Cheryl Watson's z/OS 101 Primer

2011

The z/OS 101 Primer

At the 2008 SHARE conference held in Orlando, I gave a lunchtime presentation called *The Tips Your Mentor Forgot to Mention*. It was for the zNextGen project, which was created to help those people who are new to mainframes or new to performance. The response to the project was incredible. One teacher even brought several students from his class in Georgia to the conference JUST to attend the zNextGen sessions.

Because of the project's success, and because several of our readers responded so favorably to my advice for new techies, I decided to include a new section in our Tuning Letters called "z/OS 101". Each z/OS 101 article addresses a topic that should be useful to those new to mainframes, especially in the performance, capacity planning, data center reporting, and charge back areas. They're also good as a review of the basics. If you'd like some special topic addressed, please let me know.

This particular document is a compilation of the 2011 z/OS 101 articles written so far and is offered to the public via our website. Please use it and copy it as you see fit, provided that you use the entire document if you distribute it, always credit us as the source, and make no attempt to resell it. The most recent compilation, as well as the 2009 and 2010 complications can always be found at www.watsonwalker.com/articles.html.

So, hats off to our next generation of z/OS techies, and welcome to the exciting world of performance!

z/OS 101

This series of articles is designed for those people who are relatively new to z/OS systems programming or performance. In the first article, I'll describe how and why the Systems Resource Manager (SRM) uses the 'mean time to wait' (MTTW) method in order to improve throughput in a system. The second article is a reminder about an important series of Redbooks.

Mean Time to Wait (MTTW)

All jobs and transactions that are run with a discretionary goal are managed according to their mean time to wait (MTTW). MTTW is the mean CPU time (adjusted to the speed of the processor) before the processor is released by SVC wait or other events, such as Pause. The processor is often released because I/O has been initiated on behalf of the application. Prior to Workload Manager goal mode (i.e. compatibility mode), you could specify the range of dispatch priorities and the definition of a significant CPU user using the CCCSIGUR parameter of parmlib member IEAOPTxx. CCCSIGUR was ignored in goal mode, and dropped from the documentation. z/OS 1.12, however, has resurrected that parameter.

Mean Time to Wait can improve throughput

Because many of our readers are not familiar with WLM compatibility mode, we think they will appreciate our review of both MTTW and CCCSIGUR. The rest of this article is a refresh of an article from Tuning Letter 1997 No. 4, pages 52-54.

Figure 1 – MTTW Ranges

DP	MTTW (ms)	MTTW (microsec)
	(155)	(2094-701)
C9	0- 5	0-7
C8	6-10	8-14
C7	11-15	15-21
C6	16-20	22-28
C5	21-25	29-35
C4	26-30	36-42
C3	31-35	43-49
C2	36-40	50-56
C1	41-45	57-63
C0	> 45	> 63

CCCSIGUR

The CCCSIGUR (CPU Significant User) parameter was designed to define the CPU intensity of jobs. The CCCSIGUR value is the average number of milliseconds (ms) between waits and is used to determine significant users of the CPU. The default of CCCSIGUR=45 indicates that a job that consumes more than 45 milliseconds between waits is a CPU-bound job. (A millisecond is .001 second.) This time was originally based on an IBM model 155 machine and is internally adjusted based on the machine speed.

SRM originally used the CCCSIGUR value for load balancing (which was seldom used, thank goodness!) and for determining MTTW (mean-time-to-wait) dispatch priorities (DPs). The load balancing function of SRM was eliminated in SP 4.2, but the MTTW calculation is still done. In goal mode, all discretionary workloads not belonging to a resource group are run in a single MTTW group that has a range of DPs from C0 to C9.

You can calculate the actual CCCSIGUR value for any machine by multiplying the 45 ms by the ratio of the SU/sec rate of the 155 (42.0) to the SU/sec rate of your machine. Use our CPU Chart to find SU/sec values. For example, the SU/sec for a z9-EC 2094-701 is 29520. The default of 45 ms for CCCSIGUR is really 64 microseconds (.000064 seconds) on a 2094-701 (45 * 42 / 29520).

MTTW Calculation

How to define MTTW dispatch priorities was described in our October 1991 Tuning Letter, but we'll describe it again. SRM takes the value specified for CCCSIGUR and segments it into ten categories of times. For the 155 with a default value of 45, this results in ranges of 5 msecs (0-5, 5-10, . . . ,40-45, over 45); for the 2094-701 it results in ranges of about 7 microseconds (simply divide the value of 64 microseconds by 9 to get the first 9 categories and add a tenth category for the amount over the default. Figure 1 shows the breakout for the default value of 45 ms on both a 155 and a 2094-701. The CPU-bound jobs will tend toward the low end of that range and the I/O-bound jobs will tend toward the high end of the range.

