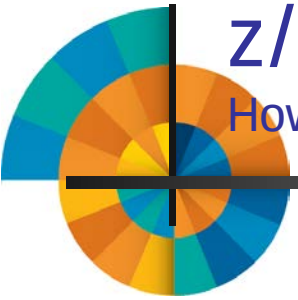
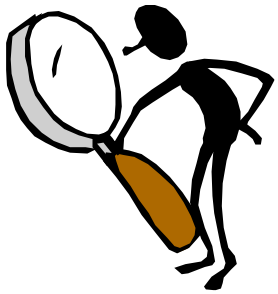


z/OS Workload Manager

How it works and how to use it

Robert Vaupel
STSM, z/OS Workload Management
IBM Development Laboratory Böblingen
Schönaicherstr. 220
Phone: +49(0)-7031-16-4239
vaupel@de.ibm.com

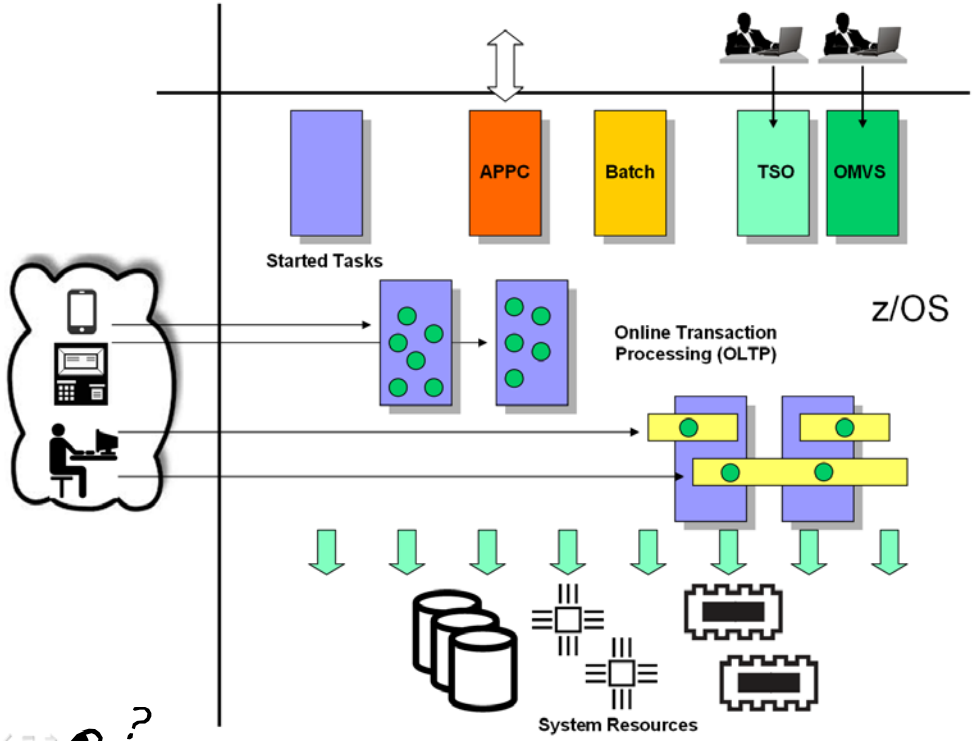
Content

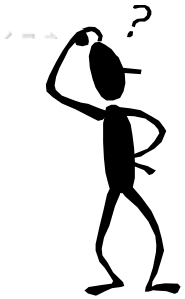


Part I: The Basics

What executes on z/OS?
How does the system deal with?
What are goals?
What needs to be done?
Importance and Goals
Service Definition

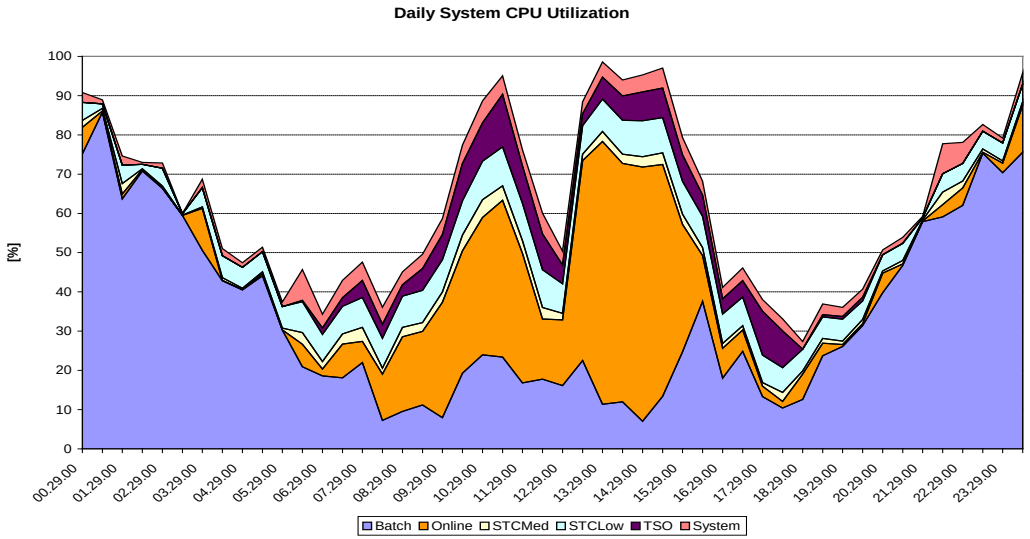
What executes on z/OS?





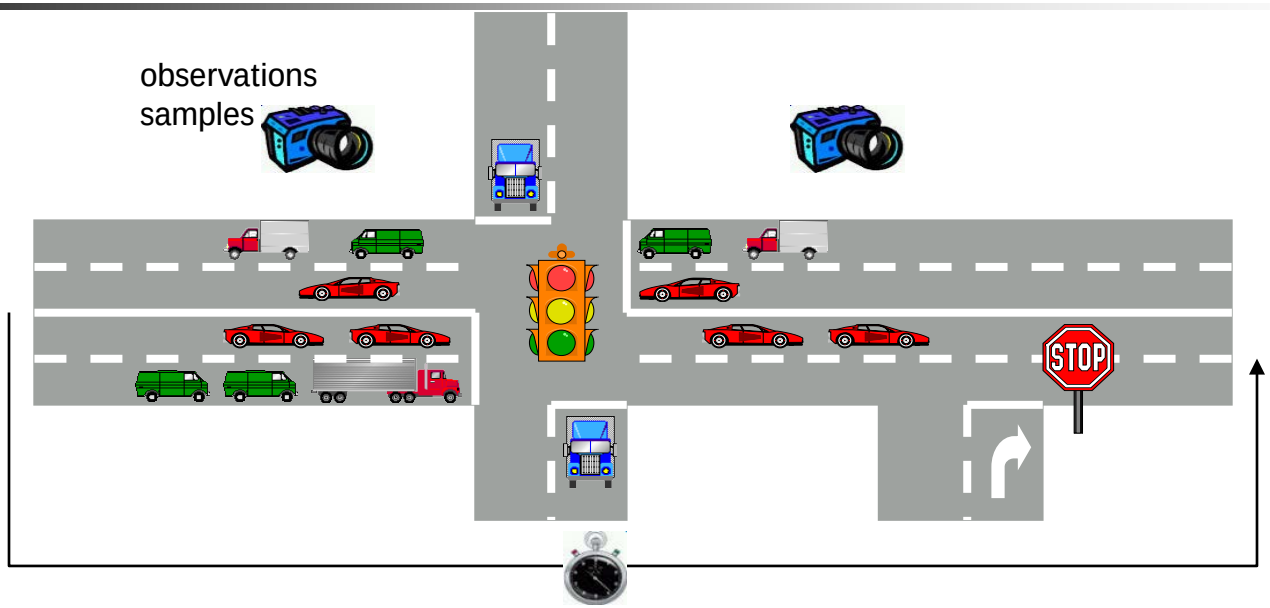
- Different types of work
  - System tasks
  - Background work (Batch)
  - Intersystem communication
  - End user (TSO/OMVS)
  - Online transaction processing
    - CICS, IMS, Websphere,...
  - Databases
- Work with different resource requirements
  - High, low CPU
  - Much, little storage
  - I/O sensitive
  - ...
- Work with different runtime requirements
  - Very short running with immediate response requirements
  - Long running but within required deadlines
  - And everything in between

How does z/OS deal with work?



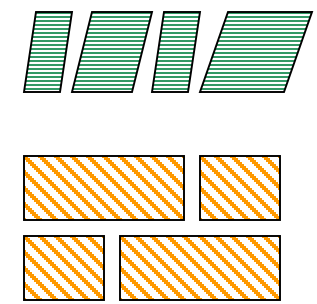
- Problem:
  - Workloads with different and changing resource requirements and different user expectations want to use the system at the same point in time
- Solution:
  - Dynamic Workload Management: Goal Mode
    - Adjusts system resources based on workload demand and user expectations (the goals)
    - This is an autonomic system function
      - It does not allow to define a completely static definition
      - But you don't have to fear it

What are the goals?



- Your expectations how work moves within the system
  - Either expressed as a time if this can be measured
    - Response Time
  - Or based on observations (samples) from resources the system can control
    - Execution Velocity: How much delay is acceptable for work in the system

What has to be done?



different Workloads

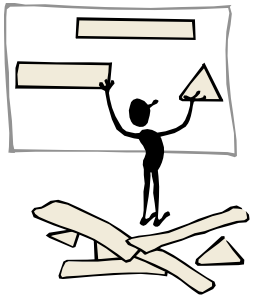
Classification
Distinguish work based on subsystem, performance expectations and resource requirements

Online
- Importance=1, Goal: 90% in 0.5s

TSO
- Period 1
  - Importance=3, Goal: 80% in 1s
  - Duration: 1000 SU
- Period 2
  - Importance=5, Goal: Exec. Vel.: 20

Batch
- Period 1
  - Importance=4, Goal: Exec. Vel.: 20
  - Duration: 5000 SU
- Period 2
  - Discretionary

- Work has to be classified!
- Service classes must be defined
  - With realistic goals!
    - What are realistic goals?
  - With a business importance!
    - Which determines how the system treats the work
  - Both business importance and goals should fit together!!



Business Importance

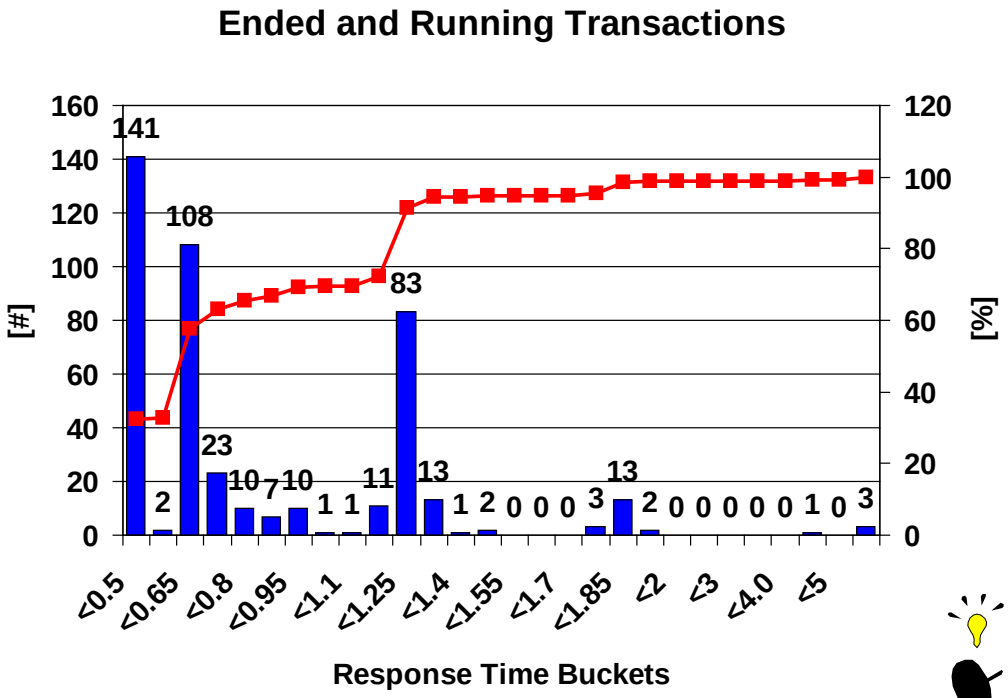
System Service Classes: SYSTEM and SYSSTC

Importance

1 2 3 4 5 Discretionary (SYSOTHER)

- The business importance determines who gets helped first if the system has to decide between service classes
- The goal determines the resource requirements of the work

