

Review: Prism Deploy 5
From New Boundary
Technologies
By Bob Kelly, AppDeploy.com



Setting Up

The first thing you must do when setting up Prism Deploy is to create a Prism Deploy "Channel." Channels are used as the highest hierarchal level in grouping clients for management. You choose a name for the new Channel and then select which computers you wish to add to the Channel. You can enter the name of the computers to add them manually or you may browse for them. The browsing process is a matter of selecting machines from a Network Neighborhood/Active Directory dialog. Here you may use the shift and control keys to select multiple machines, but with no information collected about the machines, you do not have the option of utilizing queries in this selection process. In a small to medium environment, this is not likely to be much of a problem but those of you in a large environment where you wish to selectively add computers to different groups based on hardware, location, operating system, etc. this may be seem like a tedious task. However, you should think of a Channel as a domain; groups are assigned within this Channel so creating multiple Channels is more likely a broad task that you would only need to perform at a domain level. We will discuss the options available for grouping machines *within* a Channel later.

Client Deployment

The next thing you will need to do, is deploy the Prism Deploy Client to the machines you have assigned to your Channel. You may choose one of two available methods for deployment: A direct installation (for Windows NT and later systems) in which the client installation is handled remotely from the console. The second, which will be required for any Pre-Windows NT clients on your network, is to create a file based installation.

For the direct installation, there is a checkbox where you may enforce the required reboot of the client machine. Careful with this, the installation and reboot was quick with no notification presented to target systems. Like most software requiring a restart, it is best performed after hours (or at least make sure everyone is well prepared.)

In the AppDeploy lab, all Windows 2000 and XP systems installed and rebooted quickly with no problems, returning a success status to the console.

The file option generates a Subscription File, which is an executable that will install the required client software for Prism Deploy. It may be sent via email, shared on a web page, network drive, etc.

When generating the Subscription File, you choose if the package to be generated should install, update, or uninstall the client software from the target system. It lets you choose configuration details such as the Channel to which the client should join, as well as options for asking the user before installing- and you may separately

choose if you wish a message to be shown *after* the installation is complete. It is nice to see so many options for the creation of this package; seeing so many options for the Subscription File deployment option shows New Boundary knows how to think deployment.

I did run into a bit of trouble with the installation of the Subscription File on the Windows NT client I was attempting to run it on. As it turns out, you must share the client files from where it was installed on the server (%ProgramFiles%\Prism Deploy\Client) and the setup does not perform this task for you. So if you like me and don't read directions until you get stuck- this is something you would need to read the directions to know.

With so many options for controlling the behavior of the Subscription File, I would like to see an additional option that allows you to control the message presented by the Subscription File. The message it does show is illustrated in Figure 1 below:

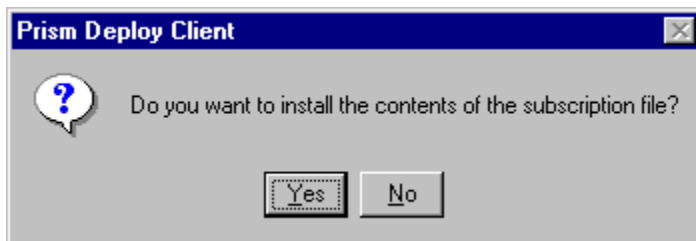


Figure 1: Prism Deploy Subscription File Installation Prompt

This message may be a bit of a confusing message for anyone not familiar with Prism Deploy terminology. A simple "Install Prism Deploy Management Client?" would be a bit friendlier for any users you instruct to run this file. And naturally, when installing a service of this kind, you do require administrative privileges on the system where it is being installed. On 9x clients you are okay, but if you want to use this deployment method on Windows NT and later clients, you will need to address this deployment issue. If you should attempt to install the client package without sufficient access, the installation fails silently presenting no messages to the user (even if the Subscription File is configured to run interactively).

