
A whitepaper on workload based performance management for PeopleSoft and DB2 on z/OS.

TARGET AUDIENCE

This document is intended for the z/OS systems programmers responsible for maintaining the WLM environment. It also provides basic information on WLM fundamentals for DB2 Database administrators maintaining a PeopleSoft environment.

DISCLAIMER

This document is provided without any warranty with respect to the content and Hewitt and Larsen, LLC cannot be made liable for any claims / representations made here. All examples are for illustrative purposes only. The reader is advised to familiarize himself or herself with PeopleSoft / WLM / DB2 before attempting to make any change to the configurations.

INTRODUCTION

z/OS workload manager (WLM) provides the foundation to establish the “Workload based Performance Objectives” on *z/OS* operating system. This document outlines the workload characteristics of a typical PeopleSoft implementation and tries to provide a framework which organizations can deploy to realize better manageability, maximized throughput and visibility of workload meeting the *service level agreements (SLA)*. It can help in proactive capacity planning. Let us first look at the fundamentals of work load manager in a *z/OS* environment.

“Installations today process different types of work with different completion and resource requirements. Every installation wants to make the best use of its resources and maintain the highest possible throughput and achieve the best possible system responsiveness. Workload management makes this possible. With workload management, you define performance goals and assign a business importance to each goal. You define the goals for work in business terms, and the system decides how much resource, such as CPU and storage, should be given to it to meet the goal. Workload Manager will constantly monitor the system and adapt processing to meet the goals.”

From IBM's WLM home page

Work load management is centered on the following variables. A work load is simply any work that demands a service. Examples are batch jobs, CICS, DB2, TSO, and APPC transactions. Resource represents CPU, Storage and I/O requests.

1. Service Policy is a set that defines the performance objectives and capacity boundary for all the work in a *sysplex*. Only one policy can be active at any time. However to meet varying demands of processes you may maintain different service policies and activate one of them to suit the needs.

2. Service class is a group of work that shows similar characteristics in terms of performance goal, resource requirements and business importance. A service class is divided into one or more service periods, each period lasting for a specified duration. Each service class period is assigned a performance goal which can be response time, velocity or discretionary. Since a work can have varying resource requirements by defining multiple periods we can control the overall performance objective thereby controlling the resource consumption by the work. The mantra is “Favor short ending work without undue delay”. As an example we can group all test batch jobs into a service class and set aside a performance objective.

3. Classification Rules help WLM determine how it should assign the workload into a service class.

4. Resource Group provides a way to guarantee CPU capacity available for a given service class or a set of workload.

In addition to the above, a **Report Class** is a group (of workload) similar to service class but used in performance reporting. We can create as many report class as we want to aid in granular performance study. As an example a report class called *PSNVS* will help you understand the resource consumption of all *nVision* activities in your system.

Two different workloads, classified in different service classes can belong to the same report class

PERFORMANCE GOALS

A Performance goal determines how fast the system should allocate the resource needs from the work. A goal can be based on the execution velocity, the expected response time or a discretionary goal. In essence it drives the business objective and importance of the work.

Execution velocity goals define how fast work should run when ready, without being delayed for processor, storage, I/O access, and queue delays.

Response time goals indicate how quickly you want the work to complete. (It can be in average or percentile)

Importance is a value from 1 to 5 drives the business importance of the work.

Duration is the total service units, a work is allowed to consume.

WORKLOAD PYRAMID

Ideally you should identify and group all workload in a Pyramid fashion. Those that are at the top of the pyramid should be classified at the highest priority while those that are at the bottom of the pyramid are classified with a low but acceptable priority. However simply assigning all workload at the same priority will do no good. The concept is in determining how we can provide objectives to WLM so the workloads can be discriminated depending upon the service level agreements. Once we determine the objectives the next step is to identify a proper classification rule for the workload and assign a service class, importance and the duration in terms of service units.

SYSTEM, *SYSSTC* and *SYSOTHER* are internal service classes. *SYSTEM* has the highest priority followed by *SYSSTC*. *Figure 1.0* shows typical workloads that fall into either of these categories. *SYSOTHER* has the lowest priority in the system. Users can set aside a goal for a workload by choosing a proper *Importance* from 1 to 5 and assigning a *Velocity* or *Response* time objective or if needed a discretionary goal.

