Robotic Telescopes

1. Scenario Overview

Give an introduction to the scenario, making the reader familiar with goals, main purpose and functional overview.

1.1 Background and Purpose

Robotic telescopes should be considered as 'normal' GRID resources. The main difference to standard GRID job is their strong requirement on schedules at pre-defined times, e.g. not 'do job A as soon as possible', but rather 'do job A now (at time t) or forget it'.

1.2 More information

http://www.aip.de/stella

2. Current Scenario description

Describe the current scenario including as much details as possible.

Note, there is no special section to describe workflows/pipelines or details about a phased execution of a program. If your scenario is a workflow/pipeline OR your application is executed in several phases, describe EACH step/part covering the sections 2.1 - 2.8. In addition you must describe how these steps/parts are interrelated to eachother (in Section 2.8).

2.1 Environment

2.1.1 Hardware

Describe the hardware resources that are currently used.

- Processing
  robotic telescope with four different, low-end control computers

- Storage
  local RAID5 (~1TB) for scientifice data, mirrored daily to similar RAID at AIP. Crucial
  control data (weather, logs, soon target descriptions) on a local and mirrored PSQL db.

- Network
  Requirements rather low, as only target descriptions are to be transfered for control (less than 100k). Scientific data envisioned at a rate of 1 GB/day

- Describe special hardware or other hardware resources that are relevant for the scenario.
  Needs a telescope attached to it ;-). Implementations differ widely, so the only possibility of exchange is clinging to a certain standard for target description. This was/is discussed in the Heterogeneous Telescope

Network group (http://www.telescope-networks.org/wiki/index.php/Main_Page).

2.1.2 Software

- Describe used software such as operating system, software libraries,
  e.g. HDF5-plugin for GridFTP, ...
  . list non-standard software with min. version
  Too many to list

- What programming language is used and what compiler/linker version
  is required?
  Java 1.2 and higher, gcc 3.3.x

- How is the program deployed?
  source code as well as pre-compiled binaries plus jar extension packages.

- How is the program compiled?
  make

- State the program license and any commercial 3rd party licenses.
  Not finally descided yet, but will be GNU. Use of non-GNU 3rd party extensions
  is avoided whenever possible.

2.2 User Interaction

Describe the user interaction necessary for starting the program and
additional interaction with a running program.

2.2.1 Initiation

- Describe how the program is started and any steps needed before the
  actual initiation.
  The user only submitts standard observational request, no on-site compilation
  is allowed.

- compilation (cf. Section 2.1.2),
  retrieve/copy/generate data/files (cf. Section 2.3.2),
  parameters (cf. Section 2.3.1)

- Where is the program executed?
  at a robotic telescope

- How is the program initiated?
  Currently only authorized operators may upload XML target definitions to the
local
  target directory on the master computer at STELLA, interaction with the
  control computers is
  currently by java RMI, will be updated to SOAP. For GRID use, a tight
  authentication is required.

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

  Definitions:
  Monitoring - Information retrieval regarding the state of the
    program. For example, "Program is running" or more application
    specific information such as "Current simulation iteration is 42".

  Steering - Remote alteration of the programs state. For example,
    program stop or application specific information like "at

iteration 42, set parameter x = 78".

Visualization - Remote access of the application data needed for visualization of, for example, a simulation.

- What type of data is produced by the program during run-time used for monitoring/steering/visualization?
  . log-files, stdout/err, intermediate results, ...
  -) ASCII log files, put in a local db and to syslog
  -) Errors/warnings during execution are also accessible via RMI (register as error listener).
  -) Weather data put in local db. Also accessible via RMI (register as weather listener)
  -) Status of telescope and instruments accessible via RMI
  -) Acquisition and guiding frames (i.e. what the telescope sees during data acquisition) are stored locally (will be mirrored to AIP soon)

- What methods/tools exists for accessing data produced by the program during run-time?
  -) RMI for status, weather, errors,
  -) http for acquisition/guiding frame retrieval.
  for automated monitoring/steering we'd need
  -) weather data and current status of telescope
  -) a statical 'phase 0' description of the telescope, but this can be compared rather with hardware requirements (i.e. instead of GRAPE boards specific instrumentation (Spectrograph, Imager, Photometer, etc.) This is much mor complex than in a standard GRID case!

- Does your application support any standard for monitoring/steering?
  no

- Describe any security measures related to program access for monitoring/steering/visualization.
  Entire system firewall protected, ssh-access limited (ask M. Weber, mweber@aip.de for details if necessary).

- Who can access the running program OR run-time produced monitoring data?
  weather data and phase zero data should be public to anyone. Science data products are
  currently propriety of the PI for a three year period. This may change, nevertheless data access will require some sort of confirmation.
  Access to the running program is currently only granted to the operator.

