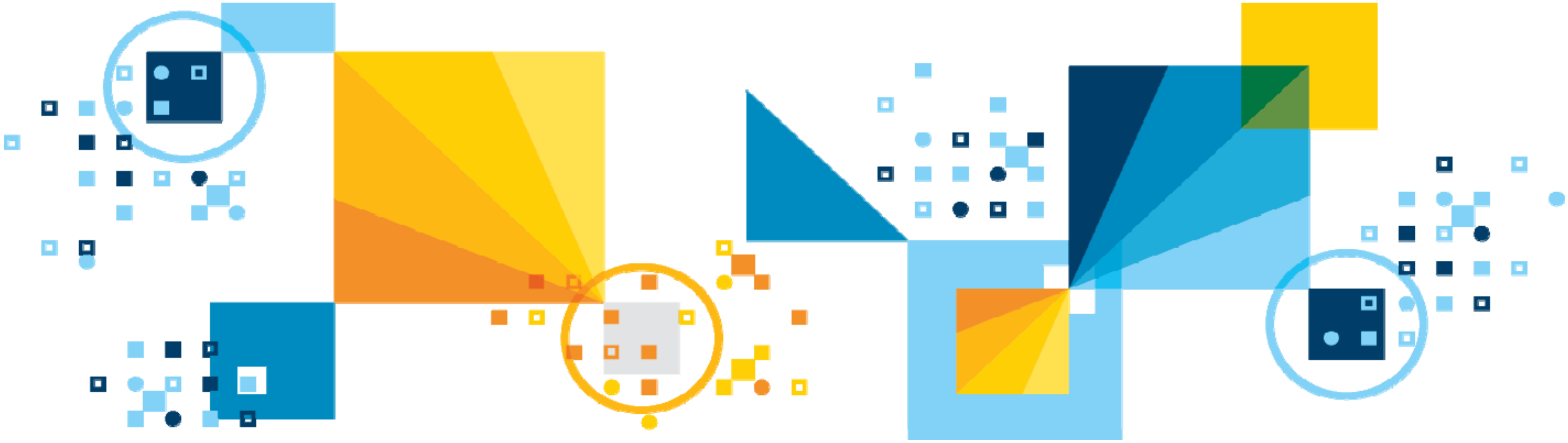
DB2 Workload Management

Module ID	10116
Length	1 hour + 1 hour Hands on Lab



For questions about this presentation contact askdata@ca.ibm.com

January 29, 2015

Disclaimer

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM’S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS AND/OR SOFTWARE.

IBM, the IBM logo, ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at www.ibm.com/legal/copytrade.shtml

Other company, product, or service names may be trademarks or service marks of others.

Module Information

- You should have completed or acquired the necessary knowledge for the following modules in order to complete this module:
  - DB2 Fundamentals

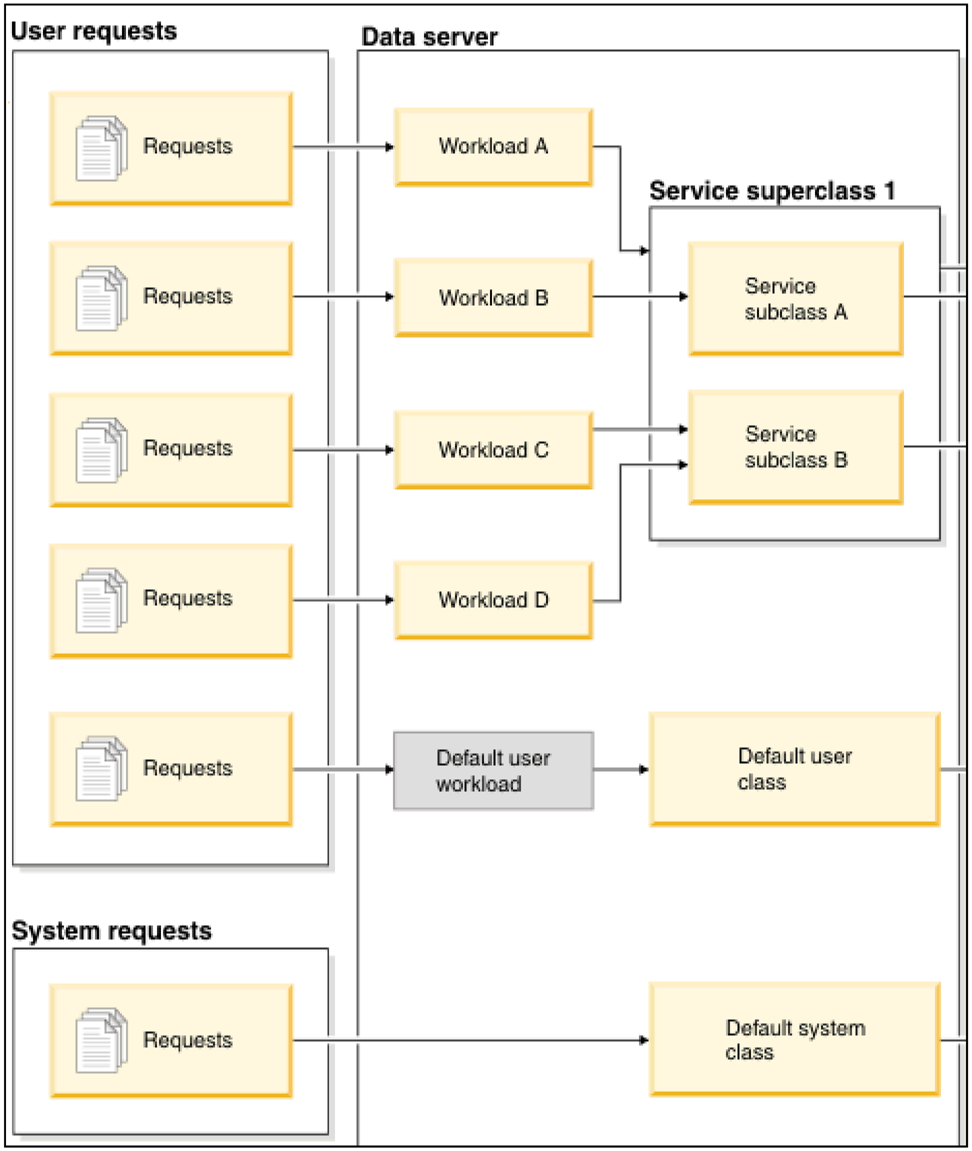
- After completing this module, you should be able to:
  - Explain the concept of:
    - WLM components
    - WLM Dispatcher
    - Data tag

Module Content

- **Workload Management (WLM) components**
  - Work class and work class sets
  - Service classes
  - Workloads
  - Work action and work action sets
  - Thresholds
- **WLM Dispatcher**
- **WLM monitoring scenarios**
- **Monitoring capabilities**
- **Data tags**
- **Automatic Workload Management in DB2 10.5**

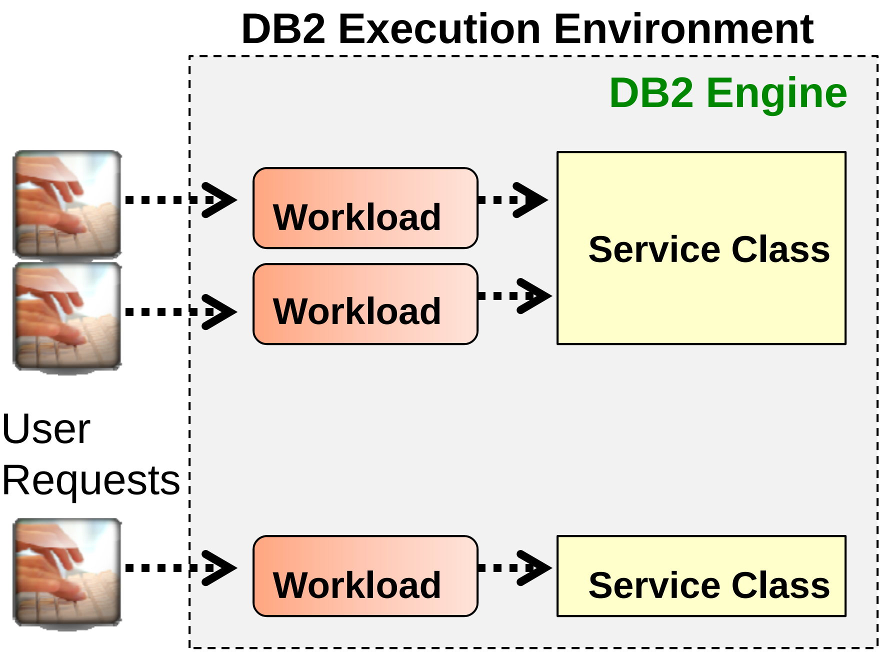
Workload Management in DB2

- Stable and predictable execution environment
- Light-weight and granular
- Better resource management
  - Explicitly allocate resources amongst work
  - Limit excessive or unexpected resource consumption
- Better request management
  - Prioritize and track performance of work



Workload Management Terminology

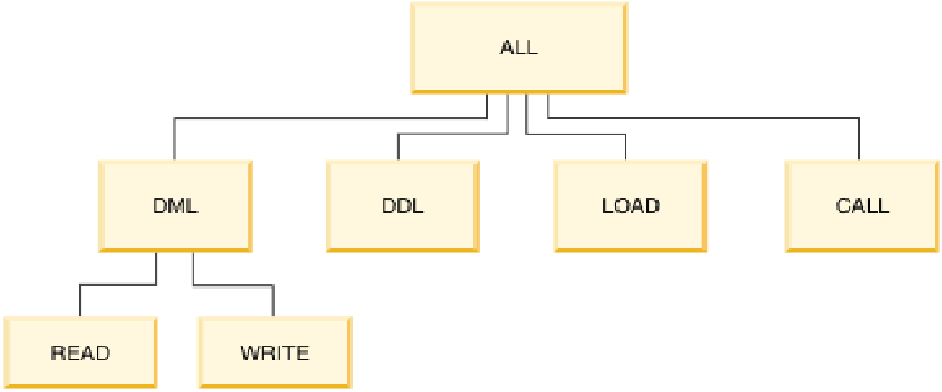
- **Work Action Sets**
  - Identification and management of activities based on the type of work
- **Service classes**
  - Single point of execution, monitoring, and control for work running on a data server
- **Workload**
  - Identification of work to be assigned to a service class
- **Thresholds**
  - Exception events that trigger an action for work on the data server
- **Monitoring**
  - Light weight methods for determining workload status, trends and problem determination



