This page intentionally left blank
Imagine you’re zManager and you want to manage workloads that jump across platforms. What information would you like access to?

- Insight into the hypervisors to see state of the physical resource utilization across virtual servers
- Some interface into the virtual servers operating systems to be able to see and manage them
- A way to identify and group work into a collection meaningful to the human administrators
- A way to quantify the relative priorities between work that has been grouped into workloads
- A way to adjust resources to meet defined goals about relative priorities
Insight into Hypervisor Activity

zEnterprise manages the hypervisors so it can have microcode probes to “see” what’s going on in the various hypervisors in the system:

This is why zEnterprise imposes some requirements on System z levels and why zManager controls the loading of hypervisors ... so it can “see” into the state of the virtualization taking place across the defined virtual servers.
Insight into Virtual Servers

This requires a probe into the virtual server operating system. That is called “GPMP” – Guest Platform Management Provider:

GPMP is a lightweight management probe into the OS of the virtual server. It’s not strictly required, but it does provide some benefits as we’ll see. You’ll enabled GPMP on your AIX virtual servers in the upcoming labs. We enabled GPMP on z/OS before the workshop.
Identifying, Classifying and Managing Work

Different work often has different performance priorities. To differentiate work it’s important to first identify it ... then you can assign performance goals.

A potentially slippery topic if you’re not familiar with concept. At a high level here is what we will cover:

1. How it’s defined in zManager
2. How it works on non-z/OS virtual servers
3. How it works with zWLM
# Reporting on Relative Goal Attainment

One element of this is simply *reporting* on how a group of work is performing against the goals you’ve defined:

<table>
<thead>
<tr>
<th>Work</th>
<th>Priority</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABC</td>
<td>Highest</td>
<td>Not Meeting Goals</td>
</tr>
<tr>
<td>DEF</td>
<td>Medium</td>
<td>Meeting Goals</td>
</tr>
<tr>
<td>XYZ</td>
<td>Low</td>
<td>Exceeding Goals</td>
</tr>
</tbody>
</table>

Knowing this brings you one step closer to having better control over the environment.

With this information in hand it would allow you to at a minimum take a look at resource allocations (CPU, memory) and make manual adjustments.

Reporting on performance against defined goals is the foundation of zManager’s Platform Performance Management functionality.

Many more details to come ...
Making Adjustments to Meet Performance Goals

This brings us to the heart of the matter – zManager’s ability to detect when goals are not met and making adjustments to attempt to meet goals.

I am aware of the environment, and I am aware of your defined workloads and goals.

I will adjust to try to meet your goals.

zManager can shift CPU between virtual servers* within a hypervisor to meet the workload goals defined.

We explore how some of this accomplished as we proceed through this unit.

This picture is obviously simplified ... there’s a great deal of sophisticated monitoring, data collection and analysis that’s going on behind the scenes.

* PowerVM and z/VM, but not PR/SM or x Hyp
Tracking Overall Response Time

This brings another piece to the table – ARM. ARM provides a way to correlate response times across different platforms:

This provides a better view of overall response time as well as the workload topology. We’ll cover:

- What “ARM” is
- What middleware supports it
- How it’s enabled
- How it works with WLM on z/OS
Role of the “Automate” Suite

Brief reminder of the two suites and where they fit in this discussion:

“Manage” Suite

- Provides the majority* of the function represented by this graphic pie chart
- With Manage you can monitor resource utilization at the hypervisor and virtual server level, but you can’t automate it with goal-oriented management.

“Automate” or “Advanced Management”

- Optional suite of function that provides the goal-oriented performance management we’ll cover in this unit
- Advanced Management covers System x blades, Automate all else.
- No charge feature code (FC0020) for z/OS, but carries a charge per connection for power blades (FC0045) and IFLs (FC0052)

Going forward in this unit we assume both Manage and Automate is present on the zEnterprise system

* There is a piece of the “Energy” function that requires “Automate” – Static Power Savings
GPMP
Understanding the role of the Guest Platform Management Provider
GPMP and What it Provides

GPMP provides zManager a better view up into the virtual server:

If you use ARM, GPMP is what feeds the ARM information to zManager so it can report on response times and draw the workload topology diagram.

We'll see examples of those reports later in this unit.

If you wish to define workloads and classify work, GPMP provides additional classification criteria beyond simply virtual server name.

More on workloads and classification coming up.

If you wish to associate zManager service class names with z/OS WLM service classes, you require ARM and GPMP on the virtual server and z/OS.

More on how zManager intersects with zWLM later in this unit.

