

Microsoft App-V: What Software Developers Should Know



Microsoft App-V: What Software Developers Should Know

Introduction

With the increased adoption of Microsoft App-V by corporate and government organizations, the software installation development landscape is beginning to change. It's a shift that is reminding many developers and industry analysts of what happened with Windows Installer (MSI) ten years ago.

Prior to Windows Installer, developers built installs as stand-alone script-based executables. Windows Installer was released as a more reliable alternative, and companies began adopting it in order to achieve more predictable and consistent software deployments. This change from script-based installs to MSI installs required software companies around the world to rewrite their installation setups and instead deliver MSI packages to their user base.

Today, a similar shift seems to be occurring with App-V. Just like with MSI, as more and more companies are adopting App-V for its cost-reducing and time-saving benefits, it is driving software companies to provide their applications to companies as both App-V and traditional MSI packages. Remember how one of the critical first steps to mainstream MSI adoption occurred when Microsoft released Office 2000 with an MSI installer? When Microsoft released Office 2010, they also provided customers the option of downloading the application suite as an App-V virtual package. And since then, other software companies have followed suit and offered their products as App-V packages.

So what exactly is Microsoft App-V? Why is it becoming popular with enterprises, and what options exist to help software developers build an App-V package? This white paper answers these questions and gives software development teams information on how to integrate App-V package authoring into their regular MSI development process.

Why Software Companies Are Providing Customers the App-V Option

Software companies today are making their applications available as App-V packages for one simple reason: their customers want it.

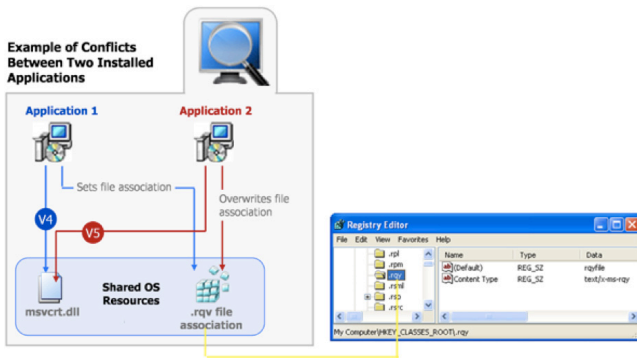
For enterprises and government organizations, App-V has many time- and cost-saving benefits over traditional installed applications. However, standardizing on App-V requires legacy and newly purchased applications be converted to the App-V format, and that can be a time-consuming process that can tie up IT resources. Software companies that proactively provide their applications natively in the App-V format are directly helping their customers reduce the costs of supporting their application by eliminating the need for this time-consuming conversion work.

Limitations of a Traditional Installation Environment

To help better explain the benefits of application virtualization technologies like App-V, let's examine the limitations of a traditional MSI installation. A typical Windows application has dependencies on components that are shared by multiple applications. Applications access these shared system resources, such as the registry or Windows system files. When an installation author recognizes that their application references a shared system component, they include a merge module to install that component.

When one of these shared components is installed during an application's installation, it is possible that a previously installed version of the same component could be overwritten, causing the existing application to break. Because of these possible problems, extensive compatibility testing needs to be performed before an application can be deployed in the enterprise environment.

The following diagram provides an example of two conflicting installed applications.



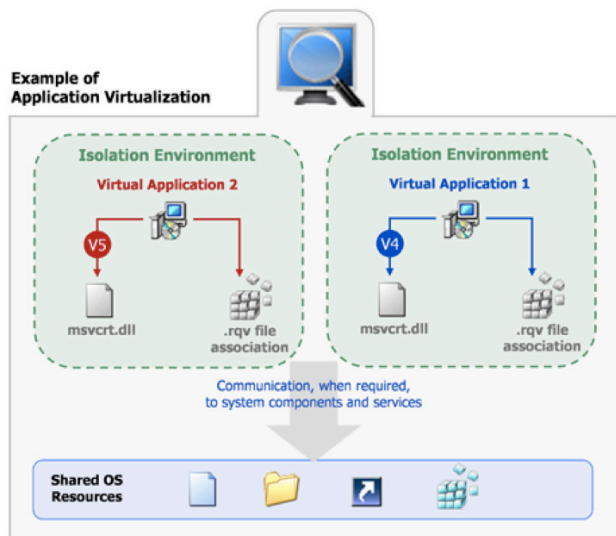
Also, if users within an enterprise run multiple versions of an operating system (like Windows 7 and XP), it makes it more difficult and time-consuming to support deployed applications, as you have to test on both versions of operating systems.

