Deploying a C# .NET Object onto a Laptop to be used with a Blaise 4.7 Data Collection Instrument

MécèneDésormice, Daniel Moshinsky, Thomas Melaney, U.S. Census Bureau

1. Background

The U.S. Census Bureau is currently designing and developing an improved annual survey, the re-engineered Survey of Income and Program Participation (re-engineered SIPP). The survey is intended to provide statistical information on household income and participation in income assistance programs and serve as an improvement to the Survey of Income and Program Participation (SIPP). In order to facilitate accurate recall for re-engineered SIPP, chronological data for the survey will be collected with the help of an Event History Calendar (EHC) which, as the name implies, will ask respondents to record key events into a calendar. The functionality of the EHC is being designed and developed in a C# .Net environment producing a DLL which will be called from the Blaise DEP.

This document will identify the steps necessary to deploy a Blaise survey with a .NET DLL component onto a system that lacks the .Net component framework. An alternative successful strategy will also be discussed. Implications for deploying this kind of instrument onto hundreds of laptops configured for use by field interviewers will be discussed as well.

The initial development of the C# DLL was done at the U.S. Census Bureau using a standard Personal Computer (PC) that was supported within a network environment. The initial programming, testing and revisions as well as the presentations for review and design consultation with the program specification specialists all took place at a programmer's PC station.

Within a few weeks the development had reached a stage when it was appropriate to move the application to a laptop computer. The actual data collection operation for the survey is to be a Computer-Assisted Personal Interview (CAPI) operation. It was important to verify early within the development that the screen design was appropriate and that the program would operate well on the type of laptop to be used by the field interviewers, referred to as field representatives (FRs). It was also important to have the instrument operating on a laptop as a convenient way to demonstrate the new instrument to interested stakeholders in different locations. The laptop configuration currently used by the U.S. Census Bureau for the field data collection operation is the Dell Latitude D400 with a Windows 2000 operating system.

After initially porting the EHC DLL to the laptop, the programmers discovered that the program could not be accessed. It was necessary to engage in a variety of modifications to resolve the porting issues one step at a time. Searching the internet for articles with information about the proper methodology for the deployment was helpful but frustrating in that there were few articles that examined the entire process of transferring a C# DLL to a station that did not include the MS Visual Studio software. Many times, suggested approaches were inappropriate for the situation at the U.S. Census Bureau. The laptops to be used for data collection have very restricted access by the end-users (field representatives) and the security on those laptops must be extraordinarily tight. The

questions in many of the surveys at the U.S. Census Bureau ask respondents for information that could, at the questionnaire level, identify individuals and provide information about their families, households and economic status that would be considered highly sensitive. In accordance with the U.S. Census Bureau's privacy principles as stated under Title 13 of the U.S. Code, the U.S. Census Bureau takes measures to assure that privacy is strictly preserved and that any data or tabulations that are made available to clients or the public are free from any personally identifiable information.

Within this document, the project is called DEWSEHC. This name is used within the coding and reflects an earlier reference name used for the re-engineered SIPP Event History Calendar.

Please note that this report is meant to inform interested parties of ongoing research and to encourage discussion of these technical issues. The views expressed in this document are those of the authors and not necessarily those of the U.S. Census Bureau.

2. Purpose

The Event History Calendar (EHC) is a Dynamic Link Library (DLL) that is written in Visual Studio 2005. It is intended to be loaded on the FR's laptop and launched from the Blaise Instrument through an Alien Router call.

The purpose of this document is:

- Show how to add and use the Blaise API Components package in C#.
- Show how to deploy and register a C# DLL or component on the FR's laptop or on a PC other than the development machine (PC)
- Identify the files that need to be sent to laptop or PC along with DLL.

3. Adding the Blaise Components Package (BCP) to a C# project

To add the BCP, you need to first create a new Visual Studio project using the Windows Control library. In this case, we are going to use the DEWSEHC project as the example for this exercise. Below is a picture of the Solution of the project (see Fig. 1).

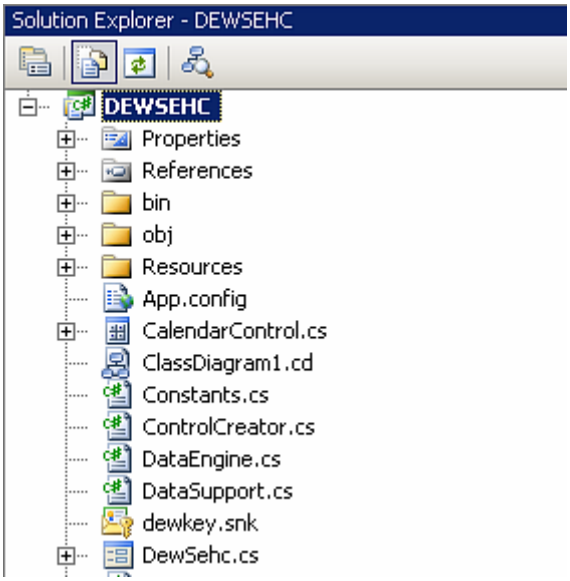

Figure 1.

