# Proposal for a

# C# to Java Translator

# Thesis Project

Harvard University

Extension School

September 7, 2005

Scott Henderson

shenders@fas.harvard.edu

# 1 Tentative Thesis Title:

A Translator to Convert Source Code from C# into Java

# 2 Abstract

Microsoft's C# is a programming language that is similar to Sun Microsystems' Java. Microsoft provides a tool for free that helps people migrate from Java to C#. The purpose of this thesis project is to create a tool to help translate C# into Java.

The technologies that are relevant are Sun Java 2 JDK, Microsoft .NET SDK and framework, antlr parser generator, and standard compiler techniques.

My approach will be to parse the C# file, translate the representation from C# to Java syntax, provide type translation where feasible, provide some semantic translation, and write the Java source as output.

# 3 Thesis Project Description

Some of the Harvard Extension School professors have discussed that there are similarities between C# and Java. It has been my experience as well that this is true. During Introduction to .NET, we also discussed that Microsoft offers a translator from J# (Microsoft Java for .NET) to C#. The intent of my thesis project is to provide an application to translate from C# to Java.

In the Extension School, I took Introduction to .NET in which we learned about the .NET architecture by writing C# applications. I also developed applications in C# at work. With respect to Java, I have written a significant amount of Java in the Extension School, and my current job is performing maintenance on my company's software, a large part of which is Java.

Regarding my ability to handle problems related to the nature of programming languages, through the Harvard Extension School, I took Theory of Computation, which is a class that requires understanding of computer languages at the theoretical level. I also have an undergraduate degree in computer science from Florida State University. For that degree, I was required to take a Programming Languages class.

The application that forms the core of my demonstration will be named cs2java. It will be a C# application that will translate source files from C# to Java. It will be necessary to discuss with my thesis advisor what the scope of the project should be, since there is significant room for development on this topic. I would like to use modern existing parser tools, such as antlr and any available grammars, in order to finish with a more sophisticated source migration tool. This would allow me to concentrate on making a more complete sophisticated solution within the time frame of the thesis project.

cs2java will take the following approach during execution:

1) The C# compiler will syntax check the input C# file to make sure that it is well formed.

2) cs2java will read in a C# source file.

3) The application will recognize the code. The purpose of this step is to convert the C# source into a more usable format. The file has not been determined to make sense, but the words of the file will have been split into discrete elements that cs2java can recognize. The code for this step will be assisted by classes generated by antlr.

4) cs2java will interpret the language. The purpose of this step is for cs2java to understand the words in the C# file, to determine if it makes sense. (Since the source file has been checked with the Microsoft compiler, steps 3 and 4 are not intended to act as a general purpose C# syntax checker. They are written to be as safe as possible while expecting a syntactically correct file.) The code in this step will be assisted by classes generated by antlr.

5) The application will represent the well-formed syntax as an object tree.

6) cs2java will pass over the tree and convert any C# syntax specific parts to ones that may be represented by C# or Java.

7) During the translation passes over the tree, the application may log warnings or errors if parts of the code are known to be not translated properly.

8) The tree will now contain only syntax constructs that may be represented by either C# or Java.

9) The application will perform a second pass over the tree, during which each syntax node will be converted from a C# syntax node to a Java syntax node.

10) A third pass over the tree of Java nodes will resolve any dangling issues and check to see if any Java needs to be added, based on the contents of the tree. This step would represent a semantic translation. For example, C# has a member function named ToString but the Java variant would be named toString. The semantic translation step would identify dangling issues and adjust the representation accordingly.

11) At this point, there will now be a Java version of the source tree. Presumably, such a translation could be done for each Java file in a project. However, the results are not likely to be satisfactory for a significant project. Any imported packages will not have been translated. This should be handled one of three ways.

    a) The translator may have translated the imported code as well, in which case the package may be available as well.

    b) The translator should have a list of packages that it should know to map. For example, calls to objects in the Console package in C# should be replaced to calls to the System package in Java. This part of the architecture should be as easy to update as possible. For real use, extending this package mapping is a place where significant further development would occur.

c) Third, if no mapping is found, the translator should provide output error logging. To the extent possible, the translator may add Java-side runtime error logging as well, in addition to stubs of missing types.