Benefits of Virtual Applications

Virtual applications run in virtual environments that keep the application layer and the operating system layer separate. Each application includes its own configuration information in its virtual environment. As a result, many applications can run side-by-side with other applications on the same computer without any conflicts.

Even though virtual applications are not installed on the local machine, they exhibit the same performance, functionality, and access to local services as a locally installed application would have.

The following diagram provides an example of how application virtualization would solve the conflicts shown in the previous example.



Application virtualization allows the configuration of an application to be standardized to an isolated environment, rather than to an individual user’s desktop machine. Application objects, files, and registry settings are contained within this isolated environment. Critical application resources are managed locally by the isolated environment, thus minimizing resource dependencies between applications.

Application virtualization technologies like App-V transform applications into virtualized, network-available services that are never installed and rarely conflict, minimizing costly compatibility testing. Users and their application environments are no longer machine-specific, and the machines themselves are no longer user-specific. This can enable an enterprise’s IT team to be much more flexible and responsive to business needs, and dramatically reduce the cost of desktop management initiatives.

Also, with App-V you can run multiple versions of an application side-by-side on the same computer. Many applications such as Microsoft Office don’t allow different versions to co-exist on a machine. However, it is often possible to create App-V packages for the different versions and deploy them simultaneously. Side-by-side execution is helpful in many scenarios, including enabling your end users to test out a new version of your application while still running the old one.

Another extension of this is the capability to execute the same App-V application simultaneously by multiple users. Some traditionally installed applications can’t be run multiple times due to resource locking, but can run when deployed as App-V applications. This is especially useful when using Remote Desktop Services.

Save Your Customers Time and Resources

Software companies that generate App-V packages natively for their own applications save their customers the time and effort of having to do the conversion work themselves. Vendor-provided virtual packages are seen by IT customers as preferable, since a software vendor will be most familiar with how to optimally create the App-V package for their specific application, rather than a system administrator who must rely on a conversion tool to accomplish the same task. By delivering a virtual package along with the traditional MSI package, software vendors can offer much greater value to their customers.

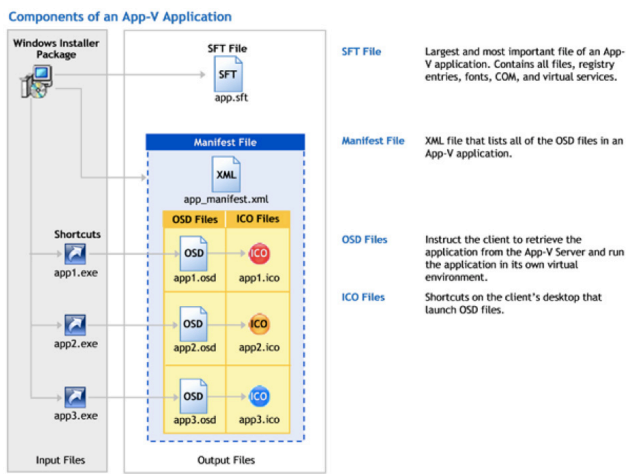
About Microsoft App-V

So what exactly is an App-V package and what are its components? Microsoft Application Virtualization (App-V) allows applications to be deployed in real-time to any client from a virtual application server, removing the need for local installation of the applications. Instead, only the App-V client needs to be installed on the client machines.

