Virtualizing Resources for the Application Services and Framework Team

Justin T. Varner
Pennsylvania State University, University Park, PA, 16802

Linda K. Crawford
National Aeronautics and Space Administration, Kennedy Space Center, FL, 32815

Nomenclature

AS = Application Services
Cx = Constellation program
IP = Internet Protocol
KSC = Kennedy Space Center
LCS = Launch Control System
NASA = National Aeronautics and Space Administration
ODIN = Outsourcing Desktop Initiative for NASA
OS = Operating System
VM = Virtual Machine

Virtualization is an emerging technology that will undoubtedly have a major impact on the future of Information Technology. It allows for the centralization of resources in an enterprise system without the need to make any changes to the host operating system, file system, or registry. In turn, this significantly reduces cost and administration, and provides a much greater level of security, compatibility, and efficiency. This experiment examined the practicality, methodology, challenges, and benefits of implementing the technology for the Launch Control System (LCS), and more specifically the Application Services (AS) group of the National Aeronautics and Space Administration (NASA) at the Kennedy Space Center (KSC). In order to carry out this experiment, I used several tools from the virtualization company known as VMWare; these programs included VMWare ThinApp, VMWare Workstation, and VMWare ACE. Used in conjunction, these utilities provided the engine necessary to virtualize and deploy applications in a desktop environment on any Windows platform available. The results clearly show that virtualization is a viable technology that can, when implemented properly, dramatically cut costs, enhance stability and security, and provide easier management for administrators.

I. Introduction

One of the most egregious problems facing IT administrators today is the obscene amount of time required to manage servers and resources rather than spending time innovating and creating new solutions. The conventional wisdom is to use one server for one purpose; for example, one server will be designated for print services, another for email services, and so on. Consequently, this method results in a vast underutilization of resources and an exponentially greater amount of time required to manage these resources, resulting in frustrating administration and expensive energy bills. As such, the market has demanded an effective new approach to managing IT resources efficiently and effectively, and this new solution is virtualization.

A. Overview of Virtualization

Virtualization is a nascent technology that has demonstrated potential in completely changing the way people compute. The modern computers of today are designed to run one operating system and one application at a time. Compatibility of applications is contingent on whether the software supports the given platform, and migration across different computing environments is either extremely difficult or impossible. A common problem in recent

years has been the migration of users from existing Windows XP machines to Windows Vista and 7 machines. Many of the existing applications would not work on the newer OS's until the manufacturer decided to release an update, which in turn resulted in outrage as users were unable to use applications that may have been required for their professional work. But virtualization can fix this dilemma.

The technology works by taking the existing resources of a modern computer (CPU, RAM, Hard Drive), and creating a virtual machine that emulates the complete functionality of a real computer. This is a simple a piece of software that tricks the computer into running multiple OS's using the same shared resources. As such, a virtual machine has the ability to run Windows, Mac OSX, Linux, and more on the same physical machine simultaneously.

However, this is just a rudimentary example of the power of virtualization. This technology can be applied to a complete enterprise environment including the servers, mainframes, databases, and more. Resources can be virtually allocated wherever and whenever they are needed in the private cloud, as it is sometimes referred to in the enterprise IT environment (a cloud is simply the total collection of all the networked resources of an organization).

This may sound great on paper, but here is a real world example to demonstrate the promise of this technology. An organization could have a high demand for a certain application, such as Microsoft Publisher 2007. If the organization relied on the traditional method of accessibility, the administrator would need to install this application on every machine that required it. Each user would have to work on their particular machine, and if they wanted to carry their files and settings with them, they would need to use a USB drive or some other portable device. Obviously, this old way of thinking is inefficient and cumbersome for administrators and users alike.

With virtualization, the administrator could simply virtualize Microsoft Publisher 2007, put it in the cloud, and any user from any machine could run the program without ever having to install or configure anything. The user's settings and files would all be stored in the cloud, allowing for unmitigated access anywhere on the corporate network. Moreover, all of the processing would be done on the virtualized server, which eliminates the need to invest in powerful desktop machines with substantial processing power; a simple terminal would suffice.