DB2 Work Class and Work Class Set

- **Work class**
  - Identify activities based on **type of work**
  - Provide information about database activities that can be used to **take action before consumption** of resources on the data server

Types of Work

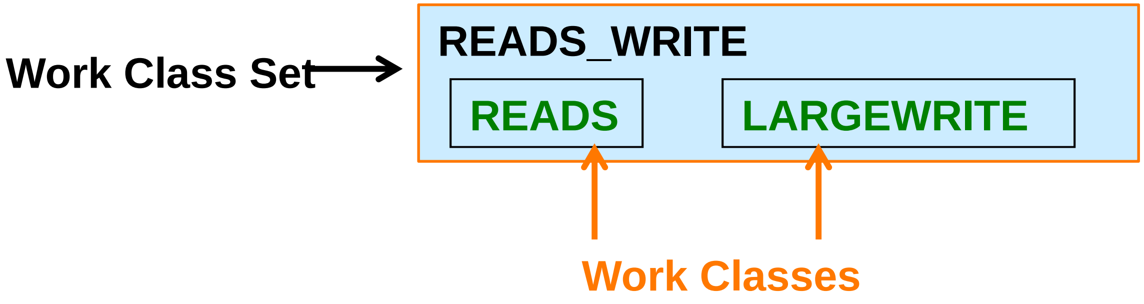


- Grouped into a **work class set**

```
CREATE WORK CLASS SET “READS_WRITES”
```

```
ALTER WORK CLASS SET “READS_WRITES” ADD WORK CLASS “READS” WORK TYPE READ;

ALTER WORK CLASS SET “READS_WRITES” ADD WORK CLASS “LARGEWRITE”
WORK TYPE WRITE FOR TIMERONCOST FROM 10000 to UNBOUNDED;
```

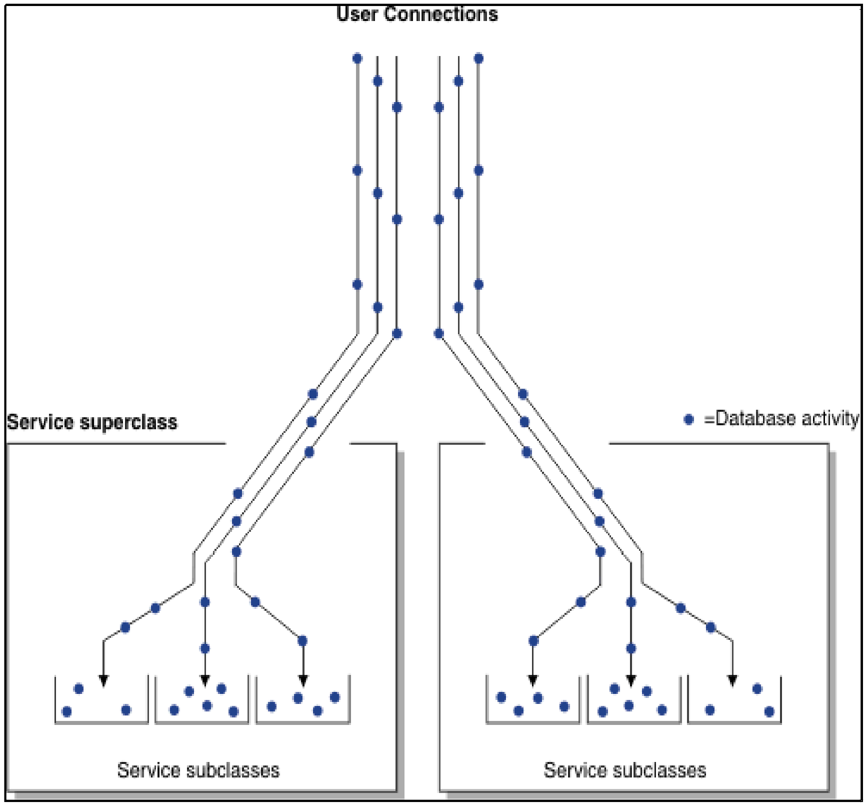


DB2 Service Class

- All database work executed in the context of a specific DB2 service class
  - Independent execution environment for both user and DB2 initiated work
- Supports a **two-tier hierarchy**
  - **Superclass**
    - Logical entity providing common attributes across subclasses

```
CREATE SERVICE CLASS 'MARKETING'
```

  - **Subclass**
    - All work executes in sub classes, one automatically created during superclass creation



```
CREATE SERVICE CLASS 'Americas' UNDER 'MARKETING'
```

DB2 Service Class Resource Controls

- **Explicitly control resources available to a service class**

Priority	Controls	Values
Prefetch (I/O)	Order in which agent prefetch requests are processed	**High, Medium, or Low**
Bufferpool (I/O)	High priority pages are less likely to be swapped out of the buffer pool than pages fetched with lower priority.	**High, Medium, or Low**

⚠ **Agent Priority (CPU) has been deprecated**

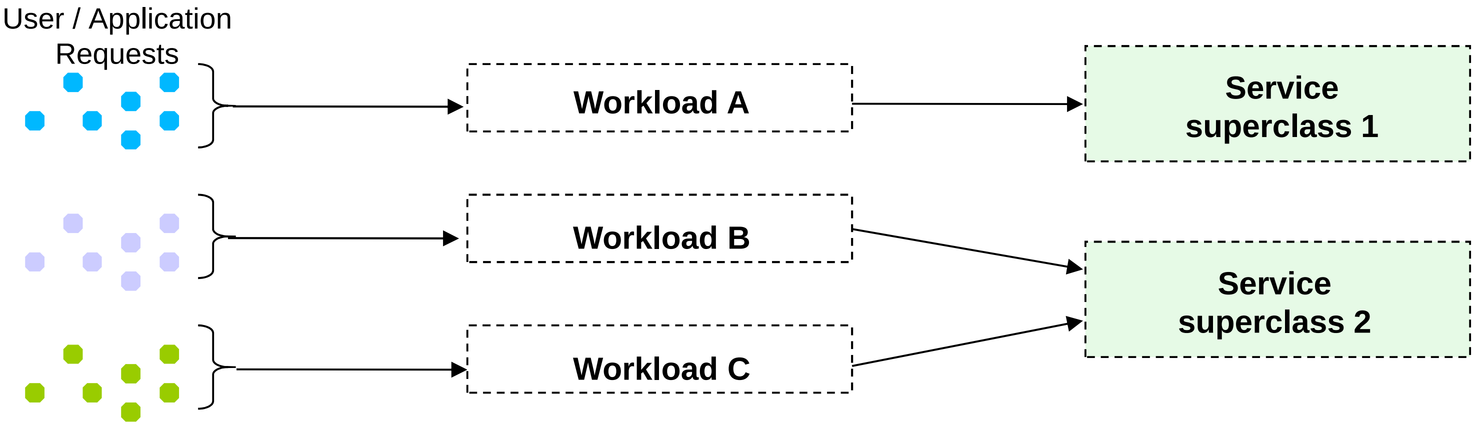
```
CREATE SERVICE CLASS ‘MARKETING’ PREFETCH PRIORITY HIGH
```

```
ALTER SERVICE CLASS ‘MARKETING’ BUFFERPOOL PRIORITY HIGH
```

DB2 Workload

- **Serves to route** incoming work to a DB2 service class
  - Identify the sources and submitters of requests from database connection properties
  - Connections are mapped to a specific workload when established
    - Re-evaluated at unit of work boundaries as required

- All database connections map to a specific workload definition



- **Grant or revoke USAGE** privilege on a defined workload (uses connection authorization ID)

- **Identify workloads** using:

  Authorization ID: System, session, group of session, and role of session

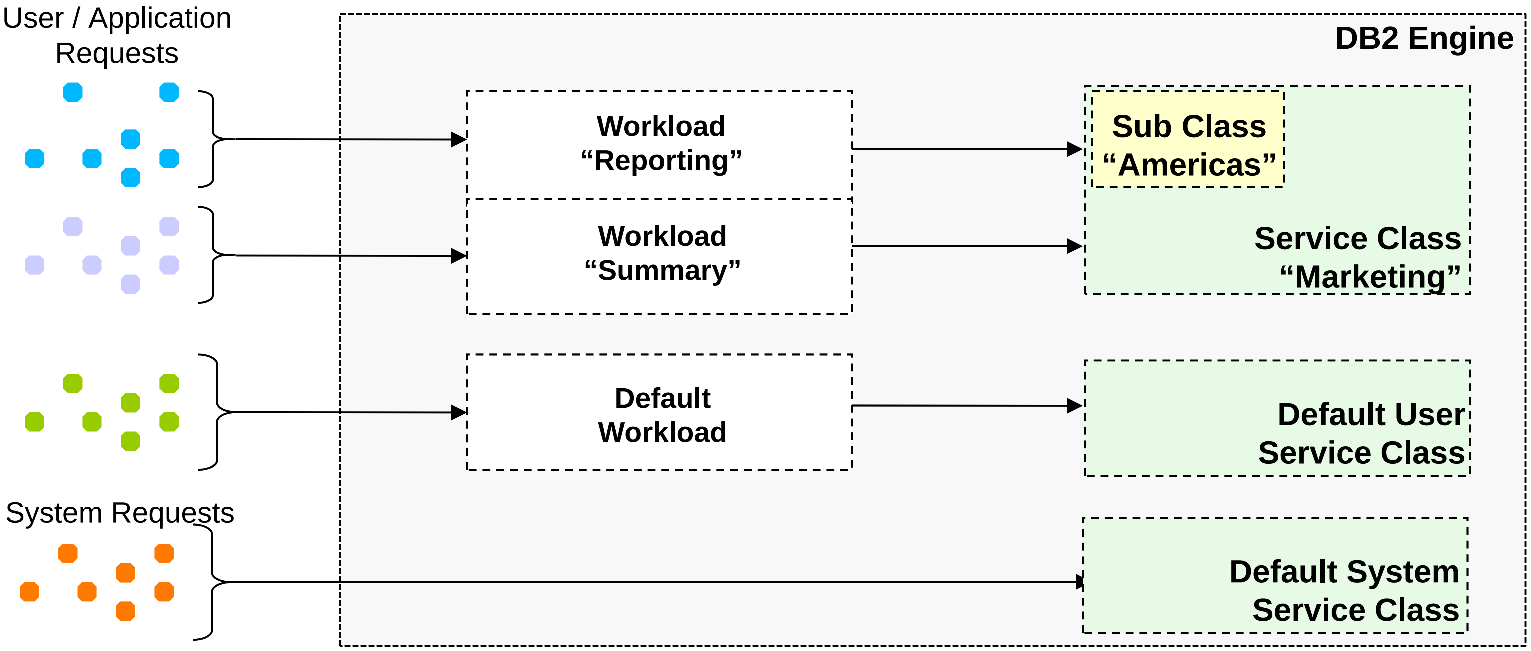
  Names: Client application, Client workstation, Application, and Client accounting string

Defining a DB2 Workload

- **Definition order matters**
  – DB2 will use first rule that matches

```
CREATE WORKLOAD “Summary” SESSION_USER_GROUP(‘Deptmgr’)
APPLNAME (‘sas’) SERVICE CLASS ‘MARKETING’
```

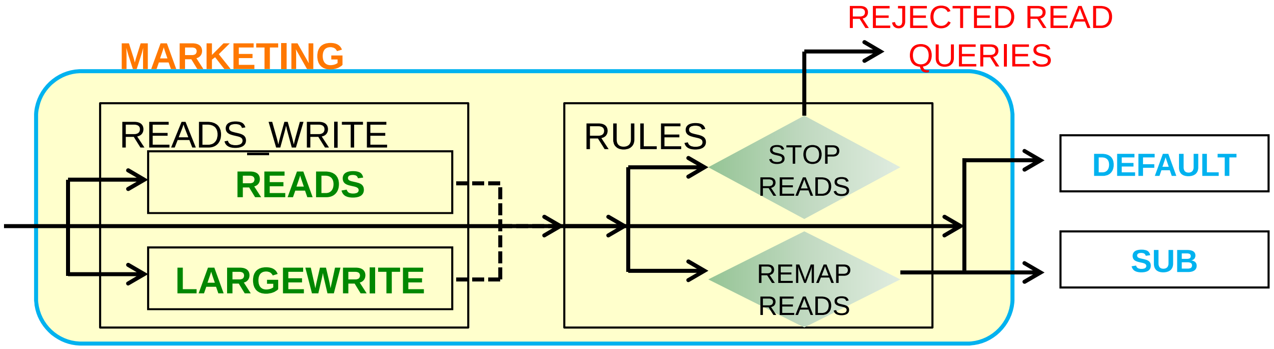
```
CREATE WORKLOAD “Reporting” APPLNAME (‘sas’) SERVICE CLASS
‘MARKETING’
```


User / Application Requests
DB2 Engine
Workload “Reporting”
Workload “Summary”
Default Workload
Sub Class “Americas”
Service Class “Marketing”
Default User Service Class
System Requests
Default System Service Class

DB2 Work Action and Work Action Set

- Exercise control over different types of activities in conjunction with a work class
  - Applied to activities in either a specific **workload**, **superclass**, or to the **database** based on type or size of work
  - There must be a work action defined for the work class to be active and have activities assigned to it

- Actions:
  - **Threshold definitions**
  - **Prevent execution**
  - **Collect activity data**
  - **Count activity**

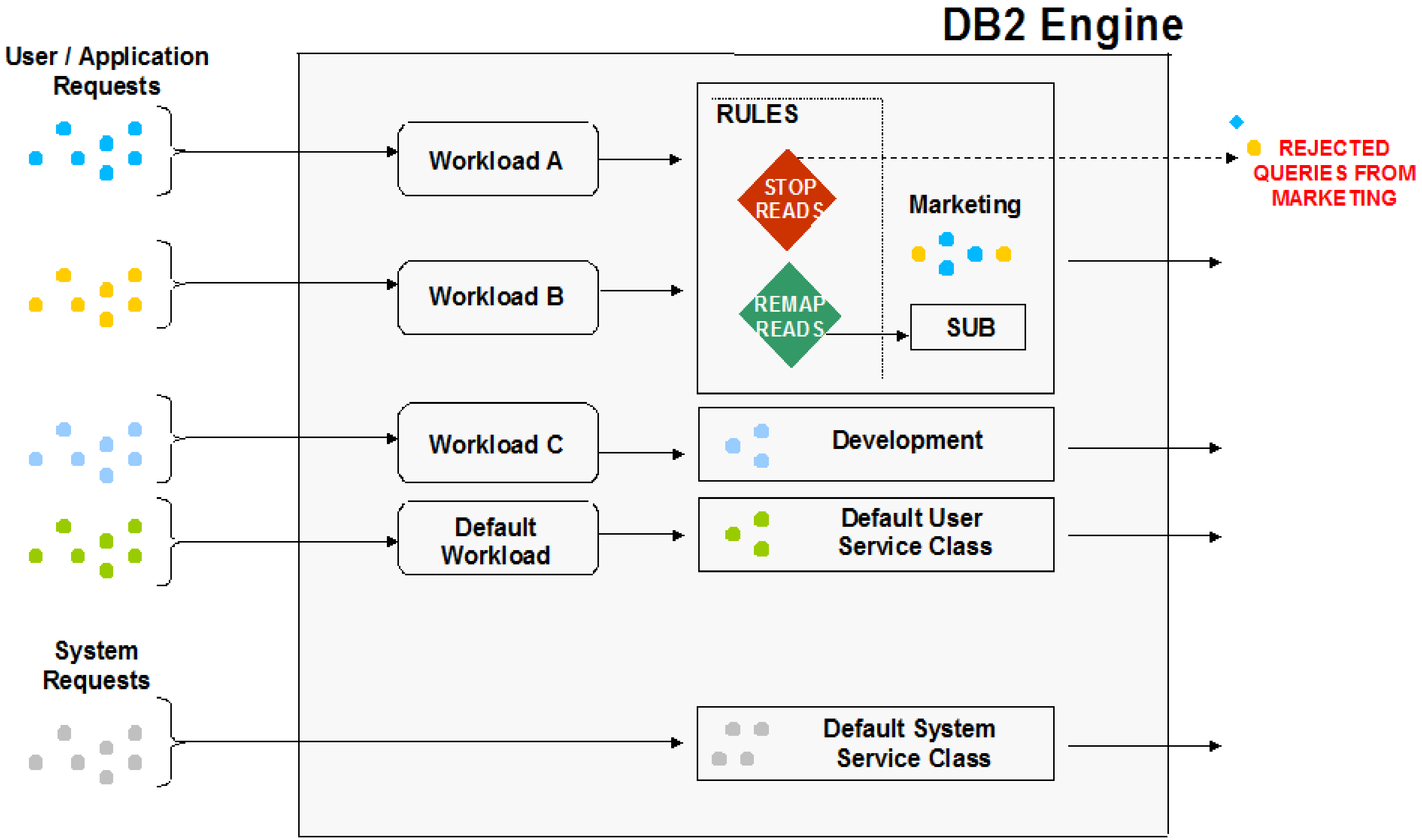


```
CREATE WORK ACTION SET “RULES” FOR SERVICE CLASS “MARKETING” USING WORK CLASS
SET “READS_WRITES”
```

```
ALTER WORK ACTION SET “RULES” ADD WORK ACTION “STOPREADS” ON WORK CLASS
“READS” PREVENT EXECUTION ENABLE;

ALTER WORK ACTION SET “RULES” ADD WORK ACTION “REMAPREADS” ON WORK CLASS
“LARGEWRITE” MAP ACTIVITIES TO “SUB”;
```

Mapping Concept with Everything Combined



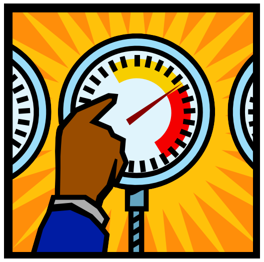
Thresholds

- **Automated enforcement of rules or establishment of limits for activities**
  - Based on predefined elements
  - May be defined at one or more levels
    - Database, service class, work action, workload, statement

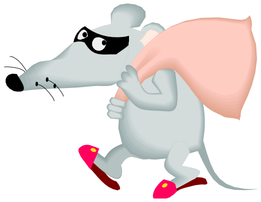
Scope	Thresholds	Enforcement
Database	ACTIVITYTOTALTIME	Max time spent processing an activity
	CONCURRENTDBCOORDACTIVITIES	Max number of concurrent coordinator activities
	CONNECTIONIDLETIME	Max time a connection can be idle
	ESTIMATEDSQLCOST	Max estimated cost
	SQLROWSRETURNED	Max rows returned
	UOWTOTALTIME	Max time spent on a unit of work
Member	AGGSQLTEMPSPACE	Max amount of system temporary table space in a subclass
	CPUTIME	Max combined processor time in a partition
	CONCURRENTWORKLOADOCCURENCES	Max number of workload occurrences
	CONCURRENTDBCOORDACTIVITIES	Max number of concurrent coordinator activities
	CPUTIMEINSC	Max combined processor time for a service class
	DATATAGINSC	Check data tag value
	SQLROWSREAD	Max rows read in a partition
	SQLROWSREADINSC	Max rows read for a service class
	SQLTEMPSPACE	Max amount of system temporary table space
	TOTALMEMBERCONNECTIONS	Max number of concurrent connections
	TOTALSCMEMBERCONNECTIONS	Max number of concurrent connections for a superclass
Workload	CONCURRENTWORKLOADACTIVITIES	Max number of coordinator and nested activities