So ... not strictly required, but it does provide benefits at a relatively low cost in processing overhead.
GPMP and Eligible Virtual Server Types

GPMP is software that is enabled in the operating system of the virtual server managed by zManager. Eligible virtual server operating systems include:

- Red Hat Enterprise Linux (RHEL) 6, 5.5, 5.4, or 5.3
- Novell SUSE Linux Enterprise Server (SLES) 11 or 10
- z/OS V1R12, V1R11, or V1R10 with PTFs
- z/VM is itself a hypervisor as well as an operating system. It does not itself run GPMP. However it does participate to a degree in zManager functions. z/VM 6.1 with PTFs is required

These details pulled from zEnterprise System Ensemble Performance Management Guide, GC27-2607-03

In this workshop we focus on z/OS and AIX

We’ll show you how it is enabled on z/OS and AIX

* All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represents goals and objectives only.
GPMP Enablement on AIX

Some of this you did in the first lab, some you’ll do in the upcoming lab ...

Enable GPMP in Virtual Server

You did this in the first lab

Enable IPv6 in Virtual Server

GPMP uses INMN to feed zManager. INMN is IPv6.

Then in AIX:

- Mount virtual media to virtual server
- SMITTY to enable eWLM services
- Use `rpm` to install AIX GPMP package
- Start GPMP and set to autostart next boot

You’ll do this part in the upcoming lab

In summary a UNIX install of a small Java software package that will act as a management interface for zManager into the virtual server operating system
Which AIX Virtual Servers?

It is not necessary to GPMP-enable every virtual server. In the lab we have you enable two of the four AIX virtual servers:

GPMP is required required to feed ARM* information, which is required to associate the distributed priorities with zWLM goals. But it is not strictly required for some zManager functions.

zManager's ability to move CPU between the “donor” and “receiver” virtual servers is based on the “Processor Management” option for the virtual server.

We want GPMP on the WAS and IHS servers because we will be defining workloads that include those virtual servers. More on workloads coming up.

* Application Response Measurement. More on this coming up.
GPMP on z/OS -- Overview

Gives zManager insight into the z/OS operating system as well as enabling collaboration between zManager workload service classes and zWLM.

You do some setup work to enable GPMP to operate on z/OS LPAR:

- Some RACF work, creates some logging directories.
- z/OS R12 and above provides some additional fields in WLM panels for GPMP.
- If before z/OS 1.12 then no automatic start of GPMP on z/OS ...

F WLM,GPMP,START

WLM starts GPMP using supplied JCL proc:

If before z/OS 1.12 then F WLM,GPMP,START

GPMP runs as an address space.

Validate with WLM display: D WLM,SYSTEMS,GPMP

Should see: STATE OF GUEST PLATFORM MANAGEMENT PROVIDER (GPMP): CONNECTED
Understanding ARM and its role in zManager operations
Very High Level Overview of ARM

ARM stands for “Application Response Measurement” and it is a mechanism for tracking response time across multiple servers in a distributed workload.


Provides a way to track and correlate time spent in each “hop” of a distributed workload that spans servers

Requires application or middleware to actively exploit the ARM interface

Information in correlators may be used by tools (zManager) to report on application response time as well as paint a picture of the workflow topology
Is ARM Required?
Strictly speaking, no ... but it provides zManager with information about overall transaction performance, and it allows association to zWLM service class.

### Virtual Server Topology Report
Shows virtual servers participating in an end-to-end ARM-enabled transaction.

### Hops Report
Shows the hop information along with successful transactions and average response time at each hop.

### zWLM Service Class
Allows you to associate a zManager workload service class to a defined zWLM service class so coordinated performance management is possible. More on this coming up.

