What is Good Monitor Data?

Bill Bitner

VM Performance Evaluation
BITNER at GDLVM7
TL 852-6022

October 1992

Introduction

- The current Monitor is lacking in several ways.
  - More data collected than in 370.
  - Less information is available.
- Objectives - answer the following:
  - What is the VM Monitor?
  - Why is instrumentation and monitoring needed?
  - What needs to be instrumented or monitored?
  - How to monitor the needed data?
- Monitor (Websters) = A person or thing that warns or instructs.

5th Edition 1

In VM/SP and VM/HPO, a performance analyst could often ask for a monitor tape and have everything he/she needed to analyze a performance problem. In VM/XA and VM/ESA, we often find ourselves asking for Monitor data and RTM data, and then finding we still don't have enough information. The amount of Monitor data collected in VM/XA and VM/ESA is huge compared to VM/SP and HPO. Unfortunately, it is often "bad" data or incomplete. In fairness, note that VM/XA is more complex than SP or HPO, but that's all the more reason to provide good monitor data.

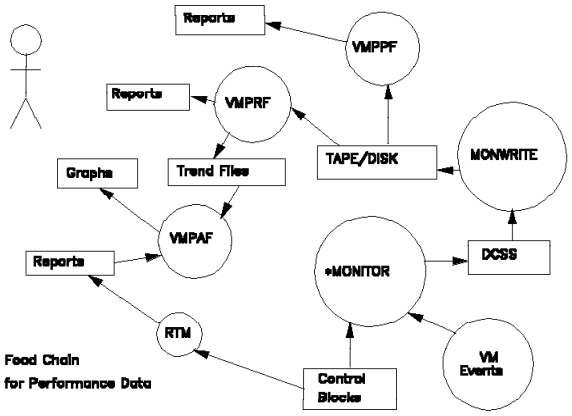
Part of the problem is in where the responsibility lies for getting good data into the Monitor. Everyone owns a piece of it. The Monitor subsystem Developers have an obvious role. The other subsystems are the only ones who really know what's in their subsystem and how it works. The Performance Evaluation area knows how Monitor data is used and what is needed. All the groups must work together.

This presentation grew out of a need to educate people on the importance of monitor data and what good data is. I consider this preventive medicine. At the risk of sounding trendy, lets call it Defect Prevention.

Listed are four questions that, hopefully, will be answered in this presentation. They are the ones that come up most often and are key to this whole concept. I'll warn you that not all your questions will be answered here. Part of the goal is to get you to ask questions.

The definition is here to jump start some thinking. The Monitor is not just a bit bucket for number crunching. It is meant to be a trusted aid to the system programmers.

Performance Data Food Chain



As in life, any weak link in the chain can be harmful to the whole system.

5th Edition 2

This may seem a little silly, but if you think about it isn't. Performance analysis is a complex business/system. This picture illustrates only a piece of the whole. Future foils will discuss the different ways that the end product (reports and graphs) are used by the analyst or customer.

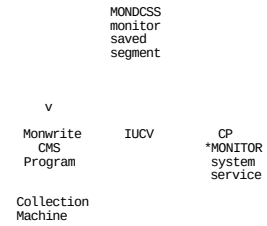
It all starts with the lower right corner in the "control blocks" and "VM Events" that make up a system. Also included in VM is the Monitor system service and the Monwrite application. The former creates the data and puts it in the DCSS, where MONWRITE can get at it.

Shown are IBM products that work off of Monitor data or data created from Monitor by other products. RTM does not use Monitor, but is there since it is a key tool in the performance world. It deals with data in "real time" fashion. There are vendor products which serve the same function, but use monitor data.

Not included are a number of internal tools (Cache Aid Analyzer?, READMON, MONVIEW, etc.) and a multitude of vendor products.

What is the VM Monitor?

Domain	Name	Event	Sample	HiFreq
0	System		Sample	Hi Freq
1	Monitor	Event	Sample	
2	Scheduler	Event		
3	Storage	Event	Sample	
4	User	Event	Sample	Hi Freq
5	Processor	Event	Sample	
6	I/O	Event	Sample	Hi Freq
7	Seek	Event		
10	Appldata	Event	Sample	



5th Edition 3

The Monitor is another CP System Service, like *BLOCKIO and *MSG. The monitor contains different classes of data called ″domains″. Each domain can have different types of data (Event, Sample, HiFreq). Event data are records cut for individual events, Sample is data collected at periodic intervals, HiFreq is data that is sampled at high frequency and then collected (records cut) at the normal sample rate.

The CP MONITOR command is used to enable the various Domains. Selectivity exists for SAMPLE (sample and HiFreq) and EVENT, Domains, and in many cases individual userids, device classes, or device ids. The CP MONITOR command is also used to START or STOP CP′s collection of this data.

The figure shows the key pieces. DEFSEF and SAVESEG commands are used to create a segment (MONDCSS) that is writable by CP and readable by an application. MONWRITE is a CMS module shipped with VM. It will use IUCV to connect to the *MONITOR system service. CP will use IUCV to signal MONWRITE about monitor data being placed in the segment and MONWRITE notifies CP about its status. MONWRITE will move data from the segment to disk or tape (see Performance Data Food Chain) picture.

Why Monitor?

- Why ask why?
  - Would you buy a car without any instruments on the dash board?
  - Would you put your money in a bank which can't report the status of your account?
- Manage normal system operation (includes tuning).
- Validate/justify investment and how it is being used.
- Plan for future changes.
- Debug Tool
- Validate the design.

5th Edition 4

The Monitor is a key ingredient to customer satisfaction. People want to know what their system is doing or where the end users can be helped the most. New functions are better received when there are methods to tell how they are being used and impacting system performance.

Monitor data is used for problem or trend analysis, capacity planning, and modelling. It has also been used at times to point out flaws in parts of the VM design. These uses will be illustrated on future foils.

As the complexity of systems grows, the need for accurate and appropriate data increases as well.

