Grid Information System Documentation

Release 1.0.0

Maria Alandes Pradillo, Laurence Field, Baptiste Grenier

Dec 02, 2018

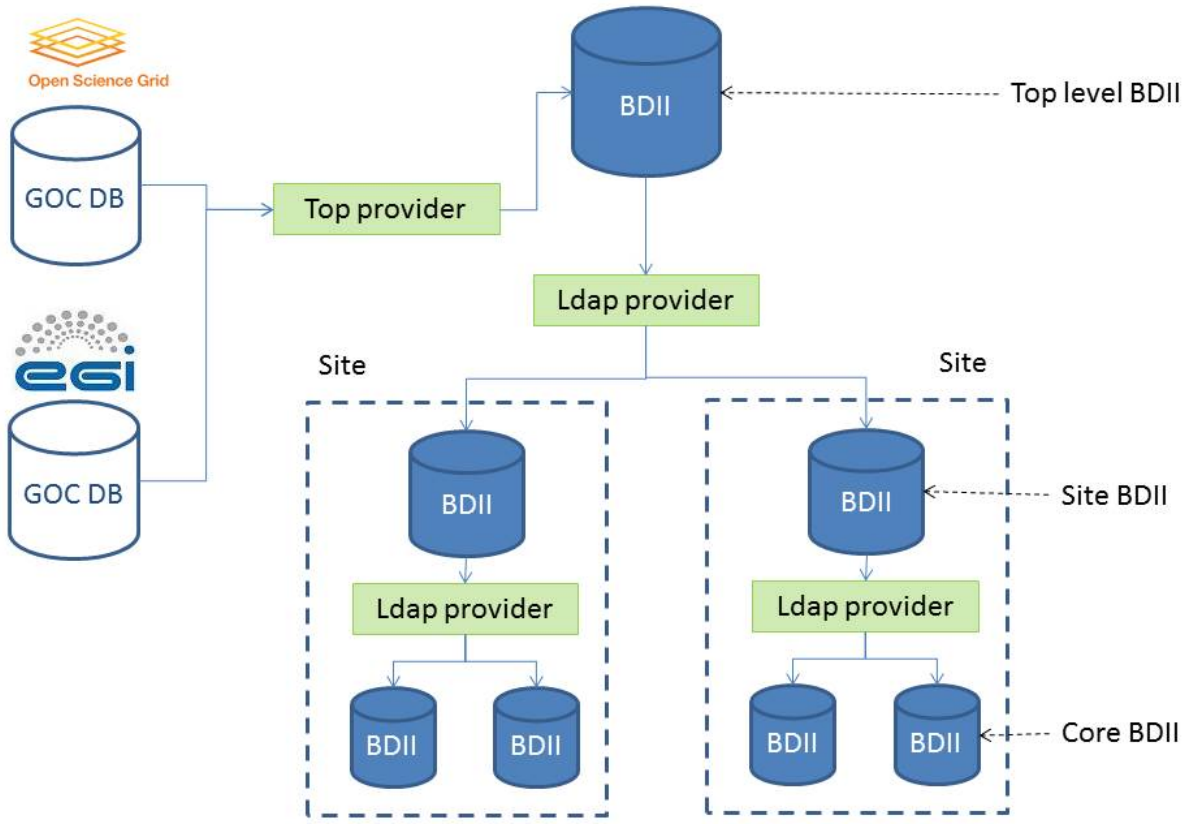
Contents:

In this page you will find documentation about the Grid Information System organised in the following way:

CHAPTER 1

Grid Information System Introduction

The grid information system is a mission-critical component in the WLCG grid infrastructure. It provides detailed information about grid services which is needed for various different tasks. The grid information system has a hierarchical structure of three levels. The fundamental building block used in this hierarchy is the Berkley Database Information Index (BDII). Although the BDII has additional complexity, it can be visualized as an LDAP database. The resource level or core BDII is usually co-located with the grid service and provides information about that service. Each grid site runs a site level BDII. This aggregates the information from all the resource level BDIIs running at that site. The top level BDII aggregates all the information from all the site level BDIIs and hence contains information about all grid services. There are multiple instances of the top level BDII in order to provide a fault tolerant, load balanced service. The information system clients query a top level BDII to find the information that they require.



The BDIIs are populated with information by running information providers. These are scripts which obtain information, format it as LDIF and print the result to standard out. These information providers can also be used to query other BDIIs which is how the hierarchy is built. The order in which these information providers are run is random.

The information in the information systems conforms to a schema called the GLUE schema. The GLUE schema started as collaboration effort between European and US grid projects to facilitate interoperation between them. The Open Grid Forum (OGF) is now responsible for the GLUE schema.

The information system is bootstrapped from the information registered in the Operations Databases of EGI and OSG grid infrastructures (GOCDB and OIM). When a site registers, it enters the URL for the site level BDII into the GOCDB/OIM. GOCDB/OIM then generates a list of LDAP URLs for all the sites in the grid and this is downloaded by the information provider running on the top level BDII. These URLs are then used to query all the site level BDII and the result is used to populate the top level BDII.

CHAPTER 2

BDII Releases

Update: Update 11 - 28.11.2018

Name & Version	Platform	Release notes	Installation & configuration
glite-info-update-endpoints 3.0.1	SL6, CentOS 7	• #1: Fix silent fail on CentOS 7 when unable to validate GOCDB certificate • #11: Totally drop OSG support • #17: Handle sane defaults in the configuration file • PR#14: Linting and build in containers	Check configuration changes.
bdii 5.2.25	SL6, CentOS 7	• #3: Fix restart failing on stale PID • PR#9: import BDII product card JSON • PR#11: Linting and build in containers	Restart to use new init script.

Update: Update 10 - 06.10.2014

Name & Version	Platform	Release notes	Installation & configuration
glue-validator 2.0.25	SL5, SL6	#GRIDINFO-58: new version of glue-validator which applies an internal workaround for not testing the Admin Foreign Key attributes in the case of StoRM services as there is a known bug in the information provider that prevents from publishing correct information.	Note that glue-validator is integrated as a Nagios Operational probe for EGI for the automatic validation of GLUE 2 data.

