Solving one problem by creating a bigger problem

devblogs.microsoft.com/oldnewthing/20060322-24

March 22, 2006

Raymond Chen

Often, people will not even realize that their solution to a problem merely replaces it with another problem. The quip attributed to Jamie Zawinski captures the sentiment:

> Some people, when confronted with a problem, think "I know, I'll use regular expressions." Now they have two problems.

For example, in response to "How do I write a batch file that..." some people will say, "First, install <perl|bash|monad|...>". This doesn't actually solve the problem; it merely replaces it with a different problem. In particular, if the solution begins with "First, install..." you've pretty much lost out of the gate. Solving a five-minute problem by taking a half hour to download and install a program is a net loss. In a corporate environment, adding a program to a deployment is extraordinarily expensive. You have to work with your company's legal team to make sure the licensing terms for the new program are acceptable and do not create undue risk from a legal standpoint. What is your plan of action if the new program stops working, and your company starts losing tens of thousands of dollars a day? You have to do interoperability testing to make sure the new program doesn't conflict with the other programs in the deployment. (In the non-corporate case, you still run the risk that the new program will conflict with one of your existing programs.) Second, many of these "solutions" require that you abandon your partial solution so far and rewrite it in the new model. If you've invested years in tweaking a batch file and you just need one more thing to get that new feature working, and somebody says, "Oh, what you need to do is throw away you batch file and start over in this new language," you're unlikely to take up that suggestion.

So be careful when you suggest a solution that has a high activation energy. Sure, something could be taken care of by a one-line perl script, but getting perl onto the machine is hardly a one-line endeavor.

Raymond Chen

Follow