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

Technical Note **LIGO-T010011-00 - E** 4/17/01

Summary of the LDAS Database Mock Data Challenge

Peter Shawhan

Distribution of this draft:

LDAS Group; LIGO Scientific Collaboration

This is an internal working note
of the LIGO Project

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

Massachusetts Institute of Technology
LIGO Project - MS 20B-145
Cambridge, MA 01239
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

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

1 INTRODUCTION

The LIGO Data Analysis System (LDAS) includes a relational database system to store various kinds of information, such as a catalog of the raw data in the main data archive, summary information (statistics, power spectra, etc.) about time intervals of interest during detector operation, and lists of astrophysical event candidates and environmental transients. These things are sometimes referred to as "metadata" to distinguish them from the multichannel time-series data which is recorded by the LIGO data-acquisition systems and archived separately. The database system is designed to be able to accommodate additional types of information as the need arises.

This document contains a brief summary the results of the "Mock Data Challenge" (MDC) which focussed on the LDAS database system. A separate document[1] will present more detailed information about test protocols and specific results.

The LDAS database system uses commercial database software (IBM's DB2 "Universal Database") to provide the core database functions, with LDAS software components ("APIs") handling database transactions (Metadata API), format translations (Lightweight API), and job control (Manager API). These were among the first LDAS APIs to be implemented, and were largely complete by mid-2000. Therefore, the plan for the database MDC[2] was formulated to address the system as a whole, with several distinct goals:

- Validate the database table design
- Test the functionality and performance of the hardware and software components of the core database system
- Test the process of inserting triggers and other data generated by the Data Monitoring Tool (DMT) system into the database
- Evaluate user interface tools (both graphical and program-based)
- Exercise various database administration tasks
- Ensure that accurate documentation exists for all aspects of the system.

When the MDC planning document was written in August 2000, it was expected that the MDC would consist of a preparation period of a few months, followed by an intense "challenge" period near the end of 2000. However, serious limitations on available manpower led to a change in strategy, with the MDC task list being worked on gradually over the course of many months. The major goals of the MDC were completed by April 2001, except for additional performance studies (awaiting installation of the final disk storage configuration) and the completion of detailed documentation about the testing protocol.

Contributors to the MDC effort included Peter Shawhan, Philip Charlton, Phil Ehrens, Mary Lei, Maria Barnes, Ed Maros, Greg Mendell, John Zweizig, and Roy Williams.

---

1. LIGO-T010045-00-E, "Details of the LDAS Database Mock Data Challenge".
2. LIGO-T000089-00-E, "Plan for the LDAS Database Mock Data Challenge".

2 VALIDATION OF DATABASE TABLE DESIGN

IBM's DB2 "Universal Database" is a relational database which uses the SQL query language to insert and retrieve information. All information is stored in "tables", each with a fixed number of pre-defined columns and a variable number of rows which represent database entries. A complete draft design for the database tables was presented in LIGO-T990101-02-E, "Table Definitions for LDAS Metadata / Event Database". Prior to the MDC, all of the tables described in the document had been instantiated in DB2 databases, and had been populated with test data to verify their self-consistency.

As part of the MDC, we researched the ways DB2 uses indexes for optimal execution of user queries involving row selection and/or sorting. This new understanding, plus a list of "common" queries expected for each database table, led to an extensive redesign of the set of indexes to be used. Several tables now have three or more indexes, reflecting the philosophy that we should optimize more for data retrieval than for insertion.

During the course of the MDC, some of the database tables were used for the first time with "real" data from the LDAS Data Conditioning API and from external sources. This led to a number of minor modifications to certain tables in response to user requests. However, the database tables for astrophysical event candidates have yet to be put into use and are likely to be revised by the science teams as their search algorithms mature.