The purpose of the MTTW feature is to allow the I/O-bound jobs to complete before the CPU-bound jobs and thus improve throughput. An example to show how this works can be seen in Figure 2. If the I/O-bound job has the highest dispatch priority in a single-CPU machine, it will be dispatched whenever it needs to, and the CPU-bound job will have to wait until the I/O-bound job goes into a wait (not a very long time). If, on

Figure 2 - MTTW Dispatching and Throughput

I/O-bound job:

CCCwwwwwwCCCwwwwwwCCCwwwwwwCCC

CPU-bound job:

CCCCCCCCCCwwwwwwCCCCCCCCCCwwwwwwCCCCCCCCCC

Example 1: I/O-bound job has the highest priority

CCCwwwwwwCCCwwwwwwCCCwwwwwwCCC

dddCCCCCCdddCCCCwwwwwwCCCCCdddCCCCCwwwwwwCCCCCCCCCC

The I/O-bound job is not delayed, but the CPU-bound job is delayed by 9 cycles.

Example 2: CPU-bound job has the highest priority

CCCCCCCCCCwwwwwwCCCCCCCCCCwwwwwwCCCCCCCCCC

ddddddddddCCCwwwwwwdddddddCCCwwwwwwdddddddCCCwwwwwwCCC

The CPU-bound job is not delayed, but the I/O-bound job is delayed by 23 cycles.

Legend: C = CPU usage, w = I/O wait, d = delayed by higher priority work

the other hand, the I/O-bound job has a lower priority, it will have to wait behind the CPU-bound job (a considerable delay). Thus, placing I/O-bound jobs at a higher dispatch priority results in the least delay overall and the best throughput. In our two examples, when the I/O-bound job has the higher priority, there is an additional delay of about nine cycles. But when the CPU-bound job has the higher priority, there is an additional delay of 23 cycles.

SRM dynamically assigns dispatch priorities to each job in the MTTW group by calculating the average (mean) time between waits for each job. SRM will only start looking at a job after it's accumulated 200 ms of CPU time (based on a 155 machine). Until then, the dispatch priority stays in the middle of the MTTW group at 'C5'. After the calculation, the shortest times indicate I/O-bound jobs and get a higher dispatch priority. Longer times indicate CPU-bound jobs and get lower dispatch priorities.

Performance Considerations

Tuning CCCSIGUR could improve your throughput for any jobs with a discretionary goal. This is a small amount and won't produce a significant impact in your jobs. Realize that it's really only applicable in a CPU-constrained environment.

The only cost to tuning CCCSIGUR is the time to analyze it. The tuning effort may not be justified by the savings.

Measurement

Look at the actual dispatch priorities from any online system, such as the RMF Address Space Data (ASD) (type 79 records) or the ASD screen (column DP PR) shown in Figure 3 To find jobs running in the MTTW range, look for dispatch priorities of C0 through C9. Dispatch priorities from C0 through C4 are definitely CPU-bound jobs. Those in C6 through C9 are I/O-bound jobs. You can't tell anything about those ending in 5 because they may have just entered the performance group period. In our example, we see four address spaces in service class STCLOM, which has a discretionary goal. Because all four are between C1 and C3, we can assume that they're all relatively CPU-bound.

Figure 3 – RMF ASD
Address Space State Data

15:03:44		S	C	R	DP	CS
JOBNAME	SRVCLASS	P	L	LS	PR	F
MASTER	SYSTEM	1	NS		FF	2564
PCAUTH	SYSTEM	1	NS		FF	122
RASP	SYSTEM	1	NS		FF	213
TRACE	SYSTEM	1	NS		FF	370
DUMPSRV	SYSTEM	1	NS		FF	372
XCFAS	SYSTEM	1	NS		FF	1883
GRS	SYSTEM	1	NS		FF	1850
SMSPDSE	SYSTEM	1	NS		FF	4631
CONSOLE	SYSTEM	1	NS		FF	1880
WLM	SYSTEM	1	NS		FF	1401
ANTMAIN	SYSTEM	1	NS		FF	1366
ANTAS000	STCLOM	1	NS		C1	1259
DEVMAN	SYSTEM	1	NS		FF	184
OMVS	SYSTEM	1	NS		FF	6569
IEFSCHAS	SYSTEM	1	NS		FF	96
JESXCF	SYSTEM	1	NS		FF	645
ALLOCAS	SYSTEM	1	NS		FF	1270
SMS	SYSSTC	1	NS		FE	367
IOSAS	SYSTEM	1	NS		FF	352
IXGLOGR	SYSTEM	1	NS		FF	1303
AXR	STCLOM	1	NS		C1	462
CEA	SYSTEM	1	NS		FF	495
SMF	SYSTEM	1	NS		FF	484
LLA	SYSSTC	1	NS		FE	2410
VMCF	SYSSTC	1	NS		FE	204
JES2AUX	SYSSTC	1	NS		FE	162
JES2	SYSSTC	1	NS		FE	3208
VLF	SYSSTC	1	NS		FE	1550
SDSF	STCMDM	1	NS		F2	885
EPWFFST	STCLOM	1	NS		C3	339
VTAM	SYSSTC	1	NS		FF	2835
RRS	STCLOM	1	NS		C1	2399

If all the jobs fall at one end of the spectrum or the other, you are not getting the full benefit of the improved throughput. If they're all at the upper end (clustered in the C6 to C9 area), then it appears to SRM that they're all I/O-bound. Adjust this by decreasing the CCCSIGUR value (probably halve it for the first pass). If they're all clustered at

the bottom end (C0 to C4 range), then SRM thinks they're all CPU-bound. Adjust the range by increasing the CCCSIGUR value (perhaps double it or increase it by 50%).
Before a change to CCCSIGUR, measure the average elapsed time of your batch jobs and long users (such as TSO last period). Also obtain screen displays from your online monitor that shows the actual dispatch priority of your normal workloads.