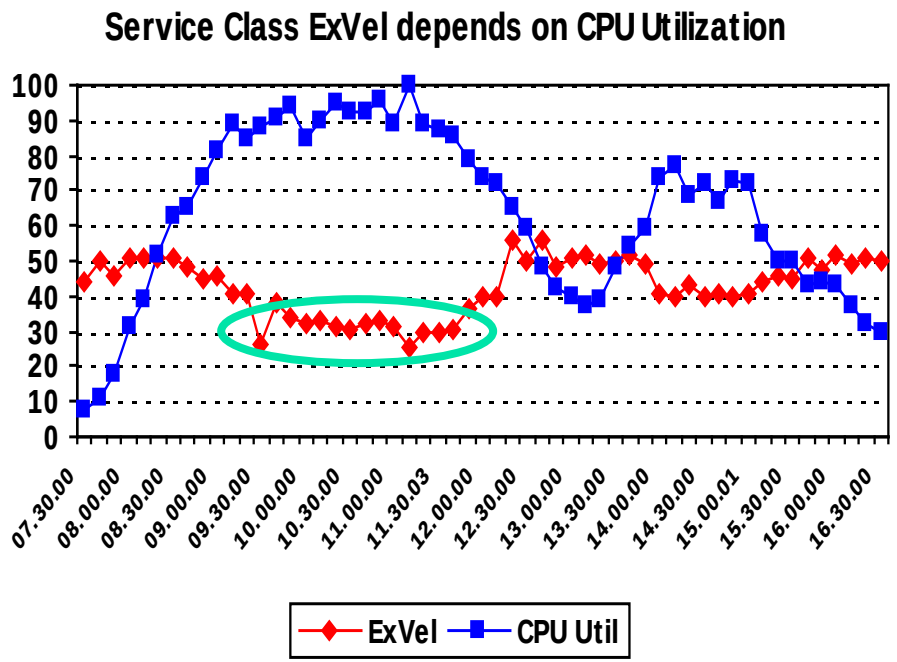
Response Time Goals



- Two possibilities
  - Average Response Time
  - Percentile Response Time
- Which is better?
  - Average Response Time
    - Assumes that work behaves uniform to some extent
  - Percentile Response Times
    - Can better deal with work which shows a higher fluctuation
- When can you use response time goals?
  - If there are enough completions
    - Rule of thumb: >9 in 15 min
  - If the ending times can be understood to some extent

Execution Velocity Goals

$$Execution\ Velocity\ =\ 100 \times \frac{Total\ Using}{Total\ Using\ +\ Total\ Delay}$$

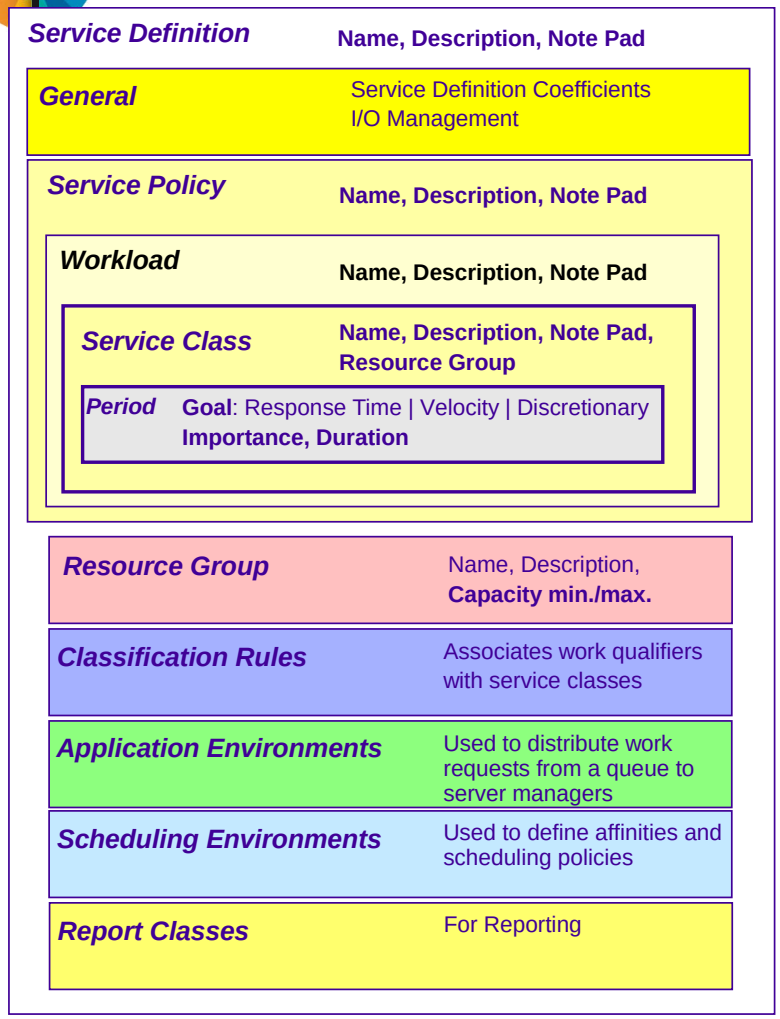


- Defines the acceptable delay of work
- Depends on
  - System Utilization
    - Use periods of high utilization
  - Whether I/O priority management has been turned on
    - Also whether Batch Management is used
  - The number of parallel executable work units of a service class
  - The number of logical processors in the system
- Needs some attention and periodic refinement

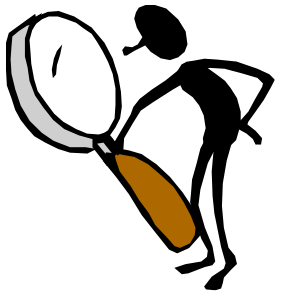
Discretionary Work

- All kind of work without a specific goal!
- Fills system when "enough" resources are available
  - Enough means
    - That these resources are not required to meet the requirements of non-discretionary service classes
    - That resources are taken from service classes with low goals if they meet their goals
      - Meet goals means: The PI is less than 0.7
      - Low means
        - Execution Velocity less or equal to 30
        - Response Time Goal longer than 1 minute
        - Any importance level
  - What to do if you don't like discretionary goal management (capping of work)
    - Define a resource group without minimum and maximum
    - Put service class with a goal eligible for capping into this resource group
    - Capping will not take place !!

Service Definition



- General
  - Classification Rules and Service Goals are defined in the Service Definition
  - Service Definition applies for the whole sysplex
- Constructs
  - 1 or multiple service policies
    - additional service policies allow to modify goals for existing service classes
  - Workloads: grouping mechanism
  - Service Class (Period): group with management scope
  - Resource Group: define cpu resource consumption minima and maxima
  - Classification Rules: basically IEAICS in a new notation and extended
  - Application Environments: management construct for queue and server environments
  - Scheduling Environments: management construct for recording resource states
  - Report Classes: provides granularity for reporting



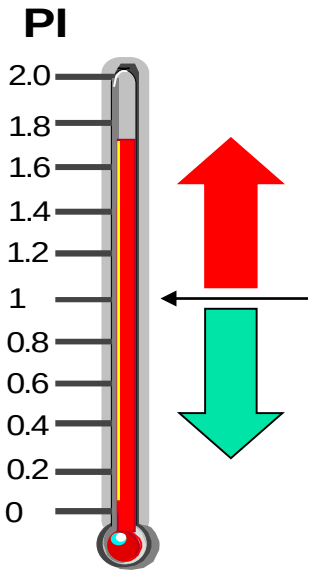
Part II:

How does WLM Work?

Resource and Goal Adjustment
Performance Index
How does WLM work?
Example: CPU Management
Overview WLM Algorithms

Performance Index (PI)

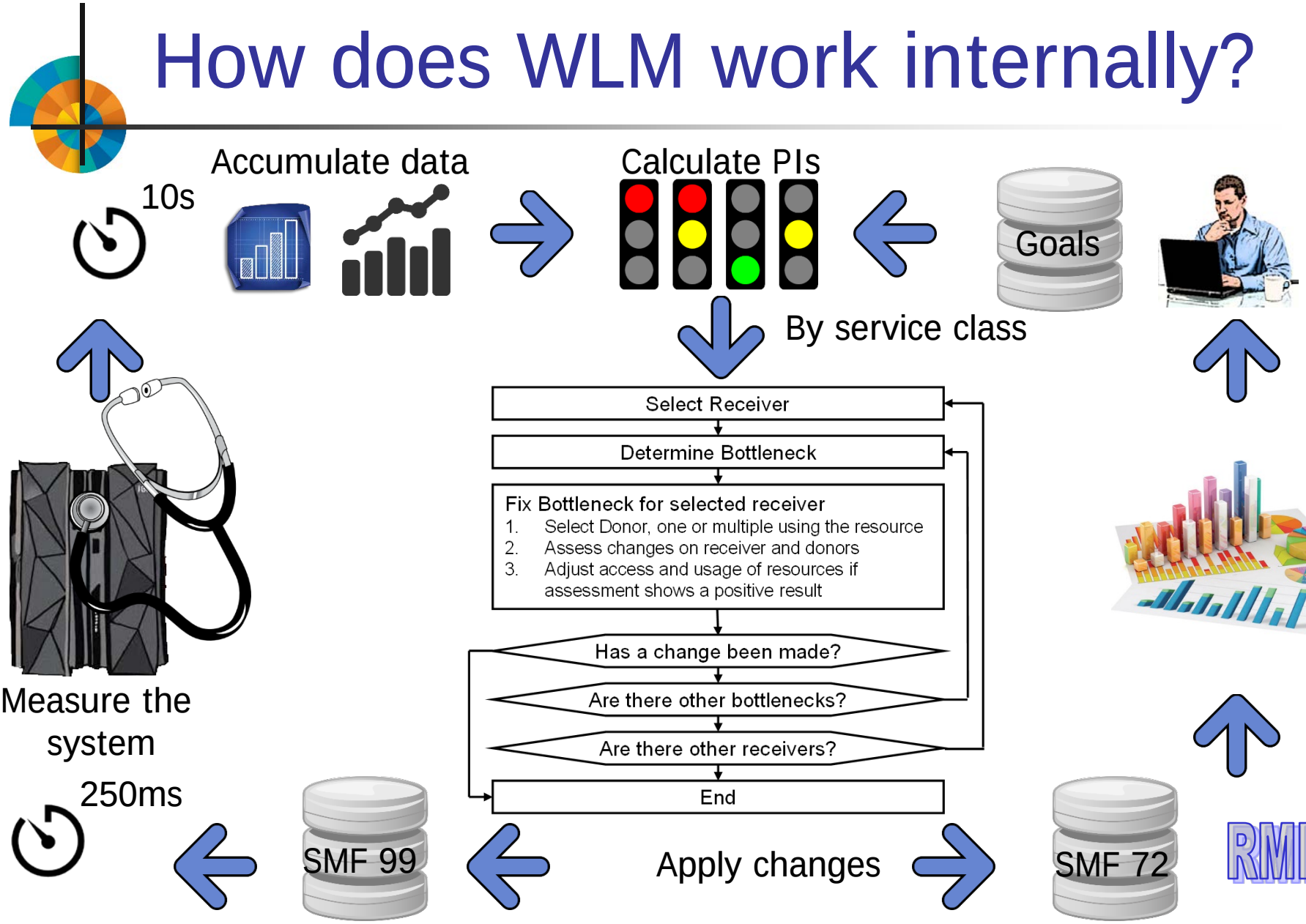
- Indicator how well work (service class) is doing
  - Two exist per service class
    - Sysplex and Local PI
- Basically
  - If PI > 1: service class is eligible for help
  - If PI < 1: service class is a potential donor
  - Independent from business importance
- How is a service class helped
  - By Business Importance
  - By worst PI



$$Response\ Time\ Goal \quad : \quad PI = \frac{Actual\ Achieved\ Response\ Time}{Response\ Time\ Goal}$$

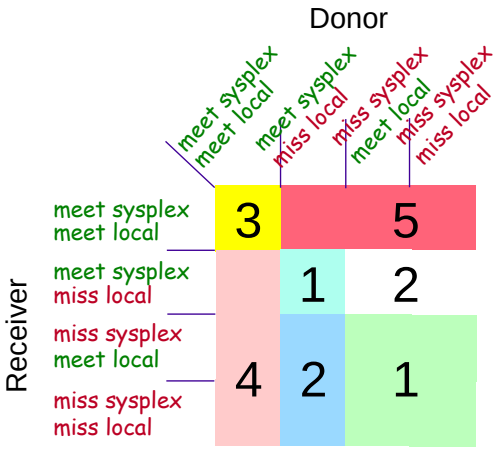
$$Execution\ Velocity\ Goal \quad : \quad PI = \frac{Execution\ Velocity\ Goal}{Actual\ Achieved\ Execution\ Velocity}$$

How does WLM work internally?

