EOPS-II

The parts:

eop_run –

1 launches EOP on a list of portfolios contained in the config file (see *The EOP Run Configuration File* on Page 2)
2 restarts EOP on a single portfolio
3 restarts portfolio monitor on a single portfolio

portfolio_mon –

4 launched by **eop_run** each time an EOP is submitted for a portfolio, each time an EOP is restarted for a portfolio, and each time **eop_run** is invoked with the -m argument.
5 it iteratively polls the database for status on EOP jobs for a given portfolio; based on the status it may or may not continue monitoring, likewise, it may or may not send email, and it may or may not warn of a stalled condition. (see *EOP Statuses* on Page 5).

portfolio_stat –

6 called by **portfolio_mon** to obtain the current EOP status of the portfolio

eop_run.cfg –

7 contains parameters for **eop_run** and **portfolio_mon** to use (see *The EOP Run Configuration file* on Page 2)

eop_run:

Generally, **eop_run** will be the only script that you will use directly; **portfolio_mon** and **portfolio_stat** are run under its aegis.

The usage of **eop_run** is:

8 To submit all the portfolios listed in the config file (see *The EOP Run Configuration file* on Page 2):

```
eop_run -s environment config username [client-password [dbms-pwd]]
```

9 To restart a given portfolio:

```
eop_run -r environment portfolio config username \
      [client-password [dbms-pwd]]
```

10 To restart the monitor process for a portfolio:

```
eop_run -m environment portfolio config username \
      [client-password [dbms-pwd]]
```

The Parameters of **eop_run** are:

flag: **-s** - submit all portfolios
-r - resubmit one portfolio
-m - restart monitoring one portfolio

environment: a regular LeasePak logical database environment (DBMS-neutral)

portfolio: the specific portfolio to which to apply the **-r** and **-m** options

config: filename of configuration file (see *The EOP Run Configuration file* on Page 2)

username: name of user submitting and monitoring EOP. Must be valid login in DBMS and in UNIX, a valid user in the logical database, and must have a LeasePak security record, and be authorized to use U0401 and U0404

client-password: the "clear" "pretty" password that is entered into the LeasePak client when connecting to LeasePak; optional; prompted for if omitted

dbms-pwd: the "scrambled" "ugly" SQL password that is used to log onto the LeasePak logical database; disallowed if ***client-password*** is omitted, otherwise optional; prompted for if omitted or disallowed

The EOP Run Configuration file

This file is used to pass a number of important parameters to **eop_run** and to **portfolio_mon**. The defaults listed in the description of the parameters are really for testing. The values actually given in the sample config file are probably more appropriate to production, and even then perhaps could go higher.

PROGRAM_LOC – this is how **eop_run** knows how to find **portfolio_mon** and **eop_launch** and **eop_restart**, and it is how **portfolio_mon** finds **portfolio_stat**.

PORTFOLIOS - this quoted, space-separated list of portfolios is what **eop_run -s** will start**. Eop_run -r** and **eop_run -m** don't care if their portfolio is or isn't in this list.

ACCRUALS & INVOICES - these apply across the board to all portfolios in **PORTFOLIOS** when it comes down to starting EOP. If there are reasons to have these switches vary by portfolio, then portfolios will have to be grouped together in different config files each with its own combination of these switches.

CHECK_NUMBER - this will be applied across the board to all portfolios in **PORTFOLIOS**, making it less useful than it might appear. If LeasePak is configured for manual check numbers, and many portfolios, then this could present some issues.

SUBMIT_INTERVAL - when multiple portfolios are listed in **PORTFOLIOS**, it is general practice to stagger their start times. There is a lot of very intense upfront activity involving the batch queues for each portfolio, and staggered start times reduce resource contention. The right length of time to wait between each submit is likely to be very site-specific. The site administrator will have to experiment with this and other parameters, both in **EOPS-II** and in other LeasePak and queue management areas. **SUBMIT_INTERVAL**, measured in seconds, puts **eop_run** to sleep between each portfolio start.

MONITOR_INTERVAL - **portfolio_mon** calls **portfolio_stat** periodically to determine the status of the EOP they're monitoring. This parameter determines how often **portfolio_stat** is asked to run. While EOP is running, each invocation of **portfolio_stat** results in a line being added to the portfolio log file, in sort of a heart beat fashion. The default of 10 seconds is OK for testing but in the real world, 60 seconds or more is more likely sufficient. Even 5, 10, 15 or even 30 minutes (300, 600, 900 or 1800 seconds) may be sufficiently often.