<table>
<thead>
<tr>
<th>Name</th>
<th>Hop Num</th>
<th>Group Name</th>
<th>Successful Transactions</th>
<th>Failed Transactions</th>
<th>Stopped Transactions</th>
<th>Infleft Transactions</th>
<th>Queue Time (s)</th>
<th>Execution Time (s)</th>
<th>Successful Average Response Time (s)</th>
<th>Infleft Average Response Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hop 0</td>
<td>0</td>
<td></td>
<td>22,281</td>
<td>0</td>
<td>0</td>
<td>202</td>
<td>0.000000</td>
<td>0.000273</td>
<td>0.165120</td>
<td>0.094535</td>
</tr>
<tr>
<td>IBM Webservice Plugin</td>
<td>0</td>
<td>IBM_HTTP_Server</td>
<td>22,281</td>
<td>0</td>
<td>0</td>
<td>202</td>
<td>0.000000</td>
<td>0.000273</td>
<td>0.165120</td>
<td>0.094535</td>
</tr>
<tr>
<td>rjaihs1</td>
<td>0</td>
<td></td>
<td>22,281</td>
<td>0</td>
<td>0</td>
<td>202</td>
<td>0.000000</td>
<td>0.000273</td>
<td>0.165120</td>
<td>0.094535</td>
</tr>
<tr>
<td>Hop 1</td>
<td>1</td>
<td></td>
<td>37,486</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>0.000000</td>
<td>0.001384</td>
<td>0.019575</td>
<td>0.015683</td>
</tr>
<tr>
<td>WebSphere APPLICATION</td>
<td>1</td>
<td>server1</td>
<td>37,486</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>0.000000</td>
<td>0.001384</td>
<td>0.019575</td>
<td>0.015683</td>
</tr>
<tr>
<td>rjawas1</td>
<td>1</td>
<td></td>
<td>37,486</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>0.000000</td>
<td>0.001384</td>
<td>0.019575</td>
<td>0.015683</td>
</tr>
<tr>
<td>Hop 2</td>
<td>2</td>
<td></td>
<td>4,363,389</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0.000000</td>
<td>0.000265</td>
<td>0.000265</td>
<td>0.000673</td>
</tr>
<tr>
<td>DDF</td>
<td>2</td>
<td>DSN9WSC</td>
<td>4,363,389</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0.000000</td>
<td>0.000265</td>
<td>0.000265</td>
<td>0.000673</td>
</tr>
<tr>
<td>TOSP11</td>
<td>2</td>
<td></td>
<td>4,363,389</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0.000000</td>
<td>0.000265</td>
<td>0.000265</td>
<td>0.000673</td>
</tr>
</tbody>
</table>
IBM ARM-Enabled Middleware

Here’s a quick summary of ARM-enabled middleware:

IBM WebSphere Application Server
All platforms ... V6, V7 and V8

HTTP Server via WAS Plugin Function
IHS, IIS, Domino, iPlanet

In the upcoming lab you’ll turn on ARM for IHS and WAS:
- IHS – modify XML file
- WAS – radio button in Admin Console
Workloads and Classification

Helping zManager understand your priorities
The zManager “Workload”

A collection of virtual servers with a defined set of priorities. zManager attempts to adjust resources to meet defined priorities:

- **Workload: WORKAB**
  - Virtual Servers A and B
  - Service classes with classification rules

- **Workload: WORKCD**
  - Virtual Servers C and D
  - Service classes with classification rules

- **Workload: WORKEF**
  - Virtual Servers E and F
  - Service classes with classification rules

- **Workload: WORKG**
  - Virtual Servers G
  - Service classes with classification rules

**Preliminary points ... then we go on to more details:**

- The objective is to give zManager some idea of how you’ve organized work on the system
- Multiple workload definitions may exist ... and virtual server overlapping is permitted
- Each workload carries with it services classes which defines the priority of the workload
- Each service class carries with it a classification rule to associate work with a workload
Classification Rules and Service Classes

This is what associates work with a workload and provides zManager and understanding of what work running where has relative priority.

**Service Class**

**Name:** MyService1

**Performance Goal:**
- Velocity
- Fast

**Business Importance:** High

**Classification Rule**

If Virtual Server = A or B

- Workload: WORKAB
  - Virtual Servers A and B
  - Service classes with classification rules

- zManager weighs all this as its watching hypervisors and virtual server CPU utilization

- If A, B then
  - Velocity, Fast, Importance=High

- If B, C then
  - Velocity, Medium, Importance=Medium

- If E, F then
  - Velocity, Highest, Importance=Highest

- If G then
  - Discretionary
Pause ... Explain Two Key Concepts
We’re just showed terminology related to defining performance priorities. We need to explain those:

“Velocity” as a performance goal

A relative measure of performance for a virtual server. At a high level it’s a measure of how quickly work gets access to system resources and completes relative to other work on the system.