Application data is stored on the virtual application server. Whichever software is needed is either streamed or locally cached from the application server on demand and run locally when a user tries to access an associated shortcut or file extension. App-V isolates the execution environment so that the application does not make changes to the client itself. The lifetime and maintenance of App-V applications can be more easily managed centrally by your end users' IT team. App-V applications can also be set-up to depend on other App-V packages through the Dynamic Suite Composition functionality

Components of an App-V Application

The components of an App-V application are shown in the following diagram:



The following table also describes the components of an App-V application:

FILE	DEFINITION
SFT	The largest and most important file in an App-V application. All of the application's assets—including files, registry entries, fonts, COM, and virtual services—reside in this file. The maximum size of this file is 4 GB.
OSD File	An XML-based file that instructs the client about the necessary details for retrieving the App-V application from the App-V Server and running the application in its own virtual environment.
ICO File	ICO files are the icons that will be used for shortcuts and file associations.
Manifest File	XML file that lists all of the OSD files in an App-V application.
SPRJ File	App-V Sequencer project file.

Components of an App-V Application

Options for Creating an App-V Package: App-V Sequencer or InstallShield

So now you know about Microsoft App-V and the reasons why enterprises want you to deliver applications in MSI and App-V formats. The next step is learning how you can generate an App-V package, and for that task there are two options available: the Microsoft App-V Sequencer and InstallShield from Flexera Software.

There are many advantages of using InstallShield's App-V support for creating App-V packages over the App-V Sequencer. But the fundamental difference, however, is the App-V Sequencer uses installation monitoring to create an App-V package (which requires a separate set of steps to follow) while InstallShield creates an App-V package directly from your InstallShield project by reading the MSI tables. It provides an integrated process for building both deployment formats – MSI and App-V – for your applications.

The following section describes both approaches and some of the many advantages of using InstallShield to create an App-V package.

About the App-V Sequencer

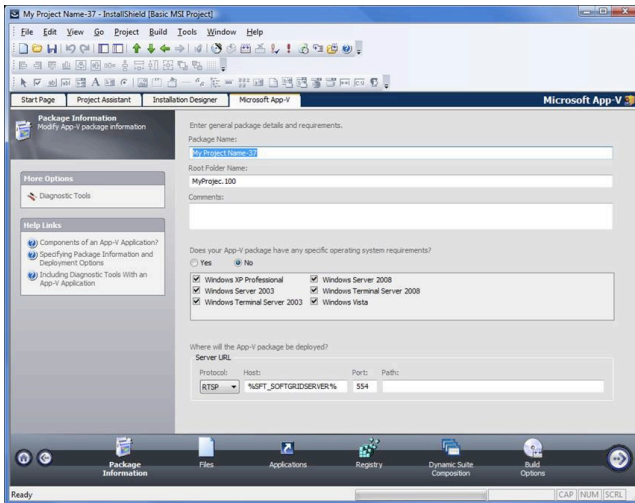
The App-V Sequencer works by monitoring the installation and setup process for an application, and it records the data necessary for the application to run in a virtual environment. It then takes this information and uses it to create an App-V package.

Sequencing, as it's more commonly known, is its own separate task that is performed after you complete your overall MSI installation development process. It requires a clean packaging environment with any prerequisites pre-installed in order for the setup to run to completion without any errors. Monitoring system changes in this fashion can be time consuming. The process can also capture a large amount of unneeded data that you would be better off excluding from the finished App-V package.

After sequencing, some changes to the final package are possible using views that loosely correspond to the different types of information within the package (files, registry, services, metadata, OSD files for shortcuts, etc.). In many cases, making changes to an App-V package with the Sequencer requires you to start the entire installation monitoring process over again from scratch. If your release management process requires you to build your installation package on a daily basis, incorporating sequencing into this process can be difficult.

Advantages of Creating App-V Packages with InstallShield

InstallShield works differently from the Sequencer. Instead of requiring you to add a whole new set of steps (using a different tool) to your installation development process, InstallShield builds the App-V package for you as part of your MSI project. It does all the work automatically, so you don't have to learn any new technologies.



