Preemptible SRBs

John Arwe

IBM Corporation
522 South Road
Poughkeepsie, NY 12601-5400

Abstract

MVS/ESA SP5.2.0 introduced some revolutionary changes, including the first new types of dispatchable work units in many years. These new dispatchable work units are Preemptible-Class SRBs, of which two types are currently being exploited: enclave SRBs and client SRBs. Exploitation of preemptible SRBs will have wide ranging impacts on the performance, performance tuning, and charge back of DataBase/2 (DB2) and the DB2 Distributed Data Facility (DDF).

This paper will examine these new dispatchable units of work and explain their role in the MVS/ESA operating system. Detailed in this paper will be changes required to an installation's System Resource Manager (SRM) and/or Workload Manager (WLM) parameters to fully exploit the capabilities of these new preemptible SRBs. The reader will also gain an understanding of the initial exploitation by DB2 version 4 of these new work units, and the benefits of this exploitation. Recommendations for managing these new SRBs in both WLM compatibility and goal mode will be presented, as well as the performance tuning and charge-back issues that must be considered.

Introduction

MVS/ESA SP 5.2.0 and DB2 Version 4 together provide support to improve response time on complex DB2 queries and to allow an installation to control DB2 Distributed Data Facility workloads as never before. *Preemptible-Class SRBs* and *enclaves* are new general-purpose facilities available in MVS/ESA SP 5.2.0 that DB2 Version 4 exploits to enhance the performance of complex queries and to enhance the management of Distributed Data Facility work. The use of Preemptible-Class SRBs and enclaves by DB2 Version 4 when running on an MVS 5.2.0 or later system will affect be of interest to persons involved with performance management, tuning, and charge-back issues.

Let us now examine the problems experienced by DB2 installations that the new MVS and DB2 support seeks to remedy, and how the MVS support solves each problem.

Note: For the remainder of this paper, "MVS" will be assumed to mean MVS/ESA SP 5.2.0 and "DB2" will be assumed to mean DB2 Version 4.

Problem 1: Response Times for Complex DB2 Queries

One way to reduce response times for complex DB2 queries is to have multiple CPUs processing each query in parallel. This is called *CPU parallelism* to distinguish it from parallelism of other resources such as I/O. The CPU parallelism of concern here is *not* concurrent access to DB2 data by multiple users but rather the notion of putting more than one CPU to work for each DB2 query to fully exploit the N-way capability of modern processors. The DB2 design prior to DB2 Version 4 presents a fundamental obstacle to exploitation of increased CPU parallelism of single queries: significant work is done under the caller's task. This design is an advantage for system management because each query is managed according to the DB2 user's dispatching priority rather than according to DB2's dispatching priority so the query is prioritized without introducing any new system externals. It is a disadvantage in that it limits the CPU parallelism of the query to one, the user's dispatchable unit.

DB2 has been working to exploit various kinds of parallelism for years in order to minimize the response times for queries: I/O has been off-loaded from the invoker's dispatchable unit to other dispatchable units and parallelized in existing releases; storage is associated with DB2 and prefetched in parallel with thread execution. DB2 Version 4 running on MVS/ESA SP 5.2.0 addresses the issue of using CPU parallelism to reduce the response time for complex queries.

When attempting to parallelize the CPU consumption of a DB2 query, MVS task-related limitations dominate. These limitations include:

1. MVS does not have support to allow DB2 to attach additional tasks in the caller's home address space from the DB2 Services address space. Thus there is no way for DB2 to implement CPU parallelism without sacrificing its existing system management characteristics.
2. Attaching a task is expensive when the intent is to create dispatchable units that do not really require a task environment. If the work to be run can tolerate the restrictions associated with SRB mode, attach would incur significant unneeded overhead.
3. TCBs still require virtual storage below the 16-megabyte line, which places an effective maximum on the number of tasks that can exist in parallel within one address space.