Options are: Fastest, Fast, Moderate, Slow, Slowest

Fastest = work should get access to system resources quickly so it can complete

Slowest = work may wait if other higher priority work is consuming the system resources

“Business Importance”

This provides a zManager with a sense for relative priorities when the performance goals themselves aren’t enough

This value is combined with performance goal as part of zManager’s broader algorithm for determining performance results

Options are: Highest, High, Medium, Low, Lowest
Service Class Goals and Business Importance

Performance Goals and Business Importance form a very granular grid of relative priority you define for any given service class:

<table>
<thead>
<tr>
<th>Velocity Goal</th>
<th>Fastest</th>
<th>Fast</th>
<th>Moderate</th>
<th>Slow</th>
<th>Slowest</th>
</tr>
</thead>
<tbody>
<tr>
<td>Highest</td>
<td>Service Class A</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>High</td>
<td>Service Class B</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Medium</td>
<td>Service Class C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low</td>
<td>Service Class E</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lowest</td>
<td>Service Class H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

A performance goal of “Discretionary” falls below even “slowest”

zManager watches all the work and weighs all these priorities as it works to manage the overall resources to meet your goals.
Performance Index (PI)

For every service class, zManager reports on how it sees actual performance against the defined goals for the service class:

\[ \text{PI} > 1 \quad \text{Service class not meeting velocity goal} \]
\[ \text{PI} = 1 \quad \text{Service class meeting velocity goal} \]
\[ \text{PI} < 1 \quad \text{Service class overachieving velocity goal} \]

Number computed relative to all other service classes active on the zEnterprise.

Example of Service Classes Performance Index Report:

- Starts out not meeting its performance goal
- Over time, zManager makes adjustments to help the service class meet its performance goal.
- In this example, zManager was able to help the service class achieve its performance goal.

You will see this in action in the upcoming lab.
Performance Policies

Provide a way to dynamically shift a group of service classes in and out of effect within a defined workload:

**Workload Definition**
Name: MyWork1

**Performance Policy**
Name: Weekday
Status: Active

Name: Weekend
Status: Inactive

From the HMC you may dynamically activate a defined performance policy.

Provides a way to change performance management profiles easily and dynamically.
Preliminary Summary of Workloads

Let’s step back and collect all the concepts into a single summary picture:

Virtual servers are added to the workload definition

The workload has a default performance policy with a default service class

Your performance policy with one or more service classes, each with a classification rule along with a default service class for unclassified work

Multiple performance policies permitted

Multiple workload definitions active concurrently
Overview of Classifying Work in z/OS WLM

Some details to follow …

zManager Workload

Performance Monitoring and Reporting
Including z/OS virtual servers

zManager can shift resources between virtual servers within the hypervisor
If Processor Management enabled and CPU shared. Based on zManager’s knowledge of all workloads and performance goals.

zManager will not shift resources within LPAR; that’s WLM’s role
But by associating that work with a WLM service class of your choice you can enable closer coordination of priorities between zManager and zWLM
**Associating zManager SC to WLM SC**

Requires GPMP+ARM on both sides as well as EWLM subsystem classification rule to connect zManager service class name to WLM service class name.

<table>
<thead>
<tr>
<th>Class</th>
<th>Action</th>
<th>Type</th>
<th>Description</th>
<th>Service</th>
<th>Report</th>
</tr>
</thead>
<tbody>
<tr>
<td>-----</td>
<td>Class</td>
<td>Type</td>
<td>Description</td>
<td>Service</td>
<td>Report</td>
</tr>
<tr>
<td>-----</td>
<td>Action</td>
<td>Name</td>
<td>Start</td>
<td>Service</td>
<td>Report</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Action</th>
<th>Type</th>
<th>Name</th>
<th>Start</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ESC</td>
<td>ZMGRT1SC 1</td>
<td></td>
<td>PPMGRT1</td>
</tr>
<tr>
<td>1</td>
<td>ESC</td>
<td>ZMGRT2SC 1</td>
<td></td>
<td>PPMGRT2</td>
</tr>
<tr>
<td>1</td>
<td>ESC</td>
<td>ZMGRT6SC 1</td>
<td></td>
<td>PPMGRT6</td>
</tr>
<tr>
<td>1</td>
<td>ESC</td>
<td>SrvClsSp 1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>ESC</td>
<td>rNameWit 9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>ESC</td>
<td>h24chars 17</td>
<td></td>
<td>PPMHGHST</td>
</tr>
</tbody>
</table>

- **Normal “DDF” used for DB2 if no ARM token**
- **The “EWLM” subsystem type for classification rules is used when request carries the ARM token into z/OS**
- **EWLM rules for each team’s zManager service class with mapping to WLM service class**
- **zManager allows SC names up to 64 characters. This shows how you accommodate up to 24 characters on panel**
RMF Monitor III and SDSF Enclave Report

Some examples to show you that once classified the work shows up like any other classified work:

Showing service class PPMHGHST activity based on propagation of zManager service class into zWLM

SDSF enclave report ... in the upcoming lab you’ll be able to see the effect of your team’s zManager service class propagation into z/OS and the mapping to your team’s zWLM service class

SrvClass of “PPMHGHST” here, but in lab you’ll see PPMGRT# where # is your team number.
zManager Reports
A survey of some key performance-related reports provided by zManager
“Workloads” Under “Ensemble Management”

Provides a handy entry-point to the workload-related reports:

- The name of the workload
- Number of virtual servers included in the workload
- Currently selected performance policy and its status
- The Business Importance assigned to the performance policy

Provides a high-level view of all the workloads present on the ensemble.

