Rule WLM024: More than three periods specified for a service class

Finding: More than three periods were specified for a service class.

Impact: This finding should be viewed as generally having a LOW IMPACT or MEDIUM IMPACT on the performance of the service class periods involved. Under some circumstances, the finding could have a HIGH IMPACT on the performance of the service class periods involved. Additionally, there is an impact as a result of increased (and unnecessary) overhead.

Logic flow: This a basic finding. There are no predecessor rules.

Discussion: A service class may be broken into multiple service class periods. Each service class has Period 1 automatically defined. Optionally, installations can define up to seven additional service class periods (although typically no more than one or two additional service class periods are defined). Each service class period can have its own performance goals, defined to the Workload Manager via the Workload Manager ISPF panels.

An address space (TSO transaction, batch job step, etc.) begins in Period 1 of the service class to which it is assigned. The address space transitions from Period 1 to Period 2 (and to subsequent periods), based upon the accumulation of "service" by the transaction. The "service" required by the address space is a combination of CPU resources, I/O resources, and memory resources. Performance period duration (DUR) values can be established so the system dynamically detects non-interactive (batch-like) transactions and migrates these transactions to lower performance periods, with less stringent performance goals and perhaps less goal importance.

The normal purpose of defining multiple service class periods is to give higher priority to interactive transactions, short batch job steps, etc. Overall response is decreased (and overall throughput is increased) when address spaces requiring relatively few resources are not conflicting with those address spaces requiring substantial resources.

It is instructive to examine the resources that may be in conflict. The resources mainly consist of CPU cycles, processor storage, and I/O operations.

- Prior to MVS/ESA SP5, the CPU resource was the main area of conflict. CPU dispatching priority is established at the performance group period

level. An address space in a particular performance group period executed at the same CPU dispatching priority as all other address spaces in the performance group period[1]. Non-interactive transactions could cause interactive transactions to be denied access to a processor for an unacceptable interval.

- Conflict with processor storage occurs when non-interactive transactions require either central storage or expanded storage, causing central storage page fault or expanded storage delays for interactive transactions. Non-interactive transactions often (but not always) require a significant amount of processor storage.

- Non-interactive transactions can cause conflict with I/O operations by either path, controller, or device delays to interactive transactions. Fortunately, (1) modern systems normally have multiple paths to devices and path delay is not usually a serious concern, (2) and non-interactive transactions normally do not access the same devices as interactive transactions, and (3) interactive transactions normally do not execute many I/O operations. The combined effect is that conflict for I/O operations rarely is a cause of serious performance problems between interactive and non-interactive transactions.

As the above discussion illustrates, the main concern prior to MVS/ESA SP5 was CPU dispatching. This concern has been almost completely eliminated with SP5.

- Prior to SP5, CPU dispatching was on a "first come first served" or FIFO basis from the Dispatcher's True Ready Queue. A non-interactive transaction could easily deny an interactive transaction from access to a processor for a prolonged period; the non-interactive transaction could retain control of a processor until it either completed or it voluntarily entered a Wait state.

- With SP5, the Dispatcher algorithm has been redesigned. One aspect of the redesign is that each interrupted dispatchable unit (a TCB or SRB) is placed at the end of the dispatching queue for its current dispatching priority.

  For example, suppose that a dispatchable unit was executing with a priority of 240. When the dispatchable unit was interrupted, it would be placed behind all other dispatchable units on the dispatch queue with a dispatching priority of 240 (but ahead of all dispatchable units with a dispatching priority of 239 and lower).

[1]This discussion ignores the implications of time-slicing algorithms and ignores the implications of the Mean-time-to-Wait (MTTW) dispatching algorithms. These special case algorithms are not normally used for Period 1 or Period 2 of multi-period performance groups.

Since interrupts occur frequently, there is a constant adjustment of dispatching order. This readjustment requires practically no additional overhead, as only queue pointers are being manipulated.

One effect of the redesign is to prevent any dispatchable unit from "seizing" a processor and denying access to other dispatchable units at the same priority. Since a dispatchable unit cannot deny access to other dispatchable units, a non-interactive transaction cannot deny CPU access to a Ready interactive transaction.

Consequently, with SP5, there is less need to "separate" non-interactive and interactive transactions, and there is less need to specify multiple periods. Nonetheless, the need to separate interactive and non-interactive transactions still exists. The SRM will be able to differentiate between interactive and non-interactive transactions only if the values specified for the DUR keyword roughly correspond to the resource requirements of trivial, interactive, and non-interactive transactions.

However, there can be a **significant** performance impact of defining an excess number of service class periods!

- The Workload Manager will adjust system resource allocation in an attempt to improve performance of only one service class period during a policy adjustment interval[2].

  Adjustment to improve performance of only one service class period is done because the Workload Manager must observe the results of the adjustment: whether the adjustment helped performance, hurt performance, or had no effect. If adjustments were made to improve performance of more than one service class period, it would be impossible to determine which adjustment helped or hurt which service class period. Consequently, the Workload Manager takes actions to improve the performance of only one service class period during each policy adjustment interval.