Update: Update 9 - 07.08.2014

Name & Version	Platform	Release notes	Installation & configuration
glue-schema 2.0.11-1	EMI 3 SL5, EMI 3 SL6	Update of the GLUE 2 schema LDAP definition with the following changes: GRIDINFO-53: All the entities but Entity moved to type STRUCTURAL GRIDINFO-9: GLUE 2 booleans should be DirectoryString not LDAP Boolean Note that the change regarding the boolean attributes affects in particular the following attributes: GLUE2ExecutionEnvironmentVirtualMachine, GLUE2ExecutionEnvironmentConnectivityIn, GLUE2ExecutionEnvironmentConnectivityOut, GLUE2AdminDomainDistributed, GLUE2ComputingManagerBulkSubmission, GLUE2ComputingManagerHomogeneous, GLUE2ComputingManagerWorkingAreaGuaranteed, GLUE2ComputingManagerWorkingAreaShared. The change refers to: The BooleanMatch has been removed from the schema. All booleans are strings now. Booleans are now DirectoryString type as well. This means that if the attribute has no value, the attribute will not be published. It also means that the boolean is expected to be as defined in GFD.147 section A.3, that is, might contain the value "undefined" when not "true" nor "false".	A BDII restart is needed to be able to use the new GLUE 2 schema revision.
glite-info-provider-service 1.13.4-1	EMI 3 SL5, EMI 3 SL6	New version of the info provider fixing a bug and publishing a new attribute: GGUS:107264 (link is external): Patch for the RTEPublisher start time GRIDINFO-54: New attribute to print GLUE schema version	The CREAM CE, site BDIIs and top BDIIs are the only services affected by this new release. No restart or reconfiguration of the BDII is needed in any case.
glue-validator 2.0.24-1	EMI 3 SL5, EMI 3 SL6	Minor release containing an update of the types.py library and the deprecation of one of the tests as requested by EGI: GRIDINFO-52: xroot should be used when referring to the xrootd protocol GRIDINFO-51: Add new ServiceType values GRIDINFO-50: Obsolete WLCG_NAME test GRIDINFO-47: Add 'notification' in the capability type GRIDINFO-46: Bugs reported by DPM (II) GRIDINFO-6: Test mandatory objects are present GRIDINFO-3: Test GLUE2ServiceAdminDomainForeignKey = GLUE2DomainID in the DN	Note that glue-validator is integrated as a Nagios Operational probe for EGI for the automatic validation of GLUE 2 data. Note that this particular version of glue-validator has not been deployed in Nagios due to a bug in StoRM that raises an ERROR for sites (GGUS:108556). The fix is required in the info provider. a new glue-validator version will be provided to workaround this issue.
bdii 5.2.23-1	EMI 3 SL5, EMI 3 SL6	GRIDINFO-55: Number of parallel threads set to 64 by default to be able to cope with high load scenarios as the one reported in GGUS:107621.	Only top BDIIs are affected by this fix. A restart of the BDII is needed after upgrading the package.

CHAPTER 3

Known Issues

3.1 Resource BDII

Important note: site and top BDIIs are also affected by any known issues in the resource BDII.

- BUG #GRIDINFO-57: Affects all versions of the bdii. When a DN changes its case from lower to upper case or viceversa, these changes are not propagated to the BDII and the new DN doesn't get published. This only affects GLUE 2 because it's case sensitive. The workaround is to add an extra difference like a "_" or "-" or anthing else, so that the BDII is able to see the DN is a new one and like this it will get published.
- Affects when upgrading to bdii >= 5.2.20-1 using EPEL 5 (SL5 installations) repository. In this scenario, the LDAP daemon fails to start. YAIM detects that Openldap 2.4 is installed and tries to use it. On the other hand, the slapd.conf file has been defined to use Openldap 2.3. This is because Openldap 2.4 is not distributed in EPEL 5 and the necessary dependencies have been resolved with Openldap 2.3 packages. Please, uninstall openldap2.4-* packages or add the following lines in /etc/bdii/bdii-slapd.conf:

```
modulepath /usr/lib64/openldap2.4
moduleload back_relay
```

- Affects when upgrading to bdii >= 5.2.17-1 using EMI or UMD SL5 repositories: Openldap 2.4 is the mandatory version needed since the EMI 3 release. This is due to some changes in the configuration of the BDII needed to be able to work with ARC resources. If YAIM is not used to configure the BDII, the /etc/sysconfig/bdii file needs to be manually changed and define:

```
SLAPD=/usr/sbin/slapd2.4
```

- Empty attributes are no longer allowed in the GLUE 2.0 schema. The corresponding object will not be published if at least one attribute is empty.
- BUG #101709: Affects bdii < 5.2.21-1. At boot time or when using 'service bdii restart', the environment is reset and some information providers (i.e. the ones for SGE batch system) fail due to missing environment varibles, publishing default values in the BDII. * *Workaround*: use '/etc/init.d/bdii restart' since this preserves the environment.

- BUG #101237: Affects bdii < 5.2.20-1. Obsolete GLUE 2 entries are not removed due to a bug in the delete functionality of the BDII.
  - *Workaround*: clean the cache used by the glite-info-provider-ldap script and restart all the BDIIs in the site:
    - Restart resource BDIIs
    - Remove contents of /var/lib/bdii/gip/cache/gip/site-urls.conf-glue2/ and restart site BDII
    - Remove contents of /var/lib/bdii/gip/cache/gip/top-urls.conf-glue2/ and restart the top BDII

3.2 Top BDII

- BUGZILLA: Slapd process on Top BDII crashes after upgrade to CentOS/SLC6.7 with openldap-servers-2.4.40-5.el6.
- Affects glite-info-plugin-delayed-delete-status 1.0.0-1: If a resource BDII or OSG resource is down for a while, its status attributes (like GlueCEStateStatus, GlueServiceStatus, GlueServiceStatus or GlueSEStatus, and the same for GLUE 2) published across top BDIIs may not match. All top BDIIs should run at least glite-info-plugin-delayed-delete-status 1.0.1-1 to get rid of this issue. Note that Top BDIIs runinng version 1.0.1-1 of the plugin would publish status 'Unknown', while previous ones would publish the last published status before the resource disappeared (although it is also possible that they publish status 'Unknown' if this value propagated from their site BDII before the resource dissapears from the site BDII cache).
  - *Workaround*: The root cause should be fixed at infrastructure level. In the meantime, the sys admin of the affected resource could at least make sure the resource is back to be published in the corresponding site BDII.
- BUG #102608: Affects 1.4.7-1 > glite-info-provider-ldap >= 1.4.5-1. Sites dissapear from the top level BDII after the BDII_DELETE_DELAY is over.
  - *Workaround*: Increase the cache validity in /usr/libexec/glite-info-provider-ldap, line 122 (there is no need to restart the BDII):

```
$ttl=300;
```

- BUG #102384: Affects bdii >= 5.2.17-1 and ldap-info-provider-ldap < 1.4.6-1 GLUE 2 Contact and Location objects are not published in the top level BDII. Although the effect of this bug is visible in top level BDIIs, this bug requires a fix in the site BDIIs. No workaround is suggested for this bug since it has to be applied in all site BDIIs which are not under the control of the top BDII sys admin.
- BUG #GRIDINFO-4: glite-info-plugin-delayed-delete-status does not set to 'Unknown' Coputing and Storage Share state attributes
- The glite-info-update-endpoints script is executed as an hourly cron job. It creates a cache file of the list of GOCDB and OIM site LDAP URLs. When the BDII is restarted, the cache file is used. Although the list of sites does not change very frequently in GOCDB and OIM, for an up to date list, re-run glite-info-update-endpoints before the BDII restart:

```
/usr/bin/glite-info-update-endpoints -c /etc/glite/glite-info-update-endpoints.conf
```

- BUG #99298: Affects bdii < 5.2.21-1. Due to the caching mode, decommissioned services are published as 'Production' until the cache expires. They should instead be published as 'Unknown'.
- BUG #99322: Affects glite-info-update-endpoints < 2.0.13-1. The glite-info-update-endpoints script does not fail when using the manual option and the input file does not exist.