There are several important advantages to using InstallShield instead of the Sequencer:

- **One installation project for both MSI and App-V packages:** Only InstallShield enables you to automatically integrate virtual package development into your normal MSI development process. With one project, InstallShield can simultaneously generate an MSI package and an App-V package based on the finished MSI. You don't need to spend time with manual sequencing; with InstallShield you develop and maintain both packages in parallel rather than individually.
- **Cleaner, more accurate App-V packages:** Letting InstallShield build your App-V package helps ensure the final output does not include unneeded data that would require manual diagnosis and removal later. The Sequencer will often capture and include unwanted files and information in an App-V package. With InstallShield, only the contents of your original installation are included.
- **Easy package editing:** Structural changes in the App-V package are possible by simply editing with the InstallShield IDE. With the Sequencer, changes often involve starting the entire sequencing process over.
- **Automated nightly builds:** Many development teams modify their application data by adding or removing files on a daily basis. As you make these changes, InstallShield can automatically edit and rebuild your App-V packages to include the latest version of your application files. You can set InstallShield to run automatically each night, ensuring when you come to work the next day your App-V packages will be up to date.
- **Testing on your local system:** InstallShield creates a bootstrapper for each App-V package that can be used to test the package on your local machine, saving you time.

- **Debugging:** InstallShield makes it easy to include debug tools such as cmd.exe for viewing the virtual file structure, and regedit.exe for viewing the virtual registry. This makes it easy to include and launch other debugging tools from within the virtual package (such as tools available from Microsoft SysInternals, Depends.exe, OllyDbg, etc.).

Learn More about InstallShield

If you wish to learn more about the capabilities of InstallShield, please visit the Flexera Software Web site at <http://www.flexerasoftware.com/installshield>.

Steps to Take to Create an App-V Application Using InstallShield

InstallShield includes a Microsoft App-V Assistant to assist you in building an App-V package. The basic steps are as follows:

1. Specifying Package Information and Deployment Options

When you are creating an App-V application, the first step is to specify the package name, root folder name, and enter a comment on the Package Information page. You also specify any operating system requirements, identify the deployment server, and specify whether to include diagnostic tools with the virtual package.

2. Managing Files in an App-V Application

InstallShield shows you all of the files and folders that are currently in your App-V application and makes it easy to add or delete files and folders as needed.

3. Setting Isolation Options for Folders and Files

You can set isolation options to control application compatibility and accessibility. The isolation option that is assigned to a file or folder specifies how the isolation environment will provide access to system resources requested by the application. Isolation options for files and folders are always inherited. The App-V isolation environment will apply the most specific reference to that resource.

4. Modifying Shortcuts to the App-V Application's Executable Files

You define application shortcuts to enable users to launch an App-V application from within the virtual environment. By default, InstallShield creates App-V applications for all of the executable shortcuts that exist in your project (or Windows Installer package). These shortcuts are listed in a checklist on the Applications page. Keep in mind that you must define at least one shortcut to enable users to launch the application from the isolation environment.

5. Modifying App-V Application Registry Settings

Using InstallShield, you can view existing registry keys, values, and data, and add or delete registry items in your App-V application. You can also set isolation options for selected registry keys.

6. Performing Dynamic Suite Composition

The point of application virtualization is to minimize the system dependencies that an application has on the underlying physical system. Many applications have common system dependencies on plug-ins or middleware, such as Adobe Reader or ODBC drivers. Dynamic Suite Composition (DSC) is a Microsoft Application Virtualization feature that enables applications to be virtualized separately from the plug-ins and middleware applications that they rely on, while still enabling them to communicate with those plug-ins and middleware applications within the virtual environment. The primary App-V application and the dependency App-V applications in the dynamic suite will run and interact with one another as if they were all installed locally on a computer. You would only need to deploy common system components once on each client, making them available for use by many App-V applications, rather than to include them with each of the App-V applications that are dependent upon them. This reduces redundancy in the local App-V cache and simplifies the construction and testing of the primary App-V applications.

7. Modifying Build Options

You can choose which releases of this InstallShield project you want to build an App-V application for when the project is built, and specify whether you want to include additional Windows Installer packages in the virtual package. Also, if you are editing a Windows Installer package in Direct Edit mode, you need to select the Build App-V application option on the Build Options page before you will be able to build an App-V application for that Windows Installer package.

8. Building an App-V Application

If you are building an App-V package for an InstallShield project, then the output of the build is the main Windows Installer setup package, the App-V package, and optionally another Windows Installer package meant for doing standalone deployment of your App-V package without the need for the App-V server infrastructure. If you are building an App-V package directly from the main Windows Installer setup package, then you get same as above minus the main setup package since it already exists.