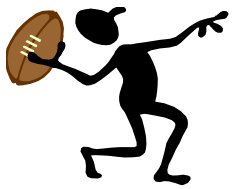


How does WLM work: Algorithms

- WLM assesses work based on its
  - Current and past resource consumption
  - Current and past work endings
  - Actual goal achievement
- WLM selects work based on its
  - Goal achievement and importance level
- WLM tries to help work
  - To improve its performance index
  - To improve its resource consumption
- WLM projects all changes and assesses always how a change effects other work in the system



1. receiver higher rank as donor => ok
   receiver same rank + pi_or_gs_gain => ok
2. receiver higher rank as donor => ok
3. if pi_or_gs_gain => ok
4. always => ok
5. never



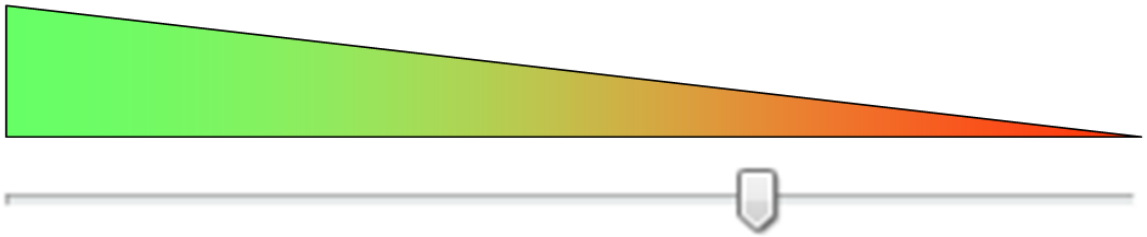
Part III: Work Protection

CPU Critical
Storage Critical
Resource Groups
I/O Priority Groups
Discretionary Goal Management

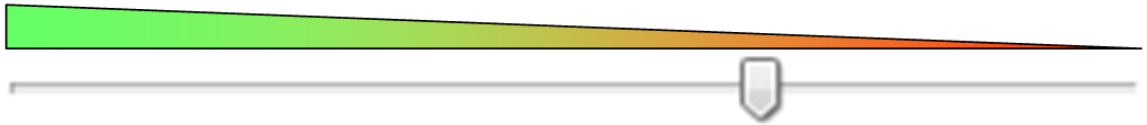
Protecting Work

Unrestricted Management
Most efficient for best resource use

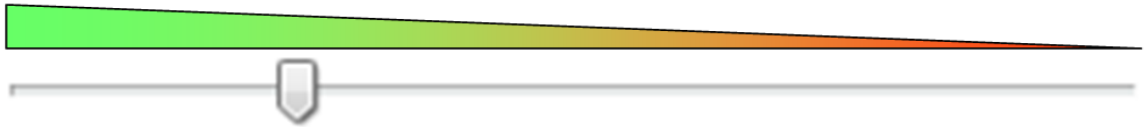
Restricted Management
Very dedicated System requirements



CPU Resource Consumption (Resource Groups)



CPU Dispatch Priority Assignment (CPU Critical)



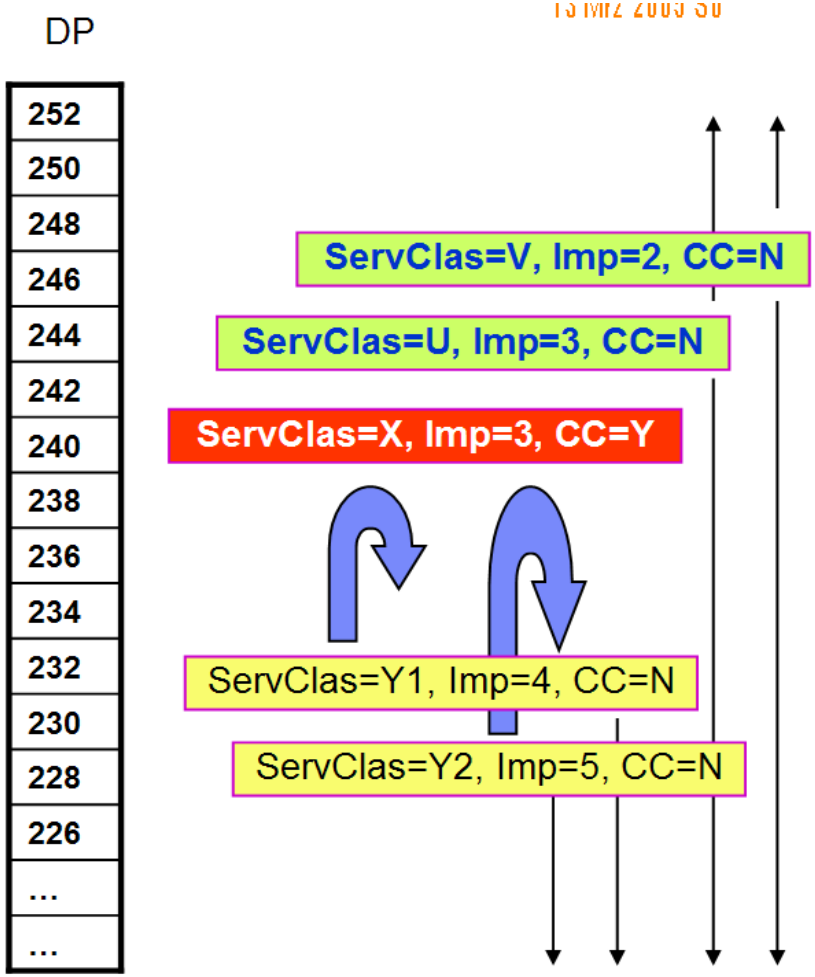
Storage Protection

Protecting Work: CPU

- To avoid that lower important work may get a higher dispatch priority than higher important work
  - Define a tight goal!
    - 0.9 < Performance Index < 1.2
    - Ensures that a high important service class does not become donor
    - But also requires a “constant” flow of work
  - Or use CPU Critical as the last resort

CPU Critical: What it is

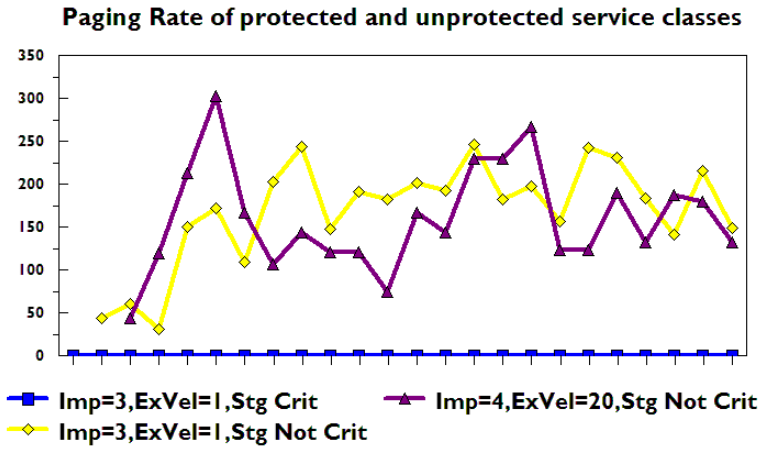
- Ensures that lower important work never gets a higher dispatch priority than high important work for which this attribute has been defined !!
  - That's it, only that !!
  - What does this mean?
    - It means that you still need a goal for CPU Critical work !!

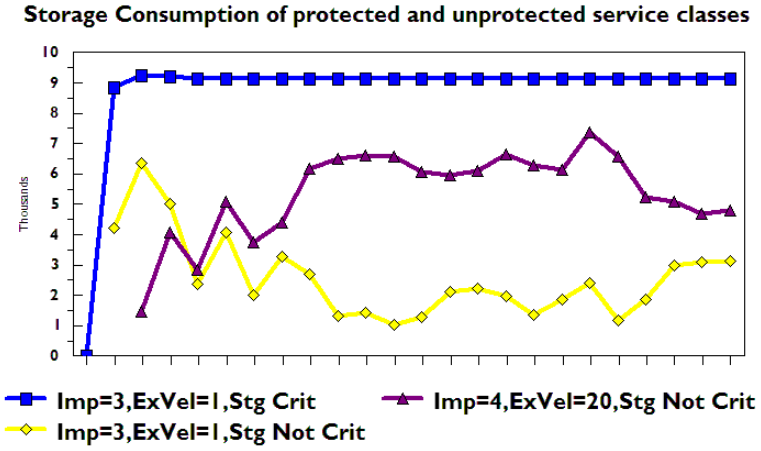


CPU Critical: What it isn't

- Not: A full protection against anything
  - That was never the intention
  - CPU Critical (and Storage Critical) do not put the system back in compatibility mode
    - They simply reduce the freedom of the algorithms a little bit
  - The system still runs in goal mode, therefore
    - The goals of a CPU critical service class still matters
    - The goal achievement is still taken into consideration for all kinds of adjustments
    - And it is not planned to limit the ability of the system to manage workloads dynamically any further

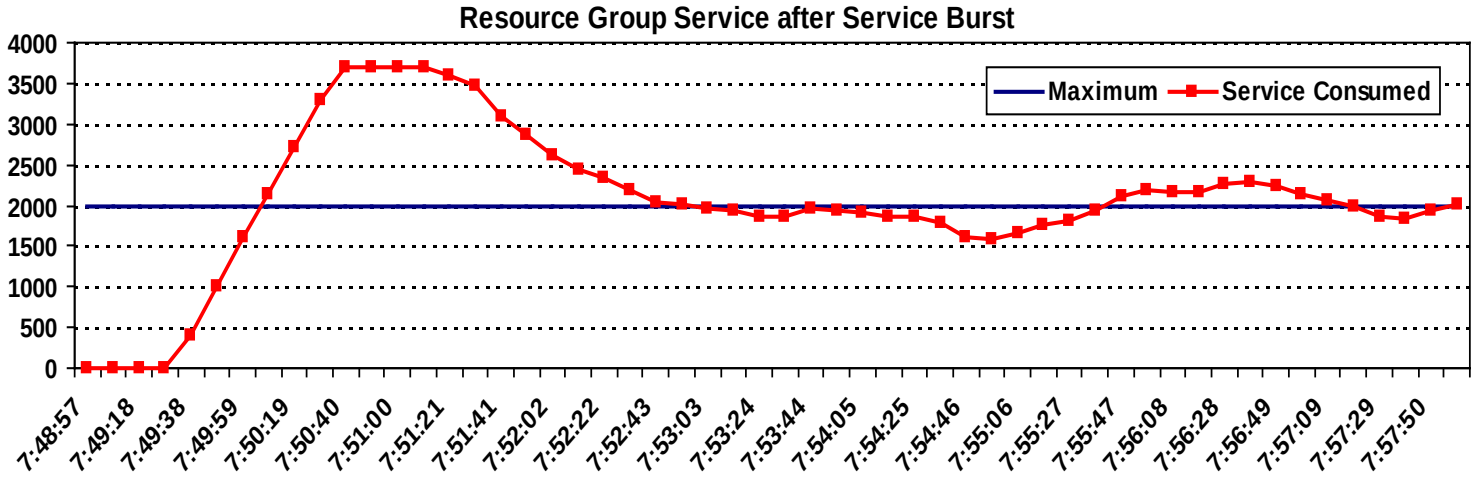
Protecting Work: Storage





- Ensures that required central storage frames of storage protected service classes is not taken away unless
  - The system is in a storage shortage
  - Higher important work needs it and no other service class is able to give up storage anymore
- Understand
  - This storage is “taken away” and increases the stress for other workloads, see above
- Notice
  - Storage protected address spaces can loose up to 25 pages/s if they don’t reference the storage anymore

Resource Groups



- Resource Groups allow you to set minimum and maximum limits for certain work on the system
  - Good for work
    - Which wants to consume too much ... Maximum
    - Which has constant requirement for CPU resources ... Minimum
  - Not so good for
    - Work with bursts and no constant resource requirements
    - Or if the resource group maximum is defined to low
      - Consider 5% of a system
      - But also take the work distribution in a sysplex into account
  - It is NOT an easy control

I/O Priority Groups

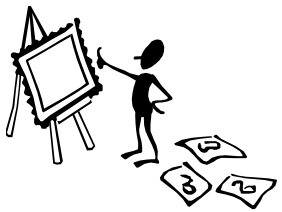
- New with z/OS V2.1
- Problem
  - CICS: 1 service class, using a distinct set of I/O volumes
  - Batch: multiple service classes, competing for I/O volumes
  - Most of the time Batch never uses the I/O volumes for CICS
    - Possible Result: Batch service classes may get higher I/O priority than CICS service class
  - Then Batch jobs start to use I/O volumes from CICS
    - Result: the Batch service classes have better access to the volumes (higher IOP) and CICS has to wait
- I/O Priority Management is very complex
  - Therefore it is done seldom by WLM
  - WLM may need minutes to recognize the situation before it can react on it
- Solution
  - Introduce I/O Priority Groups to separate the I/O priority Ranges for different types of work and to ensure that certain work has priority to other work

I/O Priority Groups: Ranges

<table>
<tr><th></th><th colspan="2">I/O Priority Management=YES</th></tr>
<tr><th>Priority</th><th>I/O Priority Groups NOT enabled</th><th>I/O Priority Groups enabled</th></tr>
<tr><td>FF</td><td>SYSTEM</td><td>SYSTEM</td></tr>
<tr><td>FE</td><td>SYSSTC</td><td>SYSSTC</td></tr>
<tr><td>FD</td><td rowspan="5">Dynamically managed</td><td rowspan="5">Priority Group = HIGH</td></tr>
<tr><td>FC</td></tr>
<tr><td>FB</td></tr>
<tr><td>FA</td></tr>
<tr><td>F9</td></tr>
<tr><td>F8</td><td rowspan="6"></td><td rowspan="6">Priority Group = NORMAL</td></tr>
<tr><td>F7</td></tr>
<tr><td>F6</td></tr>
<tr><td>F5</td></tr>
<tr><td>F4</td></tr>
<tr><td>F3</td></tr>
<tr><td>F2</td><td>Discretionary</td><td>Discretionary</td></tr>
</table>