B. Applying Virtualization to the LCS

With the basic highlights of virtualization covered, it is time to focus on my particular project as it pertains to the goals and initiatives of the LCS, and more specifically the AS group. The idea originally proposed was to virtualize the resources that the team commonly uses, namely the Netbeans IDE. The end result would be a package with the complete set of integrated tools that the team needs to complete their work; the IDE, Java Runtime Packages, widgets, and so forth. Because I was working with this technology on a smaller scale, I only needed to use the client application VMWare ThinApp, which is a desktop virtualization program designed to virtualize applications that are run on a common windows machine (server and datacenter programs are beyond the scope of this report). Moreover, it is also recommended that the user performs this packaging on a virtual windows machine. Unfortunately, I was not able to obtain a license for a separate Windows OS, so I had to package the virtual applications on the native Windows XP installation. Nonetheless, the process will be described in complete detail in the following section.

As mentioned previously, virtualization provides many benefits over conventional application deployment, and pertinent to the LCS, the specific reasons to use virtualization include:

-Full support for Windows 7 so that when ODIN decides to migrate the existing desktops to the new OS, all of the current applications used will be seamlessly transitioned to the new platform without any additional installation, configuration, or administration; just run the virtualized application and reap the rewards.
-Better performance. Due to the fact that the application leaves no digital footprint, the program can run directly from the compressed state without the need to cache data or decompress files.
-Migration of user settings and data with the simple utility of a USB drive. There will not be any reason to back up data to several different places or limit work to a particular machine. All of the information will be tightly integrated in the virtual package and can be run from anywhere from any device at any time.

These are but a few benefits of using the technology for the LCS. In essence, this is just the foundation for what virtualization can provide for the LCS and eventually NASA as an agency once more people get behind 100% virtualization as all corporations on the Fortune 100 have already done.

In order for application virtualization to come to fruition, each client machine needs to have a copy of VMWare ThinApp installed. A virtual machine is not required but is highly recommended on the machine used to package the virtual applications because a clean virtual machine ensures the virtual packages are free from bloated windows updates and excessive application data. This can be attributed to how VMWare creates packages. It works by taking a prescan and postscan of your machine and then comparing the results to create the package. If there have been any updates or changes to the system from pre to post stage, even those without the user's knowledge (windows updates,

java runtime updates, etc), then those will become inadvertently included in the new build, which will create a much larger virtual package. I was not able to work in a virtual environment for the reason stated above, but I did everything I could to ensure that all autoupdaters were turned off and only the essential application files were included in the package.

II. Details and Procedure

The process of virtualizing an application is straightforward, but many considerations need to be made before carrying out the operation. The first point to consider is the preferred method of deployment of the virtual application. There are two methods of deployment for virtual desktop packages: streamed execution and deployed execution.

Streamed execution is a method whereby the virtual package is stored centrally on a host machine or server, and the client machine streams the application upon execution. This does not require the local caching of files, but it does require the client to have a constant connection to the internet.

The other method is deployed execution, which works by delivering the packages directly to end-point devices including desktops, terminals, workstations, and so on. The key advantage here is that the user does not have to be connected to the internet at all; just copy the virtual package to the local machine or a portable device and run it anywhere. The requirements for the LCS lean towards a deployed execution model, and so I decided to follow the framework of this method when packaging the virtual applications.

A. Steps to installing a clean virtual machine operating system

As mentioned previously, I was not able to obtain a separate license for a Windows operating system. However, this is a critical step in the process and I was able to carry out the steps necessary for creating a virtual machine aside from the step of actually installing the operating system. However, there are many guides and tutorials available from Microsoft's website detailing the procedure for installing any Windows OS. The reason a virtual machine is important is because application installers will skip files that already exist on the computer, and if this happens, the ThinApp package does not include them during the application capture process. This could result in the application failing to run on other computers where the files do not exist. Thus, a clean install is an absolute must.