System Management

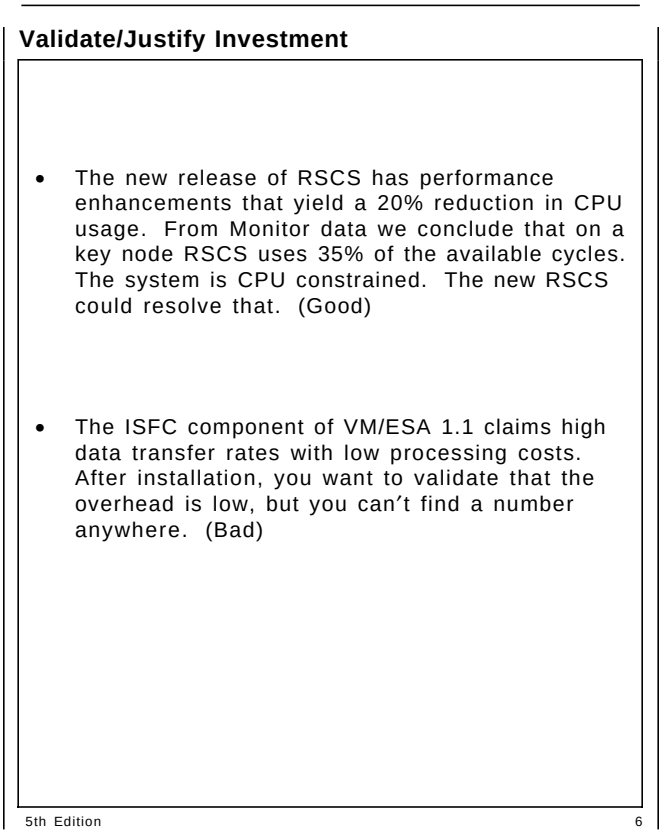
- The concept of Service Level Agreement has been around for years. This is where I/S promises to give the end users some Level of performance and/or availability. We all like guarantees. Now how do you measure it?
- Have you ever said ″Why is the system so slow today?!!″
- Response time on system is horrible for users, particularly SFS users.

VMPRF Report (subset)

	Percent of True Non Dormant Time								
	Waiting on								
						SVM and			
Userid	Running	CPU	Page	Inst Sim	Test Idle	Cons Func	Test Idle	Eligible	Dormant	Other
VSCSXA2	3.8	10.5	29.1	2.7	18.9	0.0	35.1	0.0	0.1	0.0
RWSERV1	1.8	6.3	60.3	26.4	0.1	0.0	3.0	0.0	0.0	2.0
RWSERV2	2.0	6.1	67.2	20.7	0.0	0.0	2.2	0.0	0.0	1.8

The questions in the first two bullets are things most of us can relate to. Dealing with those questions is much easier when good performance data exists.

The last bullet is an example of where monitor data points out the cause of a severe performance problem. Shown is a cropped VMPRF report (to get it to fit on the page). Users on this system were experiencing horrible response times. Through analysis of the VMPRF reports it can be seen that key server virtual machines were spending enormous amount of time in page wait. RWSERV1 and RWSERV2 are SFS File Pool server machines. This was a storage constrained system. Use of the CP SET RESERVED command for the three server virtual machines improved response time by over an order of magnitude. The number of pages to reserve were also based off monitor data from another VMPRF report.



The first case is an example where monitor data is invaluable in helping justify the expense of upgrading RSCS. The current economies make it necessary to have concrete justification for software and/or hardware expenses. I actually made this example up, since I really never do pricing stuff. You can trust me that this is a real world activity.

The second example is where lack of good monitor data could hurt us. This isn't a knock against the ISFC Design/Development team. They did a super job of integrating this into VM/ESA 1.1. It is a classic example of lack of communication and/or education. They really don't understand what the Monitor is all about. (Hopefully, they will after this presentation.)

Many customers (or their managers) are very suspicious of performance claims. They all want to see it for themselves. They want to know what it costs in CPU cycles to move a meg of data. Sometimes you need to count the little things just to prove they are little. And when problems occur is sometimes because the little things are no longer little.

Plan for Future Changes

- Your site is adding 100 users to system. Can it handle them?
- New application is being added to system that is said to double working sets. How will this affect system performance?
- You are replacing the 3090-300 with a 3090-600. How will this affect system performance?
- System CPU requirements are growing at 10% a quarter. How long before we out grow the current processor? Will any other bottlenecks be hit first?

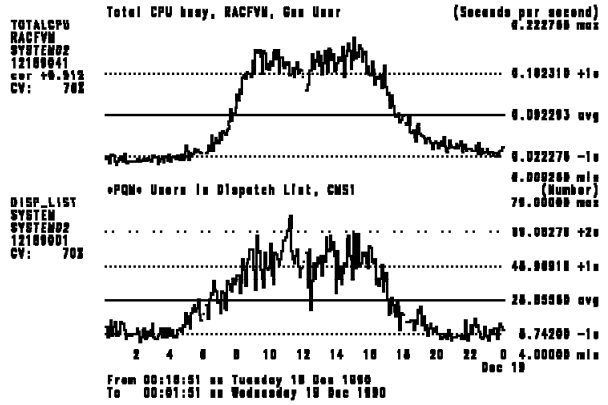
All of the above changes can be modeled with VMPPF (VM Performance Planning Facility). VMPPF uses monitor data as the key input to the model.

5th Edition 7

These questions come from the capacity planners and modelers. The data processing at many customer shops is growing at a large rate. It takes a lot of work to manage something growing that fast. The more information they have to deal with it, the better.

Listed are some typical questions that may be asked of the capacity planner. VMPPF is one tool they may use in order to help answer these questions. VMPPF is a powerful product. Like most modeling tools, it is only as good as the data provided it.

