AstroGrid-D

Deliverable D2.9

Advanced Prototype Implementation of Metadata Information Providers

Documentation and Test Report[1]

Deliverable	D2.9
Authors	Working Group WG-II
Editors	Stephan Braune, Frank Breitling, Arthur Carlson, Mikael Högqvist, Thomas Radke, Tobias Scholl, Steve White
Date	18 August 2008
Document Version	1.0.1
Current Version	1.0.1
Previous Versions	

A: Status of this Document

Public release of deliverable D2.9.

B: Reference to project plan

This deliverable document refers to the task TA II-4 *"Entwicklung von Information-Providern und die Bereitstellung der Metadaten"* of work package WP-2 in the project work plan.

[1]This work is part of the D-Grid initiative and is funded by the German Federal Ministry of Education and Research (BMBF).

C: Abstract

This document describes the advanced prototype implementation of metadata information providers used by various AstroGrid-D use cases, available as of project month 36 (August 2008).

D: Change History

Version	Date	Name	Brief summary
0.9.0	10 March 2008	Mikael Högqvist	Migrated from D2.6.
0.9.3	15 July 2008	Frank Breitling	Description of Telescope Timeline and Telescope Map.
0.9.5	16 July 2008	Mikael Högqvist	Version updates.
1.0.0	28 July 2008	Mikael Högqvist	Introduction updates.
1.0.1	18 August 2008	Mikael Högqvist	Incorporated comments and released to the public.

E:

Contents

1 Introduction

A wide variety of meta-data is produced and used in the AstroGrid-D project, and there are various ways it can be analyzed. One useful distinction is between meta-data related to the grid itself (hardware and services), and those related to a specific application. When the data is aggregated, some questions, such as which jobs have used how much compute time on which nodes, can be answered in principle by either approach. Another dimension is the volatility of the information. Some information is essentially static (How many nodes does an installation have?), some are meaningful only in real-time (What is the estimated completion time of a running job?), and some are inbetween (Is it cloudy today at my telescope site?). AstroGrid-D is working on use cases that cover various combinations of these characteristics. A rough classification is given in the following table, although some information providers have characteristics of more than one group. (The Data Streams use case was described in Deliverable 2.3 and has not changed since then.)

Table 1: AstroGrid-D information providers classified by orientation to the grid or to an application and by the volatility of the data

	grid-oriented	application-oriented
static	Resource Meta-data GridMap	
variable	Grid Service Monitoring Data Streams	Cactus Integration Tests Robotic Telescopes
real-time	Job Monitoring GEO600 Job Statistics	Cactus Simulation Monitoring and Steering

The initial state of the information providers described in Section 2, have been described in Deliverable 2.3[1] and 2.6[2]. The following section summarizes the current state of the use cases, with emphasis on progress made since the last report. This document indicates the state of the information providers in the final month of the project. Most of the information providers use the AstroGrid-D information service Stellaris described in Deliverables 2.1[3] and 2.2[4] for storage and extraction of metadata.

2 Metadata Information Providers in AstroGrid-D

2.1 Resource Metadata

Status data on Grid resources using the Globus Toolkit are provided by the Monitoring and Discovery Service (MDS). Information about each computing element (CE) - a cluster or a workstation - is contained in MDS using the Grid Laboratory Uniform Environment (GLUE) schema. The current implementation of MDS in GT 4.0.x uses GLUECE schema version 1.1. The schema is filled with reasonable data by information providers like the Ganglia Information Provider which should be configured on all AstroGrid-D resources. More information about MDS can be found in [5].

Currently the AstroGrid-D metadata can be viewed by means of a modified Globus WebMDS service running on the main GACG server. A listing of Ganglia information from Grid resources is generated by visiting the Web URL

`http://www.gac-grid.org/project-products/grid-status/astrgrid-overview.html`

and a listing of Web Services provided by these resources is at

`http://www.gac-grid.org/project-products/grid-status/astrogrid-mds.html`

The modifications to the Globus distribution used to make these listings may be found in SVN under `software/globus/MDS/WebMDS/`.

In order to provide this status information to the AstroGrid-D information service Stellaris, we established a periodically applied translation chain (cron job) from MDS' XML schema into RDF containing all the information from the individual GLUECE sub-schemes in the MDS. As the first step, the XML data are retrieved from the AstroGrid-D WebMDS-Service. Usually, the XML output of MDS contains several copies of a GLUECE schema per individual computing element in conjunction with each available queue, etc. By applying an XSL-Transformation, a new XML file is extracted which contains the collection of unique GLUECE schemes only (one per computing element).

The translation from XML to RDF is performed using the XML2RDF routine from the OwlMap package

2.2 Grid Service Monitoring

In order to monitor the availability and status of basic Grid services for individual users on registered AstroGrid-D resources, a grid service monitoring script `gm.pl` had been developed by work package WP-I *"Integration of Compute and Data Resources into GACG"* during the build-up phase of the still emerging AstroGrid-D testbed. The script proved very useful in testing the proper deployment of Grid services such as gsissh, GridFTP, and GRAM on resources added newly to the testbed. It could also be used to locate both user-specific problems (e.g., missing entries in a gridmap file) and machine-related issues (e.g., wrong firewall setups).

With the completion of the AstroGrid-D testbed, along with a solidification of the functional availability of basic grid services, the need for such a specific service monitoring script became less important. It was therefore decided not to develop its current version (as of May 2007) any further. The implementation of the script, its interaction with AstroGrid-D's metadata information service, and the web-based user interface for metadata queries have not been changed since then; for a description of these please refer to deliverable D2.3[1].

2.3 Job Monitoring

The monitoring of compute jobs within AstroGrid-D has started with the introduction of WS GRAM Audit logging [6] as part of the Globus Toolkit release 4.0.5. Also the SGAS [7] solution for monitoring has been investigated. Both systems provide information about Globus jobs. Since SGAS does not provide information about running but about finished jobs only, GRAM Audit logging was adopted as the standard for job monitoring in AstroGrid-D.

The information provided by Audit logging includes:

- ob_grid_id (String representation of the resource EPR (endpoint reference)),
- local_job_id (Job/process id generated by the scheduler),
- subject_name (Distinguished name (DN) of the user),
- username (Local username),
- idempotence_id (Job id generated on the client-side),
- creation_time (Date when the job resource is created),
- queued_time (Date when the job is submitted to the scheduler),
- stage_in_grid_id (String representation of the stageIn-EPR (RFT)),
- stage_out_grid_id (String representation of the stageOut-EPR (RFT)),
- clean_up_grid_id (String representation of the cleanUp-EPR (RFT)),
- globus_toolkit_version (Version of the server-side GT),
- resource_manager_type (Type of the resource manager (Fork, Condor, ...)),
- job_description (Complete job description document),
- success_flag (Flag that shows whether the job failed or finished successfully),
- finished_flag (Flag that shows whether the job is already fully processed or still in progress).

The configuration of GRAM audit logging is short. Only the Globus configuration files `$GLOBUS_LOCATION/container-log4j.properties`, and `$GLOBUS_LOCATION/etc/gram-service/jndi-config.xml` have to be changed, as described in [6], and [8]. GRAM audit logging writes job meta data into a PostgreSQL[9] database. Different Globus machines can share the same database. For each new job Globus creates a new database record. This record is updated whenever the job status changes.

Each entry about a job is transfered to Stellaris via database triggers. The database trigger is a perl program which is executed whenever a new record or an update is received by the audit database. The trigger creates an XML file from the new metadata which complies with the usage record format [10] and sends it via an HTTP PUT request to Stellaris. Listing 1 shows part of the trigger source code.