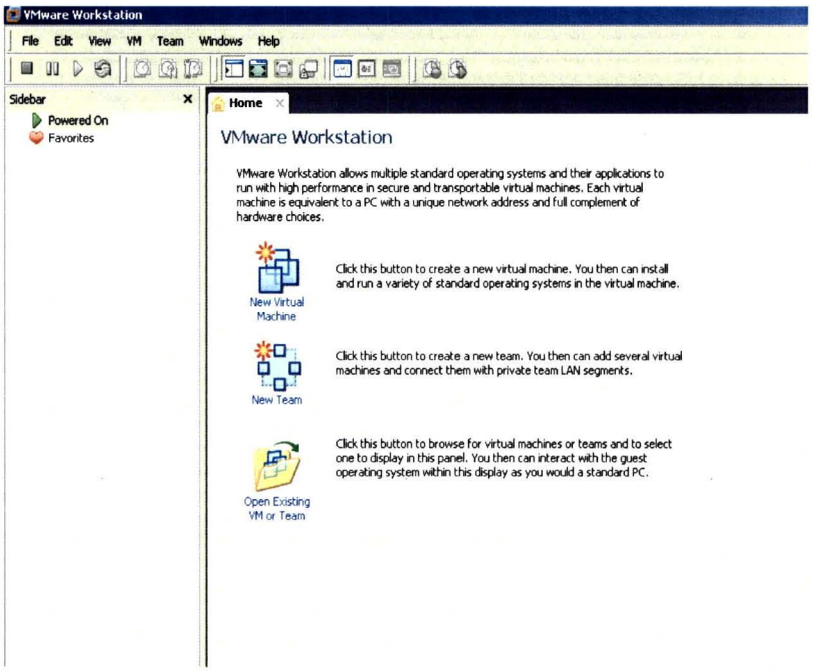

Figure 1. VMWare Workstation. *The home screen of the program used to create a clean, virtual windows machine, which is necessary to encapsulate the critical system files and registry keys in the virtual package.*

The first step is to obtain a copy of VMWare Workstation, which can be found by following the workstation link on the VMWare website. After installation, launch the program and click on the New Virtual Machine button under the Home tab (Figure 1). Make sure to choose the typical installation method as the advanced method is superfluous and complicated even for experienced technology professionals. After that, it will ask for the installer disc. If the OS disc is handy, then put it in the drive and specify the appropriate location. Otherwise, as in my case, I had to choose "I will install the operating system later" option. It will then ask what guest operating system is being installed, the virtual machine name (usually the name of the operating system followed by a number), and the maximum amount of hard drive space allocated for the virtual machine; these options are arbitrary and will depend on the user's preferences. After these

steps, a box will display the configuration settings and a finish button to finalize the process. Other than the installation process for the operating system, there is nothing else to do.

B. Steps to Virtualizing an application with VMWare ThinApp

Naturally, the first thing that must be done to virtualize an application is to install the program VMWare ThinApp, which can be found at http://www.vmware.com/products/thinapp/. The process to install ThinApp is the same with any other program: just download and run the executable, leaving the default values intact. Once the program is installed, it is time to launch it.

With the program running, the next step is to take a prescan of the system (Figure 2). A prescan is simply a baseline image (snapshot) of the machine before the application to be captured is installed. The capture process stores in a virtual file system and virtual registry the differences between the prescan and postcan images, which is used to create the package. This is where it is critically important to ensure your machine is completely clean and contains only the most basic version of the operating system installed devoid of any updates. Otherwise, the setup capture process may miss some files that are required to run the program. The entire prescan process should not take very long if your system is clean and limited only to the essential operating system components.

