The Standard in Industrial Automation and Scientific Components for Real-Time Applications

.NET WinForm Licensing & Troubleshooting Guide

Copyright

Trademarks

Printing

Initial Printing   09/2010

Manufactured in the United States of America

Table of Contents

This page left intentionally blank.

Chapter 1 – Introduction

Welcome to the .NET Licensing & Troubleshooting Guide. Licensing in the .NET framework can be very confusing and frustrating to novice as well as seasoned .NET developers. The documentation provided by Microsoft on the .NET licensing model is minimal at best, difficult to find, and confusing to follow. Microsoft designed the .NET licensing model to be much more powerful than their previous versions, but this has lead to excessive hair pulling and many sleepless nights for legions of software developers.

The .NET development environment is lacking in many situations when it comes to adding and maintaining the license keys for you. For many cases, you will need to perform manual steps to force an update or addition to the license file. There are also cases where you will need to delete the license file and force its regeneration because of bugs in the development environment which will result in the corruption of the license file (The file will look normal, the corruption is not detectable).

To make matters even worse, the .NET development environment will add a license file to each DLL sub-projects used by your EXE project. This sounds like Microsoft is trying to help you out here, but the opposite is the case. Any license key added to one of these DLL sub-projects is never used by your application. This quirk of the development environment causes the greatest amount of confusion by .NET developers when it comes to licensing.

Microsoft has placed the burden on you, the .NET developer, to fully understand the licensing model. You will need to understand the cases when the license keys will be automatically added and maintained for you. You will need to know the cases where you will need to manually insert and update the license keys. And finally, you will need to understand how to detect when the license files becomes corrupted and know how to regenerate it from scratch.

This document covers the licensing model for all Microsoft, Borland, CodeGear, and Embarcadero .Net development environments. This document only covers the Microsoft .NET development environments in detail. You can apply the same principles in the Borland, CodeGear, and Embarcadero .Net development environments since they are licensed from Microsoft and differ in appearance only.

Contacting Iocomp Software

You can always contact Iocomp Software support staff directly for assistance with our software products at the following telephone numbers and Internet addresses…

- USA & Canada Toll Free Telephone: 888-599-2929
- International Telephone: +1-407-445-2809
- Customer Support Email: support@iocomp.com
- Other Iocomp Products and Upgrades: http://www.iocomp.com/products
- Customer Support Website: http://www.iocomp.com/support

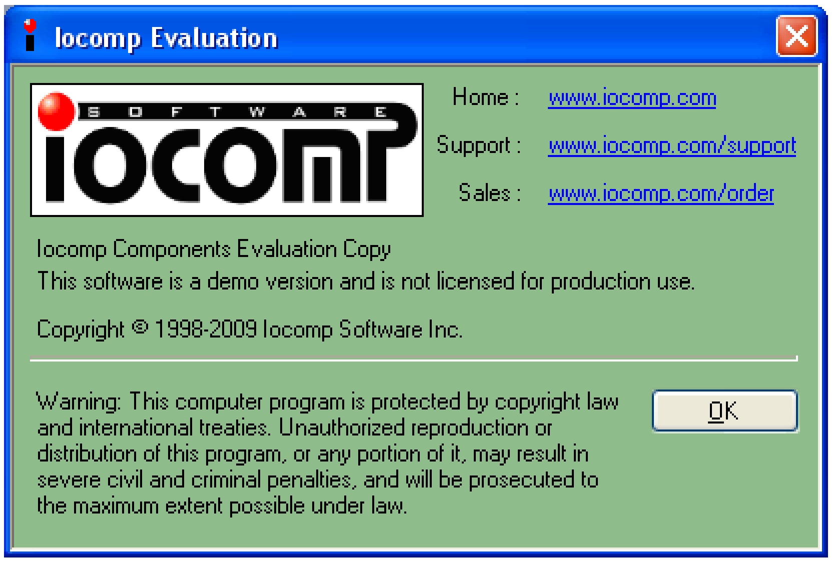
Chapter 2 – How .NET Licensing Works

Overview

Each licensed control has its own unique license key. As you build your application and add licensed controls, an entry is added to a project file called “**licenses.licx**”. The “**licenses.licx**” file does not store the actual license keys, just an entry for each control. When you compile your application, the license keys are gathered, encrypted, and embedded into your application’s binary EXE file (The license keys are encrypted to prevent end-users from extracting the license keys and using them to license controls they have not purchased.).