Debug Tool

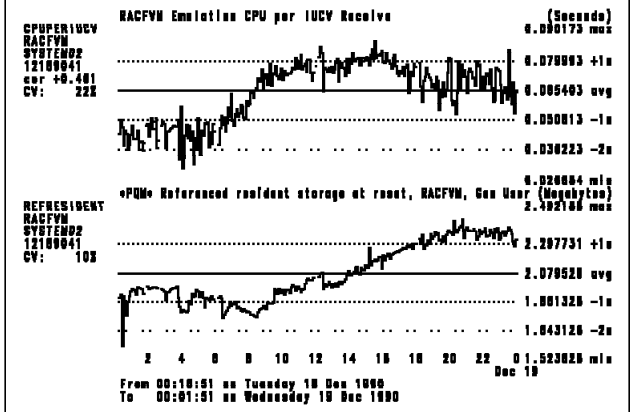


How monitor data helped uncover some bugs. Customer complained of poor performance. Degradation matched time frame when RACF installed.

5th Edition 8

This graph shows RACFVM CPU usage on top and number of users in dispatch list on bottom. DISP_LIST is used here as a measure of performance. Others could have been used, but this matched end user perception. Using VMPAF, the analyst could ask for a search of performance variables that fits the DISP_LIST curve. RACFVM CPU time was in the list of candidates.

Debug Tool ...



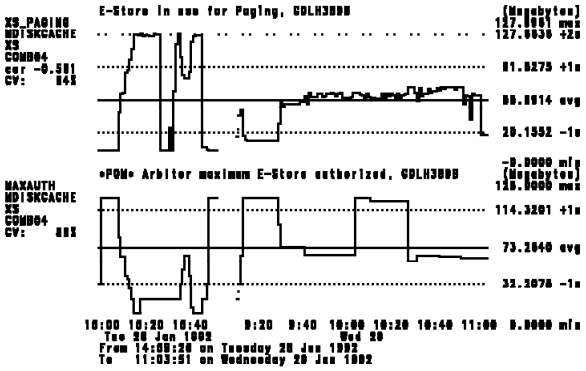
5th Edition 9

Narrowing in on RACF, two more graphs are drawn. The bottom graph is measure of resident storage for RACFVM. Strangely, it seemed to reference more storage as the day progressed even though activity on system slowed down. The top picture is CPU time normalized to the IUCV receive rate (since that's how RACF gets its requests). Over time RACF uses more Emulation CPU per request.

Graphs like these convinced the RACF developer involved that there had to be storage fragmentation problem.

This analysis lead to 4 APARs (2 RACF, 2 VM). There was some dump reading and trap writing required, but that first big step of knowing it was a storage problem was provided by monitor data.

Design Validation Example



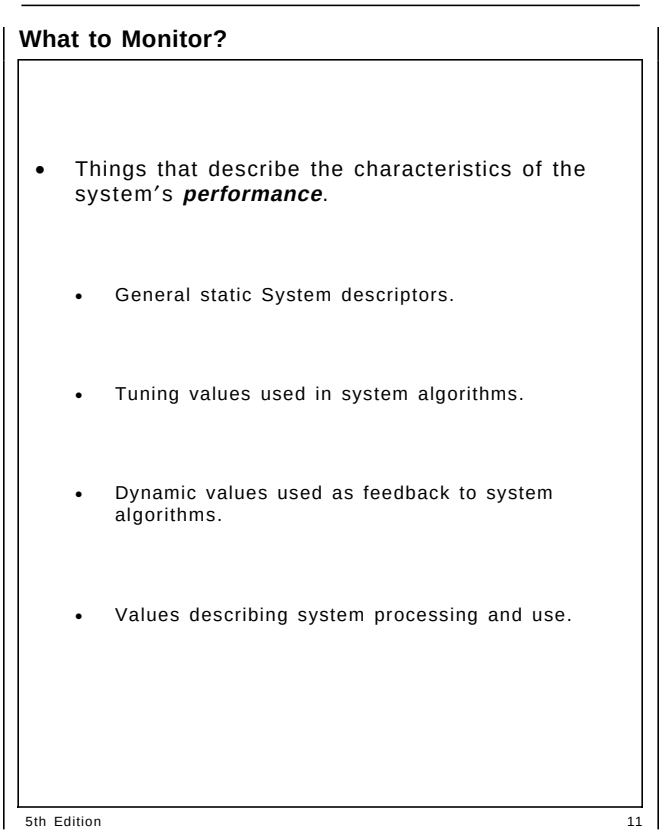
Example of monitor data showing impact of MDC arbiter changes made in VM/ESA 1.0. Lines on left half are VM/XA 2.1; on right half VM/ESA 1.0.

5th Edition 10

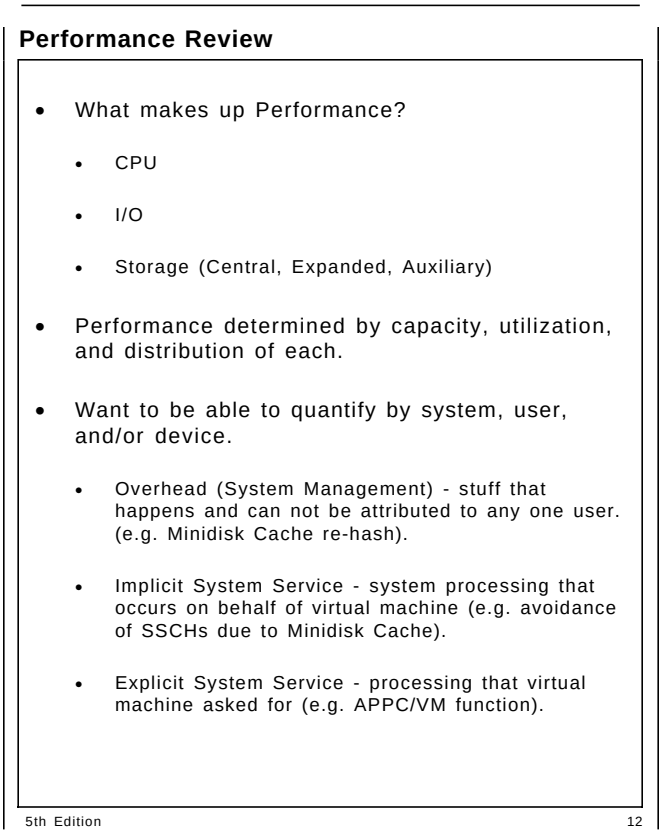
This is part of work that came out of analysis of a VM/ESA 1.1 customer concern that the MDC Arbiter was making improper decisions. Virg Meredith created above graph based on data from a test case Joe Tingley ran. On the left half of the graphs is VM/XA 2.1, on the right side is VM/ESA 1.0. The upper graph shows expanded storage in use for paging and the lower graph shows expanded storage in use for minidisk cache.