Creating SRBs to run portions of a query would avoid the task-related issues, but at what price? Any portions of a query run under local SRBs would accumulate CPU time to the DB2 Services address space SMF type 30 records rather than the DB2 caller's address space SMF records, impacting the ability of the installation to charge the user for the CPU time consumed by the query. Those same portions would also run at DB2's dispatching priority rather than the dispatching priority of the user, and would be non-preemptible. Each would be capable of dominating a CPU at its dispatching priority until the dispatchable unit *voluntarily* relinquished control. Tasks are preemptible, unlike local SRBs, which allows the dispatcher to interrupt a task at any time to run other work at the same or a higher dispatching priority. non-preemptible dispatchable units such as local SRBs, once dispatched, will continue to run until they incur a voluntary interrupt, such as suspend/page fault, or they complete *even if higher priority work is ready to use the CPU*. External and I/O interrupts are serviced if the SRB is enabled, but the SRB is redispatched after each interrupt is serviced.

Solution: Client SRBs

In order to allow DB2 to increase the CPU parallelism of queries without sacrificing its existing system management characteristics, IBM created client SRBs, one variety of a new dispatchable unit

type called Preemptible-Class SRBs. Exploitation of client SRBs in DB2 to provide CPU parallelism can result in significant elapsed time savings for complex queries. As with any change in the fundamental underpinnings of MVS or its major subsystems, this will affect performance management and tuning.

A client SRB is a preemptible SRB that runs in an address space but executes work on behalf of some other address space, called the client address space. All SRM-related dispatching controls are derived from the client address space, including the address space dispatch priority. CPU time consumed by a client SRB is accumulated back to the client address space and is included as CPU service in the client address space's current SRM transaction. When the client address space switches to a new performance period, so do any client SRBs running on behalf of the client address space. Service accumulated by client SRBs while a client address space is swapped out is *not* lost; it is accumulated to the client address space when the client is swapped in again. As a consequence, the service accumulated by client SRBs while the client address space was swapped out may trigger a period switch at swap in but will not do so before swap in.

Together these attributes allow DB2 to create dispatchable units that look, act, and are managed as if they were part of the address space on whose behalf they are executing, for example a batch job or TSO user, rather than part of the DB2 Services address space. Client SRBs are preemptible like tasks so that they will not dominate a CPU if work of higher or the same dispatching priority is ready.

DB2 CPU Parallelism

DB2 Version 4 CPU parallelism extends the I/O parallelism introduced in DB2 V3 to allow multiple client SRBs to run parts of a complex query at the same time, in order to reduce the elapsed time of the query. These client SRBs may be involved in creating result rows or analyzing separate pieces of the query. Complex queries processed on MVS may arise from either TSO QMF or batch environments, or from distributed requests running on behalf of remote users or programs. The decision to parallelize a query is made by DB2 based on factors that include table structure and package/plan binding.

DB2 packages and plans must be rebound to exploit CPU parallelism. If you are using I/O parallelism in DB2 V3, those packages and plans will run and continue to use I/O parallelism but cannot use CPU parallelism without being rebound. Application changes may be needed to allow DB2 to parallelize queries; DB2 provides new statistics for CPU parallelism in the thread-related accounting record (DSNDQWAC, in the SMF type 101 record).

Problem 2: Managing DDF Work

Prior to DB2 Version 4, DB2's Distributed Data Facility provides no ability to prioritize amongst requests according to their business value, since the requests are not reported as transactions to MVS. All requests flowing through DDF are managed under the SRM transaction for the DDF address space, thus CPU time accrues to the DDF address space. The system management consequences of this are extreme: controls for the DDF address space prior to DB2 Version 4 must be set so that the processing requirements of the requests most valuable to the business are satisfied. Any other requests that flow through the DDF address space also get this favorable treatment even if they are not as important to the business, effectively preventing tuning and performance management of requests within DDF via the normal SRM controls. Charge-back of the CPU time consumed by each request can only be accomplished by post-processing the thread-related SMF records that DB2 creates on behalf of the requests.

