

# Managing Multi-language and Multi-platform Microsoft Visual Studio® Builds With OpenMake Meister® and CA Software Change Manager

Steve Solomon

CA EITM SERVICE MANAGEMENT

Tracy Ragan

OPENMAKE SOFTWARE



---

## Table of Contents

---

<b>Executive Summary</b>		SECTION 4: CONCLUSIONS	<b>10</b>
SECTION 1: CHALLENGE	<b>2</b>	SECTION 5: REFERENCES	<b>10</b>
<b>The Challenges of Application Modernization with VS 2008</b>		SECTION 6: ABOUT THE AUTHOR	<b>11</b>
The Realities of Heterogeneous Build Environments			
SECTION 2: OPPORTUNITY	<b>3</b>		
<b>Multi-platform Builds Call for Multi-layered Support</b>			
Extending the Functionality of Visual Studio			
Life cycle-based Change Management			
CA SCM and OpenMake Meister Together			
SECTION 3: BENEFITS	<b>9</b>		
<b>Maximizing Builds With an Industrial-strength Engine</b>			

# Executive Summary

## Challenge

---

With each new major release of Microsoft Visual Studio, Microsoft changes the languages and methodologies for building executables from the source created within an application.

Without the assistance of simplification tools, then, development teams integrating Microsoft Visual Studio 2005 and Microsoft Visual Studio 2008 must re-write and redesign significant amounts of their application build processes. But, such an endeavor proves so costly and time-consuming that many enterprises choose to delay or omit upgrades to their development environments. Unfortunately, avoidance will quickly create a situation where new applications are written in the new Visual Studio but without compatibility to the vast amount of programs and systems that were developed in older versions.

## Opportunity

---

To overcome this development limitation, OpenMake, a CA partner that specializes in Build Management, worked closely with Microsoft on the new Visual Studio 2008 and created a process where Microsoft Visual Studio 2005 code can be easily integrated with code created in the newer release. This unique process bridges the technology gap between the two Microsoft development environments and provides a simple method for programmers to integrate all of their existing applications with new applications written in Visual Studio 2008.

And, the technology is so innovative and complete that even developers using MS Visual Studio 97, MS Visual Studio 6.0 and MS Visual Studio .Net 2002 and 2003 can integrate to MS Visual Studio 2008. As such, it allows you to continue to use the large code base already developed in a single integrated environment and with automated and continuous integration build support.

## Benefits

---

OpenMake Meister is a helpful tool for the Build functionality of Microsoft Visual Studio 2005 and 2008, standardizing the software build process because it extends the Integrated Development Environment (IDE) Build features of MSBuild to incorporate all versions of MS Visual Studio in a Continuous build environment — with innovative Build Forensics and the ability to link production binaries back to their origins.

CA Software Change Manager (SCM), formerly known as CA Harvest, provides complete source management and is tightly integrated with OpenMake Meister for Build and Binary Management. The combination of these products is the dynamic duo of source and build management in support of Visual Studio 2008, delivering unlimited development flexibility and unparalleled build consistency.

## The Challenges of Application Modernization with VS 2008

Visual Studio 2008 and the .Net Framework 3.5 were officially announced by Microsoft on November 19, 2007 and Visual Studio 2008 is now the flagship, leading-edge Microsoft Integrated Development Environment. The product enables developers at all levels to rapidly create applications for Windows Vista, Windows Server 2008, the 2007 Microsoft Office system and the Web and replaces all previous versions of Visual Studio, including:

- Microsoft Visual Studio 97
- Microsoft Visual Studio 6.0
- Microsoft Visual Studio .Net 2002
- Microsoft Visual Studio .Net 2003

The new release also contains more than 250 new enhancements and has been designed to support all levels of application development — from the novice hobbyist to professional programmers building business-class applications.

The core problem that faces application developers everywhere, however, is that the Build tools in Microsoft Visual Studio 2008 only support Visual Studio 2008 Languages. Further compounding this challenge is the fact that the MSBuild/Team Build product only supports the building of .Net objects. In other words, it cannot and does not support Java — the other leading development language.