When your application is initially started on a deployment machine, the license keys are extracted, decrypted, and stored in the applications memory. As the licensed controls are created one by one, the license keys stored in the applications memory are used for properly licensing them. This architecture allows the controls to execute on the deployment machine with the necessary license keys, and at the same time, preventing the end-user from accessing the licensing information.

If your application does not contain a license key for a particular Iocomp control, the Iocomp controls are designed to still function, but they will present a nag screen every 10 minutes (**See Figure 2.1**). Other 3rd party controls may or may not nag, and depends on how they were designed. Some 3rd party controls will raise an exception instead, and the control will not be created. This can cause your application to terminate if you do not handle the exception.


Iocomp Evaluation
Home : www.iocomp.com
Support : www.iocomp.com/support
Sales : www.iocomp.com/order
OK

Iocomp Components Evaluation Copy
This software is a demo version and is not licensed for production use.

Copyright © 1998-2009 Iocomp Software Inc.

Warning: This computer program is protected by copyright law and international treaties. Unauthorized reproduction or distribution of this program, or any portion of it, may result in severe civil and criminal penalties, and will be prosecuted to the maximum extent possible under law.

Figure 2.1
Iocomp Licensing Nag Screen.

You will never see licensing issues on your development machine. This is because the development machine is already licensed for the controls you are developing for. You will only see the licensing issues on the deployment machine when the license keys are not properly embedded in your applications EXE file.

If you do see the nag screen on the development machine, this does not indicate a licensing issue; it indicates you are still using one of the Iocomp products in evaluation mode. Example, you evaluated the Iocomp Ultra Pack, purchased the Std Pack, and left one of the Ultra Pack controls on one of your forms. In this case, you are going to get the nag screen when working on the project or executing it on your deployment machine. Only after you remove the control from the form and remove the references for it will the nag screens disappear.

The “licenses.licx” File in Detail

The “**licenses.licx**” file is where your project stores a list of license keys which need to be compiled into your application EXE file. This file is hidden by default (In a VS2003 or VS2005 C++ application, the file is always visible.). To make the file visible, you need to click the “Show All Files” button on the Solution Explorer mini-toolbar (See **Figure 2.2**). Note: By default, a project does not have a “**licenses.licx**” file. The file is created when you add your first licensed control to a form.

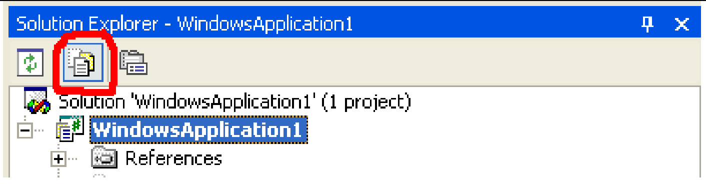
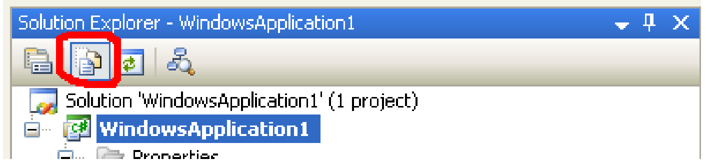
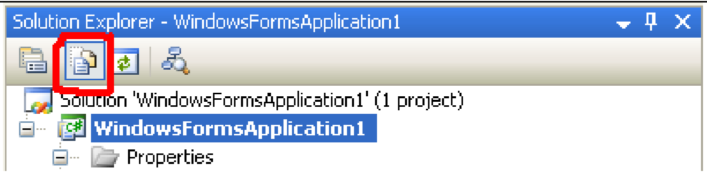
Dev Environment	Screen-Shot
VS2003	 Solution Explorer - WindowsApplication1; Solution 'WindowsApplication1' (1 project); WindowsApplication1; References
VS2005	 Solution Explorer - WindowsApplication1; Solution 'WindowsApplication1' (1 project); WindowsApplication1; Properties
VS2008	 Solution Explorer - WindowsFormsApplication1; Solution 'WindowsFormsApplication1' (1 project); WindowsFormsApplication1; Properties

Figure 2.2
Solution Explorer “Show All Files” button.

