



Open Source at Microsoft

Shared Source CLI: Building a Community Around .NET

When the Microsoft® .NET Framework was first released in 2000, computer science programs at leading universities had already begun actively researching virtual machines and the technologies surrounding them. To explore these new software solutions, however, professors and students needed source code access. Unfortunately, academics had few choices available to them.

At the time, Sun Microsystems provided source code for the Java Development Kit, which included the Java Runtime Environment. However, this runtime was intended for use by only the Java programming language. Further, Java was only one of many virtual machine systems on the market.

With the new .NET Framework, which included a new virtual machine runtime, Microsoft had developed the common language runtime (CLR), an implementation of the Common Language Infrastructure (CLI). The CLR was the “engine” at the bottom of the Framework, and was a robust runtime for languages such as C#, BASIC, APL, COBOL, and many others.

Jim Miller, a partner architect on the CLR team at Microsoft, was proud of his team’s work. To drive further innovation and enhance understanding of the technologies, he wanted a larger audience to see how this new system worked.

Reaching out to academics and enthusiasts

Before coming to Microsoft, Miller had been immersed in the academic world and spent 15 years at MIT and Brandeis University, researching computer technology and lecturing. He understood firsthand how academic inquiry increased user interest and fueled innovation.

As Miller notes, “We wanted academics to truly understand and use this technology.” But without providing source code for the runtime and the compiler, he knew that the CLR and the Framework would not gain the academic attention it deserved.

Miller and the CLR team also wanted to give commercial .NET enthusiasts a chance to better understand the Framework. Users had questions about its inner workings and were deeply curious about where they could take its potential. The team knew that if enthusiasts could examine the underlying source code, they would be able to share their insights, test the capabilities of the CLR, and build alternative and iterative versions. By providing users a better way to examine the source, the team could help build a community around .NET.

Developing the Shared Source Common Language Infrastructure

From the desire to involve a larger community came the Shared Source Common Language Infrastructure (SSCLI), originally code-named “Rotor.”

In the fall of 2000, the Rotor team first built a subset version of the commercial .NET Framework. Then they added an extensive test suite for the entire implementation: runtime, libraries, and two languages (C# and JScript® .NET). They also developed virtualization layers that enabled developers to build and run the SSCLI on FreeBSD and Mac OS X. By doing so, users could explore how the CLR worked on other operating systems.

The result is an implementation of the Framework with the compiler, runtime, and test capabilities for the ECMA C# and CLI standards.

A ground-breaking license

While developing Rotor was a challenge in itself, settling on a way to distribute it posed a different but equally difficult endeavor. At the time, no major Microsoft product had shipped with a shared source license. Never had the company made available to the public a technology with over a million lines of code—especially not code from one of its most important technology components. Yet that is exactly what the Rotor team aimed to do.

After much discussion and thought, Microsoft wrote a clear, concise—only two paragraphs—license. Users could view the SSCLI source code, modify it, redistribute it, and write about it, but they could not use the code for commercial purposes.

Says Miller, “I was really proud of the license we were able to get. The original license for Rotor is short and easy to understand and became the model for Microsoft Shared Source licenses later. It provides people a great license.”

Indeed, Miller stresses the importance of the license. The question, he notes, was no longer whether we would do open source, but rather, how many projects like this we would continue to do. The license opened the door for projects such as IronPython and IronRuby, both of which provide the source code access to the public.

Promoting research around the SSCLI

In March 2002, Microsoft released the SSCLI as a free download. Peter Drayton, a program

manager on the Rotor project, notes that the release gave academics and enthusiasts alike their first chance to “look under the hood” and see how .NET worked.

Microsoft not only provided the opportunity to view the source code of a CLI implementation, but also awarded research grants to universities under the Rotor Request for Proposals. Microsoft accepted research proposals in two rounds—the first in 2002, the second in 2004—and ultimately provided almost 100 grants totaling over U.S. \$1,000,000.

These grants produced a wealth of research and learning for students, researchers, and Microsoft. In Belgium, professors at the University Katholieke Leuven began using the SSCLI in their secure software development course. During the course, students explored advanced security issues and used .NET and SSCLI to solve security-related problems.

In England, the University of Hull received a grant for its proposal “Toward a Rotor-based Computer Science Curriculum.” Working with Microsoft UK and the Rotor production team, the university developed a master’s degree for .NET studies. The first of its kind, the course relied on the SSCLI as the cornerstone of the program.

Beyond academic research

In addition to advancing university research, the Rotor project helped foster the community of .NET enthusiasts. After the release of SSCLI, developers started newsgroups and blogs on MSDN and independent Web sites, where they asked questions and shared their insights.

In 2003, David Stutz, Ted Neward, and Geoff Shilling collaborated to write *Shared Source CLI Essentials*, published by O'Reilly. Written in part by members of the Rotor team, the book was another resource that enthusiasts and academics could use to dig deeper into .NET technology.

Further, by providing the SSCLI, Microsoft was able to show that the CLR could run on UNIX and Macintosh operating systems. This spurred new implementations of the CLI, most notably in Mono. Mono is an open source project sponsored by Novell, and is an independent implementation of the ECMA CLI that enables Linux developers to build applications on the .NET Framework.

Ultimately, by directly involving the technologists who could provide the richest feedback, Microsoft reached out to its users in a new, more open way. With the Rotor project, Microsoft advanced its commitment to providing developers more technology choices.

Related Resources

- ✓ Rotor wiki page:
http://en.wikipedia.org/wiki/Rotor_%28software_project%29
- ✓ Shared Source Common Language Infrastructure 2.0 Release:
www.microsoft.com/downloads/details.aspx?FamilyId=8C09FD61-3F26-4555-AE17-3121B4F51D4D&displaylang=en
- ✓ Shared Source CLI Provides Source Code for a FreeBSD Implementation of .NET:
<http://msdn.microsoft.com/msdnmag/issues/02/07/SharedSourceCLI/>
- ✓ Shared Source CLI Essentials:
www.oreilly.com/catalog/sscliess/

All other trademarks are property of their respective owners.

Copyright

Information in this document, including URL and other Internet Web site references, is subject to change without notice and is provided for informational purposes only. The entire risk of the use or results from the use of this document remains with the user, and Microsoft Corporation makes no warranties, either express or implied. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted in examples herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred. Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

© 2007 Microsoft Corporation. All rights reserved.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Microsoft and JScript are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.