Hosting Environment (Daemon) Services

Generated by Doxygen 1.6.1

Thu Sep 3 21:59:24 2009

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 2

Data Structure Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Data Structure Index

3.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 DREService Namespace Reference

Data Structures

- class **DREWebService**
- class **PerlProcessor**
- class **Task**
- class **TaskQueue**
- class **TaskSet**

5.1.1 Detailed Description

Implementation of a simple echo service

The reply of the echo service contains the string which was send to it.

Chapter 6

Data Structure Documentation

6.1 ARex2::ARex2Service Class Reference

The documentation for this class was generated from the following file:

- arex2.h

6.2 Arc::AREXClient Class Reference

A client class for the A-REX service.

`#include <arex_client.h>`

Public Member Functions

- **AREXClient** (std::string configFile="") throw (AREXClientError)
- ~**AREXClient** ()
- std::string **submit** (std::istream &jsdl_file, AREXFileList &file_list, bool delegate=false) throw (AREXClientError)
- std::string **stat** (const std::string &jobid) throw (AREXClientError)
- void **kill** (const std::string &jobid) throw (AREXClientError)
- void **clean** (const std::string &jobid) throw (AREXClientError)
- std::string **sstat** (void) throw (AREXClientError)

6.2.1 Detailed Description

A client class for the A-REX service. This class is a client for the A-REX service (Arc Resource-coupled EXecution service). It provides methods for three operations on an A-REX service:

- **Job** (p. 58) submission
- **Job** (p. 58) status queries
- **Job** (p. 58) termination

6.2.2 Constructor & Destructor Documentation

6.2.2.1 Arc::AREXClient::AREXClient (std::string *configFile* = "") throw (AREXClientError)

The constructor for the **AREXClient** (p. 18) class. This is the constructor for the **AREXClient** (p. 18) class. It creates an A-REX client that corresponds to a specific A-REX service, which is specified in a configuration file. The configuration file also specifies how to set up the communication chain for the client. The location of the configuration file can be provided as a parameter to the method. If no such parameter is given, the environment variable ARC_AREX_CONFIG is assumed to contain the location. If there is no such environment variable, the configuration file is assumed to be "arex_client.xml" in the current working directory.

Parameters:

configFile The location of the configuration file.

Exceptions:

An **AREXClientError** (p. 21) object if an error occurs.

6.2.2.2 Arc::AREXClient::~AREXClient ()

The destructor. This is the destructor. It does what destructors usually do, cleans up...

6.2.3 Member Function Documentation

6.2.3.1 void Arc::AREXClient::clean (const std::string & *jobid*) throw (AREXClientError)

Removes a job. This method sends a request to the A-REX service to remove a job from it's pool. If job is running it will be killed by service as well.

Parameters:

jobid The **Job** (p. 58) ID of the job to remove.

Exceptions:

An **AREXClientError** (p. 21) object if an error occurs.

6.2.3.2 void Arc::AREXClient::kill (const std::string & *jobid*) throw (AREXClientError)

Terminates a job. This method sends a request to the A-REX service to terminate a job.

Parameters:

jobid The **Job** (p. 58) ID of the job to terminate.

Exceptions:

An **AREXClientError** (p. 21) object if an error occurs.

6.2.3.3 std::string Arc::AREXClient::sstat (void) throw (AREXClientError)

Query the status of a service. This method queries the A-REX service about it's status.

Returns:

The XML document representing status of the service.

Exceptions:

An **AREXClientError** (p. 21) object if an error occurs.

6.2.3.4 std::string Arc::AREXClient::stat (const std::string & *jobid*) throw (AREXClientError)

Query the status of a job. This method queries the A-REX service about the status of a job.

Parameters:

jobid The **Job** (p. 58) ID of the job.

Returns:

The status of the job.

Exceptions:

An **AREXClientError** (p. 21) object if an error occurs.

6.2.3.5 std::string Arc::AREXClient::submit (std::istream & *jsdl_file*, AREXFileList & *file_list*, bool *delegate* = `false`) throw (AREXClientError)

Submit a job. This method submits a job to the A-REX service corresponding to this client instance.

Parameters:

jsdl_file An input stream from which the JSDL file for the job can be read.

Returns:

The **Job** (p. 58) ID of the the submitted job.

Exceptions:

An **AREXClientError** (p. 21) object if an error occurs.

The documentation for this class was generated from the following file:

- arex_client.h

6.3 Arc::AREXClientError Class Reference

An exception class for the **AREXClient** (p. 18) class.

