

## Re: Wiseman and McGhie are Ranting Again

---

*Source:*

<http://www.tech-archive.net/Archive/Mac/microsoft.public.mac.office.word/2005-09/msg00487.html>

---

- *From:* Jeff Wiseman <[throwawayacct223@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:throwawayacct223@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Sun, 18 Sep 2005 13:08:55 -0500
- 

John McGhie [MVP - Word and Word Macintosh] wrote:

Hi Jeff:

Is it Sunday already? Must be time to lose the rest of the weekend in a discussion with Jeff :-)

I appear to be getting a "reputation" in the group. I hate it when that happens (thanks for changing the subject line).

Enjoy it while you can though, it appears that I have finally gotten myself my first real full time job in a while so I may not have as much opportunity to be online as much.

As long as I am entertaining and/or informative though, I guess I should hang around...

:-)

Actually, I don't think it is so much the Mac OS X GUI as it is the true multiuser, multitasking, full virtual memory unix like underpinnings of OS X that give it that sluggishness.

OK, there's two issues there. I was talking about the Windows XP GUI. I find it quicker to get stuff done: it may be ugly, but it's fast. Even if you know it well, OS X seems to involve more keystrokes and contortions to do things.

## Re: Wiseman and McGhie are Ranting Again

That would be at least in part because I come from a Windows background, so my computer and folder structures and such are all set up that way. If I were to take advantage of all the power tools buried in OS X and change my working style, I suspect I would find less disadvantages in the UI.

I suspect that what you say is very true. The desktop paradigm as implemented over Darwin is going to require some years to "evolve" to become as effective for many applications (i.e. how people do their work) as the traditional Windows UI was. A lot of this may also be due to the Windows UI having so many different ways to do the same thing (unfortunately each with its own caveats), everyone can eventually develop their own desktop processes.

I've used X windows, Solaris, and Openwindows on sun workstations and they all seem to have that same type of sluggishness. In fact OS X seems to have the edge in terms of "feel" but that may simply be because I've only used it on workstations that are faster than the ones using the other interfaces.

Yes, they do. But I am not sure that's the fault of Unix. I suspect that part of that is due to application software vendors being unwilling to re-architect their applications for a multi-threading, multi-tasking environment.

This is certainly part of it. E.g., \*NO\* application running on any Unix type subsystem should \*EVER\* sit in a polling loop (i.e., the very way that Word functions). It should be blocked on input in an event driven style. Otherwise it is chewing up and wasting CPU cycles. It also forces the computer to leave RAM for the virtual memory assigned to that process to also be tied up. This has always been extremely bad programming style IMHO.

I guess one of the most significant issues that I was mainly alluding to is the time to load an application (or some of its processes if it is a multi-process type application) and start it/them running. The necessary

## Re: Wiseman and McGhie are Ranting Again

evil of a full-blown virtual memory, multi-tasking system is that the loading and initiating of processes tends to be more complex.

Note that the multi-tasking and virtual memory mechanisms of both the old Mac OS and Windows are really very simplistic compared to the way it is done in UNIX. More on this below.

Certainly it wouldn't be virtual memory that's doing it (unless you happen to be "using" the virtual memory...) :-) We've had virtual memory since the days of DOS and Mac OS 7 or whatever.

Again, this is true but their implementations are fairly simplistic. Loading an application into memory now requires a view of segmentation and such maintained by the processors and the OS. From what I understand, this involves more overhead in a full blown fully virtual memory system like UNIX.

This is why in OS X (or other Unix based systems) an application that is loaded but not being used (i.e., hasn't required any CPU cycles for a while) can eventually swap totally out to disk and basically will not use any of the computer resources except disk space but can be recalled almost instantly when called upon to run again. This is good in that if you have the disk space, you can have literally thousands of tasks in the running but blocked mode (i.e., waiting for input events) and the system can be just as responsive as if only a couple of active tasks were running.

