

## GEO600 Data Analysis

---

### 1. Scenario Overview

---

#### 1.1 Background and Purpose

The gravitational wave detector GEO600 near Hannover and other ground-based devices (eg. LIGO in the U.S., TAMA in Japan, VIRGO in Italy) aim at the direct detection of gravitational waves by means of a laser interferometer. The detectors constantly generate data (eg. LIGO in the order of 1GBytes/day) that needs to be searched for gravitational wave signals (eg. by filtering the raw input data, running various transformations and check for signals).

#### 1.2 More information

- \* GEO600 Homepage:  
<http://www.geo600.uni-hannover.de/>
- \* The MERLIN Gravitational Waves Cluster:  
<http://pandora.aei.mpg.de/merlin/>
- \* LIGO Homepage:  
<http://www.ligo.caltech.edu/>
- \* LIGO Scientific Collaboration (LSC) Homepage:  
<http://www.ligo.org/>
- \* LSC Data Grid Homepage:  
<http://www.lsc-group.phys.uwm.edu/lscdatagrid/>

---

### 2. Current Scenario description

---

#### 2.1 Environment

##### 2.1.1 Hardware

- Processing  
180 dual-processor AMD Athlon beowulf cluster merlin.aei.mpg.de
- Storage
  - \* 120 GB system and scratch disk per node
  - \* two 120 GB data storage disks per node
  - \* 16 250 GB shared NFS disks in RAID-5 configuration
- Network  
fast ethernet network
- Describe special hardware or other hardware resources that are relevant for the scenario.  
N/A

##### 2.1.2 Software

- Describe used software such as operating system, software libraries, e.g. HDF5-plugin for GridFTP, ...
  - \* Linux RedHat 7.2
  - \* LAM MPI

- \* Globus 3.2
  - \* Condor-G job management system
  - \* Matlab, OpenDX for visualisation
- What programming language is used and what compiler/linker version is required?  
C/C++/Fortran, Java?
  - How is the program deployed?  
\* checkout from CVS, then make
  - How is the program compiled?  
\* predefined makefile
  - State the program license and any commercial 3rd party licenses.  
\* Matlab license

## 2.2 User Interaction

### 2.2.1 Initiation

- Describe how the program is started and any steps needed before the actual initiation.
  - \* user writes a Condor script and submits the job manually
  - \* each job runs on a single processor, independently of the others
- compilation (cf. Section 2.1.2),  
make
- Where is the program executed?  
as a Condor job on the allocated number of nodes
- How is the program initiated?  
from the remote shell command line

### 2.2.2 Monitoring/Steering/Visualization during the run-time of the program

- What type of data is produced by the program during run-time used for monitoring/steering/visualization?  
  
Information about the progress of the program is written to stdout/stderr
- What methods/tools exists for accessing data produced by the program during run-time?  
none, monitoring via logfile access and Condor job return codes
- Does your application support any standard for monitoring/steering?  
no
- Describe any security measures related to program access for monitoring/steering/visualization.  
N/A
- Who can access the running program OR run-time produced monitoring data?  
just the owner (local monitoring only)
- From where can run-time produced monitoring data be accessed?

local monitoring only

- How is the program termination detected?  
via return code of the Condor job
- How much monitoring data and how often is monitoring data transferred during a program run (min/max/avg)?  
N/A
- Does your program generate metadata and stores this externally (e.g. in a catalog)?  
no
- Who accesses this metadata? From where? Does your program access metadata generated by other programs?  
N/A
- How many executions/jobs must be monitored/steered in parallel? By how many users?  
N/A

## 2.3 Input

### 2.3.1 Parameters

parameter file

### 2.3.2 Input data

- How is the input data prepared?  
All data can be directly accessed on the compute nodes via NFS.
- Where is the input data stored? Describe all central and distributed locations.  
Everything is stored on the shared NFS filesystems.
- Are file-names known in advance (before the program is started)?  
yes
- Are data locations (directory, server, ...) known in advance?  
yes
- Describe the different ways data is accessed.  
UNIX file I/O; FRAME I/O library
- Non-file based data access (XML, database, ...) should include description of  
N/A
- How much data is accessed at each run?  
smallest data unit is ca. 40 MBytes  
total amount of read-accessed data depends on the number of timesteps,  
can be  $O(10)$  Gbytes
- Is it possible that a data set/file is accessed multiple times over a short period of time?  
yes

- How many users are using the same data simultaneously?  
one

- Elaborate on the use of metadata related to input data.  
Data and metadata are stored together in the same file.

The FRAME file format is used for that purpose. Possible metadata are timestep, frequency domain, etc.