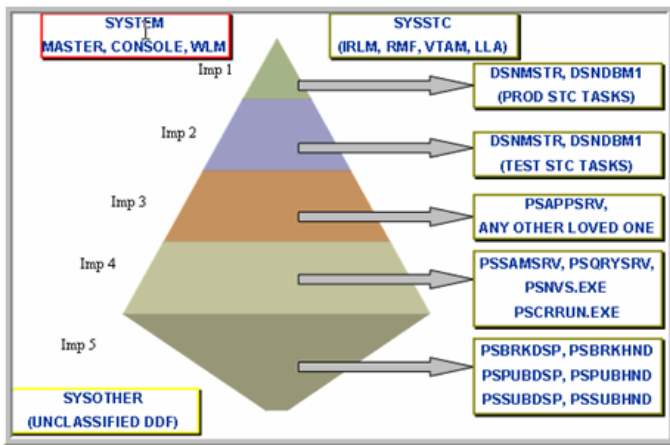


Fig 1.0 Workload Pyramid

When a work is defined in “Discretionary” goal, the system will provide resources if there are adequate resources left in the system and no other workload is in need of resources at that time. Avoid setting too aggressive or too conservative goals. There is nothing WLM can do if all workloads in the system fall in *Importance* 1. Similarly assigning everything to a low *Importance* goal is also unlikely to help. Generally *Importance* 1 and 2 are chosen for server tasks such as **DB2**, **CICS** and other system address spaces. Most of the user workloads should typically fall in *Importance* 3 or 4 followed by some low priority workloads in *Importance* 5. You should never assign a higher priority to an allied user workload than the priority of the respective subsystem address space. If DB2’s DBM1 address space is at *Importance* 2, what good it is going to do for an allied thread, assigned an *Importance* of 1.

There are exceptions to this rule. Special tasks such as Performance Monitors will need to be prioritized at or above the priority of the DBM1 address space. In fact within PeopleSoft system itself, “remote calls” will need high priority to avoid locking issues as we will see later.

In the sample workload pyramid in *Figure 1.0* we have chosen a Higher *Importance* for DB2 address spaces. We chose *Importance* 3 for PSAPPSRV tasks which represent the meat and potatoes for online workload in PeopleSoft. We chose to lower the *Importance* of online workload such as PSQRYSRV and PSSAMSRV. The illustration is to show how the discrimination takes place among the mix of workload.

WLM takes state samples of the work every 10 seconds and makes adjustments to the dispatching priority as needed. If the policy contains too many service classes, it might take several 10 second cycles for it to adjust the priority. The assignment of priority is done by gathering a list of *Donors* and *receptors* and the importance of the work. A Donor can be either a work with low priority or a work that is exceeding its set aside performance goal. Resources will be stolen first from discretionary work. WLM will try to equalize

performance index (PI) for the workloads in the same importance.

PS WORKLOAD

A typical PeopleSoft application workload falls into one of the following categories.

1. **Online**, this includes main workloads such as PSAPPSRV, PSQRYSRV, PSSAMSRV and PSQCKSRV services.

2. **Application server agents**, which comprises of processes such as *PSPUBHND*, *PSSUBHND*, *PSBRKHND*, *PSPUBDSP*, *PSSUBDSP*, *PSBRKDSP* etc. These processes scan the database every 15 seconds (configurable within the application server configuration) and look for outstanding messages to process.

3. **Process scheduler agents**, which comprises of processes such as *PSPRC*SRV and *PSDST*SRV. These processes are part of the built-in job scheduler and scan the PeopleSoft process scheduler tables periodically and look for the jobs / works to be processes.

4. **Batch** which comprises of regular *MVS* batch *COBOL*, *SQR* processes and *COBOL / SQR* processes that run in the distributed side (*Windows* or *UNIX*).

5. **OMVS**, which comprises of *OMVS* tasks such as Process Scheduler, Distribution server and *Application Engine (AE)* processes.

6. **OTHERS** We have included others to represent mixed workloads such as *PeopleSoft data mover*, *COBOL* programs that are invoked by a ‘*Remote Call*’ function that actually run in the open systems side (distributed), but issue SQL requests to DB2 so these are just a set of DDF workloads but their processing nature is similar to a typical batch process. There are also DDF transactions mainly used in reporting such as *nVision* and *Crystal*. There are many other application server processes that I have not mentioned here but the classification as above is the most frequently used processes in any installation.