Discretionary Goal Management

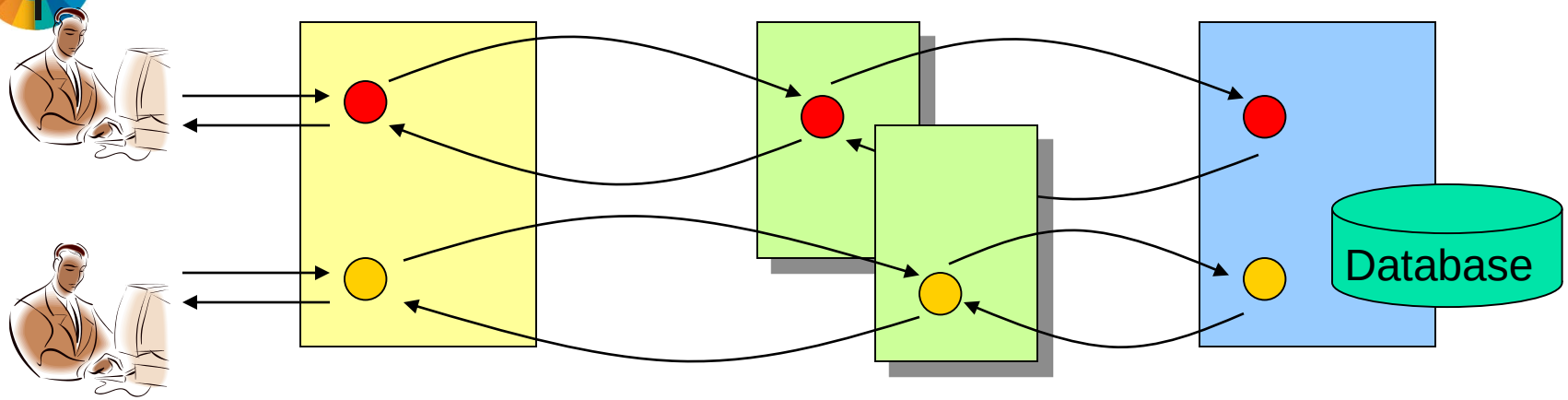
- Background
  - WLM tries to ensure that work (service classes) meet their goals!
  - But what if service classes over achieve their goals and discretionary work can't make any progress?
- Idea
  - Help discretionary work if service classes exist which overachieve their goals
- Implementation
  - Service classes can become donors for discretionary work if
    - They are not member of a resource group!
    - They have non aggressive goals!
      - Execution velocity goal ≤ 30
      - Response time goal > 60 sec
    - The performance index < 0.7
    - Plus some more – externally non visible – requirements



Part IV: Transactional Work

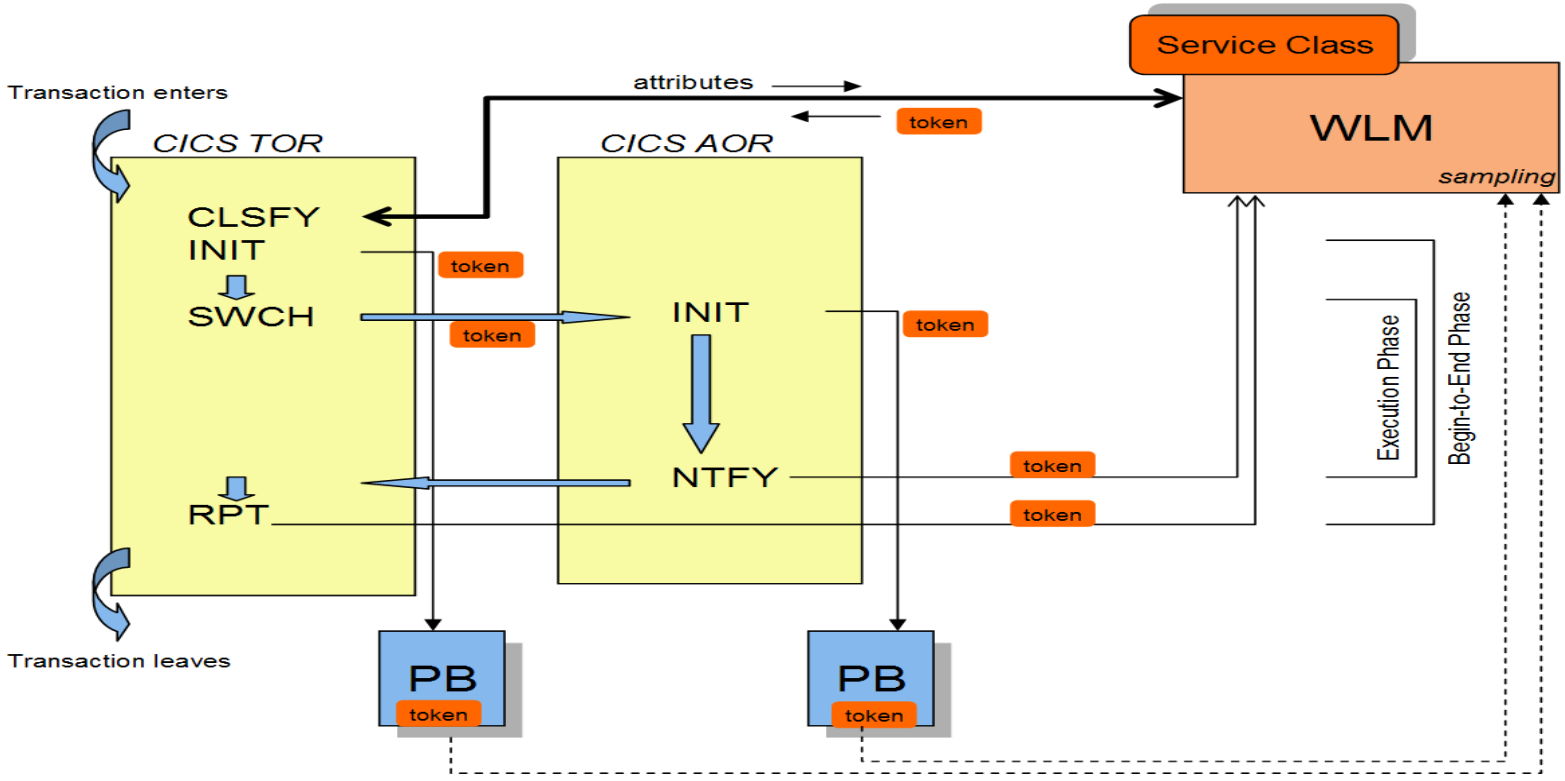
Transaction Flow
Management of CICS and IMS
Enclave Management
Examples

Transaction Flows



- Work requests span multiple address spaces
  - Typically for Transaction Monitors like CICS, IMS, Websphere, DB2 Stored Procedures, etc
- WLM provides interfaces to give these subsystems the ability
  - To tell WLM when transactions arrive and when they end
  - To tell WLM which address spaces, TCBs and SRBs execute programs on behalf of the transactions
- Two facets
  - Management for CICS and IMS
  - Management for exploiter of Enclave Services (Websphere, DB2 Stored Procedures, DB2 Distributed Data Facility, and many more)

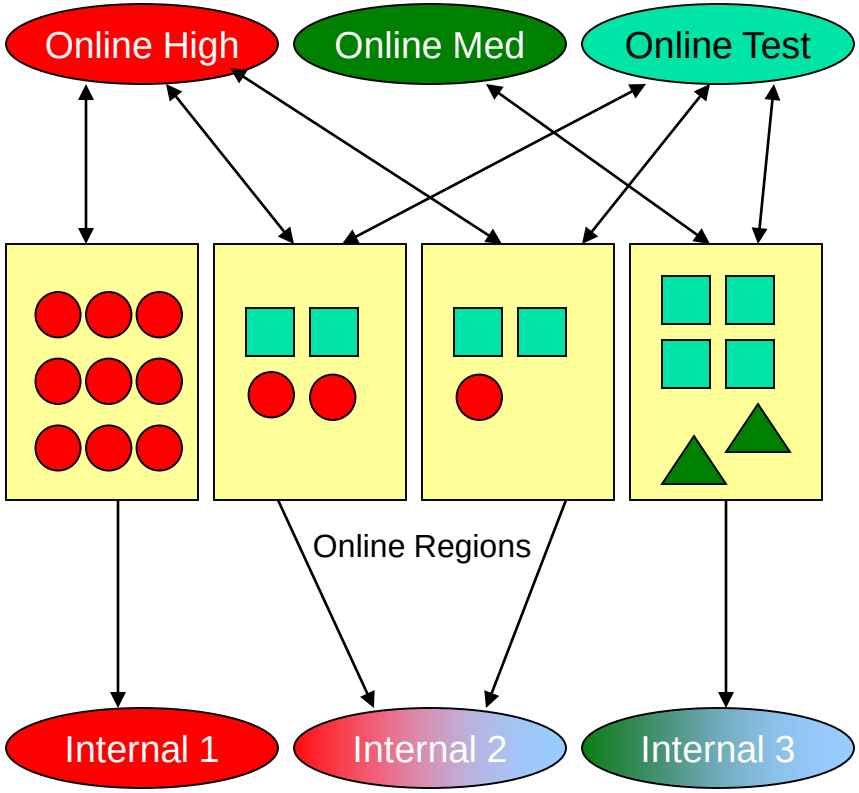
Management: CICS and IMS



- For CICS and IMS a set of interfaces is used which allows WLM to determine the set of transactions running in the respective address spaces
  - This allows indirectly to associate the address spaces with the service classes for the CICS and IMS transactions

Management: CICS and IMS

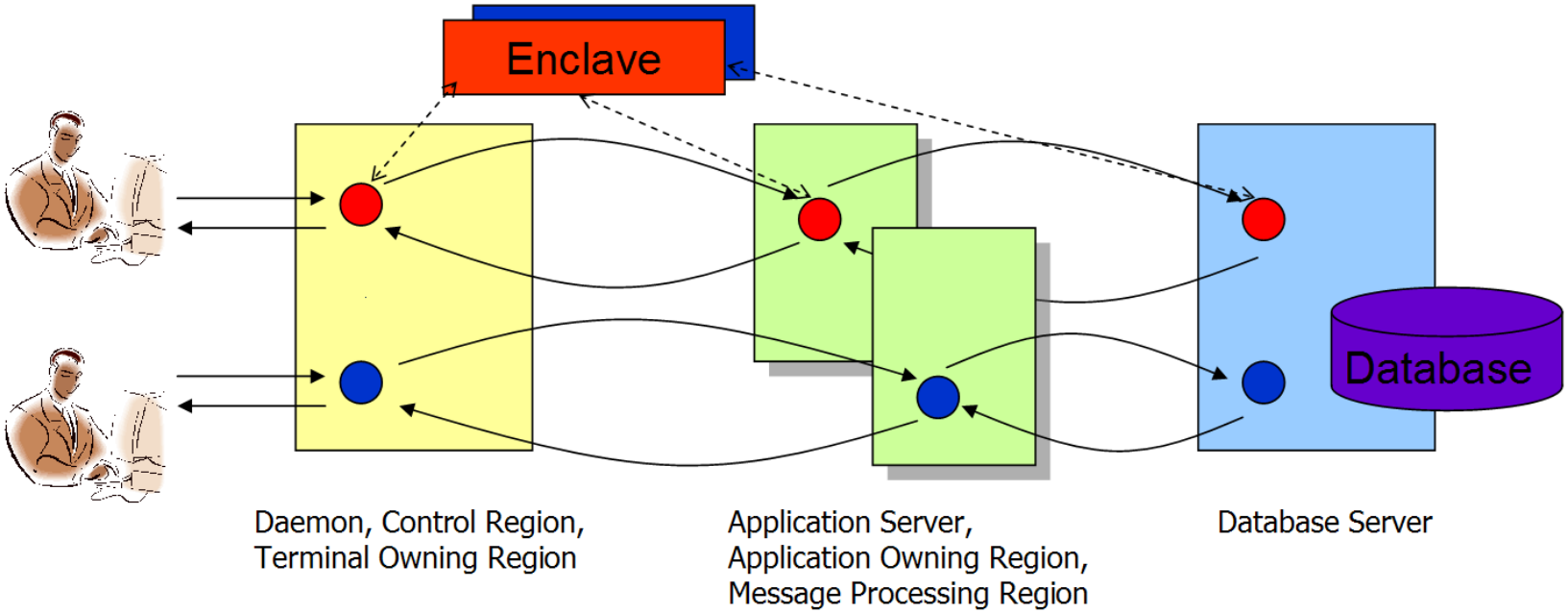
External Service Classes: Define the Goals



Internal Service Classes: Used for Management

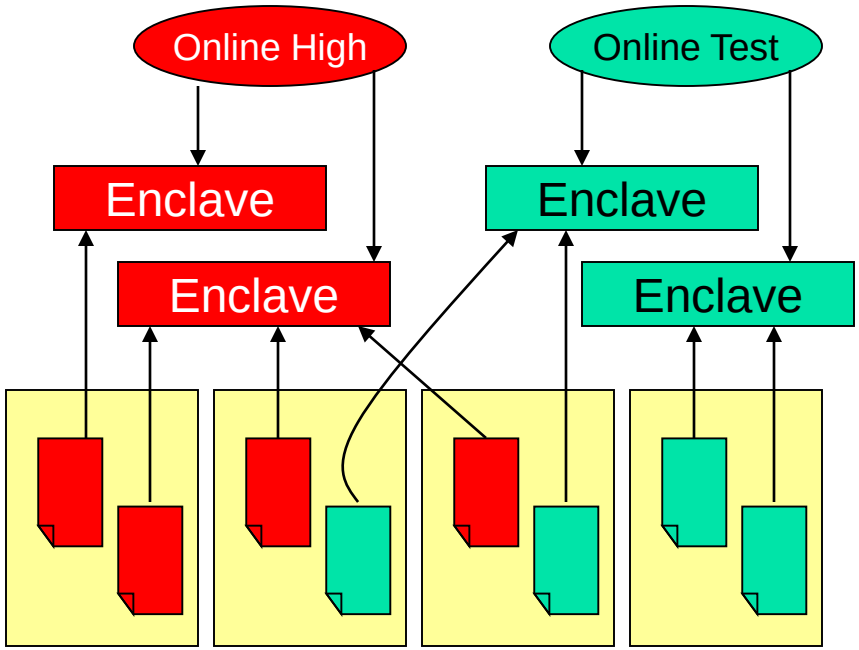
- WLM
  - Finds out which region processes which sets of transactions
  - Puts regions which process the same set of transactions to internal service classes
  - Internal Service classes
    - Inherit goal attributes from the external service classes
  - Manages the regions based on the goals of the CICS/IMS service classes
- That means
  - Too much granularity doesn't really help
  - But it is much more robust than execution velocity goals
  - It provides much better data than execution velocity goals

WLM: Transactions: Enclaves



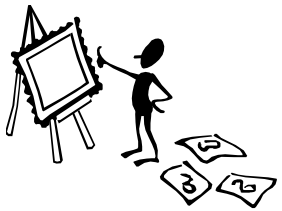
- Allows to directly manage Tasks and SRBs independent from their address spaces
  - At least for CPU and I/O
- An enclave can span multiple execution units across multiple address spaces

Enclave Management



- Tasks and SRBs join enclaves to get managed
- Internal Service Classes exist too but they are much less important

- Various types exist:
  - Independent enclaves
    - True transactions
  - Dependent enclaves
    - Continuations of other work in the system
    - Basically used by DB2 for encapsulating Batch work
  - Work-dependent enclaves
    - Continuation of independent enclaves



Part V: How to define goals

Work Classification
Define Business Importance
Define Goals
How to define periods?
How to use CPU Critical and Resource Groups?
Use Response Time Goals for CICS and IMS
Special function BOTH for CICS (and IMS)

What needs to be done?