`#include <arex_client.h>`

Public Member Functions

- **AREXClientError** (const std::string &what="")

6.3.1 Detailed Description

An exception class for the **AREXClient** (p. 18) class. This is an exception class that is used to handle runtime errors discovered in the **AREXClient** (p. 18) class.

6.3.2 Constructor & Destructor Documentation

6.3.2.1 Arc::AREXClientError::AREXClientError (const std::string & *what* = "")

Constructor. This is the constructor of the **AREXClientError** (p. 21) class.

Parameters:

what An explanation of the error.

The documentation for this class was generated from the following file:

- arex_client.h

6.4 Arc::AREXFile Class Reference

The documentation for this class was generated from the following file:

- arex_client.h

6.5 ARex::ARexGMConfig Class Reference

The documentation for this class was generated from the following file:

- a-rex/job.h

6.6 ARex::ARexJob Class Reference

`#include <job.h>`

Public Member Functions

- **ARexJob** (const std::string &id, **ARexGMConfig** &config, Arc::Logger &logger, bool fast_auth_check=false)
- **ARexJob** (Arc::XMLNode jsdl, **ARexGMConfig** &config, const std::string &credentials, const std::string &clientid, Arc::Logger &logger, const Arc::XMLNode &migration=Arc::XMLNode())
- std::string **Failure** (void)
- std::string **ID** (void)
- bool **GetDescription** (Arc::XMLNode &jsdl)
- bool **Cancel** (void)
- bool **Clean** (void)
- bool **Resume** (void)
- std::string **State** (void)
- std::string **State** (bool &job_pending)
- bool **Failed** (void)
- std::string **SessionDir** (void)
- std::string **LogDir** (void)
- int **CreateFile** (const std::string &filename)
- int **OpenFile** (const std::string &filename, bool for_read, bool for_write)
- int **OpenLogFile** (const std::string &name)
- Glib::Dir * **OpenDir** (const std::string &dirname)
- std::list< std::string > **LogFiles** (void)
- bool **UpdateCredentials** (const std::string &credentials)

Static Public Member Functions

- static int **TotalJobs** (**ARexGMConfig** &config, Arc::Logger &logger)
- static std::list< std::string > **Jobs** (**ARexGMConfig** &config, Arc::Logger &logger)

6.6.1 Detailed Description

This class represents convenience interface to manage jobs handled by Grid Manager. It works mostly through corresponding classes and functions of Grid Manager.

6.6.2 Constructor & Destructor Documentation

6.6.2.1 ARex::ARexJob::ARexJob (const std::string & *id*, ARexGMConfig & *config*, Arc::Logger & *logger*, bool *fast_auth_check* = `false`)

Create instance which is an interface to existing job

6.6.2.2 ARex::ARexJob::ARexJob (Arc::XMLNode *jsdl*, ARexGMConfig & *config*, const std::string & *credentials*, const std::string & *clientid*, Arc::Logger & *logger*, const Arc::XMLNode & *migration* = `Arc::XMLNode()`)

Create new job with provided JSDL description

6.6.3 Member Function Documentation

6.6.3.1 bool ARex::ARexJob::Cancel (void)

Cancel processing/execution of job

6.6.3.2 bool ARex::ARexJob::Clean (void)

Remove job from local pool

6.6.3.3 int ARex::ARexJob::CreateFile (const std::string & *filename*)

Creates file in job's session directory and returns handler

6.6.3.4 bool ARex::ARexJob::Failed (void)

Returns true if job has failed

6.6.3.5 std::string ARex::ARexJob::Failure (void) `[inline]`

Returns textual description of failure of last operation

6.6.3.6 bool ARex::ARexJob::GetDescription (Arc::XMLNode & *jsdl*)

Fills provided jsdl with job description

6.6.3.7 std::string ARex::ARexJob::ID (void) `[inline]`

Return ID assigned to job

6.6.3.8 static std::list<std::string> ARex::ARexJob::Jobs (ARexGMConfig & *config*, Arc::Logger & *logger*) `[static]`

Returns list of user's jobs. Fine-grained ACL is ignored.

