

LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY  
- LIGO -  
CALIFORNIA INSTITUTE OF TECHNOLOGY  
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

<b>Document Type</b>	<b>LIGO-T040XXX-00-Z</b>	2005/04/06
<b>LSC Data Analysis Software Practices</b>		
Data Analysis Software Working Group		

*Distribution of this draft:*

LSC

**California Institute of Technology**  
**LIGO Project - MS 51-33**  
**Pasadena CA 91125**  
Phone (626) 395-2129  
Fax (626) 304-9834  
E-mail: [info@ligo.caltech.edu](mailto:info@ligo.caltech.edu)

**Massachusetts Institute of Technology**  
**LIGO Project - MS NW17-161**  
**Cambridge, MA 01239**  
Phone (617) 253-4824  
Fax (617) 253-7014  
E-mail: [info@ligo.mit.edu](mailto:info@ligo.mit.edu)

WWW: <http://www.ligo.caltech.edu/>

# Contents

<b>1</b>	<b>Preface</b>	<b>3</b>
<b>2</b>	<b>Environments for data analysis (17 MARCH 2004)</b>	<b>3</b>
<b>3</b>	<b>Quality Assurance Practices</b>	<b>3</b>
3.1	LSC Software Projects (17 MARCH 2004) . . . . .	4
3.2	QA Program (21 AUGUST 2004 ) . . . . .	4
3.2.1	Components . . . . .	4
3.2.2	Analysis Software . . . . .	5

## 1 Preface

The scientific goal for which we are developing software is to constrain properties of gravitational-wave sources and/or detect gravitational waves through a correct analysis of our data. After that, we also want to correctly extract scientific information. (But first things first.)

The software requirements described below are intended as a starting point to facilitate the delivery of demonstrably correct and reliable software for data analysis to the LSC. We will revisit both the requirements and the detailed nature of the software on a regular basis; this will insure that our direction remains aligned with our scientific mission and is adaptable to the rapid changes in both computing software and hardware.

## 2 Environments for data analysis (17 MARCH 2004)

**APPROVED BY LSC EXECUTIVE COMMITTEE: 15 OCTOBER 2004**

The LSC approves the use of DMT, GEO++, LALApps, LDAS and Matlab in data analysis leading to observational publications of the LSC. It is understood that there may be other glue environments that also get used.<sup>1</sup> If a question arises about any glue environment, the LSC Executive Committee will decide about its use based on technical information provided by the DASWG. All environments are subject to the quality assurance practices described in Sec. 3

## 3 Quality Assurance Practices

*Goal:* The LSC is motivated to do good science. A QA program must be in support of the science and not capriciously interfere with scientific effort, but it must accomplish our overriding QA goal:

**Avoid any errors in public statements of LSC results.**

To this end our collaboration level software checking should be aimed at finding potentially serious errors beyond those readily anticipated in testing by individual developers and their close collaborators.

A secondary goal is:

**Detect errors early in the cycle to avoid delaying publications by finding errors at the final stages.**

Most important for this goal is to avoid misunderstandings by establishing agreement as early as possible in a review process as to what the software is intended to do.

When an analysis is reviewed and the pipeline tested, the requirements listed below will be checked. The goal is to insure the high quality of software used to produce data analysis results and to help assure LSC members of the validity of data analysis results. Failure to satisfy LSC software quality assurance practices may delay the release of results outside the LSC.

---

<sup>1</sup>Examples of these glue environments are Root, Triana, and various scripting languages.

### 3.1 LSC Software Projects (17 MARCH 2004)

**APPROVED BY LSC EXECUTIVE COMMITTEE: 26 MARCH 2004**

Each LSC software project which impacts on data analysis leading to observational publications of the LSC is required to maintain a project page under the DASWG web pages. This page must include information about the project librarian, bug-tracking, version control, mailing lists and contributors:

1. Project librarians will coordinate the development process for each project and may choose to have a separate home page for the project.
2. Version control will be managed through a DASWG approved version control system. All data analysis results reported in observational publications of the LSC are required to provide CVS tags of software used in the pipeline(s).
3. Bug tracking will be managed using a DASWG approved problem-tracking system.
4. Mailing lists should be located on one of the existing mailing list servers.
5. LSC Software projects must provide a set of guidelines for developers along with developer and user how-to documents. The guidelines will include clear statements of practice on documentation.

The DASWG coordinates aspects of LSC software development which have common impact by developing conventions and protocols, by defining data formats which are used by all LSC software projects, and by developing and implementing a long-term strategy for LSC data analysis software.