- You need to classify all work which comes into your system
- You need a schema to setup up business importance
- You define goals
  - They represent how much resources are required
  - They represent the service level which is provided

Classify Your Work

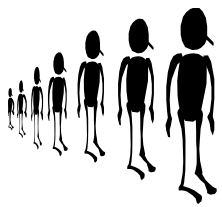
```
  Subsystem-Type  Xref  Notes  Options  Help
------------------------------------------------------------------------------
                    Modify Rules for the Subsystem Type          Row 1 to 6 of 6
Command ===>                                                  SCROLL ===> PAGE

Subsystem Type . : STC          Fold qualifier names?   Y  (Y or N)
Description  . . . Started Tasks

Action codes:   A=After      C=Copy        M=Move      I=Insert rule
                B=Before     D=Delete row  R=Repeat    IS=Insert Sub-rule
                                                          More ===>
           --------Qualifier--------                 -------Class--------
Action     Type        Name      Start                Service      Report
                                                DEFAULTS: STC
  _____  1  SPM         SYSTEM    ____                   SYSTEM
  _____  1  SPM         SYSSTC    ____                   SYSSTC
  _____  1  TN          PCAUTH    ____                   SYSSTC
  _____  1  TN          TRACE     ____                   SYSSTC
  _____  1  TN          SYSBMAS   ____                   SYSSTC
  _____  1  TN          %%%%IRLM  ____                   SYSSTC
****************************** BOTTOM OF DATA ********************************
```

from IBM Sample Service Definition: IWMSSDEF

- Started Tasks
  - Make sure that all system and privileged STCs run in SYSTEM or SYSSTC
    - Possibility: use SPM rules as shown above
  - PCAUTH, TRACE, SYSBMAS are not automatically defined to SYSSTC
    - Classify them explicitly
  - IRLM should also run in SYSSTC

Classify Work

- Make sure that you define only the number of service classes you really need
  - Hints:
    - Don’t define service classes for reporting purposes
      - With report class periods you can get any granularity you need
    - Make sure that something really executes in your service classes
      - Don’t define anything with less than 2% of CPU resource consumption
      - BUT: classify distinct work to distinct service classes, for example:
        - Don’t classify short living enclaves and STCs together
        - Never classify CICS/IMS transactions together with anything else
      - If you use multiple periods
        - Make sure that some substantial amount of work is really ending in those periods
        - More than 3 are very often counter productive
  - Use Guideline
    - Not more than 25 (max 35) [actively running] service class periods with non discretionary goals



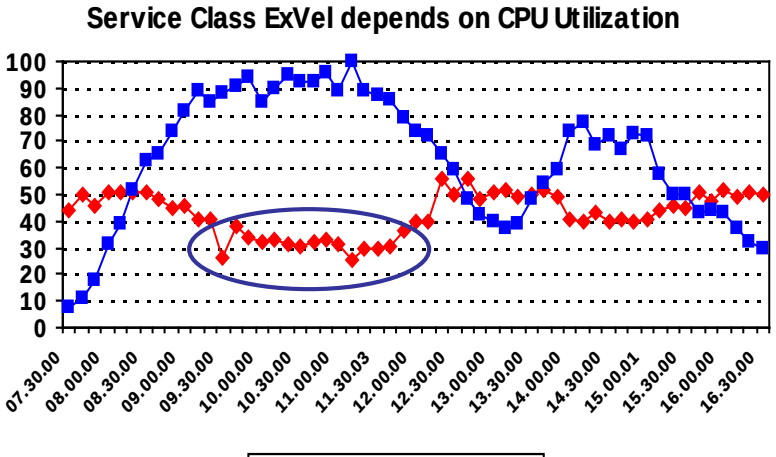
The Business Importance

- You need a schema to start with
- And some few rules how you use importance levels
  - Rule 1: Most important production runs at importance 2
  - Rule 2: Everything which needs fast access to CPU and which is used by production runs above
  - Rule 3: Everything else runs below

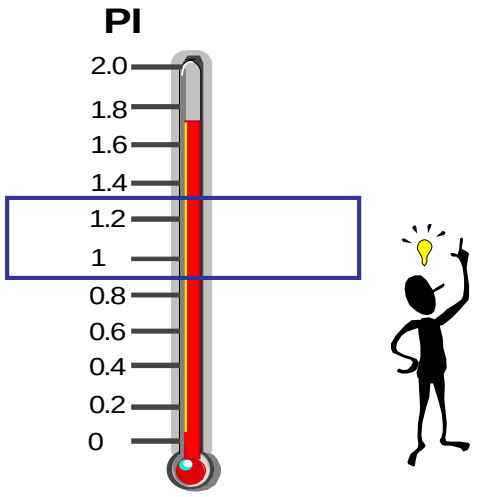
System	Pre-defined system address spaces
SYSSTC	Important monitors, automation, IRLM, everything which consumes very little CPU and requires fast and instantaneous access to CPU
Importance 1	Critical server address spaces, IMSDBCT, DBM1, MSTR, TORs when managed towards BOTH, monitors, system programmer TSO
Importance 2	Most important production work, your loved ones
Importance 3 and 4	Less important production, TSO, very critical Batch
Importance 4 and 5	Batch, TSO last period, all default service classes which do not need any attention
Discretionary	Batch

Define the Goals

- Base your definitions on measurements
  - Especially execution velocity goals can't be defined without knowing the data
    - Use periods of high utilization or contention as a base
    - Don't assume that you can enforce high execution velocities
      - There are physical restrictions

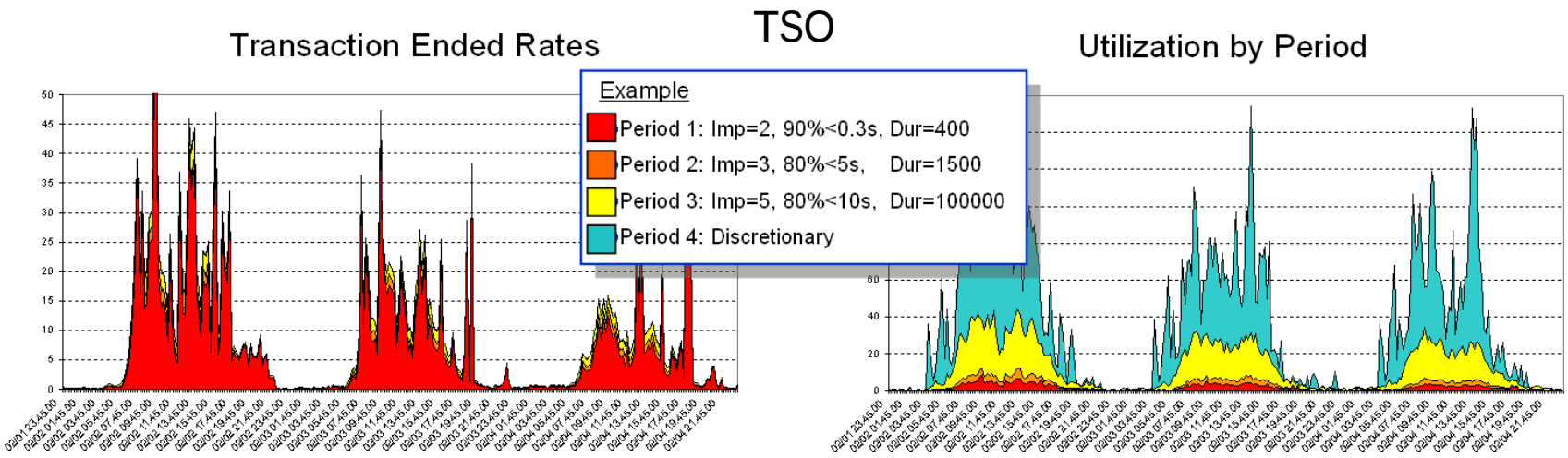


- If you want that WLM always keeps an eye on your service class
  - Define a tight goal:
    - 0.9 < PI < 1.2
    - A little bit high temperature isn't a bad thing
  - Only if that doesn't really help use CPU or Storage Critical
    - Can be avoided in many cases



How to use Periods?

- Only if necessary and only as much as really needed!
- Use lower importance levels and less stringent goals for higher periods
- Define durations which ensure that the majority of short running transactions end within the first and second period
- Suited for:
  - TSO, OMVS — 3 periods (seldom 4)
  - Mass or common Batch (no production Batch) — 2 periods (seldom 3)
  - Some DDF work with unpredictable resource requirements — 2 periods
  - Seldom: WAS — 2 periods



When to use CPU Critical?

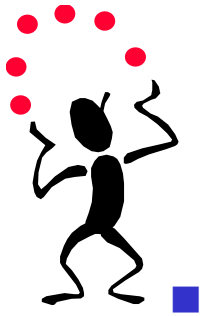
- Use it seldom
  - For 1 or 2 service classes
  - At best only for importance 1
  - Suited is work which doesn't consume much CPU, which doesn't show high consuming CPU spikes, and which needs fast access to CPU
    - Examples:
      - Critical server address spaces
- Rule: use the similar rule of thumb for work which you would place in SYSSTC
  - To SYSSTC
    - Less than 20% of 1 logical processor
    - No high consuming CPU spikes
  - CPU Critical
    - Less than 5 to 10% of total system consumption (depending on size of system)
    - No high consuming CPU spikes

How to use Resource Groups?

- First: don't use it for OLTP work
  - At least don't be surprised if your OLTP work starves otherwise
- Use of maximum
  - Possible for batch work
    - Can best deal with limitations
    - May result in elongated elapse times and you need to assure that you can afford them
- Use of minimum
  - For medium and lower important work which need to make some progress
    - But be very careful
      - A resource group minimum supersedes all goals and all importance
      - That means the resource group minimum comes first

Use response time goals for CICS or IMS!

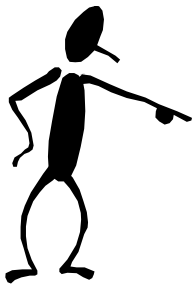
- What are the benefits?
  - You can easily find out how long the transactions need on your system
  - You can easily define report classes to monitor subsets of your transactional work
  - It can easily be related to a Service Level Agreement
  - Response time goals are very stable against configuration changes



- Execution velocity goals (disadvantages):
  - Require that you convert the response time expectation to an acceptable amount of delay
    - That requires detailed measurements
    - It often ends with too high goal settings (and CPU Critical)
  - Are very sensitive to configuration changes

Response Time Goals for CICS and IMS

- Some recommendations
  - Start with few external service classes
    - Perhaps only 1
    - The Subsystem Instance classification rule is a good starting point for classifying work for both subsystems
  - Understand how the transactions behave
    - Is think time included?
    - Is there high variability?
  - Try to use percentile response time goals
    - They can better deal with variability
  - You may want to exclude test work from being managed towards response time goals
    - Too few transactions, no constant utilization and/or no need to manage it in a sophisticated way
  - And you need to monitor the settings
    - Response time distribution charts and comparisons with subsystem specific monitors are usually easy and help



Response Time Goals for CICS and IMS

- Start with monitoring only
  - Define a single service class for CICS (IMS)
  - Exempt all regions from being managed towards TRANSACTION
    - On Classify panel scroll to right twice (F11)
  - Define at least one or multiple report classes for the transactional work
- Result
  - The work is still managed towards execution velocity goals
  - The response time and its distribution can be examined thru the report class(es)
- What to do next
  - Monitor the system and adjust the service class goal
  - When you are comfortable with the settings
    - Switch to TRANSACTION management for your regions
- Notice
  - This is an iterative process

Example: Too many external service classes

Workload	Service Class	Per	Imp	Printed
CICS	MOSTTRAN	1	2	80% complete within 00:00:00.080
CICS	SPECTRAN	1	2	80% complete within 00:01:40.000
CICS	SYSTRAN	1	3	80% complete within 00:00:01.000
STC	REGIONS	1	3	Execution velocity of 50
CICS	DEFTRAN	1	4	50% complete within 00:00:02.000
CICS	LONGTRAN	1	4	50% complete within 24:00:00.000