TYPES OF GOALS FOR ONLINE WORKLOAD

What type of Goal is appropriate for the online workload mix?

As we saw before, PeopleSoft has clearly, a mix of online, batch and reporting workload. It is important to understand the characteristics of the workload before deciding to choose the type of goal (*Percentile response time* or *Velocity*). Let us take online workload first.

PeopleSoft’s online workload is not an ideal candidate for response time goals. Why?

PeopleSoft's online workload is not conducive for a percentile response time goal. A typical PeopleSoft online workload spans one or more *units of work (uow)*. When a distributed unit of work arrives at z/OS, an enclave control block is created and the enclave control block terminates when the unit of work terminates. (Assuming we have set the DB2 system parameter *CMTSTAT* as *INACTIVE*). When a unit of work originating from *PSAPPSRV* terminates, the DB2 thread that conducted the unit of work goes inactive in DB2. Its SRM information is reset. Within each unit of work, the PeopleSoft workload can perform several tasks such as querying the database for the structural definition of the pages / panels to be built, querying the database to get the associated Peoplecode programs, validating search keys entered and making sure that the object¹ (PeopleTools object) was not changed since the last time the object was referenced last time, reading the metadata² tables to get structural information about tables and fields and last but not the least querying the database to present the user with the result set in case of an inquiry transaction or updating the database in case of a data entry transaction. A typical business transaction consists of hundreds of SQL statements. More than 95% of these SQL statements are short ending and are issued against tables that are relatively small. These short ending SQL statements do not consume much processing time in terms of average service units and the duration of the enclave control block will be fairly short. If you ever investigate the online SQL trace in the PeopleSoft system, you can see many of such short ending SQL statements. However at the very end of the trace files (not necessarily at end always) there will be one or more SQL statements that directly correspond to the context of the business transaction (For example SQL calls against *JRNL_LN* if it is a GL Journal inquiry transaction). Under these situations, if we attempt to set a percentile response time goal, the goal will be easier to meet during 90% of the time. However workload may miss the goal or even worse WLM may decide to deliberately de-prioritize a UOW when it is in need of a higher priority in the system. To be specific, consider a SQL statement such as `SELECT * FROM PSVERSION` and let us assume that this is the only statement in that UOW. It poses two issues. It can miss the goal then that alone will cause significant degradation in the response time. If it constantly exceeds the goal, then there exists a possibility that WLM can deliberately lower the availability of resources for future UOW or enclaves, because workload exceeds its objective. We made an assumption that there is just this SQL statement for the UOW. It can be seen that each UOW contains several SQL calls to relatively smaller tables such as *PSVERSION*. On the other hand, assuming we do a Journal Inquiry through the PeopleSoft's system, consider a query such as `"SELECT * FROM PS_JRNL_LN WHERE JOURNAL_ID =?"` This SQL must get high priority. When response time goal was set and on an average if WLM perceives that goals are constantly exceeding the set aside objective, key SQL statements such as the one described above can miss the objective.

Furthermore, we enable I/O priority management in WLM as a defacto standard for managing I/O response time. When WLM deliberately lowers the dispatching priority, it can result in equally lowered priority on the I/O as well. However only random I/Os are affected by this behavior because prefetch I/O is scheduled at the priority of *DBM1* address space. In DB2, any I/O done to a pageset³ containing less than 4 pages will be performed by random I/O.

Who wants to set an objective which results in frequent *GOAL MISS*? Due to this reason, a PeopleSoft online workload should not be defined with a percentile response time goal.

It is equally arguable that it is quite possible to try to find a suitable response time goal for the first period. After all even certain batch processes can be set with a response time goal. However it needs a lot of trial and error tweaks and constant revisions until we get the proper goal established. A lowest programmable response time is 15 milliseconds and many UOW originating from PSAPPSRV has the potential to end in 50 milli seconds or less. You have to find out the suitable value to suit the installation.

DDF CLASSIFICATION RULES

Note that we are talking about classification rules ahead of service classes. Unless we know how to classify, we cannot talk about service classes at all.

A workload originating from DDF can be classified by choosing one of the following identifiers.

- Accounting Information (AI)
- Collection Name (CN)
- Connection Type (CT)
- Correlation Information (CI)
- LU Name (LU)
- Net id (NET)
- Package Name (PK)
- Plan Name (PN)
- Procedure Name (PR)
- Subsystem Instance (SI)
- User ID (UI)

It is recommended that we choose *Subsystem Instance*, *User ID* and *Correlation Information* for our classification rules.