- BUG #101090: Affects bdii < 5.2.20-1. The GLUE 2.0 database backend in the LDAP server is not benefitting from a set of performance improvements due to a missing symlink to the proper configuration file.
  - *Workaround*: edit /etc/init.d/bdii and add in line 150:

```
$RUNUSER -s /bin/sh ${BDII_USER} -c "ln -sf ${DB_CONFIG}_top ${SLAPD_DB_DIR}/glue/DB_
↪CONFIG"
```

3.3 Site BDII

- Upgrade to openldap-2.4.40-12 on SL6: if clients experience slow queries, a reboot of the node solves the issue.
- BUG #GRIDINFO-5: Sometimes GLUE state attributes are not cleanly replaced with the string 'Unknown'. i.e. "GLUE2EndpointHealthStateInfo: Unknownp and running"

3.4 Information Providers

3.4.1 PBS/Torque

- CREAM-101: Fixed in EMI 2 Update 16 and EMI 3 Update 6. Wallclock time is published in hours when GLUE 1 attributes should be expressed in minutes and GLUE 2 attributes should be expressed in seconds. See the CREAM Known Issues page for the details and the workaround.

CHAPTER 4

Information System for Sys Admins

4.1 Releases

- *BDII Releases*: Summary of the BDII releases
- *Known Issues*: Summary of the known issues affecting all BDII flavours
- Former EMI BDII releases: details on the BDII updates released during the EMI project
- EPEL releases: status of the BDII releases in EPEL 6

4.2 Documentation

- *Quickstart guide*
- `Sys admin guide`: detailed documentation for BDII sys admins including:
  - Functional description
  - System Administrator Guide
    - HW requirements
    - Installation and Configuration
    - Known Issues
    - Operation and Monitoring
    - Troubleshooting
    - User Support
  - Servide Reference Card
  - Glue validator Guide

4.3 EGI documentation

The following links contain useful documentation on BDII configuration and set up aimed at EGI sys admins:

- How to publish Site information: How to configure a site BDII to publish site resources.
- BDII High Availability: How to configure a top level or site BDII with a high availability setup.
- How to change the BDII host: Steps on what to do when the BDII is moved to a new host.
- Top BDII NGI list: List of top level BDIIs per NGIs.
- 444444 Waiting Jobs: Tips to have waiting jobs correctly published in the BDII

CHAPTER 5

Quickstart guide

5.1 Configuration

The configuration file used to configure the BDII itself is /etc/bdii/bdii.conf. The format is key/value pairs with an '=' sign as the separator. A default configuration file comes with the BDII. This may require editing in order for the BDII to function as desired. The table belows describes the key/value pairs found in the configuration file.

Key	Typicaly Value	Description
BDII_LOG_FILE	/var/log/bdii/bdii-update.log	The log file for the BDII update process
BDII_PID_FILE	/var/run/bdii/bdii-update.pid	PID file for the bdii-update daemon
BDII_LOG_LEVEL	ERROR	The log level for the update process [ERROR, WARNING, INFO, DEBUG]
BDII_LDIF_DIR	/var/lib/bdii/gip/ldif	The directory containing LDIF files
BDII_PROVIDER_DIR	/var/lib/bdii/gip/provider	The directory containing information providers
BDII_PLUGIN_DIR	/var/lib/bdii/gip/plugin	The directory containing plugins
BDII_PORT	2170	The port which is used for the LDAP server
BDII_VAR_DIR	/var/lib/bdii	The directory to use by the BDII for writing data
BDII_BREATHE_TIME	120	Time to wait before updating the next database
BDII_READ_TIMEOUT	300	Time to wait for LDAP sources to return
BDII_ARCHIVE_SIZE	0	The number of updates that the changes should be logged
BDII_USER	ldap	The user runing the update process and the slapd databases
BDII_DELETE_DELAY	43200	Time to wait in seconds before deleting removed entries. Default is 12 hours. This variable enables the caching mode. For disabling it set it to 0.

5.2 Starting and Stopping the BDII

The BDII is started and stopped by the daemon script /etc/init.d/bdii. The following commands can be used:

```
service bdii start
service bdii stop
```

5.3 File Locations and Descriptions

The following table contains a list of files and locations which may be useful during troubleshooting.

Location	Description
/etc/bdii/bdii.conf	BDII configuration file
/etc/bdii/bdii-(top-)slapd.conf	The slapd.conf template for use with the bdii
/var/lib/bdii/gip/ldif/default.ldif	A default LDIF file to populate the bdii
/etc/init.d/bdii	BDII init.d script
/usr/sbin/bdii-update	Main update script
/opt/bdii/bin/bdii-proxy	Creates proxy for the BDII
/var/log/bdii/bdii-update.log	The BDII log file
/var/run/bdii-update.pid	Te bdii-update.pid of the main process

5.4 Running Processes

When a BDII is started, the following processes run:

- One multithreaded slapd process.The number of (active) threads may depend on the query load and/or the /opt/bdii/etc/bdii-slapd.conf file.
- 1 bdii-update process.
- Periodically, one ldapadd, ldapdelete or ldapmodify process.

CHAPTER 6

Information System for Users

6.1 Support

Please, use GGUS for any enquiry related to the Information System and also to report any incident.

6.2 User guides

- *ginfo*: A client to query GLUE 2.0 information from the BDII.
- `lcg-info/lcg-infosites`: Two different clients to query GLUE 1.3 information from the BDII. This guide also contains a detailed description of how to use the ldapsearch command. Note that lcg-info and lcg-infosites are now frozen in terms of new functionality and are maintained in a best effort basis.

6.3 BDII clients

- LCG_GFAL_INFOSYS configuration: Best practices from a client perspective for top-BDIIs.

6.4 User requirements

- Information System User requirements: Twiki to track general user requirements for the Information System.

6.5 WLCG evolution

In June 2015, OSG announced their plans to stop using the BDII to publish their computing resources (See `Slides` presented at the WLCG Operations Coordination Meeting in 18th of June). This announcement has triggered the review of the current WLCG Information System. It has been decided to create a task force to evaluate how WLCG is

going to evolve to be able to cover the existing use cases and finally improve all the existing drawbacks and weaknesses of its current implementation.

