Performance gains at the cost of other components

devblogs.microsoft.com/oldnewthing/20050311-00

March 11, 2005

Raymond Chen

In the operating systems group, we have to take a holistic view of performance. The goal is to get the entire system running faster, balancing applications against each other for the greater good. Applications, on the other hand, tend to have a selfish view of performance: "I will do everything possible to make myself run faster. The impact on the rest of the system is not my concern." Some applications will put themselves into the Startup group so that they will load faster. This isn't really making the system run any faster; it's just shifting the accounting. By shoving some of the application startup cost into operating system startup, the amount of time between the user double-clicking the application icon and the application being ready to run has been reduced. But the total amount of time hasn't changed. For example, consider the following time diagram. The "*" marks the point at which the user turns on the computer, the "+" marks the point at which Explorer is ready and the user double-clicks the application icon, and the "!" marks the point at which the application is ready.

*	OS Startup	+	Application Startup	!

The application developers then say, "Gosh, that pink 'Application Startup' section is awfully big. What can we do to make it smaller? I know, let's break our application startup into two pieces...

*	OS Startup	+	Application Startup 1	Application Startup 2	!

"... and put part of it in the Startup group.

*	OS Startup	Application Startup 1	+	Application Startup 2	!

"Wow, look, the size of the pink bar (which represents how long it takes for our application to get ready after the user double-clicks the icon) is much shorter now!" The team then puts this new shorter value in their performance status report, everybody gets raises all around, and maybe they go for a nice dinner to celebrate. Of course, if you look at the big picture, from the

asterisk all the way to the exclamation point, nothing has changed. It still takes the same amount of time for the application to be ready from a cold start. All this "performance" improvement did was rob Peter to pay Paul. The time spent doing "Application Startup 1" is now charged against the operating system and not against the application. You shuffled numbers around, but the end user gained nothing. In fact, the user **lost** ground. For the above diagrams assume that the user wants to run your application at all! If the user didn't want to run your application but instead just wanted to check their email, they are paying for "Application Startup 1" even though they will reap none of the benefits. Another example of applications having a selfish view of performance came from a company developing an icon overlay handler. The shell treats overlay computation as a low-priority item, since it is more important to get icons on the screen so the user can start doing whatever it is they wanted to be doing. The decorations can come later. This company wanted to know if there was a way they could improve their performance and get their overlay onto the screen **even before the icon shows up**, demonstrating a phenomenally selfish interpretation of "performance".

Performance is about getting the user finished with their task sooner. If that task does **not** involve running your program, then your "performance improvement" is really a performance impediment. I'm sure your program is very nice, but it would also be rather presumptuous to expect that every user who installs your program thinks that it should take priority over everything else they do.

Raymond Chen

Follow