

## **The Role of Testing in the Software Process of the Geant4 Low-Energy Electromagnetic Physics Working Group**

*P. Nieminen and M.G. Pia*

### **1 Introduction**

Testing forms a vital part of the software process in developments as advanced and complex as those currently in progress in the Geant4 Low-Energy e-m physics Working Group. The purpose of this document is to outline the procedures to be followed regarding testing both during development of new software, and during updates and corrections to existing code.

### **2 Testing objectives and goals**

The objective of testing is to ensure the new, or updated, code performs as intended. Testing should reveal any potential deviancies from expected behaviour of the code both from physics and performance point of view. The goal is high-quality code ready for public release, ultimately leading to easier maintenance and substantial timesaving for developers in the course of the software lifecycle.

### **3 Test designs and testing schedules**

#### **3.1 Test requirements**

1. Testing should be performed according to agreed and documented procedures.
2. Traceability through requirements-design-implementation-tests should be implemented.
3. The design should be tested for satisfying the user requirements.
4. The code implementation should be tested for compliance with the design.
5. The code should be tested for correct functionality.
6. The code should be tested for compliance with Geant4 coding guidelines.
7. The code should be tested for satisfactory quality, clarity and readability.
8. Every class of the lowenergy category shall be exercised in an appropriate system test (directly or indirectly).
9. The code should be tested on all Geant4 supported platforms.
10. The code shall be submitted to the entire set of tests above to be considered for release.
11. Tests and test tools should be documented.
12. The test code should be kept under configuration management (in Geant4 CVS repository).
13. Reference outputs, data sets for validation tests etc. should be kept in appropriate agreed locations, accessible to the whole WG.
14. Test tools should be maintained.
15. Modifications of the tests (including test tools, reference outputs, data sets etc.) should be performed according to agreed and documented procedures.
16. The most recent test results should be made available to WG coordinators for code to be included in a monthly global tag or in a Geant4 public release, according to the guidelines described in the "Testing process" section.

17. All requirements on testing agreed by the Geant4 Collaboration, and by the System Testing Team in particular, should be satisfied.

### **3.2 Design of the test system**

Use cases are utilised to test the design.

The compliance of the code implementation with the design is tested by means of code reviews and comparison of reverse engineered code versus baseline design.

The code tests include:

- Unit testing
- System (integration) testing
- Acceptance (physics validation) testing
- Regression testing

Appropriate testing tools should be developed for each of these phases.

Code reviews by experienced collaborators and code filtering through Code Wizard (<http://pcitapiwww.cern.ch/CW>) shall be used to ensure satisfactory quality standards. Pair coding is encouraged to improve the quality of the code.

Unit tests should be designed to cover all the functionalities of a class and test all its responsibilities.

The configuration (energy range, types of particles, materials etc.) of system tests should be adequate to a thorough testing.

### **3.3 Test schedules**

The lowenergy tests are carried on in accordance with the overall testing and release schedule of Geant4.

Unless differently agreed with WG coordinators in special cases, developers should complete the full testing process under their responsibility one week prior to the deadline for the code delivery assigned to the lowenergy category, both for monthly global reference tags and Geant4 public releases.

## **4 Testing tools**

The unit, validation and regression tests for the Low Energy Electromagnetic category are kept in the `geant4/source/processes/electromagnetic/lowenergy/test` directory. Tests consisting of one file only are kept in that directory directly. More complex tests, consisting of more than one file, should be kept in subdirectories. Subdirectories of the test directory should be created only with prior consultation of the WG coordinators.

System tests are kept in subdirectories of `geant4/tests`.

The location of auxiliary test material, such as, for instance, experimental data used in validation tests, is decided case by case, in consultation with the WG coordinators and the Release Manager.

The various testing tools in each of the domains should be documented in a README file by the responsible developer, and committed to the repository. The README file should be written in English and contain, as minimum, a description on

- The function of the test
- Required input
- Associated files (reference data, macros etc.)
- Clear instructions on how to run the test
- The expected output of the test
- The author name
- A record of the history of the test tool, reporting significant evolutions (such as bug fixes, new features introduced, changes in reference data etc.).

Updates and changes to the tests themselves should be done with extreme caution as these may have a direct effect on the outcome of the testing. Any such changes should be well justified, reported to the WG, and clearly documented in the README file.