CHAPTER 7

ginfo

ginfo is a client tool for GLUE 2.0. It queries information from the BDII and lists the attributes corresponding to an object. By default, all the attributes of an object are displayed.

ginfo is available in EPEL 5 and EPEL 6 repositories.

7.1 Usage

```
ginfo [options] Object [attribute_to_filter='value of the attribute'] ␣
↪[attribute_to_display]
```

Only the object is mandatory.

7.1.1 Options:

```
-H,  --host  host        Specify a host to query. By default the environmental␣
↪variable LCG_GFAL_INFOSYS will be used.
-b,  --bind  binding     Specify the binding (o=glue by default).
-l,  --list  attribute   List all the possible values of the specified attribute.
-c,  --csv               Output in CSV format
-j,  --json              Output in JSON format
-t,  --timeout           Change the ldap timeout (15 seconds by default).
-v,  --verbose           Enable verbose mode
-V,  --version           Print the version of ginfo
-h    --help             Print this helpful message
```

7.1.2 Objects and corresponding attributes:

Object	Attributes
AdminDomain	ID, Description.
ComputingManager	ID, ProductName, ProductVersion, ServiceID.
ComputingShare	ID, MaxCPUTime, MaxWallTime, ServingState, ExecutionEnvironmentForeignKey, RunningJobs, WaitingJobs.
Endpoint	ID, URL, Capability, InterfaceName, InterfaceVersion, Implementor, ImplementationVersion, QualityLevel, HealthState, ServingState, ServiceForeignKey.
ExecutionEnvironment	ID, OSName, ConnectivityOut, MainMemorySize, VirtualMemorySize.
Location	ID, Country, Latitude, Longitude.
MappingPolicy	ID, Scheme, Rule, ComputingShareID.
Service	ID, Capability, Type, QualityLevel, StatusInfo, AdminDomainID.

7.2 Output Format

7.2.1 Standard output for an Endpoint:

```
HealthState: Value
Implementor: Value
InterfaceName: Value
ServingState: Value
URL: Value
ImplementationVersion: Value
Capability: Value
ServiceForeignKey: Value
QualityLevel: Value
ID: Value
InterfaceVersion: Value
```

7.2.2 JSON output for an Endpoint:

```
[... "Value_of_the_ID": {
"HealthState": Value,
"Implementor": Value,
"InterfaceName": Value,
"ServingState": Value,
"URL": Value,
"ImplementationVersion": Value,
"Capability": Value,
"ServiceForeignKey": Value,
"QualityLevel": Value,
"ID": Value,
"InterfaceVersion": Value}, ...]
```

7.2.3 CSV output for an Endpoint:

```
HealthState,Implementor,InterfaceName,ServingState,URL,ImplementationVersion,
↪Capability,ServiceForeignKey,QualityLevel,ID,InterfaceVersion
```

7.3 Examples

- List all information for all Endpoint attributes:

```
ginfo --host bdii.example.com Endpoint
```

- Use the host from the LCG_GFAL_INFOSYS environment variable and list all Location countries:

```
export LCG_GFAL_INFOSYS=bdii.example.com:2170
ginfo Location country
```

- List all the Service types:

```
ginfo -l Type Service
```

- List all IDs from Endpoint which have 'org.glite.FileTransfer' as name of Interface:

```
ginfo Endpoint InterfaceName=org.glite.FileTransfer ID
```

- Show the version too:

```
ginfo Endpoint  InterfaceName=org.glite.FileTransfer  ID  InterfaceVersion
```

- Show all available information about these Endpoints:

```
ginfo Endpoint InterfaceName=org.glite.FileTransfer
```

- Export to CSV:

```
ginfo --csv Endpoint InterfaceName=org.glite.FileTransfer
```

7.4 Support

In case of problems, please open a GGUS ticket.

CHAPTER 8

Information System for Developers

8.1 Information System Developers

- *Developers Guide*
- BDII testbed
- *Test Plan*
- *Tests*
- *GLUE Validator Developers Guide*: A guide on how to contribute to the GLUE Validator code
- BDII bugs: Jira tracker where changes in the BDII are tracked
- BDII Performance Studies
- Improving the Currency of Information in Large-Scale Grid Information Systems

8.2 Middleware Developers

- *Resource BDII*: A guide for middleware developers on how to integrate a Resource BDII with their services
- resource_provider_guide: A guide on how to write information providers
- Publishing a new service: A guide summarising the steps to publish a new service in the information system
- GLUE 2.0 issues with Information Providers: Twiki to track GLUE 2.0 open issues with Information System providers

CHAPTER 9

Developers Guide

BDII is currently released in UMD repositories and EPEL repository.

9.1 Communication

The project-grid-info-devel email list is used for communication between BDII developers. To join this list, fill in the form on the egroups subscription page.

The EGI URT email list is used to plan UMD releases. Make sure planned releases are announced with enough time in advance at the URT meetings.

9.2 Software Repository

The CERN Subversion service is used for the software repository and can be browsed using the SVN Browser. For write access to the repository, email the *project-grid-info-devel* list. The software can be checked out with the following command.

```
svn co https://svn.cern.ch/reps/gridinfo/[COMPONENT]/trunk
```

For anonymous checkouts the following command can be used.

```
svn co http://svn.cern.ch/guest/gridinfo/[COMPONENT]/trunk
```

COMPONENT refers to the concrete component you want to check out. Components could be: bdii, gstat-web, etc. Example:

```
svn co https://svn.cern.ch/reps/gridinfo/bdii/trunk
```

9.3 Version Control

To release a component it MUST be tagged with R_x_y_z, where x, y and z are the major, minor and patch levels respectively. It must be ensured that the version numbers for the package will agree with the svn tag and that they always increase with newer versions! Tagging is achieved with the following command.

```
svn copy https://svn.cern.ch/reps/gridinfo/example/trunk https://svn.cern.ch/reps/
↪gridinfo/example/tags/R_x_y_z -m "New Release"
```

9.4 Building

Packages released in the EMI 3 repositories MUST be built with the CERN koji build system. Check the How to Build packages in the Agile Project Infrastructure to get a koji account and learn all the details to use the CERN koji build system.