Until both DB2 Version 4 and MVS/ESA SP 5.2.0 are installed, DDF has additional deficiencies that do not manifest themselves in other DB2 environments:

- All requests received by the Distributed Data Facility from the VTAM network are run as local SRBs in the DDF address space; since local SRBs are non-preemptible any one request is capable of dominating a CPU.
- The DDF address space performs some front-end functions upon receipt of a network request that require a true task environment. During periods of high DDF activity, the local SRBs running active requests can consume enough CPU to prevent the network interface task from running to accept new requests.

Solution: Enclaves

The most fundamental system management problem DDF has is that MVS provides nowhere to anchor transactions except address spaces. In order to allow MVS to report and manage transactions, DDF needs somewhere to anchor those transactions. Clearly from a performance standpoint one would not want to have DDF create an address space per request just to anchor a transaction. What has been done is to create another place to anchor a transaction, called an enclave. The dispatchable units in an enclave are called enclave SRBs, another variety of Preemptible-Class SRBs. When running on MVS 5.2.0 or above,

the DB2 Version 4 Distributed Data Facility will always create an enclave per DDF request to anchor a corresponding MVS transaction.

Enclaves allow the *management of individual transactions* flowing through address spaces, something that simply has never been possible before. Since MVS is aware of and has access to each transaction, they can be classified individually and most importantly *each transaction is subject to period switch*. This means that you can separate out the long-running CPU killers from the shorter requests in the same manner that most installations already employ to control batch, by period-level controls.

Each enclave is a single transaction, which starts when the enclave is created and ends when the enclave is deleted. DDF creates an enclave for an incoming request when it detects the first SQL statement and deletes the enclave at SQL COMMIT, thus a DDF enclave transaction consists of a single SQL COMMIT scope. The service consumed by an enclave is reported in the SMF type 30 records of the address space which created the enclave. This address space is called the *owner* of the enclave. The DDF address space owns all enclaves created by the Distributed Data Facility. There is no special connection between the management of the enclave and its owning address space; each is managed separately according to the installation's goals and each is a separate MVS transaction.

In WLM goal mode, all goal types are valid for enclaves. In WLM compatibility mode, dispatching controls are valid for enclaves with the exception of time slicing. Time slicing is not supported for enclaves; if used on a performance group associated with an enclave time slicing controls are not used to set the enclave's dispatching priority. enclaves are not included in compatibility mode domain MPLs.

No RESET or DISPLAY capability is provided by MVS for enclaves, although the classification attributes for long-running enclaves can be examined through RMF. DB2 thread controls may be used as in previous releases to control threads associated with enclaves.

Note: DDF requests are eligible for CPU parallelism just like mainline DB2 queries. If a DDF request is parallelized by DB2, more than one enclave SRB is scheduled into the enclave representing the request.

Enclave SRBs

An enclave SRB is scheduled into a target address space but executes work on behalf of an enclave. Dispatching controls are derived from the enclave, including the major dispatch priority. CPU time consumed by each SRB is accumulated back to the enclave and is reported as enclave-related CPU service in the SMF type 30 records for the address space which created the enclave.

All page faults taken by SRBs in the enclave are interpretted as cross-memory page faults on the target address space. Thus page faults incurred by enclave SRBs will manifest themselves as cross-memory paging delay samples even when the SRBs themselves do not execute in cross-memory mode.

If an enclave SRB makes an I/O request, the associated I/O service is reflected in the home address space current at the time of the request.

Where Preemptible-Class SRBs Fit In

Preemptible-Class SRBs combine characteristics of SRBs and tasks and include the following types:

- Preemptible SRBs
- Client SRBs
- Enclave SRBs

Table 1 summarizes the major differences between the different types of Preemptible-Class SRBs and compares them to existing dispatchable units. preemptible SRBs are not discussed in this paper because they have no current exploiters, but they are included in the table for completeness.
All Preemptible-Class SRBs share certain attributes; for our purposes the most important is preemptibility. preemptible dispatchable units such as tasks and Preemptible-Class SRBs may be interrupted by the dispatcher at any time to run other work at the same or a higher dispatching priority.

