# GammaLib - Feature #1079

## Improve GammaLib unit testing experience

01/13/2014 08:36 PM - Deil Christoph

|  |  |  |  |
|---|---|---|---|
| **Status:** | In Progress | **Start date:** | 01/13/2014 |
| **Priority:** | Normal | **Due date:** |  |
| **Assigned To:** |  | **% Done:** | 10% |
| **Category:** |  | **Estimated time:** | 0.00 hour |
| **Target version:** |  |  |  |

**Description**

(I think we already discussed this a year or two ago, but I couldn't find an issue for this.)

Currently test-driven development with GammaLib is pretty painful.
(Today Ellis and I tried to make sense of the test failures in #1014 and it was very slow, basically because it was hard to figure out which test_value calls corresponded to which test error)

When a test error occurs the only info printed to the console is in which GammaLib module the error occured:

```
...
PASS: test_GApplication
FAIL: test_GModel
PASS: test_GSky
PASS: test_GOptimizer
...
```

The developer than has to find the info at
https://cta-redmine.irap.omp.eu/projects/gammalib/wiki/Contributing_to_GammaLib#Running-unit-tests to know to open up the test/reports/GModel.xml file, which looks like this:
https://gist.github.com/cdeil/8406019

Then he has to parse this XML file to find the <failure> element:

```
  <testcase assertions="" classname="GModel" name="Test GModelPar: Test rescaling: Test if 1 is within 1 +/- 1e-10" status="" time="0" />
  <testcase assertions="" classname="GModel" name="Test GModelPar: Test rescaling: Test if 1 is within 42 +/- 1e-10" status="" time="0">
    <failure message="Value 1 not within 42 +/- 1e-10 (value-expected = -41)." type="" />
  </testcase>
  <testcase assertions="" classname="GModel" name="Test GModelPar: Test rescaling: Test if 1 is within 1 +/- 1e-10" status="" time="0" />
```

And then he has to search to test/test_GModel.cpp file to find the test case corresponding to Test GModelPar: Test rescaling and then try to figure out which test_value call failed.
(Note: no file name or line number in the XML report file.)

In HESS we are using googletest which provides a much nicer experience.
It tells the developer very clearly on the console where the error occured and what the problem was, see e.g. this introduction:
http://www.ibm.com/developerworks/aix/library/au-googletestingframework.html

Another feature that is super-useful for developers is to be able to select the tests to run:
https://code.google.com/p/googletest/wiki/AdvancedGuide#Selecting_Tests

My suggestion would be to switch to GoogleTest in GammaLib.

It is feature-complete (e.g. it also contains XML test report output as the GammaLib test classes do) and almost completely bug-free (because it's been tested by millions of users for years) and could e.g. be used for all CTA C++ projects, making test-driven development of CTA software (including GammaLib / ctools) a pleasure.

:-)

## History

**#1 - 01/13/2014 09:56 PM - Knödlseder Jürgen**

The actual test suite was geared towards continuous integration, not the developers  smile.png The actual format is a standard for tools like Jenkins or Sonar, but I agree that it's not very human readable. We may think about an extension or other option for developers.

**#2 - 01/30/2014 12:06 AM - Knödlseder Jürgen**

*- % Done changed from 0 to 10*

I just found out how to run a single unit test. This is pretty easy:

```
env TESTS="test_python.py" make -e check
```

runs only the Python test. See
https://cta-redmine.irap.omp.eu/projects/gammalib/wiki/Contributing_to_GammaLib#Running-only-a-subset-of-the-unit-tests

**#3 - 01/30/2014 12:06 AM - Knödlseder Jürgen**

*- Status changed from New to In Progress*

**#4 - 04/24/2014 04:12 PM - Deil Christoph**

By now I have a lot of experience with GoogleTest for C++ (https://code.google.com/p/googletest/) and pytest (http://pytest.org/latest/) for Python and I think implementing this would be the best contribution I can make for Gammalib this summer before the 1.0 release.

I think those are the best packages for C++ / Python and using those would be a huge improvement over the current hand-written Gammalib testing framework:
- Small, can be bundled in the repo (no extra dependency)
- Can be used everywhere in CTA (no extra learning effort for devs)
- Many more features (test writing, running, debugging) and better documentation.