If a package is released for the first time, the following commands need to be executed:

```
koji add-pkg --owner=malandes bdii6 package-name
koji add-pkg --owner=malandes bdii6-testing package-name
koji add-pkg --owner=malandes bdii6-stable package-name
koji add-pkg --owner=malandes bdii7 package-name
koji add-pkg --owner=malandes bdii7-testing package-name
koji add-pkg --owner=malandes bdii7-stable package-name
```

The command to build the package in koji is:

```
koji build bdii6 srpm-package
koji build bdii7 srpm-package
```

If the build is successful the package will be available in the testing repository SL6 and SL7. In order to install packages from the testing repository, use the following repo files (Example below for SL6, please change bdii6 with bdii7 for the SL7 repo):

```
[bdii6-testing]
name=bdii6-testing
baseurl=http://linuxsoft.cern.ch/internal/repos/bdii6-testing/$basearch/os/
enabled=1
gpgcheck=0
priority=30
```

9.5 Testing and Quality Assurance

Once the testing repository has been created, testing can begin. The testing required is outlined in the *Test Plan*. The tests described in the test plan MUST be carried out for each component before it is considered for release. The CERN Openstack infrastructure currently hosts the BDII testbed.

It has to be noted as well that all the standard coding conventions should be followed and installed software should use the directory structure as specified in the Filesystem Hierarchy Standard. Packages should also be sanity checked before being moved into the testing repository.

For integration testing with other middleware services, notify middleware developers that a new BDII update is available in the testing repository through the EMI EMT mailing list.

For testing in production conditions, notify sites that are participating in early evaluation that a new BDII update is available in the testing repository. Release

Packages to be released in EMI 3 that have succesfully passed all tests can be promoted to the stable repository by running:

```
koji tag-pkg bdii6-stable package-name-version.el6
koji tag-pkg bdii7-stable package-name-version.el7
```

This will create the packages also in the Linuxsoft repository.

The release should be announced and documented in the *BDII Releases*. This page offers an RSS feed.

The corresponding JIRA and GGUS tickets that are fixed in the release, should be closed including a link to the release notes.

This action MUST be accompanied with a notification to the *EMI EMT* mailing list and the URT mailing list. The EMI and UMD release managers will take care of populating the relevant repositories. Note that only UMD releases are guaranteed by a succesful staged rollout

9.6 Defect Tracking

- The main entry point for users and system administrator is GGUS.
- GGUS notification are sent to the project-grid-info-support email list.
- Defects and enhancements are tracked using the Grid Information System Jira tracker.

9.7 EPEL

BDII releases are also released in EPEL. In order to release a package in EPEL you need to become a package maintainer. Find out more details in the following links:

- How to get your packager status
- Join the package collection maintainers
- How to get sponsored into the packager group
- Package review process

The status of EPEL releases for the BDII is sumarised in this twiki. For further details, please use the Fedora Package database.

CHAPTER 10

GLUE Validator Developers Guide

10.1 Overview

The GLUE Validator is a simple python-based unit test suite that validates information against test cases. It works against both LDIF files and LDAP servers. In addition, a GLUE 2.0 LDAP schema validator is available.

10.2 Installation

To checkout the source code run the following command.

```
svn co http://svn.cern.ch/guest/gridinfo/glue-validator/trunk
```

To install an rpm, run the following command.

```
cd trunk; make clean rpm; rpm -Uvh build/RPMS/noarch/glue-validator-*-1.noarch.rpm ;␣
↪cd -
```

For the glue2-schema-check, you will need to install the glue-schema and openldap-servers packages.

10.3 Usage

To see usage, run the following command.

```
glue-validator
```

or

```
glue-schema-validator
```

Alternatively, to run from source the PYTHONPATH will need to be set. This can be achived by running the following command.

```
export PYTHONPATH=${PYTHONPATH}:${PWD}/trunk/lib
```

Then the following command should work.

```
./trunk/bin/glue-validator
```

or

```
./trunk/bin/glue-schema-validator
```

10.3.1 Note For Ubuntu

To get this working on Ubuntu you will need to disable apparmor for slapd.

```
sudo ln -s /etc/apparmor.d/usr.sbin.slapd /etc/apparmor.d/disable/.
sudo /etc/init.d/apparmor reload
```

To installed the glue-schema, run the following commands.

```
svn co http://svn.cern.ch/guest/gridinfo/glue-validator/trunk
cd trunk; make clean deb; dpkg -i build/glue-schema_*_all.deb ; cd -
```

10.4 Architecture

The testing framework consists of two main parts, the main script and the validator library.

10.4.1 The Main Scripts

The purpose of the main script (glue-validator) is to parse the command line arguments, read the LDIF/LDAP source, generate all the tests and to run the test suite. To achieve these goals, it makes use of several libraries, validator.utils and validator.*Test. The script iterates over each entry found in the LDIF/LDAP source and for each entry adds a number of tests to the test suite. After all the tests have been added, the test suite is run.

For all GLUE versions the main script runs validator.EntryTest. For the EGI profile for GLUE 2, the glue-validator also runs validator.EGIProfileTest. This library contains tests for each GLUE 2 attribute as described in the EGI profile document.

10.4.2 The Libraries

validator.utils

This library is used to parse the command line options and read the LDIF/LDAP source. It contains one or two tools for manipulating LDIF. It also contains the code to generate nagios output and a function to display failure messages in a standard way that could be then easily parsed. The internal structure of storing entities is as follows.

```
{ dn : { attribute : [ value ]  }  }
```

validator.EntryTest

This library contains the EntryTest class, which extends the standard python unittest.TestCase. This test class provides four tests; test_object_class, test_mandatory_attributes, test_single_valued, test_data_types. This library uses the data and types libraries from the test classes being used. One point to note is that the test_data_types call a function with a dynamically name of the format is_type, where the type is obtained from the data structure provided by data and the function is provided by the library types.

validator.EGIProfileTest

This library contains the EGIProfileTest class, which also extends the standard python unittest.TestCase. This test class provides one or more tests per GLUE 2 attribute. This library uses the data and types libraries from the egi-glue2 library where the EGI profile for GLUE2 definition in terms of attributes and types is specified.

validator.WLCGTest

This library contains specific tests from EGIProfileTest class thar are particularly relevant to WLCG.

data.py

This library essentially contains a description of the schema. The data structure schema has the following format.

```
{ objectclass : { attribute : ( type, single_valued, mandatory ) } }
```

Where type is a string representing the data type of the attribute, single_valued is a boolean indicating whether or not the attribute is single valued and mandatory is a boolean indicating whether or not the attribute is mandatory.

types.py

This library contains functions that test values and return True or False depending on whether or not the input value conforms to that type.

validator.messages.py

This library contains the list of failure messages for Errors, Warnings and Info messages. It is a dictionary containing the failure message code, its description and the affected attribute.

validator.KnownIssues.py

This library contains a list of the tests that are known to validate attributes that are wrongly published by the BDII or the information providers. By selecting the -k,–exclude-known-issues option in the command line, these tests are not executed by the glue-validator.

10.4.3 Testing

For each object class there should be a library of the form EntityTest.py. These are the tests for the GLUE Validator suite and can be executed directly and will be used for regression testing.

CHAPTER 11

Information Provider Guide

This guide explains how to create information provider for use with the BDII. It is intended to help developers create information providers for their specific service. Any questions, comments or suggests can be sent to the project-grid-info-support email list.

