Auditing with IMAGE Transaction Logging

Robert M. Green
Robelle Consulting Ltd.
Aldergrove, B.C., Canada

SUMMARY

The transaction logging of IMAGE is not just for recovery of lost transactions; the transaction log-files contain a vast array of information that is useful for auditing purposes. Reports generated from these files can answer basic audit inquiries (WHO, WHEN, WHERE, WHAT and HOW), can provide statistics that are useful for performance tuning (which dataset has the most puts and deletes?), and can aid in program debugging (what does this program actually change in the M-CUST dataset?)

CONTENTS

INTRODUCTION

Since MPE release "1918," the IMAGE/3000 database system has had the ability to "log" database changes to a disc or tape file. Although the format of these log-files is somewhat obscure (and not documented accurately or completely), they can provide a great deal of information that is useful for auditing. DBAUDIT (a proprietary software product of Robelle Consulting Ltd.) will analyze transaction logging files and print transaction audit reports from them.

Here are two sample IMAGE transactions (a DBOPEN and a DBPUT), as printed by DBAUDIT, which show the auditing information that is available from transaction logging:

```
OPEN    17 AUG81 11:38 N:3          L:1
U:BOB.GREEN,PUB               P:QDBM.PUB.GREEN
B:TEST.PUB.GREEN            Mode:1   Security:64
Logon device:28   as a session.  Userid:None.
Last dbstore:17 AUG81 11:15

PUT     17 AUG81 11:38 N:4          L:1
U:BOB.GREEN,PUB               P:QDBM.PUB.GREEN
B:TEST.PUB.GREEN            DE:DCOM             R:10
Data-Added:
ORD-NUM          = 11111111
COM-NUM          =       5
COM-DESC         = 55555555555555555555555555555555555555555555
```

Each transaction has a date and time stamp (17 AUG81 11:38), a unique transaction number assigned by MPE (N:3), a unique logging access number for each user who does a DBOPEN (L:1), a logon account, group and user name (U:BOB.GREEN,PUB), a program name (P:QDBM.PUB.GREEN), a base name (B:TEST.PUB.GREEN), a logon device number and batch/session indicator, an optional userid (an extra identifier that can be passed to DBOPEN as part of the password, and acts to distinguish between different users who happen to be logged on with the same MPE user name), the dataset type and name (DE:DCOM is a detail dataset), the data entry's physical record number (R:10), the key-field value (for master dataset entries), the fields that were added, deleted or changed (with field names), plus before and after data values that have been converted from binary to ASCII where necessary (ORD-NUM = 11111111).

Logging Answers Basic Questions

As you can see, logging provides answers to many questions:

```
WHO (logon user name and user id)
WHEN (date and time)
WHERE (database and dataset)
WHAT (data fields changed)
HOW (terminal number and program name)
```

The only question that cannot be answered is WHY?

Two Types of Log-files

There are two basic types of logging files that can contain IMAGE transactions: raw log-files (on disc or tape) which are filled by IMAGE as transactions occur, and user recovery files generated by DBRECOV during transaction recovery.

The original log files have fixed-length records, with large transactions split over several records.

The user recovery files hold the transactions as variable-length records (one record per transaction). User recovery files are usually generated for transactions that DBRECOV could not recovery (and which you must recover by hand). User recovery files contain one extra field which is not found in regular log-files: a recovery flag that indicates whether each transaction was successfully recovered or not ('OK' or 'NO').

SELECTING AND FORMATTING THE TRANSACTIONS

Since a great deal of paper can be consumed in printing every detail of every transaction, DBAUDIT has commands to restrict which transactions are selected and what data is printed for the selected transactions (if any).