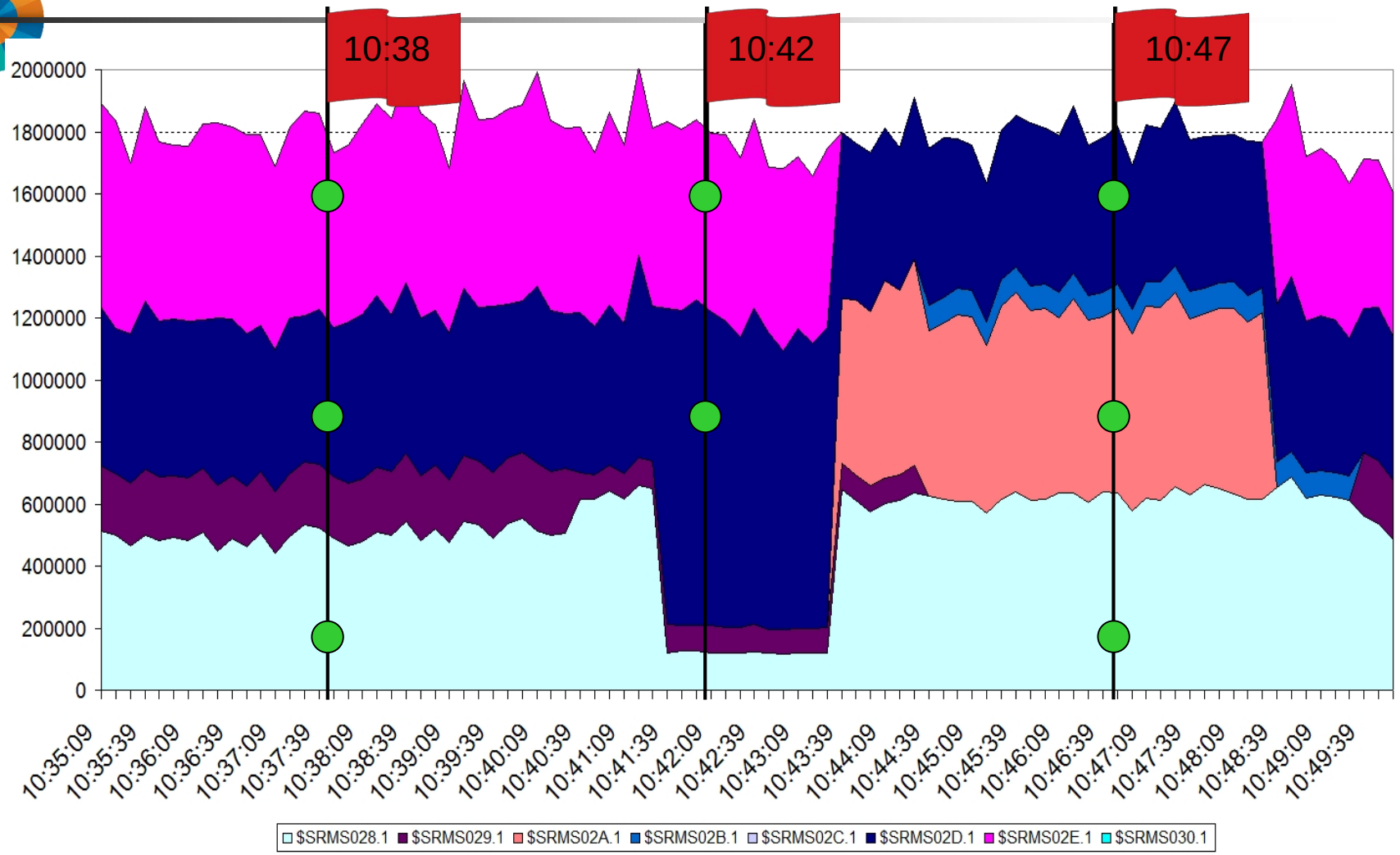
10:38	SYSTRAN	DEFTRAN	MOSTTRAN	SPECTRAN	LONGTRAN	
SSRMS028	17		19091		83	
SSRMS029	67		851			
SSRMS02A	0		0	0	0	
SSRMS02B	0	0	0			
SSRMS02C						
SSRMS02D	24	1	19857		66	
SSRMS02E	28	1	20061	5102	61	
SSRMS030	0	0	0	0		
Total Observations	136	2	59860	5102	210	65100

10:42	SYSTRAN	DEFTRAN	MOSTTRAN	SPECTRAN	LONGTRAN
SSRMS028	52		797		1
SSRMS029	35		458		
SSRMS02A	0		0	0	0
SSRMS02B	0	0	0		
SSRMS02C					
SSRMS02D	45	2	39971		160
SSRMS02E	28		20498	5333	69
SSRMS030	0	0	0	0	
Totals	160	2	61725	5333	230

10:47	SYSTRAN	DEFTRAN	MOSTTRAN	SPECTRAN	LONGTRAN
SSRMS028	146		20982		78
SSRMS029	0		0		
SSRMS02A	27		19293	5309	78
SSRMS02B	89	1	1262		
SSRMS02C	0		0	0	
SSRMS02D	13	3	19224		79
SSRMS02E	0	0	0	0	0
SSRMS030					
Totals	276	4	60761	5309	234

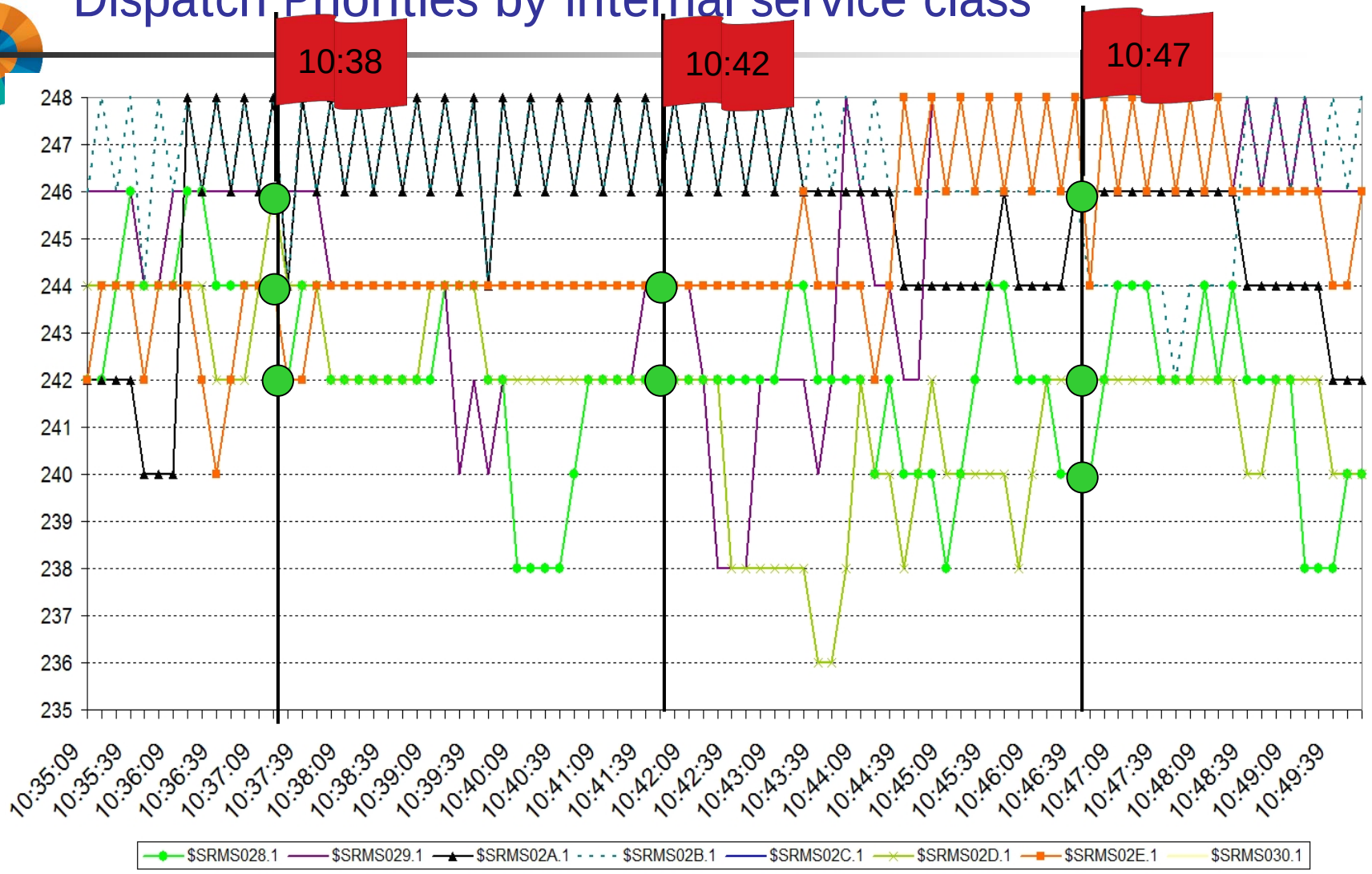
- Example shows 5 external service classes for CICS work
  - WLM creates internal service classes up to 2^n - 1
- MOSTTRAN is the most important work
- At the right side
  - Three points in time shown with a distribution of work to internal service classes (regions)
    - MOSTTRAN often runs in 3 but sometimes only in 2 internal service classes
      - This depends with which work MOSTTRAN must share the regions
- What are the effects?

Example: Too many external service classes
Service consumption by internal service class



● = MOSTTRAN transactions

Example: Too many external service classes
Dispatch Priorities by internal service class



● = MOSTTRAN transactions

Example: Too many external service classes

- What can we learn
  - MOSTTRAN transactions are all the same
    - MOSTTRAN transaction receive different treatment
    - This only depends on which other types of transactions execute in the same regions
  - MOSTTRAN are also really the most often found transaction type
    - Regions processing MOSTTRAN consume the highest amount of CPU
    - The idea was to separate some long running and some less important transactions from MOSTTRAN
    - But what is the result?
      - The only effect the other transactions have is that MOSTTRAN transactions receive different service
      - Because the other transactions occur relatively seldom compared to MOSTTRAN they consume very little resources (even if they are really long lasting)
        - If they run in regions which do not process MOSTTRAN transactions they get BETTER service than MOSTTRAN
- The net
  - ☑ Only created different service classes if the work really executes in different regions otherwise the result is unpredictable
  - ☑ Percentile response time goals take care of different types of work

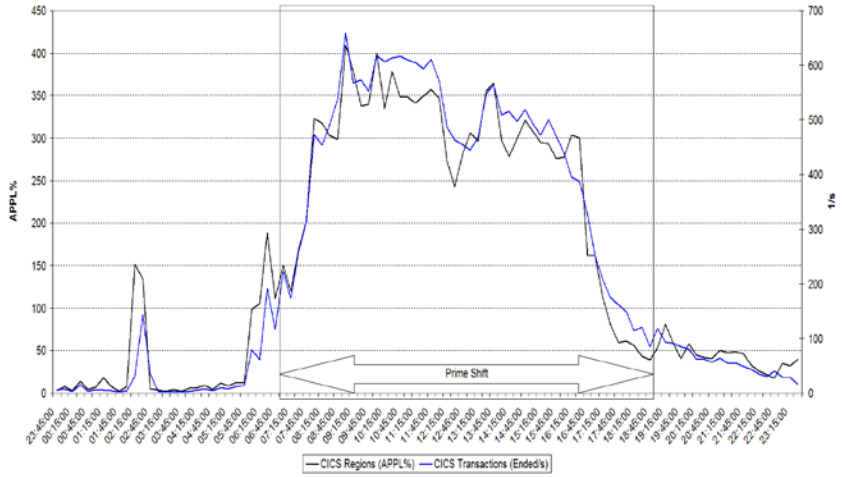
Response Time Goals for CICS and IMS
Percentile Response Time Goals

bucket#	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Fraction of Goal-Response-time	≤0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	2.0	4.0	>4.0
% of endings	30	12	8	10	10	10	4	6	0	0	0	0	5	5
Σ of % endings	30	42	50	60	70	80	84	90	90	90	90	90	95	100

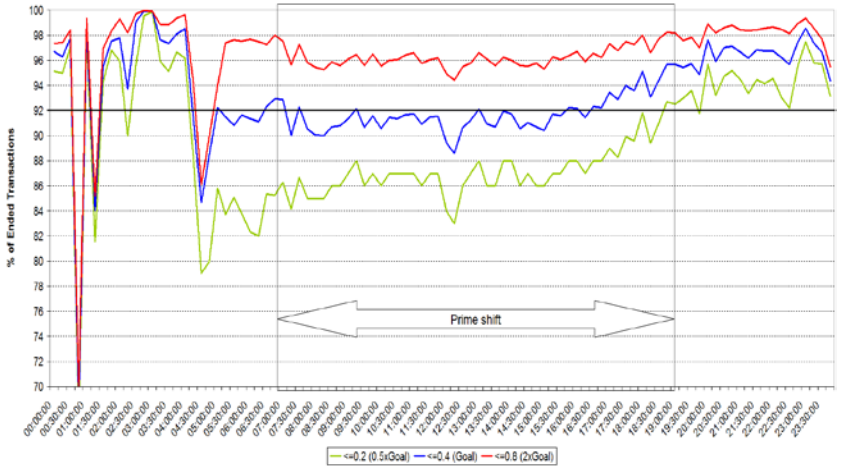
- Percentile Response Time Goals are very stable and provide the best mean to manage also divers work in the same service class
  - Example: 85% of all transactions should complete in 1 sec
    - WLM collects the ending transactions within buckets around the goal from ½ to 5 times the goal
      - RMF displays 14 buckets (example above)
    - To determine the PI WLM looks for the bucket which accounts for at least the defined percentage of ended transactions
      - In the example this is bucket 8
    - The corresponding response time value divided by the goal response time value determines the performance index
  - Consequences
    - The bulk of transactions matters for managing the work
    - Some outliers are not impoartant, just let them run in the service class, they do not disturb the management