Listing 1: Trigger source code (shorted) for data transfer from audit database to Stellaris

```
CREATE OR REPLACE FUNCTION update_stellaris() RETURNS trigger AS
    $update_stellaris$

use strict; use POSIX; use URI; use Net::hostent; use HTTP::Request; use LWP::
    UserAgent;
my $t = time(); my @now = localtime($t);
my $now_ = sprintf("%.4d-%.2d-%.2dT%.2d:%.2d:%.2d",
   1900+@now[5],1+@now[4],@now[3],@now[2],@now[1],@now[0]);
my $job_grid_id = URI->new($_TD->{new}{job_grid_id});
my $id = unpack("H*", $job_grid_id->query());
my $host = gethost($job_grid_id->host())->name();
my $status = ""; my $count = ""; my $exe = ""; my $end_time = "", my
    $wall_duration = "";

my $tc = $_TD->{new}{creation_time}; $tc =~ s/ /T/;
my $tq = $_TD->{new}{queued_time}; $tq =~ s/ /T/;
if ($_TD->{new}{finished_flag} eq "t" && $_TD->{old}{finished_flag} eq "f") {
  my $q_t = $_TD->{new}{queued_time};
  $q_t =~ s/-/ /g; $q_t =~ s/:/ /g; my @qt = split(/ /,$q_t);
  my $queued_time = POSIX::mktime(@qt[5],@qt[4],@qt[3],@qt[2],@qt[1]-1,@qt
      [0]-1900);
  $end_time = $now_;
  $wall_duration = sprintf("%.4f",($t-$queued_time)/3600.);
  $_TD->{new}{end_time} = sprintf("%.4d-%.2d-%.2d %.2d:%.2d:%.2d",
    1900+@now[5],1+@now[4],@now[3],@now[2],@now[1],@now[0]);
  $_TD->{new}{wall_duration} = ($t-$queued_time)/3600.;}
if ( $_TD->{new}{finished_flag} eq "f") {
       if ( $_TD->{new}{queued_time} eq "" ) { $status = "queued"; }
       else { $status = "started"; }}
else { if ($_TD->{new}{success_flag} eq "f") { $status = "failed"; }
       else {$status = "completed"; } }
if (index($_TD->{new}{job_description}, "<ns1:count>") > 0) {
  $count = $_TD->{new}{job_description};
  $count =~ s/.*<ns1:count>//; $count =~ s/<\/ns1:count>.*//;
  $_TD->{new}{count} = $count;}
if (index($_TD->{new}{job_description}, "<ns1:executable>") > 0) {
  $exe = $_TD->{new}{job_description};
  $exe =~ s/.*<ns1:executable>//; $exe =~ s/<\/ns1:executable>.*//;}

my $usage_record = <<"EOF";
<?xml version="1.0"?>
<JobUsageRecord xmlns="http://www.gridforum.org/2003/ur-wg#"
                xmlns:grddl="http://www.w3.org/2003/g/data-view#"
                xmlns:urf="http://schema.ogf.org/urf/2003/09/urf"
                grddl:transformation="http://www.astrogrid-d.org/xml2rdf.xsl">
  <RecordIdentity urf:recordId="$_TD->{new}{job_grid_id}"
                  urf:createTime="$now_"/>
  <JobIdentity>
    <LocalJobId>$_TD->{new}{local_job_id}</LocalJobId>
    <job_grid_id>$_TD->{new}{job_grid_id}</job_grid_id>
    <idempotence_id>$_TD->{new}{idempotence_id}</idempotence_id>
  </JobIdentity>
  <UserIdentity>
    <LocalUserId>$_TD->{new}{username}</LocalUserId>
    <GlobalUserName>$_TD->{new}{subject_name}</GlobalUserName>
  </UserIdentity>
  <Status>$status</Status>
  <creation_time>$tc</creation_time>
  <StartTime>$tq</StartTime>
  <EndTime>$end_time</EndTime>
  <WallDuration unit="hours">$wall_duration</WallDuration>
  <Host>$host</Host>
  <Queue>$_TD->{new}{resource_manager_type}</Queue>
  <Count>$count</Count>
  <Resource urf:description="application">$exe</Resource>
  <success_flag>$_TD->{new}{success_flag}</success_flag>
  <finished_flag>$_TD->{new}{finished_flag}</finished_flag>
</JobUsageRecord>
EOF

my $req = HTTP::Request->new("PUT",
 "http://stellaris.zib.de:24020/files/hosts/".$host."/urs/".$id,
  HTTP::Headers->new(Content_Length => length($usage_record)), $usage_record);
my $ua = LWP::UserAgent->new(); my $res = $ua->request($req);
if ($res->is_error) {print STDERR "ERROR:  ".$res->status_line."\n";} return "
    MODIFY";
$update_stellaris$ LANGUAGE plperlu;

CREATE TRIGGER update_stellaris_trig
    BEFORE INSERT OR UPDATE ON gram_audit_table
    FOR EACH ROW EXECUTE PROCEDURE update_stellaris();
```

Base on the GRDDL [11] specification Stellaris transforms the usage record format into RDF/XML using an XSLT.

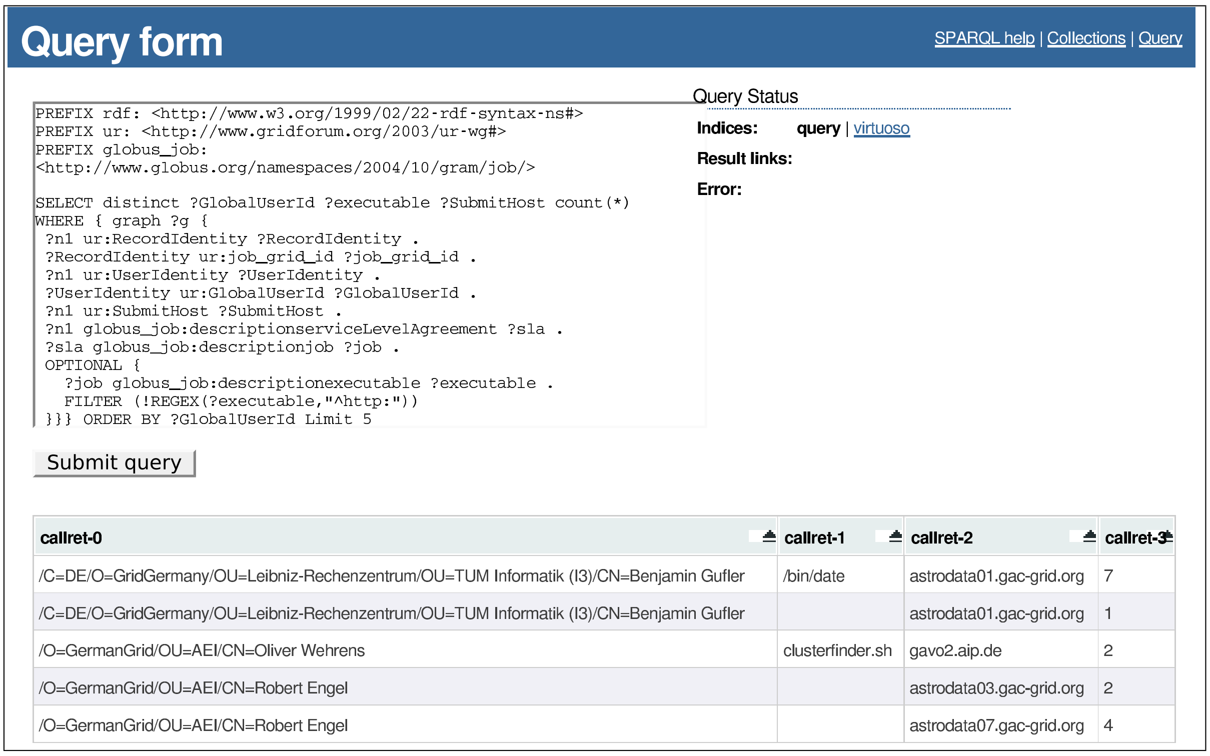
GRAM audit logging is currently tested on the Astrodata cluster and on several desktop machines at the AIP. An overview of AstroGrid-D jobs can be obtained via SPARQL query shown in Fig. 1.

