DosGetNews()

March, 1989
Vol. 2, No. 3

SYSTEMS PERFORMANCE GROUP

by Russ Blake
Manager, Systems Performance

What makes a group? One person to do all the work and a manager to watch! In our case it's Wayne Johnson churning out the results and Russ Blake making the excuses. So if you haven't heard about us before, it may be because we are pretty small. And whoever said small was beautiful didn't have to deal with all the performance issues in OS/2 and PM!

But don't get us wrong...we're not complaining. Our charter is "to make OS/2 & PM the world's fastest system for personal/workgroup productivity." We hope to do this by focusing on the user's perception of performance, pinpointing performance problems, designing/prototyping high performance solutions, and coordinating with big blue on these efforts. We really like trying to make this happen, and have had a few successes in the last few months which have made it all seem worthwhile.

Naturally we have a plan for achieving our goals. Unfortunately the plan never lasts intact for more than 24 hours. The plan incorporates some fairly routine, ongoing efforts along with some ambitious tool smithing.

One of the ways a small effort like ours can really help is by putting useful, easy to use tools into your hands. One tool we envision is a cpu profiler. Today we have a kernel profiler that tells us where the processor is spending its time. This has been incredibly useful, but requires a special kernel, and takes quite a bit of cpu and memory resource to run. The tool we envision would install more easily, consume fewer resources, take data on a thread basis, and display data in a more meaningful way.

A grandfather of such a tool is CPUMETER, which you can steal from \\danhi\public. Try it, you'll like it!

Another tool we need is a way to measure and improve working sets. In an ideal world, such a tool would reorganize your .obj files to minimize

your working set, based on some set of measurements you had taken while running your program or library. How far we can get towards achieving this ideal remains to be seen.

At its current state of development, OS/2 is still largely a black box. When something is slow, often the only tool we have at hand is the light on the disc drive! We all need to be able to tell how much time our threads block on semaphores, wait for disc or hit the cache, seek, wait for the cpu behind someone else, how much memory we are using, and a whole slew of similar performance-related values. Good general instrumentation of the system is high on our list of goals, and I think we will break some new ground here as we design these tools for the multiprogramming workstation developer.

Our group also spends quite a bit of time generating disciplined benchmarks which tell us where the system is slow. Some of these we run on every build of an emerging release, and others we cobble together just to track down a particular problem which we have noticed or which one of you has brought to our attention. We make a serious effort to automate the execution of benchmarks and the analysis of the results, so we don't have to spend too much time running the tests on a routine basis.

We presently have an effort underway to convert some of these bechmarks to the Macintosh so we can get a reading on how PM compares to the MAC.

We have spent some time looking at some of the designs coming down the OS/2 pike, and trying to see if there might be some performance liability lurking in the wings. This has been reasonably successful, but we need eternal vigilance if we are going to avert some of the nasty surprises which have recently overtaken the system.

There are a lot of problem areas that we would like to study, and quite a few parametric studies we would like to do which would add to our understanding of the system's behavior and which might help to direct future designs. One example: config.sys parameter interaction. How does cache size interact with the buffers parameter? How does the timeslice really affect performance? Over time we hope to get time to look into these kinds of issues, although problem areas tend to get our attention today.

For example, recently we have spent alot of time analysing the memory consumption of OS/2 and helping a small team of developers to reduce the size of the system. Our role in this has been to dust off the arena tool, which lists the system segments, and to write a few analysis programs to help to compare 1.1 to 1.2. We have tried to assess system growth in this fashion, and provide experiments and tools to aid in reducing the size of OS/2.

There are a couple of ways that you can help us to achieve our goals. One is to let us know anywhere you think the system is unduly slow. Our experience has been that you have not been shy about this! Another is to assist us when we need your help to understand some performance problem. Because of the size of our group, frequently the best we can do at present is to point you at a benchmark and a tool to measure it with, and set you free. Sometimes we don't even have a decent benchmark, and we'll ask you to help us to generate one. We are working to set up a performance lab where the tools and benchmarks reside, and investigations can be carried out in a disciplined fashion on retail (non-debugging) versions of the code. We have 4 machines for this purpose right now, with 2 more on order.

We can't achieve our goals without the help and cooperation of you all, but with your support, we will produce the meanest, leanest screaming demon of a system ever to hit the street!

Program Managers and Project Managers - So What Exactly Is the Difference?

by Nancy Lanning
OS/2 v1.2 Project Manager

I've often wondered about this myself, especially since the OS/2 JDA group seems to be the only group in Microsoft with Project Managers while all groups have Program Managers. What are each groups' responsibilities? How do they interact with each other and with other groups in Microsoft and/or IBM? Who should you go to with various questions, issues, problems, etc?

It was these questions and more that we tried to answer at an off-site meeting a couple of weeks ago. In this article I'll try to summarize the results and will use the Design Change Review (DCR) process as an illustration.

Missions

The first thing we did was to define "missions" for each group as a way of defining their responsibilities and interactions. Here's what we came up with.

The mission of Project Management (PjM) is to deliver a high-quality operating system, as defined by Program Management (PgM), to meet MS/IBM goals by acting as the decision point for trade-offs involving resources, content, and time-to-market. Also, it is PjM's responsibility to synchronize the product schedules of other MS groups which have dependencies on OS/2, including OS/2 marketing, international, network, DOS, and languages.