Non-preemptible dispatchable units such as local SRBs, once dispatched, will continue to run until they incur a voluntary interrupt such as suspend/page fault or they complete *even if higher priority work is ready to use the CPU*. External and I/O interrupts are serviced if a non-preemptible SRB is enabled, but the SRB is redispatched after each interrupt is serviced. This ability to hold a CPU even when higher priority work is ready makes preemptibility a crucial factor in making client and enclave SRBs a viable replacement for tasks. Any dispatchable unit that is expected to consume large amounts of CPU we would naturally want to be preemptible in order to let the dispatcher run the work that the installation feels it should be running.

Service Units

The definition of CPU service has been changed to include the work done by Preemptible-Class SRBs. CPU time consumed by Preemptible-Class SRBs is considered CPU service since they are intended as a low-cost replacement for work that would otherwise be done in task mode. Preemptible-Class SRBs will not accumulate any SRB service.

The definition of SRB service is unchanged. CPU time consumed by existing local/global SRBs is considered SRB service, as it has been in previous releases of MVS.

The definition of I/O service is unchanged. I/O service is accumulated to the home address space when an SRB makes an I/O request, as it has been in previous releases of MVS.

The definition of MSO service is unchanged in the strict sense, but enclaves do present some implications for MSO service. The definition of MSO service is the product of CPU time consumed and central page frames occupied by an address space, with a scaling factor applied. An enclave does consume CPU time, but does not occupy any central page frames so its MSO service is always zero. More importantly from the overall system point of view, the enclave CPU time is not included in *any* MSO calculations. Thus when work previously performed by dispatchable units running on behalf of an address space is moved to dispatchable units running under an enclave, the MSO service that was generated by the work previously will fall to zero. The implications of this for DDF will be discussed further in a later topic.

Table 1 Dispatchable unit control/accounting characteristics

	Task	**Non-preemptible SRB (local or global)**	**Preemptible SRB**	**Client SRB**	**Enclave SRB**
Preemptible	Yes	No	Yes	Yes	Yes
Minor (task) dispatching priority allowed	Yes	No	Yes	Yes	Yes
CPU Time Reported as ____ service	CPU	SRB	CPU	CPU	CPU
CPU Time	Home Smf30Cpt	Home Smf30Cps	Home Smf30Cpt and Home Smf30Asr	Client Smf30Cpt and Client Smf30Asr	Owner Smf30Enc
Major (address space) Dispatching Priority	Home	Home, higher than any preemptible work in home address space	Home	Client	Enclave
CPU using, or delay samples attributed to	Home	Home	Home	Client	Enclave
Created by	Attach	Schedule or IEAMSCHD	IEAMSCHD	IEAMSCHD	IEAMSCHD
Relative cost	Expensive	Inexpensive	Inexpensive	Inexpensive	Inexpensive

Nuts and Bolts

You will need MVS 5.2.0 or above and DB2 4.1 in order to exploit the new dispatchable units, but do not wait until you have everything installed to read through this section. There are some migration actions that you can take in advance to prepare for this brave new world. It turns out that the most difficult problem in some cases is verifying that client/enclave SRBs are actually being created.

The interesting data for CPU parallelism, including counts of parallelized queries and reasons that candidates were not parallelized, can be found in the accounting records that are cut on behalf of each DB2 thread. When CPU parallelism is being exploited, the resulting SMF type 101 records must be aggegrated by a monitor in order to have the full picture. The information in the SMF type 101 records will detail the degree of parallelism achieved and when a query is not parallelized it will give some indications of the factors that resulted in the decision not to parallelize the request.

When dealing with enclaves, you have more options since enclaves are separate transactions. There are changes in service that you can observe to know that enclaves are being created but it is much simpler and more direct to classify them to unique performance groups/service classes, even if for reporting rather than control purposes. *Performance Monitoring and Chargeback* provides details on the service changes that you can expect to see when enclaves are being created by DDF. Remember that MVS *does not* provide any DISPLAY or RESET capability for enclaves.