2.4 GEO600 Job Statistics

AEI's GEO600 use case[12, 13] – analysis of GEO600 gravitational wave detector data – has been running continuously since September 2007 as an AstroGrid-D production-mode Grid application on a large number of computational resources both in the AstroGrid-D testbed as well as on the D-Grid itself. At any moment several hundred data analysis jobs are running on the Grid, each taking a runtime of 8-16 hours. A script, deployed on AEI's grid server machine `buran.aei.mpg.de`, is periodically checking the status of all these jobs and will automatically submit new ones when a presently running job has finished. All job-related status information is managed in a MySQL database on `buran`; it gets updated by the job submission script as well as by the jobs themselves as soon as their status changes (e.g., from *queued* to *running* to *terminating*).

Since the MySQL database is keeping a record of all currently submitted GEO600 jobs, it can also be used to gather the following job statistics information:

- How many jobs have been submitted so far by a given Grid user / to a given Grid resource ?
- How much CPU time was spent in total by that user / on that resource ?
- How many jobs are currently in *active*, *pending*, or *suspended* state ?



```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX ur: <http://www.gridforum.org/2003/ur-wg#>
PREFIX globus_job:
<http://www.globus.org/namespaces/2004/10/gram/job/>

SELECT distinct ?GlobalUserId ?executable ?SubmitHost count(*)
WHERE { graph ?g {
 ?n1 ur:RecordIdentity ?RecordIdentity .
 ?RecordIdentity ur:job_grid_id ?job_grid_id .
 ?n1 ur:UserIdentity ?UserIdentity .
 ?UserIdentity ur:GlobalUserId ?GlobalUserId .
 ?n1 ur:SubmitHost ?SubmitHost .
 ?n1 globus_job:descriptionserviceLevelAgreement ?sla .
 ?sla globus_job:descriptionjob ?job .
 OPTIONAL {
   ?job globus_job:descriptionexecutable ?executable .
   FILTER (!REGEX(?executable,"^http:"))
 }}} ORDER BY ?GlobalUserId Limit 5
```

callret-0	callret-1	callret-2	callret-3
/C=DE/O=GridGermany/OU=Leibniz-Rechenzentrum/OU=TUM Informatik (I3)/CN=Benjamin Gufler	/bin/date	astrodata01.gac-grid.org	7
/C=DE/O=GridGermany/OU=Leibniz-Rechenzentrum/OU=TUM Informatik (I3)/CN=Benjamin Gufler		astrodata01.gac-grid.org	1
/O=GermanGrid/OU=AEI/CN=Oliver Wehrens	clusterfinder.sh	gavo2.aip.de	2
/O=GermanGrid/OU=AEI/CN=Robert Engel		astrodata03.gac-grid.org	2
/O=GermanGrid/OU=AEI/CN=Robert Engel		astrodata07.gac-grid.org	4

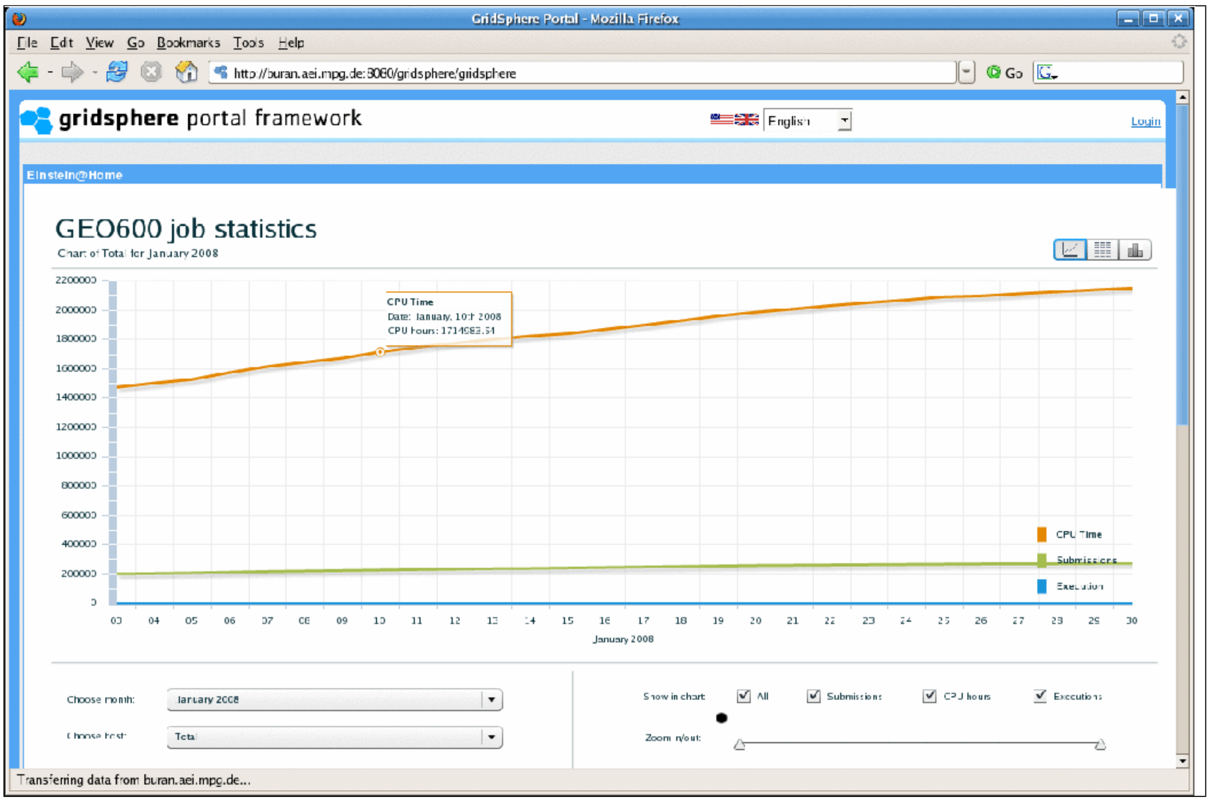
Figure 1: Snapshot of the *AstroGrid-D job statistics* page from a prototype version with a SPARQL Query.

A perl script was written which automatically extracts this statistical information at periodic intervals (every night at midnight), translates it into an RDF/XML document, and sends it off to an AstroGrid-D metadata information service. For the RDF/XML translation a proprietary RDF schema was defined to describe the GEO600 job statistics data. In this schema each record of *submitted, active, pending, suspended* jobs along with the total CPU time is associated either with a user name (the distinguished name of the user as obtained from her Grid proxy) or a resource name (the full name of the Grid resource to which this set of jobs has been submitted). In order to uniquely distinguish successive snapshots of job statistics metadata, each RDF/XML document also gets associated the timestamp of its creation.

Ultimately, users should be able to query the GEO600 job statistics metadata again, using an standardised, intuitive and easy-to-use web interface such as a portal. In collaboration with the D-MON project[14] and with working group WG-VII *"User and Programming Interfaces"* a GEO600 portlet was developed for this very purpose. It can dynamically generate specific SPARQL queries and send them to the AstroGrid-D metadata information service. The results are presented to the user in various ways, both graphical and textual formats are possible. As an example, Fig. 2 shows a snapshot of GEO600 job statistics information presenting the query results of submitted and currently executed jobs, along with the total CPU time spent, as a timeline for the month of January 2008. The portlet provides submenus to select different months or to switch from displaying accumulated sums across all resource to showing results for an individual Grid resource only. According to these user-defined selection parameters, an equivalent SPARQL query would be generated by the portlet, such as the following:

Listing 2: The SPARQL query corresponding to the information displayed in Fig. 2.

Figure 2: Snapshot of the *GEO600 job statistics* page from a prototype version of a GEO600 GridSphere portlet



```
PREFIX xsd:    <http://www.w3.org/2001/XMLSchema#>
PREFIX GEO600: <http://somehost.org/SCHEMA_PATH_PLACEHOLDER/GEO600-schema-1.0#>

SELECT ?Timestamp ?Name ?CPUtime ?Submitted ?Active ?Pending ?Suspended

WHERE {
  ?statistics GEO600:timestamp  ?Timestamp ;
              GEO600:Resource  ?resource .
  ?resource   GEO600:name      'Total' ;
              GEO600:cputime   ?CPUtime ;
              GEO600:submitted ?Submitted ;
              GEO600:active    ?Active ;
              GEO600:pending   ?Pending ;
              GEO600:suspended ?Suspended .

  FILTER ( ?Timestamp >= xsd:dateTime ("2008-01-01T00:00:00+01:00") ) .
  FILTER ( ?Timestamp <  xsd:dateTime ("2008-02-01T00:00:00+01:00") )

} ORDER BY DESC(?Timestamp)
```

This query specifically asks for all job statistics results across all resources; only those are filtered out and returned which were captured in the month of January 2008; its results are then visualised in Fig. 2.