The license file is contained within your main EXE project (Note: Other projects may also contain a license file, but they are never utilized). **Figure 2.3** shows a typical project with a license file.

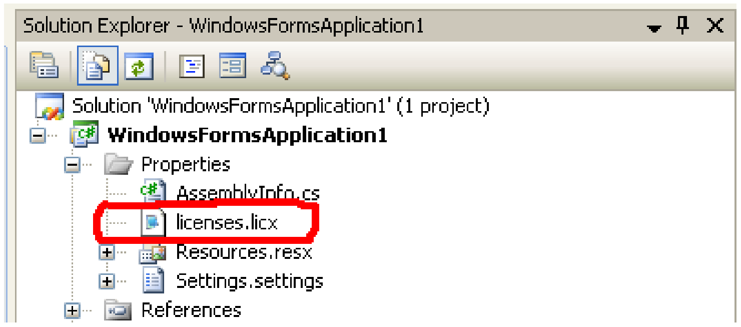

Solution Explorer - WindowsFormsApplication1
Solution 'WindowsFormsApplication1' (1 project)
WindowsFormsApplication1
Properties
AssemblyInfo.cs
licenses.licx
Resources.resx
Settings.settings
References

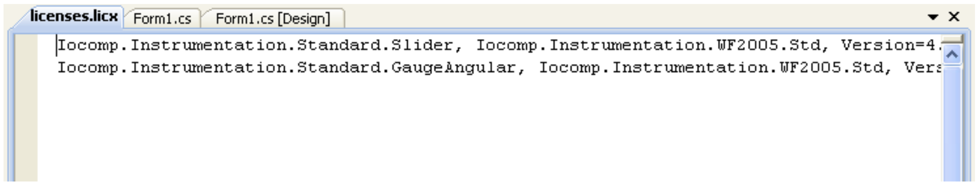
Figure 2.3
The “**licenses.licx**” file inside a VS2008 C# application.

The exact location within your project depends on the version of Visual Studio you are using and the type of project you have (See **Figure 2.4**).

Dev Environment	Language	Location
VS2003	C#	Root
	VB.Net	Root
	C++ Managed	Root
VS2005	C#	Properties Folder
	VB.Net	My Project Folder
	C++ Managed	Root
VS2008	C#	Properties Folder
	VB.Net	My Project Folder
	C++ Managed	Root

Figure 2.4
Location of the “**licenses.licx**” file within a project.

Double-Click the “**licenses.licx**” file to open it. See **Figure 2.5** for a sample “**licenses.licx**” file. This sample contains 2 license keys, one for the Iocomp Slider control and the other for the GaugeAngular control.



```
Iocomp.Instrumentation.Standard.Slider, Iocomp.Instrumentation.WF2005.Std, Version=4.
Iocomp.Instrumentation.Standard.GaugeAngular, Iocomp.Instrumentation.WF2005.Std, Vers
```

Figure 2.5
Sample “**licenses.licx**” file. Contains 2 license keys, one for the Slider control and the other for the GaugeAngular control.

The entire first line looks like this…

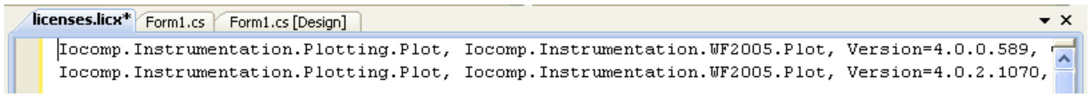
```
Iocomp.Instrumentation.Standard.Slider, Iocomp.Instrumentation.WF2005.Std,
Version=4.0.1.40961, Culture=neutral, PublicKeyToken=a217a64fde564c89
```

The first 3 items are the critical pieces of information that you will have to deal with. The last 2-pieces of information are not critical, and there is not a case where you would need to utilize them. The 1st piece of information is the name of the control including its full namespace. The 2nd piece of information is the name of the assembly (DLL) the control is located in. The 3rd piece of information is the specific version of the assembly the control is located in.

