



LCG-SPI: SW-Testing QMTTest test framework

LCG AppArea meeting (16/07/03)

Manuel.Gallas@cern.ch

LCG/SPI

LCG Software Process & Infrastructure



SPI SW-Testing

- Test Frameworks
- QMTTest
- User support





- Aim:** to help developers:
- to produce code for testing
 - to run tests in automatic way

Requirements:

- Work with different languages (C++, Python, ...)
- Allow two ways of testing: "*test the output of the code*" and "*test inside the testing-code*".
- Provide a easy way to integrate existing tests.
- Easy integration with the Nightly Building System.
- Regression testing.
- Allow the creation of dependencies among tests
- Organize the tests by components or packages.
- Provide also a graphical interface for run the tests and examine the test results.

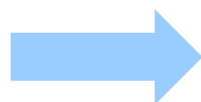
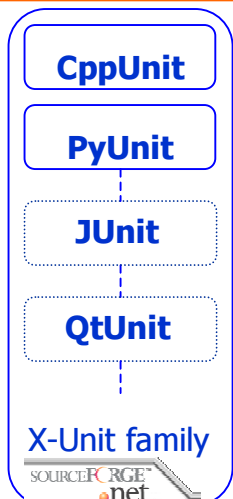
Our constrains:

- Avoid commercial software and licensing problems.
- Avoid "do it yourself solutions"
- Try to adopt commonly used open-source software.

Our inputs:

- Contacts within HEP-community.
- What is available as free open source code.





A simple test:

1. Subclass the TestCase CppUnit or PyUnit class
2. Override the method runTest().
3. When you want to check a value, call the (CppUnit or PyUnit) **ASSERT**(bool) and pass in an expression that is true if the test succeeds



```

xterm <6>
!!!FAILURES!!!
Test Results:
Run: 19  Failures: 2  Errors: 0

1) test: StringToolsTest::testToStringInt (F) line: 33 StringToolsTest.cpp
equality assertion failed
- Expected: 12345678
- Actual  : 123456789

2) test: ComplexNumberTest::testEquality (F) line: 22 ComplexNumberTest.cpp
assertion failed
- Expression: *m_10_1 == *m_1_1

PASS: test
=====
All 1 tests passed
=====
make[1]: Leaving directory `/home/user/mgallas/CppUnit-examples/newtest'
[pcitapi31] ~/CppUnit-examples/newtest >
    
```

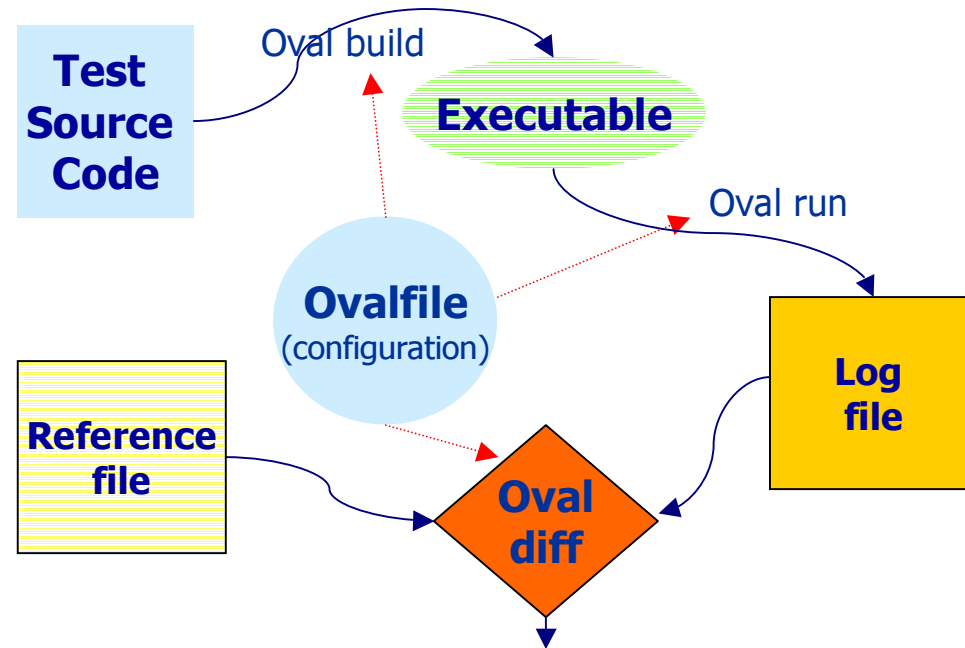
- Scope: Unit –testing
- The same “assertion style” in different languages.
- Provides:- The name of the test case that failed.
- The name of the source file that contains the test.
- The line number where the failure occurred
- Different platforms/compiler: (Linux/Solaris/Windows)