The GEO600 portlet is currently being tested on an internal GridSphere portal server. Once this testing phase has successfully finished, it will be integrated into the official AstroGrid-D user portal running on `http://cashmere.aip.de:8080/gridsphere/gridsphere`.

2.5 Cactus Integration Tests

For Cactus[15], a numerical simulation framework used by astrophysicists at the AEI and other collaborating numerical relativity groups, we have designed and implemented the AstroGrid-D use case "*Cactus Integration Tests*" – an automated procedure to regularly perform a set of individual unit tests within the rather complex Cactus software environment. Each integration test would (1) check out the latest version of the code from the corresponding software repositories, (2) create a configuration on the given test machine, (3) build that configuration and create a Cactus executable, (4) also build all utility programs included in that configuration, and finally (5) run all available test suites to verify the correctness of the code. Results of each test are collected as Cactus-specific metadata and send off as an RDF/XML document to an external AstroGrid-D metadata information service. For the presentation of the integration test results to the users, a CactusRDF portlet was developed by working group WG-VII and integrated into two portal installations: `https://portal.cactus.code.org` as a public portal for Cactus users, and `https://portal.aei.mpg.de` as a scientists' portal restricted to members of the Numerical Relativity group at AEI and its collaborators.
To avoid duplication, we refer to the AstroGrid-D deliverable documents D2.3[1], D6.3[16], and D6.4[17] for a detailed description of both the *Cactus Integration Tests* procedure providing metadata as well as the *CactusRDF* portlet as a means of querying and presenting metadata. The implementation of this use case has not changed since then.

The *Cactus Integration Tests* use case has been running in production mode for more than a year now; since November 2006, every night a set of three different Cactus application codes (`Einstein, PublicThorns, Whisky`) is tested on a number of up to five different HPC machines (clusters at AEI as well as compute servers at CCT). This has in the meantime produced a considerable amount of metadata for which a statistical summary is given in table 2:

Table 2: Total number of RDF triplets and contexts generated so far (as of 5 February 2008) by *Cactus Integration Tests*.

RDF Repository	# triplets	# contexts
`Einstein`	1296560	1447
`PublicThorns`	963024	1606
`Whisky`	17779	236

Practical experience with the AstroGrid-D metadata information service *Stellaris*[18] revealed several problems in its current implementation of the underlying RDF database backend:

1. The resource requirements of the in-memory database backend used so far became a limiting factor for maximum size of RDF repositories that can be managed by the *Stellaris* RDF server. While this backend has certain advantages in access performance when compared to *file-based* database backends, it clearly doesn't scale with the amount of metadata stored and was therefore deemed unsuitable for large-scale production use in the *Cactus Integration Tests* use case.

2. Switching to a file-based database backend dramatically increased the response time for metadata queries from milliseconds to several dozens of seconds up to minutes. This made interactive searches in *Cactus Integration Tests* results practically unfeasible. The query performance seemed to increase exponentially with the amount of RDF metadata stored.

3. Occasional maintenance work on the machines where the AstroGrid-D metadata information service was running forced the *Stellaris* service to be shut down temporarily. It turned out that, after restart, the previously stored contents of the RDF repositories could not be fully restored anymore or were accessible again only partially. Apparently, the database backend used within *Stellaris* and/or or its connection to the query engine wasn't working properly yet; it certainly needs to be improved in terms of robustness and fail-safety.

While the above-mentioned problems were being addressed and worked on in the current development version of *Stellaris*, we had been looking for an intermediate solution to keep our *Cactus Integration Tests* use case running. We found this alternative in *Sesame*[19], a fully-fledged open source RDF framework which provides all the required functionality and offers an acceptable runtime performance.

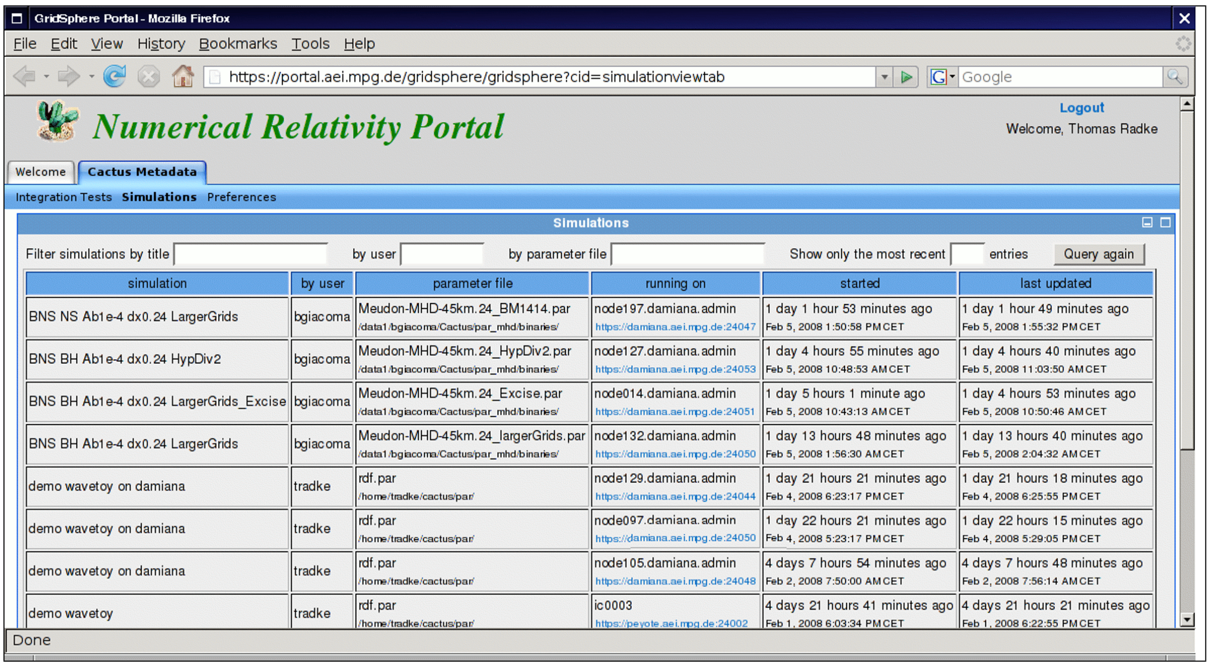
2.6 Cactus Simulation Monitoring & Steering

For AEI's use case *Cactus Simulation Monitoring & Steering*[15], which is primarily being worked on in the AstroGrid-D working groups WG-VI *"Grid Job Monitoring and Steering"* and WG-VII *User Interfaces and APIs*, a new metadata information provider has been designed and is currently being implemented. This provider is integrated into the Cactus code itself, collecting relevant runtime information about ongoing simulations and sending it to an external metadata information service for archiving and later user queries. The most basic information collected about an individual information includes the following metadata:

- the exact start date/time of the simulation as well as the time of the most recent status update operation (`lastUpdated` timestamp)
- the number of processors used by this simulation, and the host where the run was started (processor 0 for a parallel run)
- the user name of the job's owner
- the name, location, and code release of the Cactus executable
- the name and location of the parameter file
- the URL of the simulation's embedded webserver
- the current working directory and the location where Cactus output data will be written
- a full listing of all parameters (names and typed values) set in the parameter file for this simulation

During runtime it is also possible to send update information about the simulation's current status, e.g., the current iteration number, simulation time, or event logs.

