NORDUGRID-??-??
6/8/2013

ARC Information System

Developer's Handbook

v0.1, last update: 2012-08-30

F. Paganelli and various contributions from all ARC developers

Contents

Chapter 1

Overview

The *ARC Information System* by *NorduGrid* [4] is a set of components of the *ARC Middleware* [7] that retrieves information regarding subsystems managed by the ARC middleware and publishes such information via LDAP and other protocols based on web services.

This document serves as a quick reference to those who want to extend any of the Information System subcomponents.

This document does describe the internal mechanisms of the ARC CE information system architecture.

This document does NOT describe the overall middleware architecture, which is sketched in other manuals (see below).

To understand this document, it is mandatory to have a wide overview of what the ARC Middleware and the Information System are. Please read at least the introductory chapters of the following documents:

- For a general description of the ARC Middleware, see [**?**]
- For a general description of the ARC Information System architecture, see [**?**]

1.1 How to use this document

Read the architecture chapter **1.2**, *ARC Information System Architecture* for a broad overview of the ARC Information System.

Read the CE Information System architecture chapter **1.3**, *ARC CE Information System Architecture* to understand the main highlights of the ARC CE information system.

Once the above are understood, it is suggested to refer to the chapter **1.4**, *Description of topics* to identify which component to work with, then proceed to the chapter that describes it.

1.2 ARC Information System Architecture

ARC Information System is a collection of custom crafted tools and several off-the-shelf products available on the open source market.

ARC features two main information system products, ARIS and EGIIS.

ARIS, ARC Resource Information System, operates locally on a CE and provides information about the services running on a box.

EGIIS[8], Enhanced Grid Information Indexing Service, is a lightweight ldap-based index that aggregates information about clusters.

The whole information system consists of the following:

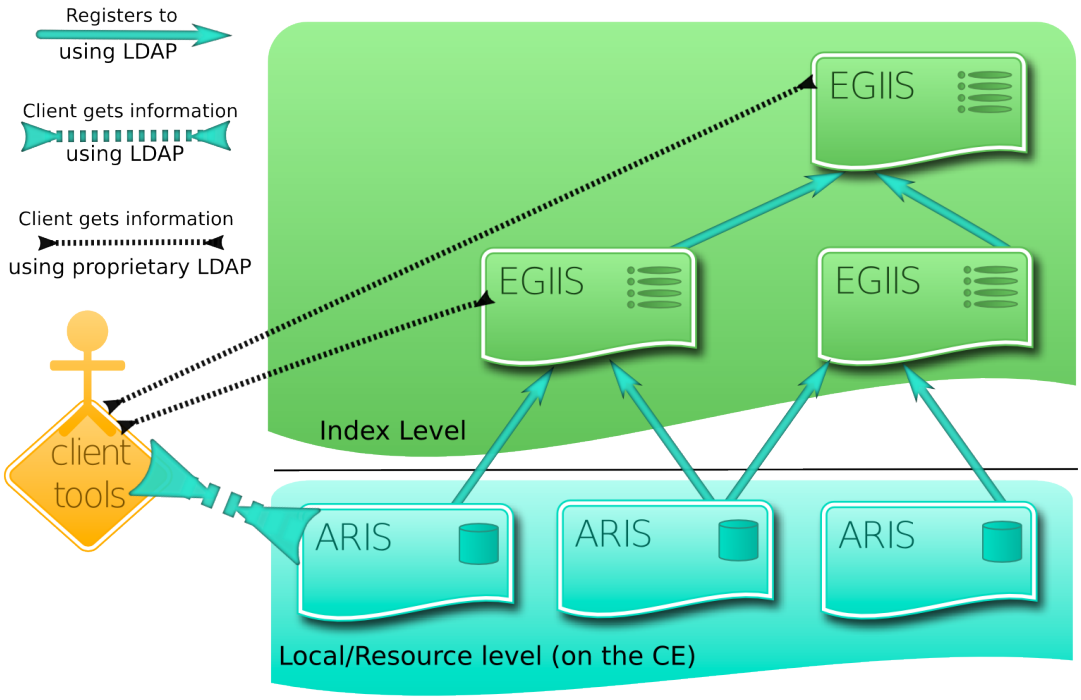

Figure 1.1: ARC information system "speaking" only the ARC "dialect".