9. Testing an App-V Application Using the App-V Application Launcher

You can use the InstallShield App-V Application Launcher to locally test a newly built App-V application. The App-V Application Launcher is a convenient testing tool that makes it possible for you to reliably and accurately test your App-V applications on your local machine or any other system that has the App-V client installed before moving it to the App-V server.

How Transforms are Included in an App-V Application

InstallShield supports the inclusion of transform files with Windows Installer packages in an App-V application.

- **How transforms are applied when building an App-V application**—When building an App-V application, transforms that you have specified are automatically applied to the base Windows Installer (.msi) package to create a temporary package, and then the App-V application is generated from that temporary package.
- **Creating a new transform**—You can create a new transform in InstallShield, and then build an App-V application from that transform file. When you create a new transform file in InstallShield, you specify the root .msi file in the Open Transform wizard. The steps you take to generate an App-V application after using that wizard are exactly the same as if you were editing a Windows Installer package in Direct Edit mode.
- **Converting a Windows Installer package with existing transforms**—If you have a Windows Installer package and one or more existing transform files, and you want to include these transforms in the App-V application, you need to open one of the transforms in InstallShield (rather than the .msi file). The Open Transform wizard will open, and you will be prompted to specify the root .msi file and which of the existing .mst files you want to include. The steps you take to generate an App-V application after using that wizard are exactly the same as if you were editing a Windows Installer package in Direct Edit mode.

How Windows Services Are Integrated into an App-V Application

When you use InstallShield to convert a Windows Installer package to an App-V application, references to Windows services that are encountered are integrated into the App-V application. In a Windows Installer package, a Windows service may be indicated by either an entry in its ServiceInstall table or by a Registry entry for Windows services.

- **ServiceInstall table**—If a Windows Installer package’s use of a Windows service is indicated by an entry in the ServiceInstall table, InstallShield will convert that entry to a standard Registry entry for Windows services.
- **Registry entry**—If a Windows Installer package’s use of a Windows service is indicated by a Registry entry for Windows services (perhaps as the result of being repackaged), InstallShield does not need to make any changes to support the application’s use of the Windows service within the virtual environment.

If an App-V application has an associated Windows service, App-V will start up the Windows service first, in the virtual environment, and then start up the App-V application. You will see the Windows service start up in the Task Manager as a separate process, but App-V will be running the service within the virtual environment.

Upon shut down, App-V will first shut down the App-V application and then shut down the Windows service.

Summary

This white paper discusses why many software companies are now delivering their applications to customers as both a traditional MSI package and a virtual App-V package. It also explains the options available to software development teams for developing App-V packages, and the advantages of using InstallShield for building App-V packages over the Sequencer.

Begin a Free Evaluation of InstallShield

You can download a free trial version of InstallShield from the Flexera Software Web site at:
www.flexerasoftware.com/installshield/eval.

Want to learn best practices for building quality installations? Join an InstallShield training class – visit www.flexerasoftware.com/training for available classes.

Also, if you have a critical installation project but are short on developer bandwidth or expertise, Flexera Software’s Professional Services team can help. Learn more at:
www.flexerasoftware.com/services/consulting/software-installations.htm.

About Flexera Software

Flexera Software is the leading provider of strategic solutions for Application Usage Management; solutions delivering continuous compliance, optimized usage and maximized value to application producers and their customers. Flexera Software is trusted by more than 80,000 customers that depend on our comprehensive solutions—from [installation](#) and [licensing, entitlement and compliance management](#) to [application readiness](#) and [enterprise license optimization](#) - to strategically manage application usage and achieve breakthrough results realized only through the systems-level approach we provide. Flexera Software is a privately-held company and an investment of private equity firm Thoma Bravo, LLC. For more information, please go to: www.flexerasoftware.com



Flexera Software, Inc.
1000 East Woodfield Road,
Suite 400
Schaumburg, IL 60173 USA

Schaumburg
(Global Headquarters)
+1 800-809-5659

United Kingdom (Europe,
Middle East Headquarters):
+44 870-871-1111
+44 870-873-6300

Japan (Asia,
Pacific Headquarters):
+81 3-4360-8291

For more office locations visit:
www.flexerasoftware.com