The sub system name is the name of the DB2 sub system as defined in *SYS1.PARMLIB*. The *User ID* is the PeopleSoft *access id* which the application server uses to establish connection to the sub system. The *Correlation name* is derived from the client process name.

The first exercise is to gather the list of all PeopleSoft related processes, their processing nature, their environments and the business objectives. Let us take the

DDF environment first. Since the *PSAPPSRV* process handles the major portion of the online workload, we need to assign a higher priority to *PSAPPSRV*. Typically, the rest should start at a medium priority before reducing gradually. It is completely up to the installation to control the classification. But the following template can give you a general idea about workload prioritization. To illustrate, the service class assignment is shown in *Figure 1-1* while the respective priority assignment is shown in *Figure 1.2*.

WORKLOAD	SVCCLASS
PSAPPSRV	DDFHI
PSQRYSRV	DDFMD
PSSAMSRV	DDFMD
PSPRCSSRV	STCMD
PSPUB*	STCMD
PSNVS*	DDFMD

Figure 1.1 Sample service class definitions

Users of PeopleSoft financial applications may want to run adhoc queries or reports more frequently. Some of these queries / reports may be critical to the business.

The illustrative example as shown in *Figure 1.1* and *Figure 1.2* treats *PSAPPSRV*, *PSQRYSRV* and *PSSAMSRV* at the same priority for the first period, even though they have been classified into different service classes. Note that *PSQCKSRV* is only used in Windows - three tier⁴ mode and not used in PIA mode⁵.

The DDF threads from Process scheduler and Distribution server (*PSPRCSSRV*, *PSDSTSRV*) running in the distributed environment should be classified into *STCMD* because **they need a single period service class**. They need to be responsive enough to service the workload. They sleep for the most of the time and when they wake up the processes should get the resources without any undue wait. Their behavior is similar to a started task. Similarly the application messaging services such as *PSPUBHND*, *PSSUBHND*, *PSBRKHND*, and *PSMSGHND* etc also needs a single period service class and should be classified into either *STCMD* or *STCLOW* depending on the intensity of the usage. Classifying them into *STCMD* (started task medium) is a generally acceptable rule. If the installation does not utilize the application messaging services (*PSBRKDSP*...) then they can be grouped into *STCLOW*. **The key here is that they will need to be grouped in a single period service class**. It is worthwhile to mention here that the scan interval for these services should be adjusted to a value between five and ten minutes. *PSDBGSRV* should be classified into *STCLOW*. Similarly the *COBOL* and *SQR* processes running in the distributed environment may need to be classified into a service class similar to that of regular MVS batch jobs. All installations will be able to classify these

workloads into one of their existing service classes (*PRDBATMD*, *TSTBATHI* etc) that have similar profiles as the ones that are being described here.

Most recent PeopleTools release includes an option to enable or disable persistent database connections for messaging server processes. A single period service class is preferable even when the persistent connection is disabled.

SVCCLASS	IMPORTANCE	DURATION	GOAL
DDFHI	3	2000	VEL - 60%
	4		VEL - 40%
DDFMD	3	2000	VEL - 60%
	4	30000	VEL - 40%
	5		VEL - 20%
DDFLOW	4	1000	VEL - 10%
	5	30000	VEL - 5%
			DISCRETIONARY
STCMD	3		VEL - 30%

Figure 1.2 Sample Priorities

Duration specified here is only for illustration. Actual duration will depend on CPU service coefficients and the system characteristics such as defined capacity, processor rating etc.

Processes such as *PeopleSoft data mover*, remote called *COBOL* programs, batch *COBOL* and *SQR* processes that actually run in the distributed side should be dispatched at a moderate velocity and moderate importance goal similar to that of a batch process (For Example, *PRDBATMD*).

Certain customers may need to do frequent remote calls as in a typical high volume online GL Journal Edit environment. They want to enter the journals and immediately invoke an online edit. Such processes may need to be dispatched at a higher priority (Batch High or perhaps even as a Hot Batch).

Remote called process will take whatever priority assigned to PSRUNRMT, executing in the PeopleSoft application server. This is run on-demand and not part of the application server domain processes.

