The performance PM: Driving for a faster product

BUSINESS SKILLS

May 10, 2002 — By John Spilker, *Interface* staff writer, and business group **contributors**

The road to building a faster product is anything but fast. Performance problems are seldom simple, and solutions often require a team of focused developers guided by a steady navigator.

That's where the **performance program manager** fits into the picture. This team member selects the destination, maps the itinerary, and makes sure all the key players are on board and traveling in the same direction.

You won't find anyone with the title of "performance PM" in the Address Book. Instead, these are PMs who specialize in performance issues. In some product groups, including NT, *all* PMs work on performance issues.

"Performance is one of the hardest areas of the product to work on," says Craig Unger, product team manager for Access. "It takes a lot of creativity to look at the various scenarios in interesting ways, and then pull out what the problems are." In Craig's view, a good performance PM knows what the customer wants; keeps the developers focused on the key issues; and maintains a balance between competing forces such as new features, limited developer resources, and tight deadlines.

A balancing act of trade-offs and challenges

Just about every product under development has the stated goal of being faster than the previous version. But it doesn't always turn out that way, and every performance PM must balance performance as a priority against many trade-offs and competing issues. It's their job to manage these trade-off decisions and to see that they are made intelligently. If they aren't, the product will get not only slower, but **painfully slower**.

Typical trade-offs

- New features and code changes usually cost performance.
- Team resources are limited.
- There may be little or no control over problems created by outside components.
- Performance work must conform to ship date schedule.

New features and code changes usually cost performance. While customers and marketing love new features, any major additions or changes to the code invariably lead to a performance hit, says James Sturms, program manager for Access. Improved hardware performance can offset these penalties, "but you don't want to rely on [user hardware upgrades] too much." The performance PM must also make sure that any degradation is offset by the benefits of the new features, adds Steven Judd, program manager for NT. "The customer will pay more if the system is faster, and the customer will pay more if the system has more functionality," he explains. "They're not going to be willing to spend capital dollars to get no result."

Team resources are limited. Most teams don't have enough resources to fix each and every performance problem, so the performance PM must make some tough decisions in prioritizing issues. Developers sometimes want to fix problems that look bad statistically but won't be noticed by customers. "It's important not to get caught up in the absolutism of the [benchmark] numbers," emphasizes Paul Vick, development lead for Visual Basic for Applications and former performance lead developer for Access 97. "If a feature goes from instantaneous to semi-instantaneous, that really isn't a problem. You have to look at things from the customer's point of view."

There may be little or no control over problems created by outside components. As more products rely on shared components, sometimes the performance PM has no control over a problem with a component created by another team. While that other team might have its own resource or deadline problems, it also may be reluctant to redesign a component for other teams that already works well. These types of problems can be hard to fix, Paul says. "A lot of performance problems are design issues rather than stupid coding problems."

Performance work must conform to ship date schedule. Like any other product feature, performance work may come head-to-head with the product's ship date, and the performance PM must ultimately decide whether these issues should delay the release of the product. Complicating this problem is the knowledge that any major changes to the code late in the product cycle can affect the stability of the product. "Performance is just like any other feature," says James Sturms. "You have to weigh the benefits of the feature [performance] against the risk of instability."

Spec'ing for performance

One of the key duties of a performance PM is writing the performance specs for the product under development. While the performance specs consist of hard, cold numbers, spec'ing is as much an art as a science. The PM must select which features to spec, decide on performance goals, and revise those goals during the development cycle.

- Selecting features to spec
- Setting performance goals

Selecting features to spec

Out of the hundreds of features making up a product, performance PMs must decide which features need performance goals as well as milestones and deadlines. Several sources of information can help them in this prioritization:

Customer feedback is probably the most important method for identifying features. "We concentrate on areas that are typically very important to customers," says Craig Unger of Access. These features are selected through input from marketing and customer support. As a result, Access tests the performance of more than 130 feature areas.

Trade magazines also play a role in selecting the features to benchmark, explains Paul Williams, test manager for Base Technologies shared feature team of Office. Often these publications conduct head-to-head comparisons of competing products. While the methodology of these tests is sometimes suspect, they nonetheless do influence customer purchasing decisions.

Industry standards play a major part in the database industry. Many corporations base their buying decisions on test results of the Transaction Processing Performance Council (TPC), a nonprofit corporation founded to define transaction processing and database benchmarks, explains Damien Lindauer, performance engineering manager for SQL Server. While the TPC tests tend to concentrate on high-end use of database servers that are beyond the requirements of most customers, they do give users an idea of the maximum performance of the various database products.

Benchmarks by independent software vendors (ISVs) are important to the database market, according to Damien. The ISVs, including SAP AG and the Baan Company, test their enterprise systems with several database backends, including SQL Server and Oracle, to give customers performance and price-performance information. These results are gaining importance because they reflect real customer workloads, he adds.