### The Realities of Heterogeneous Build Environments

Theoretically speaking, a heterogeneous build environment is one where multiple development tools are used to deliver applications. Practically speaking, a heterogeneous build environment is one that uses both Visual Studio and Java. Visual Studio is a product built by and for Microsoft products, and as such, it is a very useful tool for development within Microsoft's framework. But, many organizations will also need to build applications for other operating systems and other languages, and more often than not, these other applications will depend on projects and applications developed within Visual Studio.

So, what happens at build time? Even if the Visual Studio portion of the development builds correctly, builds on other platforms will have to be performed using some other tool and by customizing NAnt or Nmake scripts for that portion of development. This lack of standardization creates organizational and control problems when performing diverse builds on an enterprise-wide level.

## Multi-platform Builds Call for Multi-layered Support

OpenMake Software, the industry-leading provider of build-to-release management solutions, integrates with Microsoft Visual Studio 2008 to extend build support for multiplatform and cross-language development projects. As such, OpenMake Meister supports a single build workflow that will build Microsoft Visual Studio 2005, Visual Studio .Net 2003, Visual Studio 6.0, Java JEE applications and legacy systems written in UNIX, Linux or z/OS.

By generating one Build Control File that will build Java, Visual Studio .Net, Visual Studio 6, Visual Studio 2005 and Visual Studio 2008 objects, OpenMake Meister addresses the issues surrounding building applications using multiple tools and platforms. And through Build Services, standard reusable scripts provide the framework for supporting builds that use multiple languages and tools. Specifically, Build Services can be customized through reusable scripts called Build Methods to cater to the unique requirements of each development environment.

With Microsoft Visual Studio as the IDE and OpenMake Meister helping you manage the editing and building of applications, the only missing piece for development organizations is addressing a continuous flow of complex changes to application source, which is where CA Software Change Manager comes in. CA SCM tracks changes throughout the application life cycle — from impact analysis to testing to production and follow-on maintenance — while OpenMake Meister acts as the industrial-strength build engine for CA Software Change Manager.

### Extending the Functionality of Visual Studio

Microsoft Visual Studio provides streamlined functionality for executing builds outside of the IDE using MSBuild. Because it is intended specifically for Visual Studio, however, MSBuild's primary functionality is to support builds at the Solution Level, creating C# and VB.Net objects. OpenMake Meister extends the functionality of Visual Studio 2005 Builds by supporting the creation of multiple-language and multiple-platform objects, including Microsoft Visual Studio 6, VS .Net (2003) and any extended Visual Studio objects, such as Biztalk.

Figure A shows the objects supported by VS 2008 MSBuild and how OpenMake Meister expands the Build capabilities of Microsoft Visual Studio.

**FIGURE A**

Meister enhances the usability of Microsoft Visual Studio by extending the Build capabilities of the software to many additional languages.

**OPENMAKE MEISTER SUPPORTS MANY LANGUAGES**

LANGUAGE	MSBUILD/TEAM BUILD	MEISTER
C#	Yes	Yes
J#	Yes	Yes
VB.NET	Yes	Yes
C .Net	Yes, via VCBuild.exe	Yes
ASPX	Yes, via ASPCompile.exe	Yes
Multiple Solutions	No	Yes
C - VS 6	No	Yes
C++ - VS 6	No	Yes
VB - VS 6	No	Yes
MSI Installers	No	Yes
Old .Net Project file formats, Biztalk 2005	No	Yes
Java	No	Yes
COBOL	No	Yes

### BUILD DECISION MAKING WHEN BUILDING MULTIPLE SOLUTIONS

Developers working within Visual Studio can define multiple Projects as part of a single solution, which is especially important when you want to execute builds using Microsoft TeamBuild and the generated MSBuild script. When managing multiple Projects as a single solution, the Visual Studio IDE will generate an MSBuild XML file that can be called by Team Build. Contained in this MSBuild XML file will be the required logic, called "Transforms," to build the Projects in the correct order.

But in some larger enterprise efforts, it is not practical to manage the entire application under one solution. In these cases, multiple solutions are used to create a single application. For this effort, developers must write their own MSBuild XML scripts to handle the building of multiple solutions.

