CONTINUOUS INTEGRATION USING LABVIEW, SVN AND HUDSON

O. O. Andreassen, A. Tarasenko, CERN, Geneva, Switzerland

ICALEPCS 2013

ABSTRACT

In the accelerator domain there is a need of integrating and creating control applications in an easy and yet structured way. The LabVIEW-RADE framework provides the essential integration of these applications into the CERN controls infrastructure. Building and distributing these core libraries for multiple platforms, e.g. Windows, Linux and OS X, and for different versions of LabVIEW, is a time consuming task that consist of repetitive and cumbersome work. All libraries have to be tested, commissioned and validated. Preparing one package for each variation takes almost a week to complete. With the introduction of Subversion version control (SVN) and Hudson extensive continuous integration server (HCI) the process is now fully automated and a new distribution for all platforms is available within the hour.

DEVELOPMENT METHODS

Through agile methods, tasks and projects are split into smaller increments that require minimal planning. Every iteration involves a small cross-functional team working on all disciplines: planning, requirement analysis, design, coding, unit testing and acceptance testing. At the end of the iteration, the product or result is demonstrated to the stakeholders, minimizing risks and giving room for fast changes and adaptations.

The Generic Agile Concept
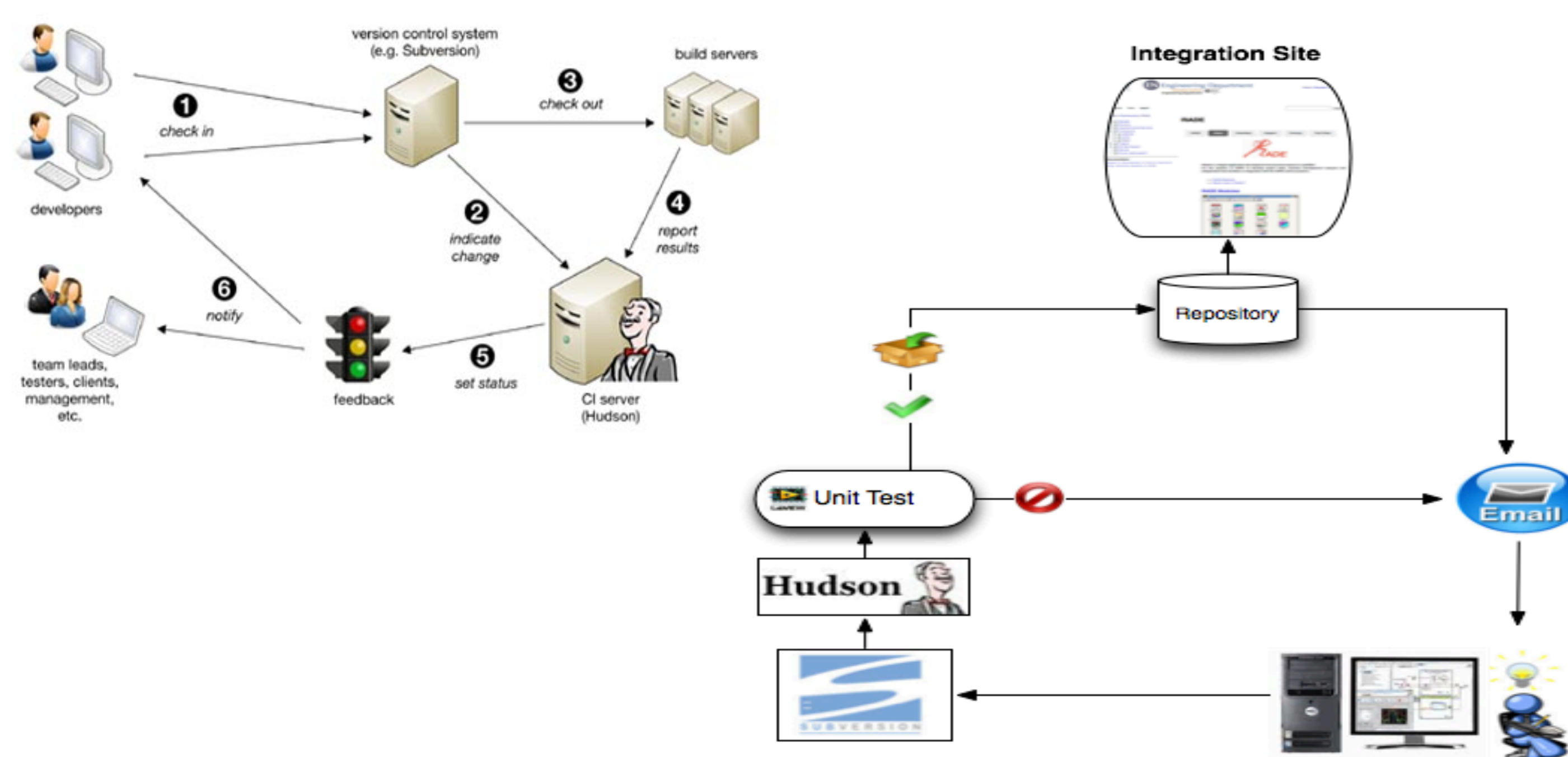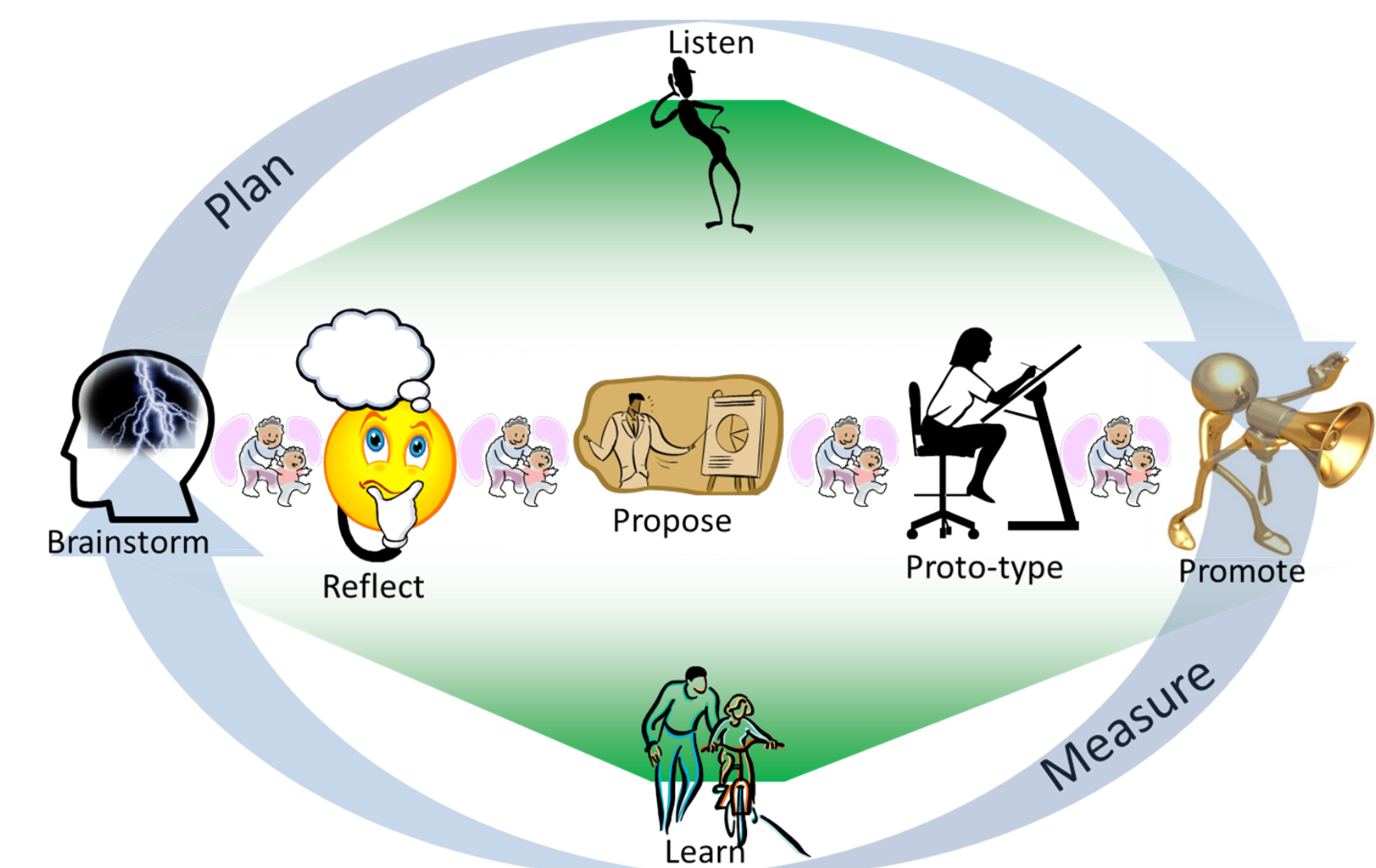

CONTINUOUS INTEGRATION PRINCIPLES

CI is a software engineering practice where small or isolated changes are immediately tested and reported on when they are added to a larger code base. Therefore if a defect is introduced in the code base, it can be identified and corrected without delay. In addition CI software tools can be used to automate testing and to automatically generate documentation. When used properly, continuous integration provides constant feedback on the status of the software and its defects are detected early on in development. In addition and as side benefit the defects are typically smaller, less complex and easier to solve.

