srcML Tool Implementation on a C# Environment

Jesus Eduardo Jaime Gandarilla
Department of Computer Science
The University of Texas at Dallas
jxj132730@utdallas.edu

Introduction

This project's main goal is to detect Common Coupling in a C# language code base using the set of tools provided by the srcML Framework.

Coupling is defined as a measure of interdependence among modules in a computer program [1]. In our case we chose, Common Coupling, also know as Global Coupling which occurs when two modules share the same global data or variable.

On the other hand, srcML is an infrastructure for the exploration, analysis, and manipulation of source code [2] which translates code from a project in Java, C++, C or C# into a single XML format document. All original text is preserved so that the original source code documents can be recreated from the XML file [3] and even keeping the directory structure of the project. The latest version of this toolkit was posted in May, 2015.

Implementation

As a first step, we searched the different options available to carry out code analysis for a project in C# and more precisely, with the intention to use these tools inside a Visual Studio Project to create an automated tool for code analysis.

At the end of the day, we found that there are three viable options to implement srcML in a C# code base.

Option 1 - Command Line Tool

This is the tool available to download in the project's website: srcml.org. The installation and use is very simple:

1. Go to www.srcml.org
2. Click on Downloads
3. Select the version for your OS
4. Install the executable.
5. Open a command line.
6. Execute srcML command as shown in figure 1.

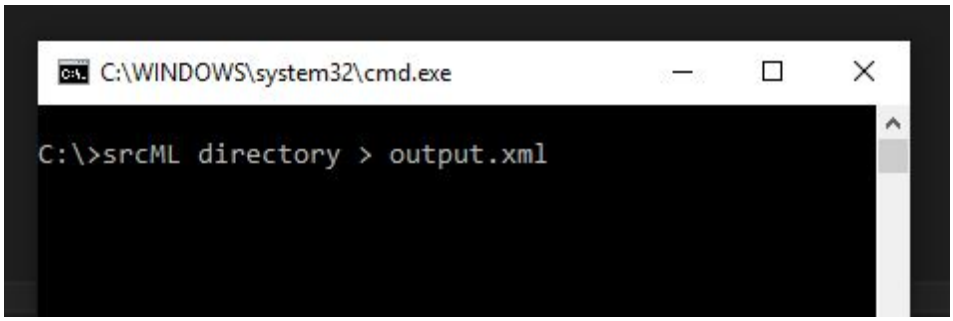

Figure 1. Converting a project into a single xml file in srcML Format.

This toolkit also offers the following commands:

1. src2srcml: Converts a directory or a file in any of the supported programming languages into a XML File in srcML Format.
2. srcml2src: Converts a XML File in srcML Format into a file or project files in the original programming language and preserving directory structure.
3. xpath: By passing a XPATH query we can get nodes in the XML File.

In our case, we will not implement this option since we are interested in creating a tool in Visual Studio to automate this process.

Option 2 - As a Visual Studio Service

srcML Service can be installed in Visual Studio by click on tools, selecting extensions and updates and then looking for it in the online tab.

This option provides analysis services for other visual studio extensions to consume. It is useful if you want to develop Visual Studio plugins, such as Sando Search tool which is an improved text search tool, also sponsored by ABB Research Center and uses srcML Service to get notified of file changes. [4]

This option is not useful for our project's needs. However, it is worth mentioning so that the audience know that there are some standalone tools that implement srcML.

Option 3 - As a package in a Visual Studio Project.

ABB Research Center provides a framework that they have been using within themselves to do both program transformation and code analysis.

This framework is named srcML.NET, and can be downloaded using GitHub or directly 'installed' into a Visual Studio Project using the Package Management Console.

To do so, one must follow the following steps:

1. Open visual studio and create a new project.
2. Click on View > Other windows > Package Management Console

3. Enter the following command:

```
PM> Install-Package ABB.SrcML
```

4. Now you can import ABB.SrcML and code a small tool. Figure 2 shows an example of how to create an instance of the ABB.SrcML class that converts a project into a srcML XML File.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace srcML_Test
{
    class srcML_TestApp
    {
        public void main() {
            ABB.SrcML.SrcML srcml = new ABB.SrcML.SrcML();

            srcml.GenerateSrcMLFromDirectory("c:\\path\\codefolder", "results.xml");

        }
    }
}
```

Figure 2. Creating a class that converts a project into an XML file.

In this project, we will choose this options as it is the one that fits best our goals:

1. Create a small tool that converts a project in C# into a XML file in srcML Format.
2. Provide code analysis capabilities with the same tool in order to detect Common Coupling in the project.

More on ABB srcML.NET

srcML.Net is a framework used within ABB Corporate Research and it's based on the srcML project from Kent State University Software Development Laboratory.

It is used to do both program transformation and code analysis, and includes the same tools and functionality offered by srcML but allows integration to Visual Studio Projects to develop code analysis tools. [5]

Projects to be analysed in this project

We have completed the first step in this project which is converting a code base in C# into a single XML file in srcML Format.

As you can see in table 1, the conversion from C# to srcML inputs a XML file that is between 2.5 and 2.8 times larger than the code base. In our biggest size project, Mono, the output XML file is 455 MB in size which is still considered, in our opinion, a processable size and does not require splitting the file into more files.

The Mono project took around a minute to be converted into XML and all the others were converted almost instantly by the tool.

Project	Number of .cs files	Total Size (MB)	Total Size of XML
CodeHub	344	1.64 MB	4.55 MB
MonoGame	1,229	9.24 MB	24 MB
Newtonsoft.Json	2,014	2.46 MB	6.57 MB
Mono (<1min)	24,819	161 MB	455 MB

Table 1: Code base conversion results

It is also worth mentioning that, for our analysis and tool creation, we will be using a computer with the following characteristics:

- 6th Generation Intel Quad Core i7 Processor
- 16 GB DDR4 of RAM memory.

Next steps

The next step will be querying the XML File in order to find which global variables are used in which methods, identifying the line number and file names of the source files that contain said method.

After completing our analysis, the tool will present a results report in a format that is yet to be defined.

References

[1] IEEE, Guide to the Software Engineering Body of Knowledge (SWEBOK v3), 2014.
[2] srcML.org Home Page. Retrieved from: http://www.srcml.org/
[3] Collard, M. L. and Maletic, J. I., "Lightweight Transformation and Fact Extraction with the srcML Toolkit.", 2011.
[4] Visual Studio Gallery. Retrieved on April 14th, 2016 from: https://visualstudiogallery.msdn.microsoft.com/ca1c4937-170f-472b-b251-11a9f4d2a161
[5] srcML GitHub repository. Retrieved from: https://github.com/abb-iss/SrcML.NET