Violations

- A violation is when a threshold is violated
  - An event is written to the threshold violation event monitor
  - DB2 will automatically take requested action:
    - **STOP EXECUTION**
    - **CONTINUE**
    - **FORCE APPLICATION**
    - **REMAP ACTIVITY TO**
    - **COLLECT ACTIVITY DATA**
- **WLM_CANCEL_ACTIVITY ()** stored procedure cancels an activity directly
  - Returns SQL4752N to application for successfully cancelled statement



Rogue Queries



- **Rogue queries can be monitored and investigated using thresholds**
- **Predictive thresholds** are determined prior to execution
  - E.g. Estimated cost
- **Reactive thresholds** are used to control during execution
  - E.g. Activity total time
- Using these, behavioral norms can be set
  - Enable proactive detection and control of problem queries
  - Information about the activity that violated the threshold is sent to the active **ACTIVITIES** event monitor when the activity completes.

```
CREATE THRESHOLD BIGQUERIES
FOR SERVICE CLASS BIGQUERIES ACTIVITIES ENFORCEMENT DATABASE
WHEN ACTIVITYTOTALTIME > 10 MINUTES
COLLECT ACTIVITY DATA WITH DETAILS AND VALUES
CONTINUE
```

Concurrency Control

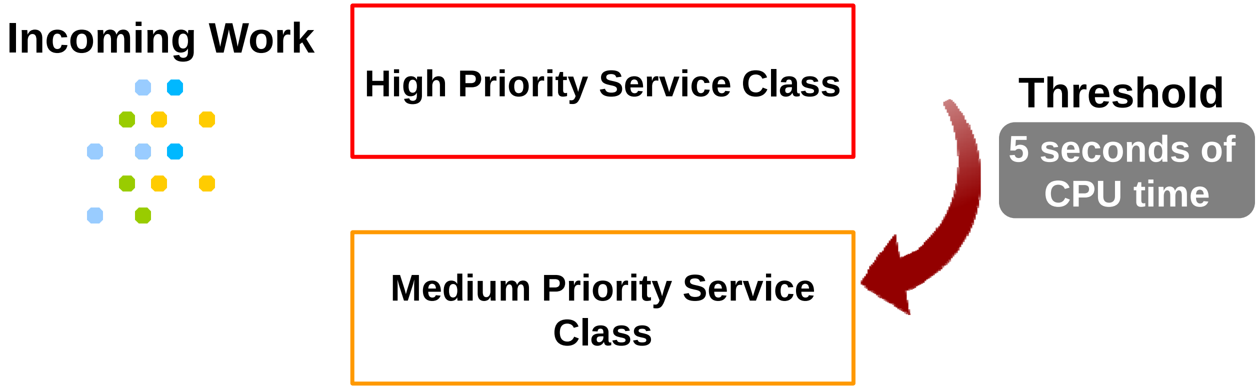
- Limit and control the maximum number of activities allowed to execute at one time
  - Limit number allowed to execute concurrently as well as how many allowed to wait
- Control surges and unexpected peaks in the active workload
- Control competing workloads sharing execution environments
- Control concurrency rates for disruptive activities

```
CREATE THRESHOLD MAXACTIVITIESINPAYROLL
FOR WORKLOAD PAYROLL ACTIVITIES
ENFORCEMENT WORKLOAD OCCURRENCE WHEN
CONCURRENTWORKLOADACTIVITIES > 10
STOP EXECUTION
```

Remapping an Activity

- Re-prioritization for queries which run longer or require more resources
  - Ability to remap an activity from subclass to another within the same superclass

```
CREATE THRESHOLD MAXACPUTIME
FOR SERVICE CLASS HIGHPRIORITY
ACTIVITIES ENFORCEMENT DATABASE WHEN CPUTIMEINSC > 5
REMAP ACTIVITY TO MEDIUMPRIORITY
```



Incoming Work

High Priority Service Class

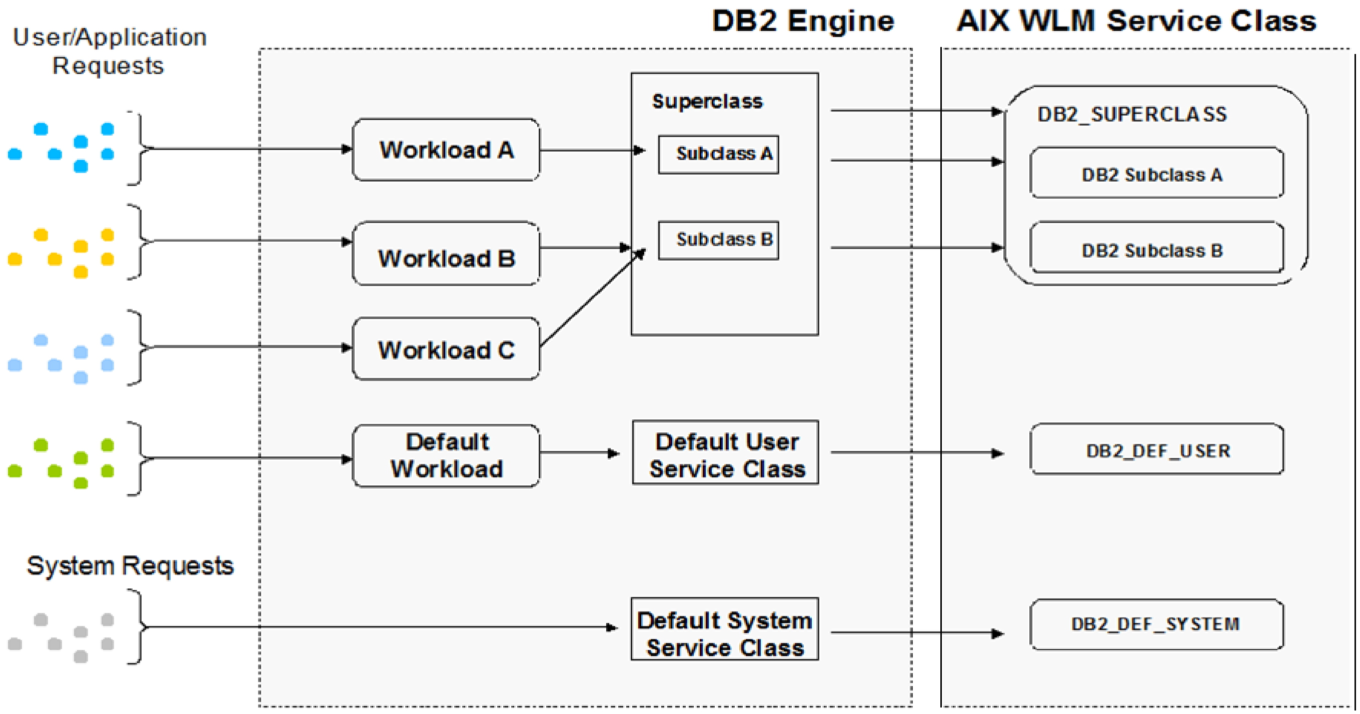
Medium Priority Service Class

Threshold

5 seconds of CPU time

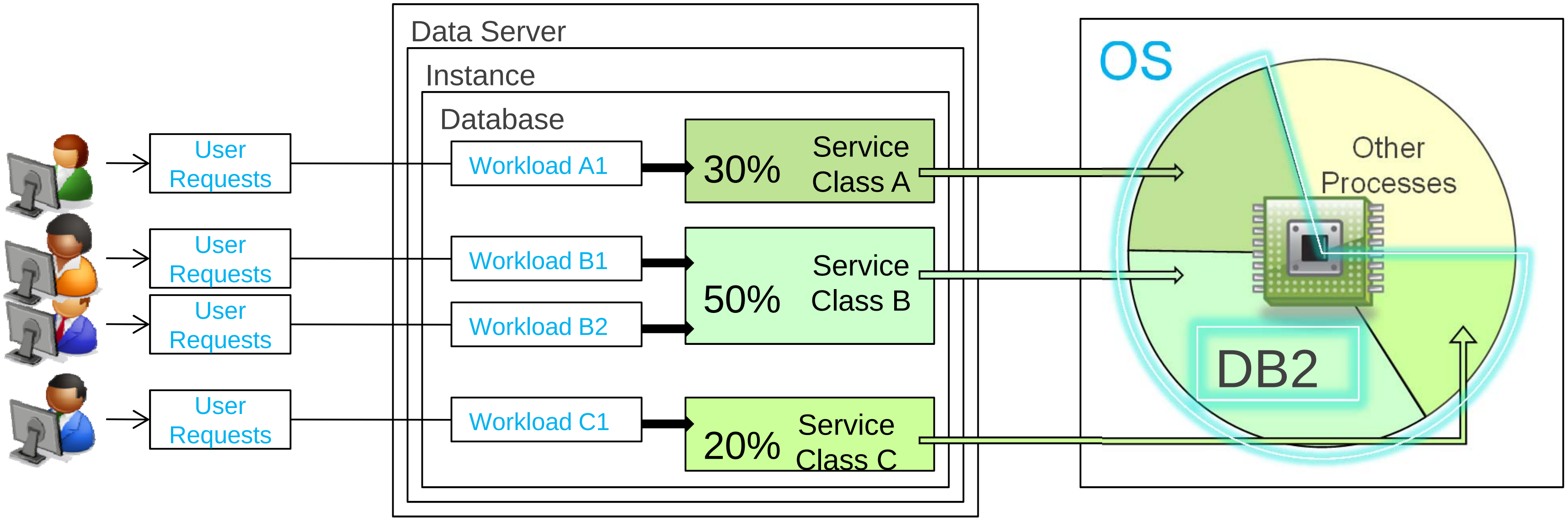
Integration with AIX or Linux

- Optional ability to tightly integrate DB2 processing with the underlying operating system’s workload management
  - AIX
  - Linux
    - Kernel version 2.6.26 or higher, 64-bit and libcgroup library package