OpenMake Meister allows developers to continue using the Visual Studio IDE to build multiple solutions — just as they would build a single solution. Meister generates a Build Control File that generates multiple solutions leveraging the underlying Microsoft framework. During the build process, Meister's deep dependency scanning links all of the parts together. This means that regardless of how the solutions are dependent upon one another, and even when the dependencies are circular, OpenMake Meister can accurately determine the order in which objects should be built.

### THE OPENMAKE MEISTER KNOWLEDGE BASE

Meister separates common build information from critical project-specific information. Common build information is managed in the Meister Knowledge Base, while application-specific information, which is likely to change over time, is managed in target definition file similar to a Microsoft Project file.

The Meister Knowledge Base server contains information on:

**BUILD SERVICES**, which provide a process for defining build best practices for various languages and operating systems, use Build Methods for calling development tools in a repeatable way and are defined for building Microsoft Visual Studio objects.

**BUILD METHODS**, which define the behavior of a compiler or linker that is needed to build different types of assemblies and are reused across applications and projects. Example Build Methods are ".NET Solution Executable" and ".NET Solution Installer".

**BUILD FORENSICS**, where Meister performs deep dependency scanning, spanning different languages and tools. This Build Forensic feature allows Meister to understand the inter-dependencies between objects — even when the objects are derived from different languages — such as C# and Java. This deep dependency scanning includes querying the Software Change and Configuration Management Server for revision information, which is then included in the Build Audit and provides the link from the executables back to the source code revisions.

**WORKFLOWS**, which represent a collection of commands that is performed before, during and after builds. Workflows can call version control tools, testing tools and build scripts and can be executed across different machines running different operating systems.

### VISUAL STUDIO BUILDS BEYOND THE IDE

The Meister Visual Studio Add-in provides a method for interacting with the Meister Knowledge base directly from within the Visual Studio IDE. Because developers require the ability to perform all of their daily activities from within a central tool, such as Visual Studio, the Meister Add-in extends the Visual Studio IDE directly to the build, even when the build itself requires different versions of the Visual Studio compiler (which calls for the assembly of multiple solution files or incorporates other tools, such as Java.)

Specifically, the Meister Visual Studio Add-in enables you to:

- Configure Meister Builds from within Visual Studio
- Automatically gather file-specific names to pass to the Meister Build Services for generating the build scripts (via the Meister Target Definition)
- Provide an IDE-centric location to generate build scripts and execute builds managed by Meister
- Assign projects and solutions to a Meister Build Project

And to execute builds across solutions and diverse languages like Java, the Meister Add-in can build all projects if they reside in one solution and are referenced through project references. But more importantly, Meister can still determine the correct build order when projects make non-project-based references to other assemblies that should still be built at the same time.

Finally, using Build Forensics, the Meister build engine enhances the Visual Studio build process by scanning the .csproj file for included .cs source code and for included references. These references can be project references, file references or COM references. Meister will recurse through the dependency tree resolving relationships between references, binaries and source. These Meister dependency relationships are stored as part of the Build Audit and Foot Printing and with the dependency relations stored, the executable, dynamic link library or assembly can be inspected and traced back to the source revision that created it.

### REMOTE BUILD SERVERS FOR MULTI-PLATFORM BUILD SUPPORT

A remote build server allows a developer to execute a build hosted on a machine other than the user's local desktop even when the remote machine is running a different operating system. This is useful for controlled build environments and to run a single build that needs to execute on multiple operating systems. Moreover, a Meister Workflow can be created to execute a build across multiple operating systems and Team Build can call the Meister Workflow that then executes builds and tasks across multiple operating systems.

### TEAM BUILD AND MS BUILD

OpenMake Meister passes to Team Build "execution commands" for Team Build to manage and orchestrate. These commands, called Meister Workflows, can include the execution of tasks across multiple machines and languages. This means that Team Build can use Meister to orchestrate a build across diverse operating systems and languages. For example, a build could include the compiling of Visual Studio 2005 objects on a Windows machine and Java objects on a UNIX machine.

Meister has extended the MS Build functionality in order to enable Team Build interaction with Meister's multi-platform and multi-language support and MSBuild is the link between Team Build and Meister.