Creating Groups

There are two kinds of groups you may create: Organizational Groups and Configuration Groups. Organizational Groups are populated manually by assigning it individual systems, NT groups or Active Directory groups. Configuration groups are dynamically populated based on the configuration rules you specify. When choosing to create a configuration group, you then specify what type of configuration group you wish to create. There are two choices: Predefined Groups and User-defined Groups. Predefined groups are based on a configuration values you chose, and subgroups are automatically created and populated based on your choices. User-defined configuration groups contain one or more subgroups populated using configuration rules you define. From here, each subgroup is defined by a separate configuration rule.

I created a Predefined Configuration Group to be based on the version of the operating system and named it "OS". Under this new "OS" group, subgroups for each

operating system on my network were created and populated in a matter of seconds as shown below:

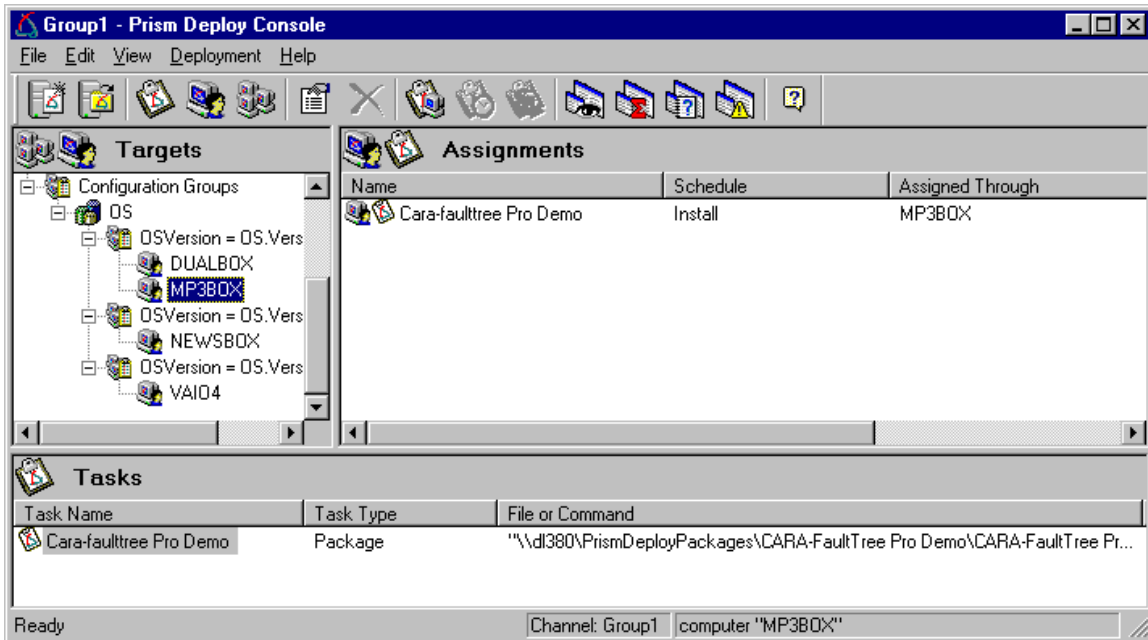


Figure 2: The Prism Deployment Console

This ability to create groups based on inventory data is a great one, but it does make you wonder why there is no capability to view any of these asset details for clients from the console. When viewing the properties of a client, no asset information is presented. However you can see the groups to which the client belongs. You can therefore create Configuration groups for any asset details you may wish to be aware of from the console. Figure 3 below, shows a configuration group being created based on the computer's domain.

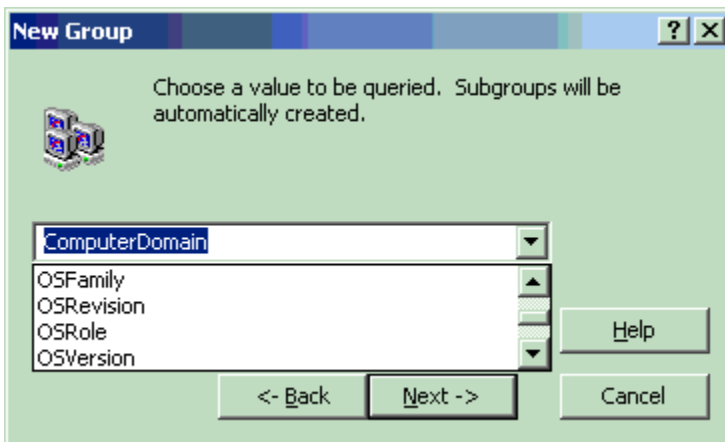


Figure 3: Configuration Group Query Dialog

Creating A Package

Creating a package is also quite easy. While the deployment capabilities are relatively new, the repackaging element of Prism Deploy (also available separately as