### 3.2 QA Program (21 AUGUST 2004 )

**APPROVED BY LSC EXECUTIVE COMMITTEE: 15 OCTOBER 2004**

**The First Line of Defense** The first line of defense is careful coding and thorough testing by developers. Team efforts are encouraged since the presence of multiple sets of eyes and minds carefully examining code during development improves its reliability. Reviewers should take into account aspects of the history of a code's development, including testing and the extent to which it has been developed in isolation, that bias toward more or less reliability.

**QA Program for Our Two Phases of Development** For QA purposes we divide our software development into component development and analysis software activities. The QA program for each will include: a) A requirements statement available at the review defining completely but concisely what the intent is for the software; b) A minimalist set of coding and documentation standards aimed at making the code readily understandable by reviewers; c) a review group and schedule plan for reviews and MDC tests; d) a coordinator who signs off on the software (and who also may have other roles depending on the phase and context such as librarian, project QA coordinator, chair of analysis review committee, etc.).

Details of each phase follow

#### 3.2.1 Components

The component phase involves the development of well-defined elements of the larger library, potentially reusable for other analyses beyond that originally motivating the developer. The components will be incorporated in the (appropriate) CVS library and they will need to be integrated into larger production packages.

When a component is reasonably mature and ready for use by others or for integration, its QA process kicks in. This includes:

- A one-page requirements statement of what the component is supposed to do, including inputs and outputs and specific (version number) reference to algorithms and other components it uses.
- Approval that it reasonably meets the LIGO software guidelines and standards and the requirements of the relevant CVS library. The latter includes a testing package, with input data and expected results, so that the librarian can determine that the component is nominally working when integrated with components already in the library into a larger package.
- A code review (walk-through) as described above with 2 or 3 reviewers (including a chair). In a walk-through (also called a code review) the developer stands in front of a group of colleagues and explains the code line by line. The developer will find most of the bugs in a code review. The physical presence of the reviewers is essential.
- Certification by the coordinator that the component at this major version level corresponds to its requirements statement, follows the spirit of the standards, keeps the code librarian happy, and has suffered a walk through.

Two important issues affect the component QA process:

- **Evolving Code.** We cannot repeat numerous QA reviews for a component. After the first time, walk-throughs become less effective. The QA coordinator and developer will have to make the important judgment call as to what constitutes reasonably mature, since components are often under continuous development. A QA signoff will have to occur in advance of the beginning of any QA process on analysis software using this component. How much code development can be permitted before a new major version level is declared and a new walk through of changes at least is required will be a continuing concern.
- **Existing Code and Large Packages.** Large existing packages, whether LIGO-developed like LAL or external like Matlab, consist of many components. Some components are in heavy use and subject to extensive scrutiny, and some are on the peripheries and have not been used or examined sufficiently to meet LIGO's standards of reliability. A DASWG review group should be appointed to examine each package and certify those components that have had sufficient usage and scrutiny and have complete and accurate requirements descriptions. Certified components may be exempted from code walk-throughs. All other components, including those not part of an identified package, should be subject to the full review process, if they are to be used in analysis software, sufficiently in advance of the analysis software review.

### 3.2.2 Analysis Software

An analysis software activity is a combination of software runs with defined data sets, calibration constants, trigger and veto lists, etc., whose results are to be included in a published LIGO talk or paper. This is the most important QA process of all:

- At the time of the review there must be a requirements statement detailing what software components and packages (versions) are being used, the pipeline script, data sets, calibration constants, etc. This statement is a complete list with attached references where appropriate (such as the component requirements, the script, etc.) of everything that could change the published result.

- The review chair (coordinator) should be satisfied that the component documentation is adequate for the review. This should have been resolved in previous phases and should not be an issue here.
- The analysis software review group should include at least two persons capable of following the pipeline script and two persons (may be the same) who are familiar with the coding environment used in the components. It may include overlap with the corresponding physics analysis review committee or it may be independent. The committee and reviewee(s) should first go over the requirement statement in detail and establish agreement on its content and its meaning. The committee should review the QA signoffs on all components and determine whether new reviews are required because of development work following the earlier component review. A walk-through of the script is essential; a sub-committee may conduct it separately from the main committee, but it must examine each line of the script and particularly examine data, calibration, trigger, veto, etc., input files scrupulously. The committee should examine results of a MDC run and/or the detection sensitivity of injected signals in the software or hardware data path and establish they are consistent with what is expected.
- When the committee is satisfied, the chair (coordinator) certifies the analysis software activity and informs the corresponding physics analysis review committee.