3 DATABASE SYSTEM FUNCTIONALITY

As mentioned in the introduction to this document, the LDAS software needed for the database system was largely implemented before the MDC began, and was known to work in "ordinary" cases. A major goal of the database MDC was to rigorously test whether the database system behaves appropriately, even in "unusual" situations. To do this, we developed a set of test scripts to submit LDAS commands and evaluate the results. In all, the test scripts use 97 files of test data in "LIGO lightweight" (LIGO_LW) format[1], and submit 217 LDAS jobs. These tests covered:

- Communication with the LDAS managerAPI
  - Basic job submission and status reporting
  - Handling of incorrect username or password
- Data insertion
  - Automatic assignment of unique IDs to database entries
  - Alternative specifications of explicit unique IDs
  - Appropriate error message when data being inserted would violate a uniqueness constraint, or a referential integrity constraint among different database tables
  - Appropriate error message when attempting to insert into a nonexistent table or column
  - Appropriate error message when attempting to insert a string value which is too long
- Database queries
  - Typical queries

1. The LIGO_LW format is a LIGO-specific adaptation of XSIL, which is a specific XML document type. (`http://www.cacr.caltech.edu/XSIL`).

  - Queries containing special characters
  - Queries returning no matches
  - Appropriate error messages when query is malformed
  - Appropriate error messages when query refers to nonexistent table or column
- Various aspects of the LIGO_LW data formatting rules when inserting or retrieving data
  - Variations on use of whitespace and delimiters in data stream
  - Valid and invalid names for XML elements
  - Appropriate error messages when input file is malformed
  - Proper handling of null values
  - Proper formatting of special characters (quotes, newlines, XML entities, etc.)
  - Proper handling of all numeric types
  - Range checking and preservation of full precision for numeric values
  - Proper formatting of binary data, including Binary Large OBjects (BLOBs)

About a dozen problems with LDAS software were found by these MDC tests and were subsequently fixed, including an overhaul of the way that special characters are handled in “LIGO lightweight” format, and more robust handling of the names of XML elements.

The fact that this testing is fully automated will allow us to easily verify that each future release of LDAS software preserves all of the current functionality of the database system.

4 DATABASE SYSTEM PERFORMANCE

The functionality tests described in the previous section exercised all LDAS components involved in data insertion and retrieval, and significant emphasis was placed on the LIGO_LW format which is used to represent the data in files. However, astrophysical event candidates found by analysis jobs running on LDAS’s parallel computing cluster will be sent to the Metadata API in LDAS’s internal “ilwd” format, without any format translation needed. Thus, database insertion performance tests have been done starting with test data in “ilwd” format. These tests find that typical event-candidate entries (100-200 bytes) can be inserted into the database at rates of a few hundred entries per second, as long as they are handled in batches of at least 100 at a time. For large database entries, such as power spectra, the insertion speed depends more directly on the total data volume, and sustained rates of 300-400 kB/sec have been measured. These results are well above the nominal insertion rates that have been envisioned for LIGO science running (a few event candidates or environmental transients per second, and a few large spectra per minute), so we are confident that the database system will be able to handle real LIGO data. More quantitative tests are awaiting installation of the final RAID disk storage configuration.

The time required to retrieve data from the database depends greatly on the complexity of the query, and in particular, on whether the records must be sorted. This latter issue becomes particularly important when there are a large number of entries in the database table, unless there is an available index in which the entries are already sorted in the desired order. For instance, in one set of tests using a table of environmental transients with a total of 320,000 entries, a request for the first 10,000 (un-sorted) entries took 19 seconds (including formatting as LIGO_LW), while a request for the first 10,000 entries sorted by time and transient type took 67 seconds. Again, a more comprehensive sent of quantitative tests will be done once the final disk storage hardware is installed.

5 DMT DATA INSERTION PROCEDURE

The LIGO "Data Monitoring Tool" (DMT) consists of computers and software running at the observatory sites, with real-time access to the full LIGO data stream, to continuously monitor the performance of the detectors and to identify environmental transients which could lead to false events. Several "monitor" processes run simultaneously, each receiving data through shared memory and performing a different monitoring task. The DMT software library includes routines to assemble database entries into C++ objects and send them to the "DMT Trigger Manager", which collects database entries from all running monitors, formats them into LIGO_LW files, and periodically sends these files to the LDAS database system for insertion.

This procedure has been exercised during the LIGO "engineering runs". For example, during the E3 engineering run (March 9-12, 2001), the Trigger Manager collected over 50,000 "triggers" from six different monitors running at the two observatory sites and successfully inserted them into the database. During the run, and in subsequent testing, the DMT Trigger Manager and the LDAS database system have demonstrated the ability to run continuously for several days at a time without human intervention.