A customized LDAP setup that implements both local and index level. It consists of the custom *NorduGrid schema*[8] and a custom LDAP-based index server called *EGIIS*[8].
It also consists of the open source LDAP server *OpenLDAP*[**?**] to publish LDAP information, and of a collection of open source Berkeley DB update scripts (*bdii-update*). Registration to an EGIIS index is achieved via custom scripts.

A web services interface based on the custom web service container HED (Hosting Environment Daemon or *arched*)[**?**] to serve local information and perform RESTful based registration.

A collection of Perl scripts the *Infoproviders*, capable of collecting information at the local level and produce representations in the form of XML or LDIF documents.

The two interfaces can live together but it is not mandatory to have both enabled. ARIS is fully functional even if just one of them is running.

1.2.1 ARC information system "speaking" only the ARC "dialect"

Figure 1.1 shows ARC information system architecture.

The lower level represents the local or resource level, where ARIS operates. In a complete ARC environment, various instances of EGIIS are organized in a hierarchy on the index level.

ARIS registers to a known EGIIS by sending small amount of information, performing simple standard LDAP operations.

Then, a client requesting information from any EGIIS must perform a custom LDAP query that will return a collection of EGIIS or ARIS endpoints/URLs.

The client will then contact the local endpoints directly using standard LDAP protocol queries to gather further information about local resources, or perform other custom queries to other EGIIS.

Width of arrows in Figure 1.1 is meaningful: amount of data exchange between ARIS and EGIIS and between clients and EGIIS is very little information, while exchange between clients and ARIS carries most of the available information, hence the thicker arrow.

1.2.2 ARC information system "speaking" other "dialects"

One of the purposes of ARIS is to easily connect and share information with multiple indexing and caching systems. Figure 1.2 shows how it interacts with existing caching technologies (BDII-top[3]) and the upcoming EMI Registry EMIR[**?**].

The orange band shows the ARC ecosystem, consisting only of ARC components. The blue band shows those software products that don't belong to ARC, such as gLite BDII-top and EMI's EMIR.

In the case of gLite, a pull model is used to gather information from an ARIS instance. Basically, BDII-top will have a list of URLs to contact the various LDAP servers for different services. By querying directly the ARIS local information system, it will pull all the data needed for caching, using standard LDAP searches. The thick arrow here shows that a large amount of information is transferred between ARIS and BDII-top. Note that the arrow between a client and a BDII-top is thick as well, because BDII-top is a cache and contains a lot of information that can be directly fetched.

In the case of EMIR, the registry is meant to contain minimal static GLUE2 endpoint information. The communication protocol towards EMIR is a RESTful web service interface. ARIS is capable of contacting EMIR and send a small amount of information regarding its GLUE2 services and endpoints. Note that, as in the ARC index EGIIS, the amount of information the client gets from EMIR is the minimal information to access services on a computing element where ARIS is running.