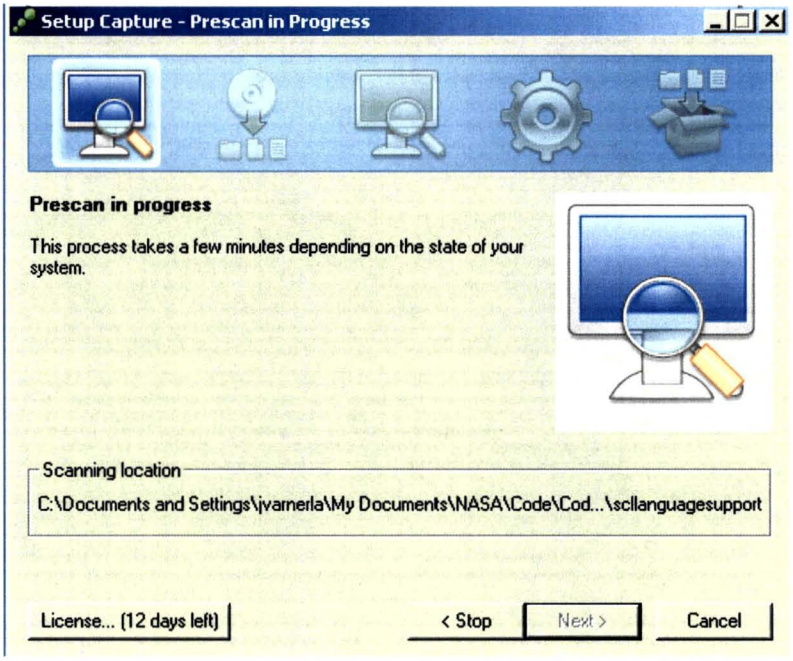

Figure 2. System prescan. *A snapshot of the system prescan that is used to create a baseline of the current status of the machine before installation.*

After the prescan is complete, the wizard will pop up with a window indicating that it is time to install the application. If the application is one which requires the machine to reboot, simply relaunch VMWare ThinApp after the machine boots, and it will resume the virtual packaging operation where it left off. The important note here is to make sure the application being installed is properly configured for the user. For instance, if Microsoft Word 2007 is being packaged and the desired font is Times New Roman and font size 12, then this needs to be set immediately after installation and before deploying the virtual package to the users. This is due to the fact that everything which is done between prescan and postscan is completely captured in the virtual package. Moreover, if the program does require an additional component to function, such as .NET framework 3.0, then it is advised to install this update before running the postscan. It may sound antithetical to the advice offered initially about using a clean virtual machine, but that advice is only relevant to nonessential updates; if there is a required runtime component for an application to run, for example Java FX for Netbeans, 6.91M, then by all means install the update before packaging the application to eliminate the additional step of installing the component on every machine that runs Netbeans.

The final step to virtualizing an application is to choose the entry points. An entry point is simply an executable that will launch a particular application. For example, if Firefox is the program being captured, it will most likely create entry points for firefox.exe and firefoxsafe.exe. Furthermore, virtualizing Office 2007 will create entry points for each program installed with the suite: word, excel, powerpoint, and so on. Naturally, it is prudent to select entry points for each of these programs. In the case of Firefox, if your organization doesn't foresee the need to run the program in safe mode, then it is suitable to select only the basic firefox.exe entry point. Once the entry points are selected, click the build button and the process of virtualization will commence.

The resulting projects will be stored in the default location C:\Program Files\VMware\VMware ThinApp\Captures. Within each of the project folders will be a bin folder containing the virtualized executable. The final point to consider in this process is how to deploy updates for the applications.

C. Updating virtualized applications with VMWare ThinApp

Creating virtual applications is a critically important step in the process, but equally important is the ability to rapidly deploy updates to extend functionality to the users. At this point, it is necessary to decide whether the responsibility to update applications resides with the user or the administrator. There are three different methods to choose from for packaging new updates: Recapture, Sandbox Merge, and Post-Capture.

Recapture involves running the setup capture process again for the purpose of incorporating the new updates in between the prescan and postscan snapshots of the Setup Capture. As an example, to capture the updated version of Microsoft Word, run the prescan, install the program with the newest service pack and/or hotfixes, and then run the postscan to build the new package.

Sandbox Merge consolidates updates from a sandbox into an existing project directory. To use this method, launch the virtualized application on a new, clean virtual machine. Run the update on the new machine, which will place the new files, registry, and configuration changes into the sandbox of that workstation. Then run the sbmerge utility to merge the changes from the sandbox of the capture machine into the existing project directory, and finally rebuild the package to incorporate the changes.

Post-Capture is a method of incorporating updates by manually placing folders in the appropriate directories of the capture, manually editing the registry files to include changes, and editing the package.ini file to change configuration settings. The package must be rebuilt with the build.bat utility, but running setup capture is not necessary.

Now that the updated package has been created using one of the above three methods, the administrator must determine how to deploy the new package. There are three methods to choose from: Package Replacement, Side-by-Side Update, and Application Sync.

Package Replacement is very easy to implement. Simply distribute the new executable to the user making sure the exact name is intact, and overwriting the existing package. The user must stop all traces of the current program for this to work.