The entire .NET framework was designed to be very version aware, and this also applies to the licensing model. This was done to eliminate the “DLL Hell” issues of past and to also allow vendors to charge a fee per update if they wish (Iocomp does not charge a fee for every update). It is critical the version of the assembly you distribute exactly matches the version listed in the license file. If they don’t, the controls will become unlicensed

When Iocomp controls become unlicensed, they will display a nag screen every 10 minutes (See **Figure 2.1**).

Over time, as you develop, upgrade, and maintain your application, your license file can start to accumulate multiple entries for a single control. See **Figure 2.6** for an example of a license file containing 2 entries for a plot control. The only difference between the 2 entries is the version of the assembly. In the sample below, the first entry is for the “4.0.0.589” version, and was probably the first version the application was developed with. The last entry is for the “4.0.2.1070” version, and is most likely the current version the application is being developed with. There is nothing wrong with having both versions listed in the license file. Having multiple entries will result in 2 license keys being compiled into the final application. You can delete the duplicate entries if you wish. The critical thing is to have an entry that exactly matches the version of the assembly you are going to distribute with the EXE.



```
Iocomp.Instrumentation.Plotting.Plot, Iocomp.Instrumentation.WF2005.Plot, Version=4.0.0.589,
Iocomp.Instrumentation.Plotting.Plot, Iocomp.Instrumentation.WF2005.Plot, Version=4.0.2.1070,
```

Figure 2.6
A “**licenses.licx**” file with multiple entries for a control.

How License Keys are Automatically Inserted and Updated!

By default, a project does not have a "**licenses.licx**" file. The file is created when you add your first licensed control to a form, user-control, and etc. From then on, the license file is only updated when you add a licensed control that is not currently used by your application. This type of updating logic is severely flawed and causes most of the .NET licensing issues.

.NET Licensing Flaws!

When you first start developing your application, the flaws are not an issue. As you develop your application over time, or increase the complexity, several things can occur to cause the licensing to become invalid. Below are 3 cases in which the licensing can become invalid.

Case 1:

The first case is updating to a newer version of the Iocomp controls. Let's say you originally used the Iocomp "4.0.0.489" version. Your licensing file will have entries for each Iocomp control, and the version of the assembly on each line will be "4.0.0.489". So far, everything is correct and your application can be deployed without any licensing issues. Let's now upgrade your development machine to use the Iocomp 4.0.2.1070 version. You will run the Iocomp installer and the newer versions of the Iocomp controls will be installed on your hard-drive. Unlike past technologies, like ActiveX, your application will not automatically start using the newer version. You will need to change out the Iocomp assemblies listed in the project's References section. Once the references are switched out, you build your application, execute it on your development machine, and everything will function normally. But once you deploy your application, along with the newer versions of the assemblies, it begins to nag immediately, indicating a licensing issue. What went wrong? If you open the project's "licenses.licx" file, it will still show the older "4.0.0.489" version. Because the version in the license file does not match the version of the deployed assembly, the .NET licensing model does not pass the licensing key along to the control at run-time for licensing the control. It does not matter that the license key stored in the "licenses.licx" file matches the name of the control being created on the deployment machine, if the versions are different, they are considered to be 2 totally different controls. Only when the name of the control, the assembly name, and the versions listed in the license file match, will the control be properly licensed. Once the license file gets out of sync, there are several possible solutions, see **Chapter 3 – Manually Adding the License Keys**.

Case 2:

The second cause is a classic case of the left-hand not knowing what the right-hand is doing. Someone at Microsoft decided that all projects types would get a license file when dropping a licensed control on a form, user-control, and etc. Sounds like a good idea, but it has one little problem, only a project which generates an EXE is going to utilize the "**licenses.licx**" file contain within it. All other project types ignore their own "**licenses.licx**" file. Remember, only the main EXE project is going to maintain the license keys.

