# Resource Groups

# and how they work

Laura Blodgett
Dieter Wellerdiek

24.02.2004

# Resource Group

The general idea behind a Resource Groups is: To have a vehicle, to give a minimum amount of service or to limit the service consumption of a Service Class. The forecast for the next ten seconds is based on the behavior of the system workload of the last minute.
Such a forecast is only accurate, if the system workload behaves over the next ten seconds in the same way, as it did in the past. This is normally not given, so you will see that the current service consumption is often above or below the defined boundaries.

The rest of the document will explain the Resource Group processing in more detail.

## *Overview*

A resource group is an amount of processor capacity across the Sysplex. It is optional. Unless you have some special need to limit or protect processor capacity for a group of work, you should skip defining resource groups and let workload management manage all of the processor resource to meet performance goals. You use a resource group to:

- Limit the amount of processing capacity available to one or more service classes.
- Set a minimum processing capacity for one or more service classes in the event that the work is not achieving its goals.

You can specify a minimum and maximum amount of capacity in unweighted CPU service units to a resource group. The minimum and maximum capacity applies to all systems in the Sysplex. You can assign only one resource group to a service class. You can assign multiple service classes to the same resource group. You can define up to 32 resource groups per service definition.

Keep in mind your service goals when you assign a service class to a resource group. Given the class combination of the goals, the importance level, and the resource capacity, some goals may not be achievable when capacity is restricted.

If work in a resource group is consuming resources above the specified maximum capacity, the system throttles the associated work to slow down the rate of resource consumption. The system may use several mechanisms to slow down the rate of resource consumption, including swapping the address spaces, changing its dispatching priority, and capping the amount of service that can be consumed. Reporting information reflects that the service class may not be achieving its goals because of the resource group capping.

By setting a minimum processing capacity, you create an overriding mechanism to circumvent the normal rules of importance. If the work in a resource group is not meeting its goals, then workload management will attempt to provide the defined minimum amount of CPU resource to that resource group.

**Note:** For more information on unweighted CPU service units, see the “System Resources Manager” chapter of *z/OS MVS Initialization and Tuning Guide*.

## Resource Group Parameter in the WLM Policy

**Name**

Eight characters that identify the name of the resource group. Each resource group must be unique within a service definition.

**Description**

Up to 32 characters that describe the resource group.

**Capacity**

Identifies the amount of available capacity you want workload management to allocate to the resource group. Capacity is in unweighted CPU service units per second, and it includes cycles in both TCB and SRB mode. The table in Appendix B, "CPU Capacity Table", in *z/OS MVS Planning: Workload Manager* shows the service units per second by CPU model. For a given resource group, you can vary the capacity minimum and maximum by service policy.

**Maximum**

CPU service units per second this resource group may use. *Maximum* specified for this resource group applies to all service classes in that resource group combined. *Maximum* is enforced. There is no default maximum value.

**Minimum**

CPU service units per second that should be available for this resource group when work in the group is missing its goals. The default is 0. If a resource group is not meeting its minimum capacity and work in that resource group is missing its goal, workload management will attempt to give CPU resource to that work, even if the action causes more important work (outside the resource group) to miss its goal. If there is discretionary work in a resource group that is not meeting its minimum capacity, WLM will attempt to give the discretionary work more CPU resource if that action does not cause other work to miss its goal. The minimum capacity setting has no effect when work in a resource group is meeting its goals.

## Sample

| Processor Model | CP's | SU/SEC |
|---|---|---|
| 2064-2C1 | 1 | 14692.3783 |
| 2064-2C2 | 2 | 13961.6056 |
| 2064-2C3 | 3 | 13377.9264 |
| 2064-2C4 | 4 | 13082.5838 |
| 2064-2C5 | 5 | 12638.2306 |
| 2064-2C6 | 6 | 12345.6790 |

For example, say an installation wants to know the maximum value to specify to cap a set of service class periods to 20% of the total Sysplex CPU capacity. To determine this number, use the sum of the service rates for each system (that is, the number of raw service units per second of TCB/SRB) and multiply it by 0.20. This service rate varies with the CPU speed, the number of CPUs in the CEC, and the number of logical CPUs in the LP. For example, the 2064-2C2 has 13961.60 service units per second of CPU and 2C4 has 13082.58 service units per second of CPU. The second is smaller due to the MP factor. However, in a 2C4 logical partition with two logical CPUs, the number is 13961.60 (like a 2064-2C2) service units per second of CPU. If the associated Service Class has a discretionary service goal, WLM achieves the minimum as long as service goals running in any other defined Service Class are not impacted. If other service goals are impacted, then WLM does not maintain the minimum.