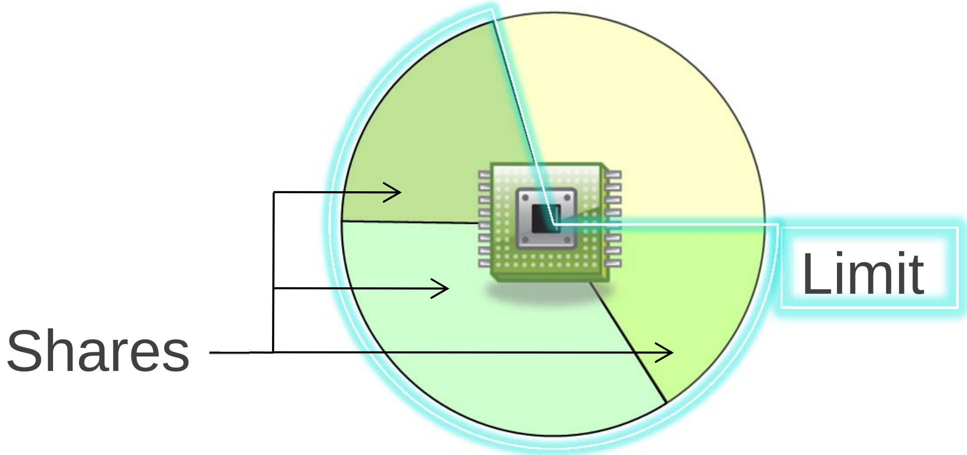
WLM Dispatcher

- IBM DB2 9.7 allowed integration with OS WLM to provide additional control over resources, in particular, the amount of CPU resources allocated to each service class.
  - Only supported on specific platforms
  - DBAs would require extra knowledge and privileges to set up the OS WLM
- **WLM Dispatcher** is a CPU scheduling technology in the DB2 engine at instance level
  - Allows for fine allocation of CPU amongst DB2 work executing in user and maintenance service classes
  - It works in parallel with OS WLM mechanisms
  - Supported for all DB2 LUW platforms
  - Provides additional flexibility with use of hard and soft CPU shares, and CPU limits



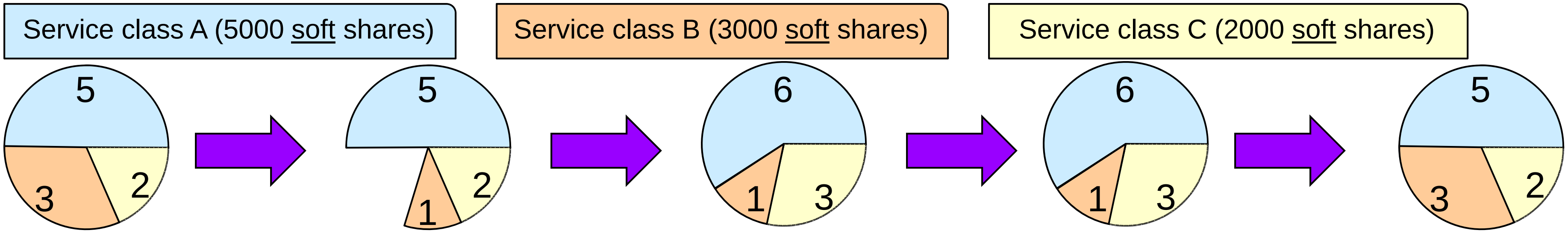
WLM Dispatcher

- Dispatcher controls CPU usage through shares and limits
- **CPU limit**
  - Maximum percentage of absolute system CPU resources that the service class can consume
- **Shares**
  - Portion of CPU usage allocated to a service class
  - By default a service class is assigned 1000 soft shares
  - **Soft CPU Shares**
    - Imposes a soft limit (i.e. can use more) when there is competition for CPU use
  - **Hard CPU shares**
    - Imposes a hard limit when there is competition for CPU use



Soft CPU Shares

- Service class shares policy attribute = SOFT



Service class A (5000 soft shares)

Service class B (3000 soft shares)

Service class C (2000 soft shares)

All service classes using their maximum amount of shares

Service class B stops using its full share (using only 1)

Service class A and C can now utilize the additional resources

Service class B requires all the shares again

Service class B uses its shares once A and C are finished

```
create service class A soft cpu shares 5000
create service class B soft cpu shares 3000
create service class C soft cpu shares 2000
```

Shares policy attribute

Number of shares

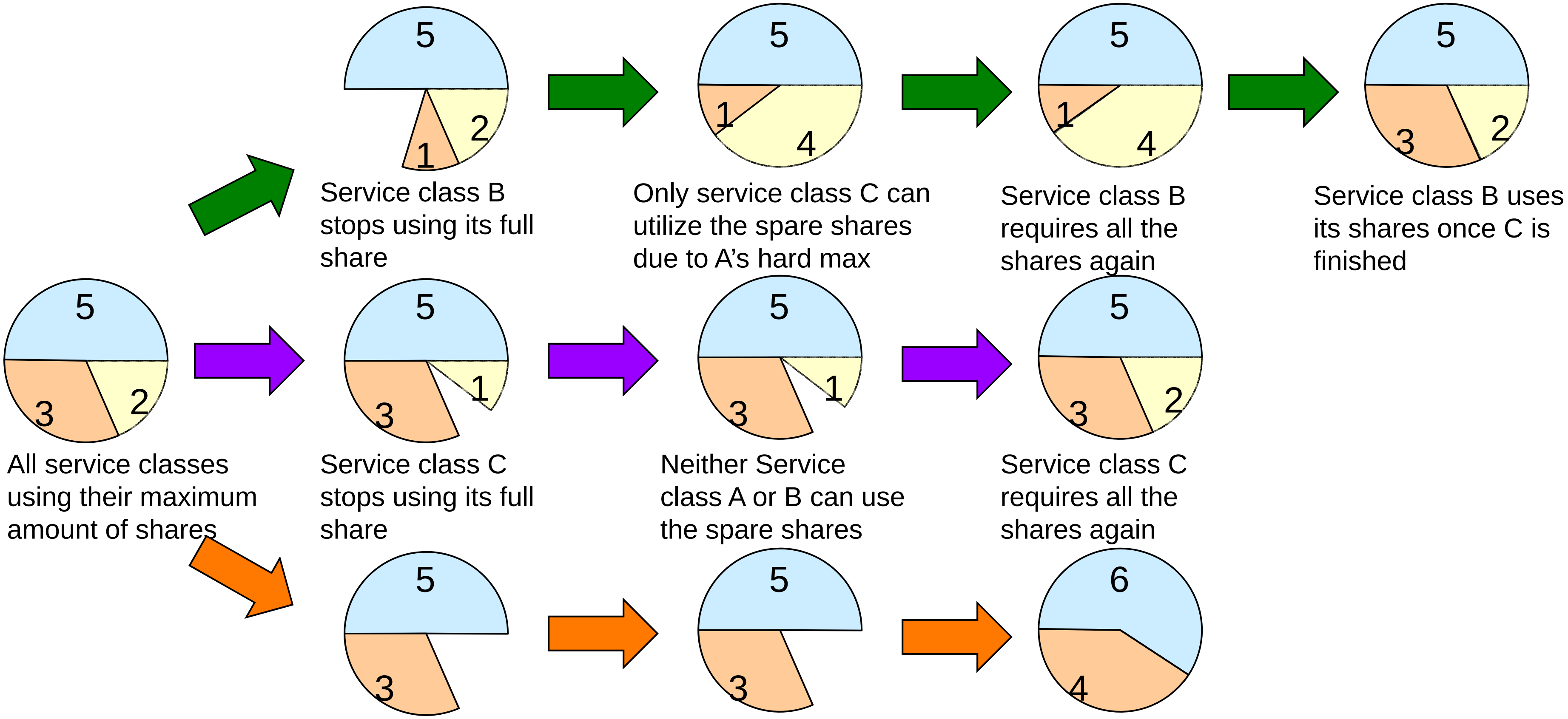
Hard CPU Shares

- Service class shares policy attribute = HARD

Service class A (5000 HARD shares)

Service class B (3000 HARD shares)

Service class C (2000 SOFT shares)