Response Time Goals for CICS and IMS
Percentile Response Time Goals

Region APPL% and Ended CICS Transactions



Accumulated Ended Transactions

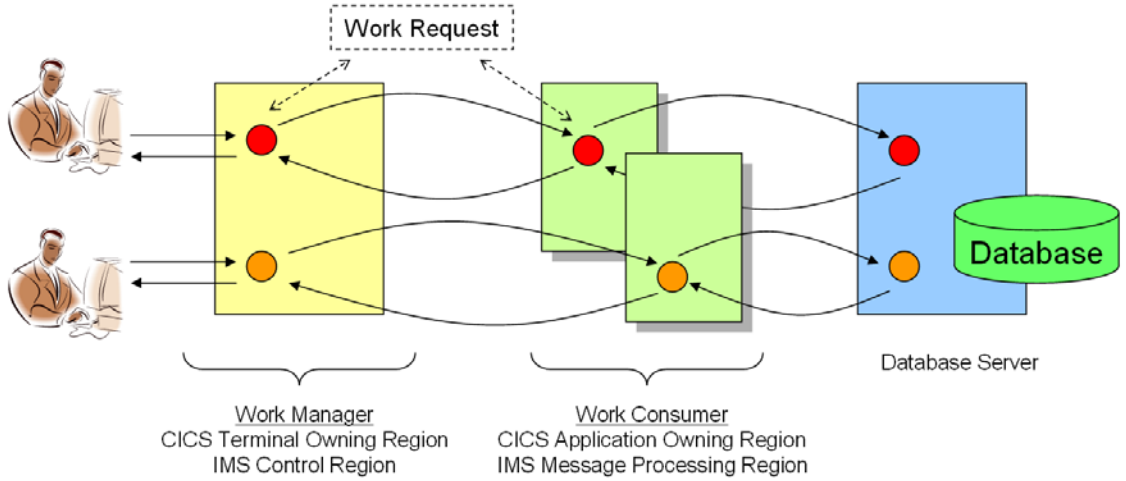


- Left Graphic: displays the load (CPU and transactions) for CICS work
- Right Graphic: displays 3 accumulated buckets for ended CICS transactions
  - Green: Transactions ended within ½ the goal value
  - Blue: Transactions ended within the goal value
  - Red: Transactions ended within 2 times the goal value
- Goal Value: 92% < 0.4 seconds
- Result: CICS response times are very stable throughout the prime shift (with high load)

Execution Velocity Goals for CICS and IMS

- Make sense for test environments
  - Without a constant flow of transactions
- If test and production runs on the same system or sysplex
  - Use response time goals for production
  - Use execution velocity goals for test
    - Exempt test regions from response time management (Option: REGION)

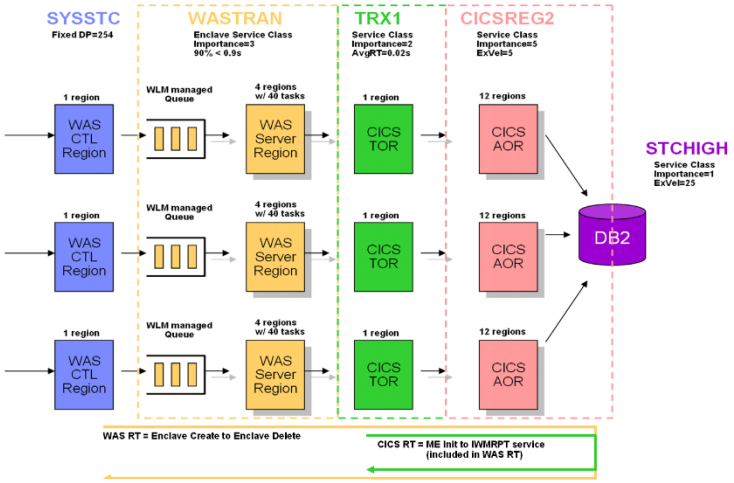
CICS and IMS: Special Function



- CICS and IMS is a work manager, consumer model like DB2 DDF and WAS
  - Remember for DDF and WAS the control regions are classified to high important region service classes
  - The work units run in lower important production service classes
- On system which run to a very high degree CICS and IMS without any work which can be displaced you should use the same schema

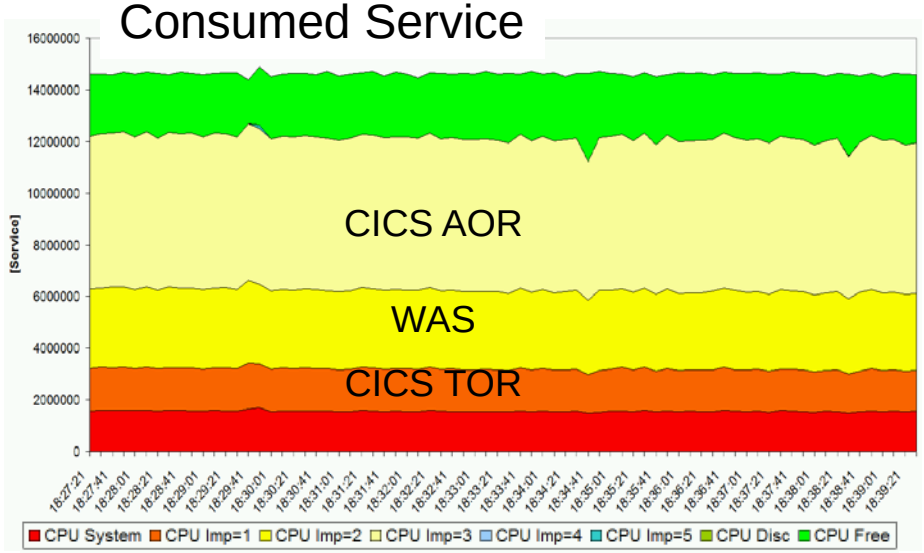
CICS and IMS: Special Function

Scenario



Results	Completed Transactions/sec	Avg. RespTime/sec
w/o „BOTH“	9765	0.197
„BOTH“	12463 **+27%**	0.026 **-86%**

Consumed Service

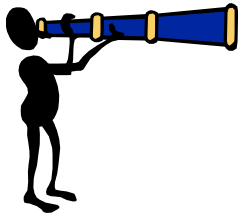


- Problem
  - CICS TOR need fast access to CPU to get work in and out of the system
  - CICS TOR have to wait behind AORs
    - Same dispatch priorities
  - Result:
    - Work throughput decreases at utilizations above 85%
- Solution
  - De-couple TORs from response time management but ensure that response time management remains in tact
  - Option BOTH

CICS and IMS: Option BOTH

```
 Subsystem-Type  Xref  Notes  Options  Help
------------------------------------------------------------------------------
                    Modify Rules for the Subsystem Type          Row 1 to 3 of 3
Command ===> ____________________________________________________ Scroll ===> PAGE

Subsystem Type . : JES          Fold qualifier names?   Y  (Y or N)
Description  . . . Batch Work ____________________

Action codes:    A=After      C=Copy         M=Move      I=Insert rule
                 B=Before     D=Delete row   R=Repeat    IS=Insert Sub-rule
                                                          <=== More
            --------Qualifier---------                Storage     Manage Region
Action      Type        Name      Start               Critical    Using Goals Of

 _____   1  TN          CICSTOR*  ____                NO          BOTH
 _____   1  TN          CICSAOR*  ____                NO          TRANSACTION
 _____   1  TN          CICS*     ____                NO          TRANSACTION
********************************* BOTTOM OF DATA ********************************
```

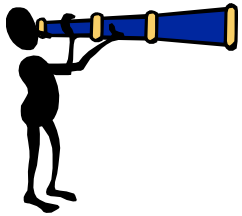
- What to do
  - Separate CICS TORs and AORs to different service classes
  - TOR service class
    - Set to importance 1, define a “high” execution velocity goal, CPU Critical may be an option too
  - Exempt the TORs from being managed towards response time goals by using option BOTH
    - This ensures that the end-to-end context for CICS transactions remains in tact
    - This ensures that CICS transactions (and the AORs) are still managed towards response time goals
    - This maintains all reporting features for CICS
- Why does it work
  - Typically TORs (as well as IMS CTL) only consumes 5 to 10% of the CPU of all CICS/IMS work



Part VI: Monitor your expectations

And Summary Part I to VI

Monitor Your Expectations



Session A - [43 x 80]

File Edit View Communication Actions Window Help

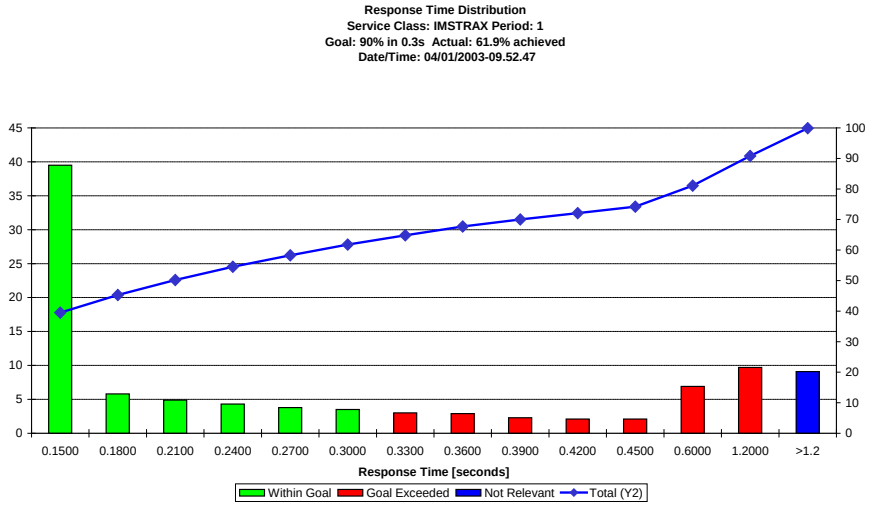
```
                RMF V1R2  Sysplex Summary - WLM1PLEX          Line 1 of 13
Command ===> _                                               Scroll ===> CSR

WLM Samples: 400      Systems: 1  Date: 07/14/03 Time: 15.30.00 Range: 100   Sec

                      >>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>

Service Definition: WLMQUEUE                Installed at: 07/14/03, 15.28.14
     Active Policy: WLMQUEUE                Activated at: 07/14/03, 15.28.23

                 ------- Goals versus Actuals --------   Trans --Avg. Resp. Time-
                 Exec Vel  --- Response Time ---   Perf   Ended  WAIT EXECUT ACTUAL
Name      T  I   Goal Act  ---Goal---  --Actual--  Indx   Rate   Time   Time   Time

BATCH     W          18                              0.290 16.43  9.914  26.31
BTCHDEF   S  5      20  33                    0.60  0.010 0.915  2.034  2.949
BTCHLONG  S  5       5  13                    0.39  0.000 0.000  0.000  0.000
BTCHSHRT  S  5      20  27                    0.75  0.280 16.99  10.20  27.15
STC       W         0.0                             0.000 0.000  0.000  0.000
STCDEF    S  5      20 0.0                     N/A  0.000 0.000  0.000  0.000
SYSTEM    W         9.2                             0.000 0.000  0.000  0.000
SYSSTC    S      N/A 6.6     N/A                     0.000 0.000  0.000  0.000
SYSTEM    S      N/A  32     N/A                     0.000 0.000  0.000  0.000
TSO       W          13                              0.880 0.000  0.245  0.245
TSODEF    S         7.7                              0.380 0.000  0.155  0.155
          1  3      7.7  1.000 80%          97%   0.50  0.380 0.000  0.155  0.155
TSOSPEC   S  1       17  0.500 AVG   0.313  AVG   0.63  0.500 0.000  0.313  0.313


 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=TOGGLE
 F7=UP        F8=DOWN      F9=SWAP LIS F10=BREF     F11=FREF     F12=RETRIEVE
```

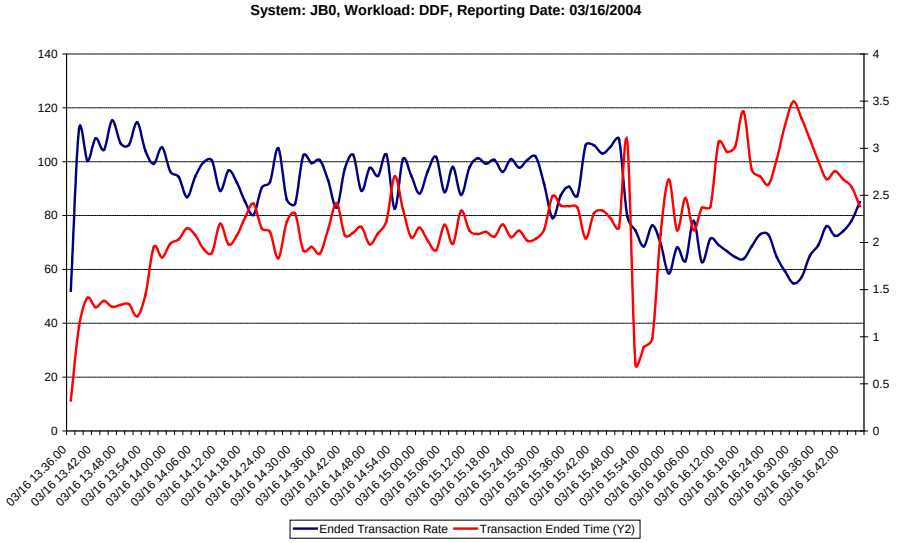
MA a 03/015

Connected to remote server/host boevmct1.boeblingen.de.ibm.com usin

- Goal Mode doesn't mean that you don't have to watch your system or that you doesn't have to do periodic adjustments
  - Many things can change and you have to react to them
  - But you have to use the correct mechanisms