**Note 1**: If each system runs with 4 logically CPU's and the split is done by weights, you have to use the 2064-2C4 as base. You have also to remember, the service distribution is not always done based on the weights.

**Note 2:** Resource Groups works on a Sysplex base and not on a single system base. Let's say we have a Sysplex with two systems and the Resource Group Maximum is set to 3000 SU's. In such an environment it can happen, that one system consumes around 2000 SU's and the other 1000 SU's. In total the consumption is around 3000 SU's as specified.

## *Resource Groups Maximum Management*

The purpose of the capping function is to control the amount of CPU service rate that dispatchable units in a set of address spaces (or enclaves) in a resource group consume. A resource group is a WLM construct used by the installation to limit a maximum (capping) or to deliver a minimum of CPU capacity to the address spaces and enclaves belonging to the service class periods that constitute the resource group. These minimum and maximum capacities are measured in the unweighted CPU service rate consumed in the resource group across the Sysplex. A resource group can be formed by distinct service classes with different importance values.

Policy adjustment code forces this objective by measuring the CPU service rate consumed by the group (locally and in the Sysplex). The CPU service rate values are accumulated first on local systems and then across the Sysplex for a total. The total value of the consumed service rate is used to determine if it exceeds the resource group maximum service rate. This comparison then determines how much to throttle address spaces in the group. This throttling is done by limiting the dispatchability of the address spaces or enclaves in the resource group.

## Capping

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 01 | 09 | 17 | 25 | 33 | 41 | 49 | 57 |
| 02 | 10 | 18 | 26 | 34 | 42 | 50 | 58 |
| 03 | 11 | 19 | 27 | 35 | 43 | 51 | 59 |
| 04 | 12 | 20 | 28 | 36 | 44 | 52 | 60 |
| 05 | 13 | 21 | 29 | 37 | 45 | 53 | 61 |
| 06 | 14 | 22 | 30 | 38 | 46 | 54 | 62 |
| 07 | 15 | 23 | 31 | 39 | 47 | 55 | 63 |
| 08 | 16 | 24 | 32 | 40 | 48 | 56 | 64 |

The table shows a capping table, where we have 16 awake slices and 48 capping slices. The number in the table shows the order, how WLM processes the capped and awake slices.

The granularity of the control is at the resource group level and not at the service class level. This means that all address spaces or enclaves in all service classes in a resource group are controlled by the same number of cap slices. Meanwhile, the dispatching priority assigned to each service class period is still being based on the goals. Therefore, work in the resource group with a more stringent goal will be more likely to run when the group is not capped.

The CPU service rate (in order to see if the limit is being exceeded) is per resource group. In a resource group formed by several service classes, WLM enforces that the total CPU service rate is not going to be above the maximum.

## Cap Time Slices

To implement capping, the elapsed time is divided into 64 time slices. Each time slice then represents 1/64th of the total elapse time. On a 2064-2C4 a capping slice has up to 817.66 SU (13082.58 * 4 CP's / 64 Slices) per second and up to this granularity WLM manages the workload.

Dispatchable units from address spaces or enclaves belonging to a resource group are made non-dispatchable during some time slices (red slices in the table above) in order to reduce access to the CPU to enforce the resource group maximum. The time slice where address spaces or enclaves in a group are set non-dispatchable is called a cap slice. The time slice where they are set dispatchable (green slices in the table above) is called an awake slice.

Because two groups may accumulate service units at a different rate, each time slice for a group is set in proportion with a previously measured average CPU service rate that the resource group collects. Therefore, by knowing how much above the maximum a resource group is, it is simple to derive the number of cap slices that the address spaces in the group are going to get during every elapse of the 64 time slices.