### BUILD AVOIDANCE

Because building faster is critical for large projects, many enterprises choose a path of lesser resistance to achieve these types of efficiencies known as Build Avoidance. Build Avoidance simply means that your builds will be done incrementally, re-compiling only the objects that are not current. With MSBuild, you can define "Transform" relationships between objects and MSBuild will check the time and date of the relationships to determine if an object is out of date. To do so, you must code the "Transform" statements manually in the MSBuild script and in order to define how you want to handle the date/time checking process. When using OpenMake Meister to manage Build Avoidance, this date/time process is handled automatically. Meister performs deep dependency discovery prior to the compile/link process and tracks this information during the build to determine what needs to be built and what can be avoided.

Meister uses machine intelligence to follow these dependencies. In large applications, it becomes very difficult for a human to determine all of the possible interrelationships. And, managing this process manually means that there is a high probability that the incremental build will overlook a critical component and result in a missed link step — and ultimately — a bad build. In addition, the process of manually tracking these relationships within the MSBuild script is time consuming and static. As lean methodologies encourage the rapid changing of applications, the process of building the applications must easily adapt to software updates. Thus, removing the manual steps of the build is critical in pursuing truly automatic builds, regardless of language.

### WEB-SPECIFIC BUILD CHALLENGES

A common build challenge for Visual Studio developers is a web project because it involves source and assemblies, as well as a requirement to interact with the Microsoft IIS web server. MSBuild can call the process that builds the Web Project, but developers must manually create the necessary MSBuild scripts. And, some developers choose NAnt, an Apache open source language to create the build logic for Visual Studio web projects.

Regardless, the build environment can be challenged by poorly controlled and executed development practices. The problem of managing assembly references for coded objects in Visual Studio seems to be unusually difficult for many development teams. This is often because the best-practice approach is widely unknown at the beginning of the development effort and due to a lack of requirements for a concerted, manually organized code management process to be sustained. Moreover, the overuse of local file references greatly hinders portability from one environment to the next.

But, the OpenMake Meister solution for managing .NET web projects provides a development team with a best-practice-based approach for managing references. Meister can provide successful builds regardless of machine and reference scheme — and a simple, machine-independent method of finding references is highly desirable.

Meister enables the standardization of the INCLUDE and LIB settings, removing the dependency of the build on machine specific configurations that can change the result of a compile.

### Life cycle-based Change Management

Faced with competitive pressures and increasing market demands, developers have less time to implement changes than in the past. Many organizations — especially those heavily invested in web technology — might need to deliver new applications in weeks rather than months or years. And with the added pressure that only top-quality, error-free applications be allowed into production, the challenge to successfully track and implement application code changes can seem insurmountable.

CA SCM delivers a single enterprise-wide solution for tracking software changes and managing the application development process. Comprehensive, integrated and repository-based, CA SCM helps you better manage the complex workflows associated with application development projects, including:

- Automation of the creation and routing of change and service requests and other routine tasks
- Collection of critical audit information
- Maintenance of change history

Through simple point-and-click, drag-and-drop operations, CA SCM helps you control changes to application documents and software, keep development schedules on track and ensure that team members and business users are always up to date with the latest status information. Altogether, this allows developers to meet demanding delivery schedules in a timely fashion while maximizing productivity, reducing development costs and improving application quality.

### CA SCM and OpenMake Meister Together

CA SCM and OpenMake Meister are tightly integrated and when used in combination with Visual Studio, they provide a full end-to-end change management environment that offers easy access to change information while providing a full Audit Trail. And by adding the OpenMake Meister Build Audit Report to CA SCM, your entire change package can be audited for years after the change was actually made.

---

## SECTION 3: BENEFITS

### Maximizing Builds With an Industrial-strength Engine

OpenMake Meister supports multi-platform builds by extending the Build features of Microsoft Visual Studio to all platforms and languages. To achieve the benefits of multi-language and multi-platform builds to the Visual Studio developer, Meister leverages Microsoft Visual Studio and its Team Build technology. CA SCM provides complete source management of the process and is tightly integrated with OpenMake Meister for Build and Binary Management. When used in combination, these three products ensure effective heterogeneous build environments that leverage the best the Meister software has to offer, such as:

- Unlimited development flexibility by supporting nonMicrosoft platform builds and extending build features to nonVS 2005 objects
- Unparalleled build consistency by managing VS 2005 builds that are common across solution files, web projects and other development tools
- Streamlining efforts by linking dependencies of multiple solution level files for Build Audit, Impact Analysis and Build Avoidance
- Minimizing confusion while maximizing throughput by allowing any user to perform a build for one module, many modules or an entire project
- Improving change management by enabling control of project and file references outside of the developer's desktop
- Creating efficiencies via a portable method of rebuilding applications at any check pointed stage in the development life cycle (REV 1.1. REV 1.2, and so on).