Sub-projects are not allowed to store license keys because, if they did, you could create a custom-control or user-control, distribute it to other developers around the world, and they would not need to pay for their own developer license. So Microsoft's decision to not build license keys into sub-projects is correct. Where they went wrong was to include a "**licenses.licx**" file in the sub-project. This leads you to incorrectly assume that the license keys are going to be compiled into the sub-project assembly (DLL).

There is nothing wrong with adding licensed controls to sub-projects, but you need to understand the licensing model and its quirks to do it successfully. If you drop a type of control into a sub-project and your main project, the main project will store the license key for the specific control type and your application will execute correctly on the deployment machine. If you drop a type of control into a sub-project and never add it to your main project, the main project will not have a license key for the specific control type, and your application will not execute correctly on the deployment machine. For the later case, see **Chapter 3 – Manually Adding the License Keys**.

Case 3:

The third case involves upgrading your project to a newer version of Visual Studio. Let’s say you start your project using VS2003, and then you upgrade it to VS2005. This upgrade can cause the “**licenses.licx**” file to become corrupted. Even if you open the file and try to fix it manually, this will not help. You need to delete the license file and regenerate it from scratch (See **Chapter 4 – Regenerating the “licenses.licx” File** for help.).

Chapter 3 – Manually Adding the License Keys

You will need to manually add license keys when the licensed controls you are using are not present in the main EXE project. This scenario occurs when you have sub-projects that have forms or user-controls that contain licensed controls. Because sub-projects do not store the licensing information in their compiled binary, you will need to manually force the addition of the licensing information into the main EXE project license file.

Sub-projects will have a "**licenses.licx**" file, but the contents are never used. Only the main EXE project will compile the contents of its own "**licenses.licx**" file into the EXE. Only the license keys store in the main EXE are used for licensing the controls in the main and sub-projects during execution of your program on the deployment machine.

Here is the procedure for manually forcing the addition of license keys to the main EXE project license file…

- Open any form which belongs to your main EXE project.
- Drop the licensed control(s) on the form.
- Save the form (Only required for VS2003. For VS2005 and above, the license file is updated as soon as the control is added to the form.).
- Delete the newly added control(s) from the form.
- Save the form.
- You can now build and distribute your application.

Chapter 4 – Regenerating the “licenses.licx” File

When it comes to .NET licensing problems, in most cases, the easiest solution is to just regenerate the licensing file from scratch. The cause could be corruption of the “**licenses.licx**” file, an entry in the file has an incorrect assembly version number, or an entry for a particular control is missing. Below are 2 different procedures for regenerating the “**licenses.licx**” file, one for VS2003 and the other is for VS2005 & above