Make the change, and then collect the same information. You should see an improvement in the elapsed time (if the system had been CPU-constrained), and a better distribution of dispatch priorities in the MTTW range.

The contents of this document are excerpted from

Cheryl Watson's
Tuning Letter

Published 6 times a year
by Watson & Walker, Inc.

Publisher: Tom Walker
Executive Editor: Cheryl Watson
General Manager: Linda May

2011 SUBSCRIPTION RAES: - Electronic version (DVD & email) $950 per year (single-site license), including DVD of issues since 1991. Multi-site discounts are available. Subscribe online at: http://www.watsonwalker.com.

Payment may be made by a check drawn on a U.S. bank or any major credit card. Send all correspondence to: Watson & Walker, Inc., 7618 Sandalwood Way, Sarasota, Florida 34231, USA. Tel: 800-553-4562 or 941-924-6565. Fax: 941-924-4892. For customer service, send email to admin@watsonwalker.com. For technical questions, send email to technical@watsonwalker.com.

WATSON WALKER

ABCs

All new system programmers should be aware of the 13-volume Redbook collection called *ABCs of z/OS System Programming.* IBM started a project in 2000 to summarize the skill set and information needed for new system programmers with a 5-volume Redbook called *ABCs of System Programming.* Then in May 2007, IBM started replacing these with a new series (this time adding 'z/OS' in the title). Here is a list of the current volumes in this important training series:

SG24-6981-01 - *ABCs of z/OS System Programming Volume 1* - Introduction to z/OS and storage concepts, TSO/E, ISPF, JCL, SDSF, and z/OS delivery and installation. (2-Apr2008)

SG24-6982-02 - *ABCs of z/OS System Programming Volume 2* - z/OS implementation and daily maintenance, defining subsystems, JES2 and JES3, LPA, LNKLST, authorized libraries, Language Environment, and SMP/E. (12Sep2008)

SG24-6983-03 - *ABCs of z/OS System Programming Volume 3* - Introduction to DFSMS, data set basics, storage management hardware and software, VSAM, system-managed storage, catalogs, and DFSMStvs. (11Mar2010)

SG24-6984-00 - *ABCs of z/OS System Programming Volume 4* - Communications Server, TCP/IP, and VTAM. (10Feb2011)

SG24-6985-01 - *ABCs of z/OS System Programming Volume 5* - Base and Parallel Sysplex, System Logger, Resource Recovery Services (RRS), Global Resource Serialization (GRS), z/OS system operations, Automatic Restart management (ARM), and Geographically Dispersed Parallel Sysplex. (20Feb2008)

Be sure to view the YouTube videos to keep them coming!

SG24-6986-00 - *ABCs of z/OS System Programming Volume 6* - Introduction to security, RACF, digital certificates and PKI, Kerberos, cryptography and z990 integrated cryptography, zSeries firewall technologies, LDAP, Enterprise Identify Mapping (EIM), and firewall technologies. (25Aug2008)

SG24-6987-01 - *ABCs of z/OS System Programming Volume 7* - Printing in a z/OS environment, Infoprint Server, and Infoprint Central. (16Oct2008)

SG24-6988-00 - *ABCs of z/OS System Programming Volume 8* - Introduction to z/OS problem diagnosis. (15May2007)

SG24-6989-05 - *ABCs of z/OS System Programming Volume 9* - z/OS UNIX System Services. (Updated 28Jan2011)

SG24-6990-03 - *ABCs of z/OS System Programming Volume 10* - Introduction to z//Architecture, zSeries processor design, zSeries connectively, LPAR concepts, HCD, and HMC. (15Sep2008)

SG24-6327-01 - *ABCs of z/OS System Programming Volume 11* - Capacity planning, performance management, RMF, and SMF. (7Dec2010)

SG24-7621-00 - *ABCs of z/OS System Programming Volume 12* - WLM (28Jan2010)

SG24-7717-00 - *ABCs of z/OS System Programming Volume 13* - JES3 (12Jun2009) ■

IBM z/OS YouTube Training

The Advanced Technical Skills team at WSC has created some excellent YouTube videos (5-15 minutes each). While these are all currently for WebSphere, we see the start of something wonderful. (Please note that if nobody watches them, they won't continue, so at least try them out.) See the WSC presentation, PRS4467, for links to the videos.