Next a reference to the Blaise API component must be added. This can be done by selecting Add Reference from the project menu or by right-clicking the References folder in Solution Explorer (see Fig. 2).

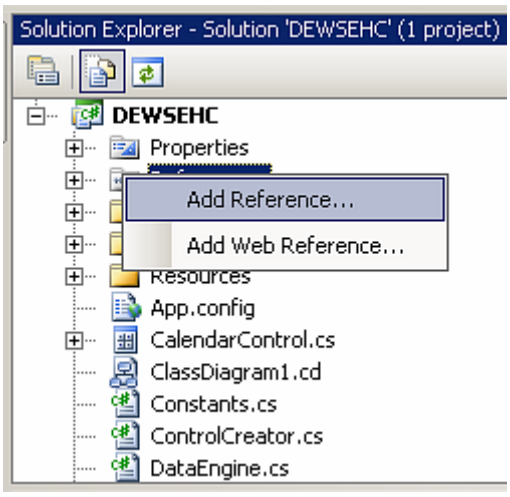

Figure 2.

Select the COM tab from the Add Reference Window, and then select Blaise 4.7 API Component (see Fig. 3).

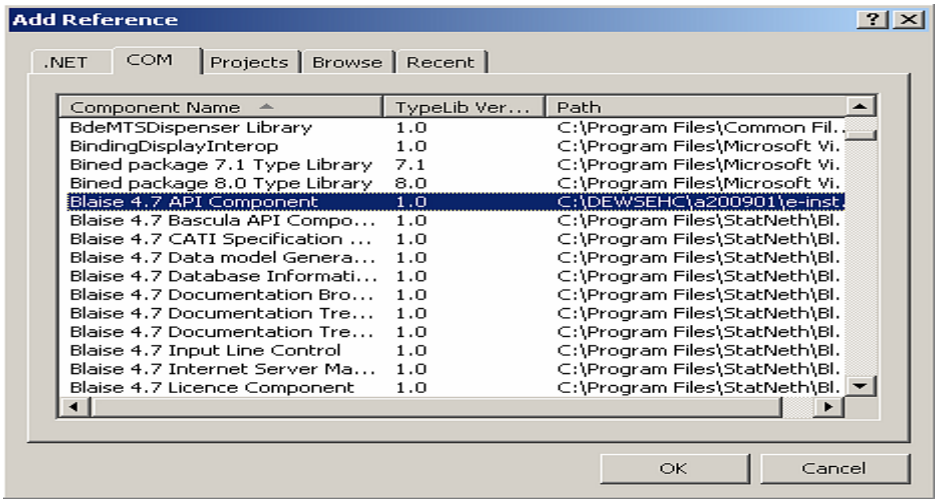

Figure 3.

C# will automatically add BlAPI4A2.dll to the project and create two new files: Interop.BlAPI4A2.dll and Interop.SiCstrCIA.dll (see Figure 4).

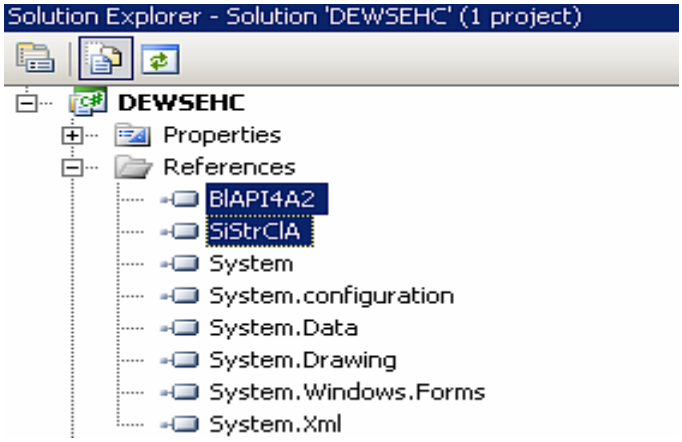

Figure 4.

These files are placed in several locations under the project folder. For example, these files may be found within the Debug folder.

Then, the reference to BlAPI4A2 must be included in the source code of the project's UserControl class. The keyword "using" is used in C# to include references into the application. From within the Blaise DEP, the Alien Router call will invoke the GetMeAField method, which, in turn, calls the EHC form – as shown below.

```
using System;
using System.Runtime.InteropServices;
using BlAPI4A2;
namespace DEWSEHC
{
  public partial class CalendarControl : UserControl
   {
      private BlAPI4A2.Database db;
      private BlAPI4A2.DepState ds;

      public CalendarControl(){InitializeComponent();
      private void CalendarControl_Load(object sender, EventArgs e){}
      public void GetMeAField(BlAPI4A2.Database DB,
                        BlAPI4A2.DepState     DS)
      {
         DEWSEHC.DewSehc mCalendar = new DEWSEHC.DewSehc(DB, DS);
         mCalendar.ShowDialog();
       }
   }
}
```

4. Preparing for deployment

The deployment of DEWSEHC to the laptop requires the following additional steps:

1. Verify that Windows Service pack version 3 or higher is installed on the laptop.
2. The .NET framework distributed package (dotnetfx.exe) must be loaded on the laptop and installed. The dotnetfx.exe is located under

   C:\Program Files\Microsoft Visual Studio 8\SDK\v2.0\BootStrapper\Packages.
3. Create the .NET object Strong Name. This can be done by right-clicking the project folder in the project Solution Explorer and selecting Properties. Then, select the Signing tab, check “Sign the assembly”, and select “New” from the Dropdown box (see figures 5 and 6).