From the user's (and her collaborators') point of view it is then probably most important to get an overview of all submitted Cactus simulations, and to be able to directly connect to running simulations (given the URL of the simulation's embedded webserver), monitor their current status and steer runtime parameters if necessary. This functionality is provided by a *CactusRDF* portlet (developed in working group WG-VII) which directs specific SPARQL queries to an external metadata information service to search in the list of archived simulation metadata, and presents the query results to the user, as can be seen in Fig. 3.


GridSphere Portal - Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://portal.aei.mpg.de/gridsphere/gridsphere?cid=simulationviewtab
Google
Numerical Relativity Portal
Logout
Welcome, Thomas Radke
Welcome
Cactus Metadata
Integration Tests
Simulations
Preferences
Simulations
Filter simulations by title
by user
by parameter file
Show only the most recent
entries
Query again
Done

simulation	by user	parameter file	running on	started	last updated
BNS NS Ab1e-4 dx0.24 LargerGrids	bgiacoma	Meudon-MHD-45km.24_BM1414.par /data1/bgiacoma/Cactus/par_mhd/binaries/	node197.damiana.admin https://damiana.aei.mpg.de:24047	1 day 1 hour 53 minutes ago Feb 5, 2008 1:50:58 PM CET	1 day 1 hour 49 minutes ago Feb 5, 2008 1:55:32 PM CET
BNS BH Ab1e-4 dx0.24 HypDiv2	bgiacoma	Meudon-MHD-45km.24_HypDiv2.par /data1/bgiacoma/Cactus/par_mhd/binaries/	node127.damiana.admin https://damiana.aei.mpg.de:24053	1 day 4 hours 55 minutes ago Feb 5, 2008 10:48:53 AM CET	1 day 4 hours 40 minutes ago Feb 5, 2008 11:03:50 AM CET
BNS BH Ab1e-4 dx0.24 LargerGrids_Excise	bgiacoma	Meudon-MHD-45km.24_Excise.par /data1/bgiacoma/Cactus/par_mhd/binaries/	node014.damiana.admin https://damiana.aei.mpg.de:24051	1 day 5 hours 1 minute ago Feb 5, 2008 10:43:13 AM CET	1 day 4 hours 53 minutes ago Feb 5, 2008 10:50:46 AM CET
BNS BH Ab1e-4 dx0.24 LargerGrids	bgiacoma	Meudon-MHD-45km.24_largerGrids.par /data1/bgiacoma/Cactus/par_mhd/binaries/	node132.damiana.admin https://damiana.aei.mpg.de:24050	1 day 13 hours 48 minutes ago Feb 5, 2008 1:56:30 AM CET	1 day 13 hours 40 minutes ago Feb 5, 2008 2:04:32 AM CET
demo wavetoy on damiana	tradke	rdf.par /home/tradke/cactus/par/	node129.damiana.admin https://damiana.aei.mpg.de:24044	1 day 21 hours 21 minutes ago Feb 4, 2008 6:23:17 PM CET	1 day 21 hours 18 minutes ago Feb 4, 2008 6:25:55 PM CET
demo wavetoy on damiana	tradke	rdf.par /home/tradke/cactus/par/	node097.damiana.admin https://damiana.aei.mpg.de:24050	1 day 22 hours 21 minutes ago Feb 4, 2008 5:23:17 PM CET	1 day 22 hours 15 minutes ago Feb 4, 2008 5:29:05 PM CET
demo wavetoy on damiana	tradke	rdf.par /home/tradke/cactus/par/	node105.damiana.admin https://damiana.aei.mpg.de:24048	4 days 7 hours 54 minutes ago Feb 2, 2008 7:50:00 AM CET	4 days 7 hours 48 minutes ago Feb 2, 2008 7:56:14 AM CET
demo wavetoy	tradke	rdf.par /home/tradke/cactus/par/	ic0003 https://peyote.aei.mpg.de:24002	4 days 21 hours 41 minutes ago Feb 1, 2008 6:03:34 PM CET	4 days 21 hours 21 minutes ago Feb 1, 2008 6:22:55 PM CET

Figure 3: Snapshot of the *Simulations* page in the Numerical Relativity portal, presenting the results from a metadata query about known Cactus simulations

Metadata queries are generated dynamically, according to what information the user asks for. For instance, it is also possible to retrieve the contents of the parameter file that a given simulation was started with, display it in full text, or compare it with other parameter files of similar simulations. All these pages are, together with the *Cactus Integration Tests* webpages, integrated in the *Numerical Relativity portal* (`https://portal.aei.mpg.de`) as a single point of service contact for Cactus users.

While for a detailed description of the technical implementation of the Cactus metadata information provider and the *CactusRDF* portlet we refer to AstroGrid-D's working group WG-VI deliverable document D6.4[17], it should be emphasised here that the *Cactus* application use case brought up a new requirement for the functionality of a metadata information service: a transaction mechanism for atomic modify/add operations on an RDF repository. In order to provide current runtime information about an ongoing Cactus simulation, two different RDF update operations need to be performed at the same time: (1) the modification of existing metadata (e.g., the `lastUpdated` timestamp) as well as (2) the addition of new metadata (e.g., an event log graph listing all the runtime parameter steering requests by users which have been processed since the last update).
However, the standard server-side upload commands of an HTTP-accessible information service usually provide only an `HTTP POST` operation to create a new RDF graph in the repository (overwriting any previously stored metadata) and an `HTTP PUT` operation to add new metadata to an

already existing RDF graph. A special transaction operation becomes necessary in order to perform combined modify/add metadata updates in a reliable and efficient way: with a single client-side operation, RDF metadata can be created/modifed/added/removed atomically; if one of these update operations within a transaction sequence fails, all previous operations will be rolled back such that the repository contents remain unchanged.

The transaction functionality is available in both the latest version of the *Stellaris* metadata information service and the *Sesame* RDF framework. It is being successfully used by the metadata information providers implemented in *Cactus*.

2.7 Robotic Telescopes

2.7.1 Temporal Information and the *Telescope Timeline*

The metadata management of robotic telescopes is also accomplished via the database trigger mechanism for job monitoring as described in Sec. 2.3 and listing 1. The metadata includes weather and scheduling information. The format for the weather data is stored in an RDF template, which was obtained through an XSLT from an RTML [20] for weather information. But it can also be transfered as RTML and translated into RDF by Stellaris using the GRDDL [11] specification. The scheduling information is provided in the Usage Record format (Sec. 2.3). Since it is the standard for job monitoring in AstroGrid-D, the same techniques and interfaces can be used. An example is the *Telescope Timeline* - an extension of the Timeline for telescopes (Fig. 4). Details about dynamic metadata management using database triggers is given in [23].