- From where can run-time produced monitoring data be accessed?
  . specific IP/netmask, anywhere, ...
  Currently specific IP/netmask, but non-crucial data should be accessible anywhere

- How is the program termination detected?
  e-mail notification of PI. Errors are handled internally and reported to the operator and/or PI if unresolvable for the system.

- How much monitoring data and how often is monitoring data transferred during a program run (min/max/avg)?
  . X MB every time a buffer is full, X GB when the program finish, ...
  . all data is transferred, partial data is transferred, last N records are transferred, ...
  Weather data is roughly only 30 number/5mins

Science data amounts to 8MB/observation (~8MB/10min) for STELLA-I and to 32MB/observation (shorter exposures, thus 32MB/min) for STELLA-II
All of the data is transferred to the AIP db ASAP.

- Does your program generate metadata and stores this externally (e.g. in a catalog)?
  . when [start, end, periodically, specific events]?
  . where is the catalog?
  . how is it accessed [protocol/API, access controlled]?
  -) Weather data stored in PSQL db at a regular rate of 5 min.
  -) Metadata of the scientific exposures stored with keyword in the FITS-headers of the data files

- Who accesses this metadata? From where? Does your program access metadata generated by other programs?
  -) Weather data access by php script and displayed on webpage.
  -) Fits headers currently unprocessed, but will be relevant for data reduction.

- How many executions/jobs must be monitored/steered in parallel? By how many users?
  -) There is always only one job running at an robotic telescope. Which one is to be executed is determined by dispatch-scheduling software, which picks a target out of a pool provided by the users.

2.3 Input

2.3.1 Parameters

Describe the program parameters in detail.
All relevant data is packed into an XML file (see attached example)

2.3.2 Input data

- How is the input data prepared?
  Can be prepared with any XML-editor. Tools (command line and GUI) exist for helping a potential user to prepare a correct file (these tools access the SIMBAD services via SOAP)

- Where is the input data stored? Describe all central and distributed locations.
  Kept at the AIP as an XML-file and transferred currently via scp to the STELLA master computer. Only targets on site and marked as valid in an input list are processed. Database storage planned.

- Are file-names known in advance (before the program is started)?
  . Could, for example, be calculated on-the-fly using a metadata catalog?
  Yes

- Are data locations (directory, server, ...) known in advance?
  . Could, for example be calculated on-the-fly using a metadata catalog?
  Yes

- Describe the different ways data is accessed.
  . POSIX read, XPath, SQL, ...
  log into the local data archive at AIP and copy the relevant science data files to your working directory.

- Non-file based data access (XML, database, ...) should include
  description of
  . name of the database management system,
  . how the database is accessed (ODBC, JDBC, WWW interface, command
    line, ...),
  . typical access patterns (bursty, continuous, ...),
  . physical location/distribution (local, externally, ...),
  . possibility to replicate the data through some mechanism,
  . any security related restrictions,
  . are user-defined Stored-Procedures used,
  . types of indices used (e.g. Hierarchical-Triangular-Mesh (HTM))
  . ...
  PSQL db, access vi JDBC, WWW, cl. Typical access continuous. Located on site
  and mirrored at AIP. Currently, only authorized users should see the science
  data.

- How much data is accessed at each run?
  . number of files/data sets: min/avg/max,
  . total-size: min/avg/max,
  . retrieved-size: min/avg/max
  No files, all targets kept locally in memory after parsing from XML input
  files.

- Is it possible that a data set/file is accessed multiple times over a short
  period of time?
  . For example by different "threads" of the program. Then, replicating and/or
    caching might be interesting.
  Currently not possible as any user has his local copy.

- How many users are using the same data simultaneously?
  Are these users geographically distributed?
  Normally only one.

- Elaborate on the use of metadata related to input data.
  . amount,
  . how it is accessed,
  . security restrictions,
  . metadata format [key/value ?],
  . life-cycle: creation, usage time, used by multiple program
    runs, deletion, ...
  Hm, if I understand that question correctly, I'd say for the input data the
  following things apply:
  -) Weather. Used to choose the target or to stop/start observations. Access
  via RMI, no special security restrictions. Key/value type. Are updated
  each second (for general purpose and storage 5min averages are used).
  -) Previous observations of the same target, needed for proper scheduling.
  This can also include expelling a target from the current observing list
  because of too many observing failures. This information follows either
  from the input file or from previous runs of the target and is kept in memory.
  Access similar to key/value.

2.3.3 Additional Notes

Describe any additional information regarding the input data which
has not yet been covered.

2.4 Output

This covers what data products are generated (INTERMEDIATE and FINAL results),

where they are generated and how they are handled after the program finished (transferring data or removing it, ...).

2.4.1 Output data

- Where is the output data stored? Describe all centralized or distributed locations.
  . local file system, remote file server, RAID, Database, ...
  local 1TB raid, mirrored to AIP.