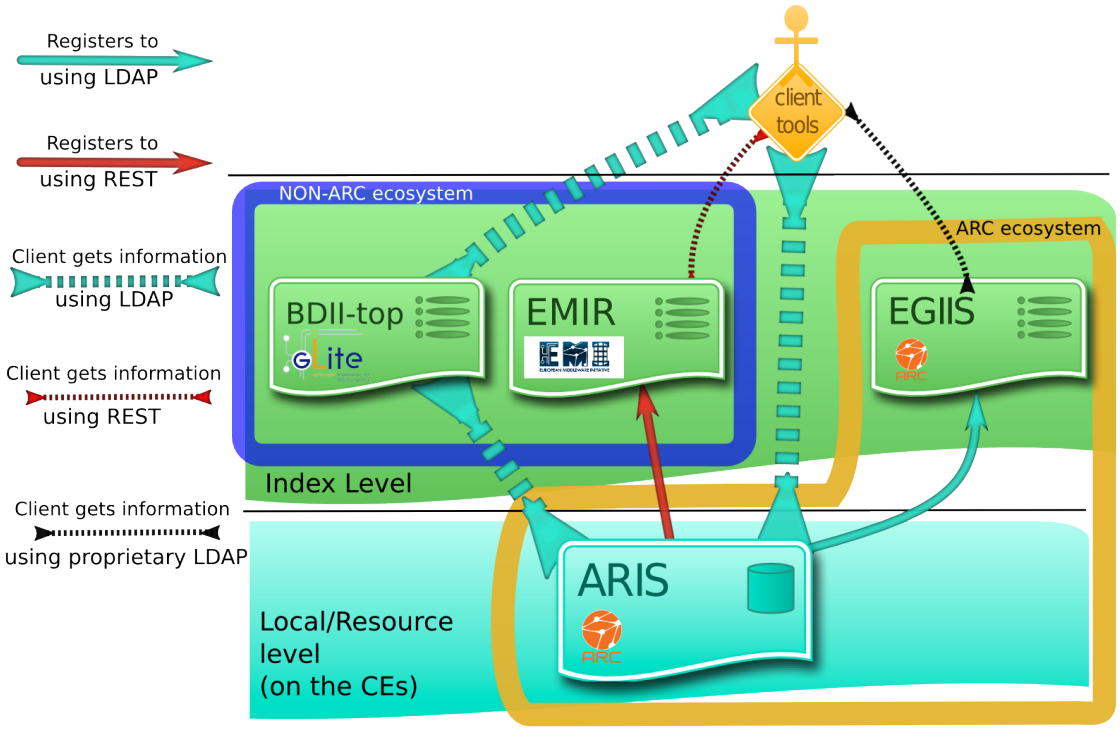

Figure 1.2: ARIS "speaking" other "dialects"

1.3 ARC CE Information System Architecture

In this section we will describe in more details ARIS components, supported schema and renderings, and the algorithm with which information is generated and served.

The process of communicating to different index levels is explained in Section 1.3.2.1.

1.3.1 Components

Below we describe ARIS internals as shown in Figure 1.3. *arched* is ARC's own open source web services container. Written in C++, it is extensible and allows running web services with different purposes. Job submission interfaces are served through arched, and they usually allow information system requests. A-REX, the ARC computing element and job management service, exposes three interfaces: WSRF, XBES (eXtended BES), EMI-ES (EMI Execution Service). While WSRF is basically used to obtain an XML information document, XBES and EMI-ES are well defined interfaces that feature information system retrieval functionality. All this information is generated from an XML document.

LDAP information is served via the open source LDAP server OpenLDAP and its daemon slapd. The slapd daemon manages and serves the information contained in Berkeley databases. LDAP information is described in terms of LDIF trees or LDIF documents[**?**]. Such information is updated by LDAP protocol[10] operations.

Another open source external component called *bdii-update*, a python script, takes care of gathering LDIF files representing database data and merges them in an efficient way so to perform add/delete/update LDAP operations against the slapd server.

The *infoproviders* are set of Perl scripts, developed by the ARC team, that can generate XML and LDIF documents needed by the former two subsystems. The information contained in these documents is shaped according to three main schema: NorduGrid[**?**], Glue 1.2/1.3 [**?**], GLUE2[**?**]. In particular, GLUE2 is rendered in both XML and LDIF, while NorduGrid and Glue 1.2/1.3 only comes as LDIF. All these can be published together. A fully enabled ARC CE can, at the same time, publish the all the combinations of schema and protocols listed in Table 1.1.

Schema	**Protocol**
NorduGrid	LDAP
Glue 1.3	LDAP
GLUE2	LDAP
GLUE2	WSRF
GLUE2-based	XBES
GLUE2-based	EMI-ES

Table 1.1: The ARIS schema and protocols they can be queried with. GLUE2-based means that the protocol might change the GLUE2 data structure according to the protocol needs.

The two startup scripts (*a-rex*, *grid-infosys*) configure all the subsystems and start them. It is important to note that even if ARC uses off-the-shelf component such as OpenLDAP and *bdii-update*, these are configured in one single central configuration file (arc.conf[**?**]) in such a way that configuration and startup of these services when working together with ARC is independent from their canonical system setup. This means that an organization can seamlessly run an LDAP server or a BDII system together with an ARC CE without changing a single line of their distribution-based configuration. ARC will configure and run special instances of those for itself.

1.3.2 High-Level Algorithm

The high-level algorithm with which the components interact is as follows:

1. Concurrently,
   (a) *a-rex* startup script configures and starts the *arched* daemon.
   (b) *grid-infosys* startup script configures and starts the OpenLDAP slapd daemon and the *bdii-update* script, then starts the whole LDAP subsystem.
2. *arched* periodically executes the *infoproviders*.