Select a workload and a little twisty appears which reveals additional options.

The reports found here are what we’ll focus on next.
A Roadmap to the Location of Reports

The following is a reference showing the access path you would use to get at the reports we’ll show you:

- Service Class Resource Adjustments Report
- Hops Report
- Virtual Server Topology Report
- Performance Index (PI) Report for Service Class
- Hypervisor Report
- Adjustments Report
- CPU Utilization for Hypervisor
- Adjustments Report for selected workload
- Virtual Servers Report for selected workload
- Service Classes Report for selected workload
- CPU Utilization Distribution for workload

Some reports are accessible through multiple paths

With time and a little practice the location and content of these reports becomes second nature
Note about Granularity and Retention of Data

There are limitations to the granularity and retention of historical data.

- **36 Hour Rolling Window of Retention**
  - **Past**
    - Previous Hours
    - zManager maintains 36 hours of recorded performance data
  - Most Recent 1 Hour
  - The most recent 1 hour of data is maintained with 1 minute interval granularity
  - The 35 hours prior to the most recent hour are maintained with 15 minute interval granularity
  - **Future**
Workloads Report

Provides a high-level view of all workloads and how they’re doing:

- **All the workloads**
- **Selected Workload**

**Showing the time interval for the information displayed. This is configurable**

**Actions against selected workload. These take you to the workload-specific reports**

**Works on selected Workload**

**Workload Charts**

Most reports have a twisty here that expands to show other reports in the lower half of the page display for *the workload selected*
Workloads Report – CPU Utilization

Below the twisty you’ll find two reports – CPU utilization and Performance Index

Chart indicates workload has three virtual servers:

- 1 at 0–10% utilization
- 1 at 10–20% utilization
- 1 at 30–40% utilization

A chart that looks better if the number of servers in each CPU utilization range wasn’t the same value of “1” 😊

A chart that becomes more meaningful the more virtual servers are included in the workload
Workloads Report – Performance Index

Here’s the Performance Index report for the selected workload:

The Y axis represents the PI value for the service classes.

The PI over time for the indicated service class in the workload.

The X axis represents time for the (configurable) interval displayed.

This workload had three service classes defined to it, but only one that had work classified to it during the interval. Hence only one service class appears on the chart.
Virtual Servers Report

This provides a snapshot of the virtual servers that are associated with a workload.

Access from “Select Action” pulldown from Workloads Report

Access from Virtual Servers Report from main context menu

Configure columns

Virtual servers that are part of the workload

CPU utilization chart for the selected virtual server is present under twisty

Virtual server CPU information as seen by the hypervisor

Information from the operating system in the virtual server is based on feed from GPMP agent if present
Workload Resource Adjustments Report

Shows successful and failed attempts to adjust CPU resources between virtual servers under the control of a hypervisor:

- Donor to Receiver CPU transfer by zManager

For this to work, “Processor Management” must be enabled for the hypervisor and the virtual servers, and virtual servers must be defined as “Shared” CPU (not dedicated).

You can see the incremental transfer of CPU from donor to receiver.

This is how zManager attempts to manage the PI to the goal.
Virtual Server Topology and Hops Reports

Relies on ARM+GPMP to understand and portray the topology of the flow within the workload across the associated virtual servers:

- Drill down on a select workload
- Virtual server names
- Number of requests
- Topology Report

This data can be exported to CSV format.
Summary and Lab Review
Summary

**GPMP and ARM**
- GPMP is a lightweight performance agent that runs in the OS
- ARM is a means of tracking end-to-end response time
- Neither is strictly required though certain things are lost if not there

**zManager Workloads**
- A means of logically associating virtual servers into a work group
- Service classes define the relative priority
- Classification rules assign work to the service class
- Performance policies contain the service classes
- Workloads contain the performance policies
- zManager then reports performance and manages to goals
- Propagation of zManager service class into z/OS WLM possible

**Reports**
- Accessible through HMC
- Reports we explored were related to the workload
- Information about performance attainment, virtual server CPU, etc.
Enable ARM, GPMP and other setup
Define the zManager workload
Start the work
Observe adjustments and PI