Oval:

- validation and regression
- used in CMS

- Can be use for Unit-test to Validation tests.
- Compare the output log file with a given reference file (Smart comparison of those lines which start with "[Oval]")
- It is possible to set different run environments.
- Can run external scripts and external binaries.



```

xterm <6>
[pcitapi31] ~/OVAL-examples/subtest1 > oval p
Prog1: build, run, diff.
Prog2: build, run, diff (DIFFS).

xterm <6>
[pcitapi31] ~/OVAL-examples/subtest1 > oval diff Prog2
===== diff =====
(DIFFS)
=== log #29 !~ ref #65 (>15%)
log: [OVAL] result = 1.2
---
ref: [OVAL] result = 1
=====
[pcitapi31] ~/OVAL-examples/subtest1 >
  
```





- Uses a graphical interface for creating and running tests
- Runs tests in parallel
- Organizes tests hierarchically
- Supports execution of a single test or many at once
- Records dependencies between tests
- Can be run in batch mode -> easy integration with the Nightly-Building systems

The screenshot shows a Mozilla browser window displaying the QMTest Test Results page. The page title is "QMTest: Test Results - Mozilla (Build ID: 2002100315)". The URL is "http://127.0.0.1:36526/test/show-results". The page content includes the LCG logo and navigation links for "News", "Help", and "Policies". A "Statistics" table is displayed, showing the following data:

Outcome	# of Tests	% of Total	# Unexpected	% of Total
FAIL	3	37.5%	3	37.5%
PASS	5	62.5%	0	0%
Total	8	100%	3	37.5%

Below the statistics table, a list of tests is shown with their outcomes:

- datasvc_cachemgrtest: PASS
- datasvc_datasvcctest: PASS
- datasvc_reftest: FAIL
- datasvc_retrievstest: FAIL
- pool_1_0_0_simplereader: FAIL
- tests_datasvc_crossreference: PASS
- tests_datasvc_inheritance: FAIL
- tests_filecatalog_functionality: FAIL

The screenshot also shows a "QMTest: Result Detail" window for the "datasvc_cachemgrtest" test. The outcome is "PASS". The "Cause" field is empty. The "Annotation" field contains the following text:

```

ExecTest.expected_stdout

===== unitTest_DataSvc_CacheMgrTest =====
run
Testing get, set and remove on Token methods
-----
Testing clearCache method
-----
ExecTest.stdout

Testing get, set and remove on Null Token methods
-----
[OVAL] Test result: 0 Errors
===== unitTest_DataSvc_CacheMgrTest =====
===== diff =====
    
```





Use in LCG AppArea projects:

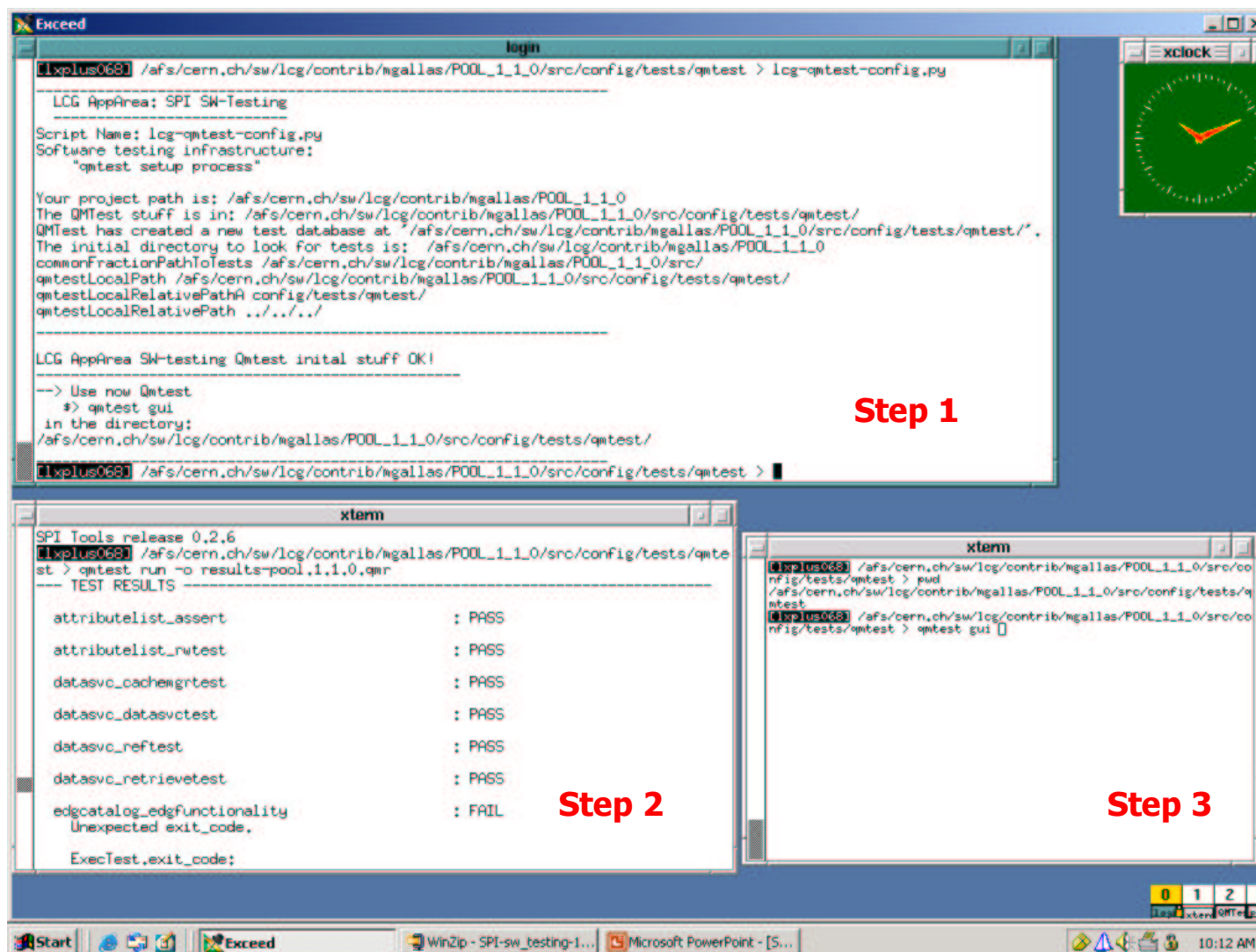
- All the QMTest configuration files will be created by a python script at once and later committed to CVS. Dependencies among tests, expected values and results could be also in CVS with the appropriate tag.
- These xml configuration files will be in → \$PROJECT/src/config/tests/qmtest
- Only when a new test subdirectory is added is needed to re-run the script
- To create the xml files we use the "lcg-qmtest-config.py" script . It will scan the whole project looking for the OvalFiles. The configuration files will describe "test cases" and "test suites"
- In each project release QMTest should be run and the test results saved:
`$PROJECT/src/config/tests/qmtest> qmtest run -o results_release_n_y_x.qmr`
- The test results can be browsed later using the graphical interface:
`$PROJECT/src/config/tests/qmtest> qmtest gui`

QMTest (at LCG AppArea) requirements:

- Oval ≥ version 2_15_3
- CppUnit 1.8.0 used with the SPI test driver.
- PyUnit used with SPI test driver.
- stderr output in use only by Oval, CppUnit and PyUnit

QMTest installed in the SPI external sw-service
 (Available in the SCRAM ToolBox ≥ LCG_10)





Step 1

```

Exceed
login
[lpplus068] /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest > lcg-qmtest-config.py

-----
LCG AppArea: SPI SW-Testing

Script Name: lcg-qmtest-config.py
Software testing infrastructure:
  "qmtest setup process"

Your project path is: /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0
The QMTest stuff is in: /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest/
QMTest has created a new test database at "/afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest/".
The initial directory to look for tests is: /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0
commonFractionPathToTests /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/
qmtestLocalPath /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest/
qmtestLocalRelativePathA config/tests/qmtest/
qmtestLocalRelativePath ../../..

-----

LCG AppArea SW-testing Qmtest initial stuff OK!

--> Use now Qmtest
$> qmtest gui
in the directory:
/afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest/

[lpplus068] /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest >
  
```

Step 2

```

xterm
SPI Tools release 0.2.6
[lpplus068] /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest > qmtest run -o results-pool.1.1.0.qmr
-----
TEST RESULTS
-----

attributelist_assert           : PASS
attributelist_rwtest          : PASS
datasvc_cachemgrtest          : PASS
datasvc_datasvc_test          : PASS
datasvc_ref_test              : PASS
datasvc_retrieve_test         : PASS
edgcatalog_edgfunctionality   : FAIL
  Unexpected exit_code.
ExecTest.exit_code:
  