The *SELECT* command allows you to select transactions for specified bases, datasets, programs, users, time periods, and a range of record numbers.

```
>SELECT BASE TEST
>SELECT DATASET TEST,DCOM
>SELECT USER BOB.GREEN
>SELECT BEFORE 1132
>SELECT NFROM 212
```

The *LIST* command allows you to control which transactions are printed (*LIST* CALLS) and which field values are printed for the transactions (*LIST* FIELDS). In order to print only the date and time of DBOPEN transactions, these commands would be used:

```
>LIST CALLS NONE
>LIST CALLS O+
>LIST FIELDS NONE
>LIST FIELDS D+T+
```

Here are the full options of *LIST:*

```
>LIST FIELDS N+D+T+C+L+U+P+B+S+R+K+F+M+X+
These flags determine how much information is
printed for each transaction:
L:  unique id number assigned by DBOPEN
U:  user.account,group name
P:  program.group.account
B:  basename.group.account
S:  set name/type; MA=master, DE=detail
R:  IMAGE record number of entry
D:  date of transaction
T:  time of transaction
N:  sequence number assigned by user logging
C:  call type (OPEN, CLOSE, ...)
F:  field values
K:  key-field value
M:  memos from DBMEMO, DBBEGIN, DBEND
X:  extra fields on DBOPEN (mode, etc.)

>LIST CALLS  O+C+P+D+U+B+E+A+M+T+L+
These flags enable/disable listing of the different
IMAGE and MPE intrinsic CALLS in the log-file.
  O=open C=close P=put D=delete U=update B=begin E=end
  A=abort of program between BEGIN and END calls
  T=termination of program without calling DBCLOSE
  L=logging status records (header, trailer, ...)
```

OTHER USEFUL INFORMATION

DBAUDIT also provides three other useful pieces of information for the auditor or database administrator:

1. Reliability of the log-files for recovery purposes.
   DBAUDIT checks each log-file to ensure that transactions are in the proper sequence (for date, time, transaction number and logging access number). If there are any inconsistencies in the log-file, they are detected and reported. Without DBAUDIT, the only way to test a log-file is to restore the original database and actually run a logging recovery. DBAUDIT's double-checking feature has already detected a number of bugs in IMAGE transaction logging.
2. Detection of program aborts.
   DBAUDIT reports program aborts separately from regular program DBCLOSEs. By selecting only the abnormal termination transactions, you can see which programs are aborting. This can be helpful in ensuring program quality.
3. Detection of program "abends."
   DBAUDIT reports programs which terminate with an unmatched DBBEGIN. This can happen because the program aborted during a logical transaction, because the programmer forgot to terminate the logical transaction with a DBEND, or because the system crashed during a logical transaction. DBAUDIT gives you a quick summary count of the number of ABENDs in the file (it should normally be zero), plus additional details if the count is non-zero (where in the file the ABENDs occur, whether the DBBEGIN has a put, delete, or update after it, etc.).

SUMMARY TOTALS AND STATISTICS

From this basic transaction data, it is possible to generate a number of useful summary statistics.

Transaction Breakdowns

One way of analyzing the transactions is to break them down into the different types of transactions, and then total them by program, user, base and dataset:

```
  For all bases, current:    1 entries  20 maximum.
TEST.PUB.GREEN
O:1        X:1        P:2        D:0        U:0
   DCOM               P:2        D:0        U:0

  For all progs, current:    1 entries 200 maximum.
QDBM.PUB.GREEN
O:1        X:1        P:2        D:0        U:0

  For all users, current:    1 entries 200 maximum.
BOB.GREEN,PUB
O:1        X:1        P:2        D:0        U:0
```

In these tables, O equals DBOPENs, X equals DBCLOSEs, DBBEGINs, DBENDs, and DBMEMOs, P equals DBPUTs, D equals DBDELETEs, and U equals DBUPDATEs. Note that for datasets (such as DCOM), only P, D and U totals are collected.

Summary Totals

Here are the types of summary totals provided by DBAUDIT:

```
Logging Records Read from File              32
Transactions that were Selected             6
  Transactions read but not Selected        0
Transactions Selected and Printed           6
  Transactions Selected, not Printed        0
Transactions,but no OPEN (should be 0)      0
Inconsistencies in File (should be 0)       0
Number of ABENDs in file (should be 0)      0
```

FUTURE POSSIBILITIES

Information that could be generated from the log-files, but is not currently collected by DBAUDIT, is the total "changes" to a given numeric field, such as ACCOUNT-BALANCE in a CUSTOMER-MASTER entry. By periodically summing the field values in the entire database and comparing changes in this sum with the incremental changes to that field (as recorded in transaction logging), it should be possible to ensure that all transactions are being logged (i.e., the database is in balance with the transactions).