Classifying DDF Transactions

If you are preparing to migrate to MVS/ESA SP 5.2.0 with DB2 Version 4, regardless of whether you are running in WLM compatibility mode or WLM goal mode, you will need to define classification rules for enclaves in a WLM service definition if you are running any Distributed Data Facility work. Remember that when running this combination of MVS and DB2, DB2's Distributed Data Facility will *always* create enclaves. The enclave transactions are *always* classified to a service class via the active service policy, *even in compatibility mode.* New attributes are allowed on the WLM administrative application for subsystem type DDF, a new IBM-defined subsystem used to anchor the classification rules for enclaves created by the Distributed Data Facility.

If you want to take advantage of the new Distributed Data Facility transaction management capabilities or you have any DDF work and are running in goal mode, do the following:

- Define service classes and optional report classes for DDF work. If you are running in goal mode see the section entitled *Goal Mode* for information on setting goals for DDF transactions. If you are running in compatibility mode the goals on the service class periods are irrelevant because MVS will not be managing the enclave transactions to those goals.

  Note: It is *strongly* recommended that you separate enclave work into service classes distinct from those used for address spaces but this *is not* enforced. Mixing two such different types of work in the same service class in goal mode may lead to unpredictable results.

- Define classification rules in a WLM service definition for the new DDF subsystem.

- Install the WLM service definition and activate a WLM policy. You must actually activate a WLM policy in order to read in the new classification rules, even if the system is running in compatibility mode.

Listed below are the various work qualifiers which are valid for DDF-related transactions in the WLM administrative application.

Accounting Information
: The value of the DB2 accounting string associated with the DDF server thread.

Correlation Information
: The DB2 correlation ID of the DDF server thread.

Collection Name
: The DB2 collection name of the first SQL package accessed by the DRDA requester in the unit of work.

Connection Type
: The DB2 connection type of the DDF server thread. This contains the value 'DIST ', indicating the thread is a distributed server thread.

LU Name
: The VTAM LU name of the system that issued the SQL request.

Net ID
: The VTAM net ID of the system that issued the SQL request.

Package Name
: The name of the first DB2 package access by the DRDA requester in the unit of work.

Plan Name
: The DB2 plan name associated with the DDF server thread.

Subsystem Instance
: The DB2 server's MVS subsystem name.

Userid
: The DDF server thread's primary AUTHID, after inbound translation.

Goal Mode

If you run Distributed Data Facility work and are either already running in goal mode or are preparing to migrate to goal mode: make sure that you have

classification rules for the DDF subsystem type defined, installed and activated, prior to starting DB2 V4. If you fail to do so the DDF enclave work will run with discretionary goals in the SYSOTHER service class.

If you are already running in goal mode on MVS/ESA SP 5.2.0 or above when you upgrade to DB2 Version 4, you should extract the information from your existing SMF type 101 records. Use the response times in these records to understand what is achievable for enclave response times in your system. Since the Distributed Data Facility (DDF) and the DB2 Services (ADMF) address spaces have probably been given high velocity goals, the response times currently being achieved may be more aggressive than what you need for the less important work. If you have had relatively unimportant requests getting the favorable treatment intended for important requests because they all ran according to the Distributed Data Facility address space controls, now is your chance to correct this situation. Business needs, tempered by the knowledge of achievability, should be used to set response time goals. Response time goals are probably most appropriate for work that is interactive in nature, as when there is a person on the other end of the network connection awaiting the response. You can also use discretionary goals if you have work that has no definite performance requirements. While velocity goals are allowed, there is no pre-existing data on which to base initial goal estimates for velocity goals. Should you elect to try out velocity goals for DDF enclave transactions, hedge your bets by using a low importance until you are sure the goal is achievable without unduly hurting other work. This will prevent WLM from sacrificing other workloads for an over-aggressive velocity goal.

Note that there may be multiple threads per request once CPU parallelism is enabled; since there will be one SMF 101 per thread, you or your monitoring product may have to aggregate multiple SMF records in order to gain a true picture of a DDF transaction. The shorter response time for parallelized queries may affect the performance index (PI) for periods with response time goals, since the PI compares the goal to the actual response time and the actual response time has decreased. If you are not using a response time goal on last period, the goal need not be changed.

