MSBuild/TFS with Artifactory: Optimizing Build Automation and Continuous Integration in a .NET Ecosystem

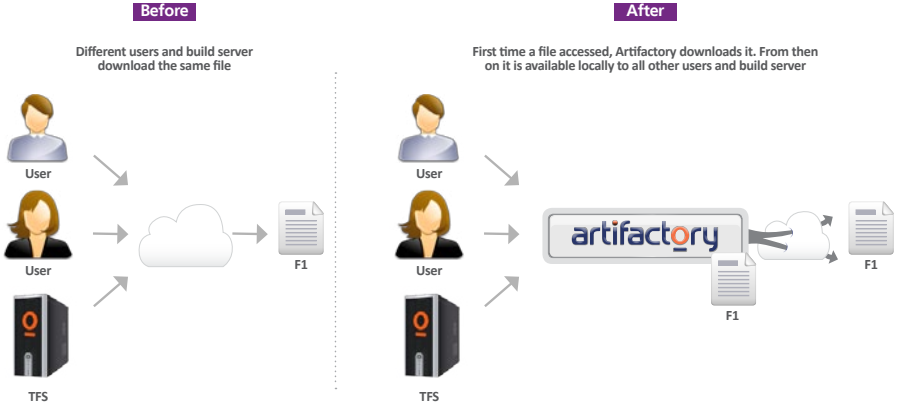
Build automation and continuous integration systems have become best practices in software development. A variety of tools and products are available to cater to the many different development platforms in common use today, but the de facto standards used when developing in a Microsoft .NET ecosystem are MSBuild with Team Foundation Server.

To complement its support for NuGet packages, Artifactory offers tight integration with these tools through the MSBuild Artifactory Plugin. Through this plugin, you can use Artifactory to resolve dependencies for your build, and deploy artifacts to your local build repositories.

This article presents some of the benefits of running your MSBuild/TFS builds through Artifactory.

Optimize Builds by Reducing Network Traffic

Artifactory is an intermediary between developers or build servers, and external resources such as NuGet Gallery. It gives you quick and consistent access to remote artifacts by caching them locally in a **remote repository**. A typical project may depend on tens if not hundreds of artifacts from external resources. For the server to build these projects, all remote artifacts must be available to the MSBuild environment. Downloading all those required artifacts may generate Gigabytes of data traffic on the network which takes a significant amount of time delaying the build process. By caching remote artifacts locally, the build process incurs much less networking, and is therefore much quicker.



Remote Repositories

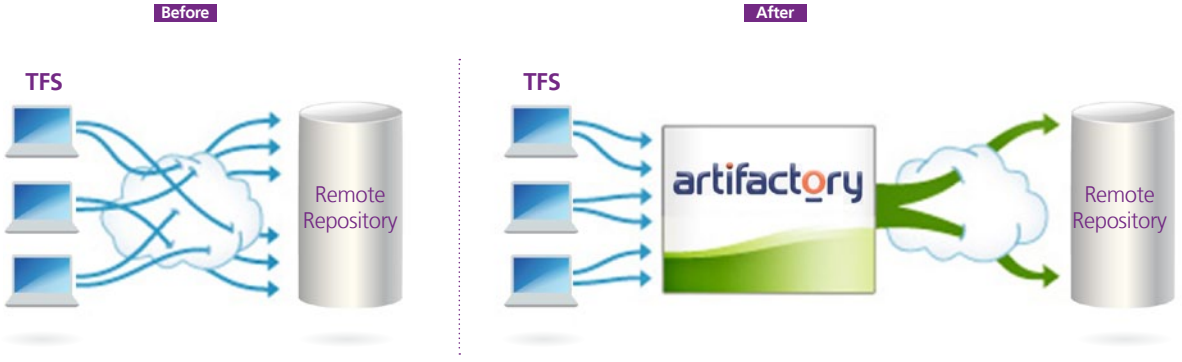
A remote repository serves as a caching proxy for a repository managed at a remote site such as NuGet Gallery. Artifacts are stored and updated in remote repositories according to various configuration parameters that control the caching and proxying behavior.

Learn more >

Reliable Access to Remote Artifacts

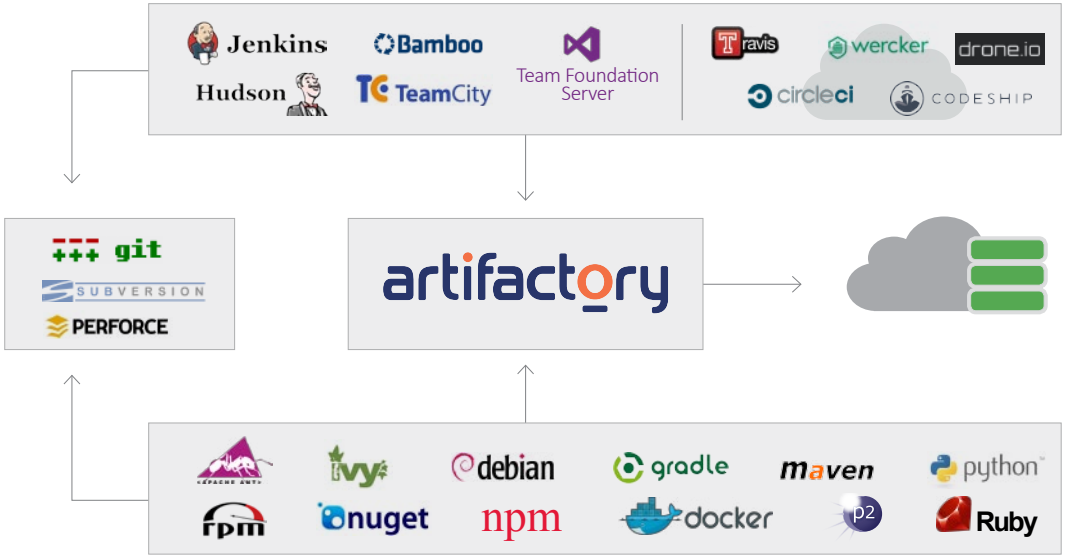
Since Artifactory maintains a local cache of remote artifacts on your local network, your builds are independent of external networking issues, and are not affected if a remote resource goes down. Any artifacts needed for a build are available from the local cache (i.e. the Remote Repository). Even in the extreme case that a remote resource ceases to exist altogether, any artifacts that have previously been downloaded are readily available from the cache.

Accessing Remote Repositories



Fully Reproducible Builds

Another big advantage of running builds in your MSBuild/TFS ecosystem through Artifactory, is the exhaustive build information that Artifactory generates. This information is uploaded to a local repository together with the build and includes specific artifact versions, dependencies, system properties, environment variables, user information, timestamps and more. With this information, it is easy to faithfully reproduce a build at any time, and with built-in "Diff" tools you can compare builds and therefore know exactly what changes were introduced from one version to another. These capabilities can be invaluable when trying to track down bugs that were reported in specific versions released.



Summary

By running builds in your MSBuild/TFS continuous integration ecosystem through Artifactory, you can overcome challenges presented by the network and the reliability of remote repositories, and enjoy the benefits of fully reproducible builds. Given that modern automated systems may run builds several times a day, this presents a significant boost to productivity. Since Artifactory is agnostic to the binary types that it manages, it can optimize build automation and continuous integration for virtually every packaging format in common use today.

For more information on how Artifactory can boost your organization's performance, please contact us at info@jfrog.com.