In both systems, you see MDC starting out with all of expanded storage. Then some super paging jobs are started. In VM/XA 2.1, you see paging quickly begin to gain expanded storage for its use, to the point where it needs almost all of it. This doesn't happen in VM/ESA. Paging only gets about 1/2 the expanded storage. The experiment was run twice on VM/ESA and the results were the same. Work was put in place to re-evaluate this design consideration.

This foil is the end of the "Why Monitor?" question. Take a deep breath and go on to "What to Monitor?".



Listed are four groups that data could fall into. I'll admit there are some gray areas. The order is along the lines of what the system programmer has control over. Each area will be discussed in detail to follow with examples that show what these mean to a customer. But before we do that, there is a foil or two on what is "performance"?

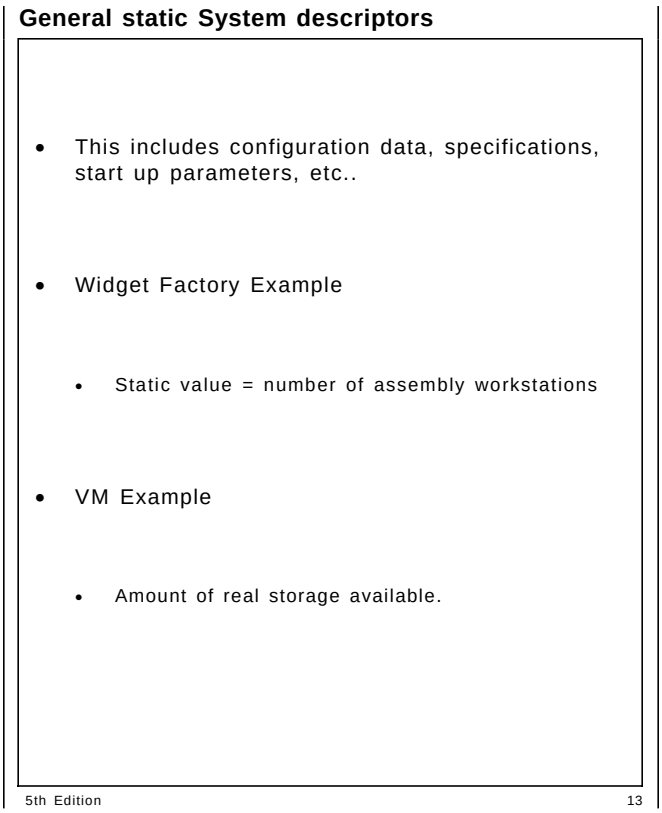


This foil was added after the first presentation of this material due to requests to explain what ″performance″ really was. I wish it was as simple as a single foil (then again, I might be out of a job). These are just the highlights.

The basic building blocks are CPU, I/O, and Storage. Your performance is determined not only by the capacity and utilization, but also by distribution. A chain is only as strong as the weakest link, so you have to know what will bottleneck first.

Distribution is between virtual machines and/or devices. Is a subset getting/using too much/little?

Listed are three areas that are not truly distinct groups, but they blend into each other. The first, overhead, is what many people want to be zero. The second, implicit, needs to be measured to tell what is ″accidentally″ happening in the system. Explicit service are things we know about. They need to be measured in order to tell where the problem is.



These are the things that typically don't change much in the system, hence the name "static". How was the system configured? What hardware is involved? What initialization parameters are used or determined during initialization? The system programmer usually takes care of these things before the system is IPLed.

These are called "static" here, but it is very important to also note if they change somewhere in time. Also, these may include things we just assume. For example, 3990-3 with 64Mb installed, but CE only enabled 32Mb (true story).

I'll use this Widget Factory example throughout the foils. The Factory has a number of workstations upon which a widget can be built. Each day you expect the same number of workstations, but a day may come when one breaks down. This is important! A performance analyst spends most of the time trying to figure out what changed.

There are all sorts of examples in VM. Real storage is just one example. A case of missing data in VM/XA monitor for this area is the model number of the processor. The monitor will tell if it's a 3090, but not whether it's an S, J, etc.. The model number (version code) was finally added in VM/ESA 2.0.

Tuning values used in system algorithms

- Think of all the software knobs in a system, both commands and parameter files.
- The settings on those knobs are important.
- Widget Factory Example
  - Tuner = Salary. The more you pay the workers, the faster they build widgets.
  - Tuning Value = Pay scale value.
- VM Example
  - Tuner = SET SRM DSPBUF command. Tuning knob for maximum number of users allowed in the dispatch list.
  - Tuning values = SRML1DSP, SRML2DSP, and SRML3DSP fields of HCPSRMBK

5th Edition 14

The system programmer can affect performance with a variety of CP commands (tuning knobs). If you're the performance analyst, you are very interested in the setting of these knobs. The system programmer, may not know (or remember) what was changed. It is nice to have numbers to validate it.

In the Widget factory, one knob is the wage scale. The better you pay, the harder they work, the more widgets produced. Like many knobs there is a point of diminishing returns. To some one analyzing the factory, they would be interested in salaries.

In the VM Example, we see something missed when a tuning knob was added to VM/XA. SET SRM DSPBUF controls the number of virtual machines allowed in the dispatch list for the various classes. Performance is affected by the number of users in the dispatch list (and where users are if they aren't in the dispatch list). The DSPBUF settings were added to monitor in VM/ESA 2.0.