3. the *infoproviders* generate an XML document and a set of scripts able to generate LDIF trees, compliant to different schemas (see table 1.1)
4. Concurrently,
   (a) *arched* takes the XML document and wraps it to serve GLUE2 information over all the supported web services based protocols (WSRF, XBES, EMI-ES)
   (b) the *bdii-update* scripts use the LDIF generator scripts to update information served by slapd.

Figure 1.3 shows the components involved in the process.

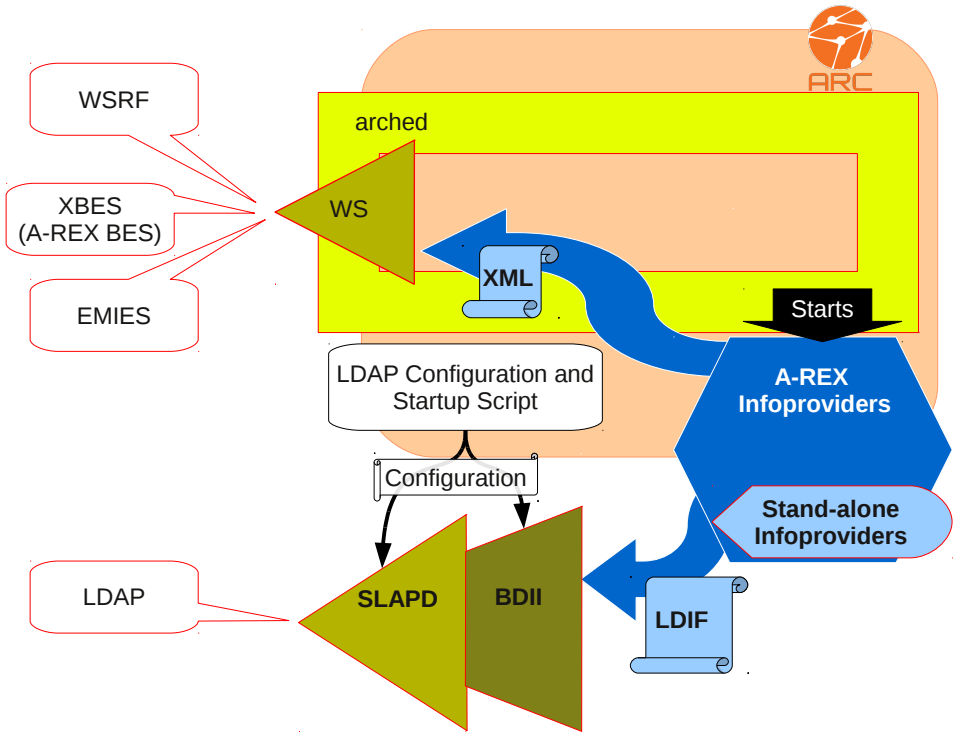

Figure 1.3: **ARIS components.** The orange container surrounds the ARC components. The grid-infosys startup script provides configuration files for slapd and BDII. BDII represents the Berkeley DB update scripts. Stand-alone infoproviders can be components external to ARC but also legacy infoproviders prior to ARC v.1.0.0.

1.3.2.1 Connection to the Index level

As mentioned in Section 1.2.2, ARIS can talk to different index level information systems.

This is achieved by the different subsystems enforcing the technologies the index level supports.

Figure 1.2 shows which interface is responsible for communication to different index technologies, please refer to it while reading below.

1.3.3 Registering to an EGIIS

ARC index EGIIS is a custom LDAP server using the Globus-MDS based registration schema. Registration of a cluster to EGIIS is achieved by sending simple *ldapadd* requests to an index where the cluster is allowed.

Registration is actually performed by a set of binaries run from time to time. The time interval can be defined by the system administrator. These scripts are independent from *arched*, and are configured and run by the *grid-infosys* startup script.

EGIIS was designed to be a lightweight index, so the registration information is the minimal amount needed to reach the local resource level.

1.4 Description of topics

Infoproviders The infoproviders scripts are a set of Perl scripts responsible for the following:

- Gathering information on systems, jobs, users, LRMSes
- Create LDIF and XML* renderings following the supported schemas NorduGrid, Glue1.2/1.3, GLUE2.