Setting performance goals

After the key features are selected, the performance PM must determine performance benchmarks for each feature. If the product has a direct competitor, the PM wants the product to beat the competitor's, and sets the performance bar accordingly. But if there isn't such a competitor, the version under development is usually pitted against the previous release. This means that features are expected to remain as fast as before, if not to improve slightly.

When features are added to a product, performance of existing features often degrades—sometimes a drop in performance of up to 100% can be expected. Steven Judd of NT says there are instances where the performance PM doesn't have any meaningful data to devise realistic goals. In such cases, the PM may have no other choice than to spec a 50% decrease in performance and revise the goals as the product is developed.

However, such decreases are acceptable only when the product gains new functionality and improves in other areas. Tim Moore, PM for Windows NT (Winsock, quality of service, and multicast), echoes that sentiment: "If you gain more somewhere else, then you're probably all right." He adds, "Improvements in some network components often help one group of users while hurting others."

What does the successful performance PM do?

Just a few years ago, performance tuning was rarely started until after a product was code-complete. That just doesn't work anymore, especially as products get more complicated. Here are some of the things that performance PMs—or any PMs who need to worry about product performance—do to stay in front of the performance eight ball:

- Start the planning early.
- Work with developers to find creative solutions.
- Test the product in the real world.
- Monitor the product's performance closely between builds.
- Develop a system for prioritizing performance issues.
- Be prepared to work with other teams.
- Communicate performance goals to the developers.
- Learn from failures.

Start the planning early. Performance is tough to fix once the product's development is well under way, says Art Shelest, network transports program manager for Windows NT. "Keep performance in mind during the design stage, because a lot of decisions you make will be hard to reverse once they are implemented." Grant George, test director for Office, points out it's not realistic to fix major performance problems on a product nearing completion. "One thing we learned with Office 97 is that if we didn't pay attention to performance early in the product cycle and treat it like a feature area, we couldn't expect to deliver on it late in the game."

Work with developers to find creative solutions. A performance PM can encourage developers to look for innovative ways to boost performance and increase user satisfaction. As with many Office products, the Access team found that loading fewer features at startup helped to improve performance significantly for most customers. "Delay loading [a feature] until you absolutely need it," advises Craig Unger of Access. "The person who uses that feature pays the penalty anyway. But it makes life better for the person who doesn't use it."

Test the product in the real world. While performance lab tests are good to identify problems, nothing beats getting a product under development into the hands of a small group of customers for real-world testing, says Steven Judd. Through its Rapid Deployment Program, NT has identified problems that went by undetected in performance models. "You can have a system that benchmarks very well but doesn't respond to the customer's satisfaction," he explains. "You can crunch numbers and come up with wrong answers, even though the crunching you did was right."

Paul Vick of Visual Basic says it's important for the development team to get a feel for a product's performance on a typical user's machine. While he agrees that testing must be done on pristine test machines that can provide reproducible results, the PM should run "reality checks" on the product to avoid problems when customers run the product on less-than-ideal machines. "You'll get a reality check when you send out the beta, so you might as well find out sooner," he counsels.

Monitor the product's performance closely between builds. Performance issues must be closely monitored during the development cycle in order to identify and fix problems, advises Craig Unger. "The most important thing is to track trends in performance over time." Program managers should be wary of developers who promise that performance will improved once more features are complete, warns Paul Vick. "It never works that way. It really helps to have performance targeted as you go along."

Develop a system for prioritizing performance issues. Most teams treat performance problems as bugs. The Jet database engine team has an elaborate system for determining the severity of performance bugs, says Program Manager Kevin Collins. After a build undergoes 1,800 performance tests over a 1½-day period under nine memory configurations in Windows 95/98 and NT 4.0, bugs are reported only on features that degraded on all memory configurations. A high-priority bug is a feature whose performance degraded at least 20% and took five seconds or more to run. A low-priority bug is a feature that degraded 5% or more and increased one second or more.

Still, according to Kevin, it's not enough to look at percentage increases. "A 100% increase in a fraction of a second doesn't mean much," he says. However, that doesn't mean that small problems don't matter. Kevin looks for ways to develop lengthy test scenarios to test features that execute fast. That way, he knows a small problem won't escalate into something major.

Be prepared to work with other teams. As mentioned above, the solution to a performance problem may lie in the hands of another team, so a performance PM has to be a good diplomat and negotiator, observes Paul Vick of Visual Basic. "We're all part of the same company, so your success is my success," he says. "But it's not easy if I've got my own deadlines to meet." A good PM identifies the problem early and is prepared to look at creative solutions that could even include lending staff to a team facing a deadline. "It's the general magic of planning," says Paul.

