A Free AIX Performance Monitor

Joint G.U.I.D.E / SHARE Europe Conference
10 - 13 October 1994, Vienna, Austria
Session NN-0025

Jussi Mäki
Helsinki University of Technology
Computing Centre
Otakaari 1
FIN-02150 Espoo
FINLAND

tel. +358 0 451 4317
fax. +358 0 464 788
Email: **`jmaki@hut.fi`**
URL for author's homepage: **`http://www.hut.fi/~jmaki/`**
Anonymous ftp: **`ftp.funet.fi/pub/unix/AIX/RS6000/monitor-1.12.tar.Z`**

Abstract

A freely available performance monitoring tool is introduced for IBM AIX/6000 systems performance and utilization monitoring and bottleneck analysis. This text screen application can be used to show at a glance most of the available system status information. The development and some examples and future possibilities are also discussed.

1. Site and author introduction

Helsinki University of Technology (HUT) is the largest university of technology in Finland. HUT Computing Centre (HUTCC) provides various Information Technology services to other departments and students in the university. Computer resources at HUTCC consist of servers for scientific computing including IBM SP, RS/6000 servers and SGI ONYX. More services are provided by the Finnish National Supercomputer Center (CSC - Center for Scientific Computing) including Cray XMP, Convex C3840 and others. Other computer services are provided for general purpose usage. These systems consist of DEC Alpha and IBM RS/6000 servers and a cluster of HP workstations. About 11000 separate users have accounts in these systems.

The author of the **`monitor`** is working in the Computing Centre as a systems analyst in Systems Support Division. His main duties are UNIX and IBM SP-systems management, procurement projects, computer systems performance measurement and performance monitoring tools.

2. History

When the first RS/6000 machines were bought in 1990, no comprehensive tools existed for performance and utilization monitoring. Only tools available were the standard UNIX statistics tools like **`ps`**, **`vmstat`**, **`iostat`**, **`nfstat`**, **`netstat`** etc. These raw tools do not give one "at a glance" view which is needed when monitoring the whole systems utilization.

A monitoring tool was needed to find out how one of the RS/6000 servers with several thousand user accounts and over a hundred simultaneous logins was used. Similar applications have been available for other systems like the DEC VAX/VMS MONITOR. For other UNIX systems there also exists a tool called **`top`** which shows top CPU users processes, memory, and CPU usage at the same time.

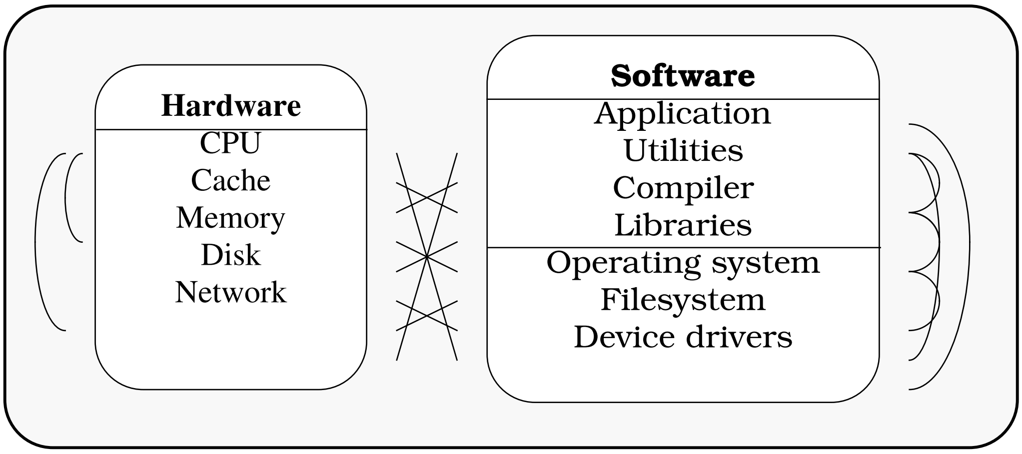
The first version of **`monitor`** was published in the beginning of 1991. Later new features like **`top`** displays were added to it. There were also ideas on making it X/Motif based but due to lack of time this has not been implemented. The current version of **`monitor`** program uses curses interface for ASCII terminals.

