Rule WLM121: Significant transaction time was in Ready state

Finding: A significant amount of the transaction response time for the service class missing its performance goal was spent in the Ready state. This finding applies to service classes that are part of a subsystem (e.g., CICS transactions).

Impact: This finding has a MEDIUM IMPACT or HIGH IMPACT on performance of the service class. The level of impact depends on the percent of transaction response time spent in the Ready state.

Logic flow: The following rules cause this rule to be invoked:

Rule WLM104: Subsystem Service Class did not achieve average response goal

Rule WLM105: Subsystem Service Class did not achieve percentile response goal

Discussion: When CPExpert produces Rule WLM104 or Rule WLM105 to indicate that a subsystem service class did not achieve its performance goal, the logic of these rules tries to identify the cause of the delay. The cause of the delay initially is analyzed from the "served" service class view. The delays from the served service class are reported by CICS (with CICS/ESA Version 4.1 and later) or by IMS (with IMSVersion 5 or later). Interaction with the Workload Manager is accomplished using the Workload Management Services macros[1].

CICS reports two separate views of the transactions: the *begin_to_end phase* and the *execution phase*[2].

- **Begin_to_end phase**. The begin_to_end phase starts when CICS has classified the transaction[3]. This action normally is done in a CICS Terminal Owning Region (TOR).

- **Execution phase**. The execution phase starts when either CICS or IMS (Version 5 or later) has started an application task to process the

[1]Please refer to Section 4 of this document for more detail about the Workload Management Services macros and how the subsystems use these macros to exchange information with the Workload Manager.

[2]IMS Version 5 reports only *execution phase* samples.

[3]Classifying the transaction into a service class is done by the Workload Manager when the subsystem manager issues the IWMCLSFY macro. Please refer to Section 4 for a more complete discussion of the subsystem work manager (e.g., CICS) interaction with the Workload Manager.

transaction. For CICS, this normally is done in a CICS Application Owning Region (AOR). For IMS, this is done in an IMS Message Processing Region (MPR).

Within each phase, CICS or IMS report the "state" of the transaction, from the view of CICS or IMS. The state of the transaction is reported in the following categories[4]:

- **Idle state**. (Both CICS and IMS report this state.
- **Ready state**. Only CICS reports this state.
- **Active state**. Both CICS and IMS report this state.
- **Wait state**. Both CICS and IMS report this state, but IMS provides only Wait for I/O state and Wait for Lock state.
- **Switched state**. Only CICS reports this state.

If the subsystem supports work manager delay reporting, the delay information is available in the "Work Manager/Resource Manger State Section" of SMF Type 72 (Subtype 3) records. When a transaction service class fails to achieve its performance goal, CPExpert analyzes the information to identify the primary and secondary causes of delay.

CPExpert produces Rule WLM121 when the primary or secondary cause of delay was that the transaction service class was in the Ready state for a significant percent of its response time. The Ready state indicates that a task associated with the transaction was ready to execute, but was not selected by the work manager.

For CICS transactions, this is the time accounted for by tasks executing in the CICS region. These tasks would be shown as "Dispatchable" by the CEMT INQUIRE TASK command.

The following example illustrates the output from Rule WLM121:

[4]Please refer to Section 4 of this document for a more comprehensive discussion of the transaction states and the interaction between the subsystem (CICS or IMS) and the Workload Manager.


```
RULE WLM121:  SIGNIFICANT TRANSACTION TIME WAS IN READY STATE

   A significant amount of the transaction response time for CICUSRTX Service
   Class was spent in the Ready State.  For CICS transactions, this is the
   time accounted for by tasks that were not executing in the CICS region,
   but that were ready to be dispatched.  The tasks were not dispatched
   because CICS had given priority to another task.  These tasks would be
   shown as "Dispatchable" by the CEMT INQUIRE TASK command.  If this
   finding is consistently made for an important service class, you may
   wish to consider (1) investigating the long-running tasks that ARE
   being dispatched, (2) adjusting the priority which CICS gives to tasks,
   or (3) adding another CICS region to reduce the Ready time.
```

Suggestion: CPExpert suggests that you verify the performance goals specified for the transaction service class that has missed its performance goal.

If the performance goals for the transaction service class represent your management objectives, CPExpert suggests that you consider the following alternatives:

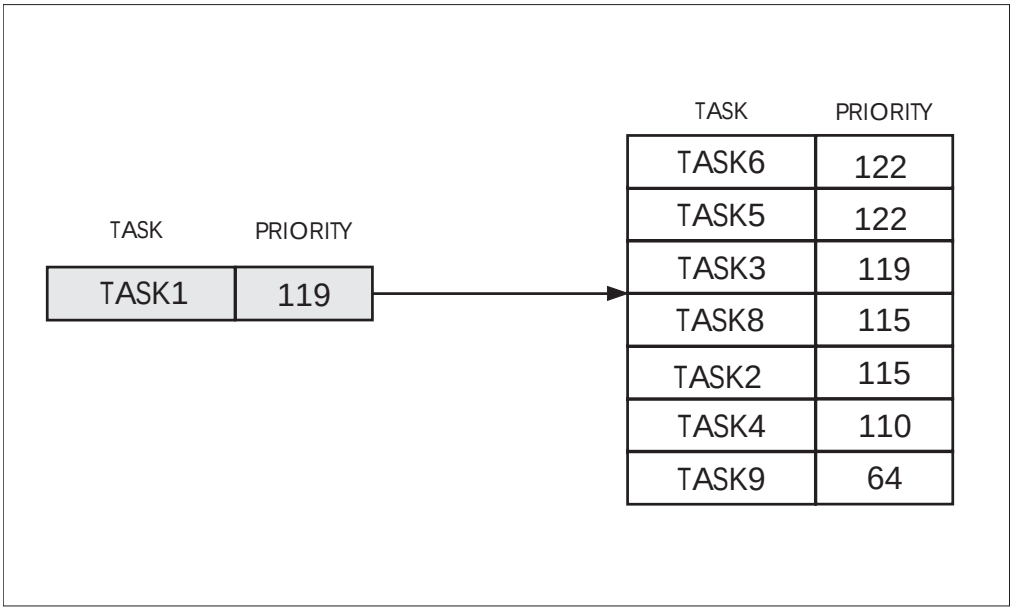
- **Review CICS task prioritization**. The task supporting the transaction service class that missed its performance goal was waiting for dispatch within the CICS region, since the CPExpert determined that the tasks spent a significant amount of time in the Ready state. One way to improve the response of important transactions is to give specific tasks preference in being dispatched by CICS.

  Dispatching priority of tasks **within a CICS region**[5] is specified in three ways: (1) priority by terminal in the CEDA TERMINAL definition (the value of the TERMPRIORITY keyword), (2) priority by transaction in the CEDA TRANSACTION definition (the value of the PRIORITY keyword), and (3) priority by operator in the signon table (the value of the OPRTY keyword in the SNT). Additionally, the three priorities can be specified via the CEMT command. The overall priority is determined by summing the priorities in the three definitions for each task, with a maximum resulting priority of 255.

  CICS maintains a dispatch queue of tasks that are ready to execute. The dispatch queue is ordered by priority, and CICS selects tasks from the top of the queue to dispatch. If task prioritization is not implemented, tasks are placed on the bottom of the queue as they become ready to execute. Thus, CICS selects tasks for dispatching in the order in which the tasks becomes ready to execute.

[5]The dispatching priority within a CICS region has no relationship to the dispatching priority from the perspective of MVS. The dispatching priority within the CICS region controls the order in which tasks are placed onto the dispatching queue in the region.

If task prioritization is implemented, a task that becomes ready to execute is placed on the queue based on its priority. A high priority task becoming ready to execute is placed on the queue ahead of all lower priority tasks, but below tasks at the same priority. The following figure illustrates the placement of tasks on the queue:



TASK1 is shown as a newly-ready task with a priority of 119. TASK1 will be placed in the CICS dispatching queue ahead of TASK8 but below TASK3.

Additionally, the dispatching priority can be increased based on the length of time a task has remained on the dispatching queue without being dispatched. The PRTYAGE parameter in the System Initialization Table (SIT) controls the frequency with which a task is examined to determine whether its priority should be increased.

The PRTYAGE specification is in milliseconds, and directs CICS to increase the priority of a task once the task has been on the dispatch queue for the PRTYAGE duration. The default value of the PRTYAGE parameter is 32768, indicating that a task's priority will be increased by 1 when the task has been on the dispatch queue for 32,768 milliseconds.

Task prioritization should be used sparingly, with task priority given to only the most important CICS tasks. The *CICS/ESA Version 4.1 Performance Guide* (Section 4.7.6 - Task Prioritization) explains the effects, uses, limitations, and implementation of task prioritization.

- **Remove selected transactions from the CICS region**. CICS task prioritization is not interrupt driven, as is MVS dispatching. The CICS task prioritization scheme simply relates to the relative position of tasks on the CICS dispatching queue. Once CICS has selected a task for dispatching, the task will remain dispatched until the task returns control to CICS.

  The Workload Manager allocates resources to address spaces (e.g., CICS regions), not to transaction service classes. The CICS region could be providing good service to other, less important transactions in different service classes. These service classes could be using significant system resources and delaying CICS in its dispatching the important transactions.

  If you have relatively long-running tasks serving transactions with a relatively low importance, these tasks may retain control of CICS for prolonged intervals. The result may be that the transactions with a high importance are delayed waiting for CICS to select their corresponding tasks for dispatch. One further result may be that the Workload Manager may allocate more resources to the service class representing the CICS region. Unfortunately, the additional resources may not help improve performance of the important tasks since CICS internally controls dispatching of tasks and these tasks may not release control.

  The only solution to this problem may be to identify the long-running transactions and remove them from the CICS region altogether. In general, this would be the preferred approach (that is, it normally is preferable to have a CICS region serving only your most important transactions.) This approach may require that another CICS region be generated, however.

- **Identify "long" transactions and optimize their related tasks**. This approach may result in large benefits, but generally requires a significant amount of application programmer time.

- **Speed the flow of all transactions through the CICS region**. The CICS region operates within the standard MVS environment. The CICS region may be delayed for various reasons (CPU dispatching, I/O access, etc.). CPExpert will analyze the "server" service class to determine whether the server (i.e., the CICS region) was using the CPU, whether the server was denied access to the CPU, etc. As a result of CPExpert's analysis, other rules may be produced to provide more information.