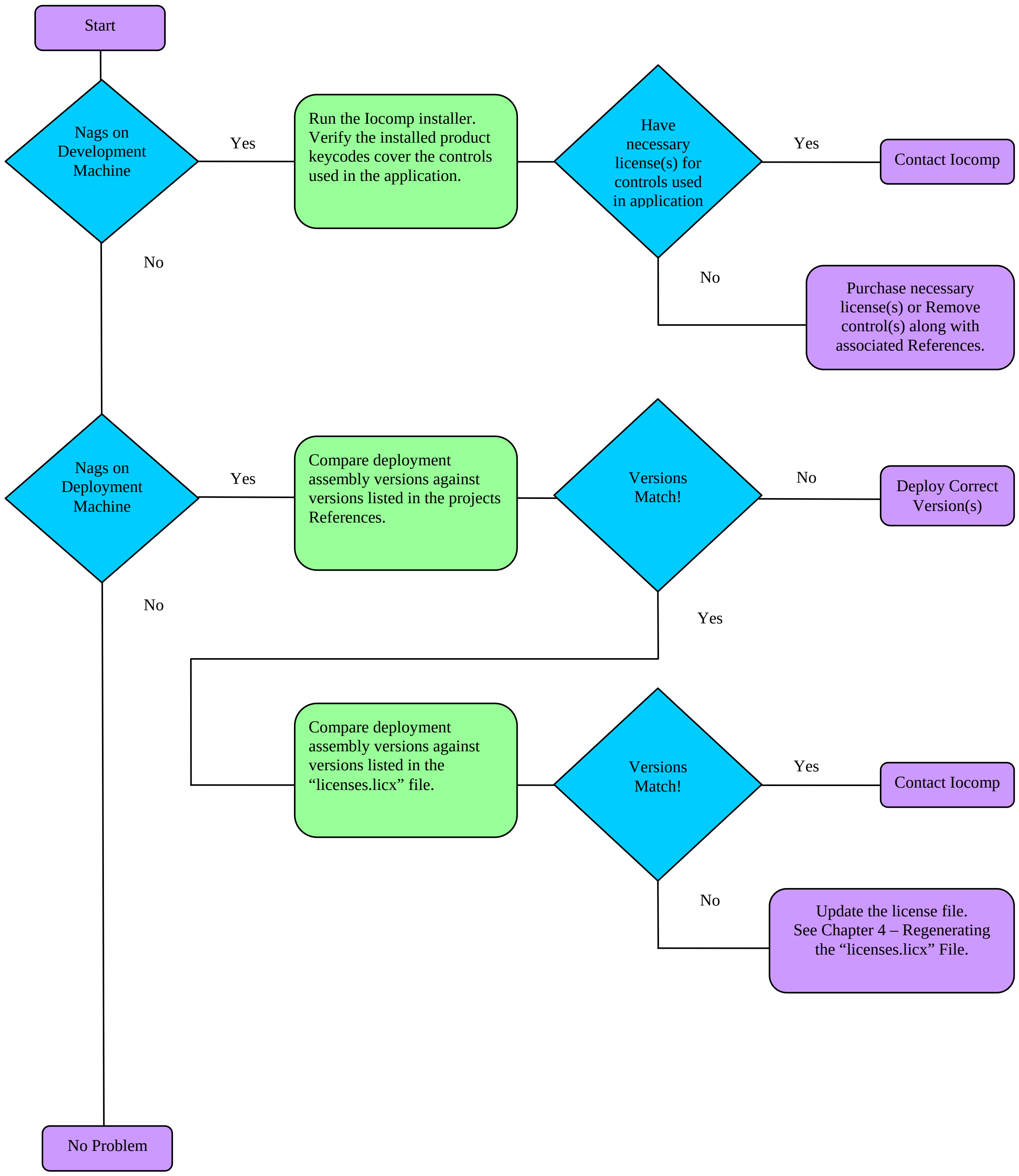
License File Regeneration for VS2003:

- Open your main EXE project.
- Find the file “**licenses.licx**”, and delete it. See **Figure 2.4** if you need help in determining the location of the “**licenses.licx**” file.
- Open enough forms such that one of each type of control is present on the forms.
- Make the forms dirty by moving a control around or toggling a property back and forth.
- Save the forms. This will force the regeneration of the “**licenses.licx**” file.
- If you need license keys for controls that only appear in sub-projects, drop these licensed controls onto a form in your main EXE project, save the form, delete these newly added controls, and save the form again.
- You can now build and distribute your application.

License File Regeneration for VS2005 & Above:

- Open your main EXE project.
- Find the file “**licenses.licx**”, and delete it. See **Figure 2.4** if you need help in determining the location of the “**licenses.licx**” file.
- Open enough forms to cover all of the various types of licensed controls in your application. Opening the first form with licensed controls will force the recreation of the “**licenses.licx**” file. For each form you open, it will only add entries for the controls types present in that particular form.
- If you need license keys for controls that only appear in sub-projects, drop these licensed controls onto a form in your main EXE project. Then delete these newly added controls, and save the form again.
- You can now build and distribute your application.

Chapter 5 – Troubleshooting Licensing Problems


Start
Nags on Development Machine
Yes
No
Run the Iocomp installer. Verify the installed product keycodes cover the controls used in the application.
Have necessary license(s) for controls used in application
Yes
Contact Iocomp
No
Purchase necessary license(s) or Remove control(s) along with associated References.
Nags on Deployment Machine
Yes
No
Compare deployment assembly versions against versions listed in the projects References.
Versions Match!
No
Deploy Correct Version(s)
Yes
Compare deployment assembly versions against versions listed in the “licenses.licx” file.
Versions Match!
Yes
Contact Iocomp
No
Update the license file. See Chapter 4 – Regenerating the “licenses.licx” File.
No Problem