O.9 The test farm and its evolution

Jean-Michel Beuken[1]

[1] *Université Catholique de Louvain, Institute of Condensed Matter and Nanosciences, Louvain-la-Neuve, Belgium.*

In order to keep the individual developments in line with the global objectives of the project, all contributions have to be periodically reviewed because every new development has a significant probability to break the correct behaviour of another feature. This concern has been addressed in ABINIT thanks to the setup of a test suite and a Test Farm [1]. It examines on-demand the tentative contribution of each developer. This test farm does not only build the latest contributions, but also runs the test suite and validates the results. Thanks to this tool, the contribution of each developer is validated before it is considered for merge in the the trunk. Our Test Farm is made of two distinct parts: a series of computer platforms providing various development and running environments, and BUILDBOT [2], a software development continuous integration tool which automates the compile/test cycle required to validate changes to the project code base. By automatically rebuilding and testing the source tree each time something has changed, build problems are pinpointed early, before other developers are inconvenienced by the failure. By running the builds on a variety of platforms, developers will at know shortly afterwards whether they have broken the build or have encountered portability issues.

Since 2009, the Test Farm has grown and evolved. Today, twenty-one slaves with a diversity of open source and proprietary operating systems, compilers, MPI libraries and numerical libraries are provided [3].

[1] Y. Pouillon, J-M Beuken, T. Deutsch, M.Torrent and X. Gonze, *Organizing Software Growth and Distributed Development: The Case of Abinit*, Computing in Science and Engineering, vol. 13, no. 1, pp. 62-69, (2011).

[2] https://buildbot.net

[3] https://wiki.abinit.org/doku.php?id=bb:slaves