This mission is achieved through internal resource planning, management of relationships, and product management activities. PjM examines the goals for the product, develops priorities with input from PgM and Marketing, and commits and schedules work for the project. This involves making trade-off decisions based on strategic implications, resource constraints, and size/speed/functional considerations. In addition, PjM serves as the primary interface to other MS groups and to IBM on project status, coordination, and issues management.

The PgM mission is to formulate, specify, and negotiate the product content for a complete, competitive, and high-quality operating system.

This mission is achieved through developing release goals and objectives for each technology area, acting as the focal point for design resolution within MS and IBM, and analyzing, comparing, and communicating OS/2 functionality relative to the competition. PgM acts as a catalyst to ensure that issues are addressed, problems are resolved, and design decisions are made in such a way as to maximize OS/2's success. This is accomplished through data gathering, joint discussions between all affected groups, and prioritization/decision based on the overall goals for the product.

Our next step in refining these missions is to define the relationship and interactions with the development and test organizations. We hope to be able to do this over the next couple of months.

Another thing we're trying to do is map our organization to the IBM organization to determine where the main contact points are. While this is not as easy as you might think, it's roughly true that our PgM group maps to their Planning and Design

groups and our PjM group maps to their Development and Test groups (of course our CT group also interacts directly with their CT group). Recenting IBM reorganized it's OS/2 development group to form a Project Management organization primarily responsible for inter-and intra-site coordination, status maintenance, and issues management for a given release. While they haven't said anything directly, we like to think this was influenced by their recognition of the effectiveness of the MS organization.

DCR Process - An Illustration

The DCR process is sometimes a mystery to the innocent bystander. Exactly how do decisions get made? What are all of the reviews boards that a DCR must go through? Who writes the DCR package? etc. It's actually fairly straight-forward in a convoluted kind of way. The diagram at the end of this article is an attempt to show the DCR activities and primary parties responsible at each phase of the DCR's life. But first, a little terminology. (Note: This illustration uses the current Sloop (1.2) DCR process as its model. Things may change for Cruiser (2.0).)

ART =
Architectural Review Team. Responsible for design/technical review of a DCR to decide if it's the best way to accomplish the goal of the DCR. Theoretically this group does not make decisions about whether the goal is desirable or not, but realistically they often make this assessment as well as reviewing the design. The primary team members are IBM and MS PjMs, PgMs, and release architects. Other people are consulted as appropriate for specific DCRs.

CCB =
Change Control Board. Responsible for assessing impacts of a DCR on other groups responsible for product activities and for providing management commitment for a DCR. Board members include MS and IBM representatives from PjM, component test, system test, publications, DOS, compatibility, etc.

Results from either the ART or CCB include approval for the current release, rejection for all releases, or rejection for the current release but should stll be considered for a future release. A DCR is not committed for a specific release until the CCB has approved it.

ESCALATION =
When someone doesn't agree with a decision of the ART or CCB s/he has the right to escalate to the next level of management. While anyone can escalate any issue, the escalator must be prepared to defend his/her position against fairly tough opposition (the ART and CCB don't often reject things without having some pretty good reasons). The PjM group is the primary escalation point for MS adn we will get others involved as necessary. If we can't get it resolved then it goes to Petern. On the IBM side there are many more levels and branches of management that an escalation must go through, so this can often be a time-consuming process.
To prepare for an escalation the escalator should be able to defend the issue technically, justify why it is necessary for a specific release, give development and test work estimates, and identify what the problems will be and who will be impacted if the work is not done. A good way to win an escalation is if you can identify a dependency on the work by some other Plan of Record work (e.g., we need to do the resource compiler changes for writing icons in EAs to support the Shell).

I hope this clears up at least some of the confusion/uncertainty without creating even more. Our goal is to help make our OS/2 product development efforts more effective and productive, and most of the feedback we've received so far indicates that we're moving in the right direction. Please let me know if you have any questions, suggestions, concerns, etc. Since all of this is still evolving, your input may help shape the future! still evolving, your input may help shape the future!

DCR PROCESS CHART

	Problem Identification	ART Review	CCB Review	Development and Test
Inputs:	Problem statement from outside group DCR draft from IBM Problems identified internally	Draft DCRs with problem statement and implementation details	DCR from ART	DCR with spec pages as design document
PgM:	Create/review design solution Prioritize work given other product content commitments Coordinate preliminary design and priority resolution with MS and IBM groups Draft DCR with problem statement and implementation details (may be done by IBM or technical leads)	Coordinate final design and prioritization with MS and IBM Escalate as necessary	Monitor status of DCR resolution Escalate as necessary	Consult on implementation details as necessary
PjM:	Monitor status of problem resolution	Work with development and test to create work estimates and schedule work Work with development to get spec pages added to DCR Drive process (ART meetings, action items, escalations)	Monitor status Drive process (set priorities, escalations, action items, etc.) Coordinate impact identification and scheduling	Work with development and test to schedule work Monitor progress Coordinate dependencies
Outputs:	Draft dcrs Work prioritization	DCR with final design, work estimates, spec pages, and schedule	DCR from ART with impacts to other groups	Code!

THANKS TO ALL RECENT

CONTRIBUTORS TO THE

DosGetNews() !!