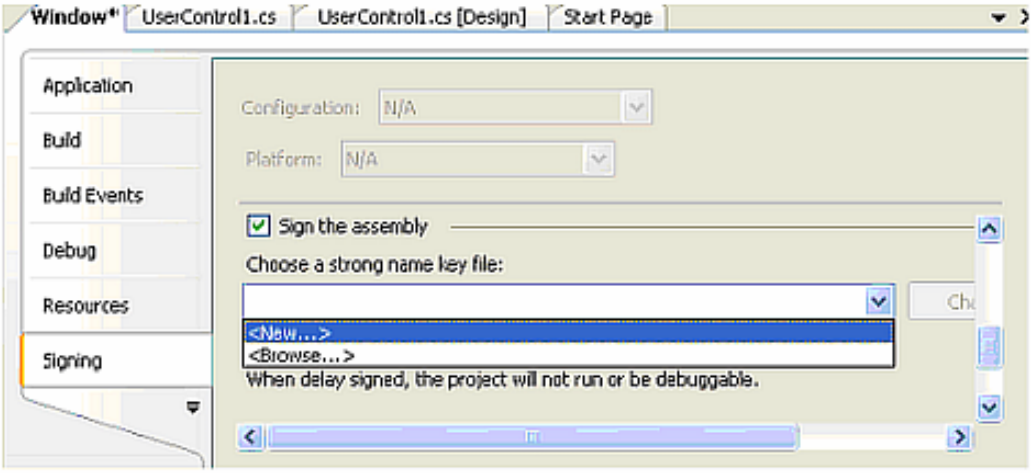

Figure 5.

Figure 6.

- Set ComVisible to true. To set ComVisible to true:
    1. Right-Click on the project name, then select Properties
    2. On the Properties Window, Click on Build and Check Register for COM interop.
    3. The dropdown box, select "**On"**

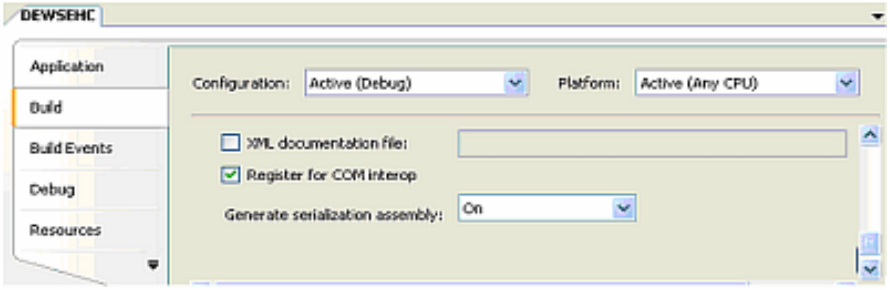

Figure 7.

- Rebuild the application.

5. Deploying the DLL

Now that the DLL has been prepared, we are ready to deploy. The following files must be copied into the project directory on the laptop:

- BlAPI4A2.dll – This is the run-time file that will interact with the Blaise DEP
- Interop.BlAPI4A2.dll – This file provides information needed to call methods exported from the Blaise API shared library
- Interop.SiStrClA.dll – This file also provides information needed to call methods exported from the Blaise API shared library
- DEWSEHC.dll – This is the application DLL
- DEWSEHC.tlb – This is the application type library automatically generated at build-time

Register the application DLL using the following command.

C:\WINNT\Microsoft.NET\Framework\v2.0.50727\regAsm.exe /codebase /tlb=DEWSEHC.tlb DEWSEHC.dll

Register the Blaise API DLL as follows:

Regsvr32 BlAPI4a2.dll

A batch file that includes these two commands can be created.

Now, the Blaise instrument should successfully invoke the DLL on the laptop.

6. Alternate Suggested Strategy

In communications with colleagues at Westat and Statistics Netherlands, an alternative strategy was suggested for resolving the configuration problem.

The suggestion was to create a setup program to deploy the .NET DLL by using the following steps.

1. Give the .NET DLL a Strong Name
2. Create a “setup and deployment project” within Visual Studio
3. Run the setup on the target machine and install the files in an appropriate directory.

This approach was not appropriate for our situation. The “setup” program requires that the user input values to reply to questions. Our current process of distributing our applications to thousands of laptops in the field is fully automated. It did not seem to be practical to alter our current production routines to run the setup program since we could avoid the issue by using the strategy described in this paper instead.

7. Questions for Further Investigation

The project discussed in this paper is still in a relatively early stage of development. We have yet to attempt to deploy the data collection instrument to field laptops on a large scale. We will be working to determine what, if any, adjustments may need to be made to our data collection instrument or to our survey distribution system in order to assure a smooth interface.