  The policy adjustment interval is 10 seconds. If too many service class periods have been defined, the Workload Manager may be able to adjust system resource allocation to help only a few service class periods (the most important service class periods with the worst performance). Performance of other service class periods may never be improved, or performance improvement actions may take a long time - simply because of the elapsed time necessary for the Workload Manager to make

[2]The Workload Manager will consider adjustments to improve performance of several service classes (starting with the most important service class which has the highest Performance Index). If performance of the first service class analyzed cannot be appreciably improved, the Workload Manager will select the next worst performing service class, etc. After the Workload Manager has "committed" to a policy adjustment for a service class, it will stop analysis and adjust resources for no other service class.

changes, collect data, analyze the effect of the changes, make additional changes, collect more data, and continue the process.

- Perhaps of equal significance is the overhead associated with analyzing service class periods. The Workload Manager tries to improve performance of the service class period with the worst performance at the highest goal importance. Resources may be taken from the least important service class period with the best performance. The Workload Manager will not simply remove and add resources; rather, the Workload Manager will analyze the net value of the planned action.

  The Workload Manager will not add resources unless there is an appreciable net gain to the service class period receiving the resources. Within the same goal importance, the Workload Manager will not remove resources from a service class period unless the net gain to the receiver outweighs the net loss to the service class period the resources are being removed from. The overhead involved with the analysis and decision process increases as the number of service class periods becomes large.

Additionally, the system will incur other overhead caused by excess service class periods. As examples of the overhead:

- Additional SRM control blocks are created and processed.
- Additional Workload Manager control blocks are created and processed.
- RMF requires additional processing of the control blocks.
- SMF Type 72(Subtype 3) records are written for each service class period defined (regardless of whether the service class periods are used by the Workload Manager).

CPExpert produces Rule WLM024 when more than three periods are defined for a service class.

The following example illustrates the output from Rule WLM024:


```
RULE WLM024:  MORE THAN THREE PERIODS WERE SPECIFIED FOR A SERVICE CLASS

  More than three periods were specified for the TSOPROD Service Class.
  CPExpert believes that you may wish to revise the service class period
  structure to specify no more than three periods for the service class.
  Please refer to Rule WLM024 in the WLM Component User Manual for a
  discussion of the reasons for this recommendation.
```

Suggestion: CPExpert suggests that you review the performance period structure associated with the service class identified by this rule. Unless there are unusual circumstances, CPExpert suggests that you define no more than three[3] performance periods.

Some performance analysts have suggested that no more than two periods be defined[4]. Specifically, some authors have suggested defining TSO Period 1 with a very high duration (e.g., 50,000 service units) and defining TSO Period 2 to handle all other transactions. CPExpert strongly disagrees with this advice.

- Management at most installations would want a TSO transaction with 50,000 service units to be classified down with the "bottom feeders" (if for no other reason to discourage TSO users from sending in such a "transaction" rather than doing a batch job submit).

- The new dispatcher algorithms should minimize the CPU effect that a very large (e.g., 50,000 service unit) "transaction" would have on interactive TSO transactions in the same service class period. This would be true even if a lot of these lengthy transactions were using CPU cycles in Period 1. **That CPU time must come from somewhere, though**. All address spaces in a service class period have the same dispatching priority. The large transactions would be at the same CPU dispatching priority as TSO trivial and they would be using CPU time which probably should be available to other more important work.

- In addition to the effect on other workloads, since a very large (e.g., 50,000 service unit) "transaction" would be in the same service class period as interactive transactions, the large "transaction" would receive the "favored status" bestowed by the Workload Manager on service class periods with a short response goal. The Workload Manager gives the same expanded storage policy to all address spaces in a service class period with less than or equal to 20 seconds response goal.

[3]In fact, we would greatly appreciate being informed of any unusual circumstance in which more than three performance periods are required!

[4]Reference IBM TalkLink "Initial Dispatching" thread.

- Additionally, the Workload Manager gives a protective processor storage target to all address spaces in a service class period with a short response goal. In many shops, this favored status for large "transactions" could cause some serious performance problems[5]. If the large transactions were moved down to a period with over 20 seconds goal, the Workload Manager would treat them as individual address spaces, rather than collectively with all address spaces in the service class period.

Reference: MVS Planning: Workload Management

MVS/ESA(SP 5):	Chapter 8: Defining Service Classes and Performance Goals
OS/390 (V1R1):	Chapter 8: Defining Service Classes and Performance Goals
OS/390 (V1R2):	Chapter 8: Defining Service Classes and Performance Goals
OS/390 (V1R3):	Chapter 8: Defining Service Classes and Performance Goals
OS/390 (V2R4):	Chapter 8: Defining Service Classes and Performance Goals
OS/390 (V2R5):	Chapter 8: Defining Service Classes and Performance Goals
OS/390 (V2R6):	Chapter 8: Defining Service Classes and Performance Goals
OS/390 (V2R7):	Chapter 8: Defining Service Classes and Performance Goals
OS/390 (V2R8):	Chapter 8: Defining Service Classes and Performance Goals
OS/390 (V2R9):	Chapter 8: Defining Service Classes and Performance Goals
OS/390 (V2R10):	Chapter 8: Defining Service Classes and Performance Goals
z/OS (V1R1):	Chapter 8: Defining Service Classes and Performance Goals
z/OS (V1R2):	Chapter 8: Defining Service Classes and Performance Goals
z/OS (V1R3):	Chapter 8: Defining Service Classes and Performance Goals
z/OS (V1R4):	Chapter 8: Defining Service Classes and Performance Goals

MVS/ESA 5.1.0 Presentation Guide (GG24-4137)
Section 4: Dispatcher Redesign

[5]A very large "transaction" is not a typical TSO interaction - who knows what evil would be generated by giving these "transactions" protective processor storage?