One problem is the database in which IMAGE schema does not accurately describe the actual data fields. This situation usually happens when users over-

flow the 255 item name limit of IMAGE; it is especially common among users of QUIZ/QUICK (from Quasar Systems Ltd.), IMACS/RAPID (Hewlett-Packard's new tool acquired from David Dummer) and PROTOS (COBOL Generator from Cole and Van Sickle), since these tools use some form of data dictionary to re-define the IMAGE data entries. It is likely that DBAUDIT will be enhanced in the future to provide a user hook (via LOADPROC) to allow non-IMAGE examination and formatting of each transaction.

There is one change to the IMAGE transaction logging that HP could make to improve the audit potential. For DBUPDATE transactions to detail datasets, IMAGE logs only the record number and the data fields that were actually changed. Since the search items and sort items can never be altered by a DBUPDATE, they are not included in the values that are logged. In some applications, this can make it very difficult to determine which record was actually changed (i.e., which customer). For master datasets, the key-field value is always logged. For DBPUTs and DBDELETEs, the critical fields are always logged.

HINTS ON TRANSACTION LOGGING

1. The first thing you need in order to use transaction logging is: answers to a lot of questions. Here is where to look:
   a. PAPERS.QLIBDOC.ROBELLE; this file (which is included on all Robelle product tapes) contains several interesting articles about transaction logging, including information on performance and answers by HP to the most commonly asked questions.
   b. IMAGE MANUAL; this HP manual describes the transaction logging and recovery system, DBRECOV, DBUTIL commands to enable/disable logging and recovery, the DBBEGIN, DBEND and DBMEMO intrinsics, the IMAGE log record format and MPE log record format (do not trust this information completely).
   c. MPE INTRINSICS MANUAL; this manual describes the User Logging Facility (upon which IMAGE logging rests), the OPENLOG, WRITELOG and CLOSELOG intrinsics.
   d. MPE SYSTEMS SUPERVISOR MANUAL; this manual describes the ALTACCT and ALTUSER commands which are needed to give out LG capability (needed by user/account to access the logging facility).
   e. MPE COMMANDS MANUAL; this manual describes the GETLOG, RELLOG, ALTLOG, LISTLOG and SHOWLOGSTATUS commands (these require LG capability) and the User Logging Facility.
   f. MPE CONSOLE OPERATOR MANUAL; this manual describes the LOG command, which can only be executed on the master console and which is needed to start and stop the actual logging process.
2. Should I log to disc or tape?
   a. The log-file may reside on either disc or tape.
   b. If on disc, the integrity of the log-file may be no better than that of your data. If you reach EOF on disc, logging stops, subsequent transactions are rejected, and data may be inconsistent.
   c. If on tape, the tape drive is dedicated to logging for the duration of the logging process. If logging to tape, transactions go to disc temporarily to smooth the data flow. If EOT is reached, transactions go to disc while the operator mounts another reel.
3. Power Fail Recovery (Tape)
   a. After power is back on and the system is running, the console will receive a NOT READY message for the tape.
   b. Press the following buttons on the tape drive in this order:
      LOAD
      RESET
      ONLINE
   c. After the ONLINE button is pressed, the tape will move back to the beginning of data and move forward to recover the log records. The system will then resume logging.
4. Getting Started in Transaction Logging
   a. You will need the cooperation of your system manager, your account manager, and someone at the master console. Arrange this cooperation first, or you will be very frustrated. In this example, it is assumed that you wish to turn logging on for a single database, generate a number of transactions, then disable logging and experiment with printing audit reports via DBAUDIT.
   b. Have the system manager give LG capability to your account, and your account manager give LG capability to your user name.

```
:HELLO MANAGER.SYS
:RUN LISTDIR2.PUB.SYS
>LISTACCT GREEN
  ....
 CAP: AM,AL,GL,ND,SF,IA,BA,PH,DS,MR
```

```
>EXIT
:ALTACCT GREEN;CAP=AM,AL,GL,ND,SF,IA,BA,PH,DS,MR,LG
:BYE

:HELLO MGR.GREEN
:ALTUSER MGR.GREEN;&
:CAP=AM,AL,GL,ND,SF,IA,BA,PH,DS,MR,LG
:ALTUSER BOB.GREEN;&
:CAP=AM,AL,GL,ND,SF,IA,BA,PH,DS,MR,LG
:BYE
```