Dynamic Values used as Feedback

- There are values used in system algorithms which are not attached to knobs, because they are computed on the fly. Can often be used in design validation steps.
- These are often out of control of system programmer, but very important to how system runs.
- Widget Factory Example
  - Algorithm = The colder it is the faster the workers move in order to keep warm.
  - Feedback value = reading on the shop floor thermometer.
- VM Example
  - MDC Arbiter = The greater the need for Virtual I/O over Page I/O, the more expanded storage should be given to MDC.
  - Feedback value = HCPARBCE and HCPARBPG which are counts of outstanding cache eligible I/O requests and Page I/O requests, respectively.

5th Edition 15

The system programmer has pretty much lost control at this point. Here we are talking about values that are determined by interactions within the VM system. Think about key values used in system decisions. If they are important in deciding what VM is doing, they are important to record for the analyst who must figure out what VM is doing.

In the Widget factory, the workers are affected by the temperature on the shop floor. This can be controlled to some degree by the heating/cooling system, but it isn't a very well constructed building and the outside temperature is a big factor.

The VM example was uncovered when looking at the design validation mentioned earlier. We knew what the MDC arbiter was using to make decisions. But the customer monitor data did not contain the values for those key variables.

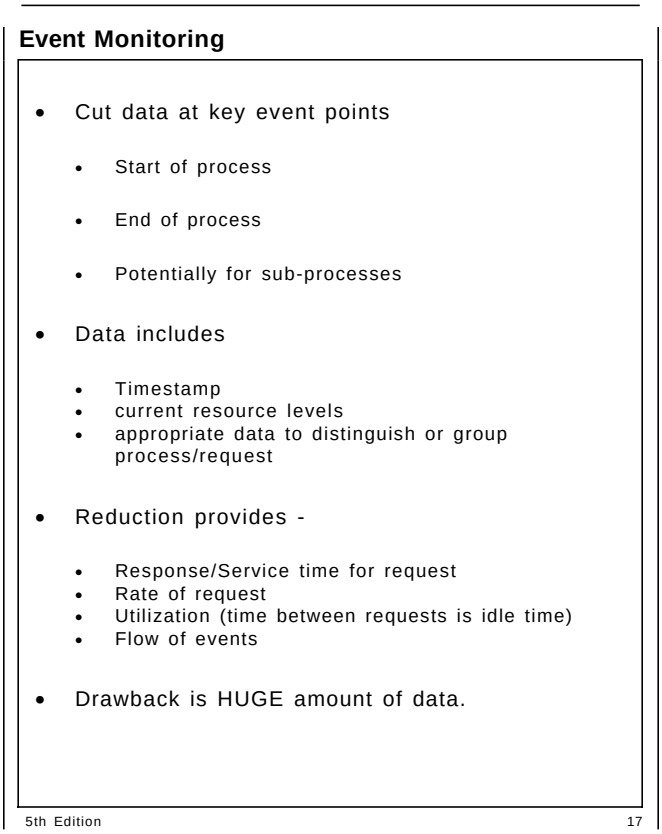
Values describing System Processing

- Most common type of data.
- Answers questions of -
  - What the System is doing?
  - Where is time being spent?
- Three approaches -
  - Event Tracing
  - Interval Monitoring
  - State Sampling

5th Edition 16

These are the types of things we most often think of when we hear ″monitor″ or ″performance data″. Those critters per second with a bunch of digits after the decimal point. They describe system utilization. This includes not only what resources are being used, but who/what is using them.

There are several approaches you can take. We′ll talk about three general ones. They all have strengths and weaknesses.

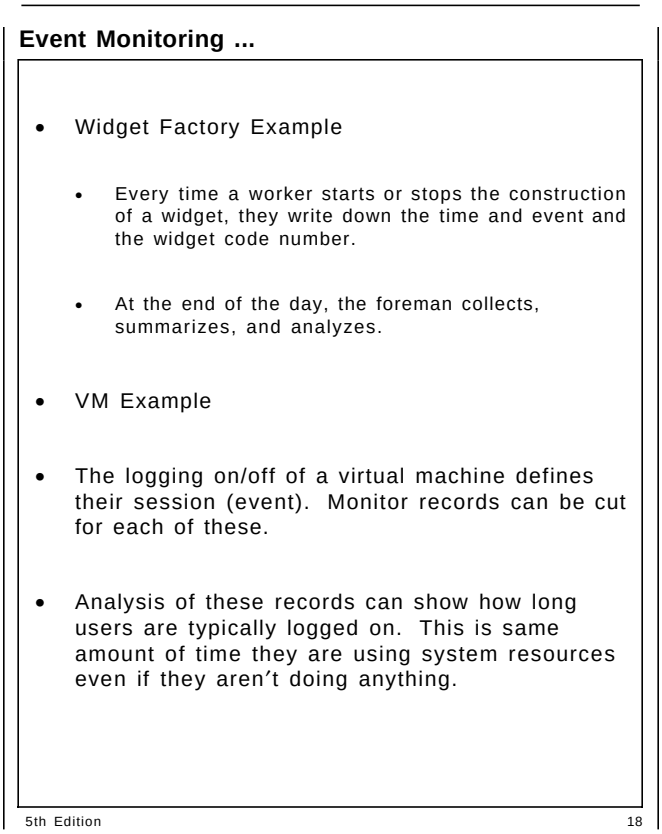


This often comes close to debug type activities. You are tracking exactly what happens and when it happens. When collecting this type of data, it is important to include information that can be used to sort or distinguish this event. This is in order to find out who/what is causing it. Data analysis might show the system is attaching and detaching expanded storage at a high rate, but to solve the problem you need to know who is doing this.

Reduction of data gives you good information (Service Time, Rates, Utilization). Two advantages this has over other approaches are:

1. Flow of Events
2. Ability to throw out mavericks from sample. Arithmetic ″mean″ can be misleading when numbers are skewed by inappropriately short or long datum.

The big drawback is the amount of data collected. It takes resources in moving it around and in holding it.



In both examples, there is info that only event monitoring could determine. Looking at averages could be misleading. In the widget example, one widget might take days to complete because the workstation broke down or a special part had to be ordered. This could really skew the average high. Statistics such as standard deviation and median could point out the situation. The same could be true in the VM case, where a single user could be logged on for days.

Both cases are also examples of important, but infrequent events. Therefore the data collected would not be overwhelming.