STALL_TOLERANCE - This parameter gives the number of **MONITOR_INTERVAL**s to wait for a stalled EOP to start running before sending email**. STALL_TOLERANCE** is just the starting point in stall handling. The default, 120, times the default **MONITOR_INTERVAL** of 10 seconds gives 1200 seconds or 20 minutes. If the **MONITOR_INTERVAL** is 1800 seconds then **STALL_TOLERANCE** might be 1 or 2, giving 30 or 60 minutes. It depends on how closely daily processing is scheduled.

MAIL_TO & **MAIL_FROM** - if it is desired to have **EOPS-II** send email, these two parameters must be set to appropriate values, and sendmail must be properly configured on the server. Multiple addresses may be used if separated by semicolons (";") and the list in "-quoted. Mail is sent

- on each failure to submit or resubmit a portfolio,
- on each portfolio termination, whether failure or successful completion,
- when a stalled portfolio exceeds **STALL_TOLERANCE**, and
- when a stalled portfolio starts to run (but only if mail was sent our previously about the stall).

PRE_EOP_PROC, START_PORT_PROC, END_PORT_PROC - these three parameters specify command lines to be executed by **eop_run** or **portfolio_mon** at appointed times. They are to be complete command lines, as **EOPS-II** will not add to them in any way except by the action of the shell itself. They may contain parameter references ($-values) to any parameter defined at the times they are processed. They are processed twice, once when the config file is read and the assignments are evaluated and performed, and once upon invocation, which is done in the form of an 'eval': **eval `echo** $**PRE_EOP_PROC`** for instance. The variable value of **PRE_EOP_PROC** is made available as the statement eval is to evaluate by **`echo ...`**, and then any further shell substitutions or metacharacter processing, etc, are performed at that time. Then the resulting command line is executed.

PRE_EOP_PROC - This is performed by **eop_run -s** before any portfolios are processed. The example in the config file simply clears the "screen log" generated by **eop_run**'s main loop, but can be used for anything. Backing up the database (and the data directory!!!) before starting EOP is extremely advisable; this is a perfect place to do it. **PRE_EOP_PROC** is called only on **eop_run -s**, never on **eop_run -r** or **eop_run -m**.

START_PORT_PROC - This is performed by **eop_run** immediately before submitting each portfolio. In this config file, it is used to call a small script that clears out a destination directory that is to receive files at the end of the portfolio's processing. **START_PORT_PROC** is called only on **eop_run -s**, never on **eop_run -r** or **eop_run -m**.

END_PORT_PROC - This is performed by **portfolio_mon** on successful completion of EOP in the portfolio being monitored. This config file uses this parameter to call a small script that copies that portfolio's **udata** files into the directory cleared out by **START_PORT_PROC**. **END_PORT_PROC** is called only on successful completion of the portfolio, never on failure.

LOGICALS – This allows inclusion of "client logicals" similar to the .ini file used by the LeasePak Windows client. It must be "-quoted, with each logical assignment or definition separated by semicolons (";").

*(The **START_PORT_PROC** and **END_PORT_PROC** scripts are listed out on Page 4, following **eop_run.cfg**)*

[[eop_run.cfg
[[

```
# Required:
#   PROGRAM_LOC=        - path where portfolio_stat, portfolio_mon, eop_launch and
#                         eop_restart are located, typically $ubin
#   PORTFOLIOS=         - "-quoted list of portfolios to submit
#   ACCRUALS=           - Y or N
#   INVOICES=           - Y or N
# Optional:
#   SUBMIT_INTERVAL=    - defaults to 5 seconds between submitting portfolios
#   MONITOR_INTERVAL=   - defaults to 10 seconds between checks for EOP status
#   STALL_TOLERANCE=    - defaults to 120 iterations of MONITOR_INTERVAL
#   CHECK_NUMBER=       - if required by LeasePak setup
#   MAIL_TO=            - email address
#   MAIL_FROM=          - email address
#   PRE_EOP_PROC=       - complete "-quoted path and text of job to perform
#                         before the first portfolio is submitted, such as backups
#   START_PORT_PROC=    - complete "-quoted path and text of job to perform
#                         before each portfolio is submitted
#   END_PORT_PROC=      - complete "-quoted path and text of job to perform
#                         when each portfolio completes
#    LOGICALS=          - "-quoted list of ";" separated client logical values
```