c. Log on with LG capability, build a log-file on disc, acquire a log identifier with the GETLOG command and link it to the log-file. If you are already logged on, don't forget to log off and log back on (otherwise, you will not have the LG capability that you have given yourself above).

```
:BUILD TESTLOG;DISC=3000,8,1;CODE=LOG
:GETLOG AUDIT;LOG=TESTLOG
```

d. Use DBUTIL to link your database to the log identifier.

```
:RUN DBUTIL.PUB.SYS
>SET TEST LOGID=AUDIT    (assume base=TEST)
>EXIT
```

e. The first three steps are one-time operations; you do not have to do them again (for this combination of user, account, log-file, log identifier, and database). At a later time, you can link other databases to this log identifier. The steps that follow are repeated for each "cycle" of transactions that you wish to capture.

f. Ensure that no one is accessing the database, then use DBUTIL to enable LOGGING for the database. Once you have done that, no one can open the database until the log-identifier has been "activated" at the master console.

```
:RUN DBUTIL.PUB.SYS
>ENABLE TEST FOR LOGGING
>EXIT
:LISTLOG
  LOGID          CREATOR      LOGFILE
  AUDIT          BOB.GREEN    TESTLOG.PUB.GREEN
:SHOWLOGSTATUS
NO LOGGING PROCESS CURRENTLY RUNNING (CIWARN 1230)
:RUN APPLPROG.PUB.GREEN
**** UNABLE TO OPEN DATABASE: TEST ****
LOGGING ENABLED AND NO LOG PROCESS RUNNING
```

g. Now comes the hard part, unless you are located at the computer site. Convince someone to do a :LOG AUDIT;START command at the console. Perhaps a phone call to an influential friend would be useful at this point.

```
:LOG AUDIT;START
```

h. You can tell if logging has been started with the :SHOWLOGSTATUS command.

i. Now log on a number of terminals and generate changes to the TEST database using QUERY, application programs, or SUPRTOOL/Robelle. All of these changes will be logged to the file TESTLOG.PUB.GREEN as they occur.

j. Once again, contact the console and have a command entered to stop the log-file.

```
:LOG AUDIT;STOP
```

k. Terminate the programs that are generating the transactions; when the last database accessor has closed the database, MPE will terminate the log process and close the logfile. Now, use DBUTIL to disable logging to this database (so that DBAUDIT can access it!).

```
:RUN DBUTIL.PUB.SYS
>DISABLE TEST FOR LOGGING
>EXIT
```

l. Run DBAUDIT and specify TESTLOG as the source of input records. If you do not want the report on the lineprinter, you can use a :FILE command to the file DBREPORT to redirect it to $STDLIST or to a disc file.

```
:FILE DBREPORT=$STDLIST;REC=-79
:RUN DBAUDIT.PUB.ROBELLE
>INPUT TESTLOG
>EXIT
```

5. Users report that the system load of transaction logging may not be as bad as was first rumored. Considering the performance improvement that is likely to accompany MPE IV, now may be a good time for users to consider activating transaction logging for their databases.
6. In one of the releases of MPE IV, IMAGE was changed in a way that can affect user programs. Previously, a user program could invoke the DBBEGIN, DBEND or DBMEMO intrinsics whether or not logging was active for the database in question. The intrinsics always returned STATUS=0, as long as the parameters were legal. Now, these three intrinsics return an error (non-zero STATUS) if logging is not active at the time of the call (i.e., the DBBEGIN, DBEND or DBMEMO could not be logged). User programs that don't care whether logging is running, must *not* check the STATUS result. User programs that wish to ensure that the operator has activated logging can now do so by checking the STATUS from these intrinsics.