Compatibility Mode

If you want to manage the DDF work as it was managed before MVS/ESA SP 5.2.0 and DB2 Version 4 were both installed: do not change your IEAICSxx or IEAIPSxx. This will cause all enclaves created by DDF to be given the performance group of the address space which created them, namely DDF. You will still get the benefit of the network interface task not being starved for CPU time when DDF is running large quantities of SRB work because DDF will be using enclave SRBs instead of local SRBs.

When you want to actively manage the DDF work: the classification rules currently governing the active WLM policy will generate a service class for the DDF enclaves even when the system is running in compatibility mode. You need to define service and optional report classes for the enclave transactions that DDF will create, install the service definition on the WLM couple dataset, and activate the policy; all of this can safely be done while still running in compatibility mode. The IEAICSxx parmlib member is used to map the service class to a performance group. Enclaves whose service class is not associated with a performance group in the IEAICSxx member default to the owning address space's performance group. Only one report performance group is allowed per enclave. For example:

```
SUBSYS=DDF,
   SRVCLASS=scl,PGN=p,RPGN=r
```

In the IEAIPSxx member you will need to define performance group p and give it a dispatching priority.

```
/* Pgn P is used ONLY for enclaves.
   DMN is irrelevant for enclaves.
   TSGP, TSDP are ignored for enclaves. */
PGN=p,(DP=F51,DUR=30K),
      (DP=M4)
```

If you want to manage work the old way but collect reporting data on groups of DDF enclave transactions: create an ICS entry for DDF and specify a service class and report performance group only (no performance group).

```
SUBSYS=DDF,
   SRVCLASS=scl,RPGN=r   /* no PGN= so
enclaves get
                            DDF pgn as today
*/
```

Performance Monitoring and Chargeback

The first part of this section discusses changes to the two most frequently used SMF records, types 30 and 72. The second section discusses differences you will observe in the system based on the DB2 functions being exploited. A final section discusses new RMF support for enclaves.

In a nutshell:

- CPU time and service for client SRBs generated by CPU parallelism of non-DDF work are reported as part of the address space on whose behalf they are running. This results in no change for accounting packages using existing SMF type 30 record fields. Optionally the accounting packages can be changed to isolate the client SRB time component due to CPU parallelism from the non-parallelized task component. client SRB service is included in the existing SMF type 72 record fields and cannot be isolated.
- CPU time for enclaves generated by DDF is reported in the SMF type 30 record of the owning address space in existing fields so that no changes to accounting packages using existing SMF type 30 record fields are required. The enclave-related information can be isolated if the accounting packages wish to change to collect new information.
- CPU service for enclaves generated by DDF have their CPU service reported in the SMF type 72 records in existing fields so that no changes to existing code are required. Their service is included in the existing SMF type 72 record fields and cannot be isolated.

SMF Type 30 Record

The type 30 record provides resource consumption data at the address space level. The Processor Accounting Section and Performance Section in MVS/ESA SP 5.2.0 contain information on Preemptible-Class SRB and enclave usage on behalf of the address space. No equivalent to the SMF-30 record is cut for enclaves; all recording of enclave service is done via the owning address space. This implies that there are no required changes to existing accounting packages which use SMF-30 service fields. If you wish to separately detail service consumed by enclaves created by an address space changes will be required to retrieve the values from the new fields. There is no reason to do so now unless you are curious; the value in the SMF type 30 record will contain service for all completed/in-flight enclaves since the previous SMF type 30 record was cut. There is not enough granularity in the SMF type 30 record to measure CPU usage by enclave; DB2's SMF type 101 records provide that level of detail or you can classify the enclave of interest to a separate report class/performance group.

The following existing fields in the SMF type 30 record have been modified.

Smf30Cpt
: Used to contain TCB CPU time; now contains the sum of TCB, enclave SRB, preemptible SRB, and client SRB CPU time. To compute TCB time subtract out Smf30Enc and Smf30Asr.

Smf30Csu
: Contains CPU service; CPU service now contains TCB service as it used to and new components for preemptible SRB and client SRB CPU service. There is no corresponding Smf30 field containing only the new components that can be subtracted out to isolate TCB service. Service consumed by enclaves created by the address space *is not* included.