LDAP subsystem The LDAP subsystem is responsible for publication via slapd of the LDAP output of the infoproviders. Its components are:

OpenLDAP the third party component `slapd` LDAP server.

BDII, and bdii-update the third party BDII tools and the bdii script responsible to keep LDAP information fresh

grid-infosys bash startup script configures and start all of the LDAP subsystem

ARC BDII provider scripts created by grid-infosys and by the Infoproviders, they are the input to bdii-update to update the slapd LDAP database.

registration tools set of tools used to register to an EGIIS index.

index server the server process run to create a EGIIS index server.

Web Services subsystem This subsystem is responsible to:

- serve XML GLUE2 content generated by Infoproviders via the ARC WSRF, BES, XBES and EMIES information interfaces.
- register the GLUE2 services to ISIS or EMIR

GLUE2 rendering reference A description of ARC's GLUE2 rendering and attribute values.

*XML is rendered only for the GLUE2 schema

Chapter 2

Infoproviders

Infoproviders are a collection of PERL scripts that

- Gather information on systems, jobs, users, LRMSes
- Create LDIF and XML* renderings following the supported schemas NorduGrid, glue1.2/1.3, GLUE2.

ARC currenlty support two independent collections of infoproviders. In this document they will be referred as **old infoproviders** and **new infoproviders**.

This manual will mostly cover the **New Infoproviders**. For the sake of completeness, a description of Old Infoproviders is given at the end of this chapter.

2.1 New Infoproviders

The new infoproviders are the following:

- **Main scripts**

  CEinfo.pl Coordinates collection information for a CE and generates relevant output in all the schema and renderings. It generates XML as default output, and can generate a LDIF provider script `ldif-provider.sh` if a proper configuration option is set.

  se.pl Gathers information on a single Storage Element and outputs it only as a NorduGrid LDIF file.

- **Configuration utilities**

  ConfigCentral.pm Builds an intermediate config hash that is used by the A-REX infoprovider and LRMS control scripts, can read arc.conf, XML and INI style files.

  ConfigParser.pm Configuration parser used by se.pl (comes from Old Infoproviders) only reads arc.conf style files.

  IniParser.pm Configuration parser library for arc.conf / INI style files.

- **Information collectors**

  LRMSInfo.pm This perl module is responsible of defining a skeleton for all the LRMS information collectors for ARC > 0.6. A developer willing to implement a new LRMS infoprovider MUST adhere to the model defined in this module. This module also take care of handling LRMS modules designed for ARC ≤ 0.6 to be used with ARC > 0.6.

  LRMS.pm This perl module is responsible of defining a skeleton for all the LRMS information collectors for ARC ≤ 0.6. A developer willing to implement a new LRMS infoprovider MUST adhere to the model defined in this module. The existing modules implemented are shown in table 2.1

*XML is rendered only for the GLUE2 schema

Module	Supported LRMS
Condor.pm	Condor HTC [9]
DGBridge.pm	Special information provider for Desktop Grids[**?**]
Fork.pm	The special ARC Fork backend[**?**]
FORKmod.pm	The special ARC Fork backend. This module is for ARC $>$ 0.6[**?**]
GridFactory.pm	GridFactory Distributed Computing[**?**]†
LSF.pm	Platform Load Sharing Facility[12]
LL.pm	IBM Tivoli Workload Scheduler LoadLeveler[6]
PBS.pm	TORQUE Resource Manager[2]; PBS Professional[**?**]; openPBS[5]
SGE.pm	Sun Grid Engine[1]; Open Grid Scheduler/Grid Engine[**?**]; Oracle Grid Engine[**?**]
SGEmod.pm	Sun Grid Engine[1]; Open Grid Scheduler/Grid Engine[**?**]; Oracle Grid Engine[**?**]. This module is for ARC $>$ 0.6
SLURM.pm	Simple Linux Utility Resource Management[11]

Table 2.1: Supported LRMS information collectors. Unless specified otherwise, all the modules are designed for ARC $\leq$ 0.6. Modules designed for ARC $>$ 0.6 have the suffix `mod`.

GMJobsInfo.pm this module scans the control directory and retrieves information about A-REX jobs.

HostInfo.pm this module retrieves information about the Host machine where A-REX is running.

Sysinfo.pm this module is a collection of methods for fetching information about the Operating System of the machine where A-REX is running.

RTEInfo.pm this module is responsible for fetching information about RunTime Environments. See [] for details.