Interval Monitoring

- Three things to collect:
  - Running Count of requests
  - Timing of requests (stop watch)
  - Time stamp when Sample is taken which is included in Monitor data.
- Deltas of three items from sample to sample give:
  - Count - delta of count of requests
  - Usage - delta of timings
  - Interval - delta of time stamps
- From these we can calculate:
  - Rate = count / interval
  - Response Time = usage / count
  - Utilization = usage / interval
- Many times we only collect count and time stamp.

5th Edition 19

Another approach to data collection is interval monitoring. It is a powerful method for dealing with lots of data. From a few small pieces you can gain valuable information.

Start by collecting three key items at a periodic interval. In VM, this falls under the Monitor Sample data. After sampling these values for multiple intervals, we determine deltas for a period of time by subtracting the second from the first.

Simple arithmetic on the deltas provides the information that analysts crave for. Rate is the frequency at which the requests are completed. Response Time (or service time) is the average time it takes to complete the request. Utilization is a measure of how busy we were over the given interval.

Interval monitoring does not lose any data, but it does not provide the complete breakdown. Also, event flow is lost. It is very efficient with large amounts of data that would be prohibited for event monitoring.

In VM, we often only collect the count and time stamp provided by the Monitor. Many times this is acceptable. The timing of requests is sometimes difficult since multiple entry/exit points may exist or there is not a good stop watch available. Many times by an extra counter (Queue Length) can be added so that Little's Law can be applied to get Time in Queue. Little's law states -

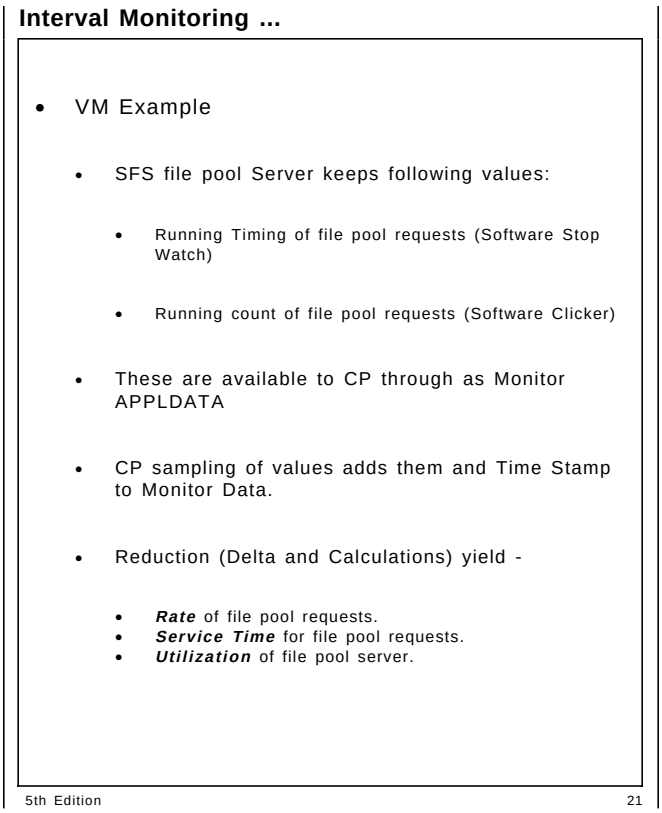
```
     Queue Length =  Arrival Rate * Time in Queue
or
     Time in Queue = Queue Length / Arrival Rate
```

Interval Monitoring ...

- Widget Factory Example
  - Data Collection
    - Hit clicker each time widget assembly starts.
    - Start stop watch when assembly starts.
    - Pause stop watch when assembly ends.
    - Collect clicker, stop watch, and wall clock values each hour.
  - Data Reduction - Delta of clicker, stop watch, and wall clock values.
    - Clicker - ***Count*** of widgets built that hour
    - Stop Watch - ***Usage*** time for widget workstation.
    - Wall Clock - Elapsed measurement time for ***Interval***.
  - Data Reduction - simple calculations.
    - Count / Interval = ***Rate*** of widgets being built.
    - Usage / Count = ***Service Time*** to construct a widget.
    - Usage / Interval = ***Utilization*** of widget workstation.

5th Edition 20

This would provide enough information and status to make any Factory manager happy.



This isn't a CP example, but it is VM. Shared File System did a great job of using this concept and exploiting APPLDATA. If you remember the example on System Management where the File Pool servers were paging extensively and causing poor performance, this data came into play early on. A breakdown of end user request time showed that a large percentage of the time was spent in the SFS server machine. That led to further investigation and seeing the paging problem. It would not have been of much value to fix RWSERV1 and RWSERV2 if it didn't help the majority of the end users.

State Sampling

- Useful when -
  - State Machine system (must be distinct states)
  - States change frequently
- Sample the state of the ″machine″ periodically
- Sample rate may need to be higher than normal sampling
- At sample time update counter corresponding to current state.
- Reduction provides - percentage of time in each state.

5th Edition 22

State sampling can be very useful for dealing with high frequency events (state changes), but has restrictions. The system must have finite number of distinct states. Another drawback is that with sampling you lose data. This is usually not a problem if you have sufficient samples to be statistically valid. State sampling can be used to answer the age old question of ″What are we waiting for?″.

State Sampling ...