The implementation of **`monitor`** was partly done by searching monitorable variables and partly from stat-tools, include files and kernel variables. Some of the kernel variables were found by looking symbols from **`/unix`** using tools like **`nm`**.

IBM AIX/6000 version 3.2 also has few monitoring interfaces, however, these are undocumented (e.g. **`getproc()`** system call for getting the process list). In addition to these system calls a direct kernel interface via kernel memory (pseudo device **`/dev/kmem`**) was also used to access the kernel counters. The addresses of counter variables were found using **`nlist`** library call. Small utility was also made for displaying these kernel counters and showing difference of values between samples, so that the meaning of these counters could be found out while stressing the system and watching the output of other stat-tools.

3. Monitoring targets

Performance of a computer system can be described with various criteria. Different aspects which should be taken in concern are application, system software and hardware related issues. Naturally, all these aspects are further interdependent.



Picture 1, overview of different performance aspects.

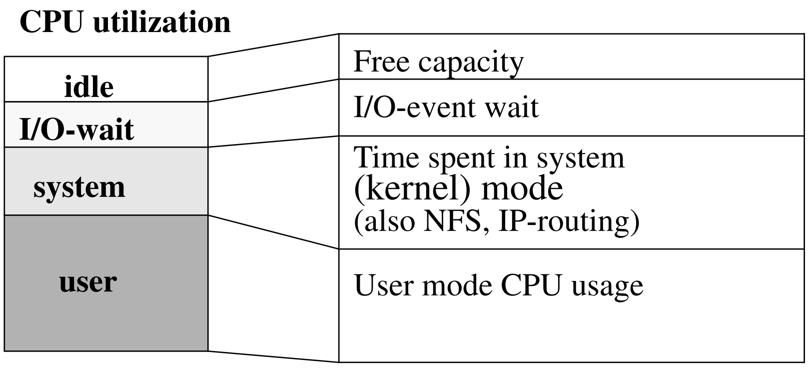
Thus for example the performance of the application is a combination of the performance of the hardware, operating system, compiler, and the application algorithm itself. The performance monitoring targets itself on most of these areas.

4. Monitoring areas

4.1 CPU utilization

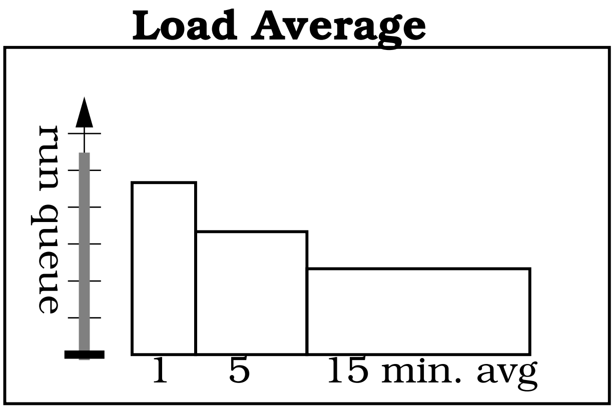
The CPU utilization is the most obvious utilization monitoring area. However, one has to take into consideration that CPU utilization does not give the whole view, there are also the I/O and memory related bottlenecks.

The CPU utilization distribution in AIX and typically all UNIXes can be shown in percentages for user, system, I/O-wait and idle processes. If system time is high it may mean that some process is running in spinloop. This may happen for example when an application is trying to read terminal input by calling **`read()`** system call without checking status value. There has been a lot of public domain applications which have done this error. This error appears especially when programs are run from the **`tcsh`** command interpreter.