Generally the processing time for these 'Remote Called' processes depends on the volume of records to be processed and the resulting SQL performance but in terms of elapsed time these processes typically run for 1-2 minutes (elapsed time). Since online invocation do not typically process large data volume, the process executes within thirty seconds and a high velocity goal is generally justifiable.

A "Remote called" process has the potential to introduce locking conflicts and if it is unnecessarily canned, performance issue would shift elsewhere. Sometimes it is even better to prioritize them much higher than HOTBATCH. If you encounter consistent locking issues and the unaccounted time reported by the DB2 performance monitor is higher, you should consider increasing the priority of "Remote Called Processes".

However it is strongly recommended to guard against a possible misuse of the high priority privilege by quickly lowering the priority of those transactions that do take more than two minutes of elapsed time.

DDF SERVICE CLASSES

We should classify **DDF** workload into two or three service classes such as **DDFHI**, **DDFMD** and **DDFLO**. Usually **DDFMD** is chosen as the default service class.

The assignment of *Importance* and goal definition should be chosen in accordance with the work load pyramid as described above. We also need to make sure that the assignment fits well within the existing mix of workload in the *SYSPLEX*. However you decide to classify, never exceed the importance and the velocity you defined for the DB2 subsystem address spaces. For example, assuming that *DBM1* address space is defined with an *Importance* of 2, you can define the goals for *DDFHI*, by starting with an *Importance* of 3 and a starting velocity goal of 60. Similarly set an *Importance* of 3 for *DDFMD* and assign a starting velocity goal of 60. For *DDFLOW*, assign an importance of 4 and a low velocity goal of 10. All unclassified workloads should fall into *DDFMD*. This will ensure that those short ending transactions among the unclassified work will still be able to complete fast.

It is very important to fit the workload within the existing policy. We need to make sure that we give enough priority so that the existing workload is not affected by the change. In doing so, analyze the current service definition and choose the importance appropriately. For example, you may choose to assign an importance of 2 for *DDFHI* and *DDFMD* transactions, assuming the *DBM1* is at Importance 1 (which is likely to be the case in Production) so that we are able to maintain the workload pyramid, we discussed earlier.

OMVS CLASSIFICATION RULES

Let us take a look at the OMVS environment now. Apart from the interactive *UNIX* workload (UNIX commands), we have Process scheduler, Distribution server and application engine processes to manage. Failure to classify OMVS child tasks is a very common issue with many PeopleSoft installations. By default they may get the same dispatching priority as *OMVS* itself. Sometimes, in an installation where *OMVS* transactions have been classified properly it is very normal for the default *OMVS* workload to fall into a multi period service class, which suits an interactive workload

(UNIX commands), or a low *Importance* and low velocity service class. In such a case the AE jobs may be attaining a very low throughput. We need at least two service classes for the *OMVS* workload. Classify the Process scheduler and Distribution server into a single period service class with high importance and medium velocity goal such as *STCMD*. You can use the Process scheduler's job name to do that.

The *AE* batch jobs can come into the system in three different ways. When a user creates a process request to run a process in the PeopleSoft system, the Process Scheduler can spawn the *AE* process as a child process. Another way is when the user uses the *BPXBATCH* interface in the *JCL* and submits it to the system. The third but un-common way is to invoke *AE* from a UNIX command line (as in *psae <parameters>* from within the shell). You have two important variables that can be set in the shell environment. *_BPX_JOBNAME* and *_BPX_ACCT_DATA* are the two variables that can be set as part of the user's environment (*.profile*) or perhaps even in the *psconfig.sh* file, which is used by PeopleSoft to provide environmental parameters for the shell. *_BPX_JOBNAME* is also a parameter in the Process Scheduler configuration file. It is referred as *AE_JOBNAME* in the Process Scheduler configuration file.

WLM can use either of the two variables to classify the *AE* jobs. You should typically use a service class similar to *PRDBATCH* for high importance jobs and a *TSTBATCH* service class for lower priority jobs.

OMVS SERVICE CLASSES

Typically, you may have *PRDBATMD*, *TSTBATHI*, and *TSTBATMD* etc. Try to fit the *AE* jobs as a batch process that adheres to one of the service classes as mentioned. In essence try to prioritize the same way the regular PeopleSoft batch workloads such as *COBOL* and *SQR* executing on the mainframe.