---

#### SECTION 4: CONCLUSIONS

OpenMake Meister is a critical tool when working with the Build functionality of Microsoft Visual Studio 2005 and 2008, standardizing the software build process and extending the IDE Build features of MSBuild to incorporate all versions of Visual Studio in a Continuous Build environment. CA SCM provides complete change management and is tightly integrated with OpenMake Meister for Build and Binary Management. The combination of these products provides the source and build management support developers require to use their existing code within the new Visual Studio 2008 and also support multi-language, multi-platform Visual Studio 2008 projects.

---

#### SECTION 5: REFERENCES

“OpenMake plugs build automation into Eclipse, Visual Studio,” November 6, 2007, Tony Baer, CBR. [http://www.cbronline.com/article\\_news.asp?guid=3ABCAA9D-887F-40CB-812B-79FCE7464B5F](http://www.cbronline.com/article_news.asp?guid=3ABCAA9D-887F-40CB-812B-79FCE7464B5F)

“Microsoft to Give Partners More Access to Orcas IDE Code,” November 5, 2007, Darryl K. Taft, eWeek. <http://www.eweek.com/c/a/Application-Development/Microsoft-to-Give-Partners-More-Access-to-Orcas-IDE-Code/>

“OpenMake Software Utilizes Microsoft Technologies to Improve Enterprise Software Builds,” November 29th, 2007, <http://www.telecommarketnews.com/2007/11/openmake-softwa.html>

“Separating Duties to Meet IT Compliance,” August 02, 2007, Tracy Ragan, *Dr. Dobbs Journal*, <http://www.drdobbs.com/development-tools/201202677;jsessionid=RKLZIYO4RATGAQSNDLRSKH0CJUNN2JVN>

“Build Management Solutions for the Microsoft Developer,” January 10, 2008, <http://www.openmakesoftware.com/Build-Management-Solutions-for-the-Microsoft-Developer/>

## About the Author

A Senior Product Marketing Manager of EITM Service Management at CA, Steve Solomon has been involved in Software Configuration Management (SCM) for 20 years, having served as the ChangeMan ZMF Product Manager at Serena Software, and a Director of worldwide software change management at American Express. Mr. Solomon received his Bachelor of Business Administration in Computer Information Systems from Boise State University, Boise, ID.

**Steve Solomon**  
CA EITM SERVICE MANAGEMENT

**Tracy Ragan**  
OPENMAKE SOFTWARE

Tracy Ragan is Chief Operating Officer of OpenMake Software. She has extensive experience in the development and implementation of business applications and began her consulting career in 1989 working with Fortune 500 organizations in the areas of testing, configuration management and build management. It was during this period that Ms. Ragan recognized the lack of build management procedures for the distributed platform that had long been considered standard on the mainframe. In the four years leading to the creation of OpenMake Software, she worked with development teams in implementing a team-centric standardized build management process. Ms. Ragan currently serves on the Eclipse Foundation Board of Directors as an Add-in Provider Representative. She received her Bachelor of Science Degree in Business Administration from California Polytechnic University, Pomona, CA.

---

To learn more about the CA Software Change Manager architecture and technical approach, visit [ca.com/us/products](https://www.ca.com/us/products).

---

## Notes

---

## Notes

CA (NYSE: CA), one of the world's leading independent, enterprise management software companies, unifies and simplifies complex information technology (IT) management across the enterprise for greater business results. With our Enterprise IT Management vision, solutions and expertise, we help customers effectively govern, manage and secure IT.

MP32780 TB05OMAKE01E

---

Learn more about how CA can help you transform your business at [ca.com](https://www.ca.com)