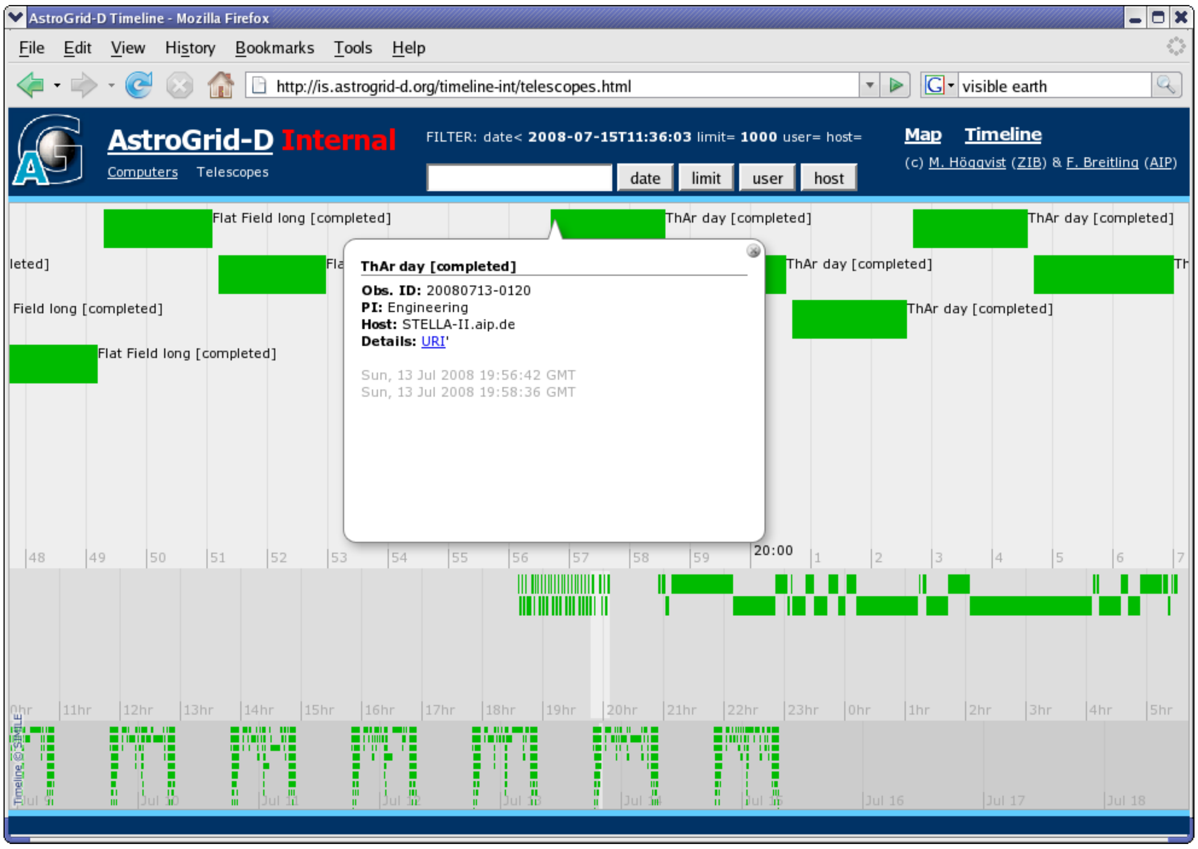
Alternatively the iCalendar [21] format can be used. iCalendar has a corresponding RDF representation called RDF Calendar [22]. The iCalendar standards has the advantage that the information can be displayed by standard calendar tools such as Google Calendar and RDF browsers. On the other hand it is not compatible to the Usage Record format adopted within AstroGrid-D.

2.7.2 Geographic Information and the *Telescope Map*

For geographic information related to the location of telescopes the *Telescope Map* is used. It is an enhanced version of the GridMap (Sec. 2.8). In addition to telescope locations and instrumentation it also displays daylight regions and weather information. The daylight regions are calculated by a javascript found at [24]. The weather information is obtained by another javascript from the same site. It provides a mosaic of satellite cloud images from one of the following sources in this order:

- Google Earth's Weather layer, sourced from the US Naval Research Lab (updated hourly)
- NASA's Earth Science Office (global imagery updated every 6 hours)
- xplanet's cloud layer (updated every 3 hours)

An example is shown in Fig. 5. As the GridMap the *Telescope Map* is interactive and displays additional information for each telescope on selection.


AstroGrid-D Timeline - Mozilla Firefox

http://is.astrogrid-d.org/timeline-int/telescopes.html

FILTER: date< **2008-07-15T11:36:03** limit= **1000** user= host=

ThAr day [completed]
Obs. ID: 20080713-0120
PI: Engineering
Host: STELLA-II.aip.de
Details: URI'

Sun, 13 Jul 2008 19:56:42 GMT
Sun, 13 Jul 2008 19:58:36 GMT

Figure 4: A browser screenshot of the Telescope Timeline showing recent observations of the STELLA-I telescope.

2.8 GridMap

Introduction GridMap is a demo application for Stellaris which initially displayed the location and status of compute resources in AstroGrid-D. The demo version has since then been improved to include jobs and robotic telescopes as well as an alternative timeline view of current job submissions and telescope requests.

Sources of Metadata The data used for the application is taken from three different sources. First, the computer resource metadata is collected from the Globus MDS service, which is deployed as a central service in AstroGrid-D.[2] This MDS service aggregates data from all the compute resources. As an alternative, the data can also be retrieved from the central D-Grid MDS service[3] which has information on all compute resources from the different community grids. MDS-data is retrieved periodically and transformed to RDF using a Python script. After the transformation it is uploaded to Stellaris. Second, in order to place the resources correctly on the map, a manually edited file with GPS coordinates for each resource is used. Third, the job monitoring data is uploaded directly to Stellaris from selected jobs via a script wrapping the job submission.

Currently, the produced metadata is stored in three different graphs, one for the mds-data, one for

[2]http://mintaka.aip.de:8080/webmds/webmds?info=indexinfo
[3]http://webmds.lrz-muenchen.de:8080/webmds/webmds?info=indexinfo

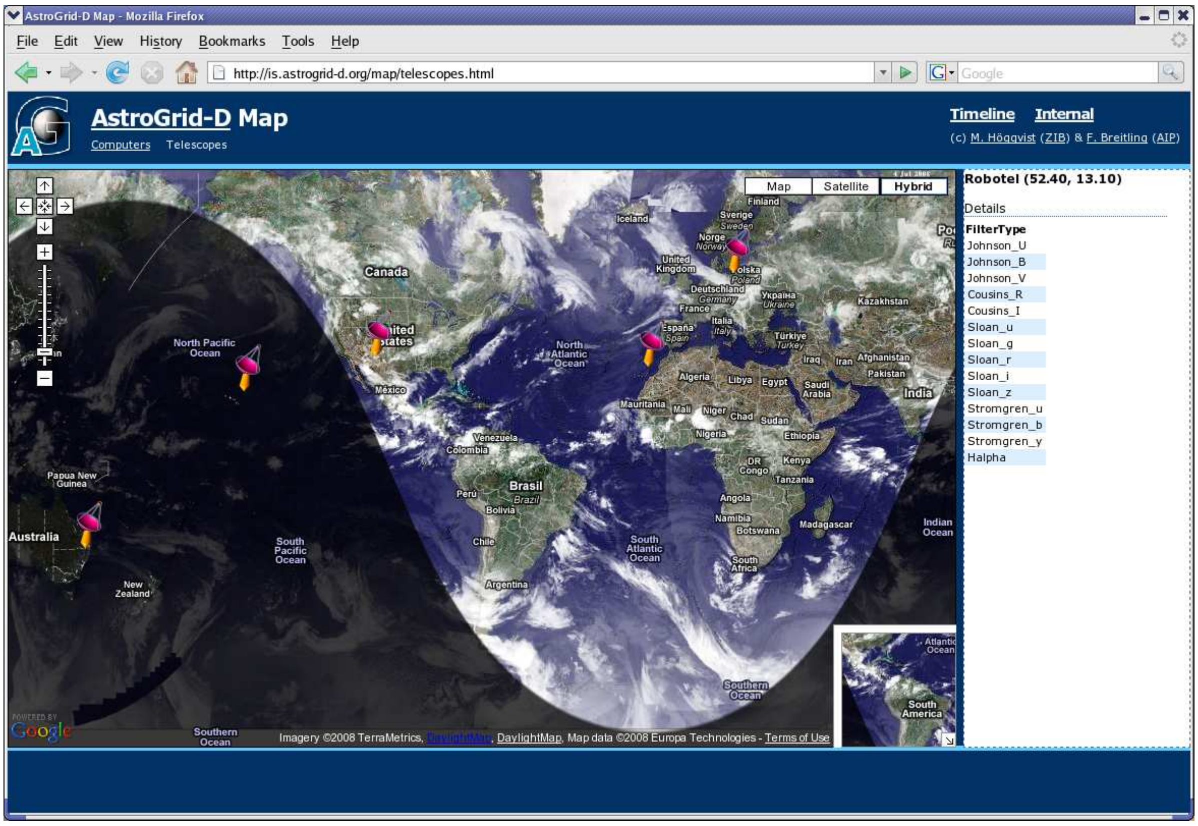

AstroGrid-D Map - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://is.astrogrid-d.org/map/telescopes.html
AstroGrid-D Map
Computers Telescopes
Timeline Internal
(c) M. Högqvist (ZIB) & F. Breitling (AIP)
Map Satellite Hybrid
Robotel (52.40, 13.10)
Details
FilterType
Johnson_U
Johnson_B
Johnson_V
Cousins_R
Cousins_I
Sloan_u
Sloan_g
Sloan_r
Sloan_i
Sloan_z
Stromgren_u
Stromgren_b
Stromgren_y
Halpha
Imagery ©2008 TerraMetrics, DaylightMap, DaylightMap, Map data ©2008 Europa Technologies - Terms of Use

Figure 5: A browser screenshot of the TelescopeMap showing daylight and weather information.

jobs and one for resource locations. This can be improved, since storing all jobs in a single graph is especially inefficient if a common pattern is to access a single job.