Side-by-Side Update works by placing the new application package in the same directory as the original package and incrementing the filename extension to an integer number. New updates can be placed in the directory with extensions .2, .3, .4, and so on. The application automatically launches the file with the highest numeric extension. However, it is imperative to leave the original executable file in the directory to serve as a pointer for the new update. This is also useful because it does not require the user to terminate any process that is running. The update will simply pick up when the program relaunches.

Application Sync provides updates to unmanaged machines (deployed execution only) that connect over networks at least some of the time. It works by querying a web server to check for an updated version of the package. If a new version is found, the differences between the existing and new package are downloaded and used to build a new virtualized package automatically. The only stipulation is the user must have sufficient privileges to perform the update.

Ultimately, the best method is the one most suitable to the application in question. If the application has frequent updates, then AppSync is probably the best method. If the application has occasional updates which the user will be responsible for, then either Side-by-Side or Package Replacement will suffice; the decision will depend on the idiosyncrasies of the individual user.

III. Results

The results of implementing virtualization are more long term in the sense that it takes time to see the benefits of reduced cost and efficiency in the enterprise environment. However, for my project, it is sufficient to look at the ways in which virtualization makes the process of installing, running, and configuring applications better. First, the overall size of the application is smaller and more streamlined. Instead of dealing with large folders with several more subfolders and files within the root structure, virtualization allowed for the consolidation of all the data in a single dat file and an executable. Moreover, the total size of the virtual package is only slightly larger than the entire program directory of the installed application (875MB vs. 713 MB), and this package can be distributed to multiple machines without the need for additional installation or configuration. Keep in mind that the total size comparisons are misleading due to the fact that the virtualized application only needs to be created once for distribution, whereas the regular application has to be installed on each machine that requires it. Thus, the more machines that run Netbeans, the more disparate the size requirements become. And although the programs I worked with are not large enough to see the immediate performance benefits, in an enterprise environment where the machines were running very demanding and complex programs, the improvements in speed and execution time would be readily apparent

due to the way that ThinApp reduces page file usage and increases sharing between multiple instances of the same program.

Additionally, another important benefit is time. To install Netbeans 6.91M on my desktop machine, which has very sufficient processing power, it took approximately two minutes and 31 seconds. But to completely virtualize Netbeans from start to finish, it took a mere one minute and 43 seconds to finish – a difference of 47 seconds, or about 31% less time. Now consider that this scenario only accounts for one machine and imagine how many machines an organization would need to run Netbeans, and it becomes quite evident that ThinApp can save administrators vast amounts of time and prevent myriad headaches due to installation conflicts and issues.

IV. Conclusion

From the extensive research I conducted and the results obtained from virtualizing desktop applications, it is evident that this technology could prove very useful for the LCS and NASA as an agency. In order for the potential of virtualization to be fully realized, there needs to be widespread acceptance and interest in implementing the technology. As demonstrated in this report, virtualization has the ability to greatly simplify deployment of applications, improve security and efficiency, reduce costs, and assist in easier administrative control, among other things. Every fortune 100 company has already completely integrated virtualization from the desktop all the way to the datacenter, and many federal and state government agencies have also jumped onboard. It is time for NASA to do the same.

Acknowledgments

Special thanks to NASA and the KSC for hosting the program, USRP for funding the program, my mentor Linda Crawford for overseeing the project, the members of the Application Services and Framework team for helping me along the way, Benita Desuza of the Education office who coordinated the internship, and everyone else at the KSC Education Office who made this experience possible.

References

[1]Black, A., "VMWare ThinApp: A Review of Best Practices and Design Consideration," *VMWare ThinApp Deployment Reference*[online database], URL: http://www.vmware.com/files/pdf/VMWare_ThinApp_Deployment_Guide.pdf [cited April 15, 2010].

[2]Chu, D., "ThinApp User's Guide," *ThinApp Documentation Database* [online database], URL: http://www.vmware.com/pdf/thinapp45_manual.pdf [cited April 17, 2010].

[3]Devine, S., "Introduction to Application Virtualization," *ThinApp Documentation Database* [online database], URL: http://www.vmware.com/files/pdf/thinapp_intro_whitepaper.pdf [cited April 12, 2010].

[4]Hutchinson, J., "Department of Defense to Roll Out Virtual Next Generation Desktops," *Network World,* Vol. 106 No. 11, 2010 pp. 5,6.