[Prism Pack](#)) has been around for some time. Formerly known as PictureTaker, the filename for the repackager is still named "Pictaker.exe." It uses a simple before/after snapshot process to assemble the delta into its own proprietary package format. You may either deploy it as-is, or you may easily convert it to a Windows Installer MSI package within the Prism Deploy Editor. The Prism Deploy Editor is shown in Figure 4 below.

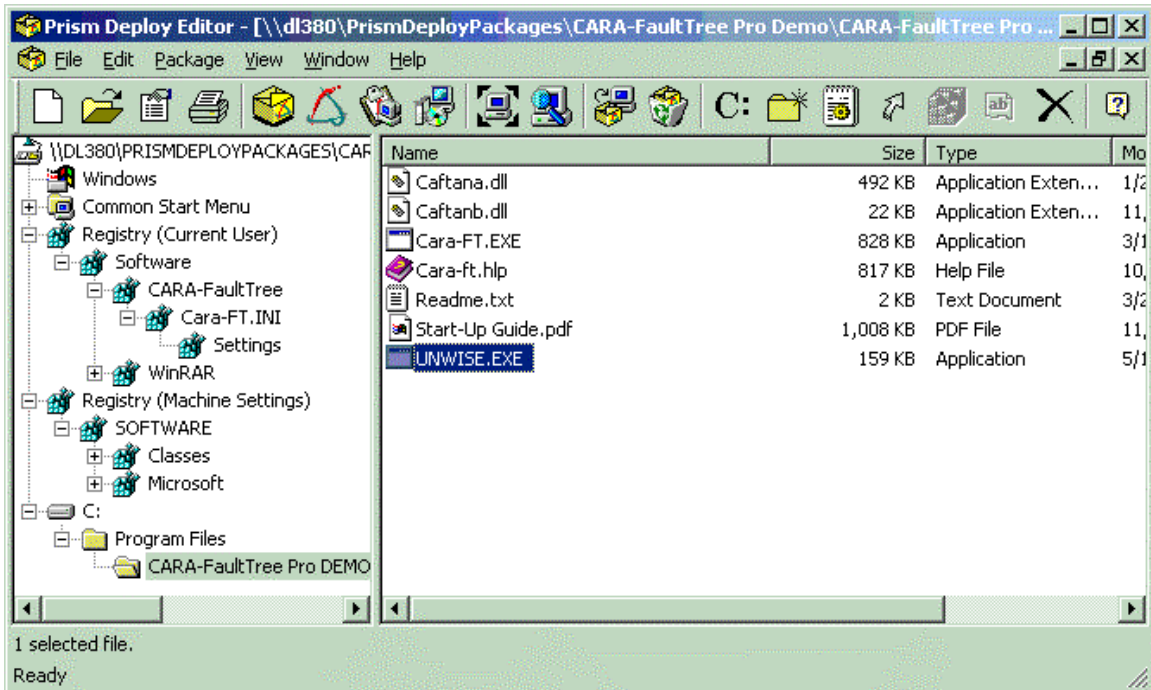


Figure 4: The Prism Deploy Editor (PictureTaker)

Prism Deploy also includes a Conflict Checker tool that allows you to scan for potential problems between the packages you have created. Now a more common feature in the repackaging suites, Conflict Checker was among the first to provide this capability. In the QA process, checking for conflicts between packages is a "must have" feature- and Prism Deploy has it covered.

Package Deployment

To deploy a package, you need to create a task. A task may consist of a package, a command line, or a script file. When selecting a package, it must be a native Prism Deploy package (.PWC). In testing, Prism Deploy packages installed quickly and without any problems. This is certainly the "preferred method" when using Prism Deploy to distribute software.