11.1 GLUE 2.0

Any information published about the service must confirm to the GLUE 2.0 information model. This document contains a detailed description of the allowed attributes and attribute groupings. To compliment the specification, an LDAP rendering document (doc pdf)is also available. Examples of the resulting LDIF file can be found here.

Before any attempt is made to create and information provider, it would be useful to become familiar with the GLUE 2.0 information model. It is also recommend to become familiar with LDAP and LDIF.

11.2 LDIF, Provider or Plugin

An information provider and be provided in three different forms (*LDIF*, *provider*, *plugin*) and the first step is to decide which should be used. It may be necessary to provide the information using a combination of all three.

11.2.1 A standalone LDIF file

An LDIF file should be used for static information that can be set during configuration time and will not change. This can contain default values and entires that my be overwritten by a provider or plugin.

11.2.2 Provider

An information provider is this is a script that returns LDIF to stdout.

11.2.3 Plugin

A plugin is a script that returns LDIF containing only LDAP modify statements. For more detail on the syntax see the man page of the ldapmodify command.

Examples of how LDIF files, plugins and providers work can be found here.

11.3 Integration with the BDII

The LDIF files, providers and plugins are deployed by placing them in the relevant directory as specified in the bdii.conf file. See the *Resource BDII* documentation for more details.

11.4 Testing

Use the *GLUE validator guide* tool to make sure the information provider correctly generates GLUE1.3 and GLUE 2.0 information.

CHAPTER 12

Resource BDII

The resource-level BDII is an instance of the BDII that contains information about a Grid Service and is typically deployed with the service itself. It consists of an OpenLDAP database that is periodically updated by a parallel process that obtains information about the Grid Service from one or more information sources. For more detailed information about the BDII, please read the *Grid Information System Introduction*. All services MUST publish their existence in both the GLUE 1.3 and GLUE 2.0 data models.

For any further questions, please contact project-grid-info-support.

12.1 Installation

To provide a resource-level BDII, a dependecy must be set on the following rpms:

- bdii
- glue-schema
- glite-info-provider-service

The latest version is available in the EMI repositories.

12.2 Configuration

The BDII should need no further configuration, however, information on the configuration parameters used by the BDII are described in the BDII *Quickstart guide*. The service can be started with the following command.

```
/sbin/service bdii start
```

12.2.1 Information Provider Setup

By default, the BDII uses the following three directories to obtain information sources.

```
/var/lib/bdii/gip/ldif
/var/lib/bdii/gip/provider
/var/lib/bdii/gip/plugin
```

These directories are specified as configuration parameters in the BDII's configuration. Static LDIF files should be placed in the ldif directory, information providers should be put in the providers and information plugins should be put in the plugin directory.

12.2.2 Service Information Provider

By default, all services should publish themselves by using the service information provider package. A number of template configuration files for some services can be found in /etc/glite/info/service. If a template file can be used, a configuration file can be created by running the following command.

```
cp  /etc/glite/info/service/glite-info-service-xxx.conf.template /etc/glite/info/
↪service/glite-info-service-xxx.conf
```

where xxx is the name of the service. The file may need to be edited to include information which is only available at configuration time, e.g. the list of supported VOs.

The service information provider can be added by creating a wrapper script in the _provider" directory.

```
cat  << EOF > /var/lib/bdii/gip/provider/glite-info-provider-service-xxx
#!/bin/sh
/usr/bin/glite-info-service /etc/glite/info/service/glite-info-service-xxx.conf MYSITE
/usr/bin/glite-info-glue2-simple /etc/glite/info/service/glite-info-service-xxx.conf␣
↪MYSITE
EOF

chmod +a /var/run/bdii/gip/provider/glite-info-provider-service-xxx
```

where xxx is the name of the service. Environment variables can be set to be used in the provider and MYSITE is the GlueSiteUnqiueID. More details on the service information provider can be found in the README and README-GLUE2 file.

12.2.3 Service Specific Information Providers

For any other service specific information, LDIF file, information providers and plugins may need to be created and placed into the respective directories. For further advice on what is required for a specific service, please contact *project-grid-info-support*.

12.3 Testing

Basic testing can be achieved by doing an ldapsearch and looking at the output.

12.3.1 Test that the BDII is operational

```
ldapsearch -x -h $(hostname) -p 2170 -b o=infosys
```

12.3.2 Test that the BDII is updating

```
ldapsearch -x -h $(hostname) -p 2170 -b o=infosys "*" modifyTimestamp
```

12.3.3 Test that the Glue 1.3 root entry is available

```
ldapsearch -x -h $(hostname) -p 2170 -b mds-vo-name=resource,o=grid
```

12.3.4 Test that the Glue 2.0 root entry is available

```
ldapsearch -x -h $(hostname) -p 2170 -b o=glue
```

12.3.5 GLUE Validation tests

The *GLUE validator guide* should also be used to test for GLUE 1.x and GLUE 2.0 conformance.

CHAPTER 13

Test Plan

13.1 Philosophy

The Grid Infrastructure is composed of many Grid Services that work together to provide users with the resources they need to achieve their goals. As such, the Grid is "service-centric" and hence the service is the focus of the process. This concept fits well with The Information Technology Infrastructure Library (ITIL) practices for managing Information Technology services. In particular the Service Support discipline can be leveraged, which addresses Incident Management, Problem Management, Change Management, Release Management and Configuration Management.

Grid Services are able to work together in the infrastructure as they implement well defined interface specifications (eg. HTTP for the web). Failure to strictly adhere to these specifications will break the infrastructure and such these interfaces must be fixed and owned by an oversight body for the infrastructure (eg. W3C). A Grid Service must always adhere to these specifications and a service update must not break or diverge from the specification. If a new specification is required, this must be endorsed by the oversight body for that infrastructure and implemented in a way that is backwards compatible. This typically requires services to support two versions during the migration period (eg. IPV4 to IPV6 transition).

As Grid Services are linked by the interface specification alone, services may be provided by multiple vendors as long as each has faithfully implemented the interface specification. Multiple versions of a service should be able to coexist in the infrastructure. The only exception is after a specification migration when the old version becomes obsolete.

It must be pointed out a service can be simple, a daemon process running on a single machine, or complex, hundreds of machines running different clustered sub services such as databases and web servers. The user is typically hidden from the implementation details and only experiences the service provided via the interface, however, the Service Provider does care about the implementation details of the service. In the case of a complex service, the components of that service may be provided by many different vendors and in the scenario where sub services are used, these are also linked by interface specifications. The suggest that where as the infrastructure is integrated at the service level, services are are integrated at the sub service level or component level.