TOOLS SELECTION

The CI engine has to:
- Be compatible with the existing SVN repository
- Be able to execute any programming language or script
- Run on all our main operating systems (Linux, Windows and OS X),
- Report any issue(s) encountered automatically.
- Be easy to maintain
- Have a plugin based and flexible pool of tools available

| Name	Platform	SVN SCM support	Mail Support	Other Builders
Bamboo	Servlet	Yes	Yes	cmd. line, Bash
Hudson	Servlet Container	Yes	Yes	Most scripting tools
CControl	Cross Platform	Yes	Yes	catch-all 'exec'
Continuum	JDK	Yes	Yes	====
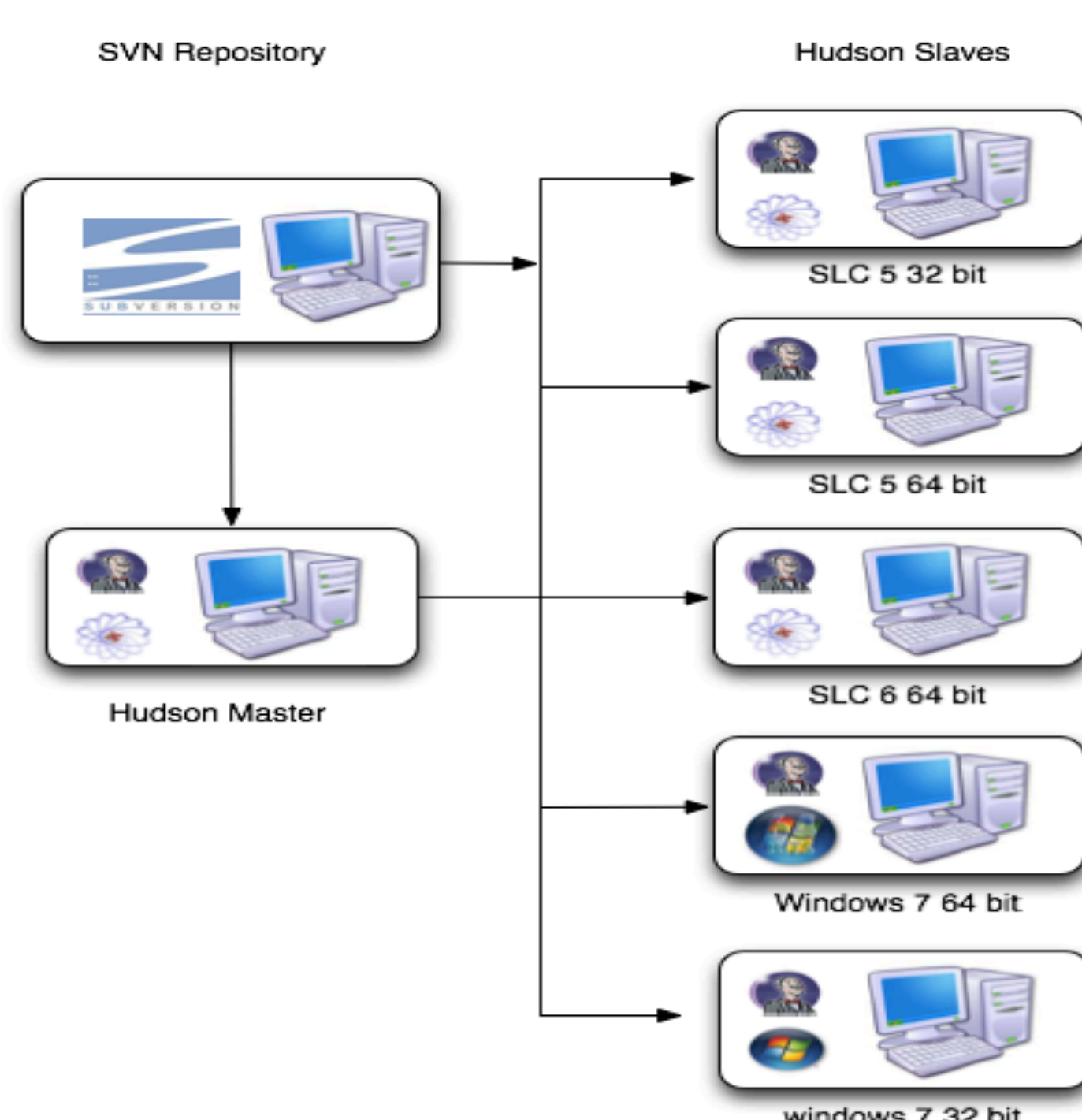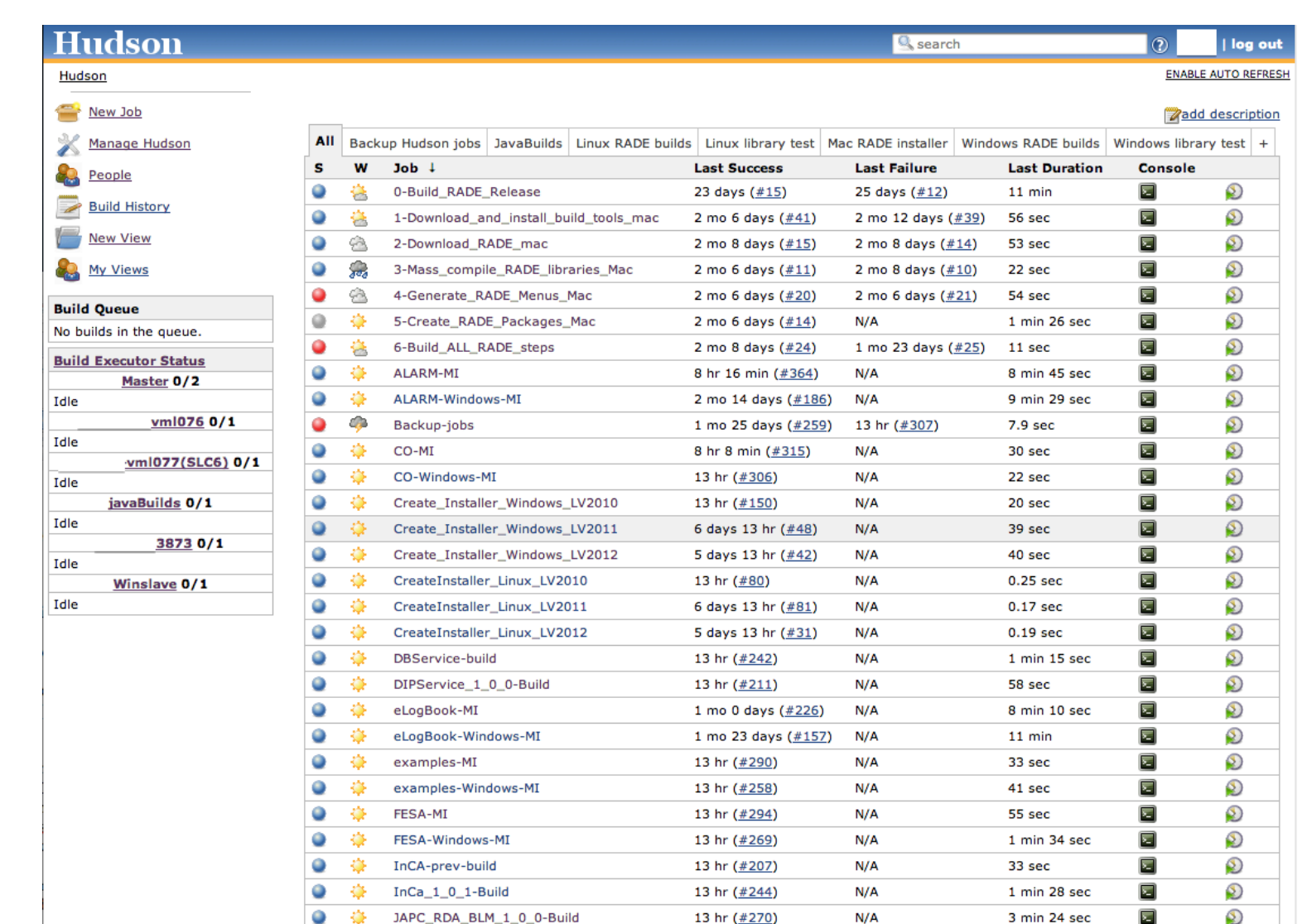
BUILD ENGINE

The RADE LabVIEW package has to be compiled for 5 different platforms, 32 and 64 bit operative systems and consist of many different build types. All the builds are orchestrated trough a main instance of Hudson, and built on 5 slave nodes.