Metadata vocabularies

Three different vocabularies are used for describing jobs, resource location and compute resources. The three tables below describe the attributes from each vocabulary. Since these vocabularies were developed exclusively to support the user interface, they may not be re-usable in other contexts. However, the information could be used in other applications, for example, to list all active AstroGrid-D resources.

Entries Typical entries for each of the different vocabularies are shown below. An important observation in these entries is how the unique naming of resources allow us to reference the same resource in different RDF-graphs.

Listing 3: Job description example

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:gacgjobs="http://www.gac-grid.org/schema/jobs#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">

  <rdf:Description rdf:about="http://www.gac-grid.org/jobs/749b8930-aa53-11dc
     -948b-8c626b724edb">
```

Table 3: Job vocabulary, namespace: http://www.gac-grid.org/schema/jobs#

Attribute	**Description**
currentState	Current state of the job
jobname	Name of a job
site	The compute resource assigned to the job
stageintime	Timestamp indicating when data stage in started
stageouttime	Timestamp indicating when data stage out started
starttime	Job start time
activetime	Timestamp when job became active in the job queue at the assigned compute resource
user	The name of the user
timeStamp	A timestamp in RFC 2822 format, used for indicating state changes
state	Current job state as reported by Globus
Job	Used to indicate that an RDF subject is of type Site

Table 4: Location vocabulary, namespace: http://www.w3.org/2003/01/geo/wgs84_pos#

Attribute	**Description**
long	Longitude in decimal notation
lat	Latitude in decimal notation

```
    <gacgjobs:currentState rdf:resource="http://www.gac-grid.org/749b8930-aa53
       -11dc-948b-8c626b724edb/state"/>
    <gacgjobs:jobname>dynamo-photon.aip.de</gacgjobs:jobname>
    <gacgjobs:site rdf:resource="http://photon.aip.de"/>
    <gacgjobs:stageintime>Fri, 14 Dec 2007 15:47:16 +0100</gacgjobs:stageintime
       >
    <gacgjobs:stageouttime>Fri, 14 Dec 2007 15:52:52 +0100</
       gacgjobs:stageouttime>
    <gacgjobs:activetime>Fri, 14 Dec 2007 15:47:37 +0100</gacgjobs:activetime>
    <gacgjobs:user>O=GermanGrid/OU=AIP/CN=Steve White</gacgjobs:user>
    <gacgjobs:starttime>Fri, 14 Dec 2007 15:47:16 +0100</gacgjobs:starttime>
    <rdf:type rdf:resource="http://www.gac-grid.org/schema/jobs#Job"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://www.gac-grid.org/749b8930-aa53-11dc-948b-8
     c626b724edb/state">
    <gacgjobs:timeStamp>Fri, 14 Dec 2007 15:53:55 +0100</gacgjobs:timeStamp>
    <gacgjobs:state>ERROR</gacgjobs:state>
  </rdf:Description>
</rdf:RDF>
```

Listing 4: Location example

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:geo="http://www.w3.org/2003/01/geo/wgs84\_pos#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <rdf:Description rdf:about="http://hydra.ari.uni-heidelberg.de">
    <geo:long>8.687</geo:long>
    <geo:lat>49.418</geo:lat>
  </rdf:Description>
</rdf:RDF>
```

Listing 5: Compute resource description example

Table 5: Compute resource vocabulary, namespace: http://www.gac-grid.org/schema/gridmap#

Attribute	Description
RunningJobs	Currently active and running jobs in the queue
FreeCPUs	Number of unused CPUs at the resource
TotalCPUs	Total number of CPUs provided by the resource
TotalJobs	Number of waiting and running jobs
WaitingJobs	Total number of waiting jobs for a given queue
ClusterName	The name of the cluster
CE	Compute Element, i.e., a job queue at a resource
Site	Used to indicate that an RDF subject is of type Site

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF xmlns:gridmap="http://www.gac-grid.org/schema/gridmap#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <rdf:Description rdf:about="http://hydra.ari.uni-heidelberg.de">
    <gridmap:ClusterName>ARI-ZAH GridGateWay</gridmap:ClusterName>
    <gridmap:CE rdf:resource="http://hydra.ari.uni-heidelberg.de/PBS/default"/>
    <rdf:type rdf:resource="http://www.gac-grid.org/schema/gridmap#Site"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://hydra.ari.uni-heidelberg.de/PBS/default">
    <gridmap:RunningJobs>0</gridmap:RunningJobs>
    <gridmap:FreeCPUs>1</gridmap:FreeCPUs>
    <gridmap:TotalCPUs>10</gridmap:TotalCPUs>
    <gridmap:TotalJobs>0</gridmap:TotalJobs>
    <gridmap:WaitingJobs>0</gridmap:WaitingJobs>
  </rdf:Description>
</rdf:RDF>
```

Querying the data

Access to the above described data is mostly focused on the resources. Two example queries are listed below, they express "What is the location of the current compute resources?" (listing 6) and "What jobs are currently active at resource X?" (listing 7).

Listing 6: A query returning all compute resources and their locations.

```
PREFIX glue:<http://www.gac-grid.org/schema/gridmap#>
PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX geo:<http://www.w3.org/2003/01/geo/wgs84_pos#>

SELECT ?site ?longitude ?latitude
FROM NAMED <http://stellaris.gac-grid.org/context/gridmap/mds#context>
FROM NAMED <http://stellaris.gac-grid.org/context/gridmap/location#context>
WHERE {
  ?site rdf:type glue:Site .
  OPTIONAL {
    ?site geo:long ?longitude .
    ?site geo:lat ?latitude .
  }
}
```

In listing 6, the `FROM NAMED` defines that the query should be executed over two graphs; one which has the aggregated MDS-data and one with the manually entered locations of the compute resources. The data is then reduced by only selecting RDF-subjects which are of type Site resource vocabulary (table 5 and which optionally has a latitude and longitude. Since the keyword `OPTIONAL` is used, sites which have not yet been assigned a location are also part of the result set.

Listing 7: A query returning the name, state and a unique job identifier of all active jobs at the compute resource hydra.ari.uni-heidelberg.de.