6.6.3.9 std::string ARex::ARexJob::LogDir (void)

Returns name of virtual log directory

6.6.3.10 std::list<std::string> ARex::ARexJob::LogFiles (void)

Returns list of existing log files

6.6.3.11 Glib::Dir* ARex::ARexJob::OpenDir (const std::string & *dirname*)

Opens directory inside session directory

6.6.3.12 int ARex::ARexJob::OpenFile (const std::string & *filename*, bool *for_read*, bool *for_write*)

Opens file in job's session directory and returns handler

6.6.3.13 int ARex::ARexJob::OpenLogFile (const std::string & *name*)

Opens log file in control directory

6.6.3.14 bool ARex::ARexJob::Resume (void)

Resume execution of job after error

6.6.3.15 std::string ARex::ARexJob::SessionDir (void)

Returns path to session directory

6.6.3.16 std::string ARex::ARexJob::State (bool & *job_pending*)

Returns current state of job and sets job_pending to true if job is pending due to external limits

6.6.3.17 std::string ARex::ARexJob::State (void)

Returns current state of job

6.6.3.18 static int ARex::ARexJob::TotalJobs (ARexGMConfig & *config*, Arc::Logger & *logger*) [static]

Return number of jobs associated with this configuration. TODO: total for all user configurations.

6.6.3.19 bool ARex::ARexJob::UpdateCredentials (const std::string & *credentials*)

Updates job credentials

The documentation for this class was generated from the following file:

- a-rex/job.h

6.7 ARex::ARexService Class Reference

The documentation for this class was generated from the following file:

- arex.h

6.8 CacheConfig Class Reference

`#include <conf_cache.h>`

Public Member Functions

- **CacheConfig** (std::string username="")
- void **setCacheDirs** (std::list< std::string > cache_dirs)

6.8.1 Detailed Description

Reads conf file and provides methods to obtain cache info from it.

6.8.2 Constructor & Destructor Documentation

6.8.2.1 CacheConfig::CacheConfig (std::string *username* = "")

Create a new **CacheConfig** (p. 28) instance. Read the config file and fill in private member variables with cache parameters. If different users are defined in the conf file, use the cache parameters for the given username.

6.8.3 Member Function Documentation

6.8.3.1 void CacheConfig::setCacheDirs (std::list< std::string > *cache_dirs*) [inline]

To allow for substitutions done during configuration

The documentation for this class was generated from the following file:

- conf_cache.h

6.9 CacheConfigException Class Reference

`#include <conf_cache.h>`

6.9.1 Detailed Description

Exception thrown by constructor caused by bad cache params in conf file

The documentation for this class was generated from the following file:

- conf_cache.h

6.10 ArcSec::Charon Class Reference

`#include <charon.h>`

6.10.1 Detailed Description

A Service which includes the ArcPDP functionality; it can be deployed as an independent service to provide request evaluation functionality for the other remote services

The documentation for this class was generated from the following file:

- charon.h

6.11 CommFIFO Class Reference

Data Structures

- class **elem_t**

The documentation for this class was generated from the following file:

- commfifo.h

6.12 ARex::ConfGrp Class Reference

The documentation for this class was generated from the following file:

- **configcore.h**

6.13 ARex::Config Class Reference

`#include <configcore.h>`

Public Member Functions

- const std::list< **ConfGrp** > & **GetConfigs** () const
- std::list< std::string > **ConfValue** (const std::string &path) const
- std::string **FirstConfValue** (const std::string &path) const

6.13.1 Detailed Description

Core configuration class.

6.13.2 Member Function Documentation

6.13.2.1 std::list<std::string> ARex::Config::ConfValue (const std::string & *path*) const

Get the configuration values from key.

6.13.2.2 std::string ARex::Config::FirstConfValue (const std::string & *path*) const

Get the first configuration value from key. This is meant as a short cut when it is known that the key is not multivalued.

6.13.2.3 const std::list<ConfGrp>& ARex::Config::GetConfigs () const

Returns the parsed options.

The documentation for this class was generated from the following file:

- **configcore.h**

6.14 ARex::ConfigError Class Reference

`#include <configcore.h>`

Public Member Functions

- **ConfigError** (std::string message)

6.14.1 Detailed Description

Error configuration class.

6.14.2 Constructor & Destructor Documentation

6.14.2.1 ARex::ConfigError::ConfigError (std::string *message*) [inline]

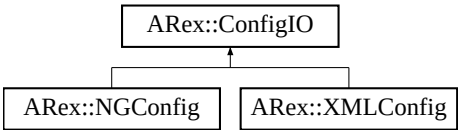
Constructor for the **ConfigError** (p. 34) exception. Calls the corresponding constructor in ARCLibError.