GRAPHICAL BUILD ON A HEADLESS MACHINE

One of the main challenges with the RADE CI engine was building a graphical tool in a non graphical environment. TigerVNC provided a good solution for this
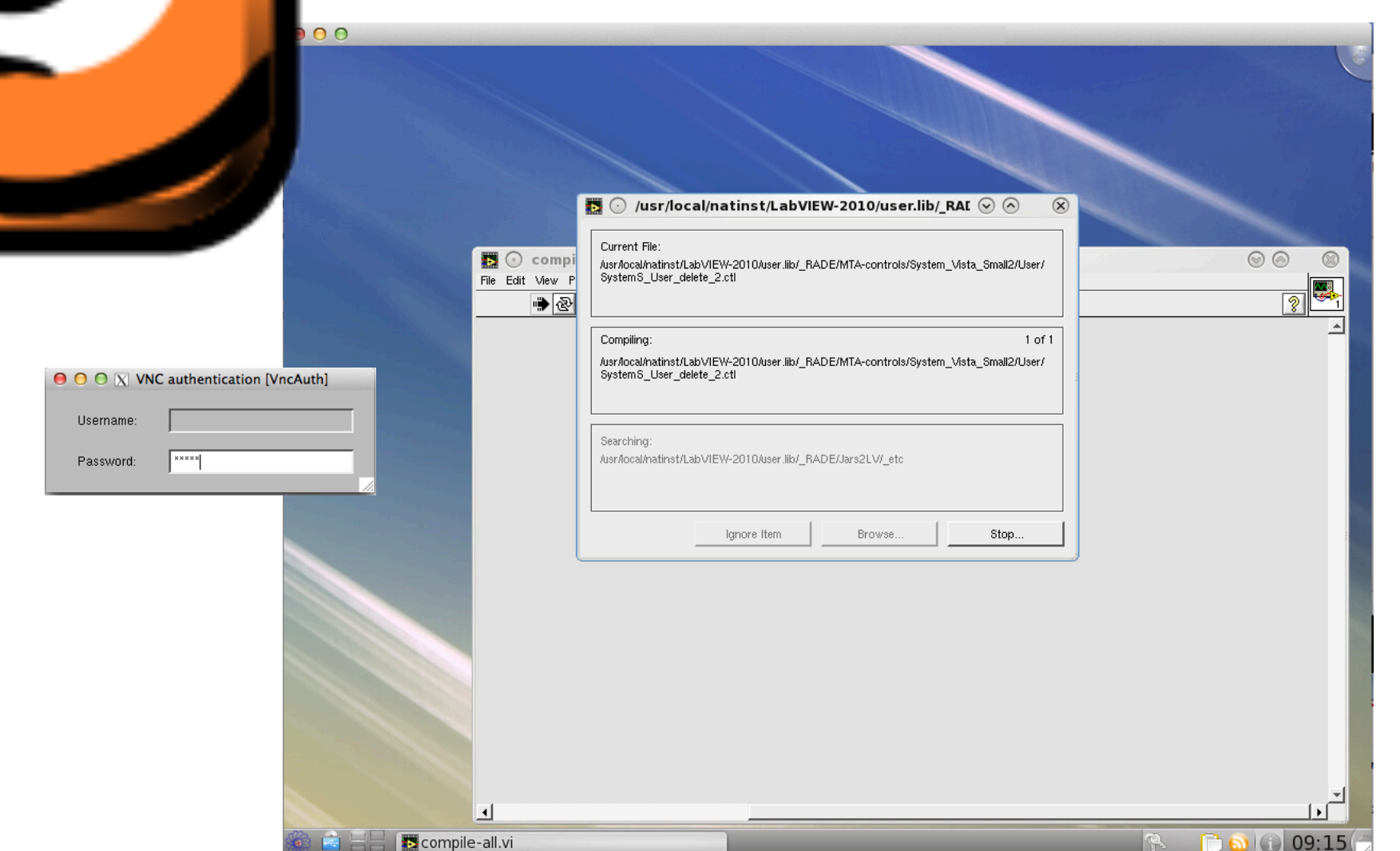
CONCLUSION

The continuous integration and automation of the RADE framework has greatly improved the delivery time (from 1 week to 1 hour), quality and frequency of new software. It has made the framework more robust through preventive testing and fault elimination before distribution. Automating these tasks add some maintenance overhead for the build environment itself, nevertheless the advantages and overall time saved makes it worth the effort.

Industrial Controls
Engineering Department

EN Engineering Department