

---

## Rule DAS111:      **SEEKING WAS PROBABLY CAUSED BY INDEPENDENT APPLICATIONS**

---

**Finding:**      CPEXpert determined that seeking was the major cause of delay in DASD response for the device. More than half of the I/O queuing to the device was explained using a queuing model. Consequently, CPEXpert believes that the seeking probably was caused by independent applications.

**Impact:**      This finding may have a MEDIUM IMPACT or HIGH IMPACT on the performance of the device. If the finding is correct, and the independent applications are referencing different files, then the corrective actions could result in significant performance improvements.

**Logic flow:**    The following rules cause this rule to be invoked:  
                  DAS100: Volume with the worst overall performance  
                  DAS110: Seeking was the major cause of response delay

**Discussion:**    CPEXpert uses a M/M/1 queuing model to calculate an estimated queue time for each measurement interval being analyzed. The underlying assumptions of the model are exponential interarrival times, exponential service distributions, and an infinite population. If device activity occurs in this way, the queuing model can predict the expected queuing delays.

If the queuing delay as measured by RMF is significantly different from the estimated delay from the model, it would be clear that the activity did not occur in a random fashion, and most likely the cause of the difference would be that the interarrival times are not randomly distributed.

However, the I/O delay to this pack was fairly well explained by the queuing model. More than half of the I/O queuing delay was explained by the model, for a majority of the measurement intervals. This indicates that the activity was mostly random, as would be expected from independent applications accessing files on the pack.

**Suggestion:**    The most improvement for this pack would likely result from (1) separating the files to different packs, (2) rearranging the files within the pack, (3) tuning the file structure (for example, compressing a shared partitioned data set (PDS), (4) scheduling the applications to avoid contention, or (5) examining the applications doing most of the I/O.

---

You can identify the files with the most activity in one of several ways, depending upon the volume and the application involved. The method you select should depend upon the availability of information and tools in your environment.

- Discuss the file activity with applications personnel or systems personnel (depending upon the volume). If knowledgeable personnel are available, this is often the quickest and easiest way to determine how to solve the problems. Once a file access problem has been brought to their attention, applications or systems personnel often can easily decide upon a good solution.
- Use an exit available in MXG or in MICS to select Type 30(DD) information just for the volume. The Type 30(DD) information is rarely retained in a performance data base because it is so voluminous. However, you easily can code an exit in MXG or MICS to select the Type 30(DD) information for a specific volume. The amount of data selected would not be too large in most cases.

You can then write a SAS program to list the DD names used by applications, weighted by the number of accesses. For example, you could code the following to analyze data extracted during MXG update processing:

```
PROC FREQ DATA=pdblib.file;  
  WHERE DEVNR = addressX;  
  TABLES DDNAME/OUT=TEMP NOPRINT;  
  WEIGHT EXCPS;  
PROC SORT DATA=TEMP;  
  BY DESCENDING COUNT;  
PROC PRINT;  
RUN;
```

The device address is displayed by RULE DAS100, so the "address" in the above coding would be replaced with the actual address displayed by that rule. If you have a MXG performance data base, the address is retained in hexadecimal format, so you would suffix an X to the address.

If you have a MICS performance data base, the variable names must be changed appropriately (for example, DEVNR would be changed to DEVADDR). Additionally, the address is retained in MICS as a character representation of the value, so you do not need to suffix an X to the address. Rather, you must enclose the address in quotes.

The result from the above code would be a list of all DDNAMEs referencing the device being analyzed, weighted by the number of

---

EXCPs to the device, and ordered descendingly with the DD statements for the most active files listed first. You often cannot be certain that the most referenced files are causing problems. However, in most cases, you will find that only a few files (often only 2 or 3) account for over 90% of the accesses. These files generally will be the ones causing arm contention problems.

You can then write a SAS program to list the applications (jobs) referencing the device, weighted by the number of accesses. For example, you could code the following to analyze data extracted during MXG update processing:

```
PROC FREQ DATA=pdbl.lib.file;  
  WHERE DEVNR = addressX;  
  TABLES JOB/OUT=TEMP NOPRINT;  
  WEIGHT EXCPS;  
PROC SORT DATA=TEMP;  
  BY DESCENDING COUNT;  
PROC PRINT;  
RUN;
```

The result from the above code would be a list of all jobs referencing the device being analyzed, weighted by the number of EXCPs to the device, and ordered descendingly with the jobs with the most active files on the device listed first. These jobs generally will be the ones causing arm contention problems. (If you have a MICS performance data base, you would change the code as described earlier.)

- Use a commercially-available DASD activity monitor to isolate the files accessed on the problem volume. If such a monitor is available, this would be a direct way to determine the problems.