- **Utility modules**

Shared.pm this module contains helper functions and variables common to all the infoproviders.

LogUtils.pm this module contains helper functions and variables common to all the infoproviders for logging.

ARC0mod.pm helper module that loads ARC $\leq$ 0.6 modules to be used with ARC $>$ 0.6.

InfoChecker.pm Performs checks on configuration data structures

InfosysHelper.pm Helper functions and semaphores between Infoproviders and the LDAP subsystem.

- **Rendering modules**

ARC0ClusterInfo.pm module that builds data structures needed by NorduGrid schema rendering.

ARC1ClusterInfo.pm module that builds data structures needed by GLUE2 schema rendering.

LdifPrinter.pm library module with all the needed to print LDIF documents.

XMLPrinter.pm library module with all the needed to print XML documents.

NGldifPrinter.pm module to print information according to the NorduGrid schema, as a LDIF document.

GLUE2ldifPrinter.pm module to print information according to the GLUE2 schema, as a LDIF document.

GLUE2xmlPrinter.pm module to print information according to the GLUE2 schema, as an XML document.

†ARC support for this backend is planned to be phased out.

2.1.1 Dependencies

Figure X?? shows the dependencies between modules as a UML class diagram.

2.1.2 Execution flow

A-REX information collection is performed by CEinfo.pl. This script parses the configuration files and calls all the needed modules to generate information.

A CEinfo.pl run is initiated by A-REX on a defined interval. A-REX might eventually kill the CEinfo.pl script if it is unresponsive for long time. Unresponsiveness might happen because some module is taking too long to process. Common use cases are LRMS modules running cluster commands that are hanging, or jobs scanning very slow on overloaded systems.

Figure 2.1 shows the typical workflow of a CEinfo.pl run and all the operations involved.

Storage element information collection is performed by se.pl. A run is started by the LDAP subsystem on a defined interval.

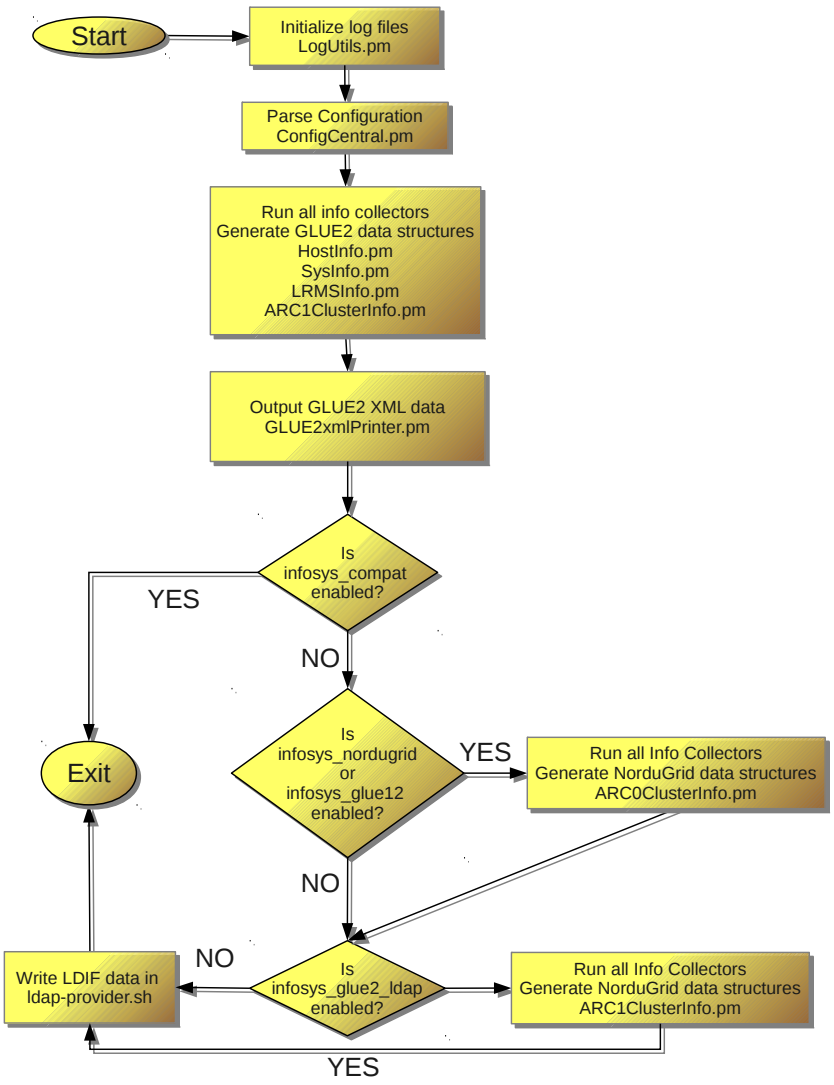

Figure 2.1: CEinfo.pl loop. The loop is initiated by a-rex...