It must also be noted that developing stand-alone program and running an on-line service are two very different things. Instead of releasing a new version of a program every couple of years, on-line services live in a perpetual cycle of product improvement and instantaneous user feedback. As such, service development typically takes a roll-forward approach where the instance of the service should always be running the latest version and fixes are provided in a newer version, rather than applying the fixes to a branch of an old version.

13.2 Testing Strategy

The focus of the strategy is the concept of a service update. A service is a collection of stand-alone software components that work together to provide the service. A service update is a sub-set of these components which have a newer version than what already exists for the service. This naturally divides the process of providing a service update into the update itself and providing an update to the component. As a component may be used in many other places other than the service, these to processes should be view as decoupled from each other. Development represents the area of producing the component and Integration is the process of creating a service update from a collection of components. During Development, developers are responsible for the quality of the components which they maintain. As they know the component in detail, they should also understand how best to test the component. Each component should contain tests that can be run in developer mode ie. directly from the VCS. The implementation of this is the choice of the developer, however, the testing method must be documented. Before a release tag is made for the component, the developer must run the tests and ensure they all pass. The focus of this level of testing is to verify all the supported functionally of the component. The process of Integration starts when all the components are available and ends with the provision of a service update. It is the responsibility of the integrator to ensure that the service update is of high quality. In order to achieve this goal a multi-level testing strategy will be employed.

13.2.1 Level 0: Component

Aim: To verify the main functionality

Each component should be accompanied by a set of tests that verifies the basic functionality of that component. These tests should be included in the same source repository and be able to run both during development and final deployment. The tests should be run by the developer before the component is tagged and hence the developer is responsible for developing these tests. A test for the new functionally or software bugs should be created before development work commences.

13.2.2 Level 1: Service Ping

Aim: To verify that the service up.

The result of the integration phase is a set of deployable packages along with configuration methods. After an instance has been installed and configured, it must be verified that the service is working. Typically a service is up or down and hence a simple service 'ping' test should suffice. The tests are required during day to day operations to ensure that the service is up and hence can be reused for this purpose.

13.2.3 Level 2: Service Functionality

Aim: To verify the full functionality of the service.

After a service has been installed and it is up, functionality tests can be carried out to ensure that all the supported functionality is working. If possible, this can be achieve by re-running the component tests. However, additional tests will be required if this is not possible or the component tests do not cover functionality provided by the integrated service.

13.2.4 Level 3: Service Performance and Stress Testing

Aim: To verify the performance of the service.

A benchmarking test is required to measure the performance of the service. This should be run to ensure that each new release does not degraded the performance. In additional, how the performance changes over time must be measured.

13.2.5 Level 4: Infrastructure

Aim: To verify the full system work-flow.

Full end-to-end Grid work flows need to be tested. This will include verifying that different versions of the Services work together. This will be achieved via the use of a staged roll out process.

CHAPTER 14

Tests

14.1 Level 0: Component

14.1.1 bdii

Run the test-bdii script found in the tests directory in SVN.

14.1.2 bdii-config-site

Run the test-site-bdii script found in the test directory in SVN.

14.1.3 glite-info-provider-ldap

Run the run-tests script found in the test directory in SVN.

14.1.4 glite-info-site

Run the run-tests script found in the test directory in SVN.

14.1.5 glite-info-static

Run the run-tests.sh script found in the test directory in SVN.

14.1.6 glue-schema

Run the test-ldif.sh script found in the LDAP2 directory in OGF SVN mirror at CERN.

14.1.7 gstat-validation

No test currently available. This package contains tests itself.

14.1.8 gstat-web

Installation of gstat-dev and verification by manual inspection.

14.1.9 lcg-info

Testing method unknown

14.1.10 lcg-infosites

Testing method unknown

14.1.11 nagios-plugins-bdii

No test currently available. This package contains tests itself.

14.2 Level 1: Service Ping

14.2.1 Site BDII

The package nagios-plugins-bdii contains a probe check_bdii_entries that can be used to connect to the BDII and measure the response time as well as the number of entries returned. The gstat-validation contains probes that can be used to validate the information content.

14.2.2 Top BDII

The package nagios-plugins-bdii contains a probe check_bdii_entries that can be used to connect to the BDII and measure the response time as well as the number of entries returned.

14.3 Level 2: Service Functionality

14.3.1 Site BDII

The service should be first installed using the prepare script and the test-BDII-site script should be run.

14.3.2 Top BDII

The service should be first installed using the prepare script and the test-BDII-top script should be run.

14.4 Level 3: Service Performance and Stress Testing

The package ldapbench is used to stress and instance of the BDII. This is achieved by executing multiple quires on a number of threads to generate a query load an measuring the response time.

14.5 Level 4: Infrastructure

Infrastructure tests are part of the Staged Rollout activity managed by the EGI project. For more information, please check the EGI Staged Rollout pages.

CHAPTER 15

GLUE

15.1 GLUE schema

- GLUE 1.3
  - Glue Use: The Use of Glue version 1.3 in EGEE
  - Installed Capacities: How to publish the install capacities in WLCG
- GLUE 2.0
  - EGI profile document for GLUE 2.0
  - GLUE 2.0 in HTML for easy browsing
  - GLUE 2.0 monitoring reports: A twiki collecting:
    - Monthly reports on the status of GLUE 2.0 information quality for WLCG
    - Known issues with GLUE publication in the BDII
- OGF GLUE Working Group
- Open Enumerations

15.2 GLUE Validator

- glue-validator guide: A guide on how to use the GLUE validator
- glue-validator training: An online webinar to learn how to use glue-validator. Find the latest version of the slides used for the webinar here (please, use this version when following the recording).
- glue-validator error codes: A twiki page describing all the existing glue-validator tests and their associated error codes

15.3 GLUE Monitoring

- EGI Nagios monitoring dashboard for GLUE 2
- BDII deployment metrics
- GLUE 1 and GLUE 2 deployment metrics
  - GLUE 1 and GLUE 2 per site
- WLCG GLUE 2 validation metrics
- Storage metrics
  - Storage Deployment metrics for DPM, dCache, StoRM, Castor and EOS
  - T1 Storage metrics
- LHCb Information System metrics:
  - Maximum CPU time
  - BDII vs SRM Storage Capacity numbers
  - GLUE validator LHCb
- ATLAS Information System metrics:
  - xroot and http endpoints in the BDII
- ALICE Information System metrics:
  - Waiting Jobs

CHAPTER 16

GLUE validator guide

The glue-validator command is a very useful command for system administrators and middleware developers who want to validate whether the information published by the service they are managing or developing is compliant with Glue 2.0 and Glue 1.3. For more information about the GLUE schema, please visit the *GLUE* section in this web.

The glue-validator is also able to validate against the EGI profile for Glue 2.0. This is the recommended and default validation as it specifies how the Glue 2.0 information schema should be used in EGI. The EGI profile gives detailed guidance on what should be published, how the information should be interpreted, what kinds of uses are likely and how the information may be validated to ensure accuracy.

