



Improving Software Quality with a Clean Build Process

*Vanessa Wasko
Wise Solutions, Inc.*

WHITE PAPER

Abstract

Automating the software build process is a major step towards consistently producing quality software. To achieve complete control over the build quality, however, software must be built in a truly clean environment. Wise Solutions offers a complete solution for an automated clean build process.

Introduction

Software quality continues to be increasingly important due to the high risks associated with security flaws and the growing intolerance for product defects. At the same time, software is becoming more complex and the pressure to release major new versions more frequently is increasing. Successful software development teams address these issues by implementing a consistent, repeatable development process that is built to ensure software quality. An automated software build process is a critical piece of that process.

Most independent software vendors and corporate development teams have created some type of automated build process. In many cases, this process is a homegrown combination of scripts and batch files that retrieve the source code from a source control system, compile the source code, and build the product installation. This build process typically runs on a developer's PC, or for larger teams, on the PC of the build engineer. While an automated build process is a key step towards consistently producing high quality software, there is a significant element missing from this process – the clean build environment.

This document outlines how using an automated build process in a clean build environment can increase software quality, decrease security risks and decrease time to market. The Clean Build feature of Wise for Windows Installer – Professional Edition and Wise for Visual Studio .NET is then discussed to show how to easily create an automated build process in a clean build environment. This white paper is intended for professionals in development, quality assurance, or release management who maintain any portion of the build process.

Automating the Build Process

Complete control of the software build environment is required to prevent partial or complete compilation failures, security breaches or contamination of files. This controlled environment is utilized to pull code from the password encrypted source code control software, compile the source code, and finally to compile the resulting binaries into the chosen distribution method (.MSI, .EXE, .CAB). This resulting distribution file can then be passed to quality assurance for testing. Following this methodology can improve software quality while streamlining the build process.

The build process is a critical element for all software development projects. Do not be tempted to get by without one – particularly in a team development environment. You should configure a build server and create the necessary build scripts as early as possible in the development cycle – certainly well before you are ready to begin integration testing.

Microsoft Corporation

Through interviews with its existing customers, Wise Solutions has found that many development groups use an ad-hoc build process methodology. Although this may have worked well in the past, the increasing demands for producing frequent, high quality releases have changed this. Developers are finding an automated build process methodology to be a necessity in today's software market.

Performing continuous or nightly builds allows you to see how the code works when it's integrated into a production environment. Incorporating unit testing and regularly automated builds throughout the development cycle assures both you and your clients that your code will be solid on delivery.

Erik Hatcher

Senior Architect, eBlox

The industry recommended build-process method is to automate the process via scripting methodologies, or automation. When compilation is automated, builds can be made more frequently and on a regimented schedule. This provides a more reliable testing schedule and more accurate depiction of the current state of the product during testing.

Common Problems with Automated Build Processes

Most automated build processes are performed on the PC of the build engineer or one of the developers. This approach is inherently flawed due to the lack of control over this environment. Here are some common problems that can occur with this approach:

- A virus on the build PC could be inadvertently included in a software build
- Installation of new software or software updates on the build PC could change resources that are included in the software build
- Uncontrolled access to the build PC could result in unauthorized changes to the software build, including potential malicious changes that could be difficult to detect

The Clean Build Environment

While an automated build process is an important step towards higher quality software, the only way to achieve complete control over the build quality is for the software to be built in a clean build environment. This clean build environment is comprised of one or more build PCs that are used only for the purpose of creating the builds, and is typically not even connected to the company's network. In some cases, this will be a single PC, while for companies like Microsoft, there may be several dedicated build labs consisting of dozens of PCs.

The Clean Build Environment is created following the standards listed below. Based on the size of your organization, this may involve one or more PCs.

- A completely clean PC that includes only the operating system; no software products of any kind should be installed on this PC
- A network connection that can be easily disabled, or a CD-ROM or ZIP drive to transfer the software resources to this PC
- A product to perform the software build that does not require a software installation and does not modify the registry or system files in any way

All development work is performed on the developers' workstations, and is then checked into a source control system. The binary resources must then be copied to the build PC for the creation of the software installation package. Until a recent innovation from Wise Solutions, the process of building the installation in a clean build environment was very difficult.

Most installation development tools require that they be installed on the build PC, as the required system settings, registry keys, and binary files need to be installed prior to running the compiler. Having a third-party product installed on the build PC violates the concept of the clean build environment because it tarnishes the pristine environment that is required for a stable, reliable, and secure system.