Picture 2, CPU utilization parts

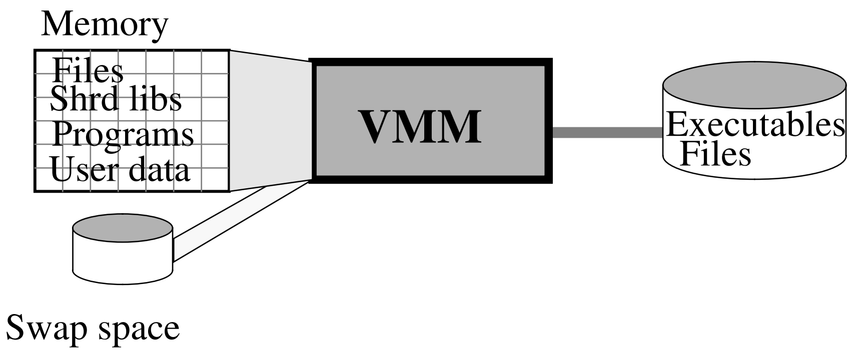
The load average describes the average CPU runqueue length in 1, 5 and 15 minute average. The load average number can be used to estimate the factor that is given to a individual user. For example if the load is 5.0 one consequent user will have 1/5.0 part of the CPU for users process.



Picture 3, Load Average

4.2 Virtual memory management

The virtual memory management forms the second most important performance area. AIX used to differ in this area compared with other UNIXes. For example all file I/O is done by mapping files to memory where actual operations are done. Currently there are also other vendors UNIXes which cache file access through whole memory.



Picture 4, VMM subsystem in AIX

The **monitor** utility (from version 1.12) can find out the real memory usage between ordinary processes and memory cached files. The amount of free and total size is also shown for both paging space and real memory. The paging rates between physical memory and paging space are also shown.

Many file I/O operations can lead to situation where very little memory is free. The lack of free memory is caused by that the AIX does not clear memory mapped file pages automatically but only on need. This situtation can be examined by the **monitor**.

```
Memory     Real      Virtual
free       0.5 MB    90.2 MB
procs     26.7 MB    53.8 MB
files    100.8 MB
total    128.0 MB  144.0 MB
```

Heavy file I/O operations can cause memory shortage for other applications. AIX swapper/pager will take pages back from I/O usage for conventional applications but the response time can get quite slow due to competition between I/O and process usage.

The AIX virtual memory management algorithms can be tuned with **vmtune** and **schedtune** tools (in directory **/usr/lpp/bos/samples**) and watched using **svmon** command (part of AIX 3.2 performance tools). More information is available in AIX performance tuning guide /2/.

4.3 Disk I/O

For database and general file servers the disk I/O is one of the critical performance issues. Since AIX disks are typically SCSI-disks the operations that are meaningful to measure are reads and writes. It would be interesting to count the seek operations but those are done internally in the SCSI disk and thus are not directly monitorable by the operating system.

The monitor displays an overview of the whole disk subsystems usage and a list of disks sorted by the amount of disk I/O operations. Also the busy utilization percentage is displayed, since it can be calculated from time the device is busy according to AIX kernel statistics.

```
DiskIO      Total  Summary
read          83.0 kByte/s
write         90.6 kByte/s
transfers     40.3 tps
```

```
active      13/14 disks

TOPdisk read write     busy
hdisk10   81   12 kB/s  14%
hdisk0     0   37 kB/s  17%
hdisk8     0   15 kB/s   5%
hdisk2     1   13 kB/s   4%
hdisk1     0    4 kB/s   1%
hdisk3     0    4 kB/s   1%
hdisk11    0    2 kB/s   0%
hdisk9     0    0 kB/s   0%
```

4.4 Process events and general I/O

Some more or less general I/O counters are shown in column **`File/TTY-IO`** and parts of system call information. These include the amount of **`read()`** and **`write()`** system calls and number of bytes transferred by these, **`readch`** and **`writech`**.

4.5 Network I/O

In typical client-server environments the network traffic is quite critical in performance problems. In AIX environments only the usage of the system itself can be measured by default. There are other methods for measuring the total network traffic (e.g. using network sniffers and other traffic measurement devices, for ethernet only systems utilities like **`tcpdump`** can be used, the author has done a port of this for AIX with a kernel extension to access the promiscuous interface).