However, when a task is blocked waiting on input and has been swapped out, there can appear to be delays as it swaps back in to process its events. This can appear as a sluggishness to respond to UI input. In the original Mac OS (and I believe Windows as well), part of the application that handles input is left in RAM space in order to avoid the delays. However, that means you can't use that RAM space for anything else until you quit the application. This is why leaving unused applications running on the old Mac OS or in Windows is a bad idea whereas leaving unused applications running on a Unix system doesn't really hurt anything and in fact can be good since the next time it is required to run, it is available nearly instantly.

My point was basically that simplistic VM typically was built in a way that full swapouts didn't normally occur because it was so difficult to reload them fast for certain types of input (usually because the pieces being swapped in and out were monolithic--huge chunks decided upon by the developer). The Unix type VM swaps in many tiny pieces and is transparent to the programmer. This has the advantage that the entire application can be swapped out if necessary, or only enough pieces to allow what ever

## Re: Wiseman and McGhie are Ranting Again

other processes need to be running. The down side is that there are:

- 1) potential delays swapping an app back in to handle its UI input and:
- 2) Extra complexity (i.e., delays) loading an application the first time.

Virtual memory simply assigns a portion of the disk to impersonate "real" memory. If the user runs more applications than he has memory for, the system "pages" the content of one or more chunks of an existing application out to virtual memory to make room. If that happens much, the system gets seriously slow. Anyone with a gig of real memory in OS X (or Windows) will never hit virtual memory, PROVIDED they quit each application when they finish with it.

All true. However my point was that with a system that has defined very small "chunks" as swappable (e.g., UNIX), pieces that are rarely used get swapped out first so swapping activity slowdowns are not as severe in general. The RAM use becomes far more effective. A skilled programmer can design an application to function effectively with this in both the UNIX or the older Windows, Mac OS environments. However, a very bad programmer can not cause as much trouble through his application design on a UNIX system as he can on the older personl OS type systems

If they adopt the old Mac user's operating method of leaving everything running between uses, then they WILL be hitting virtual memory and their system will be slow :-). I can't seem to get this through to Windows XP users either: if you don't quit stuff you are not using, eventually your system starts flogging the disk looking for memory and everything gets veerrry s l o o o o w w w w . . . .

Exactly. But on something like UNIX, although slowdown will occur, you won't get the extremes unless many processes are written poorly (e.g., polled inputs like Word uses) and try to all lock themselves into memory.

"Multiuser" doesn't really enter into the performance discussion. Really, it's an accounting mechanism. It determines who is allowed to do what, and where to send the bill for the services consumed. But it's the same

To support a very effective multiuser system, there will be other

## Re: Wiseman and McGhie are Ranting Again

overheads added. If not anything else there are more file trees to navigate and added over head and security processes running on the system (e.g. group, world, owner type ownership checks, etc.). I have no idea how significant the extra overhead this adds--it may be trivial--but I think it is finite so that is why I mentioned it.

BTW, a good multiUSER system (as opposed to mutli tasking) is going to quarantee that a single user can't take over the system. If a user starts a CPU intensive task (such as compiling aprogram, etc.), the OS is not going to allow that to go very far before yanking the cpu away from that task in some fashion. UNIX as a very effective way of dynamically assigning task prioritys to do these that has been more or less in places for many decades now.

The analogy is that Word is like a bacteria in a healthy body when it is run over Darwin. The prioritizing algorithm of Darwin sees this process that wants to just keep running because of its silly "idle loop" design, and so it will attempt to protect the rest of the system by dropping Word's run time priority in stages.

