

Re: Green Hills CEO: Linux threat to free world!

Source: <http://coding.derkeiler.com/Archive/General/comp.arch.embedded/2004-04/0587.html>

From: Guy Macon (<http://www.guymacon.com>)

Date: 04/10/04

Date: Sat, 10 Apr 2004 09:33:02 -0700

Lewin A.R.W. Edwards <larwe@larwe.com> says...

>

>

>> *Well, I don't recall ever installing an application onto Windows, finding it
>> required an updated DLL, and when that DLL is installed finding that other
>> applications installed earlier in the day mysteriously stop working, which*

>

>*ROFL! But Windows is _legendary_ for this kind of behavior!*

Here is what Steven Pratschner of Microsoft Corporation says about this issue:

http://msdn.microsoft.com/library/en-us/dndotnet/html/dplywithnet.asp?#dplywithnet_problem

(He correctly describes the problem but offers another Microsoft proprietary monopolistic solution).

Given the well know nature of this problem, Microsoft admitting that it is a problem, and his claim to be a long-time windows user, I am at a loss as to how Lewin Edwards came to the conclusion that he posted above.

Here is a quote from the Microsoft article. As you read it, think about high-reliability embedded systems and whether you want to use an OS with these sort of problems in such systems.

"From a customer perspective, the most common versioning problem is what we call DLL Hell. Simply stated, DLL Hell refers to the set of problems caused when multiple applications attempt to share a common component like a dynamic-link library (DLL) or a Component Object Model (COM) class. In the most typical case, one application will install a new version of the shared component that is not backward compatible with the version already on the machine. Although the application that has just been installed works fine, existing applications that depended on a previous version of the shared component might no longer work. In some cases, the cause of the problem is even more subtle. For example, consider the scenario where a user downloads a Microsoft ActiveX® control as a side effect of visiting some Web site. When the control is downloaded it will replace any existing versions of the control that were present on the machine.

If an application that has been installed on the machine happens to use this control, it too might potentially stop working."

"In many cases there is a significant delay before a user discovers that an application has stopped working. As a result, it is often difficult to remember when a change was made to the machine that could have affected the app. A user may remember installing something a week ago, but there is no obvious correlation between that installation and the behavior they are now seeing. To make matters worse, there are few diagnostic tools available today to help the user (or the support person who is helping them) determine what is wrong."

"The reason for these issues is that version information about the different components of an application aren't recorded or enforced by the system. Also, changes made to the system on behalf of one application will typically affect all applications on the machine—building an application today that is completely isolated from changes is not easy."

"One reason why it's hard to build an isolated application is that the current run-time environment typically allows the installation of only a single version of a component or an application. This restriction means that component authors must write their code in a way that remains backward compatible, otherwise they risk breaking existing applications when they install a new component. In practice, writing code that is forever backward compatible is extremely difficult, if not impossible."

--Microsoft

--

Guy Macon, Electronics Engineer & Project Manager for hire.
Remember Doc Brown from the Back to the Future movies? Do you have an "impossible" engineering project that only someone like Doc Brown can solve? My resume is at <http://www.guymacon.com/>