The monitor shows a list of interfaces and their read and write rates in kilobytes. There is also a separate more detailed display (character command '**`n`**' or **`-net`** option) which shows also the counts of packets and average packet sizes.

```
cactus-fs1 load averages:  0.38,  0.31,  0.28    Thu Aug 11 16:39:09 1994
Cpu states:        0.3% user,   5.3% system,   5.7% wait,  88.7% idle
Real memory:       8.6M free  26.7M procs  92.7M files 128.0M total
Virtual memory:  90.2M free  53.8M used  144.0M total

Netw  read   write         rcount wcount rsize wsize
lo0     0.0    0.0 kbyte/s      0      0     0     0
en0     8.9   18.7 kbyte/s    425    464   214   412
en1     0.0    0.0 kbyte/s      0      0     0     0
fi0   120.2  120.1 kbyte/s   1147   1145  1073  1075
en2     0.0    0.0 kbyte/s      3      3    88   139
```

4.6 NFS I/O

Another network subsystem I/O event which is critical for typical disk server environments is the amount of NFS (Network File System) operations done. This information can be shown normally by `nfsstat` command, but monitor shows some of these statistics as a part of default screen.

```
Client Server NFS/s
   0.0   54.9 calls
   0.0    0.0 retry
```

```
0.0   19.6 getattr
0.0    8.4 lookup
0.0   19.4 read
0.0    2.0 write
0.0    5.6 other
```

4.7 Process CPU usage

In multiuser environments it is important to know how the CPU and other resource usage is distributed among different processes. Normally this could be done by **ps** command, but **monitor** shows an updated image of top cpu processes.

```
leka.hut.fi      load averages:  6.50,  6.65,  6.61    Thu Aug 11 16:44:38 1994
Cpu states:      86.0% user,  14.0% system,   0.0% wait,   0.0% idle
Real memory:     17.9M free 117.5M procs  56.6M files 192.0M total
Virtual memory: 463.5M free 136.5M used  600.0M total

   PID USER      PRI NICE   SIZE     RES PFLTS STAT USERTIME  SYSTIME CPU% COMMAND
 56281 arska      69   0    556K    664K  0.0 run       9:34     0:06  18.3% matlab
 27612 mtorma     69   0    304K    248K  0.0 run      69:24     0:05  18.3% simuclus
 51672 spalko     69   0   6668K   6764K  0.0 run       6:36     0:00  17.2% xcimtd
 80345 mheinila   69   0    308K    340K  0.0 run      34:18     0:00  16.0% silver
 35733 fbernard   69   0  17484K  17556K  0.0 run      12:17     0:00  16.0% cpv.x
     1 root       64   0    296K    220K  0.0 sleep     2:59     0:22   3.4% init
 78252 jmaki      61   0    364K    408K 25.4 run       0:00     0:00   3.4% monitor
 62968 rsalmio    76  15   3688K   1016K  0.0 run     162:32     0:00   2.2% grad
 91853 jmajan     61   0   3292K   3288K  0.0 sleep     0:20     0:10   2.2% Mosaic
 81033 ksippola   60   0    148K     56K  0.0 sleep     0:00     0:00   1.1% rlogin
 46855 root       60   0    228K    176K  0.0 sleep     0:00     0:07   1.1% telnetd
   514 root      127  21     28K      8K  0.0 run       0:00    87:27   0.0% Kernel (wait)
  9517 root       60   0    592K     80K  0.0 sleep     2:17    19:38   0.0% rpc.lockd
 66158 amiettin   60   0   7956K   8220K  0.0 sleep     6:18     0:01   0.0% mathexe
   771 root       39  21     32K     16K  0.0 sleep     0:00     6:05   0.0% Kernel (netw)
```

In the example following columns are shown:
- PID is the process identifier number
- USER is the user name
- PRI is the run time scheduling priority
- NICE is the UNIX scheduling parameter
- SIZE is the size of application (note this is not necessary correct value since the shared libraries are included in all processes' SIZE)
- RES is the size of resident memory usage of the process
- STAT is the status in which the process is (run, sleep or zombie)
- TIME is the amount of CPU time (USERTIME+SYSTIME) used by the process (is showed in narrow display)
- CPU% is the percentage usage of CPU in the display update interval
- COMMAND is the name of the executable

The following entries are displayed only when using wide terminals (>100 colums)
- PFLTS is the number of pagefaults (this is shown only with wide screen)
- USERTIME and SYSTIME shows the CPU time for user and system parts (with wide screen).