#### **4.1 Use cases**

Use cases should be coded in UML.

#### **4.2 Unit test tools**

Unit tests test a single code unit (for instance, a single class).

More than one test code may be required to fully test a class to cover all its features and functionalities at unit test level (example: the AlongStepDolt and PostStepDolt functionalities of a process may be tested by two distinct test codes, even if they may logically belong to the same unit test).

A unit test tool may be functional to test more than one class (example: the same unit test may be used for Low Energy and Standard Compton processes).

The evolution of unit test tools is logged in the History file of the lowenergy category.

#### **4.3 System test tools**

System tests consist of full-scale applications, exercising the lowenergy code together with other system components.

System tests should be compliant with the requirements of Geant4 System Testing<sup>1</sup>.

Two macros for batch running the test, respectively for normal and large statistics, should be provided in the CVS repository of the test. Reference outputs for the two cases should be provided in the same directory too.

A History file should be provided in the test directory for logging the test tool evolution, and regularly updated.

#### **4.4 Validation test tools**

Validation tests compare the behaviour of a lowenergy physics process/model with experimental data, or theoretical expected behaviour, whenever no experimental data are available.

Validation test tools can be of any type: simple single test programs as well as full scale application codes.

The experimental data should be provided as part of the test tool (for instance, in a file to be read in), together with their experimental errors.

The sources of experimental data should be documented.

The sources of experimental data can be:

- Publications on refereed journals or well established public databases (preferable source);
- Publications on preprints, conference proceedings etc.: in such case, the evolution of the data source should be followed, and the data set should be replaced, if the results are published on a refereed journal later;
- Data from experimental tests of collaboration parties: in such case an agreement should be established with the collaborating party to ensure that the data can be posted on the group web pages and referenced in a possible publication by the LowE group, with full credit to the experimental party.

The test should provide plots or tables of comparison between simulated and experimental data (with their errors) and information of statistical significance for the comparison.

Validation test code and associated data should be kept under configuration management and should be publicly available to all WG members. Developers should consult the WG coordinators to agree on an appropriate location for their validation test code and data.

---

<sup>1</sup> To be listed in a document currently in preparation.

The responsible of a validation test should provide a significant selection of results to the WG coordinators, to be posted on the WG web site.

Testing with external, undocumented and “proprietary” test cases will not be sufficient for acceptance of new or updated code.

#### **4.5 Regression tests**

Regression tests can be (a subset of) unit or system or validation tests, or independent test codes.

Reference outputs should be maintained in an appropriate location, accessible to all WG members.

Any deviation of the output of a regression test from the reference one should be reported to WG coordinators.

Reference outputs may require updates because of new features introduced in the Geant4 code, bug fixes etc. Updates should be agreed with the WG coordinators and should be communicated to all WG members by the test responsible.

#### **4.6 Scripts**

Scripts to automate the testing process in all its phases should be developed and documented. At present they are not available yet.

### **5 Testing process**

This section describes the testing process concerning the lowenergy category. The testing process concerning the examples under the responsibility of the Low Energy Electromagnetic Physics Working Group are described in a following section.

The following items are under the responsibility of the developer:

1. Perform the testing process in agreement with the present document.
2. Verify that the code is strictly compliant with the agreed design.
3. Verify that the code is compliant with Geant4 coding and style guidelines.
4. Compile frequently while developing code to test the formal correctness of the code and remove compilation defects (errors and warnings) at an early stage.
5. Test the code for compilation on at least two of the Geant4 supported platforms.
6. Commit the developments to the CVS repository frequently, to allow for code reviews to test the quality of the code. Only code that compiles on at least two supported platforms should be committed to CVS.
7. If new features are developed, write appropriate new unit tests, or adapt existing ones; commit them to CVS and document them.
8. Perform coding/unit test cycles until all the responsibilities and functionalities of the code are exercised and all unit tests provide satisfactory output.
9. If new unit tests have been developed, contact the WG coordinators to contribute (a subset of) new unit tests and associated reference outputs to the regression test suite.
10. Make the results of validation tests available in publicly accessible locations, previously agreed with WG coordinators.
11. If new features are developed, consult the WG coordinators about writing an appropriate new system test, or adapting an existing one; follow the guidelines indicated in the WG document about the Development and Tag process, concerning System Tests.
12. If the reference output of a system test is expected to change, as a result of modifications to the test code, introduction of new features or bug fixes in the lowenergy code, provide a new reference output, to be committed to CVS. This case should be discussed and agreed with WG coordinators prior to moving to action.
13. Run a system test appropriate to the new developments on two supported platforms at least; remove defects until the test runs successfully.