PRS4467 - *Advanced Technical Skills - YouTube Video Flyer with Hyperlinks*. (14Apr2011)

Currently you will find classes on:

- Introduction to WebSphere Application Server z/OS Version 8 (4 parts).
- WebSphere Application Server and WAS on z/OS (7 parts). An upcoming one is about WAS z/OS and WLM. That should be good.
- WebSphere Optimized Local Adapters (6 parts).
- WebSphere Compute Grid and Modern Batch (6 parts).

IBM Wildfire Workshops

IBM is hosting no-charge workshops about new technologies, such as Java and WebSphere. You simply need to contact your IBM rep to have them enroll you. The following workshops are currently provided, but the list may change:

- CCLX1 - Cloud Computing on zEnterprise and System z
- LXOR6 - Customizing Linux and the Mainframe for Oracle DB Applications
- WBSR7 - WebSphere Application Server for z/OS Version 7
- WMB07 - WebSphere Message Broker for z/OS Version 7 Workshop
- WSW07 - Security Workshop: WebSphere Application Server for z/OS
- ZPRT1 - WebSphere Portal Server on Linux for zSeries Version 6.1
- ZJAV1 - z/OS JAVA Exploiters and JAVA Batch Workshop
- VC001 - Virtualization and Consolidation to Linux of System z
- WMQ07 - WebSphere MQ V7 and WMQ FTE for z/OS Workshop
- ZWPS6 - WebSphere Process Server V6.2 for z/OS Implementation Workshop

The workshops are described on the WSC website (www.ibm.com/support/techdocs) under the Skills Transfer section on the left. Here is a direct link to their schedule and descriptions: http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/PRS1778 or click here. The schedule only goes up to July, but more workshops will be added. The summer is their slow period.

We know that your training budgets have been cut, but you still need training, especially for these newer technologies. This is a wonderful option provided by IBM. Even if you can't attend a workshop, the handouts are made available publicly on the WSC website. Just do a search for 'wildfire'. There were seven of them out there when I last checked. The material is fantastic.

IBM ATS Webinars

The Advanced Technical Skills (ATS) team started an interesting blog in December 2010 that describes past and upcoming no-charge technical webinars on storage technology. While this is provided under the IBM DeveloperWorks program, anyone can access the older webinars and subscribe to the blog to automatically receive updates.

The direct link is https://www.ibm.com/developerworks/mydeveloperworks/blogs/accelerate/?order=desc&maxresults=100&sortby=0&lang=en or click here. You may also access this by going to the DeveloperWorks website at http://www.ibm.com/developerworks and click on Community, then Blogs. A recent webinar was on *DS8000 Storage Configuration and Best Practises* [sic]. ■

Service Levels - Part 1

What are service levels? They're the level of service (in terms of response time and availability) that the data center provides to the users. In an ideal world, these are levels of service that users and the IT management have mutually agreed are necessary and sufficient. In too many installations, the level of service is not known, reported, or tracked. More important, and more common, is the situation where IT staff thinks they're doing a good job, but they are not meeting the users expectations. Users could be unhappy without anyone being aware of it.

Dissatisfaction or misunderstandings about the level of service provided by the data center is one of the main reasons that upper management takes the step of considering outsourcing alternatives. In other words, if you can't control your own data center or don't know what your users need, watch out or someone else will!

Many of our subscribers are outsourcers or customers of outsourcers. So many of their questions end up being related to service levels. But we're finding that the understanding of service levels is a little fuzzy these days. In the early 1990s, we wrote several articles regarding service levels. We think that the topic is even more important today than it was back then.

Service level objectives (SLOs) are the defining and management of service levels to an objective, such as 99% availability or an average CICS response time of .2 seconds. These are needed by every installation today. Service level agreements (SLAs) are agreements between the data center and the users on specific SLOs. These are often associated by financial penalties if the objectives are not met. These are needed in many installations, especially outsourcing environments. Additionally, both service levels and SLOs are monitored and used for performance analysis, capacity planning, data center management, and management reporting. They often have an impact on chargeback. But managing service levels is not an easy task. As you'll see, I strongly believe that having service level objectives are critical to the success of any data center, and are the basis for any successful capacity planning, performance, or chargeback methodology.

OVERVIEW

Defining, managing, and reporting service levels seem to many people to be a waste of time. So why I am recommending it so highly? I consider the setting of service levels to be ***the first*** and most important step in any capacity planning or performance project. Without them, an installation is operating without a plan or sense of direction.

DEFINITIONS

I first want to define the following terms: service level objectives, service level agreements, service level management, online response time, batch turn-around time, data access, availability, quality, cost, and load.

