Contribution ID: **3** Type: **Poster**

Moving away from ETICS... to Jenkins, or how I learned to stop worrying and replace ETICS with a 300-line script

Monday, September 16, 2013 9:00 AM (8h 30m)

Description of Work

CESNET's solution relies to a great extent on work already done within the EMI project, where a source package for each component is constructed as a specific target in the component's Makefile, and deployment + functionality testing is carried out in CESNET's on-demand virtualized infrastructure.
Since Jenkins does not natively support the use of packages and repositories, we rely on a single-purpose script to build binary packages in a mock or pbuilder environment. This script has been developed during EMI for testing purposes.
The overall process is inspired by that used in ETICS or, for that matter, EPEL. Source packages are passed to the build system, which produces binary packages built in a chroot environment. In ETICS (since EMI-3) as well as in EPEL, however, source packages are created outside the build system and only then fed to its input. Jenkins, on the other hand, is used also to make source tar balls, src.rpm and dsc/tar.gz packages –a single set for each component version.
Jenkins itself is used to initiate and manage the builds, and to organize results, namely to generate repository metadata for build artifacts. Resulting repositories, available directly from the Jenkins machine, can then be used in deployment tests. Separate build repositories for each product are sufficient for the purpose of deployment and functionality testing. However, if required, Jenkins could also be used to maintain global development, testing and release repositories with fresh versions of all products.
Each successful build in Jenkins is followed by an automated deployment and functionality test. The continuous testing infrastructure has already been presented separately at the EGI Community Forum 2012 (see Mass Testing of EMI Products in Czech NGI's Virtualized Environment). A master testing job runs regularly in the national grid, reading information of new successful builds from Jenkins' RSS feed, and submitting deployment tests for newly generated packages. The autonomous testing scripts developed within EMI were reused 1:1, only Jenkins repositories are now passed to the autonomous testing scripts instead of the ETICS ones.
Obviously, the Jenkins-based solution is lightweight when compared to ETICS. There is no automated dependency resolution –that is achieved by a fixed build order of the packages. There is no fine-grained access control or dynamic build machine allocation. However, Jenkins allowed us to reuse tools and procedures developed during EMI, requiring no or very little modification, which makes it an interesting choice for other product teams finding themselves in a similar situation. Their methods of producing source packages may differ, but the essential principles (strict build of binary packages in a chroot environment) should remain the same.

Printable Summary

The middleware release process used within the EGEE series of projects and within EMI depended heavily on ETICS to provide centralized build, integration, repository and archiving functions. Although there was certain progress towards independence in the final era of EMI, in the end, each product team had to deal with the decommissioning of ETICS separately. The middleware development team at CESNET, the providers of four distinct grid products within the EMI stack, have adopted Jenkins –the continuous integration tool, also reported as the tool of choice by the Italian Grid –to provide integration functions missed after the end of EMI. This poster gives details of CESNET's solution, used to continuously build and integrate its four products, consisting, in total of more than thirty packages. The continuous integration process takes care not only of regular or release builds, but also of continuous packaging and testing.

Primary authors: DVORAK, Frantisek (CESNET); SITERA, Jiri (CESNET); SUSTR, Zdenek (CESNET)

Presenter: SUSTR, Zdenek (CESNET)

Session Classification: Posters display