```

Step 3

```

xterm
[lpplus068] /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest > pwd
/afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest
[lpplus068] /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest > qmtest gui
  
```


QMTEST xml configuration files:

Test suite

Test case

Results



```

Exceed
QMTEST xml configuration files:
xterm
results-pool.1.1.0.qmr
[lxplus068] /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest > ls
QMTest
attributelist.qms
attributelist_assert.qmt
attributelist_rptest.qmt
datasvc.qms
datasvc_cachemgrtest.qmt
datasvc_datasvcctest.qmt
datasvc_reftest.qmt
datasvc_retrieveftest.qmt
edgcatalog.qms
edgcatalog_edgfunctionality.qmt
edgcatalog_edglookupstest.qmt
edgcatalog_edgmeta.qmt
filecatalog.qms
filecatalog_systemtools.qmt
filecatalog_uriparser.qmt
integration_system.qms
mysqlcollection.qms
mysqlcollection_inserttest.qmt
mysqlcollection_readtest.qmt
mysqlcollection_updatetest.qmt
results-pool_1.1.0.qmr
rootcollection.qms
rootcollection_writetest.qmt
rootcollection_readtest.qmt
rootcollection_updatetest.qmt
tests_collection_multifileupdate.qmt
tests_collection_multifilewrite.qmt
tests_collection_read.qmt
tests_collection_update.qmt
tests_collection_write.qmt
tests_complexobjectstorage_functionality.qmt
tests_constref_functionality.qmt
tests_datasvc_appendread.qmt
tests_datasvc_appendwrite.qmt
tests_datasvc_crossreference.qmt
tests_datasvc_inheritance.qmt
tests_datasvc_objectreference.qmt
tests_datasvc_ptnumber.qmt
tests_datasvc_ownership.qmt
tests_datasvc_readupdate.qmt
tests_datasvc_readwrite.qmt
tests_datasvc_simpleappend.qmt
tests_datasvc_typechecking.qmt
tests_datasvc_writedelete.qmt
tests_datasvc_writetwice.qmt
tests_explicitcollection_functionality.qmt
tests_filecatalog_cmsfilemanager.qmt
tests_filecatalog_functionality.qmt
tests_generalref_functionality.qmt
tests_implicitcollection_functionality.qmt
tests_persistencysvc_functionality.qmt
tests_polymorphicref_functionality.qmt
tests_roottree_performance.qmt
tests_staticref_functionality.qmt
tests_storagesvc_basicfunctionality.qmt
xmlcatalog.qms
xmlcatalog_xmlfunctionality.qmt
xmlcatalog_xmlimporttest.qmt
xmlcatalog_xmlmetatest.qmt
[lxplus068] /afs/cern.ch/sw/lcg/contrib/mgallas/POOL_1_1_0/src/config/tests/qmtest >
    
```

lcg-qmtestconfig.py will produce these files



The screenshot shows two overlapping browser windows displaying the QMTest web interface. The left window shows a directory listing of test files, and the right window shows a list of test suites.

Directory Listing (Left Window):

- Test
- [attributelist assert](#)
- [attributelist rvtest](#)
- [datasvc cachengctest](#)
- [datasvc datasvcctest](#)
- [datasvc reftest](#)
- [datasvc retrievetest](#)
- [edgcatalog edgfunctionality](#)
- [edgcatalog edglookuptest](#)
- [edgcatalog edgmeta](#)
- [filecatalog systemtools](#)
- [filecatalog wriparser](#)
- [mysqlcollection inserttest](#)
- [mysqlcollection readtest](#)
- [mysqlcollection updatetest](#)

Suites List (Right Window):

- tests_datasvc_typechecking
- tests_datasvc_writedelete
- tests_datasvc_writetvice
- tests_explicitcollection_functionality
- tests_filecatalog_cmsfilemanager
- tests_filecatalog_functionality
- tests_generalref_functionality
- tests_implicitcollection_functionality
- tests_persistencysvc_functionality
- tests_polymorphicref_functionality
- tests_roottree_performance
- tests_staticref_functionality
- tests_storagesvc_basicfunctionality
- xmlcatalog_xmlfunctionality
- xmlcatalog_xmlimporttest
- xmlcatalog_xmlmetatest

Suites (Right Window):

- attributelist
- datasvc
- edgcatalog
- filecatalog
- integration_system
- mysqlcollection
- rootcollection
- xmlcatalog

A red circle highlights the 'Suites' section in the right window.