Because the cap slices for different groups are evenly spread across the time slices and the time slices are of relatively short duration, the changes in dispatchability do not appear as abrupt service changes.

All address spaces or enclaves in a resource group on each system are made dispatchable for the same proportion of time and during the same intervals. Address spaces in the same group but on different systems may be made dispatchable for different proportions of time and during different intervals. The proportion of time that address spaces are made dispatchable on different systems is based on the importance and quantity of work in the group running on each system. Work on the system that has more of the group's higher importance work may be made dispatchable for a larger proportion of time than lower importance work in the same group on another system. The intent is to equalize performance indexes for service class periods of the same importance within the resource group constraints across the Sysplex.

Capping delay is treated as another form of processor delay when managing dispatching priorities to meet goals. Consequently service class periods within a capped resource group may have their dispatch priorities increased to address capping delay as well as processor delay.

Every 10 seconds (the policy adjustment interval time) all resource groups are reappraised to determine if further adjustment is necessary. If so, the times that groups are to be set dispatchable or non-dispatchable are reevaluated. The 64 time slices and the cap slices are then reassigned.

**Attention:** Keep in mind your service goals when you assign a service class to a resource group. Given the combination of the goals, the importance level, and the resource group capacity, some goals may not be achievable when capacity is restricted. The RMF Workload Activity Report shows you the service class delay because of resource capping.

## Capping Delays seen in RMF

Capping delay is one of the delay states, recorded by WLM during sampling. These capping delays are reported with the other delay states in the RMF Workload Activity report. WLM samples in fixed intervals how the workload behaves and if there are any delays.

A Capping Delay gets reported, when the WLM sampling code finds a non-dispatchable unit of work in the system. This capping delay combined with the other delay types are used to calculate the execution velocity of the Service Class. All this sampled information is passed unfiltered to RMF, which displays this sampling data unfiltered. Refer to: *zOS Resource Measurement Facility Report Analysis*, for more information on this report.

Note: It may happen that you see Capping Delays, even if the service consumption is below the value on the Maximum parameter of the Resource Group. This is correct!
(See the sample below: *Service consumption is much below the Maximum*)

# *Resource Groups in Action*

When an installation uses Resource Groups, often results occur which are hard to understand. This section will discuss some of the behaviors you see after the implementation of resource groups.
All this examples are theoretical results. On a real system you may see different results, because there you see a mix of the examples described below and others which are not explained.

## Service consumption is much below the Maximum

Environment:

- Single system (2064-2C4) offers 52330.32 SU (13082.58 * 4 CP's) per second
- Constant system workload at 2330.32SU per second

- Maximum set at 3000 SU per second
- One service class consuming fixed amount of service units. The workload runs already for some minutes.

One awake slice has than 781.25 SU ( (52330.32 SU – 2330.32 SU) / 64 ) per second.

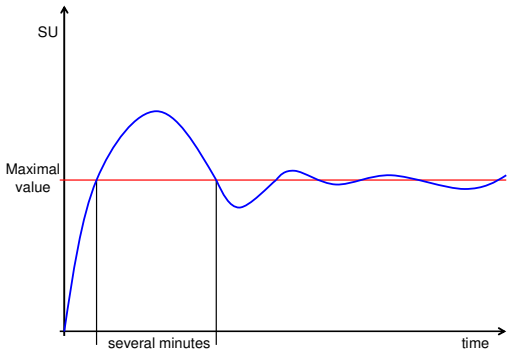
| Interval | SU consumed | awake slices |
|---|---|---|
| next | | ???? |
| -1 | 2343.75 | 3 |
| -2 | 2343.75 | 3 |
| -3 | 2343.75 | 3 |
| -4 | 2343.75 | 3 |
| -5 | 2343.75 | 3 |
| -6 | 2343.75 | 3 |

WLM will continue to use three awake slices during the next ten seconds. A fourth slice would assign to much service to the capped Service Class (4 slices * 781.25 SU = 3125).

