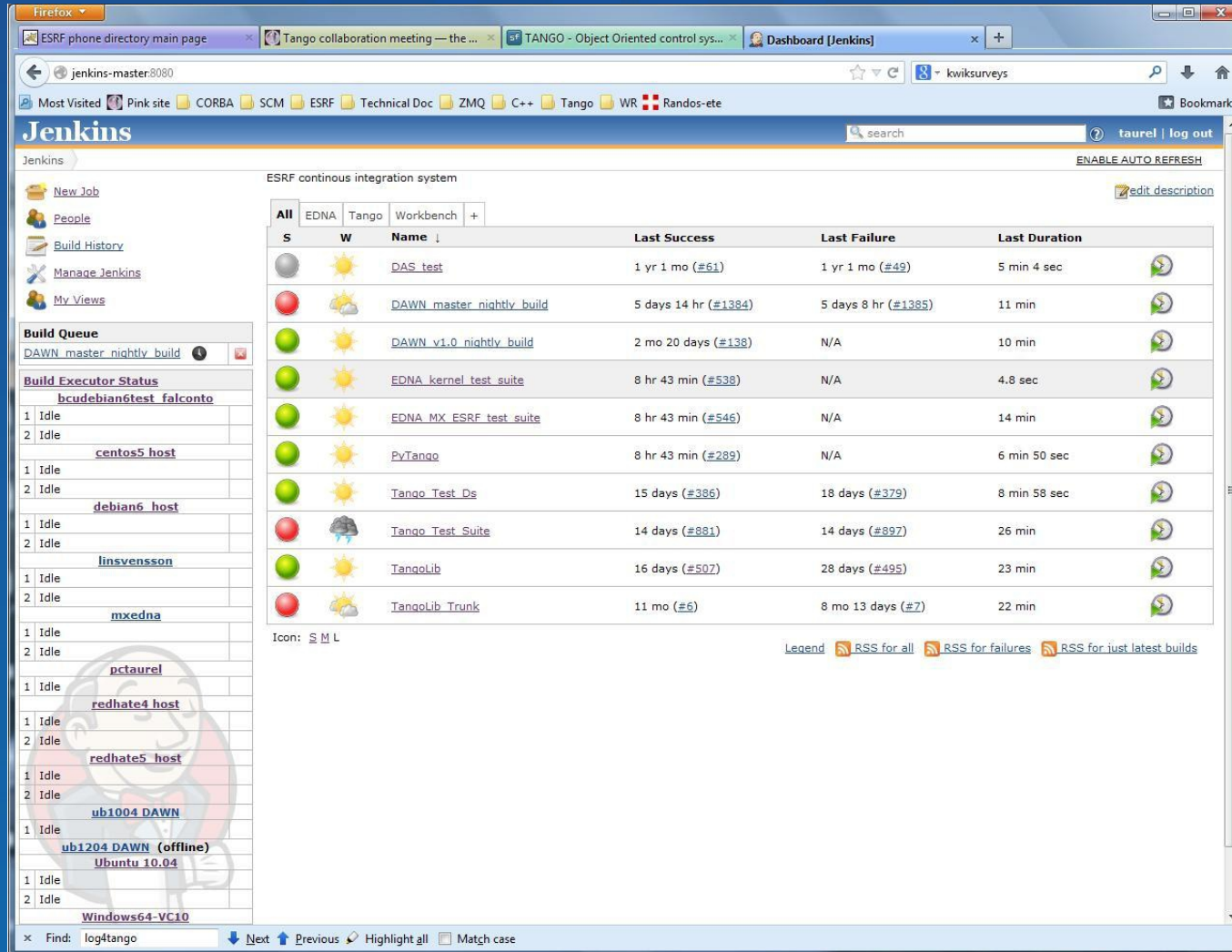# Continuous Integration @ ESRF

- Not widely used at ESRF
  - Only for 3 software
    - EDNA
    - Dawn
    - Tango

- Jenkins is the ESRF CI tool
  - Thank's to Vincent Hardion (Soleil – Max Lab)

Continuous Integration @ ESRF

Continuous Integration @ ESRF

- 4 projects related to Tango
  - Tango library
  - A test device server
    - Downstream project of Tango lib
  - The Tango test suite
    - Downstream project of the test DS project
  - PyTango

- The 4 projects use SourceForge Tango-cs SVN repository

Continuous Integration @ ESRF

- Tango library (developed with Ubuntu 12.04)
  - Compiled for 7 OS
    - Debian 6 – Ubuntu 10.04 – Windows VC10 - Red Hat 5 – Red Hat 4 – Centos 5 – Windows VC9
  - Compiled both in archive/shared and debug/release mode
    - 28 different libraries !!!
  - Resulting libs available for ESRF users (but at their own risks)

- The test device server
  - Compiled for the same OS using both shared and archive libs
  - 14 binaries

Continuous Integration @ ESRF

- The test suite
  - Compiled and run for the 7 OS
  - Project considered as a success if all tests are OK

- 72 minutes (23 + 9 + 40) needed between the start of Tango libs compilation and the end of last test

- PyTango
  - Compiled for 4 OS
    - Windows VC10 – Debian 6 – Red Hat 4 – Red Hat 5

