of system control, security is probably the most misunderstood. Most people consider security to consist of preventing unauthorized access to production data. This, in truth, is part of security, but is hardly the entire subject. Security should be considered as the protection of the operating environment from compromise or damage WITHOUT SERIOUSLY IMPACTING USERS AND PREVENTING THEM FROM DOING THEIR JOB. This includes providing the operations personnel responsible for the maintenance and monitoring of the system with the tools necessary to their work. Any good programmer, with the time to work at it, can access or change data surreptitiously in a production environment. It is better to spend

time and money in ensuring that all accesses to production data are properly logged, and that unauthorized access can be spotted easily, than to try to prevent all access to the data. Tools such as QUERY should be controlled, as opposed to eliminated. If security is made so tight that programmers and users cannot do their jobs properly, the security system will be circumvented at the first opportunity. The balance user friendly and tight security is a fine one, and should be based on the actual worth of the data and system involved, not on a 'global' security mandate.

DATA INTEGRITY

The last area that we sill discuss from the auditor's checklist is the controls needed to ensure the veracity of the production data. At each step in processing, checks should be established to ensure that the data at this step is proper. All programs should, at a minimum, provide a record count of input and output records. Programs that do serial reads should also provide some form of balancing total that can be used to trace data as it moves through the system. A transaction log, containing dates, times, users and change data, should be updated for every change to the data base. Control totals from this log should be checked against the data base at appropriate intervals. Programs to insure the integrity of the database should be part of the original system design, along with maintenance programs that update the transaction log while updating the database. For image databases, DICTDBA, HOWMESSY, DBSTAT (or any of the other programs that do forward and backward reads of all

the chains on a database), can be run on a scheduled basis, though a program to do the same function can be written and tailored to the application and would provide a better test of the database. In general, audit trails and balancing controls should be kept as simple as possible, and should be easily located for crosschecking against each other. If they are not, they will be ignored.

AS THE WORLD TURNS

One of the major problems that plague ongoing projects is that the personnel that finish a project may not be the ones that started it. The job of the auditor is made that much harder by constantly changing procedures and styles of coding and documentation. An auditor that does not understand your system is one that will criticize it heavily. A few important dos and don'ts may save hours of headaches writing responses to your auditors reports.

CONSISTENCY

If you feel the need to change standards in mid stream (and there may be very good reasons to do so), take the time to retrofit previous coding and documentation to match the current standard. But it works, I hear you say. If it works that well, then maybe your standards don't really need changing, at least as far as this project is concerned. On the other hand, if there is such a pressing need for revision, then there is a need for the recoding effort. If you don't schedule it as part of the change, it will never ever get done. It is far worse to attempt the maintenance of a system that does not follow a consistent standard than to attempt the maintenance of a system that follows a bad one.

INFORMATION AS IT HAPPENS

Copy your auditor on every design change memo, the minutes of every design meeting, and copies of all user memos. It may seem that by deluging him with what you might think to be trivia, you will be actually hurting your cause, but this is not so. If your auditor can see the development of your system as it happens, you will not have to explain as many of the design decisions you make to him down the road, when those reasons are clouded by time and subsequent problems. Also, by keeping track of the evolution of your system, your auditor will develop a sense of continuity within which your decisions will be more explicable.

AND THE MOMENT OF TRUTH

In the final analysis, whether or not your auditor is happy with your project will depend as much on how the project is accepted by its users as on anyone else's opinion. Your auditor will usually look in those places that have been pointed out to first. If your users are will informed, and happy with the results of the project, the auditor will begin his audit on a positive note. If your users are plagued by an unreliable system, and are not kept informed of changes, both planned and unplanned, the auditor will have half of his report written long before he enters the data processing department. It is difficult to keep all users happy all of the time, but by getting user signoffs on all changes, and by making the user feel as if he had a hand in designing the system that will ultimately determine how well he does his job, you can keep him happy most of the time. And this will certainly increase your chances of receiving a successful audit of your system.

AND SO ...

In summary, most points that your auditor sill bring to your attention are common sense approaches to the design of your system. Your auditor is most concerned with the protection of the production environment, and the ability to prove the results of the system under consideration. He will want to be able to pick up any report, pick a figure, and trace it back to its original source document. To do this, he will need good system documentation, a good audit trail of transaction history, the confidence that the report he holds coincides with the database at a particular time, and the knowledge that the programs that led to the production of the report have been fully tested and integrated into the system as a whole. In general, these points are the points that you should be concerned with as well. In some respects, your auditor is your conscience, keeping your from cutting corners that you know you shouldn't. Remember this, and you will survive even the most stringent audit.