## Service ramps up

Environment:

- Single system (2064-2C4) offers 52330,32 SU (13082.58 * 4 CP's) per second
- Some system workload
- Maximum set at 3000 SU per second
- No work was running during the last minutes




Now work comes in, which consumes up to 10000 SU per second. In the beginning it starts with lower service consumptions, because some IO, paging and necessary setups need to be done before in can consume the maximum amount of service units.

When the workload starts, no history data is available, so the workload runs ten seconds without any capping. In the second interval we have history data, but the workload still rams up his service consumption, so we still not cap at the correct point.

The analyzed SMF 99 records from some customers show a behavior as shown in the diagram. It can take up to several minutes before WLM caps with the right amount of cap slices. During the swing-in, we see service consumptions which may twice as high as specified on the Maximum value.

## Two Service Classes capped

Environment:

- Single system (2064-2C4) offers 52330,32 SU (13082.58 * 4 CP's) per second
- Constant system workload at 2330.32 per second
- Service Class 1 (Importance 3) with a Maximum at 25000 defined in Resource Group 1
- Service Class 2 (Importance 3) with a Maximum at 25000 defined in Resource Group 2
- Over the last minutes the service consumption was fixed on 25000SU

| Interval | Service Class 1 | | Service Class 2 | |
|---|---|---|---|---|
| | SU consumed | awake slices | SU consumed | awake slices |
| -1 | 30000.00 | 32 | 20000.00 | 32 |
| -2 | 25000.00 | 32 | 25000.00 | 32 |
| -3 | 25000.00 | 32 | 25000.00 | 32 |
| -4 | 25000.00 | 32 | 25000.00 | 32 |
| -5 | 25000.00 | 32 | 25000.00 | 32 |
| -6 | 25000.00 | 32 | 25000.00 | 32 |

Now additional workload starts to run in Service Class 1 for 10 second consuming additional service (5000 SU). This new workload gets service, because he runs on the same important than the other workload currently running in the system. There are no capping adjustments, readjustments only done all 10 seconds.

When it comes to calculate the awake slice count for the next ten seconds, the additional service consumptions is taken into account.

| Interval | Service Class 1 | | Service Class 2 | |
|---|---|---|---|---|
| | SU consumed | awake slices | SU consumed | awake slices |
| next | | 30 | | 33 |
| -1 | 30000.00 | 32 | 20000.00 | 32 |
| -2 | 25000.00 | 32 | 25000.00 | 32 |
| -3 | 25000.00 | 32 | 25000.00 | 32 |
| -4 | 25000.00 | 32 | 25000.00 | 32 |
| -5 | 25000.00 | 32 | 25000.00 | 32 |
| -6 | 25000.00 | 32 | 25000.00 | 32 |

For Service Class 1 the awake slice count decreases and for Service Class 2 the count increases based on the past.

Our assumption was, the additional workload is running only for ten seconds. So when the new capping pattern becomes active the workload is already gone.

As a result Service Class 1 consumes less service as expected and Service Class 2 runs over the defined Maximum value.

| Interval | Service Class 1 | | Service Class 2 | |
|---|---|---|---|---|
| | SU consumed | awake slices | SU consumed | awake slices |
| next | 23437,50 | 30 | 25781,25 | 33 |
| -1 | 30000.00 | 32 | 20000.00 | 32 |
| -2 | 25000.00 | 32 | 25000.00 | 32 |
| -3 | 25000.00 | 32 | 25000.00 | 32 |
| -4 | 25000.00 | 32 | 25000.00 | 32 |
| -5 | 25000.00 | 32 | 25000.00 | 32 |
| -6 | 25000.00 | 32 | 25000.00 | 32 |

## Summary

The three samples above explain only some of the behavior you see with Resource Groups. There are many other scenarios which even harder to understand. Here are only some samples:

- Many Service Classes capped with Resource Groups (One Service Class / one Resource Group)
- Many Service Classes capped with one Resource Group (Many Service Classes / one Resource Group)

- Many Service Classes capped with Resource Groups (One Service Class / one Resource Group) and the Service classes have different Importance
- Many Service Classes capped with one Resource Group (Many Service Classes / one Resource Group) and the Service classes have different Importance
- Workload running distributed in a Sysplex
- Workload running in a LPAR with shares CP's