The screenshot shows two Mozilla browser windows. The left window displays the 'QMTest: Test Results' page, which includes a summary table and a list of test outcomes. The right window shows the 'QMTest: Result Detail' for a specific test, displaying its execution log.

Outcome	# of Tests	% of Total
FAIL	11	19.64
PASS	45	80.36
Total	56	100.00

Test Results List:

- attributelist assert (PASS)
- attributelist rvtest (PASS)
- datasvc cachemgrtest (PASS)
- datasvc datasvc test (PASS)
- datasvc reftest (PASS)
- datasvc retrievetest (PASS)
- edgcatalog edgfunctionality (FAIL)
- edgcatalog edglookup test (PASS)
- edgcatalog edgmeta (FAIL)
- filecatalog systemtools (PASS)

Test Detail: datasvc_cachemgrtest

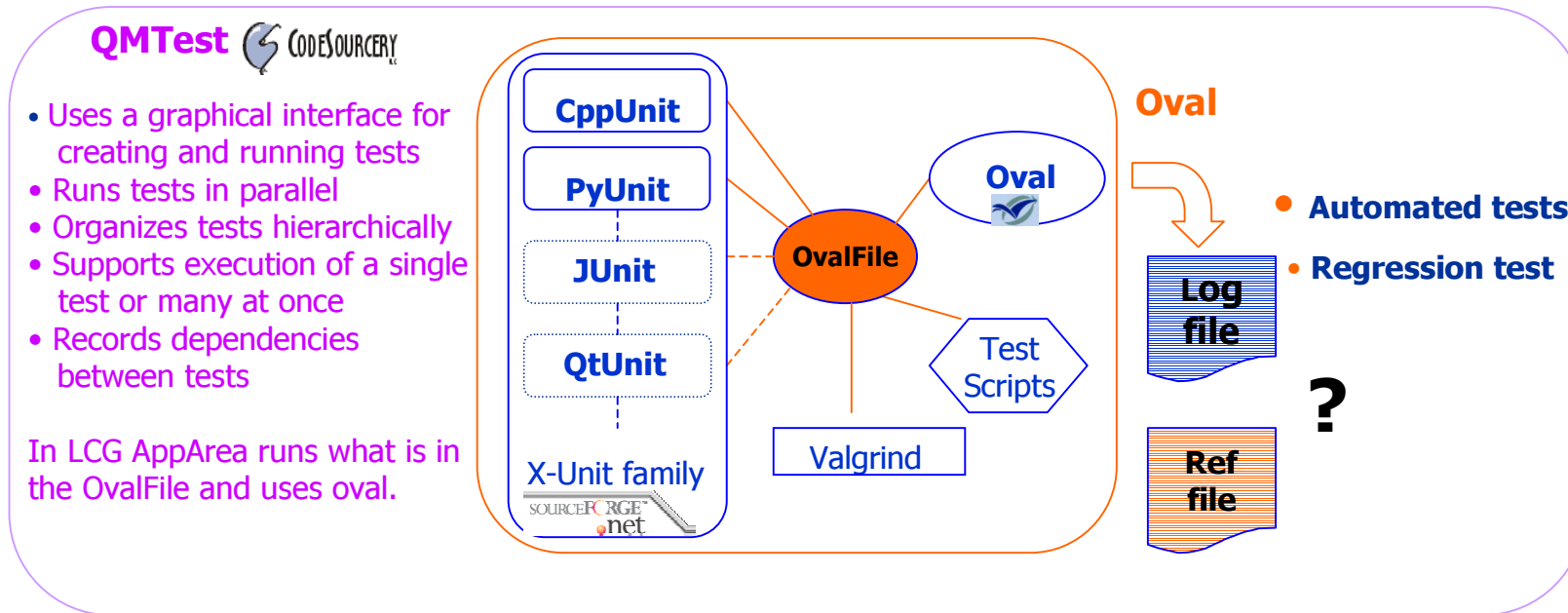
```

Outcome: PASS
Cause:
Annotation: Value
ExecTest.expected_stdout:
===== unitTest_DataSvc_CacheMgrTest =====
===== run =====
Testing get, set and remove on Token methods
-----
Testing clearCache method
-----
Testing get, set and remove on Null Token methods
-----
[OWAL] Test result: 0 Errors
===== unitTest_DataSvc_CacheMgrTest =====
===== diff =====
ExecTest.stdout:
nmtest target
rh73 acc??
    
```