To push a Windows Installer package, you must use the "command line" action type when creating the task. You may then enter a command line installation. I used the following to push an MSI package using a command:

```
%windir%\system32\msiexec.exe /i
"\\dl380\Packages$\WinZip8\WinZip_PrismPack.MSI" /qb
```

The MSI failed to install due to a legitimate problem with the client (MSI Error 2738, a custom action could not run because the VB runtime was not installed). However, the console recorded the deployment as a success, despite the installation failure. The console is dependant upon the return code from the command executed. Because Windows Installer did not actually return 2738 as an error (this is an internal MSI error, not a Win32 error code as is returned by most calls) and instead returned a zero, the console saw the deployment as successful and recorded as such. Realizing this, I tried the following command line for the installation of the MSI Package instead:

```
%comspec% /c %windir%\system32\msiexec.exe /i
"\\dl1380\Packages$\WinZip8\WinZip_PrismPack.MSI" /qb
```

This returned an error as expected and the console showed it as an Alert instead of a Success in the reports with a description of "Command execution failed".

Reporting

It is plain to see that someone who knew what they were doing designed the reporting features for Prism Deploy. All too often generating a meaningful report is either difficult to automate, or very time-consuming to generate manually. Prism Deploy provides very helpful reporting, with a couple of views to choose from. Exactly what you get depends upon the scope of the report, which you dictate by selecting a computer, package or group prior to viewing the reports. When selecting a group, Reports look as follows:

Item	Assigned	Installed	Uninstalled	Errors	Installed + error	Pending
Cara-faulttree Pro De...	4	2	2	0	0	0

Figure 5: The Quick View tab

First is the "Quick View" tab (above, Figure 5). This provides a list of machines and totals for "assigned, installed, uninstalled, errors, installed with errors, and pending. The next report is shown on the "Summary" tab (below, Figure 6) and provides the status for each task for each machine it was assigned to as well as the time the status was recorded. The "Log" tab of Deployment Reports dialog provides a chronological summary of events and the "Alerts" tab lists only the failures that have occurred and summary information on each. Any of these reports may be exported to a TXT file. The text file is actually formatted as a comma-separated value (CSV) file, which may be easily imported to a database or viewed in a spreadsheet program (such as Microsoft Excel).

The screenshot shows a window titled "Deployment Reports" with a "Report Options" menu. The "Summary" tab is selected, displaying a table with the following data:

Status	Task	Computer	Time
✓ Installed	Cara-faulttree Pro Demo	NEWSBOX	11/5/03 12:23:32 PM
✓ Installed	Cara-faulttree Pro Demo	DUALBOX	11/5/03 12:21:02 PM
✓ Installed	Cara-faulttree Pro Demo	MP3BOX	11/5/03 12:10:05 PM
⊘ Pending	Cara-faulttree Pro Demo	VAIO4	

At the bottom of the window, there is a status bar with the text: "Deployment summary for the selected items Administrator time task 'Cara-faulttree Pro Demo'"

Figure 6: The Summary tab

Security

While not granular, Prism Deploy does provide limited security in its ability to password protect Channels. This is a single password that will need to be shared by anyone who requires access. It is for this reason that you may be more inclined to segment your network into more than one Channel. Going back to the beginning of this review, you recall that adding computers to a Channel is a manual process and cannot be based on queries as Configuration Groups can. Perhaps in a future release, you will be able to move computers between channels so that you can take advantage of the power provided by Configuration Groups to help automate Channel membership.

In Conclusion...

Prism Deploy is a very capable repackaging and deployment tool. There are few products that actually tackle both packaging and deployment, and it is nice to see Prism Deploy has taken on this challenge. The deployment capabilities have definitely improved since [the review AppDeploy.com published for the beta release of Prism Deploy 4.0 a couple of years ago](#). While you can generate MSI packages and deploy them as command line installations, this is not the "bread and butter" of Prism Deploy. Given that Windows Installer can often cause as many administrative problems as its complexity helps you to solve, there are some who have decided to steer clear of Windows Installer (when possible). If you are not banking on Windows Installer as your deployment technology of choice, Prism Deploy provides an excellent solution for package creation, conflict checking and deployment.

Bob Kelly
 AppDeploy.com
 11/05/03