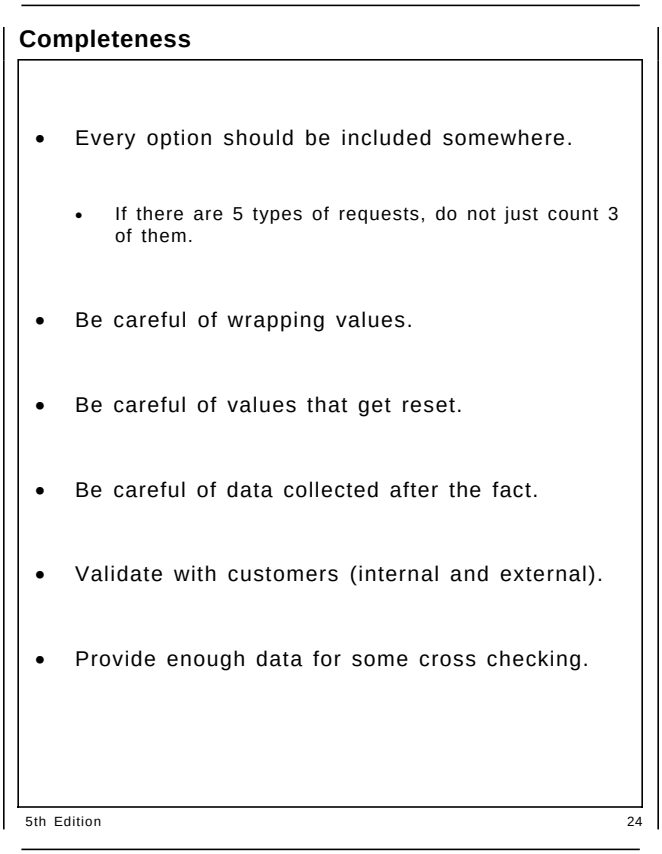
- Widget Factory Example
  - Every 15 minutes check to see if worker is idle, getting parts, or working.
  - From this data, conclude where time is spent.
- VM Example
  - Scheduler List Sampling. At high frequency sampling interval, check to see if virtual machine in Dispatch List (Q0 - Q3), Eligible List (E1 - E3), or Dormant List, or Running.
  - From this data, determine how much virtual machine is being delayed. (with a few quirks).

5th Edition 23

The Widget example is typical of many managers' style. But it is only effective if statistically valid.

VM uses state sampling in a number of places. The one shown here is only one example. The system management example VMPRF Report was based off of state sampling data. There are problems with the virtual machine state sampling. Some are due to not being a true state machine. In the system management example, the numbers are so strong that skewing from these quirks is not a problem.

That concludes a look at some methods of collection. There are a few other characteristics that need to be covered.



Seldom does a single number tell the whole story of how a VM system is performing (except maybe the number of end user complaints). The whole picture is needed. Here are several things to keep in mind to insure completeness.

Cover the different options. When the total number of widgets produced is compared to the sum of the widgets produced at the various workstations, the numbers should be equal. Unfortunately, there are times when we ignore Workstation Z. For a VM/ESA example, try figuring out what is causing SIE Breaks. You get some total counters, but the sum of the parts doesn't equal it.

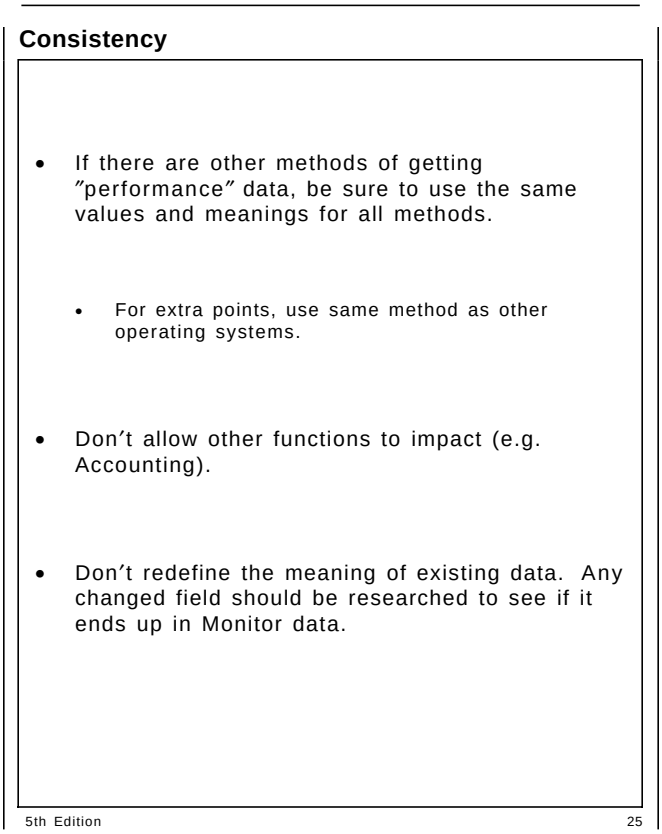
Some counters may only be sampled once every 15 minutes. The size of VM systems today leads to counters that can wrap in that time. Some of VM's problems here are related to hardware restrictions. It goes back to understanding how a fully loaded system is going to be using your code.

Occasionally, when plotting VM performance data you'll see blips in the curve that hit 0. They match timer pops or normal processing resetting counters they use. This can lead to a loss of data.

Sometimes we don't know a process is complete until we start the next one. When the next one completes, we can update the data. Problems arise when the wait between the two processes is long. This can lead to data associated with processing in one time interval being associated with a different time interval in the data.

It's okay to check with customers on what kind of data they need or if what will be provided is sufficient. This may include internal and external customers. Use of conference disks is encouraged.

Providing data for cross checks can be valuable. This may include keeping a total counter along with individuals. Over time things change, and we can sometimes get out of sync. If the total doesn't equal the sum, that's a warning.