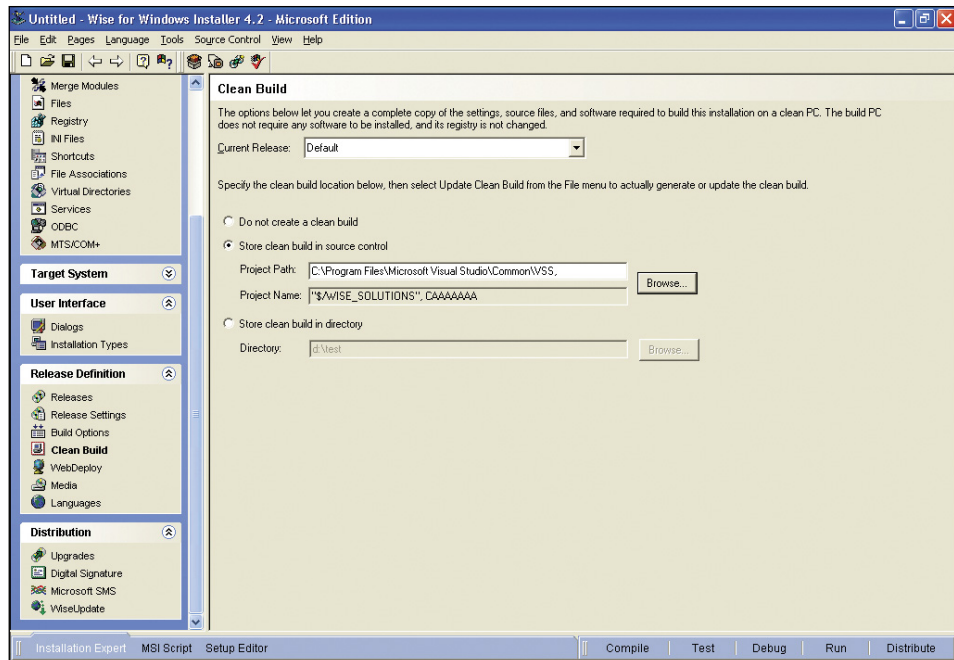
Wise Solutions' Clean Build Innovation

In June 2002, Wise Solutions introduced the first and only commercial solution for creating builds in a clean build environment. This innovation is available in both of its Windows Installer authoring environments: Wise for Windows Installer – Professional Edition and Wise for Visual Studio .NET.

The Clean Build feature creates a single build folder that contains every resource needed to compile a Wise-generated project file (.WSI) into a Windows Installer package (.MSI). This folder just needs to be copied to the build PC and the compile needs to be initiated. No software needs to be installed or uninstalled on the build PC. Here are some additional details about this feature:

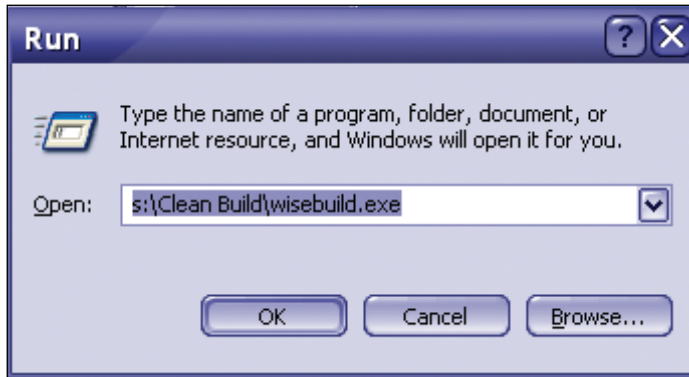
- The Wise compiler and support files are automatically stored in this clean build folder using a single command in the Wise product.
- There is less than 9MB of overhead associated with the Wise application.
- The clean build folder can be copied to the build PC using a ZIP disk, CD-ROM, network directory, or from a source control system.
- The tight integration with source control systems provides a complete audit trail for your software builds.
- To maintain the pristine environment on the build PC, there is no installation or uninstall needed on the build PC. The build folder is just copied to the build PC, the compile is performed, and the build folder is deleted. After this process is complete, the build PC is back to its original pristine state.
- All of the settings defined during the authoring of the installation are automatically used when compiling on the build PC.