5. An example study

5.1 Lightly loaded NFS-server

Here follows an example what the monitor displays in one fileserver
with lightly loaded NFS traffic.

```
AIX monitor v1.12: cactus-fs1.csc.fi              Thu Aug 11 16:24:19 1994
Sys  5.0% Wait 19.3% User  0.7% Idle 75.0%        Refresh: 10.01 s
0%             25%              50%               75%              100%
===WWWWWWWWWWWWWWW.......................................................
Runnable processes  0.20 load average:  0.18,  0.24,  0.23

Memory    Real     Virtual    Paging (4kB)    Process events     File/TTY-IO
free       0.5 MB   90.2 MB      2.7 pgfaults     256 pswitch      89 iget
procs     26.7 MB   53.8 MB      2.9 pgin          43 syscall       3 namei
files    100.8 MB               16.9 pgout          4 read          3 dirblk
total    128.0 MB  144.0 MB      0.0 pgsin          3 write       793 readch
                                 0.0 pgsout         0 fork        192 writech
DiskIO     Total Summary                            0 exec          0 ttyrawch
read          11.6 kByte/s    Client Server NFS/s   0 rcvint        0 ttycanch
write         67.5 kByte/s       0.0   54.9 calls   0 xmtint       38 ttyoutch
transfers     17.8 tps           0.0    0.0 retry
active        9/14 disks         0.0   19.6 getattr   Netw  read   write
                                 0.0    8.4 lookup    lo0     0.0    0.0 kB/s
TOPdisk read write     busy      0.0   19.4 read      en0    15.5   77.8 kB/s
hdisk11    2   18 kB/s   8%      0.0    2.0 write     en1     0.0    0.0 kB/s
hdisk0     0   18 kB/s  11%      0.0    5.6 other     fi0     0.0    4.1 kB/s
hdisk8     0   17 kB/s   5%                           en2     0.0    0.0 kB/s
hdisk10   10    4 kB/s   2%
hdisk3     0    4 kB/s   1%
hdisk2     0    3 kB/s   1%
hdisk5     0    2 kB/s   0%
```

To get the picture of the system utilization, following observations can be done:

A. CPU utilization, I/O wait period is quite high, so it implicates that processes are waiting for disk I/O to complete
B. Memory usage, the files part of memory usage is quite high, this implicates that lots of memory is used as buffer files
C. Some amount of NFS calls, read and getattr mainly
D. Some amount of Disk I/O, multiple disks are read and written in the same time, since the machine is fileserver, these operations are either minor local operations or major remote operations
E. Network traffic, data is transferred to network, to found further which are the hosts using network services from this machine one could use AIX commands: (part of AIX performance tools /5/)

```
netpmon > netpmon.report;sleep 30;trcstop; more netpmon.report
```

6. Future

The future of **`monitor`** development is not necessarily very promising, since the development of a full fledged monitor requires quite an effort. Some goals for a better monitor would be the support of agent/manager typed environments and the possibility to record and play back performance monitoring data. Actually a prototype of this kind of application was done in the spring of 1994 but the further development has not been continued yet.

6.1 Improvement hopes

The current UNIX flavours including AIX do not give comprehensive view of one user system utilization in multiuser environments. This area would require further investigation. Of course in single user systems or workstations the global system statistics can be sufficient but this isn't true in multiuser environments.

Some information that could be useful are system and library call level statistics (for example I/O read,write call counters with number of bytes transferred) per process, disk and network usage per process. Also more detailed memory usage information and network statistics per protocol could be useful.

The AIX trace facility could also be enhanced to support more information on system call level tracing. New information that should be supported are arguments of all system calls like SUN-OS **`trace`** tool does. This kind of information could be useful when making performance modelling and optimization of applications.

The AIX should also provide an interface to the filesystem level disk statistics instead of plain physical volume e.g. disk level statistics.