The documentation for this class was generated from the following file:

- **configcore.h**

6.15 ARex::ConfigIO Class Reference

`#include <configio.h>`Inheritance diagram for ARex::ConfigIO::



Public Member Functions

- virtual **Config Read** (std::istream &is)=0
- virtual void **Write** (const **Config** &conf, std::ostream &os)=0

6.15.1 Detailed Description

Virtual base-class for reading and writing configuration files. Concrete instances include **NGConfig** (p. 88) and **XMLConfig** (p. 120).

6.15.2 Member Function Documentation

6.15.2.1 virtual Config ARex::ConfigIO::Read (std::istream & *is*) [pure virtual]

Read the named configuration source.

Implemented in **ARex::NGConfig** (p. 88), and **ARex::XMLConfig** (p. 120).

6.15.2.2 virtual void ARex::ConfigIO::Write (const Config & *conf*, std::ostream & *os*) [pure virtual]

Write configuration to named configuration destination.

Implemented in **ARex::NGConfig** (p. 88), and **ARex::XMLConfig** (p. 120).

The documentation for this class was generated from the following file:

- configio.h

6.16 ConfigSections Class Reference

The documentation for this class was generated from the following file:

- conf_sections.h

6.17 Paul::Configurator Class Reference

The documentation for this class was generated from the following file:

- configurator.h

6.18 ContinuationPlugins Class Reference

Data Structures

- struct **command_t**

The documentation for this class was generated from the following file:

- plugins.h

6.19 Daemon Class Reference

The documentation for this class was generated from the following file:

- daemon.h

6.20 DREService::DREWebService Class Reference

Public Member Functions

- **DREWebService** (Arc::Config ∗cfg)
- virtual ∼**DREWebService** (void)
- virtual Arc::MCC_Status **process** (Arc::Message &inmsg, Arc::Message &outmsg)

Protected Member Functions

- Arc::MCC_Status **makeFault** (Arc::Message &outmsg, const std::string &reason)

Protected Attributes

- Arc::NS **ns_**

Static Protected Attributes

- static Arc::Logger **logger**

6.20.1 Constructor & Destructor Documentation

6.20.1.1 DREService::DREWebService::DREWebService (Arc::Config ∗ *cfg*)

Constructor which is capable to extract prefix and suffix for the echo service.

6.20.1.2 virtual DREService::DREWebService::∼DREWebService (void) [virtual]

Destructor.

6.20.2 Member Function Documentation

6.20.2.1 Arc::MCC_Status DREService::DREWebService::makeFault (Arc::Message & *outmsg*, const std::string & *reason*) [protected]

Method to return an error. Creates a fault message and returns a status.

Parameters:

outmsg outgoing message

Returns:

Status of the result achieved

6.20.2.2 virtual Arc::MCC_Status DREService::DREWebService::process (Arc::Message & *inmsg*, Arc::Message & *outmsg*) [virtual]

Implementation of the virtual method defined in MCCInterface (to be found in MCC.h).

Parameters:

inmsg incoming message

inmsg outgoing message

Returns:

Status of the result achieved

6.20.3 Field Documentation

6.20.3.1 Arc::Logger DREService::DREWebService::logger [static, protected]

Arc-intern logger. Generates output into the file specified in the arched configuration file used to invoke arched services.

6.20.3.2 Arc::NS DREService::DREWebService::ns_ [protected]

Class which specifies a XML namespace i.e. "echo". Needed to extract the content out of the incoming message

The documentation for this class was generated from the following file:

- dREWebService.h

6.21 FileData Class Reference

The documentation for this class was generated from the following file:

- info_types.h

6.22 fs_usage Struct Reference

The documentation for this struct was generated from the following file:

- fsusage.h

6.23 ARex::GridManager Class Reference

The documentation for this class was generated from the following file:

- grid_manager.h

6.24 GridScheduler::GridSchedulerService Class Reference

The documentation for this class was generated from the following file:

- grid_sched.h

6.25 Hopi::Hopi Class Reference

The documentation for this class was generated from the following file:

- hopi.h

6.26 Paul::HTMLRequest Class Reference

The documentation for this class was generated from the following file:

- configurator.h

6.27 Paul::HTMLResponse Class Reference

The documentation for this class was generated from the following file:

- configurator.h

6.28 ARex2::JobDescription::InputFile Class Reference

`#include <job_descr.h>`

6.28.1 Detailed Description

Class represents the one of the input file of the job

The documentation for this class was generated from the following file:

- job_descr.h

6.29 Paul::InvalidMessageException Class Reference

The documentation for this class was generated from the following file:

- configurator.h

6.30 ISIS::ISIService Class Reference

The documentation for this class was generated from the following file:

- isis.h

6.31 ISIS::ISISSecAttr Class Reference

The documentation for this class was generated from the following file:

- security.h

6.32 Janitor Class Reference

Class to communicate with **Janitor** (p. 53) - Dynmaic Runtime Environment handler.