The following screen shot shows the interface for this feature in Wise for Windows Installer:



The approach to creating the clean build folder can be customized for each release type. For example, the production release can store the clean build folder in Visual SourceSafe, while the evaluation release can be stored on a network directory. Providing this flexibility for each release type extends support for those development teams who have different release managers for individual releases.

To run the Clean Build compilation on the build PC, simply launch WiseBuild.exe located in the root directory of the clean build folder. No command line options are required, as the Wise project file (.WSI) in the clean build folder will be automatically compiled into an .MSI. All compilation settings defined in Wise for Windows Installer or Wise for Visual Studio .NET are automatically used during the compilation on the Build PC.



NOTE: You must use the Wise project file (.WSI) to use the Clean Build feature. This feature will not recompile an .MSI, it will only compile a .WSI into an .MSI.

With each compilation, a log file is generated documenting the compilation process. The log file will be the same name as the Wise project file (.WSI) and is saved in the root of the clean build folder. Any compilation errors will be written to this log file. See Figure 1 for an overview of the complete process for creating a clean build.

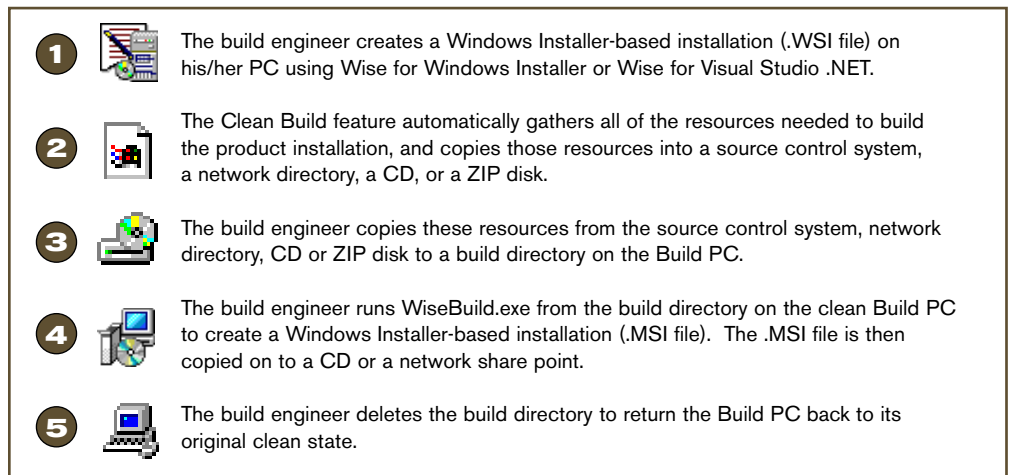


Figure 1: Overview of the clean build process using Wise for Windows Installer or wise for Visual Studio .NET.

Conclusion

Without this Clean Build feature from Wise Solutions, there is no way to compile an .MSI package on a clean build PC without spending hours developing VBScripts and .MSI build processes with Orca or other homegrown utilities. Wise Solutions continues its track record of innovation with this major improvement upon the automated build process.

As software quality and security becomes more important, responsibility for ensuring high quality and secure installation packages must be shared by all software developers. Utilizing a clean build environment for compilation and release management provides the level of quality necessary to provide a solid and secure product.

For More Information

Microsoft has created a document that provides development and procedural guidance for project teams building .NET applications with Visual Studio .NET and Visual SourceSafe. It discusses the processes, disciplines, and .NET development techniques that team members must adopt in a team development environment. It also describes how to create the necessary development infrastructure, which includes source control databases, development workstations, and build servers.

See <http://microsoft.com/downloads/release.asp?ReleaseID=35981> for more details.

Wise Solutions Inc.
47911 Halyard Dr.
Plymouth, MI 48170 USA
734 456 2100
www.wise.com

Wise for Windows Installer is the next generation of Windows Installer authoring tools that guide users through the entire installation development lifecycle, including the design, import, author, customize, build and distribution phases. Developers can create professional, Web-based installations that support the latest technologies, including: Microsoft .NET Framework, XML Web services, MTS/COM+ and 64-bit installations. You can learn more about Wise for Windows Installer at www.wise.com/wfwi.asp.

Wise for Visual Studio .NET is a specially designed, fully functional Wise installation authoring product that operates directly within Visual Studio .NET. By merging the installation and application development lifecycles so that applications and installations are designed, coded and tested together, you can automatically create and test reliable, high-quality installations from the first line of code. You can learn more about Wise for Visual Studio .NET at www.wise.com/visualstudio.asp.