Service Level Objectives (SLOs)

"Service Level Objectives" or "service goals" generally refer to guidelines that are kept internal to the systems department. Service objectives are generally divided into the following types:

- Online response time
- Batch turn-around
- Data access
- Availability
- Quality of output
- Load or volume
- Cost (where applicable)

Defining SLOs is a critical first step

The first five items represent service provided by the data center to the users, while the sixth represents the load placed on the data center by the user. Tracking these elements gives the performance analyst an indication of the impact of any change to the system or changes in user requirements. Degradation in the first five items can lead to reduced productivity, lower morale of the users, and in some cases, a loss of business. To manage a data center properly, you must be aware of the level of service provided to the users and the level the users need to meet their business objectives. In an outsourcing environment, the contract will probably define the product or service as a function of each of the seven items.

Defining SLOs is the first critical step in managing your service levels. This step typically takes from two days to two weeks, and is the absolute least that you should do. Later articles will get into more detail about how to set each of the SLOs for each type of workload. Availability for CICS, for example, is defined differently than availability for batch.

Service Level Agreements (SLAs)

The difference between service level objectives (SLOs) and service level agreements (SLAs) is the user's participation. SLOs can be defined and tracked by the systems department and can be easily implemented within a few days. Agreements or SLAs, on the other hand, involve meetings with users, possibly training sessions with users, discussions, compromises, and commitments between the data center and the users. They can involve several people and can take as long as a year (maybe more) to implement. Service level agreements are the basis of most outsourcing contracts.

Service Level Management (SLM)

This term refers to the collection, reporting, and managing of either service level objectives or service level agreements. The reporting aspect is critical when dealing with service level agreements, because your users are going to want to know whether you've met your agreement.

Online Response Time

There are two definitions of online response time, and they differ by several seconds! The easiest time to collect is internal response time. This is the time from the moment the subsystem, such as CICS or TSO, sees the start of a transaction until the end of the transaction when it's sent back to the terminal. This is sometimes called "host" or "application" response time and is in the range of 0.01 second to 2.00 seconds. This is the easiest to collect and report and is therefore the most common form of online service reporting. It's almost always used for service level objectives. Some products, such as Netview, consider internal response time to be from the input to VTAM until it leaves VTAM. The Workload Manager (WLM) can manage service class periods to an internal response time goal, and thus provide great reporting and management for any service class managed by response time goals.

External response time (also called end-user response time) is meant to indicate the response time as seen at the terminal. This is typically in the range of 1.0 to 10.0 seconds. This is much, much harder to collect and report, but is the only one that is meaningful to the user. Most service level agreements attempt to provide some form of external response time measurements.

Batch Objectives

Batch has three types of objectives: 1) test batch, 2) production batch, and 3) ad hoc production. Test batch turnaround time is generally considered to be the time from job submission to job termination, excluding print time. There are variations on this, such as using only the time until an initiator is started. Most installations have different turnaround objectives for each test job class.

Production objectives vary from installation to installation, primarily because they are very difficult to quantify. Most installations define some critical production jobs and set an objective for either job completion or printing completion.

More and more installations are having to deal with ad hoc production jobs submitted by the users. These might be BMP jobs, or TSO and CICS batch submission jobs. The jobs require production level response times because they are critical to the users, but they aren't managed by a scheduler. These can normally be handled as very high priority test batch jobs.

Data Access

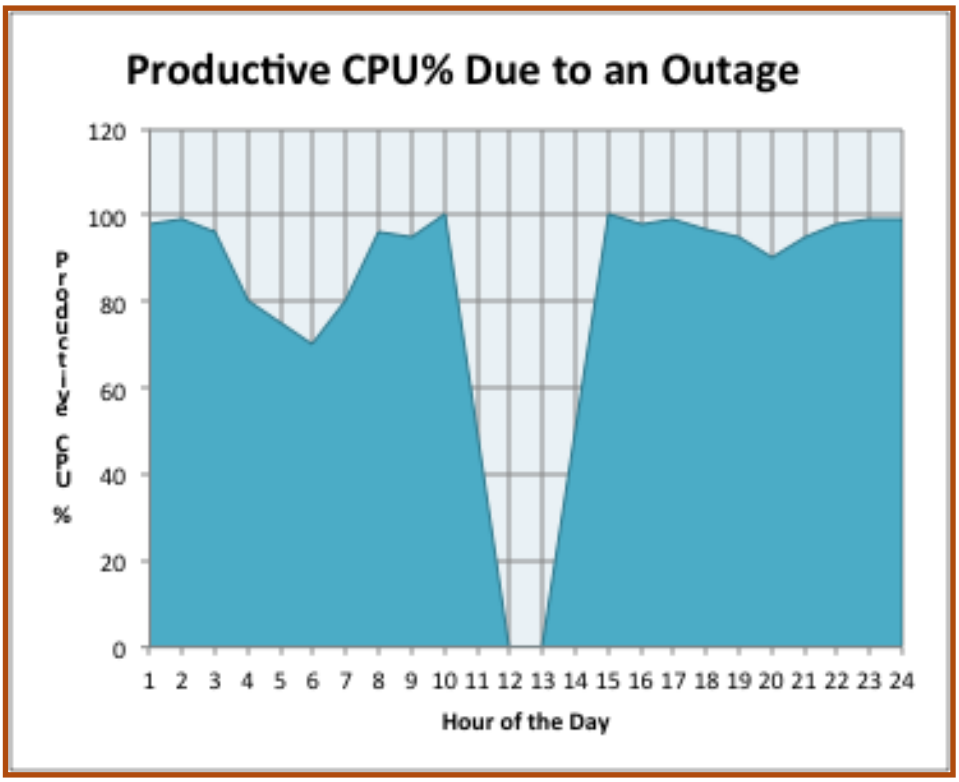
System Managed Storage (SMS) has brought a greater awareness of the importance of the users' access to their data. More and more installations are adding service level objectives to address timely backup, recovery, migration, tape retention, tape accessibility, and retrieval from migrated data sets.