All service classes using their maximum amount of shares

Service class B stops using its full share

Only service class C can utilize the spare shares due to A's hard max

Service class B requires all the shares again

Service class B uses its shares once C is finished

Service class C stops using its full share

Neither Service class A or B can use the spare shares

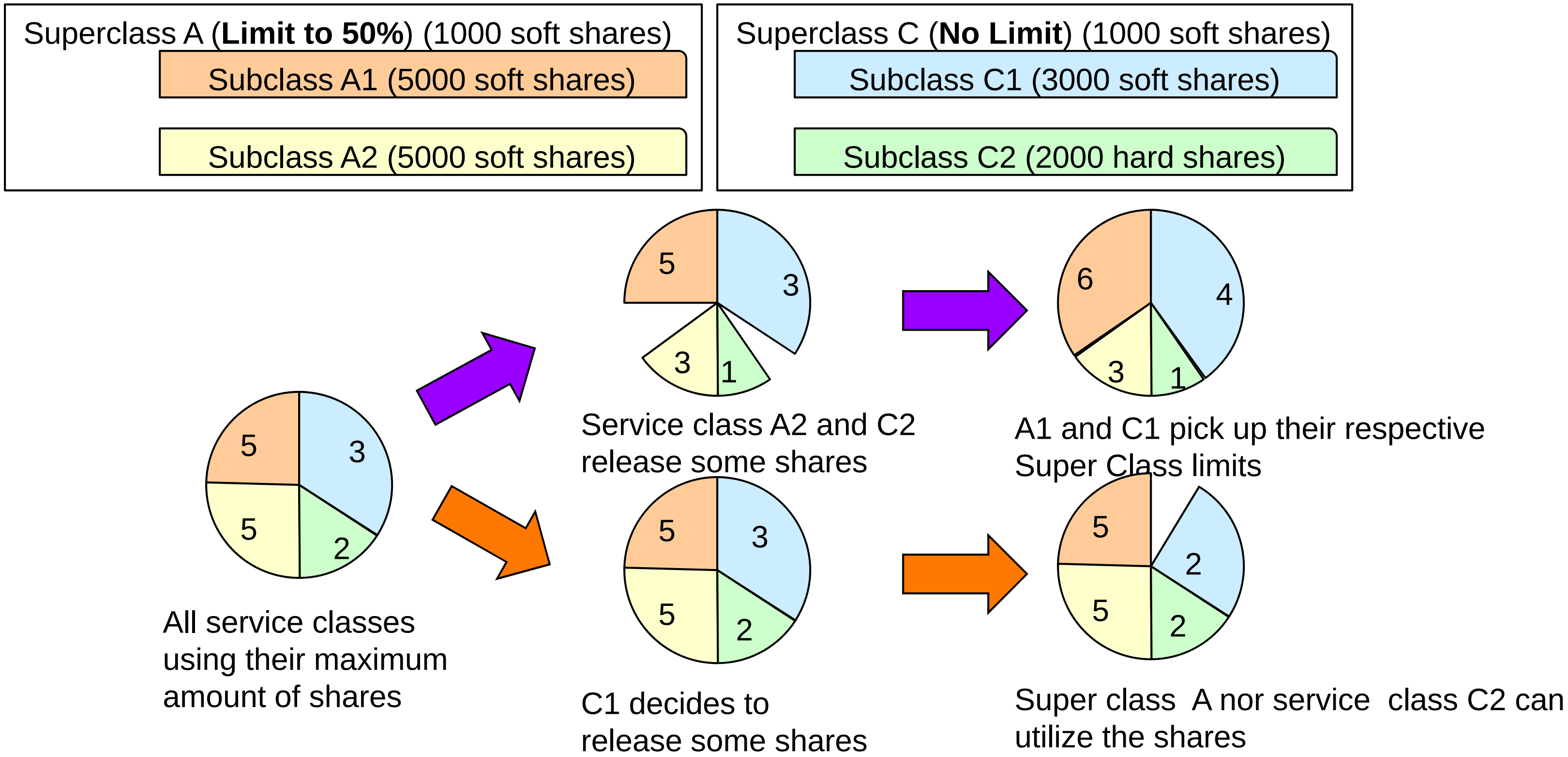
Service class C requires all the shares again

Service class C stops using any of its shares

Since C is no longer a competing service class A and B can use the spare shares

CPU Limits

- CPU Limits allow control of absolute CPU consumption
- More restrictive than CPU Shares



Superclass A (**Limit to 50%**) (1000 soft shares)

Subclass A1 (5000 soft shares)

Subclass A2 (5000 soft shares)

Superclass C (**No Limit**) (1000 soft shares)

Subclass C1 (3000 soft shares)

Subclass C2 (2000 hard shares)

All service classes using their maximum amount of shares

Service class A2 and C2 release some shares

A1 and C1 pick up their respective Super Class limits

C1 decides to release some shares

Super class A nor service class C2 can utilize the shares

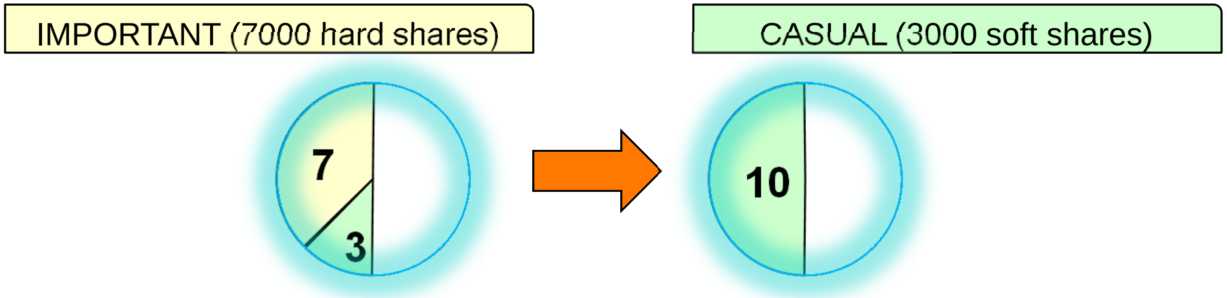
```
create service class A cpu limit 50
create service class C cpu limit NONE
```

Percentage of CPU

Scenario: Prioritizing a workload while setting CPU usage limits

- **Service Class “IMPORTANT”**: High Priority, workload is sporadic
  - This workload is favored but a small amount of resource are reserved to CASUAL
- **Service Class “CASUAL”**: Lower Priority, workload is constant
  - Not very privileged, but it should be able to take advantage of unused CPU resources
- Important and Casual Daily tasks should only use a maximum of 50% of CPU resources available

```
db2 update dbm cfg using wlm_dispatcher YES
db2 update dbm cfg using wlm_disp_cpu_shares YES
create service class DAILY cpu limit 50
create service class IMPORTANT under DAILY HARD cpu shares 7000
create service class CASUAL under DAILY SOFT cpu shares 3000
```



Both service classes using to their maximum allocation

If “IMPORTANT” has no usage, “CASUAL” can pick up unused portions – limited to 50% of available CPU

Monitoring

- Collect aggregate information
  - “Push” and “Pull” approaches
  - Activities, service classes, workloads, work classes, threshold queues, and threshold violations
- **Table functions**
  - Real-time monitoring
  - Problem determination
  - Performance tuning
- **Event monitors**
  - Historical monitoring
  - Statistics
  - Threshold violations
  - Individual Activities
- **Stored Procedures**
  - Cancel activities
  - Capture details of an activity
  - Reset statistics

Table Functions for Monitoring

- Real-time monitoring for statistics, metrics and information of current work on the system
- All table functions starting with “WLM_” are used for WLM monitoring
  - E.g. **WLM_**GET_WORKLOAD_OCCURRENCE_ACTIVITIES
  - More statistical
- Monitoring metrics functions start with “MON_”
  - E.g. **MON_**GET_ACTIVITY_DETAILS
  - Complete set of raw monitoring data
- Pushing information
  - Light-weight
  - Granular control
  - Fast access with minimal impact

Event Monitors

- Capture point-in-time information on events on the system

Monitor Name	Monitor Trigger
Locking	Locks and deadlocks
Activities	Database activity (execution of SQL or other operations)
Statements	Execution of SQL
Unit of work	Completion of a Unit of work (transaction)
Package cache	Entries evicted from the package cache
Connections	Applications connecting to the database
Database	Database deactivation
Buffer pool and table spaces	Database deactivation
Tables	Database deactivation
Statistics	Can be triggered at regular intervals or manually
Threshold violations	Violation to a threshold
Change history *NEW*	Parameter of configuration, DDL, command or utility completes

Stored Procedures

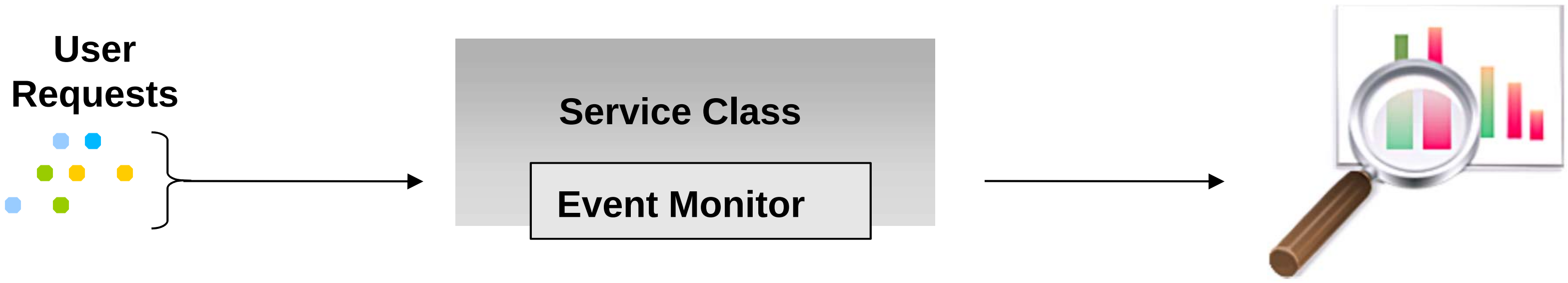
- Stored Procedures for WLM:
  - **WLM_CANCEL_ACTIVITY ()**
    - Cancel a running or queued activity
  - **WLM_CAPTURE_ACTIVITY_IN_PROGRESS ()**
    - Immediately send the information of an activity to the event monitor rather than waiting for the activity to complete
  - **WLM_COLLECT_STATS ()**
    - Collect and reset statistics for a WLM object
  - **WLM_SET_CLIENT_INFO ()**
    - Set connection info of application or end-user

Histograms

- A collection of discrete ranges of data

Type	Description
CoordActQueueTime	Time non-nested activities spent queued
CoordActExecTime	Time non-nested activities spent executing
CoordActLifeTime	Time non-nested activities are identified to execution completion
CoordActInterArrivalTime	Time interval between non-nested activity arrivals
CoordActEstCost	Estimated cost of non-nested activities
ReqExecTime	Request execution times
UowLifeTime	Time from when unit of work is identified to execution completion

- Aggregate information pulled via automated collection
  - Available for service subclasses, workloads, and work classes