2.2 Old Infoproviders

The old infoproviders are the following:

cluster.pl Coordinates collection information for a CE.

qju.pl Gathers information for a single queue.

se.pl Gather information on a single Storage Element.

LRMS.pm This perl module is responsible of defining a skeleton for all the LRMS information collectors. A developer willing to implement a new LRMS infoprovider MUST adhere to the model defined in this module. The existing modules implemented are shown in table **2.1**, *Supported LRMS information collectors. Unless specified otherwise, all the modules are designed for ARC* ≤ 0.6. *Modules designed for ARC* > 0.6 *have the suffix* `mod`

Shared.pm this module contains helper functions and variables common to all the infoproviders.

LogUtils.pm this module contains helper functions and variables common to all the infoproviders for logging.

Chapter 3

LDAP Subsystem

The LDAP Subsystem is responsible for publishing information via an LDAP server. The information ARC serves is compliant to three models: NorduGrid, Glue 1.2/1.3, GLUE2. Each of these models have been realized as a LDAP schema, and for each schema there are rules on how to build the DIT.

ARC LDAP subsystem components are described in detail in this chapter. In short, ARC features:

grid-infosys startup script A bash startup script that automatically configures and starts the ldap subsystem.

OpenLDAP A third party LDAP server.

bdii-update A perl script that periodically updates the LDAP databases.

registration binaries A set of binaries used to register to EGIIS.

EGIIS ARC's unique index server.

3.1 Overview: loop of the LDAP subsystem

A description of the operation of the LDAP subsystem is described in the flow-chart in figure **??**.

3.2 grid-infosys bash startup script

configures and start all of the LDAP subsystem

3.3 OpenLDAP

A third party component, the `slapd` LDAP server.

In this section we will highlight the use ARC does of the `slapd` server. Detailed information about this software can be found in []. The reader is expected to be confident with basic concepts of `slapd` operation.

Using the startup script described in 3.2, ARC creates all the configuration files needed for its instance of `slapd` to run. This instance of `slapd` is independent from any other configured instance on the same machine. Default port is usually 2135, but can be changed via configuration files.

The configuration files are deployed by default in the directory **/var/run/arc/bdii/**. This directory contains the slapd configuration file bdii-slapd.conf, `slapd` pid file, LDAP Berkeley Databases.

The slapd configuration file `bdii-slapd.conf` contains references to LDAP schemas and sets the root elements of the LDAP DIT.

This path can be redefined via configuration files.

3.4 BDII, and the bdii-update script

Third party BDII tools and the bdii script are responsible to keep LDAP information fresh.

The BDII system works by repeatedly executing the python script `bdii-update` to update the `slapd`'s Berkeley Databases mentioned in **3.3**, *OpenLDAP*. The execution flow of bdii-update can be described as follows:

1. A set of scripts called *providers*, are executed to produce several LDIF documents (See **3.5**, *ARC BDII provider scripts* for detailed description).
2. The LDIF documents are parsed and, for each ldap record, a comparison with the previous update is made.
3. Depending on the differences, ldapadd, ldapdelete or ldapmodify operations are submitted locally to the `slapd` server for its databases to be updated.

The grid-infosys script **3.2**, *grid-infosys bash startup script* takes care of creating configuration files, provider files and start the script loop in an independent way from any other instance of the same script. In this way, the same machine can host ARC and gLite**??** at the same time, as gLite information system also relies on bdii-update.

BDII-related files are stored by default in the following directories:

/var/tmp/arc/bdii Contains the bdii-infosys providers that: generate the DIT tree root objects; generate GLUE1 site information; perform synchronization between ARC Infoproviders and bdii-update.