Now: Multitasking is a different issue. To begin with, no affordable computer can do it, and never has been able to :-). Multitasking literally means that the computer can process work from more than one task

Of course, I was referring to virtual tasks and the time sharing/time slicing mechanisms of the OS. So within the context of my original comments all PCs/Macs are multi-tasking. Some are orders of magnitudes above the others in their capabilities.

simultaneously. How? It only HAS one CPU!! It can only ever do one thing at a time :-). What the computer industry laughingly calls "multitasking" in fact describes a computer's ability to task-switch very rapidly. Computers

I will forego a sudden impulse to discuss the incremental theory of time and some of Dr. Who's exploits, regardless of how relevent they may be here :-)

<<great discussion on time slicing, multithreading, and prioritizing deleted>>

## Re: Wiseman and McGhie are Ranting Again

But splitting an application up is horrendously complex and difficult to test, because all of the pieces may depend on work being done by another piece. So application designers won't attempt it unless they can begin their design that way.

Exactly. As you pointed out, with the original PC and Mac systems it was, in fact, up to the application designers to cooperate with each other to "tune" all these things. These is NOT a cohesive way of controlling system level operations. A system's performance should not depend on an application's design. System tuning and control must be available at the system level. I.e., nearly all system tuning must be possible via the OS itself and not depend on how smart a particular application's developer was. UNIX (i.e., Darwin) does this. PCs and the original MAC OS really don't and require the cooperation and skills of the developers following a certain set of standards and philosophies in order to achieve a responsive system. In Unix, if a process tries to hog the system, the OS will take away it's ability to eat CPU cycles at a high rate.

Most "modern" applications have never gotten beyond "idle loop processing".

This is a sort of "cheat" multi-tasking. You take all of the bits that interact with the user and put them in the Main thread. You take everything else (all the bits that actually perform work) and place them into the Idle Loop. The main Thread runs very frequently to check if the user wants to do something, then falls quiet and calls its Idle Loop. Word uses this technique.

The issue is that there should always only be only one "idle loop" in the entire system. Everything else should be event driven. In Unix, that idle loop (i.e., the scheduler) is in the OS kernel. Word wants to be the entire OS which is the wrong approach. Unfortunately, because of it's massive legacy and the fact that this is an infrastructure issue, it is very unlikely that this will ever be corrected. However, I believe that there may be hope in OS X since the OS provides enough tools to make such a change easy. The problem is that if MS took advantage of this, their OS X product(s) would then have infrastructures significantly different than their PC products so again, this is unlikley to happen.

## Re: Wiseman and McGhie are Ranting Again

For that reason, I suspect that it's going to be hard to again match the great responsiveness and feel of a single user pseudo multitasking OS such as the original Mac where you could sizzle through operation after operation and the computer could actually keep up with you.

No. But it is a LOT of laborious detailed work, splitting applications up into multiple threads, adjusting the priorities of each, tweaking their priorities, coding, and testing. It doesn't add ANY "new features" that you could give to Marketing to SELL.

Again, I think that the development issue that you present, although entirely valid on a PC or even a classic Mac OS, doesn't apply as well on a UNIX based system such as the Mac OS X IMHO. I suspect it would still be a massive restructuring effort to convert it to an event driven interface but it would result in a significantly smaller application since many of the tests and gotchas currently in the product would not be necessary

This is not something software vendors are actually "enthusiastic" about.  
The move to MacIntel may assist them to learn to love it :-)

Unfortunately, I believe the issue is the OS and not the hardware it runs on. Although the exact way the VM works is dependant on the processor's capabilities, with few exceptions multitasking is a software bound function. Darwin has the capabilities but the infrastructure of Word assumes a totally different environment where it is so mated to that environment that environment specific issues saturate the innards of Word.

In summary, I do think that the characteristics that make the Unix subsystem of OS X as great as it is will also contribute a bit to a sluggishness that earlier systems didn't have. A lot of that can be reduced by system tuning and applications eveoling to use better design techniques. Applications that doen't follow good programming rules will impact the system some, but not like they used to in the old style OSs.

Re: Wiseman and McGhie are Ranting Again

--

Jeff Wiseman

to reply, just remove ALLTHESPAM

.