Enabling Grids for E-science in Europe

www.eu-egee.org

JRA1 all-hands meeting, 28-30 June 2004

Unit testing coordination and interface testing.

David Collados
Testing Team

EGEE is a project funded by the European Union under contract IST-2003-508833

Contents

- Unit testing coordination.
- Interface testing in the build system.

Unit Testing Coordination 1/5

Enabling Grids for E-science in Europe

OBJECTIVES:

- Unify the Unit Testing (UT) procedures, formats and tools among testing activities in JRA1, JRA3 and JRA4.
- Gather information about the status and ideas of UT in those groups.
- Produce Unit Testing recommendations.
- Write “How Unit Tests should be written” chapter in the developers’ guide.

Unit Testing Coordination 2/5

Enabling Grids for E-science in Europe

CURRENT STATUS:

- Survey document in EDMS: https://edms.cern.ch/document/477813/1
- People seem to agree on UT tools. 
- And accept a common naming convention. 
- Only in Italy, people are producing unit tests by now. 
  - So practically, there is no integration of UT in the build.

Unit Testing Coordination 3/5

Enabling Grids for E-science in Europe

RECOMMENDATIONS:

- Unit Testing Tools: JUnit for Java, CppUnit for C/C++, Test::Harness for Perl and PyUnit for Python.
- Tests must be placed under “test” directory (SCM doc.)
- Reports must be placed under “report” directory.
- Naming convention:
  - for each “className” → “classNameTest” file.
  - for each “methodName” in my class → “testMethodName” in my “classNameTest”.
  - for Perl, “nameTest.t”
- Unit tests should be designed before the code, according to what software is specified (not observed) to do.

Unit Testing Coordination 4/5

Enabling Grids for E-science in Europe

OBLIGATIONS:

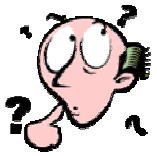
- WE need to enforce Quality Assurance, so
- YOU MUST PROVIDE UNIT TESTS FOR YOUR CODE.
  - Test coverage tools will be used to check acceptance.
  - **If UT fails ⇨ Build fails ⇨ No deployment in testbed.**

Unit Testing Coordination 5/5

Enabling Grids for E-science in Europe

ISSUES/DIFFICULTIES:

- XUnit frameworks don't produce XML reports. JUnit ANT task lets you easily format the output to XML. Do something similar for the other XUnit.
- No documentation: Interface specifications. Waiting inputs from PTF.
- How to retrospectively add unit tests to existing code?



  - Lot of code already written.
  - UT must be produced for it ⇨ **A HUGE EFFORT!**
  - Remember: absence of unit tests will make your build fail!!

Interface Testing 1/2

Enabling Grids for E-science in Europe

OBJECTIVES:

- Test external interfaces generated in the build.

  - WSDL documents: compare generated WSDL files with architecture specifications.

    * Waiting input from PTF.

  - Executables: script calls binaries with arguments and checks output.

    * Developing code to test WMS executables.

  - Libraries: cpp class calls library methods. We check if the linking is ok.

    * Developing code to test WMS libraries.

Interface Testing 2/2

Enabling Grids for E-science in Europe

DIFFICULTIES FOUND:

- No documentation: Difficult to find specifications for executables and libraries.
- No auto-generated documentation in the build by now.
- Delay in emails response.

Summary

Enabling Grids for E-science in Europe

- Agreeing on UT recommendations ⇨ Developers' Guide.
- UT must exist for old and new code ⇨ **HUGE EFFORT!**
- Interface Testing advances despite lack of documentation.