/var/run/arc/infosys Contains the `bdii-update` configuration files and the ARC-specific providers that generate NorduGrid, GLUE1, GLUE2 renderings. It also contains a fifo used to syncronize ARC infoproviders with bdii providers.

3.5 ARC BDII provider scripts

Created by grid-infosys and by the Infoproviders, they are the input to `bdii-update` to update the `slapd` LDAP database.

A description of the scripts created by grid-infosys and their purposes is as follows:

/var/tmp/arc/bdii/arc-default.ldif.pl This perl script outputs the root elements LDIF file of the ARC LDAP DIT.

/var/tmp/arc/bdii/provider/arc-nordugrid-bdii-ldif This perl script creates the fifo (`/var/run/arc/infosys/ldif-provider.fifo`) between A-REX infoproviders and bdii-update to synchronize, and when A-REX Infoproviders have created `ldif-provider.sh` (see **2.1**, *New Infoproviders*), it executes `ldif-provider.sh` and `arc-glue-bdii-ldif` (see below)

/var/tmp/arc/bdii/provider/site_<someSiteName>.sh This perl script runs a gLite modified perl script (glite-info-provider-ldap) to mimic the behaviour of a glue1.2/1.3 site-bdii. However, this is higly experimental and planned to be phased out.

/var/tmp/arc/bdii/<someSiteName>.ldif This is not a script but a ldif file that is used by `site\<someSiteName>.sh`.

/var/run/arc/infosys/arc-glue-bdii-ldif This perl script is a translator between the NorduGrid schema and the Glue1.2/1.3 schema. When run, it outputs the ldif file for the latter.

3.6 registration tools

set of tools used to register to an EGIIS index.

3.7 EGIIS, the index server

the server process run to create a EGIIS index server.

Chapter 4

Webservices Subsystem

4.1 A-REX and the XML output

4.2 Registration via Web Services

Chapter 5

GLUE2 ARC Reference Documentation

This chapter describes in detail the structure of ARC GLUE2 rendering and the values

5.1 XML rendering

5.2 LDIF rendering

5.3 Attribute values

5.4 Mapping between job description languages and GLUE2 attributes

Acknowledgements

This work was supported in parts by: the Nordunet 2 program, the Nordic DataGrid Facility, the EU KnowARC project (Contract nr. 032691), the EU EMI project (Grant agreement nr. 261611) and the Swedish Research council via the eSSENCE strategic research program.

Bibliography

[1] Sun Grid Engine. Web site. URL `http://gridengine.sunsource.net`.

[2] Torque. Web site. URL `http://www.supercluster.org/projects/torque`.

[3] Berkeley Database Information Index V5. Web site. URL `https://twiki.cern.ch/twiki/bin/view/EGEE/BDII`.

[4] The NorduGrid Collaboration. Web site. URL `http://www.nordugrid.org`.

[5] OpenPBS. Web site. URL `http://www.openpbs.org`.

[6] LoadLeveler: Users Guide. Doc. No. SH26-7226-00IBM , IBM Corporation, 1993.

[7] M. Ellert, M. Grønager, A. Konstantinov, et al. Advanced Resource Connector middleware for lightweight computational Grids. *Future Gener. Comput. Syst.*, 23(1):219–240, 2007. ISSN 0167-739X. doi: 10.1016/j.future.2006.05.008.

[8] B. Kónya. *The NorduGrid/ARC Information System.* The NorduGrid Collaboration. URL `http://www.nordugrid.org/documents/arc_infosys.pdf`. NORDUGRID-TECH-4.

[9] M. Litzkow, M. Livny, and M. Mutka. Condor - A Hunter of Idle Workstations. In *Proc. of the 8th International Conference of Distributed Computing Systems, pp. 104–111*, 1998.

[10] M. Smith and T. A. Howes. *LDAP : Programming Directory-Enabled Applications with Lightweigt Directory Access Protocol.* Macmillan, 1997.

[11] A. Yoo, M. Jette, and M. Grondona. SLURM: Simple Linux Utility for Resource Management. 2862: 44–60, 2003. doi: DOI:10.1007/10968987_3.

[12] S. Zhou, J. Wang, X. Zheng, and P. Delisle. Utopia: A load sharing facility for large, heterogeneous distributed computing systems. Technical Report CSRI-257, Computer Systems Research Institute, University of Toronto, 1992.