16.1 Glue-validator Error Codes

Check the glue-validator error codes containing tips to fix the errors.

** Important Note: This user guide documments glue-validator version >= 2.0.20.**

16.2 Support

Please, use GGUS for any enquiry related to the glue-validator and also to report any incident.

16.3 Quickstart guide

The glue-validator can be downloaded from CERN Linuxsoft (SL5 and SL6) repository and also EMI 3 repository (Bear in mind that EMI 3 repository is updated once per month and there may be some delay to propagate the very last version there).

The glue-validator command is used in the following way:

```
Usage: ./glue-validator [LDIF OPTIONS] [-g] [-s] [Other Options]

Mandatory Arguments:

Server Mode: Obtains LDIF from an OpenLDAP server
 -H --hostname      Hostname of the LDAP server
 -p --port          Port for the LDAP server
 -b --bind          The bind point for the LDAP server

File Mode: Obtains LDIF directly from a file
 -f --file          An LDIF file

Optional Arguments:

GLUE version: Selects the GLUE schema version to be tested
 -g --glue-version        The glue schema version to be tested [glue1|glue2|egi-glue2␣
↪(default)]

Tesuite type: Selects the set of tests to be executed against the LDIF
 -s --testsuite     The testsuite  [general|egi-profile (default)]

Other Options:
 -k --exclude-known-issues  Do not run tests for wrongly published attributes due to␣
↪known bugs
 -t --timeout               glue-validator runtime timeout, default 10s
 -v --verbose               Verbosity level 0-3, default 1
 -r --separator             Defines the separator for the output messages, default \n
                            This is only available for the verbosity level 3.
 -V --version               Prints glue-validator version
 -h --help                  Prints glue-validator usage
```

16.3.1 Examples:

- EGI profile against GLUE 2.0 validation:

```
glue-validator -H localhost -p 2170 -b o=glue
```

- GLUE 1.3 validation:

```
glue-validator -H localhost -p 2170 -b o=grid -g glue1 -s general
```

- GLUE 2.0 validation:

```
glue-validator -H localhost -p 2170 -b o=glue -g glue2 -s general
```

By default, the output of the command reports a summary with the number of Errors, Warning and Info messages.

The command can be run with -n option which allows for 3 different verbosity levels. Level 1 is the output explained in the previous paragraph; Level 2 is a summary per error, warning and info messages; Level 3 gives more details for each message.

The messages for the EGI profile for GLUE 2 validation are documented in detail in the following twiki page.

16.4 EGI profile for GLUE 2.0 compliance

To validate against the EGI profile for GLUE 2.0, the following command must be run, replacing the site-bdii.host and port with the hostname and port of the site BDII:

```
glue-validator -H site-bdii.host -p port -b o=glue
```

Note that in this example the validation is done at site level but it can be done at any level.

Querying a site may take some time, in some cases it is necessary to define the -t option with a reasonable timeout value to leave the site BDII enough time to respond to the query.

When validating against the EGI profile for GLUE 2.0 it is recommended to choose Nagios output and verbose level of '3' for all the details, otherwise '2' for a summary of the encountered problems.

If the validation is done by a site admin, it is recommended to run the tool using the option –exclude-known-issues. This option will not run tests to variables that are wrongly published due to known bugs in the middleware information providers:

```
glue-validator -H site-bdii.host -p port -b o=glue --exclude-known-issues
```

Tips to know what to do to fix the errors reported by glue-validator can be found in this twiki.

16.5 Glue 2.0 compliance

To test GLUE 2.0 information, the following command must be run, replacing the site-bdii.host and port with the hostname and port of the site BDII.

```
glue-validator -g glue2 -H site-bdii.host -p port -b o=glue -s general
```

Note that in this example the validation is done at site level but it can be done at any level.

16.6 Glue 1.3 compliance

To test GLUE 1.3 information, the following command must be run, replacing the site-bdii.host and port with the hostname and port of the site BDII.

```
glue-validator -g glue1 -H site-bdii.host -p port -b o=grid -s testsuite
```

Note that in this example the validation is done at site level but it can be done at any level.

CHAPTER 17

Docs

17.1 Presentations

- MB July 2016: Proposals and next steps
- GDB July 2016: Proposals and next steps
- GDB December 2015: WLCG Information System Status
- GDB September 2015: WLCG Information System Status
- WLCG Operations Coordination Meeting June 2015: Information System Use Cases
- GDB September 2014: WLCG Information System Status
- White Area Lectures June 2014: The Grid Information System
- CMS Computing Project Office Meeting May 2014: Prototype CMS Information System Based on AGIS (Restricted Access)
- 3rd LHCb Computing Workshop May 2014: BDII Use Cases for LHCb (Restricted Access)
- GDB April 2014: WLCG Information System Status
- GDB December 2013: GSR
- Common Projects Brainstorming November 2013: Information/Configuration Systems
- GDB October 2013: WLCG Information System Status
- GDB March 2013: WLCG Information System Status
- pre-GDB January 2013: Information System and the Experiments
- GDB December 2012: Requirements for GLUE 2.0
- GDB November 2012: WLCG Information System Status and Evolution
- EGI TF September 2012: EMI Registry Pilot Service
- GDB September 2012: Multi-core support in IS

- GDB July 2012: How to identify the best top level BDIIs?
- GDB June 2012: WLCG Information System Status and Evolution
- GDB June 2011: Cached BDII Update and WLCG Information System Use Cases
- GDB May 2011: Update on the WLCG Information System
- GDB January 2011: EMI status and plans and Information System Open Ends
- LCG Management Board December 2010: Short term improvements for the WLCG Information System
- Tier 1 SC Meeting November 2010: Top level BDII proposal
- WLCG Reliability workshop November 2007: BDII story
- GDB March 2007: BDII - The EGEE Information System

17.2 Articles

- Grid Deployment Experiences: The path to a production quality LDAP based grid information system
- A Directory Service for Configuring High-Performance Distributed Computations
- Grid Information System Interoperability: The Need For A Common Information Model
- A performance study of monitoring and information services for distributed systems
- Benchmarking Grid Information Systems
- Improving the currency of Information in a large scale grid information system
- An Evaluation of Information Consistency in Grid Information Systems
- Towards a Global Service Registry for the World-Wide LHC Computing Grid
- The EMI Registry: Discovering Services in a Federated World
- GLUE 2 deployment: Ensuring quality in the EGI/WLCG Information System
- Consolidating WLCG topology and configuration in the Computing Resource Information Catalogue

CHAPTER 18

Useful links

- *Grid Information System Introduction*
- Report an incident or a request: GGUS
- Contact us

CHAPTER 19

Indices and tables

- genindex
- modindex
- search