Success Story

JetBRAINS

www.jetbrains.com

TeamCity

FLUXYS

EXCELLENCE IN GAS TRANSPORT

As an infrastructure company operating on the North-Western European natural gas market, Fluxys wishes to contribute to security of supply and the functioning of the market in that region by promoting cross-border natural gas flows and transfers.

In Belgium, Fluxys builds and operates infrastructure for natural gas transmission, natural gas storage and terminalling of liquefied natural gas. To that effect, Fluxys Belgium and Fluxys LNG are appointed as independent system operators in Belgium. Their activities being of general economic interest, both companies offer competitive tariffs and make investments to strengthen the role of the Fluxys grid as a natural gas crossroads for international gas flows in North-Western Europe.

Industry:
Natural gas transmission

Fluxys SA
Avenue des Arts 31
http://www.fluxys.com/

How do TFS and TeamCity ownership costs compare? (e.g. for MSDN subscribers, etc.)

At first, TeamCity for us was an extra cost/investment because we already are running TFS. However, for our particular scenario of automating builds for consumables delivered through NuGet, it proved a much cheaper solution.

Much of the functionality we were looking for (e.g. NuGet package creation and publishing) is supported out-of-the-box, and adding custom scripts does not require us to hire a TFS expert to customize the workflow activities, impacting our entire build infrastructure.

TFS has a steep learning curve in terms of product: it's hard to accomplish any customizations without knowing anything about TFS. In TeamCity, you add build steps that suit your needs and customize them, which are much more flexible and require less to no knowledge about how TeamCity works or is built.

This is not really a TCO, but rather a description of how we got our ROI.

What were the top reasons to choose TeamCity over TFS?

We distinguished two different release cycles: consumables versus consumers. We mainly use TeamCity for consumables, such as shared libraries and reusable components. We are transitioning these repositories towards NuGet, and TeamCity has built-in support to create and push packages.

Furthermore, extending build configurations with custom actions through scripting is way easier to achieve in TeamCity due to its flexibility.

It would have been possible on TFS as well, but would have required much more effort, knowledge and money to accomplish the same in the same amount of time.

A nice benefit from choosing TeamCity is the ability to integrate with it using the TeamCity Service Messages feature. We already built a custom testing tool taking benefit from it, reporting test results in real “test artifacts” (interpreted by TeamCity using Service Messages) instead of simple strings in the TFS Build logs.

What were the major issues during migration from TFS to TeamCity and how were they solved?

We didn’t do a full migration: TFS and TeamCity are running side-by-side. The goal is to have all consumables created by TeamCity, consuming applications which require different testing approaches will still be built on TFS (because of a dependency on Visual Studio Test Manager, hence also on the TFS Build Agents for stamping the builds and having them available in the TFS Data Warehouse).

For consumables on TeamCity, we needed integration with TFS Work Item tracking. We found a plug-in for TeamCity that does it, but integration isn’t as nice and fluent as we’ve seen for Git and Mercurial repositories for instance. We feel this is something that could be improved and would make adoption of TeamCity easier for those who are using TFS Source Control today.

We also wonder whether having TeamCity mimick the TFS Build Agent behavior (custom build step?) and report back to the TFS Data Warehouse might make it possible to have integration with Test Manager (and thus allow for consumers to be built on TeamCity)?

Our build agents don’t have internet access. We also use a corporate proxy. The NuGet plug-in didn’t deal with this very well, so we logged issues on the TeamCity online issue tracker. We’ve seen most of them fixed. However, uploading a version of NuGet.exe to the build agents through the TeamCity admin UI is a bit cumbersome, because it still checks for available versions of NuGet.CommandLine package on nuget.org before it times out and shows the upload button.

Do you still use TFS as an issue tracker or version control? If not, what are the current choices?

Yes, we both use TFS Source Control and Work Item tracking. We have a lot of extra (custom) tools in the organization built on top of TFS that pull information from the Data Warehouse (reporting, tracking, planning etc). This won’t change without a huge amount of effort, both on the technical level as well as on the project management level.