IVI Web Forum Question and Answer

Answered by Kirk Fertitta, Pacific Mindworks
for the IVI Web Forum

Note: Offering an IVI driver is a requirement for all LXI-conformant instruments. The IVI Web Forum is hosted by the IVI Foundation. You can sign up at www.ivifoundation.org.

Question:

How to find and select installed instruments.
How, using the IVI interfaces do I find and create a list at runtime of the installed instrument drivers for a particular instrument class. For instance, I normally create my own generic drivers for my program and then I present to the user a list of all the instruments supported for each class and he selects his particular model for use in that test. There are different classes for Scopes, DMMs, Anlysers etc. How do I find out at runtime what IVI drivers are loaded for what models of what instruments in each class and list then for selection to the end user. I would then want to tell the class driver which model driver to use and then communicate with it through the high level class driver. Can I do this with IVI or do I need to continue with my own driver system?

Answer:

Thanks for your question. This kind of thing is actually one of the benefits of using IVI -- that is, there is a single place you can look for this information and dynamically discover all sorts of things about the installed drivers. Each driver is required by the specifications to register the information you mention (and a lot more) in the IVI Configuration Store (a.k.a., "Config Store"). When you install the IVI Shared Components, these include APIs for accessing the Config Store at runtime. There is a native COM API, a .NET API, and an ANSI-C API -- all are functionally equivalent but are intended to provide convenience based upon your desired development environment.

Here is a very simple code sample for discovering the information you mention. As you can see, it's only a few lines of code. This sample is using C#. Just create a C# console application, and then you must add a reference to the Config Store -- within Visual Studio, choose "Add Reference", select the "COM" tab within the Add Reference Dialog, and then select "Ivi Configuration Server 1.6 Type Library". With that in place, the following code should work:

```
var store = newIviConfigStore();

var location = store.MasterLocation;

store.Deserialize(location);

foreach (IIviSoftwareModule sm in store.SoftwareModules)

{

var classes = String.Join(",", sm.PublishedAPIs.Cast<IIviPublishedAPI>()
.Select(a => a.Name)
.ToList());
Console.WriteLine("Driver: Name = {0}, Models = {1}, Classs = {2}",
sm.Name, sm.SupportedInstrumentModels, classes);

}
```