- Aim:** to help developers:
- to produce code for testing
 - to run tests in automatic way





<http://spi.cern.ch/>

The screenshot shows a Microsoft Internet Explorer browser displaying the LCG SPI website. The browser's address bar shows <http://spi.cern.ch/>. The website content is organized into several sections:

- SPI Quick Links:** Links to SPI Home, SPI Index, Savannah Portal, and LCG App. Area (Home Page, LCG Agenda).
- External Links:** Links to CERN, EP Division, and IT Division.
- LHC experiments:** Links to ALICE, ATLAS, CMS, and LHCb.
- Grid Links:** Links to Globus, EU DataGrid, PPDG, and GPhyN/VDGL.
- Sw-Testing Policies:** Software testing policies within LCG AppArea projects.
- FrameWorks for Sw-Testing:** Currently four test-framework tools are provided as free software by the SPI External Software Service: CppUnit and PyUnit for unit-tests, Oval for validation tests, and QMTest for GUI interface.
- Sw-Testing HowTo:** A list of documents including HowTo make SW-Tests, HowTo for CppUnit TestFramework, HowTo for Oval TestFramework, HowTo for PyUnit TestFramework, HowTo-TestFramework-QMTest, and HowTo for Test Execution Frameworks.
- Sw-Testing documentation and planning:** Documents like testplan_template.html and testcase_template.html.
- Support:** Information on reporting bugs and asking for support via the SPI Project Portal.

Annotations with blue arrows point to specific sections: 'Policies' points to the Sw-Testing Policies section, 'Test Frameworks' points to the FrameWorks for Sw-Testing section, 'HowTo' points to the Sw-Testing HowTo section, and 'Test doc' points to the Sw-Testing documentation and planning section.





Sw-Testing Policies:
Software testing policies within LCG AppArea projects

FrameWorks for Sw-Testing:
Currently four test-framework tools are provided as External Software Service. These are: CppUnit and PyUnit can cover from unit to validation tests, and QMTest will interface to organize and run the tests. A set of HowTo's are available and can help with this tools.

Sw-Testing HowTo

- [HowTo make SW-Tests](#)
- [HowTo for CppUnit TestFramework](#)
- [HowTo for Oval TestFramework](#)
- [HowTo for PyUnit TestFramework](#)
- [HowTo-TestFramework-QMTest](#)
- [HowTo for Test Execution Frameworks](#)

Sw-Testing documentation and planning:

- [testplan_template.html](#) - The main purpose of this document is to schedule the testing process with all necessary software and staff responsibilities. Under testplan you can find test case specification and data along with the expected results for a particular test.
- [testcase_template.html](#) or test case specification template

Support:
If you need to report a bug or ask for support, please use the [bug report system](#) and [support area](#).

Grid Links

- [CERN](#)
- [EP Division](#)
- [IT Division](#)
- [LCG](#)

LHC experiments

- [ALICE](#)
- [ATLAS](#)
- [CMS](#)
- [LHCb](#)

HowTo for CppUnit TestFramework

HowTo for QMTest TestFramework

What is QMTest?
QMTest (<http://www.codesourcery.com/qm/qmtest>) is an open-source, cross-platform software testing tool written in Python. QMTest is a general purpose testing solution that allows an organization to implement a robust, easy-to-use testing program tailored to its needs. QMTest works with most varieties of UNIX, including GNU/Linux, and with Microsoft Windows. See the [SPI supported platforms](#) at [SPI external software service](#).

QMTest uses a graphical interface for creating and running tests, runs tests in parallel, organizes tests hierarchically, supports execution of a single test or many at once and records dependencies between tests together with the results and expected results.

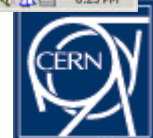
[return to top of page](#)

How to use QMTest in LCG AppArea?

1. [QMTest](#) tool and the "lcg-qmtest-config.py" script should be in your path. QMTest path was included in the SCRAM ToolBox and the path to the correct version installed in the [SPI external software](#) is managed by SCRAM. So, if you are in a SCRAM based project you do not need to take care of this. The "lcg-qmtest-config.py" will be available after set up the LCG-SPI environment with:

```
source /afs/cern.ch/sw/lcg/app/spi/tools/latest/setup/lcgspi.csh
```

3 Follow sw-testing policies



SPI SW-Testing

- Test Frameworks
- User support
- Test policies
- Test documents



Thanks to:

- LCG-POOL team
- LCG-SEAL team

Feedback and interaction are always welcome!!