Xcel Energy, USA

Xcel Energy selects VB Migration Partner after comparing it with its competitors

Xcel Energy evaluated software tools to migrate Visual Basic 6 application to Visual Studio 2008. The business requirements were to migrate approximately 120,000 lines of code – of programs ranging from 2K to 75K lines of code each. The analysis identified three particular categories of programs:

- Form based application – used for editing configuration data.
- Form based ActiveX – used for displaying runtime data (used in a non Visual Studio based application).
- Extract-Tranformation-Load (ETL) programs used for importing data into a system.

The end-state of the conversion process had to be Visual Studio 2008 applications (preferable in C#) based on the CSLA.NET framework developed by Rockford Lhotka, as described on www.lhotka.net, having a presentation layer, a business object layer, and a data access layer. Where possible, the goal was to convert 3rd party components to standard Visual Studio 2008 components to minimize software maintenance costs.

In addition to Visual Studio 6 to Visual Studio 2008, Visual Studio 2003 VB.NET to Visual Studio 2008/C# conversion tools were also examined. Based on this parallel analysis; the company concluded that once VB.NET has been refactored correctly, the conversion from VB.NET to C# is straightforward and can be done quickly at a very low cost. From an evaluation perspective, means that the goal of the conversion effort can be relaxed to VB.NET.

At the outset of the evaluation process, a short-list was developed using various search engines:

1. Built-in migration wizard in Visual Studio 2008.
2. A conversion program that converts VB6 into VB.NET or C#. (Upgrade from Option 1).
3. A T4-based "add-in" to Visual Studio – this package imports VB6 – and then allows you to create your own mappings into either C# or VB.NET.
4. Code Architects' VB Migration Partner that focuses on converting VB6 to VB.NET.

Option 1 was able to convert the very simplest ETL program. After converting the code into VB.NET, SQL code was refactored into Stored Procedures. Using LinqToSql and the CLSA.NET framework, 1879 lines of code were reduced to about 95 lines of code. This was a promising first step.

In the larger more complex programs; option 1 and option 2 had difficult in converting some of the larger ActiveX and Windows Forms projects. After the conversion, many of the forms could not be opened in the Visual Studio 2008 designer. This required editing the partial class to eliminate or

correct references to obsolete controls. After conversion there were errors ranging from 10 to 200 errors per program. The results were promising but indicated more manual work involved.

Option 3 was used to convert a different ETL program. This option generated some sort of CsharpAsic.NET code! "CsharpAsic.NET" is where C# and VB.NET are combined in same module. It took over 2 days to fix the resulting code that was produced from one single program, as the Visual Studio 2008 compiler really does not like VB.NET and C# to be placed in the same module. So there was a fair amount of manual code conversion required. As before, SQL was removed from the code and placed into stored procedures – then made accessible to the code via LinqToSql. Code was re-factored and common repeating code patterns converted into standard .NET library calls. This largely manual effort produced a 15% reduction in the lines of code. There was the added benefit of fixing an error in the system that had been place for the last 5 years. Option 3 looks very promising but the vendor needs to spend more time in developing a comprehensive VB6 parser.

In contrast to Option 3, the VB Migration Partner option was able to correctly parse and convert all of the Forms and ActiveX controls to VB.NET. After analyzing the output and reviewing their on-line knowledge base (which is quite extensive), two pragmas were inserted to reduce the number of conversion errors to a handful in most of the programs. All of the forms could be edited without any further intervention. Grid controls that could not be converted were initially left as red rectangles. Within 2-days, all of the source code was cleaned up with no compiler errors. This meant that the existing presentation layer could be converted from VS6 to VS2008 – so the VB Migration Partner option works very well as part of an overall approach.

One of the things that a conversion does not do is eliminate SQL from the underlying code. (In our architecture, SQL code should always be in stored procedures). It is also unreasonable to expect a converter to separate out business logic and data access logic and presentation layer logic, thus to meet the final requirements of the end-state, the following tools were identified to complement the tools from VBMigration.com:

- CodeSmithTools – This tool can generate the Data Access Layer and Business Object layer given a database schema
- Instant C# -- This tool can be used to convert VB.net over to C#

Quote from the customer

Give my experiences, I would recommend that companies take a serious look at VB Migration Partner.
William Hullsiek
MS Software Engineer – University of St. Thomas
MCPD/MCITP-BI