6 USER INTERFACES AND UTILITIES

The development of user interfaces to the database was deemed to be a critical part of the database Mock Data Challenge. Prior to the MDC, there was only one interface available, called `guild`, which allowed a user to construct a query, submit it to LDAS, retrieve the results, and display them in tabular form. A number of incremental improvements have been made to `guild` since then, and it continues to be an important tool. In addition, a number of other tools have been developed, following the outline set forth in the MDC plan:

- Two simple utilities, `putMeta` and `getMeta`, were created to insert and retrieve metadata from the Unix command line or from within a script. `putMeta` transmits a LIGO_LW file to LDAS and sends the appropriate user command to cause the data to be inserted into the database. `getMeta` sends an SQL query (constructed either by hand or using `guild`'s dialog boxes) to LDAS and retrieves the results either into a local file or to standard output. Both make use of the "data flow manager" (`dfm`), a helper utility which runs in the background and handles all communication with LDAS. The `dfm` may be used to develop other client interfaces in the future.

- A C library called `metaio` was written to parse LIGO_LW files containing metadata tables.[1] The library code is able to parse a typical event-candidate table at a rate of several thousand row per second.

- A few simple command-line utilities were written using the `metaio` library:
  - `lwtscan` displays the names and types of the columns and counts the number of rows
  - `lwtprint` prints selected rows and/or columns in plain ASCII
  - `lwtdiff` reports any differences between the contents of two LIGO_LW files, allowing

1. Although the LIGO_LW format is based on XML, for which a number of parsing packages are available, LIGO_LW imposes certain additional formatting conventions, so customized code is needed to read it.

for formatting variations which do not affect the information content

- The `metaio` library was also used to build a MATLAB "MEX-file", allowing LIGO_LW table data to be read directly into MATLAB arrays for subsequent histogramming, post-selection, plotting, etc.

Finally, the development of these software tools helped to spur on a project called "LIGOtools", which provides a convenient mechanism to distribute useful software to LIGO/LSC collaborating institutions. Thus, all LSC institutions which have installed LIGOtools already have all of the tools listed above, and will easily be able to download updates and/or newly-developed tools as they become available.

7 DATABASE ADMINISTRATION TASKS

During the course of the MDC, a number of different administration tasks have been performed:

- Creating, clearing, and deleting tables
- Creating indexes and triggers
- Adding a table to an existing database
- Adding a column to an existing table
- Deleting selected database entries (keyed by a unique ID assigned to each process which writes to the database)
- Backing up and restoring the entire database

We learned about the limitations on how existing tables can be modified. For example, columns can be added to an existing table, but not deleted; and a fixed-length string column cannot be changed to a variable-length string column (although the maximum length of a variable-length string column can be increased). These will somewhat limit the changes that we can make once the database tables are in active use. However, we have recently modified the LDAS APIs and user commands to support multiple databases simultaneously, so that, if necessary, a "new" database with updated table definitions can coexist with one or more "old" databases.

Notes on database administration procedures are being collected and will be assembled into a set of web pages, to be referenced and updated as needed.

8 DOCUMENTATION

Documentation about LDAS user commands, and formatting rules for LIGO_LW files, are given on the LDAS web site.[1] As part of the MDC, this documentation was reviewed; a number of flaws (e.g. unclear descriptions, or documentation for functionality not yet implemented) were noted and corrected.

1.http://www.ldas-sw.ligo.caltech.edu

Some user interface tools and utilities have built-in documentation. For others, documentation may be found by following links from the LIGOtools web page.[1]

A formal report about the MDC, enumerating the functionality tested and giving the results of quantitative performance tests, will be completed by the end of May 2001.

9 SUMMARY

The LDAS database Mock Data Challenge has accomplished its goals, although it took longer than expected due to manpower limitations. The functionality of the system has been rigorously tested, and the performance seems to be easily adequate for our needs. The DMT and LDAS have demonstrated the ability to run uninterrupted for long periods of time. The database table design is mature, but further revisions are possible as more tables are put into active use and as people come up with additional uses for the database. Several user interface tools are now available, including a communication helper utility and a parsing library. We have gained more experience in database administration.

A detailed report on test protocols, results, and quantitative performance measurements with the final disk storage hardware will be presented in LIGO-T010045-00-E, “Details of the LDAS Database Mock Data Challenge”.

1.http://www.ldas-sw.ligo.caltech.edu/ligotools