REPORT CLASSES

Once you have classified the service classes you need to group similar workloads into report classes so you are

Proper classification rule is just a part of the work. The other important part is to actually find out and analyze whether you are meeting the goals or not.

able to report the performance of related workload. For example, you can assign the workloads originating from *PSAPPSRV* to a report class called *PSONLINE*. Similarly tasks such as *SQR* and *Crystal* can be assigned a report class called *PSREPORT*. Assigning report class enables you to run performance reports at a granular level. You can determine which workload consumes most service units.

You can assign workloads defined in two different service classes to the same report class.

We have chosen the names PSREPORT, PSONLINE etc to descriptively illustrate the type of work they represent.

MONITOR THE CHANGES

Periodic monitoring is a crucial part. Keep monitoring the performance index (PI). A PI of 1 means you are meeting the goal. A PI value less than 1 signifies that you are exceeding the goals. A PI value greater than 1 signifies that the workload is unable to meet the goal. It could be because of other higher importance jobs or delays. A variance of about .2 on either side of 1 is probably OK. However goals that consistently exceed the PI can be set with less aggressive values. Similarly goals that consistently miss the performance index should be studied and if necessary their priorities should be adjusted, assuming dispatching priority is what is causing the workload to miss the performance index. While making Velocity adjustments always try to do in increments of 10. You may not influence the goal significantly by adjusting from 30 to 31 or 30 to 35.

LIST OF PEOPLESOFT PROCESSES

With every new release of the application, new processes are being added. In Figure 1.3, we have given below a list of most commonly encountered processes, their workload type and their processing nature.

For a complete list of all the processes, please look at the bin directory in the application server.

PROCESS	WORKLOAD	CHARACTERISTIC
PSAPPSRV	DDF	Short ending; Online interactive
PSSAMSRV	DDF	Short ending; Online conversational
PSQRYSRV	DDF	Long running; PeopleSoft Queries
PSQCKSRV	DDF	Short ending; Online search, used in 3 tier Windows mode only
PSRUNRMT	HOTBATCH	Short ending; Online Remote Call
PSMONITORSRV	STC	Periodic; Domain resource monitor
PSPPMRSRV	STC	Periodic; Domain Performance Monitor
PSWATCHSRV	STC	Periodic; Server Process watch guard
PSDBGSRV	STC	Periodic low priority; Server process to aid debugging

PROCESS	WORKLOAD	CHARACTERISTIC
psrns.exe	DDF	Long running; nVision Reports
pscrun	DDF	Long running; Crystal Reports
sqrw.exe	BATCH	Long running; SQR batch process / Reports
psqr	BATCH	Long running; SQR batch process / Reports
PSAESRV	BATCH	Long running; Application Engine
psae	BATCH	Long running; Application Engine
PSMSTSRV	STC	Periodic; Master Process Scheduler
PSPRCSSRV	STC	Periodic; Process Scheduler server process
PSDSTSRV	STC	Periodic; Distribution Server process
UVSH	BATCH	Long running; used in EPM application to move data
psdaemon	STC	Periodic; Posts Report files

PROCESS	WORKLOAD	CHARACTERISTIC
PSMSGHND	STC	Periodic; Message Handler, Light weight
PSMSGDSP	STC	Periodic; Message Dispatcher, Light weight
PSBRKHND	STC	Periodic; Message Broker Handler
PSBRKDSP	STC	Periodic; Message Broker Dispatcher
PSSUBHND	STC	Periodic; Message subscription handler
PSSUBDSP	STC	Periodic; Message subscription dispatcher
PSPUBHND	STC	Periodic; Message Publication handler
PSPUBDSP	STC	Periodic; Message Publication dispatcher

PROCESS	WORKLOAD	CHARACTERISTIC
pside.exe	DDF	Short ending; PeopleSoft application development tool
psqed.exe	DDF	Short ending; PeopleSoft windows Query development tool
psdmt.exe	BATCH	Potentially long running; PeopleSoft data mover
db2bp	DDF	Adhoc; DB2 Command Center
Java / javaw	DDF	Adhoc; Java programs used in PeopleSoft
PSRENSRV	STC	Periodic; Real time event processor
PSMCFLOG	STC	Periodic; Multi channel Log
PSUQSRV	STC	Periodic; used in Multi channel framework

Figure 1.3 PeopleSoft Processes and their characteristics

COMMON ISSUES

Failure to recognize problems with work load balancing.

Setting too aggressive goals.