- How is the output data structured?
  . single file, distributed file, database table(s)...
  . data formats: plain text, HDF5, FITS, VOTable, proprietary, ...
  Single file per observation, FITS format.

- Describe what happens when the program finishes? How are the results used?
  . remains at the output location,
  . moved/copied/deleted elsewhere [manually/automatically],
  . used as input to a subsequent call or to another program, ...
  keep at local RAID until full, copied automatically to mirror at AIP. Used as input to a reduction pipeline which runs at the AIP. The output of this pipeline is writtn in the AIP archive. The output of this pipeline is normally accessed by the scientist, but he/she might as well get the raw data, if desired.

- Describe the different ways data is created/changed.
  . POSIX write, XPath, SQL, ...

- Non-file based data access (XML, database, ...) should include description of
  . name of the database management system,
  . how the database is accessed (ODBC, JDBC, WWW interface, command line, ...),
  . typical create patterns (bursty, continuous, ...),
  . physical location/distribution (local, externally, ...),
  . possibility to replicate the data through some mechanism,
  . any security related restrictions when data is written,
  . ...
  Will be PSQL based, currently only files.

- How much data is written by the program at each run?
  . size: min/avg/max,
  . number of files/data sets: min/avg/max
  1 file per run per telescope, 8MB STELLA-I, 32 MB STELLA-II, uncompressed.

- Describe the parameters which influence the amount of data and number of files/data sets generated.
  time of clear skies and the exposure time per target.

- Elaborate on the use of metadata related to output data.
  . amount,
  . how it is accessed,
  . security restrictions,
  . metadata format [key/value ?],
  . life-cycle: creation, usage time, used by multiple program runs, deletion, ...

  Note, the decision where results are stored is supported by

information about the further use or free data storage.

All relevant observing parameters are stored as FITs header key/value pairs.
What parameters are relevant is to be decided...

2.4.2 Additional Notes

Describe any additional information regarding the output data which
has not yet been covered.

2.5 Information resources

Give a summary of each information resource that is accessed by
the program. Include information about data input/output, locations,
access methods (XQuery, SQL, ...), security related restrictions,
search of metadata (exact key search i.e. "ABC", range queries
i.e. "AB*", ...)

Does not apply.

2.6 Data Stream Management

Definitions:

Data Stream - intermediate results can be processed by the
subsequent processing module before the current module has
processed the last element of the input.

- Can single operations be performed on any compute node or do they
need special hardware or software?
Does not apply

- Are data exchanged between distributed parts of the application?
does this happen at the beginning, during run-time or at the end?
Applies only for the limitied case that prior observations of a target may
influence further scheduling.

- Are operations compute intensive?
No

2.7 Resource Security and Access Restriction

Describe all security related information that considers access
of resources. User based, Group based, by IP-address/netmask,
certificates, nodes/resources within a private network, firewall
restrictions, ...

Direct access to the resource currently limited to a single operator (user)
using ssh. Local firewall permits certain IP-addresses.

2.8 Additional Information

Give additional information not covered by the sections above.

- How are workflow/pipeline steps interrelated to eachother?

Reductional pipelines used on observational data, but this is not part of this
proposal
(more a VO thing).

- Is the application executed in several phases where each phase
  may have different resource requirements or may be executed at
  a different resource?
  Single observations can be grouped together to form a science goal. Only in special circumstances a shift from one telescope to an other is useful once operation has started at a certain telescope (different instrumentation make interpretation of data a nightmare or even impossible). But shifting jobs between telescopes prior to execution would be the ultimate goal, as than weather constraints can be overcome.

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

  Between, say, 60 sec and 1 hour.

- How often will the scenario be executed?

  Depends on the science program. Programs may span over years (!)

- Are the executions time-critical?

  yes!!!

---
3. Future Scenario and AstroGrid-D Usage

Describe the future scenario and envisioned usage of AstroGrid-D
as detailed as possible. It is not assumed that the questions can
be answered as detailed as in Section 2. Focus on what is expected
by the Grid environment and how this new functionality can be used.

Note, there is no special section to describe workflows/pipelines or
details about a phased execution of a program. If your scenario is a
workflow/pipeline OR your application is executed in several phases,
describe EACH step/part covering the sections 3.1 - 3.5. In addition
you must describe how these steps/parts are interrelated to eachother
(in Section 3.6).

3.0 General goals

Such as:
- use more compute resources,
- use other data resources,
- provide data to other users,
- completely new scenario,
- overcome deficiencies of current approach,
- ...