PROGRAM_LOC=$ubin
PORTFOLIOS="2 4"
ACCRUALS=Y
INVOICES=Y
SUBMIT_INTERVAL=45
MONITOR_INTERVAL=60
STALL_TOLERANCE=30
CHECK_NUMBER=
MAIL_TO=garysk@mccue.com
MAIL_FROM=eop@mccue.com
PRE_EOP_PROC="rm -f $ueop/log/eop_run.log"
START_PORT_PROC="$uprg/clear_data $PORTFOLIO $ENVDIR/ready_data"
END_PORT_PROC="$uprg/copy_data $PORTFOLIO $ENVDIR/ready_data"
LOGICALS="LEASEPAK_CORE=1;LEASEPAK_CMD_BUFFER_LOG=1"

[[clear_data
[[

```
PORT=$1
DEST=$2
```

rm -f $DEST/p${PORT}*.* $DEST/*p${PORT}.* $DEST/p${PORT}_done.txt

[[copy_data
[[

```
PORT=$1
DEST=$2
find $udata/ \( -name "p${PORT}*.*" -o -name "*p${PORT}.*" \) -print |
while read FILE; do
        if cp -f $FILE $DEST; then
                echo "copy_data $PORT $DEST: $FILE copied"
        else
                echo "copy_data $PORT $DEST: $FILE copy FAILED"
        fi
done
touch $DEST/p${PORT}_done.txt
```

EOP Statuses

There are 16 LeasePak EOP status codes; three are not supported by **EOPS-II** (split EOP).
The remaining 13 statuses provide more information than we need for the purposes of monitoring EOP's progress, detecting when human intervention is required, and detecting completion. Therefore, **`portfolio_stat`** and **`portfolio_mon`** recast them in the following manner:

Traditional Status	EOPS-II	Class	Significance	Monitor…
LPEOP_POST_HALT = 2048	103	Halted	EOP has terminated abnormally	sends mail & terminates
LPEOP_HALTED = 1024				
LPEOP_POST_FAIL = 512	101	Failed		
LPEOP_FAILED = 256				
LPEOP_CURRENT = 64	102	Running	EOP is Running	continues to run (if pending too long, EOP has stalled; sends email) (sends email again if it starts to run after stalling; see Page 6)
LPEOP_PENDING = 32	104	Pending (Stalled)		
LPEOP_QUEUE_STPD = 16				
LPEOP_HOLD = 8				
LPEOP_AFTER = 4				
LPEOP_PENDING_INIT = 2				
LPEOP_HOLD_WAIT_BCK = 1				
No status 1 – 4096	105	Unknown		
LPEOP_SKIPPED = 4096	0	Done	EOP is complete	executes END_PORT_PROC, sends email & terminates
LPEOP_COMPLETED = 128				
Commandline error	201	Setup	portfolio_stat has terminated abnormally	terminates
Script error	202			
Xsql error	203			

How the portfolio status is derived (since there will nearly always be a mix of statuses present):

- All the statuses of the portfolio's jobs are sorted uniquely; this usually leaves a list of 2 to 4 statuses.
- Starting at the top of the above table, the list is searched for each traditional status in the table in turn from 2048 down to 1.
- When a status is found in the list, then that becomes the **EOPS-II** status of the entire portfolio and searching stops.
- If no status in the range 2048 to 1 is found in the list, then these steps are performed:
- The list is searched for 4096, if found *skipped* is noted.
- The list is searched for 128, if found *completed* is noted.
- If neither *skipped* nor *completed* is noted, then the **EOPS-II** status is 105, otherwise it is 0.

Stall Handling

When the status of a portfolio as returned from **`portfolio_stat`** is *Pending*, **`portfolio_mon`** starts a counter, which is then incremented each time **`portfolio_stat`** returns *Pending*. If that counter exceeds the **STALL_TOLERANCE** parameter (see *EOP Run Configuration file* on Page 2), and email is sent notifying the operator that the portfolio has *Stalled*. Some situations are innocuous and some indicate problems which may require operator intervention,

The stall counter is then set back to zero and **STALL_TOLERANCE** is doubled. This is so that it doesn't become too pesky, yet does not just give up either. Again, when **STALL_TOLERANCE** is exceeded, another email is sent. This process repeats indefinitely.

When a *Stalled* portfolio begins to run again the status goes back to *Running*. If **STALL_TOLERANCE** had been exceeded (and email alert of the stall was sent), then email is sent alerting the operator that the portfolio is running. **STALL_TOLERANCE** is then reset to an average of its last value and its initial value, making it somewhat more tolerant of subsequent *Stalled* periods.