Setting low importance for critical workload is a classic issue.

Too low a priority may shift the problems elsewhere especially DB2 threads can hold locks for a longer duration.

Too high a priority may shift the problem elsewhere where SYSTEM tasks could be competing too hard to achieve their set aside goal.

Improper classification rules for the work load.

Too many workloads falling into default service class.

Keep the policy simple, maintaining only the definitions that you actually require.

Failure to investigate "un-accounted" time reported by DB2 performance monitors will result in poor utilization.

GLOSSARY

Classification Rules:	A set of rules to assign an incoming work into a service class and / or report class.
DDF:	Distributed Data Facility.
Duration:	Length of the service class performance period expressed in service units.
Enclave:	A z/OS transaction or unit of work that does not have any allied address space. Enclaves can have multiple engine dispatchable units and are either dependent on another address space or independent.
Execution Velocity:	Represents how fast a work should be dispatched in the system.
Goal Mode:	z/OS implements goal mode to prioritize various work based on business importance to manage resources.
Importance:	A value from 1 to 5 (1 being higher) specifies how important a work is for the business.
Performance Index:	A measure of performance in WLM.
Performance period:	Provides a way to specify a goal and performance duration depending on the varying characteristics of the workload. (In lay men's term it is similar to specifying Job 1 should get the maximum CPU for the first 10 minutes and then it is sufficient if it gets 40% CPU for the next 20 minutes and so on.) In reality elapsed time is never used in the picture. It is measured in service units.
Report Class:	It represents a logical group of all tasks / processes which exhibit similar characteristics so that they can be reported by a common report class to report on the performance characteristics. A report class is separate and independent of the service classes. We can group a work into the same report class even though they are part of different service classes.
Response Time goal:	
Service class:	It represents a logical group of all tasks / processes which exhibit similar processing characteristics so that we can assign a similar business importance thereby treating the task / process at the same priority levels.
Service Definition:	A service definition is a collection of all service classes, classification rules, resource groups, report classes and service policies.
Service Policy:	It represents the scheme of workload management. Although we can have multiple service policies there can be only one active policy at any given moment. A policy can be changed dynamically.
Service unit:	It is the measure of the amount of services a task / process consumes
Work:	A work represents one or more engine dispatchable units. A work can be any request into the system. (For example, a batch job, DDF thread or a CICS transaction, DB2 system etc).
Work load:	A work load is simply a mix of the works in the system.

Application Engine:	Application Engine is a proprietary code base which is used by PeopleSoft to perform batch processing. In z/OS it utilizes UNIX System Services. It is commonly referred as AE process.
Application messaging:	Application messaging provides the framework for asynchronous / synchronous data transfer between PeopleSoft and / or non PeopleSoft applications. This term is now deprecated and replaced by integration broker. (See integration broker below)
Application Server:	Application Server provides the platform for transaction handling in PeopleSoft application. It is based on BEA TUXEDO framework.
Integration Broker:	Provides the framework for asynchronous / synchronous data transfer between PeopleSoft and / or non PeopleSoft applications.
Crystal Reports:	A facility to produce reports. It utilizes a Windows environment and used extensively in HR system.
Domain:	A domain is commonly known as the application server domain.
nVision:	A proprietary facility to generate Financial Reports. It utilizes Windows and Microsoft EXCEL.
OMVS:	Open Edition MVS system; Unix System Services.
PeopleCode:	A proprietary code base which is used in PeopleSoft application to perform business logic / validations in both online and batch processes.
Process Scheduler:	A server task which periodically looks for jobs / processes that need to be submitted to the system.
PSAPPSRV:	A server task as part of the Application server used to conduct the online transactional workload. The PSAPPSRV service implements a pseudo conversational method for communication.
PSBRKDSP:	A server task as part of the Application Server's messaging system. A dispatcher assigns the incoming work to a handler.
PSBRKHND:	A server task as part of the Application Server's messaging system. A handler runs the routing and processing rules on the work, the dispatcher assigns. A broker handler invokes the appropriate PUB / SUB dispatchers to perform the work.
PSCRRUN:	Crystal executable.
PSDBGSRV:	A server task as part of the Application Server provides the developers with the ability to debug Peoplecode. This is used primarily (only) in a development environment.
PSDSTSRV:	A server task as part of the Process Scheduler, it copies the report files generated by PeopleSoft Processes to the report repository. A report repository is a location in the web server where reports are stored for later viewing.
PSMONITORSRV:	A server task as part of the Application Server; it reports host / domain resource information to Performance Monitor (PSPPMRSRV). It supports canceling a PeopleSoft Query and resolves master / slave failover for Application messaging system.
PSMSGDSP:	A server task as part of the Application Server's messaging system. Message Dispatcher and Message Handler represent a low-weight system that is primarily used in low volume environment. They handle all functions otherwise handled by PSBRKDSP, PSBRKHND, PSPUBDSP, PSPUBHND, PSSUBDSP and PSSUBHND.
PSMSGHND:	see PSMSGDSP above.
PSNVIS:	nVision executable; see nVision above.