Linking robotic telescopes provide two major advantages that can NEVER be met by single robotic telescopes: Instant reaction and arbitrary long observations of the same target. While instantaneous observation of a transient object might be possible with a single telescope (if you are lucky and the location of the sky in question is above the horizon, you are at night-time and the weather and your telescope is
good - these restriction amount to a only 10% possibility of a hit), continuous observations of a single target are not possible for longer than a single night on a given location. This possibility can only be granted by a cooperating network of robotic telescopes, as observing time on satellites is much too

valuable to be spent on a single object (but not the exception of the HST deep-field).

3.2 Environment

- Are there any constraints due to your participation in other projects or international collaborations?
  . specific Grid middleware, hardware, standards, virtual organizations
  The AIP is part of the heterogeneous telescope network. There is no use in not following their standards, which are (heterogeneous!) the least common denominator.

3.3 User Interaction

- Which parts should be automated?
  . resource selection,
  . data transfer before initiation and after termination,
  In a first step, resource selection can be based on phase-0 descriptions (is this telescope able to process my request), later metadata (i.e. weather, but also availability of the telescope) should be incorporated in a request ala 'Can you do target x now', or 'How high is the probability that you can do x within the next hour'.

- Which user interface are you planning to use?
  . WS [SOAP], API, WWW portal, ...
  xml-documents transported via SOAP.

- Are you planning to use any standard for application monitoring/steering?
  . Do you want to use such standards in collaboration with the DGI
    or the other communities OR will you develop your own methods?
  currently I lack the knowledge on these, but if they are applicable to our case, why not.

- Aspects of a Portal / WWW based interface:
  . Which portal features are mandatory/optional (e.g. credential management, job management, job monitoring/steering, data transfer, ...)?
    credential management is needed, job monitoring & data transfer would be nice

  . How are user managed? Where is information about users defined / stored?

    Not specified (NS).

  . Which authentication/authorisation methods are needed ?

    NS.

  . Do you want to access specific data services (web services, databases, etc.) via a portal?

    Database access.

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

    Input file generation is specific, thus a simple upload is sufficient.

  . Should there be a central AstroGrid portal OR do you want to set up a portal server for each scenario/application ?
    NS.

. Does the scenario require any special interfaces OR is it sufficient to use generic interfaces ?

see above

- 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.) ?
NS.

. What programming languages must be supported ? Which platforms ?

Does not apply.

. Which Grid Middleware should be supported (Globus, Unicore, gLite, etc.) ?

Does not apply.

. For specific GAT functionality, which protocols/packages/tools should be supported ?
eg. for job management: clusters with PBS, SGE, Condor

Does not apply.

3.4 Input

- Do you handle input data manually or do you need an automated management of data?

Currently manual is sufficient.

3.5 Output

- Do you handle output data manually or do you need an automated management of data?

Data mirroring sufficient.

3.6 Additional Information

- How long (avg) does the scenario execute (minutes, hours, days)? Do you aim at a specific speedup?

No speedup possible.

- How often will the scenario be executed?

see above

- Which restrictions of the current approach (as described in section 2) do you want to overcome?

Network for new science.

4. Bigger Picture for the far future

4.1 Organization of Multiple Runs

Maintain a list of all simulations, to repeat simulations with a different binary, with different input data, to check if a program was already executed with a certain set of parameters/input data, ...

    Organize long-term continuous monitoring of a single target, spanning arbitrary long periods.

4.2 Handling relationships between data products

For example, store metadata on how a data product was generated (from which input data, by which program, with which parameters) and how it can be used by others.

4.3 Constructing More Complex Runs

For example, combine existing single programs.

An email exchange clarifying some issues. Initial file (>>, also see use case above) from T. Granzer, questions (>) from T. Roeblitz, answers from T. Granzer:

>>2.1.1 Hardware
>>
>>- Storage
>>   local RAID5 (~1TB) for scientifice data, mirrored daily to similar
>>   RAID at AIP. Crucial
>>   control data (weather, logs, soon target descriptions) on a local and
>>   mirrored PSQL db.
>>
>>
>
> How much data is mirrored? 1 TB? Is data (observation results) kept at
> the place of the telescope OR is it moved (rather than mirrored) to AIP?
>
> If it would be mirrored, after ~1000 days the RAID5 would be full. More
> important, more and more data would be mirrored or does the mirroring
> only consider new data.
>
>
>
The 1Tb local raid is used as a buffer until data is finally transferred to the archive at AIP. So, maybe this is not called mirroring, but this is the envisioned process:
As soon as data is taken, it is stored on the local RAID. If internet connection permitts, data is transferred to the AIP archive. If it arrives there, local data will be ereased if necessary.

>>2.1.2 Software
>>
>>- Describe used software such as operating system, software libraries,
>> e.g. HDF5-plugin for GridFTP, ...
>> . list non-standard software with min. version
>> Too many to list
>>
>>

>
> Ok, but which OS is used at the control nodes?
>
>
>
>
Debian Linux, main programming language will be java, with some C/C++
for basic I/O (read out a CCD).