Steven Judd of NT says a PM should have the problem clearly identified before handing it over to another team. "You've got to make their job as easy as possible," he explains. "You need to be sensitive to the fact that everyone is under a lot of pressure... You also want to give them the solution so that it minimizes the work they have to do."

Communicate the performance goals to the developers. A performance PM must be able to clearly communicate and emphasize the performance goals to all developers. "It's not always clear to the developer what values the PM brings to the table," says Craig Unger of Access. "It's up to the PM to prioritize, keep the right things on the developer's plate."

Learn from failures. Nobody likes failure, but it's extremely important to know why your team didn't meet its goals so that future mistakes can be avoided, says Damien Lindauer. "You have to go back and analyze why you missed [meeting a goal]," he explains. "Did you do incorrect planning? Did you not know enough of the factors? Or did you make an incorrect assumption?"

Necessary tools for the performance PM

A good performance PM needs a solid infrastructure to accomplish the difficult task of making the product run faster.

- Upper management's support
- A good test lab
- Easy-to-understand and timely test results
- Stable builds
- A solid lead performance developer

Upper management's support: Any attempt to build performance into a product isn't likely to go far unless senior management is on board, says Paul Williams, test manager for Office. Support from Jon DeVaan, vice president of desktop applications, let everyone within Office know that performance was important, adds Jim Walsh, development manager for Office. "That support made it a lot easier to get the budget and headcount that we needed."

A good test lab: Good, solid, irrefutable benchmark numbers are the only way to convince developers and other team members that the product has a problem. The methodology for producing the numbers must be beyond reproach. In addition, the test lab must be able to prove beyond any doubt that it is testing the performance of the software and not accidentally measuring hardware performance, cautions Paul Williams. "Once you've got people believing your results and trusting your numbers, you can start to concentrate on the results and not the process of getting those numbers." From the developer's perspective, "if the numbers are not reliable, then what's the point in worrying about them?" says Paul Vick of Visual Basic.

A well-designed test lab does more than tell developers that there's a problem; it shows them where the problem originates. After the Jet performance lab runs all 1,800 tests, it isolates the DLL or other software component responsible for the performance decrease. "We try to give the developers the maximum amount of information," Kevin Collins says of the fully automated Jet test lab system with 36 computers. Between each set of tests, the system reformats the hard drives, reinstalls the operating system, and isolates the test machines from the network. "We like to operate in a perfect, pristine environment," he explains. The Jet database engine is used in dozens of Microsoft products.

Good test results are especially imperative to convince another team that the problem lies with their component, adds Kevin. "The best thing you can do is have a definitive, reproducible scenario. We can't do much with a bug that we can't reproduce."

Easy-to-understand and timely test results: As more people become involved in performance, the lab results must be simple to use and understand. "You just can't give everyone an incredibly complicated matrix of numbers," says Paul Williams of Office. Besides being easy to use, the results must be provided to the developers quickly, adds James Sturms of Access. "If my developers are three builds beyond the results, too many things have changed since the numbers were released."

Stable builds: Stable builds during the development cycle are crucial for test lab to identify performance problems. "You can't performance-test a product that you can't run," notes James Sturms. Office 2000 achieved greater stability by debugging newly introduced features before proceeding to the next milestone, says Grant George, test director for Office. However, some developers aren't crazy about this system, as it can leave them waiting for one milestone to be completed before launching into the next, explains Paul Vick. As an alternative, he suggests giving developers a block of time during development to fix performance issues.

A solid lead performance developer: Every performance PM needs a good performance developer who isn't afraid to spend many hours searching for the obscure reason behind a performance problem. Many products have one performance developer who identifies possible causes o performance problems and then turns the findings over to the developer most familiar with the code. "It's a lot of detective work," says developer Paul Vick. "You need someone who likes to solves mysteries and is really motivated to keep asking, 'Why is that happening?'"

It isn't always easy to find such a person, short of luring them away from another team. It may be necessary for the PM to encourage one or more developers to train for the position by learning more about performance issues, adds Paul. "Being a performance lead is a great position to be in if you are interested in moving up the ladder."

Other performance and planning links:

Office .NET Performance Site

Performance Information

Access Performance Specs

The *Interface* team thanks the following contributors:

Kevin Collins	Program manager for Jet
Grant George	Test director for Office
Steven Judd	Program manager for Windows NT

Damien Lindauer	Performance engineering manager for SQL Server
Tim Moore	Winsock, quality of service, and multicast program manager for Windows NT
Art Shelest	Network transports program manager for Windows NT
James Sturms	Program manager for Access
Craig Unger	Product team manager for Access
Paul Vick	Development lead for Visual Basic for Applications
Jim Walsh	Development manager for Office
Paul Williams	Test manager for Office

Microsoft confidential. Intended for internal use only. |