PSPPMRSRV:	PeopleSoft Performance monitoring agent.
PSPRCSSRV:	PeopleSoft Process Scheduler.
PSPUBDSP:	A server task as part of the Application Server's messaging system; this is invoked every time a message is published to another PS / non PS system. The dispatcher assigns work to handler.
PSPUBHND:	A server task as part of the Application Server's messaging system; this is invoked by the PSPUBDSP service every time a message needs to be processed. The handler runs the routing rules to determine the destination of the message and forwards message to that destination.
PSQCKSRV:	A server task as part of the Application Server implements quick searches in three tier mode ⁴ .
PSQRYSRV:	A server task as part of the Application Server implements adhoc database Queries. Users use Query manager to build queries and those queries are executed by PSQRYSRV.
PSRENSRV:	A server task used in real time event notification.
PSSAMSRV:	A server task as part of the Application Server; this is used to handle conversational transactions.
PSSUBDSP:	A server task as part of the Application Server's messaging system; this is invoked every time a message is subscribed (received) by a PS system from a PS / non PS system. The dispatcher assigns work to handler.
PSSUBHND:	A server task as part of the Application Server's messaging system; this is invoked by the PSSUBDSP service every time a message needs to be processed. The handler accepts the message and processes it so that the incoming message can be put in the database.
PSWATCHSRV:	A server task as part of the Application Server; the Watch Server process combines process details from the operating system and Tuxedo and executes a rule set on the resulting data. If a process from this domain is known to the OS and not known by TUXEDO it is killed. If a process has been handling a service request for longer than the service timeout, the process is killed.
Remote call:	A mechanism implemented by PeopleSoft to launch batch Processes that operate on a small set of input records.
SQR:	A codebase used by PeopleSoft to run batch reports.
USS:	UNIX system services.

FOOT NOTES

¹ object:	PeopleSoft uses the term "Tools object" to refer to tables, fields and Peoplecode.
² Metadata:	Metadata represents the structural information about objects such as online panels, table, fields and Peoplecode stored in the database.
³ Pageset:	A pageset represents either a tablespace or an indexspace in DB2
⁴ Three tier mode:	It is commonly used in development; a user (client) uses windows workstation to communicate with PS system through the PeopleSoft application server.
⁵ Two tier mode:	It is commonly used in development; a user (client) uses windows workstation to communicate with PS system directly without involving the application server.
⁶ PIA mode:	A user (client) uses a compatible internet browser to communicate with PS system through the web server.

REFERENCES

- 1.0 Cheryl Watson's Quick start policy at <http://www.watsonwalker.com/quickst.html>
- 2.0 z/OS V1R4.0 MVS Planning: Workload Management, SA22-7602-05, IBM manual
- 3.0 IBM whitepaper preserved at the internet location as indicated below:
<http://www.hewittandlarsen.com/documents/WLM/work%20load%20manager%20for%20ps.pdf>
- 4.0 MVS Performance Management, z/OS Version 1 Edition by Steve Samson.

COPYRIGHT ACKNOWLEDGEMENTS

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

ibm.com®
z/OS®
zSeries®
DB2 Connect™
DB2 Universal Database™
DB2®
IBM®
MVS™
OS/390®
System/390®

The following terms are trademarks of ORACLE Corporation in the United States, other countries, or both:

ORACLE®
PeopleSoft®
PeopleTools®

The following terms are trademarks of BEA Systems, Inc in the United States, other countries, or both:

BEA TUXEDO®

The following terms are trademarks of Novell, Inc in the United States, other countries, or both:

TUXEDO® is a trademark of Novell Inc, licensed to BEA Systems Inc.