```
1  PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
2  PREFIX jobs: <http://www.gac-grid.org/schema/jobs#>
3
4  SELECT ?name ?state ?jobid
5
6  WHERE {
7    ?jobid jobs:site <http://hydra.ari.uni-heidelberg.de> .
8    ?jobid jobs:currentState ?cs .
9    ?cs jobs:state ?state .
10   FILTER(?state = 'Active' || ?state = 'ACTIVE') .
11   ?jobid jobs:jobname ?name .
12 }
```

The query in listing 7 is executing over the full dataset since at the time of designing the application, it was not yet decided which graphs would contain job data. However, reducing the queried data-sets by using `FROM NAMED` as in listing 6 is preferable from a performance perspective since a subset of the data is processed. The query itself is first selecting all jobs at the site `http://hydra.ari.uni-heidelberg.de`, and then continues to select the `currentState`-predicate. By comparing with the job instance listing, we can see that the RDF-object matching with `currentState` is also a URI (`http://www.gac-grid.org/749b8930-aa53-11dc-948b-8c626b724edb/state`) or another node from a graph perspective. That is, the patterns on line 8-9 are following a path starting with the job and ending with a given state of the job. Since we only want active jobs with this query, the selected jobs up until this point are filtered to only have the value of "Active" or "ACTIVE". Note that the query ends by selecting the `jobname`-attribute. This pattern could also have been put earlier in the query, but the result set for the ?name-variable would then have been larger since the filtering on the state had not yet been done. This hand-tuned optimization is necessary to improve query performance since current SPARQL-engines often lack good support for query optimization,

Conclusions The GridMap-demo application shows several basic development aspects when developing applications for Stellaris. This starts with modeling and converting data to RDF, upload and usage of the Stellaris-service and information extraction through queries. An example of the final product is shown in figure 6.

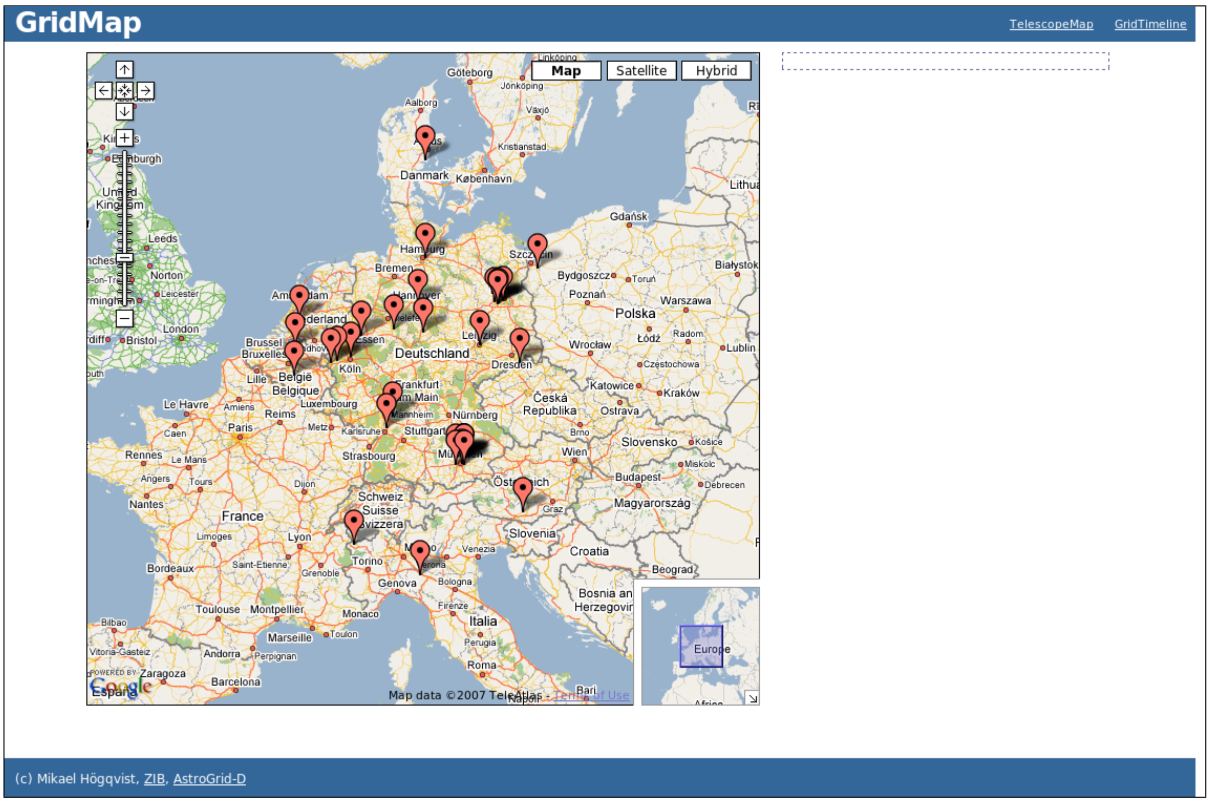

GridMap
TelescopeMap
GridTimeline
Map
Satellite
Hybrid
Map data ©2007 TeleAtlas - Terms of Use
(c) Mikael Högqvist, ZIB, AstroGrid-D

Figure 6: A browser screenshot of the GridMap application.

3 Conclusions

The use cases of AstroGrid-D provide good coverage of information provider features required in practice by the astronomy community. In most cases, Stellaris has proven to be a practical framework for managing metadata, but the use cases have also uncovered limitations, particularly scaling to large volumes of data, that require further work.

F: References / Bibliography

References

[1] Metadata Management – Metadata Information Providers. Work Group II deliverable document D2.3, AstroGrid-D project;
http://www.gac-grid.org/project-documents/deliverables/wp2/Metadata_Providers_D23.pdf

[2] Metadata Management – Second Report on Metadata Information Providers. Work Group II deliverable document D2.6, AstroGrid-D project;
http://www.gac-grid.org/project-documents/deliverables/wp2/Metadata_Providers_D26.pdf

[3] AstroGrid-D Information Service Requirements Specification and Architectural Design. Work Group II deliverable document D2.1, AstroGrid-D project;
http://www.gac-grid.org/project-documents/deliverables/wp2/Metadata_Providers_D21.pdf

[4] Stellaris: The AstroGrid-D Information Service. Work Group II deliverable document D2.2, AstroGrid-D project;
http://www.gac-grid.org/project-documents/deliverables/wp2/Metadata_Providers_D22.pdf

[5] The Globus Alliance, *GT Information Services: Monitoring and Discovery System (MDS)*, http://www.globus.org/toolkit/mds/

[6] Audit Logging; http://www.globus.org/toolkit/docs/4.0/execution/wsgram/WS_GRAM_Audit_Logging.html

[7] SGAS; http://www.sgas.se/

[8] Globus Installation in the Astrogrid-D Standard (GACSI); http://www.gac-grid.net/project-products/grid-support/grid-installation.html#gram_audit

[9] PostgreSQL; PostgreSQL http://www.postgresql.org/

[10] R. Mach, R. Lepro-Metz, S. Jackson, L. McGinnis [Editor]: Usage Record Format Recommendation; http://www.psc.edu/~lfm/PSC/Grid/UR-WG/UR-Spec-gfd.58.pdf

[11] Gleaning Resource Descriptions from Dialects of Languages; http://www.w3.org/TR/grddl/

[12] GEO600 Use Case Description, AstroGrid-D requirements document. http://www.gac-grid.org/project-documents/UseCases/geo600.pdf

[13] GEO600 Project Homepage http://geo600.aei.mpg.de/

[14] D-MON – Horizontal Integration of Ressource and Service Monitoring in D-Grid. Project Homepage http://www.d-grid.de/index.php?id=401

[15] Cactus Use Case Description, AstroGrid-D requirements document. http://www.gac-grid.org/project-documents/UseCases/Cactus.pdf

[16] Thomas Radke: Architecture of generic grid-enabled Monitoring & Steering Method s in AstroGrid-D Applications. Architecture Specification, Work Group VI deliverable document D6.3, AstroGrid project;
http://www.gac-grid.org/project-documents/deliverables/wp6/WG6_D6_3.pdf

[17] Thomas Radke: Prototype Implementation of grid-enabled Monitoring Methods. Documentation and Test Report, Work Group VI deliverable document D6.4, AstroGrid project;
http://www.gac-grid.org/project-documents/deliverables/wp6/WG6_D6_4.pdf

[18] *Stellaris* – A Metadata Information Service for AstroGrid-D. Project homepage http://stellaris.zib.de

[19] OpenRDF.org Homepage. http://openrdf.org/

[20] Hessman, F.V., et al., Remote Telescope Markup Language (RTML), Astronomische Nachrichten, 2006

[21] Internet Calendaring and Scheduling Core Object Specification (iCalendar), http://www.w3.org/2002/12/cal/rfc2445

[22] RDF Calendar Workspace, http://www.w3.org/2002/12/cal/

[23] Providing Dynamic Metadata of Robotic Telescopes to Stellaris. Technical Report D2.7, AstroGrid-D project, In preparation.

[24] DaylightMap.com/ Homepage. http://www.daylightmap.com/