14. Verify that the new code does not break any of the other existing code: run regression tests. Remove defects until the test runs successfully.
15. If new features have been developed, write appropriate validation tests; commit them to CVS and document them. Make the reference data available in publicly accessible locations, previously agreed with WG coordinators.
16. If new validation tests have been developed, contact the WG coordinators to contribute (a subset of) new validation tests and associated reference outputs to the regression test suite.
17. Perform validation tests on at least one supported platform. Remove defects until all the tests run successfully.
18. Contact the WG coordinators to update the traceability map.

The processes above should be performed as part **of development-test cycles, not as a code-first, test-at-the-end sequence**. Also cycles of the various types of tests should be performed as needed, not necessarily strictly in the sequence listed above: for instance, after fixes to remove defects found in validation testing, it may make sense to re-run integration tests; or unit tests may be used to investigate defects found in system testing etc.

**Intermediate releases of new developments in progress, still with limited functionalities, are strongly encouraged**; however, in any release the whole testing process should be performed.

The following items are under the responsibility of WG coordinators:

1. Perform the testing process in agreement with the present document.
2. Test that the design satisfies the user requirement.
3. Test that the design is of satisfactory quality.
4. Verify that the code provided by developers is strictly compliant with the agreed design.
5. Verify that the code provided by developers is compliant with Geant4 coding and style guidelines and other
6. Verify the quality, clarity and readability of the code.
7. Verify that the testing tools provided by developers are complete (i.e. include documentation, reference outputs, macros etc.
8. Verify the output of all unit tests provided by developers; re-run them, if necessary, after the integration of contributions from various sources.
9. Test the category code meant for inclusion in monthly global reference tags and public releases; run all system tests on at least two of the Geant4 supported platforms and all regression tests.
10. Verify the output of validation tests provided by the developers.
11. Contact the developers to remove the defects found at any stage of the testing. Reject the code, if the defects are not removed.
12. Prompt the developers to run, or re-run, any tests, if needed during the category integration and release process.
13. Perform the acceptance tests appropriate for the release of a category tag.
14. Verify that the user requirements are satisfied.
15. Update the traceability map.
16. Maintain the reference outputs for regression testing.
17. Maintain the collection of outputs of validation test results.
18. Collect quantitative information, to be used for improvements of the testing process, and analyse it; propose process improvements, based on such analysis.
19. Perform defect analysis and discuss it within the WG, for the purpose of preventing future occurrences of the same defects.
20. Train WG members to the testing process.
21. Verify the agreed testing process to be correctly applied by all WG members.

22. Maintain the present document.

### 5.1 Testing advanced examples

This section describes the testing process concerning the examples under the responsibility of the Low Energy Electromagnetic Physics Working Group.

The following items are under the responsibility of the example developer group:

1. Perform the testing process in agreement with the present document.
2. Verify that the code is strictly compliant with the agreed design.
3. Verify that the code is compliant with Geant4 coding and style guidelines.
4. Test the code for compilation on all Geant4 supported platforms. Remove any defects until the code compiles.
5. Test the code for building, running and producing meaningful results on all Geant4 supported platforms. Remove any defects and re-test the code until it runs successfully, producing correct results.
6. Perform the test at the previous two points at least at the time of each Geant4 scheduled release: after the completion of the system testing phase, with the resulting global candidate tag issued by the Release Manager, and well before the scheduled release date. Contact the Release Manager to agree an appropriate schedule together. Better, the test at the previous point should be performed every month, after the release of a global reference tag, to avoid accumulating many integration problems at the last minute.
7. Inform the WG coordinators of the results of the tests.

## 6 Notes

**Only the code that has been submitted to the whole testing process, with successful results, can be released.** The code that has been submitted to partial testing only, or does not satisfy all the tests, will not be accepted for a Geant4 release.

Any modification of the testing process should be discussed and agreed within the WG. Once agreed, it should be properly documented by updating the present document.

## 7 Acknowledgements

Many thanks to G. Cosmo for reviewing this document.