Continuous Integration @ ESRF

- If required, other institutes can set-up their own Jenkins and compiled Tango lib for their platform
  - Tomasz has defined a framework to easily extend what exist today
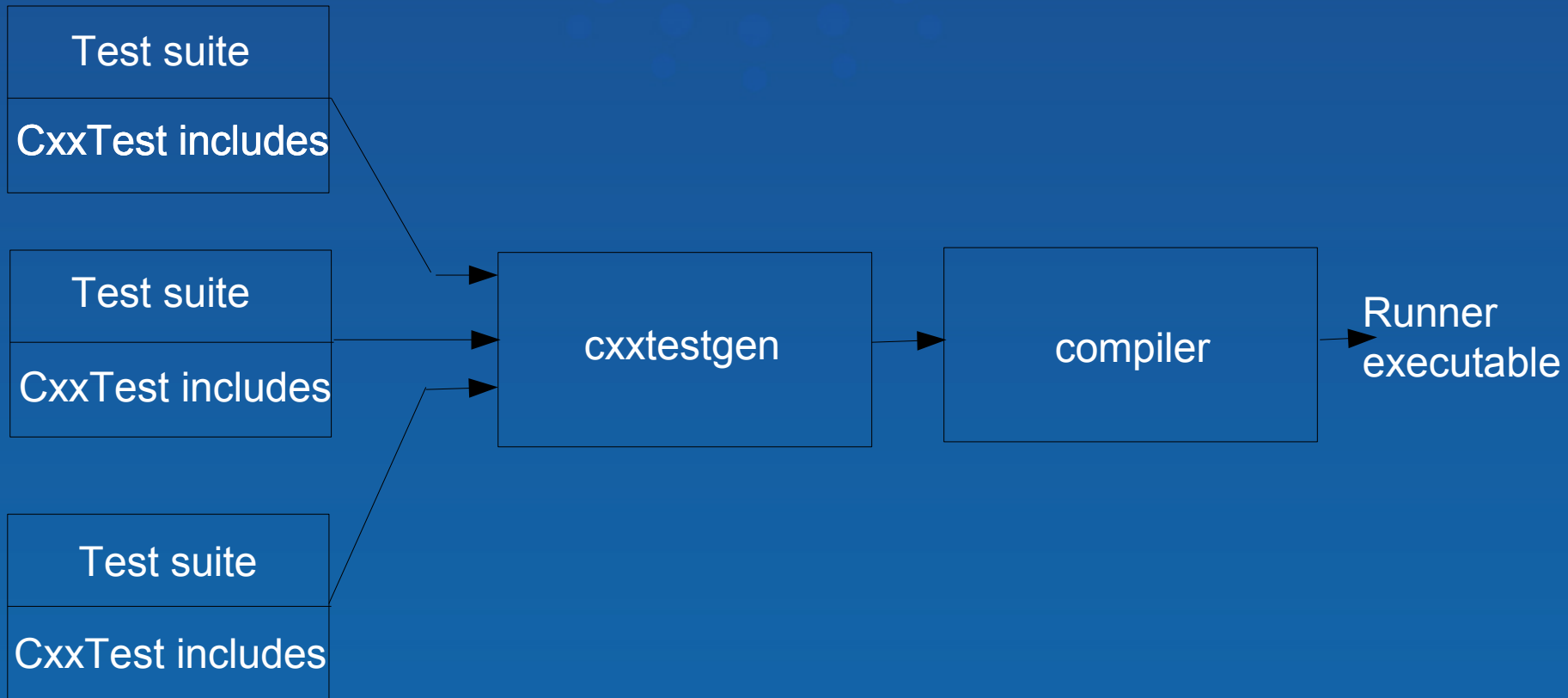   - Alba uses it

The Tango test suites

- Tango test suites built with two kind of test suites
  - Home made suites
  - Using a unit test framework

- Tomasz has selected CxxTest as unit test framework (http://cxxtest.com)
  - From their web pages:
    - CxxTest is a unit testing framework for C++ that is similar in spirit to JUnit, CppUnit, and xUnit.

- CxxTest implemented as include files set

The Tango test suites (CxxTest)

- A CxxTest test suite is a C++ class (inheriting from CxxTest ::TestSuite) with
  - Constructor
    - Define your test suite argument(s) and connect to your device(s)
  - Destructor
    - Restore your device(s) state
  - Set of methods defining each test cases
    - One method for each test case
- Use CxxTest provided macros to check your results
  - 18 differents assertion macros
    - TS_ASSERT() - TS_ASSERT_EQUALS() TS_ASSERT_THROWS_NOTHING() - …
  - If one assertion fails, the test case is reported as a failure

The Tango test suites (CxxTest)

Test suite

CxxTest includes

Test suite

CxxTest includes

Test suite

CxxTest includes

cxxtestgen

compiler

Runner
executable

The Tango test suites (CxxTest)

- None of the testing frameworks studied by Tomasz cover all requirements
  - But CxxTest is "easily customable"
  - Tomasz has added
    - Command line argument management
    - Runing test unit or test suite in loop
    - Restore points
    - Exception printing
- All customization in one file
  - TangoPrinter.h
- 32 CxxTest test suites have been written

The Tango test suites (home made)

- Other test suites are:
  - Set of small C++ code (a single main function) with classical assert() function call
- A shell script which chained all these little executables
  - Script aborts if one of the executable returns -1

- We still have
  - 13 home made test suites for asynchronous calls testing
  - 22 home made test suites for event testing
  - 50 home made test suites for miscellaneous features test

The Tango test suites

- Old and new system
  - 117 test suites!

- All these tests use devices implemented in a very specific Tango class with
  - 104 commands
  - 98 attributes
  - Behavior of all these commands and attributes well defined
    - Possible to code assertions on command execution or attribute read/write

The Tango test suites

- Everything available from tango-cs on SourceForge
    - api/cpp/cpp_test_ds
        - For the specific Tango class
    - api/cpp/cpp_test_suite
        - For all test suites (old and new)
        - With the customized CxxTest