There has also been a standardization process going on together with X/OPEN and OSF under a Performance Management Working Group (PWMG). The result at the moment is a proposal on Universal Measurement Architecture (UMA). The proposal specifications are available via anonymous ftp /4/. The proposal gives quite a substantial goal on implementing large monitoring system, which could monitor almost all events in a distributed computer environment. This possibly also makes the full implementation quite heavy.

6.2 IBM SP systems

For the IBM SP and other multiple processor clusters a single system image tool is needed. At the moment IBM does not provide single system image tools for performance monitoring. The PTX/6000 product can be used to get a system level statistics from a node at a time.

The author has done some small tools to show overview of system utilization in a IBM SP environment. These tools include **`sptop`** which shows top CPU users in the whole SP, **`spsystat`** which shows both the total usage of SP and separated node utilization. These tools are formed in agent and manager part, where the agent part provides the information in a node and the manager (user application) collects the information from several nodes. For larger SP environments(eg. 64 nodes) a hierarcical data collection method should be used. With hierarcical data collection some nodes could collect system utilization information from other nodes and forward this information for monitoring application. Thus the network and CPU load caused by monitoring applications itself could be reduced.

6.3 Other applications

At the moment IBM provides an application called PTX/6000 which is made in agent/manager recorder/player style. One disadvantage in this application is that it does not display CPU utilization per process level like **`monitor -top`** does.

IBM also has a collection of AIX subsystem specific tracing and statistical analysis tools for virtual memory usage, network usage, filesystem usage /5/.

In addition to IBM products there are also other commercial performance monitoring applications available from various vendors like the BGS Systems, CIS Inc and Landmark Systems Corporation.

7. POWER2 Hardware Performance Monitoring

When IBM published the POWER2 architecture it contained one very interesting aspect on performance monitoring area. The POWER2 supports CPU hardware performance monitoring. However the public information that has been available does not describe detailed implementation information /1/.

The user community requirements of hardware performance monitor similar to CRAY **hpm** would be probably needed in order to enforce IBM to implement similar one. There are also some concerns if this kind of applications are long living since hardware performance monitors are very CPU dependent and the current evolution time of CPU implementations is very short. An alternative to tools provided by IBM are user community provided tools if only there would exist a public interface to the POWER2 hardware monitor registers.

Some of these tools could provide information on the whole systems CPU usage, periodic line by line information (like **vmstat**) and at least per application execution statistics like the Cray **hpm**.

A sample of data that would be useful could look like this:

```
Simulated IBM Rios 2 Performance Statistics for: dmm-400
This program shows all processes CPU usage during the execution
CPU seconds            :       0.5500   CP executing    :     36600000
Elapsed seconds        :       0.6300
FPU0 results/sec       :      58.00M    F.P. in Math0   :     32000000
FPU1 results/sec       :      58.00M    F.P. in Math1   :     32000000
F.P. add ops/sec       :     116.00M    F.P. add        :     64000000
F.P. mul ops/sec       :       0.00M    F.P. mul        :            0
F.P. div ops/sec       :       0.00M    F.P. div        :            0
F.P. ma  ops/sec       :     116.00M    F.P. ma         :     64000000
MFLOPS ratio           :     233.00M    F.P. math ops   :    128000000
```

References

/1/ POWER2 Performance Monitor; E.H. Welbon, C.C. Chan-Nui, D.J. Shippy, and D.A. Hics, PowerPC and POWER2, Technical Aspects of the New IBM RISC System/6000, SA23-2737-00, 1994 International Business Machines.

/2/ AIX 3.2 Performance Monitoring and Tuning Guide, 1994 International Business Machines.

/3/ Vmstat, iostat, nfsstat, crash; Manual pages of various performance statistics related AIX/6000 commands, 1994 International Business Machines.

/4/ UMA specification draft; Anonymous ftp to tarpon.instrumental.com, directory **pub/pmwg**, 1994 PMWG, XOPEN, OSF.

/5/ svmon, rmss, netpmon; Manual pages of AIX 3.2 Extended commands **bosext1.extcmds.obj** performance tools, 1994 IBM.