The directory structure is set up so that input data can be easily classified by

- \* the interferometer
- \* the detector channel
- \* the science run
- \* the timestamp

### 2.3.3 Additional Notes

N/A

## 2.4 Output

### 2.4.1 Output data

- Where is the output data stored? Describe all centralized or distributed locations.  
results are stored on the local filesystem

- How is the output data structured?  
compressed ASCII files

- Describe what happens when the program finishes? How are the results used?  
MatLab scripts and/or OpenDX networks visualise the compressed ASCII files and generate histograms

- Describe the different ways data is created/changed.  
UNIX file I/O

- Non-file based data access (XML, database, ...)  
N/A

- How much data is written by the program at each run?  
not exactly clear but much less than the input data

- Describe the parameters which influence the amount of data and number of files/data sets generated.  
number of timesteps in the time series to analyse, number of frequency domains

- Elaborate on the use of metadata related to output data.  
graphical histograms which can be embedded in webpages

### 2.4.2 Additional Notes

N/A

## 2.5 Information resources

N/A

## 2.6 Data Stream Management

N/A

## 2.7 Resource Security and Access Restriction

Currently merlin users are managed by standard UNIX passwd file.

Experimental remote data access is done via GridFTP using DOE/GermanGrid certificates.

## 2.8 Additional Information

- How long (avg) does the scenario execute (minutes, hours, days)?  
some hours

- How often will the scenario be executed?  
repeatedly for a different set of input parameters

- Are the executions time-critical?  
no

---

## 3. Future Scenario and AstroGrid-D Usage

---

### 3.0 General goals

- use more compute resources if available

The individual analysis tasks are simple enough to be distributed to any computing resources that become available. One has to take care though of transferring the input/output data there.

- provide data to other users

especially the results of a science run should be presented in a collaborative grid environment (eg. a web-based portal)

- completely new scenario

Would be possible if real-time analysis is in place: if a detector finds a (potential) signal, one could trigger other observational devices to focus on that area and maybe discover a supernova explosion

### 3.2 Environment

- Are there any constraints due to your participation in other projects or international collaborations?

Everything has to be compatible to other developments in the LIGO Science Consortium (LSC). This means specifically: Globus 4.x.

### 3.3 User Interaction

- Which parts should be automated?

Standard data analysis finds a certain event which in turn triggers a more detailed analysis of the corresponding data (eg. also take data from other detectors into account).

- Which user interface are you planning to use?

A web-based portal, potentially based on GridSphere.

GAT as potential Grid API to automatically submit jobs, deal with files etc.

- Are you planning to use any standard for application monitoring/steering?  
no

- Aspects of a Portal / WWW based interface:

- . Which portal features are mandatory/optional  
these are all mandatory:
  - \* credential management
  - \* resource/job monitoring
  - \* information service for simulation metadata (browse through all simulations' metadata to find similar parameter settings etc.)
 these are optional:
  - \* job management (start/stop)
  - \* file management
- . How are user managed? Where is information about users defined / stored?
  - \* left open to the portal developers
- . Which authentication/authorisation methods are needed ?
  - \* standard grid authentication/authorisation methods
- . Do you want to access specific data services (web services, databases, etc.) via a portal?
 

N/A
- . Are there any existing programs, on which the user interface should be based OR which should be replaced by the portal?
 

no
- . Should there be a central AstroGrid portal OR do you want to set up a portal server for each scenario/application ?
 

not decided
- . Does the scenario require any special interfaces OR is it sufficient to use generic interfaces ?
 

We do want to use generic interfaces wherever possible. But they must be flexible enough to fit the application's needs.
- Aspects of a generic Grid Application Programming API (GAT)
  - . Which GAT functionality would you like to make use of (eg. job submission, file handling, resource brokering, etc.) ?
    - \* job submission
    - \* remote file access
  - . What programming languages must be supported ? Which platforms ?
 

see 2.1.2. above
  - . Which Grid Middleware should be supported (Globus, Unicore, gLite, etc.) ?
    - \* Globus (currently 3.2)
  - . For specific GAT functionality, which protocols/packages/tools should be supported ?
 

eg. for job management: clusters with PBS, SGE, Condor

    - \* job submission: Condor, PBS, LSF, SGE
    - \* file access: LSCDataFind (see <http://www.lsc-group.phys.uwm.edu/lscdatagrid/LSCGridCamp/Lab4-LSCdataFind.html>)

### 3.4 Input

- Do you handle input data manually or do you need an automated management of data?
  - \* handled by LSCDataFind

### 3.5 Output

- Do you handle output data manually or do you need an automated management of data?
  - \* there are MatLab scripts and OpenDX networks to generate histograms from the output data, these should be automatically stored in a metadata catalogue

### 3.6 Additional Information

- How long (avg) does the scenario execute (minutes, hours, days)? Do you aim at a specific speedup?
  - hours, speedup possible if more resources become available
- How often will the scenario be executed?
  - many times on different input data sets
- Which restrictions of the current approach (as described in section 2) do you want to overcome?
  - \* manual management of output files and visualisation results
  - \* lift the limitation on merlin as the the local compute resource

## 4. Bigger Picture for the far future

### 4.1 Organization of Multiple Runs

### 4.2 Handling relationships between data products

### 4.3 Constructing More Complex Runs

(almost) real-time analysis of detector data (required when the space-based detector LISA comes into operation)