COM Plugin - Feature #418

32/64 switch

02/08/2012 04:29 PM - Omar van Galen

Status:	Closed	**Start date:**	02/08/2012
Priority:	Normal	**Due date:**	
Assignee:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			
Browser (if web client):			

Description

I would like to submit a feature request. It would be nice to be able to tell the ComPlugin which Jacob dll to load the 32-bit version or the 64-bit version.

History

#1 - 02/08/2012 04:52 PM - Patrick Talbot

- Description updated

- Status changed from New to Closed

What you are asking here makes no sense.

The architecture of the DLL must match the architecture of the JVM, you don't pick and choose like that!
When the plugin is loaded that choice has already been made: either your Java architecture is 32-bit in which case the 32-bit DLL must be loaded, or your Java architecture is 64-bit in which case the 64-bit DLL must be loaded.

#2 - 02/09/2012 11:28 AM - Omar van Galen

Patrick,

Thanks for your patience ;-) I supposed that the COM component would be run in a seperate process just like Windows does with 32-bit dll's in 64-bit Windows. If what you say is true this means that it will be impossible to use Visual FoxPro dll's (there will be no 64-bit version) in Servoy for all users with 64-bit systems unless there is a way to force Servoy to run in 32-bit mode entirely?

Regards,
Omar

#3 - 02/09/2012 01:50 PM - Patrick Talbot

Yes, what I say is true. There's no if about that, believe me.

And there's no such thing as forcing the architecture in Java on Windows...

On Mac OS X you have a JVM switch that you can use (-d32 or -d64), but this is non standard Java command line parameter.

What you can do is ask your users to install a 32-bit JRE, pointing them to the correct download.

An external process using an external JVM that would use a Socket to communicate with the main (Servoy) app may be designed but that's an awful lot of work. In any case the 32-bit JRE would have to be pre-installed as well (unless you want to ship it yourself).

Another way is to use the target DLL on the server side and communicate with it using RMI, encapsulating all the calls to the distant DLL using a proxy object, but then your server would also need to run in 32-bit, which would limit the available heap space available server-side.

Now one good thing to note is that if your users have both 32 and 64-bit JVM, there's a fair chance that the JVM that will be started is the 32-bit one, especially if they are using Internet Explorer to launch your jnlp URL.
This is because on Windows 64-bit, Internet Explorer 32-bit is still the default browser (you have a link to the 64-bit version but you need to launch it explicitely), so by default your users are in a 32-bit launching environment.

You could also creat a bat file that would launch the 32-bit javaws with your URL (but once again you would need a 32-bit JVM to be present and your process would need to explore the registry or the C:\Program Files (x86)\ folder to find the location of the Java 32-bit executable, an installer could probably do that...