Availability

Most installations keep track of z/OS availability (i.e. the time that z/OS is up and running). While this is fairly easy to collect data on, it doesn't really mean much to the users. For online users, if the communications system (e.g. TCP/IP), the online subsystem (e.g. CICS), the application (e.g. payroll), the database (e.g. payroll master file), or the network (line, modem, controller) is down, then the system is unavailable. Many of these conditions are difficult to measure and quantify. For example, would you consider the system 'available' if two out of ten lines were down?

Not only is the percent of available time important, but so is the number of downs and the length of each down. In addition to percent availability, many sites choose to collect mean times to failure (MTFs) and the lengths of each downtime.

Figure 4 – Productive CPU% Due to an Outage



Availability can be collected and reported in several ways. After all, when the system comes down, you lose availability in three time periods: 1) before the crash, 2) during the crash, and 3) after the crash. A typical chart of productive CPU usage for a system outage is shown in Figure 4. This is a case where the system crashed at noon. But

you've really lost much of the effective time prior to the crash due to work that will need to be redone (e.g. files that weren't saved, jobs that need to be rerun, databases that need to be recovered, TSO work not saved, transactions that weren't completed). You lose the entire time that the system was down - let's assume it was an hour because dumps were needed, and automatic re-IPL wasn't in effect, but was instead initiated by an inexperienced operator who made mistakes. And then you lose effective CPU after the system becomes available. Just because z/OS is back up doesn't mean that the subsystems are back up. And if that takes an hour, it's unlikely that the users will know that it's back up - they've given up and gone on to other work. There is a built-in delay there. There's also lost time due to the fact that everyone has to log onto the system again. And then there's lost time because they have to recover to the point of the crash. Surely you've heard the cries of "I've lost two hours worth of work!!!." There's no good way to measure this "re-do" time. It's "uncaptured" down time. Just be aware that each "down" is a loss of more work than is recognized by the pure availability numbers.

95% availability in a 40-hour week is unacceptable

Much of this lost time is also present for a scheduled outage. If the machine is going to be IPLed at 6 am, the batch initiators will be stopped long before that, possible delaying work until after the outage. But the CPU will be running well below the capacity of the system. Online users may lose work if they don't log off in time.

All this goes to point out that the system availability itself is not an adequate measure. Not only should you care about the amount of down time, but also the frequency of the down time. As an example, assume that the prime shift is an eight-hour a day, and you have a service objective of 95% availability. It sounds good, but it really isn't. 95% availability in a 40-hour week is two hours of downtime! That has a severe impact on the users. Ok, so let's make it 99% availability. That's still 24 minutes a week, which is probably equivalent to three hours of lost time. Now if you can deal with that, consider whether you can live with 24 outages of one minute each. You'll have all the users pounding at your door.

The point to this is that you should probably have two availability objectives: one for total availability (e.g. 99.5% for prime shift), and one for the number of outages (e.g. less than two per month). A third objective might be the maximum length of any outage. You could, for example, set the following objectives for monthly CICS availability for 07:00 through 19:00, Monday through Saturday:

99.5% availability.
No more than 3 outages of between 0 and 30 minutes.
No more than 1 outage of over 30 minutes.

Quality

An often-overlooked aspect of service levels is the quality of the service. For example, CICS users may get great response time, but the applications keep abending. They won't be satisfied because of the quality of the service. Or batch users get great turnaround, but half of the listings are lost between the printer and delivery. Production jobs are greatly affected by a large number of reruns. This aspect is somewhat difficult to measure, but certainly worth tracking.

Load

Load is another frequently ignored portion of service level objectives and agreements. The service objectives of response time, turnaround time, and availability are really dependent on the volume of transactions or jobs and the nature of the jobs. As an example, if a user submitted fifty jobs with the same name, JES might single thread them and run one at a time. The jobs would miss their objective.

In another case, suppose you set an objective of class S jobs being completed within 15 minutes of submission. But this goal is dependent on some volume of work, such as 10 jobs an hour, 100 jobs an hour, or 1000 jobs an hour. If the objective had been set assuming 100 jobs an hour and one user submitted 500 jobs at one time, the service objective could not be met. The same is true for online transactions. Therefore, part of service level management is defining the load and tracking it over time.

In an outsourcing environment, the load is critical. If a customer exceeds their intended volume for a day, not only will that customer receive poorer service, but it could also impact other customers.