12) The tree will be Java-only at this point and well formed. cs2java will write the Java tree to file, converting it to Java source code in the process.

# 4 Work Plan

## *4.1 Assumptions, Risks and Alternatives*

For this project, I will use the following tools on Windows XP:

- Microsoft Visual Studio .NET: The source code for the project will be written primarily in C#, using the Visual Studio environment. My intent is to use Visual Studio as an IDE for development.

- Microsoft .NET SDK: Microsoft's .NET SDK includes a command line compiler for C# for Windows.

- Eclipse: Eclipse is an open-source IDE with a native ability to understand Java. For any Java development, I will use Eclipse.

- Sun Microsystems' JDK: Sun's JDK is available as a free download for all platforms.

- ant: For command line development with Java, ant is a standard build tool it is freely available from apache.org. My intent is to use ant with the Sun JDK to build any Java code.

- nant: To support command line development with the .NET SDK, nant is available. It is a port of ant to .NET. It is freely available. My intent is to use nant to run the C# compiler from the .NET SDK.

- antlr: antlr is a package that generates language recognizers and syntax parsers from grammars. It is a modern tool that is similar in purpose to yacc. antlr is compatible with C#, Java, and Python, and is available for free.

- Microsoft Word: All documentation will be written using Microsoft Word.

- Quattro Pro: For requirements, task, and bug tracking,. Quattro Pro has the ability to export to other spreadsheet formats that could be useful if communicating with others.

- cvsnt: I will use cvsnt on Windows XP for revision management. All reasonably mature source code for the project will be under revision control, as will any documentation.

This project is not without potential risks or limits.

- A tool to migrate from C# to Java would be a significant project for a business to undertake. The scope of the project would be quite large if it were done for a general case. The project will be limited to providing a source translator, some package translation, and the ability to add other maps.

- At this point, I have not done a full analysis of the C# language. It is possible that parts of the language will not translate well or will be too complicated to do within the scope of this project.

- Most packages will not be translated, due to the scope of the project. Such extensions would be perfect enhancements if cs2java were turned into an open source project following completion of the thesis project.

## *4.2 Preliminary Schedule*

This is a breakdown of activities that could act as major milestones for my work.

| **Task** | **Time Frame** |
|---|---|
| Install any support software and establish build framework. | 1 week |
| Translate the C# grammar into an antlr representation. Alternatively, if a C# grammar exists, it may be possible to take advantage of this work. | 2 weeks |
| Use antlr to construct a working C# recognizer. | 2 weeks |
| Construct C#/Java syntax tree. | 4 weeks |
| Translate nodes from C# syntax to Java representation. | 4 weeks |
| Perform semantic translation from C# to Java. | 2 weeks |
| Handle packages as necessary. | 4 weeks |
| Write Java source from the Java syntax tree. | 2 weeks |
| Test C# to Java translation and fix outstanding bugs. | 4 weeks |

# 5 Glossary

This section should include definitions of major terms and an explanation of acronyms.

- ant: A command line build tool for Java.
- antlr: A parser-generator tool that creates Java, C# and Python code based on an input grammar.
- C#: An interpreted language created by Microsoft for the .NET architecture.
- grammar: A description of a language. antlr takes a grammar as written in a particular format as input.
- IDE: Integrated Development Environment. This is usually an editor and debugger for a language. For example, Java has Eclipse, and C# has Microsoft Developers Studio .NET. These are applications that are intended to make writing software in those languages easier.
- Java: A popular interpreted language from Sun Microsystems. It is compiled into bytecode that runs on the Sun virtual machine.
- nant: A port of ant for the .NET framework.
- .NET: A Microsoft architecture for application development, normally on Windows.
- revision management (source control): A process or an application that manages changes made to a system during software development.
- translator: An application to translate a source file in one language into another.

# 6 References

To support my research, I expect to consult the following books:


1. The C# Programming Language, by Hejlsberg, Wiltamuth, and Golde, 2004.
2. The Java Language Specification, Third Edition, by Gosling, Joy, Steele, and Bracha, 2005.
3. Building Parsers with Java, by Metsker, 2001.
4. Modern Compiler Implementation in Java, Second Edition, by Appel, 2002.