i-Technology Viewpoint: Why Java Is Not Slow

Not to directly pick on Peter Bell at all (he's a very smart guy and I really enjoy his blog, so this is nothing personal at all), I just noticed something he mentioned in a recent comment on a blog entry and felt that it was something I should address in a blog post rather than a comment. Also, writing this entry gives me a good reference point later to send to people when they present the same talking point to me.

Anyhow, here is the comment that Peter made which inspired this blog entry (which in context, is a quip):

> "I also have to question why you'd even consider using CF for a performance critical app - even Java is a little sluggish. If you want bare metal performance and think your time is worth less than the cost of a server, go for it and write the darn app in C++!"

I wanted to point out and correct a common fallacy that I see repeated over and over again are statements rooted in comments like "If you need speed, you could convert your Java application to C or C++". I don't claim to be a Java expert by any means, but I do think I have a good enough understanding of the Java platform as well as basic operating systems and compiler theory and the like to explain why this is a false argument. The answer is actually more along the lines of "it depends on what you are doing", and I'll explain why.

It's very easy to see where this misconception started – pretty much every software developer or system administrator has encountered a slow Swing/AWT based Java application which took forever to start, and generally seemed pretty sluggish. Not to mention it probably looked terrible. One of the best Java based desktop applications, Eclipse, even takes awhile to start. On the contrary, C++ desktop applications typically start very quickly, and are very responsive to user input. Generally speaking the symptoms of running a Java desktop application match the reality– C++ is almost always faster for short running programs such as desktop applications.

Another common assumption is that it would seem that the JVM itself is an abstraction above machine level code because it acts as an interpreter, and that C/C++ will always be faster because the resulting compiled code is "closer to the hardware" which is also false. In the early days of Java, the JVM did in fact interpret code, but eventually it started using a Just In Time compiler in Java 1.2 to compile the code down to the machine level. Different implementations of the JVM use various JIT compilers, but the Sun JVM uses the HotSpot JIT. Not only are Java programs compiled down to machine specific code, but the longer a program runs, the better it gets optimized.

The HotSpot compiler monitors and analyzes an applications performance over time, and uses adaptive optimization to eliminate bottlenecks in the machine code, known as "hotspots". It essentially looks for pieces of code that are ran time and time again, and marks them for optimization. Those pieces of code are then recompiled again into machine specific code for performance. This allows for ongoing performance enhancements that you would never get from machine code produced by a C or C++ compiler such as branch prediction hints.

All of these things are not just unfounded theory, there have been several independent studies about

this including this one by the University of Southern California.

> Java is now nearly equal to (or faster than) C++ on low-level and numeric benchmarks. This should not be surprising: Java is a compiled language (albeit JIT compiled).

This study is several years old now and doesn't include testing against the many performance enhancements which were made available in the 1.5 JVM, so I'd be interested in seeing an updated study.

That said, Java is probably much faster than you think. One way I've seen the Hotspot compiler in action (and you can try at home) is when working on a simple "Hello World" JSP and keeping track of it's execution time. Typically on the initial run or two, the JSP page is pretty quick, but then after a few calls of the page the Hotspot compiler kicks in and makes it wicked fast so that it's execution time is effectively nil.

Now that we've spoke to the "Java Is Slow" audience, we should cover the answer to the earlier question of "Is Java slower than C/C++?". This is more of an "it depends" answer. Generally speaking C++ has the advantage in short running applications, and Java is as fast if not faster than C/C++ in long running applications such as servers, web applications, etc. It also might surprise you to know that Java is also very good in real time systems, but that is another blog post of its own.