COST

Depending on the installation, cost may or may not be included in the service level objectives. When it is included, the intention is to provide a level of service for a specific cost. In fact, the data center can provide different levels of service for different costs, and let the users select their own level of service. For example, you might have two rates for a class S job; one with standard priority and one for a user requesting a higher priority. Discounts can be applied for work run after prime shift and penalties applied for work exceeding the objective load.

On the other hand, the data center may need to give a discount to any user who doesn't receive the agreed upon level of service, either with response time or availability. This is a basic element of any outsourcing contract. I know one outsourcing vendor that guarantees their availability on a daily basis. If the objectives aren't met, the user doesn't have to pay for the entire day for that service. That vendor seldom, if ever, misses their objectives!

In another use of costs, you could set thresholds that define the range of costs for a service, such as a $2 to $4 for a COBOL compile. If the costs exceed that range, then the job would be identified for further analysis. Either the job exceeded the limits set for COBOL compiles or the costs were increasing with no reason.

PHILOSOPHY

Defining and tracking (trending and reporting) service level objectives is absolutely necessary in order to manage any installation today. You can't simply wait for service to deteriorate to the point of receiving complaints. Users will be unhappy and lose faith. One of the biggest reasons today that some installations are taken over by outsourcing vendors is that service levels haven't been established and tracked. The data center staff doesn't see user dissatisfaction until it's too late.

Also, I don't really see how anyone can do capacity planning without service level objectives. I once heard a statement that I whole-heartedly agree with: "Without service level objectives, you have unlimited capacity." Yep - that means you could run 500 TSO users, 5000 CICS users, and 500 batch jobs on a baby z9 without service level objectives. Of course, response time could be measured by a sundial, but capacity is available for the work (eventually!).

Without SLOs, you have unlimited capacity

Everyone has service levels - they're measured by the telephone complaints received. I've heard overworked performance analysts say that they tune the system when someone calls and complains about response time! Users take note!

AVERAGES

I'd be doing you a disservice if I didn't expound a bit on the subject of "AVERAGES." I dislike and distrust any "average", but unfortunately I must often use them. For many measurements, it's the only value available. But I'd like you to have a general wariness about them before you try to prove anything with them! Figure 5 shows an example that I used to present in my Capacity Planning class. It shows ten response times for each of four users. Assume for a moment that you're one of the users and can see the screens for the other three users. Imagine what type of feeling you'd be getting.

Figure 5 – User Response Times

USER1	USER2	USER3	USER4	
1.7	1.0	1.0	.3	
1.2	.5	2.0	.3	
1.7	1.0	.2	.3	
2.5	1.0	4.0	.3	
1.2	9.0	.3	.3	
.9	1.0	4.0	.3	
1.7	1.0	4.0	.3	
2.2	.5	.3	14.3	
1.7	1.0	.2	.3	
2.2	1.0	1.0	.3	
=====	=====	=====	=====	
17.0	17.0	17.0	17.0	TOTAL
1.7	1.7	1.7	1.7	AVERAGE
.48	2.54	1.53	4.20	STD DEV
10%	90%	60%	90%	% WITHIN 1 SECOND
70%	90%	70%	90%	% WITHIN 2 SECONDS

Notice that the "average" for all four users is the same, 1.7 seconds. Most people respond in the following manner - Most everyone would want to be USER4 with the very quick response time, and an occasional long delay. USER2 is the next most desirable with an apparent average of 1.0 second and an occasional delay. In reality, USER1 gets the most consistent response time, which is best for productivity. Almost nobody wants to be USER3 with the erratic response time that you can't depend on. Since consistency and short response times are important to users, what does an "average" really tell you? The answer is: not much! Unfortunately, it's often the only thing we've got. Its primary purpose is to show changes. For example, if the "average" increases by 25%, you would expect users to start complaining. So it's best used for a comparative evaluation and not really an indication of what the users are seeing. Averages are typically used for availability.

Standard Deviation

One method of interpreting the averages is to use the standard deviation, which is an indicator of the variability of the responses. The higher the standard deviation, the less the actual responses cluster around the mean. The actual standard deviation requires two passes of the data; one to determine the average and the second pass to calculate the standard deviation. This could take a tremendous amount of time if you're processing millions of transactions per day. The calculation for standard deviation is as follows (where n represents each observation and "r" is the response time):

$$\text{sdev} = \sqrt{\frac{\sum_{i=1}^{n}(r_i - \text{avg})^2}{n}}$$

A shorthand method can be used with a single pass. At the same time you collect the response time, you collect the square of the response time. After the data has been accumulated, the average is calculated by taking the sum of the responses and dividing by the number of responses. The difference between the square of the average and the average of the squares is the standard deviation. Don't worry about it, just realize that it's an approximation. The calculation for this approximation is:

$$\text{sdev} = \sqrt{\frac{\sum_{i=1}^{n} r_i^2 - \frac{\left(\sum_{i=1}^{n} r_i\right)^2}{n}}{n}}$$

RMF collects and reports this estimated standard deviation for all response and turnaround times. Figure 6 shows an extract from two different RMF Workload Activity reports for TSO service class periods. The first example shows an average response time of .663 seconds with a standard deviation of 1.85 seconds, while the second example

shows an average response time of .333 seconds (about half of the first), but a standard deviation of 5.764 seconds. For the standard deviation to be so high on the second example, the response times must be extremely variable, which cause dissatisfied users.