Monitor Your Expectations



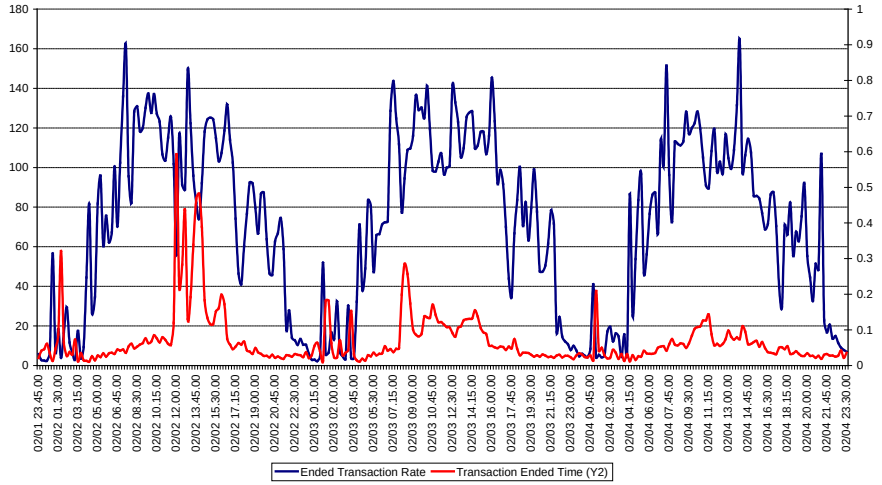


- Make sure to use tools which allow long-term analysis of your data
  - Which data: RMF SMF is the preferred source
  - If you don't have tools in place:
    - Take a look at the RMF and WLM homepage
      - For example: RMF Spreadsheet Reporter provides easy charting and analysis capabilities

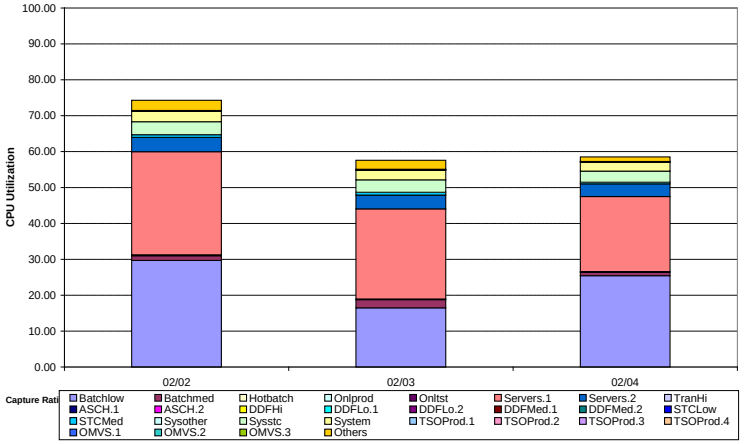
Finally: Reporting

One of the Big Advantages

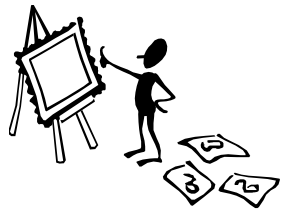
Transaction Counts and Response Times



Utilization by System Workloads



- WLM provides easy mechanisms for long-term data analysis, reporting and capacity planning
  - Report Classes are the base
    - Use them extensively to create a granular picture of your workloads
    - They provide an easy mechanism to capture application growth
  - Transaction counts and response times
    - Allow detailed analysis of application throughput and end use expectations
- Report Classes provide the same level of granularity then service classes



Summary

Setting up a service definition

- Start with business importance
  - That's the key point for distinguishing work
  - Make sure you use as many as possible
- Try to understand which work depends on other work
  - Work which just provides services to other work such as IRLM needs good access to resources
- Define goals which make sense
  - Use periods of high contention
  - Try to understand execution velocities
    - They can vary dramatically
    - Ignore periods of low utilizations
    - They depend dramatically on the number of logical processors and the number of parallel execution units
  - Use always measurements to ensure your settings
    - Some things can't be achieved just be theory
- Don't underestimate the reporting
  - It is one of the biggest benefits of WLM
  - Use report classes to distinguish work and to understand parts of your work
    - Use it extensively it provides a lot of good insight into the system
- Try to use response time goals for CICS and IMS
  - Don't try to setup goals to granular
  - But a response time goal is much more reliable than an execution velocity goal
  - And the reporting aspects already pays every effort back

Summary

- Many problems with existing service definitions occur because people know too much about system internals
- If you haven't done it already
  - Take a step back and forget how you managed a system in compatibility mode
    - Try to think how you would manage a traffic control system
- Try to express your expectations and do not try to internally tune the system
  - You still have to monitor the results
  - You still have to make adjustments
- Use features with care
  - Start to define a service definition without any restricting attributes
  - Apply them selectively and don't over use them
- Periodically reassess your settings

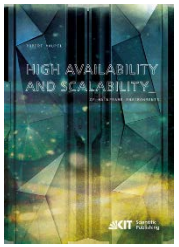
- If you need help
  - There are various places to go, see next page for examples

End

Questions

Documentation

z/OS MVS Planning: Workload Management
z/OS MVS Programming: Workload Manager Services
Systems' Programmer Guide to: Workload Manager SG24-6472
Effective zSeries Performance Monitoring Using RMF SG24-6645



Specials

High Availability and Scalability of Mainframe Environments
Robert Vaupel, 2013, KIT Scientific Publishing, ISBN 978-3-7315-0022-3
free download at: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000034624

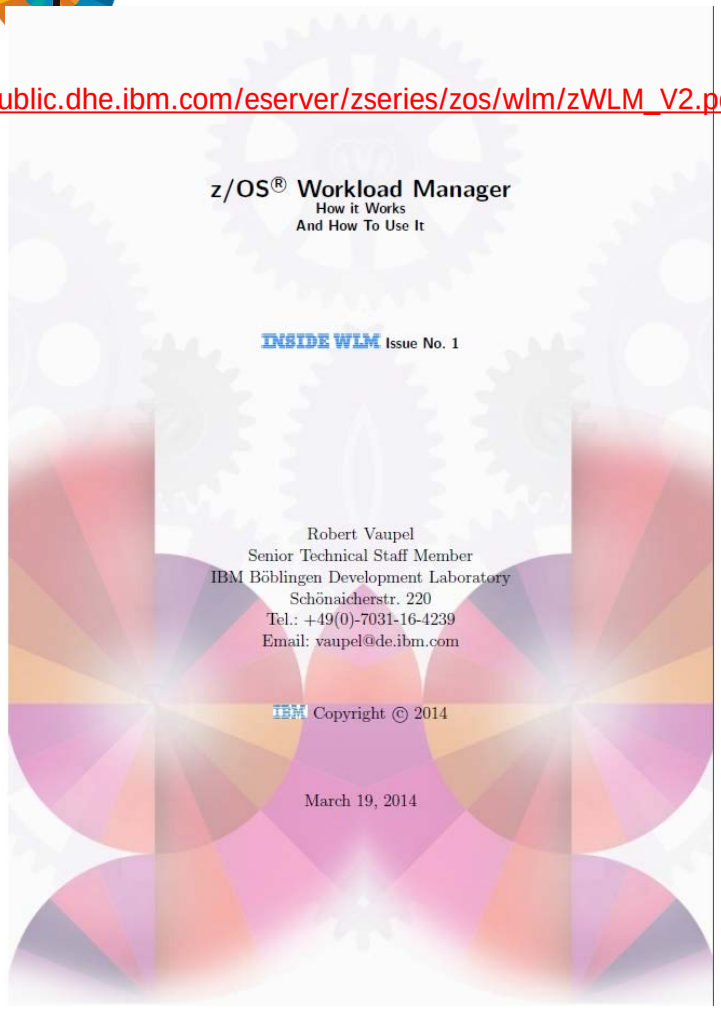
Das Betriebssystem z/OS und die zSeries – Die Darstellung eines modernen Großrechnersystems, M.Teuffel/R.Vaupel, ISBN 3-486-27528-3 (in german)

Internet Links

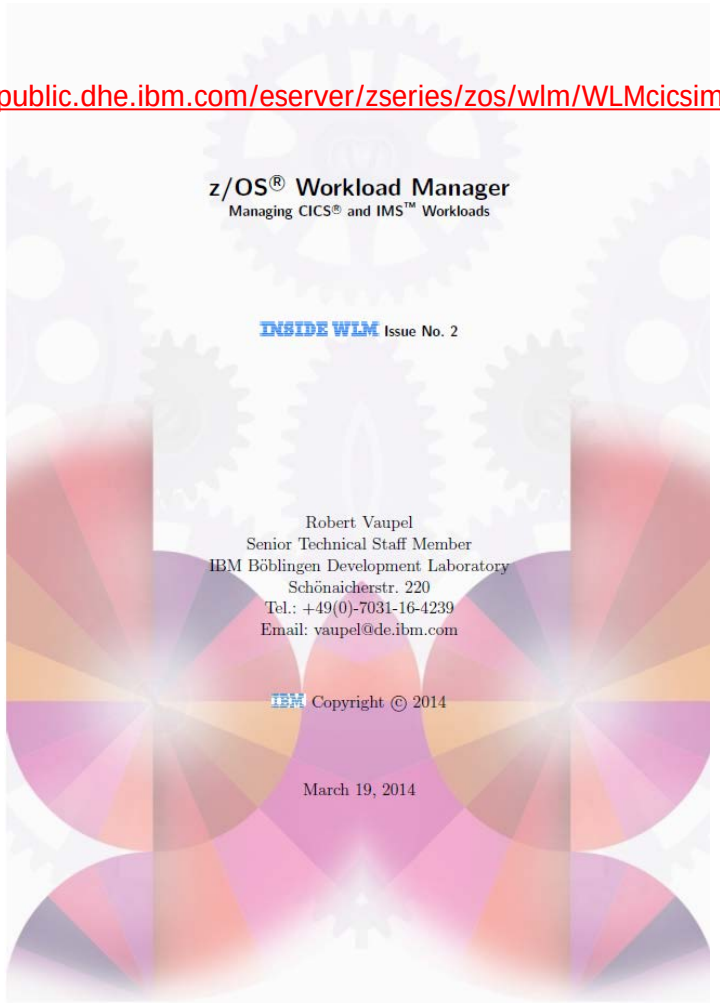
WLM	http://www-03.ibm.com/systems/z/os/zos/features/wlm/
RMF	http://www-03.ibm.com/systems/z/os/zos/features/rmf/
IRD	http://www-03.ibm.com/systems/z/os/zos/features/wlm/WLM_IRD.html
WSC	http://www-03.ibm.com/support/techdocs/atsmastr.nsf/Web/Techdocs
SWPRICE	http://www-03.ibm.com/systems/z/resources/swprice/

Literature on WLM Webside

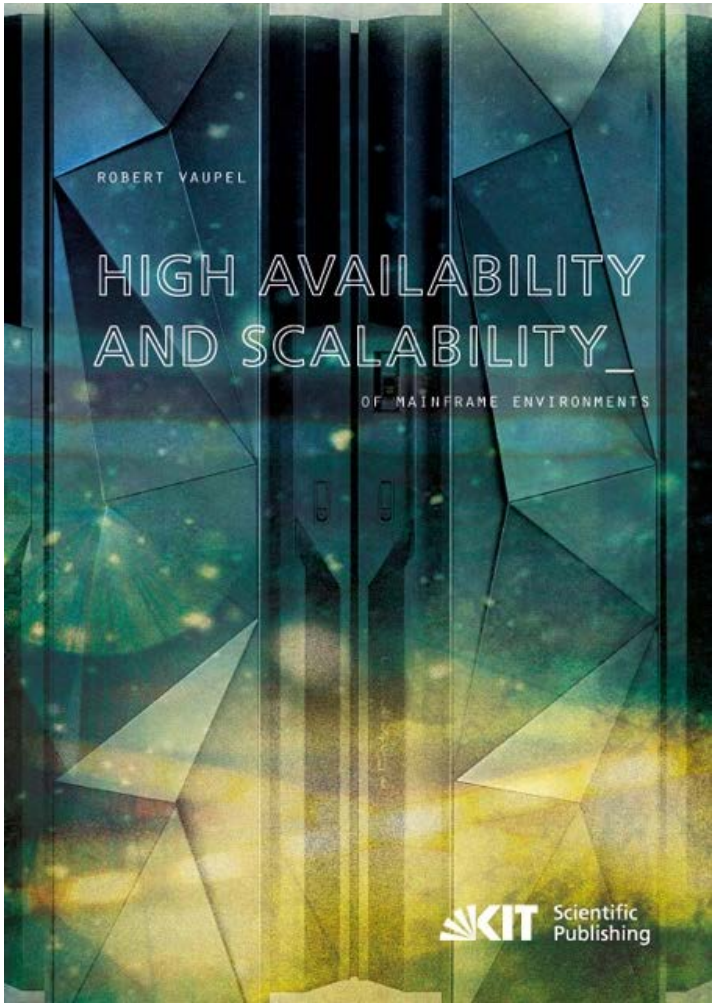
ftp://public.dhe.ibm.com/eserver/zseries/zos/wlm/zWLM_V2.pdf



ftp://public.dhe.ibm.com/eserver/zseries/zos/wlm/WLMcicsims.pdf



z/OS Books



http://digbib.ubka.uni-karlsruhe.de/volltexte/1000034624