For example, If **STALL_TOLERANCE** is given as 5 in the config file, then email will be sent on the 5th time *Pending*, and then on the 10th time *Pending*, and on the 20th time *Pending*. If the portfolio begins to run again, then **STALL_TOLERANCE** is set to (20 + 5) / 2 = 12. The next time that the portfolio is *Pending* more than 12 cycles then it will be considered *Stalled*.

The scope of **STALL_TOLERANCE** is restricted to one portfolio and one invocation of **`portfolio_mon`**. If portfolio_mon has to be restarted, then **STALL_TOLERANCE** will be set as indicated in the config file.

Installing EOPS-II

As part of a regular build, the following files will be found in the locations indicated:

Directory	Filename	Owner:Group	Modes	Description
$ubin	eop_run	MSIADMIN:MSIGROUP	550	main EOPS-II tool
$ubin	eop_launch	MSIADMIN:MSIGROUP	550	binary client
$ubin	eop_restart	MSIADMIN:MSIGROUP	550	binary client
$ubin	portfolio_mon	MSIADMIN:MSIGROUP	550	portfolio EOP monitor
$ubin	portfolio_stat	MSIADMIN:MSIGROUP	550	portfolio EOP status fetcher
$ulib	eop_run.cfg	MSIADMIN:MSIGROUP	440	SAMPLE config file
$ulib	clear_data	MSIADMIN:MSIGROUP	440	SAMPLE pre-portfolio
$ulib	copy_data	MSIADMIN:MSIGROUP	440	SAMPLE postportfolio

NOTE: The sample files in $ulib should never be modified. Instead, make copies of them in the database environment's $uprg directory, which is set up to contain user-customized scripts. The eop_run.cfg entries for scripts should contain absolute pathing to the commands desired as there are no guarantees made as to the validity of any shell path variable during End of Period.

*** * * WARNING * * * WARNING * * * WARNING * * * WARNING * * * WARNING * * ***

Do not attempt to combine any of the scripts or the config file from EOP Suite with **EOPS-II**. EOP Suite and **EOPS-II** script systems are completely incompatible. Specifically:

USE THESE FILES	NOT THESE FILES
eop_run	eop_submit_job
eop_run.cfg	eop_submit.cfg
portfolio_mon	eop_status_loop
portfolio_stat	lpeopmon

Configuring EOPS-II

1. Still logged in as the MSIADMIN for v52a, retreive copies of the *SAMPLE* config file and copy scripts from **$ulib**. Do not modify the originals in **$ulib**, Copy the scripts **clear_data** and **copy_data** into **$ubin**, and set their modes to 550. They should remain owned by MSIADMIN:MSIGROUP.

   ```
   cp $ulib/clear_data $ulib/copy_data $ubin
   chmod 550 $ubin/copy_data $ubin/clear_data
   ```

   Of course, if you need to change them make them writable but be sure to reprotect them when you're done.

2. Copy the config file sample from $ulib to your first environment where you wish to implement **EOPS-II**. Set its modes to 660 for now, as it will neeed to be customized before use.

   ```
   cp  $ulib/eop_run.cfg  $uetc
   chmod 660 $uetc/eop_run.cfg
   ```

   You can use what ever names your choose for the config files.

3. Edit the config file as appropriate for your site. Try to do your first test on a level7 database just to get the kinks worked out. See detailed explanations beginning on Page 2 of the config file content.
4. Make sure your test database environment is ready to run an EOP.
5. To use **EOPS-II**, you must set accruals and invoicing in the Portfolio Maintenance [U0212] update to run every day. To not run accruals or invoicing on a particular day, set ACCRUALS and INVOICES to N. You can associate different configuration files with eop_run and schedule these different versions to run at different times through cron, allowing you to determine when accruals and invoices are processed. Alternatively, you can provide a script to generate appropriate config files based on criteria you determine.
6. Once this is set up you can test it from the command prompt.
7. Log into the server as the user who will be used to submit and monitor the EOP.
8. Execute the following command line:

   eop_run –s <*your-env-name*> $uetc/eop_run.cfg <*username*> <*password*>

9. The logfiles used are **$ueop/log/eop_run.log** and **$ueop/log/p<port>_status.log** for each portfolio. Running **tail –f** on the portfolio log will allow you to watch the "heart beat" messages from the monitor program.
10. The same eop_run command can be executed from within a cron-driven script as well.