Figure 6 – Extract from RMF Workload Activity Report

```
Happy Users:

    TRANS.–TIME HHH.MM.SS.TTT
    ACTUAL                663
    EXECUTION             663
    QUEUED                  0
    R/S AFFIN               0
    INELIGIBLE              0
    CONVERSION              0
    STD DEV             1.850

Unhappy Users:

    TRANS.–TIME HHH.MM.SS.TTT
    ACTUAL                363
    EXECUTION             363
    QUEUED                  0
    R/S AFFIN               0
    INELIGIBLE              0
    CONVERSION              0
    STD DEV             5.764
```

The most important thing about the standard deviation is that it shows consistency (or lack thereof) in response times. Consistent response times are even more important than short response times for productivity and user satisfaction.

PERCENTILES

A preferred, and most common, method of reporting service objectives is the percentile method. It answers a question like, "how many responses were 1 second or less?" Now look at the analysis in Figure 5. Both USER2 and USER4 had 90% of their transactions complete within one second. This is a much better technique for evaluating response times, but is only available if you either collect every individual response time and sort them or if you pre-define the response categories. That's why the introduction of Workload Manager (WLM) was so important in the mid-1990s. For response time goals, WLM creates response categories so that you can identify what percent of responses fell into certain categories, such as 86.7% within .5 seconds. This measurement technique of percentiles is a much better way of reporting any response time objectives. In z/OS 1.13, WLM adds the ability to identify percent response times for work with velocity goals, as well as response goals. We'll discuss those in more detail in future articles as we cover SLOs for different types of work.

COLLECTION PERIOD

The period over which the objective is collected is extremely important. The longer the period of collection, the lower the average response time and the higher the availability. You can choose to set the objective for peak hours only, for prime shift, or for 24 hours. I highly recommend using peak interval for response times and prime shift for availability, even though most sites use prime shift for both response times and avail-

ability. Avoid averaging for a longer time period than an hour, or the data will be meaningless. You also might consider different objectives for each shift.

Let's take an example. An average TSO response time could be .3 seconds for a 24-hour day or 1.0 second during peak hours. The users only really care about the peak hours. Reporting .3 seconds doesn't address the users perception of the system. In this case, reporting on peak hours or peak intervals only would more closely match what the users see.

If you want to have SLOs for other than prime shift or peak hours, then define them separately. For example, you might define the following:

99.5% availability during prime shift
95% availability for non-prime shifts.

Percentile response SLOs are much better than average response SLOs

Another consideration is attention to the business cycle. Many businesses have their peak periods during a specific time during the month (such as the last two business days during the month). Often the load is double the activity from other periods during the month. You might want to establish different service levels reflecting the business cycles.

REPORTING PERIOD

Once you've established your collection period, you should determine your reporting period. What is the service level objective trying to reflect? Typical reporting periods provide daily objectives, weekly objectives, or monthly objectives. The longer the reporting period, the better the service looks.

Let's assume you had an objective where the prime shift (8 hours per day) availability should be 99%. In any given day, that means a loss of 5 minutes is acceptable. During a month of activity, however, this means 1.7 hours of downtime is acceptable. There's a big difference between losing the system for 1.7 hours on one day a month and having the system crash every day for 5 minutes. A solution is to add some additional objectives. For example, your availability objective might be defined as:

98% availability
No more than 1 down time per day
No more than 30 minutes down time at once

For response times, you might include the following:

90% of all transactions completing within one second
No period over 30 minutes exceeding the response objective

TYPES OF WORKLOADS

You often need to divide your workload to allow different objectives for each type of workload. For example, online transactions could be very short transactions, such as a display of a record, or a simple calculation. Other transactions could be very long and involve long browses of several hundred records or significant computations. You certainly wouldn't want the same objective for all transactions. Typically, sites define short, medium, long, and (sometimes) excessively long transactions based on resource usage. Then they set objectives for each type of transactions. For example:

.7 second average response time for short TSO transactions
2.4 second average response time for medium TSO transactions

90% of short CICS transactions within .8 seconds
80% of medium CICS transactions within 2.0 seconds

Test batch jobs are typically divided by jobclass. When you get to service level agreements, you may want to define different objectives based on application, user, or accounting code. It's very common to have different TSO objectives for users who are primarily using one specific product. For example, you might have longer response objectives for FOCUS, SAS, or SQL users than you would for ISPF editing users.

In this article we've tried to provide an overview and description of service level management, including the differences between SLAs and SLOs. In our next article, we'll get into the specifics of ways to define the SLOs for each type of workload. ■