Smf30Srv
: Contains total service, the sum of CPU, SRB, I/O, and MSO service; total service now contains its old value and new components for preemptible SRB and client SRB CPU service. There is no corresponding Smf30 field containing only the new components that can be subtracted out to isolate TCB service. Service consumed by enclaves created by the address space *is not* included.

The following are new fields in the SMF type 30 record that are related to client SRBs and preemptible SRBs. They are non-zero if preemptible SRBs run in this address space or if this address space was a client on whose behalf client SRBs were run.

Smf30Asr
: preemptible SRB and client SRB CPU time

The following are new fields in the SMF type 30 record that are related to enclaves. These fields contain the sums for all completed and existing enclaves created by an address space. They will be non-zero only for address spaces which own enclaves.

Smf30Eta
: Enclave transaction active time

Smf30Esu
: Enclave CPU service

Smf30Etc
: Enclave transaction count

Smf30Enc
: Enclave total CPU time

SMF Type 72 Record

The SMF type 72 record provides data collected by RMF monitor 1. The service consumed by Preemptible-Class SRBs has been included in total CPU service (SMF72CTS) and total service (SMF72SER).

Enclave transaction completions, active time, elapsed time, and delay samples are included in the existing fields. The sampled address space count includes enclaves so that calculations of average samples per time need not be changed. Since there are more things being sampled, both the numerators and denominators in any averages have been also changed.

With MVS/ESA SP 5.2.0 and DB2 Version 4, you can define WLM classification rules to segregate enclaves into unique service and/or report classes/performance groups just as you would for existing work like TSO or batch. This allows you to understand the work requests flowing through DDF better, and provides differing levels of granularity for workload analysis and performance management/tuning. It can of course be tracked over time to identify changes in the workload that might influence capacity planning decisions as well.

RMF Workload Activity Report Changes

It is not surprising that with the advent of new work types and new enclave transactions, changes are to be expected in the workload activity report data for the DB2 Services and Distributed Data Facility address spaces.

Stepping back for a moment however, there is a larger issue to note: quite simply, the definitions of CPU and SRB service have been refined. The change is not major, is not incompatible, but the terminology can be confusing if you don't adjust your thinking right now. People commonly think of CPU time as "TCB time" and CPU service as "TCB service", and likewise with SRB time/service. This is quite natural given the workings of MVS over its history, but it is no longer adequate. We now have to deal with SRBs that generate CPU service, and the old shorthand fails.

This is my suggestion: replace "CPU" with "preemptible" and "SRB" with "Non-preemptible SRB". This yields:

CPU = preemptible
SRB = Non-preemptible SRB

Changes Due to Exploiting DB2 CPU Parallelism

Because CPU parallelism for TSO and batch work is exploiting client SRBs, the changes to the SMF records and workload activity report are minimal. SMF30CPT will include the CPU time consumed by any client SRBs; SMF30ASR contains *only* the client SRB time; SMF30CPU and SMF30SRV will include the corresponding client SRB service.

There may be a slight upward change in CPU consumption for CPU-parallelized queries due to the split/merge overhead, and a sometimes dramatic corresponding reduction in elapsed and active time. The number of transactions is not affected, just their run-time execution environment. DB2 statistics on CPU parallelism candidates/successes may be found in the SMF 101 records.

Delay samples for client SRBs are treated as if the client address space experienced them so few changes are expected in reported delay samples due to client SRBs themselves, however the change in parallelism will lead to a change in the sample distribution. If only one task existed before, it would receive at most one sample per sampling interval; if *n* client SRBs have replaced the one task, there will be *n* samples per sampling interval. For CPU-bound queries this implies a rise in the CPU delay samples per elapsed time. Since MSO service is based upon CPU time and central page frames, no systemic change is to be expected. There will be an increase in MSO service if CPU time increases due to parallelization overhead.

Since the cross-memory environment of the work units will change when DB2 uses client SRBs for CPU parallelism, I/O service may move from the TSO/batch address space to the DB2 Services address space. The magnitude of the change will depend upon the number of I/O requests executed in each environment; given the current DB2 design which minimizes the number of I/O requests issued under a thread, it is not expected to be significant.

