A License to Kode

While it's sometimes tempting to blame the coders, the seeds of many problems are sown well before any lines of code (dodgy as they may be) have been written. Everything from the choice of tools to the choice of a software license can affect the quality, usability, and commercial potential of a product. This month Kode Vicious takes a step away from coding technique and addresses some of these tough decisions with which developers must grapple. Have something you'd like to ask KV? Please send your missives to kv@acmqueue.com.

Dear KV,

I'm in the QA group for a medium-size startup in Silicon Valley, and one of our VPs sits on the board of a company that makes code-scanning software—you know, the stuff that spits out warnings about all the bad things you can do in C and C++. We've definitely found our share of buffer overflows and other problems in our code, but this stuff is expensive—more than \$5,000 a seat—and I'm just not sure it's worth it.

What do you think of these tools?

Scanning for an Answer

Dear Scanning,

Over the years there have been both free and for-pay programs that would try to point out potential problems in people's code. Though the state of the art has certainly advanced since the Lint program was written, there has consistently been one failing in the use of such programs, and that is the programmers themselves.

In the same way that people build systems to log data, and then never look at the logs, most people buy expensive and fancy tools to make their code faster, better, or more secure, only to ignore the advice that the tools give. I was recently working with an engineer on a project

Got a question for Kode Vicious? E-mail him at kv@acmqueue.com—if you dare! And if your letter appears in print, he may even send you a Queue coffee mug, if he's in the mood. And oh yeah, we edit letters for content, style, and for your own good!

A koder with attitude, KV ANSWERS

YOUR QUESTIONS.

MISS MANNERS HE AIN'T.

using an open source tool to scan his team's source code for possible buffer overflows and other security vulnerabilities. At

one point the tool pointed out more than 100 possible flaws in the system. The flaws were all identical, and, it turned out, false positives. Instead of using that age-old programming paradigm, the function call, to encapsulate the offending code and thereby reduce the number of false positives from an annoying 100 to a tolerable 1, the engineer asked if we could "fix" the scanning tool so that it would not flag these false positives.

It is at such moments that I'm happy to be bald, because if I weren't, I'd be able to tear my hair out, and that hurts, or at least it used to. The impulse to "fix" the tool when it is giving plenty of good advice on other areas and not to fix the code, or, more importantly, not to fix the poor practices that lead to software problems, is going to be the biggest problem you will have.

So, my answer to you is, "It depends." That is, it depends on what type of shop you work in. Do you work in a shop where people will see such tools as a help or as a threat? If people fight the tools, then they are worthless. If your programmers are willing to learn from the tools and to change their behavior, then the tools might be worth what you will pay for them. If your people need behavior modification, then I suggest you look to the film *A Clockwork Orange* for guidance.

Just remember, Beethoven never coded a buffer overflow.

KV

Dear KV,

I run a small set of Web servers at home, hosting Web pages for friends. I've been building a small package for analyzing Web logs over the past few months, in my "spare" time of course. Now I want to post it on SourceForge, which requires an open source license on the code. I was going to put it under the GPL (GNU public license), but then I read about all the fighting over the GPL3 and I was wondering if this would affect my package. I did a

search for other licenses, and it seems that everyone has their own.

How do you decide which one to use?

Unlicensed

Dear Unlicensed,

First, let me state, right up front, I am *not* a lawyer and have never played one on TV. I know some lawyers—purely on social, well, really, drinking, terms—but that’s the extent of my legal knowledge. With that caveat out of the way, which will hopefully placate the lawyers at *Queue* (with whom I have never had a drink), let me try to answer your question.

Personally, I use the BSD (Berkeley Software Distribution) license, also known as the “New BSD” license (see <http://www.opensource.org/licenses/bsd-license.php>). I don’t just use the BSD license because I’m an old cranky BSD coder, though I am, but because it gives me and my code the greatest amount of freedom with what I perceive to be the least amount of risk. I believe most programmers want people to use their code and to feed

Charybdis—for those of you who have forgotten Greek mythology, it puts you between a rock and a hard place. Big Corporate licenses are completely closed; you use the code on their terms. But the GPL is similar in that it is dictating what you do with your code—the code *you* wrote—because it’s an extension of theirs. Most people call this a “viral” license, and many projects that don’t use the GPL are very careful to wall off GPL code from the rest of their code to prevent the virus from poisoning them.

There is also the route of putting your code in the public domain, which provides you no control and theoretically no protection. I’ve not done that since I started using the BSD license, because with the BSD license you’re also saying, “Don’t sue me! This has no warranty!” These are good things to say. You don’t want people integrating your code into their product and then looking to you for damages when they go out of business.

It’s my belief that many people use the GPL because it’s famous, not because they actually want to force anyone who touches their code to share with them. On numerous occasions I’ve contacted authors of libraries I wanted to extend to ask them if they could not use the GPL, and usually they said, “Sure, why?” Once they understood why, well, they usually changed to something less viral, sometimes even BSD.

Intellectual laziness of this sort is the very thing that leads people to use technologies for projects for which they are totally unsuited, like putting a mission-critical system on Windows, but I digress...

Of course, don’t take my word for it; ask a lawyer, if you’re truly paranoid.

Just so you know, “THIS ARTICLE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ‘AS IS,’ AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, ARE DISCLAIMED.”

KV

The best way I can describe the **GPL** is that it’s the roach motel of licenses: code checks in, but **it doesn’t check out.**

back patches and to stay out of legal trouble. All well and good, and that’s what the BSD license gives you. The GPL, well, that’s another kettle of hot boiling oil.

The best way I can describe the GPL is that it’s the roach motel of licenses: code checks in, but it doesn’t check out. Putting the GPL on your code means that not only are you sharing it, but also anyone who uses your code must share with you. Enforced sharing has always seemed to me like saying, “I’m taking my marbles and going home.” You see, the part of the GPL that’s problematic to me is that if I use your code, then I have to give you my code if I ship a product with your code in it. What if I am willing to give you only part of my code but for some reason I don’t want to give you all of it? You can’t do that with the GPL.

In reality, the GPL is the Scylla to Big Corporate’s

CODE VICIOUS, known to mere mortals as George V. Neville-Neil, works on networking and operating system code for fun and profit. He also teaches courses on various subjects related to programming. His areas of interest are code spelunking, operating systems, and rewriting your bad code (OK, maybe not that last one). He earned his bachelor’s degree in computer science at Northeastern University in Boston, Massachusetts, and is a member of ACM, the Usenix Association, and IEEE. He is an avid bicyclist and traveler who has made San Francisco his home since 1990.

© 2007 ACM 1542-7730/07/0200 \$5.00