User Requests

Service Class

Event Monitor

Multi-temperature Storage Integrates with DB2 WLM

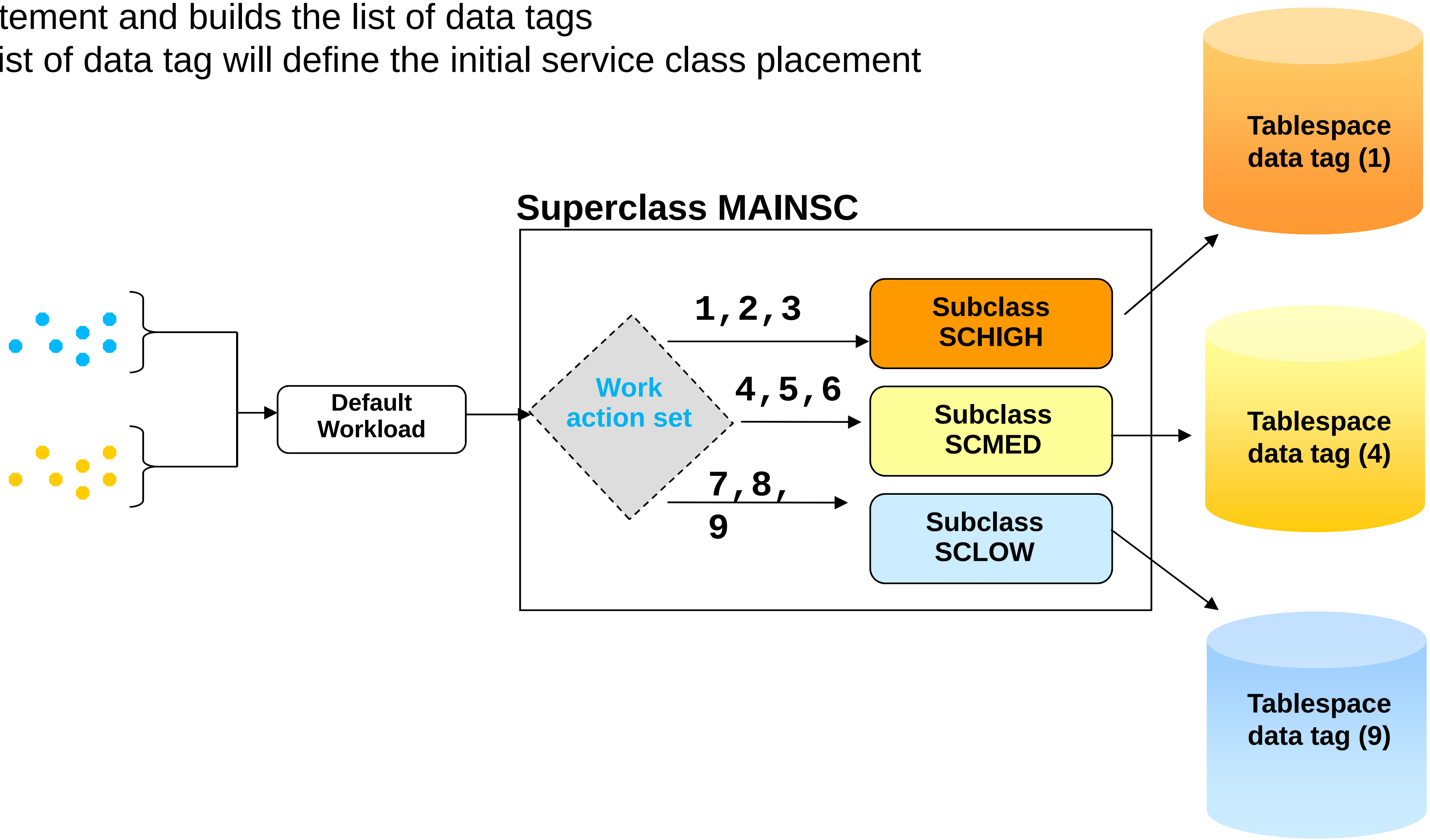
- Existing WLM perspectives are user-centric (who) and request-centric (what)
- Introducing a new perspective "**data-centric**" (where)
  - New **data tag** attribute
    - For storage group or table space
    - Priority can be given to requests based on what data is accessed [Values 1 (high) – 9 (low)]
- WLM work class and threshold DDL have been extended to support the **new data tag attribute**
- Work class sets are *predictive* based on compilation information
  - Sometimes there isn't enough information at compile time (e.g. queries with parameter markers) to predict which table spaces will be touched
- Data tag thresholds are *reactive* and use information that is available at runtime

Data Tag Attribute Used in DB2 WLM

- Data-centric perspective is used in two ways
  - **Predictive**
    - Create the work class & work class set to differentiate the priority of the data based on a **“data tag”**
    - Create work action & work action set to redirect the work class to the service subclass desired
  - **Reactive**
    - New threshold **DATATAGINSC**
    - You can prioritize activities based on what data the activity accesses during activity execution using the threshold value.

Using Data tags in DB2 WLM: Predictive

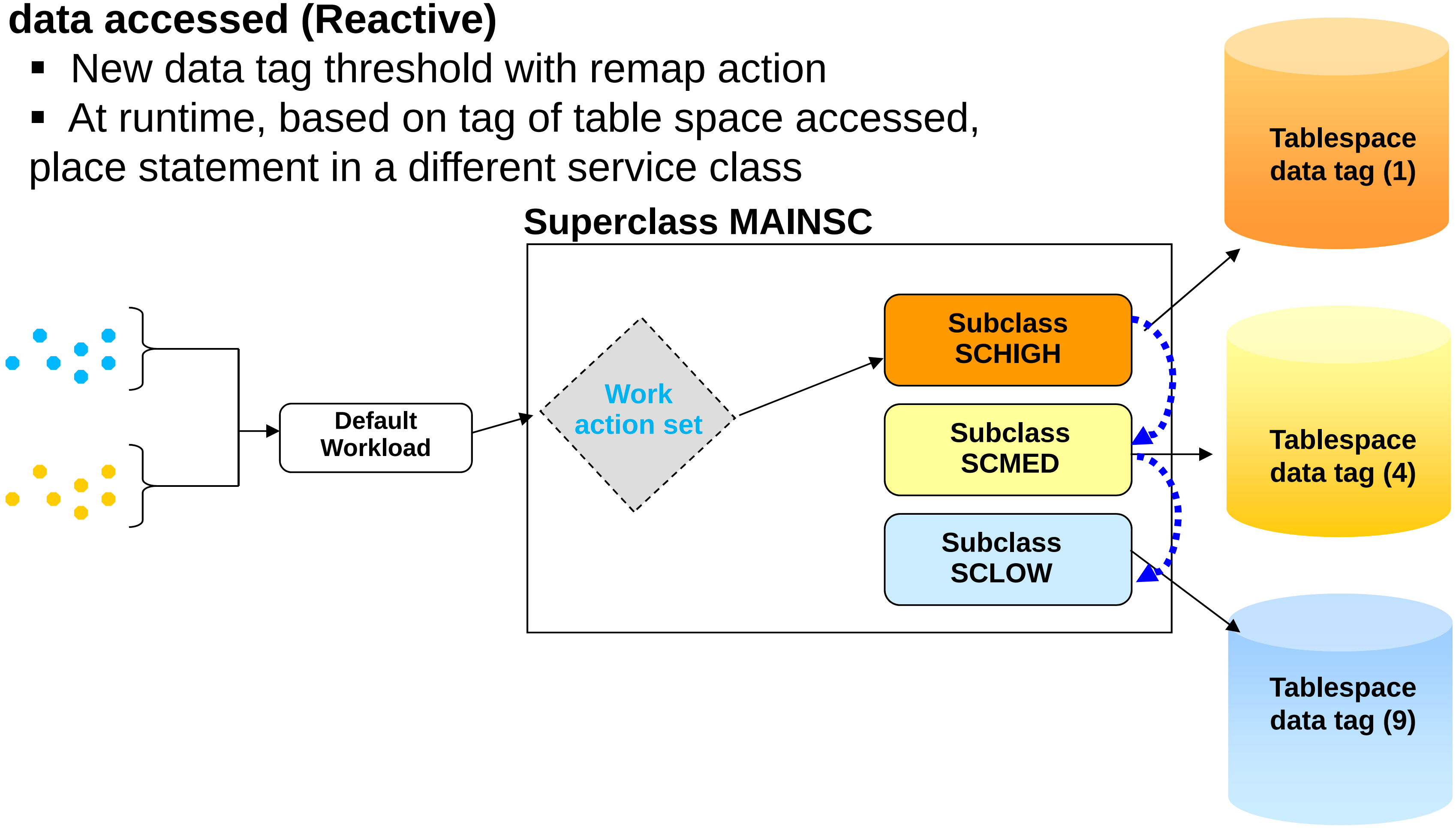
- **Map statement to service class (Predictive)**
- SQL compiler predicts what data (and table spaces) will be touched by the SQL statement and builds the list of data tags
- List of data tag will define the initial service class placement