Changes Due to Exploiting Enclaves

Even if you as an installation do nothing to exploit the functions provided with DB2 Version 4 Distributed Data Facility and MVS/ESA SP 5.2.0, the instant you IPL this combination you will see a shift in service within the DDF address space, from SRB to CPU service. This is due to the exploitation of enclave SRBs by DDF.

One can expect further changes once you have begun to actively manage DDF work. See *Classifying DDF*

Transactions for details on how to go about controlling DDF work.

- SRB service consumed by the DDF address space will plummet as the bulk of it moves to CPU service in the enclave-related service classes/performance groups. If you do nothing special to classify enclaves at first and you are running in WLM compatibility mode, the enclave-related performance group will be the DDF performance group.
- CPU service consumed by the DDF address space itself should be virtually unchanged, assuming the DDF address space was reported separately in the past and no enclaves are classified to the same service class or performance group.
- MSO service for the DDF address space will be unchanged. Since SRB time is not used to calculate MSO service, it was not being used before. When the same time is consumed by enclaves, which do not accrue MSO service, the CPU time consumed will still not be used for any MSO service.
- I/O service should be unaffected since the cross-memory environment of the work units has not changed; the I/O will be attributed to the same home address space as before.
- Total active time will increase although active time for the DDF address space itself should be unchanged. This is a general effect of enclaves since multiple transactions can be executing in the same address space concurrently.
- The total number of transactions will likewise increase based on the volume of DDF work since enclave transactions are now reported to SRM.
- Execution delay and using samples reported will move from DDF to the enclave service classes/performance groups. Any changes due to parallelized queries in the DDF enclave environment will go unnoticed unless CPU-parallelization is done separately after exploitation of enclaves.

By actively managing the enclaves, CPU will now be flowing through DDF according to the installation's goals and logons during peak periods should not be gated by previously existing work.

RMF Support

RMF provides a high-level category for enclaves (*ENCLAVE) similar to *TSO and *BATCH on the SYSINFO report in monitor 3. The *ENCLAVE field is cursor-sensitive; clicking on it displays a pop-up containing classification information from the enclave itself:

```
------------------------------------------------------------
|                RMF Enclave Classification Data
|
| The following details are available for enclave ENC00002 :
| Press Enter to return to the Report panel.
|
|   Transaction program name : WLMTOOL
|   Userid                   : BHIM
|   Transaction class        : JES2
|   Network id               :
|   Logical unit name        :
|   Plan                     : TEST
|   Package                  : UPS
|   Connection               :
|   Collection               : COLLECTION
|   Correlation              : CTT
|   Subsystem type           : DDF
|   Function name            : FUNCTION
|   Subsystem name           : DDF
|
|
------------------------------------------------------------
```

Figure 1 RMF Enclave Classification Pop-Up Sample

The identifier ENC00002 is generated by RMF since there is no equivalent to a jobname for enclaves.

Do not expect to be able to watch every enclave here; only long-running enclaves will exist long enough for you to see and then query them before their transactions finish and DDF deletes them. The *ENCLAVE identifier can also show up on the delay report, for example as a significant CPU user.

Summary

This paper has described the exploitation of features in MVS 5.2.0 by DB2 V4 and explained their value. The immediate effects on the SMF types 30 and 72 records and the RMF Workload Activity Report have been described, along with information necessary to interpret the changes that you observe on your own systems. Information necessary to use the new capabilities to better manage your system capacity has been presented to encourage a hands-on approach to dealing with your DDF work.

References

MVS/ESA Initialization and Tuning Reference SC28-1452-01

MVS/ESA Planning: Workload Management GC28-1493-02

MVS/ESA Programming: Authorized Assembler Services Reference, Volume 2 GC28-1476-01

MVS/ESA Programming: Workload Management Services GC28-1494-01

Acknowledgements

The author wishes to thank the following people for their constructive review and suggestions for this paper:

- Ed Berkel
- Peter Enrico
- Dave Emmes
- Wayne Morschhauser