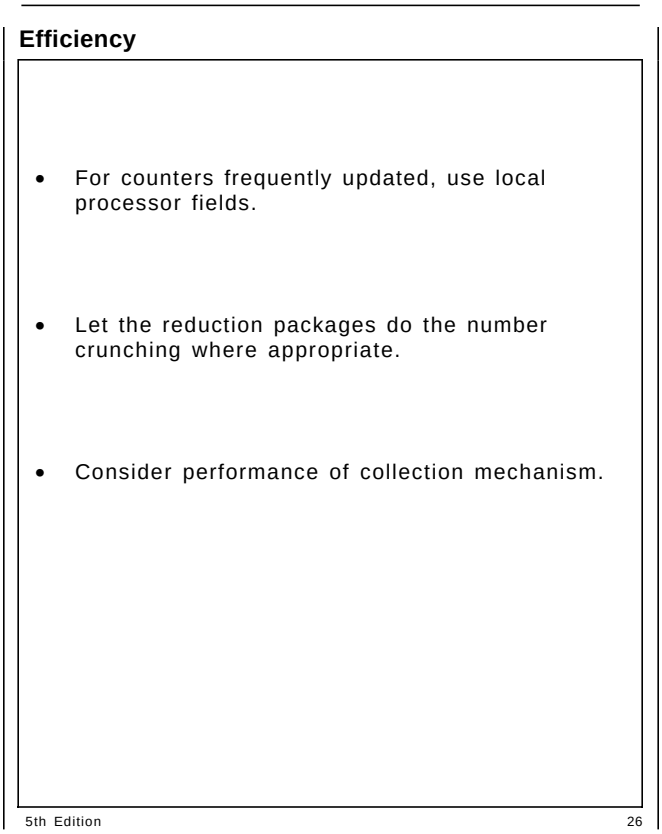
If you can′t rely on the meaning of data, it may as well not be there. The Widget Factory Director wants to rely on the production numbers being the same units each week, and the Widget Company Owner wants all the factories to report on those same units. In VM, make sure command output, monitor output, etc are related and meaningful.

It is difficult enough for customers to understand what all the different parts of VM mean, but we make things more confusing by redefining things. The fields from QUERY FRAMES do not match the fields in Monitor which do not match RTM screens.

Don′t let the non-Monitor processing impact data in Monitor. Accounting use to mess up monitor data in VM/SP. I haven′t seen any problems with Accounting and Monitor in VM/ESA (knock wood). (digression - is CP INDICATE an accounting or performance command?)

Changing the meaning of a field used in monitor records isn′t nice. If they changed the meaning of the box on pay stubs for vacation to be half days instead of days, it would cause problems (especially if no one knew about it). In VM/ESA 1.1, look at changes in block paging Domain 3 Record 8 (Block Paging Data). Not knowing that the meaning changed, you could think that your blocking factor was cut in half. Ouch!

Providing consistency protects the investment in existing monitor reduction programs and in knowledge of programmers/analysts. Most reduction programs can tolerate new records or longer records, but changing what they know is guaranteed to cause problems.

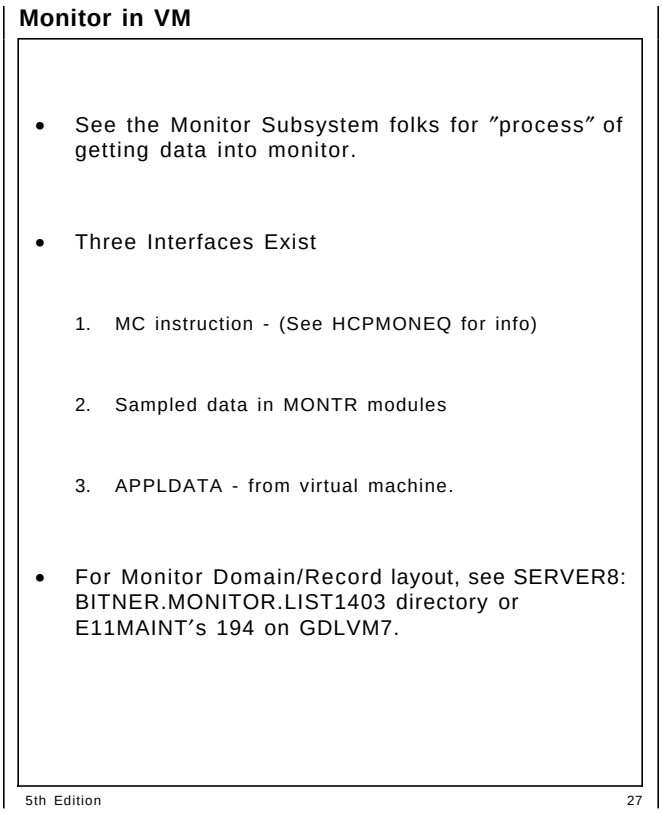


Monitor data is important, but not if it costs 10% of the system to collect. Customers are willing to give up a little performance for the added function and capability to manage their systems. There are a couple things that can be done to minimize overhead.

For the high frequency counters, use of local processor storage can minimize expensive cache misses.

Don't waste time (CPU and storage) to do calculations the reduction packages will do any way.

Three approaches to collection system processing data were discussed. They have different degrees of overhead associated with them. Knowing which collection mechanism to use depends on the frequency of the data changing. Here's where we need to combine knowledge of how the subsystem works and how a fully loaded system runs.



This presentation wasn't meant to detail the development process for adding monitor data. I don't have the experience or knowledge to do that justice. The Monitor Subsystem Developers have that piece of the pie. This foil is just meant to give a feel for what is involved. Your first step is to see someone in MONTR land.

For event type records, you do need to update your code. See HCPMONEQ comments for some background. You'll need to work with MONTR folks for the record layout and such.

For sample data, that can usually be added to a current record. You need only to work with the MONTR folks to determine which domain and record is the appropriate one.

APPLDATA is special and really not addressed in this pitch. I do have a separate pitch "Care and Feeding of CP Monitor APPLDATA" that goes into all sorts of detail. It is designed for server virtual machines.

If you want to get an idea of what is currently in the monitor, see one of the LIST1403 files for all the record layouts.

Summary

- We all have a responsibility to insure good data gets into the Monitor.
- Understand what ″good″ is.
- Use the appropriate collection mechanism.
- When in doubt, look to the Performance Evaluation area for help.

5th Edition 28

The Performance Area, the Monitor Subsystem Developers, and all the other Subsystem Developers share a responsibility to provide quality data in the monitor. It does lead to customer satisfaction and acceptance of VM. If you don′t believe me, look in the requirements database or at a number of APARs related to lack of good monitor data.

This pitch was meant to describe what good data is and how to deal with it. I know it′s not all inclusive, and may just lead to more questions. But thinking about Monitor and asking those questions is an improvement to the process.

If you have questions, go to the Performance person involved on the line item and ask. If there isn′t a performance person assigned or they don′t know, see me (Bill Bitner - BITNER at GDLVM7).