Using Data tags in DB2 WLM: Reactive

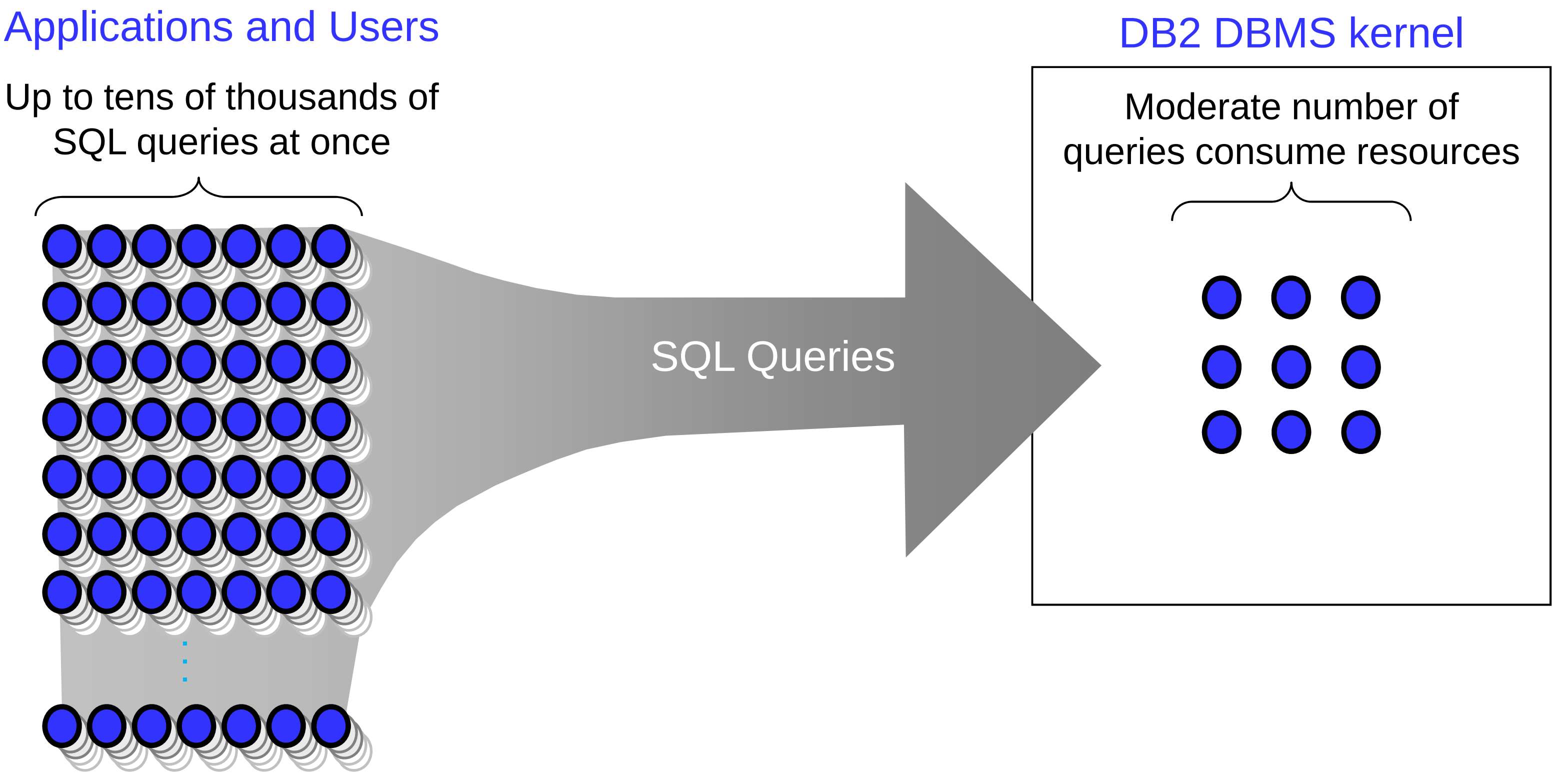
Allows changing priority of workload at runtime based on data accessed (Reactive)

- New data tag threshold with remap action
- At runtime, based on tag of table space accessed, place statement in a different service class



Automatic Workload Management

- Built-in and automated query resource consumption control
- Enabled automatically when `DB2_WORKLOAD=ANALYTICS`
- Many queries can be submitted, but limited number get executed concurrently


Applications and Users
Up to tens of thousands of SQL queries at once
SQL Queries
DB2 DBMS kernel
Moderate number of queries consume resources

New Workload Manager (WLM) Objects

- Created when new db created or old db migrated
- SYSDEFAULTMANAGEDSUBCLASS: service subclass for heavy queries
- SYSDEFAULTCONCURRENT: CONCURRENTDBCOORDACTIVITIES threshold for heavy queries
  - Default concurrency threshold computed based on the hardware
  - Current concurrency threshold in SYSCAT.THRESHOLDS
  - Enabled by default when DB2_WORKLOAD=ANALYTICS and
    - New db is created, OR
    - DB2 Configuration Advisor is run for a migrated old db
- SYSDEFAULTUSERWAS: Work action set to map queries
  - Heavy queries: read-only and timeron cost exceeds threshold
  - Default queries: all other queries

Workload Manager

- Map all queries as non-heavy:

  ALTER WORK ACTION SET SYSDEFAULTUSERWAS DISABLE

- Map only above-threshold queries as heavy:

  ALTER WORK ACTION SET SYSDEFAULTUSERWAS ENABLE

- Enable or disable concurrency threshold for heavy queries
  - ALTER THRESHOLD SYSDEFAULTCONCURRENT ENABLE
  - ALTER THRESHOLD SYSDEFAULTCONCURRENT DISABLE

- Change heavy query timeron threshold to 100000:

  ALTER WORK CLASS SET SYSDEFAULTUSERWCS ALTER WORK CLASS SYSMANAGEDQUERIES FOR TIMERONCOST FROM 100000 TO UNBOUNDED

- Limit concurrent heavy queries to 100

  ALTER THRESHOLD SYSDEFAULTCONCURRENT WHEN CONCURRENTDBCOORDACTIVITIES > 100 STOP EXECUTION

Tuning a System with Workload Manager

- Find queue time for each subclass

```
SELECT varchar(service_superclass_name,30) AS
  service_superclass,
  varchar(service_subclass_name,30) AS service_subclass,
  sum(wlm_queue_time_total)
FROM TABLE(MON_GET_SERVICE_SUBCLASS('','',-2)) AS t
GROUP BY service_superclass_name,
         service_subclass_name;
```

- Find queries under each subclass

```
SELECT appl_id, s.service_superclass_name,
       s.service_subclass_name
FROM TABLE (mon_get_activity(null, -2)) AS t,
     TABLE (mon_get_service_subclass('', '', -2)) AS s
WHERE s.service_class_id = t.service_class_id;
```

Tuning a System with Workload Manager

- System under-utilized and heavy query queue time exceeds 0
  - Reduce heavy query portion by increasing timeron threshold
  - Increase concurrency threshold

- System over-utilized
  - Increase heavy query portion by reducing timeron threshold
  - Queue more heavy queries by reducing concurrency threshold

Summary

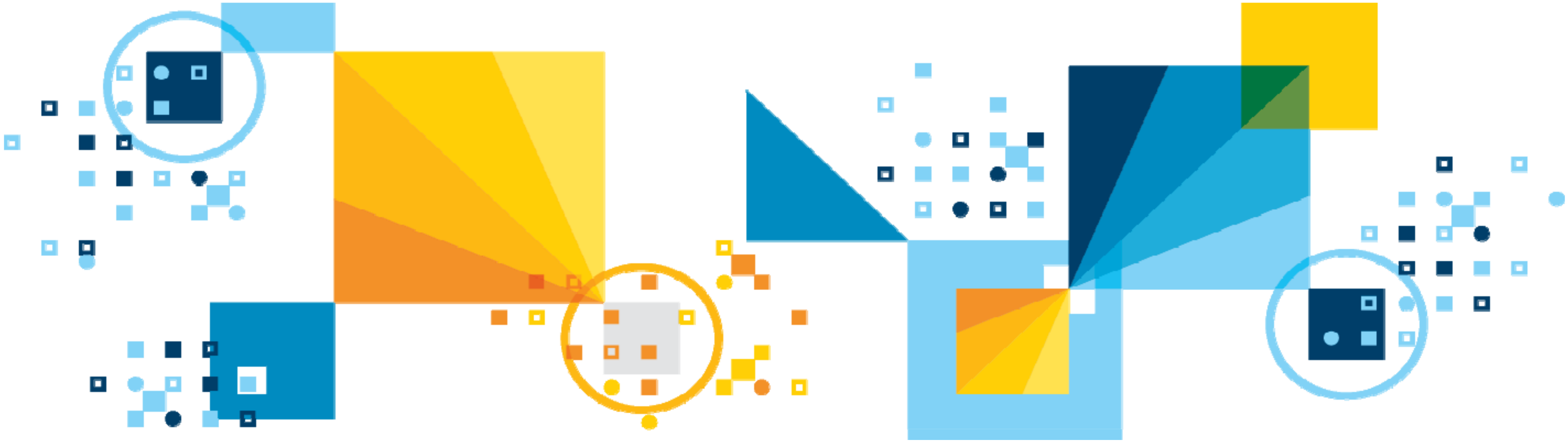
- **Work, service classes, workloads, work actions, thresholds and violations**
- **Table functions, stored procedures, histograms and event monitors**
  - Gathering the information to set the behavioral norm
- **Priorities: buffer pool and prefetcher**
- **WLM Dispatcher provides a mechanism to allocate CPU shares to different workloads**
  - Platform independent
  - Built in the DB2 engine
  - Soft and hard CPU shares can be used to assign a relative amount of CPU resources to workloads
  - CPU limits impose absolute limits on the CPU consumption by workloads
- **WLM data tag allows you to prioritize workload based on the data being accessed and where it resides**
  - Predictive: work classes provide flexibility to define which subclass a particular workload will run under
  - Reactive: thresholds can dynamically reallocate workload priority at runtime
- **Automatic Workload Management in DB2 10.5**

The next steps…

The Next Steps…

- Complete the Hands on Lab for this module
  - Log onto SKI, go to “My Learning” page, and select the “In Progress” tab.
  - Find the module
  - Download the workbook and the virtual machine image
  - Follow the instructions in the workbook to complete the lab
- Complete the online quiz for this module
  - Log onto SKI, go to “My Learning” page, and select the “In Progress” tab.
  - Find the module and select the quiz
- Provide feedback on the module
  - Log onto SKI, go to “My Learning” page
  - Find the module and select the “Leave Feedback” button to leave your comments

Questions?
askdata@ca.ibm.com



January 29, 2015