`#include <janitor.h>`

Public Member Functions

- **Janitor** (const std::string &id, const std::string &cdir)
- **operator bool** (void)
- bool **operator!** (void)
- bool **deploy** (void)
- bool **remove** (void)
- bool **wait** (int timeout)
- bool **result** (void)

6.32.1 Detailed Description

Class to communicate with **Janitor** (p. 53) - Dynmaic Runtime Environment handler.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 Janitor::Janitor (const std::string & *id*, const std::string & *cdir*)

Creates instance representing job entry in **Janitor** (p. 53) database. Takes id for job identifier and cdir for the control directory of A-Rex. constructor does not register job in the **Janitor** (p. 53). It only associates job with this instance.

6.32.3 Member Function Documentation

6.32.3.1 bool Janitor::deploy (void)

Registers associated job with **Janitor** (p. 53) and deploys dynamic RTEs. This operation is asynchronous. Returned true means **Janitor** (p. 53) will be contacted and deployemnt will start soon. For obtaining result of operation see methods **wait()** (p. 54) and **result()** (p. 53). During this operation janitor utility is called with command register and optionally deploy.

6.32.3.2 bool Janitor::remove (void)

Removes job from those handled by **Janitor** (p. 53) and releases associated RTEs. This operation is asynchronous. Returned true means **Janitor** (p. 53) will be contacted and removal will start soon. For obtaining result of operation see methods **wait()** (p. 54) and **result()** (p. 53). During this operation janitor utility is called with command remove.

6.32.3.3 bool Janitor::result (void)

Returns true if operation initiated by **deploy()** (p. 53) or **remove()** (p. 53) succeeded. It should be called after **wait()** (p. 54) returned true.

6.32.3.4 bool Janitor::wait (int *timeout*)

Wait till operation initiated by **deploy()** (p. 53) or **remove()** (p. 53) finished. This operation returns true if operation finished or false if timeout seconds passed. It may be called repeatedly and even after it previously returned true. If no operation is running it returns true immeaditely.

The documentation for this class was generated from the following file:

- janitor.h

6.33 Paul::Job Class Reference

The documentation for this class was generated from the following file:

- paul/job.h

6.34 Job Class Reference

`#include <job.h>`

Public Member Functions

- **Job** (void)
- **Job** (std::string path)
- ~**Job** (void)
- **operator bool** (void)
- bool **Start** (void)
- bool **Cancel** (void)
- bool **Resume** (void)
- std::string **GetState** (void)
- std::string **GetSessionDir** (void)

6.34.1 Detailed Description

Collect all information (status, lrms info, user) required to handle job

6.34.2 Constructor & Destructor Documentation

6.34.2.1 Job::Job (void)

Constructor: Creates empty job

6.34.2.2 Job::Job (std::string *path*)

Constructor: load job information form files

6.34.2.3 Job::~Job (void)

Destuctior

6.34.3 Member Function Documentation

6.34.3.1 bool Job::Cancel (void)

Cancel processing/execution of job

6.34.3.2 std::string Job::GetSessionDir (void)

Returns the session directory of the job

6.34.3.3 std::string Job::GetState (void)

Returns the string represnetation of job state

6.34.3.4 Job::operator bool (void) [inline]

Helper logical operators

6.34.3.5 bool Job::Resume (void)

Resume execution of job after error

6.34.3.6 bool Job::Start (void)

Starts job. Most of the cases it means to submit to LRMS

The documentation for this class was generated from the following file:

- arex2/job.h

6.35 Arc::Job Class Reference

The documentation for this class was generated from the following file:

- sched/job.h

6.36 job_state_rec_t Struct Reference

The documentation for this struct was generated from the following file:

- a-rex/grid-manager/jobs/job.h

6.37 ARex2::JobControl Class Reference

`#include <job_control.h>`

6.37.1 Detailed Description

Represents job controll information like session dir, control dir

The documentation for this class was generated from the following file:

- job_control.h

6.38 ARex2::JobController Class Reference

The documentation for this class was generated from the following file:

- job_control.h

6.39 ARex2::JobDataCache Class Reference

`#include <job_data_cache.h>`

6.39.1 Detailed Description

Data cache

The documentation for this class was generated from the following file:

- job_data_cache.h

6.40 JobDescription Class Reference

Friends

- class **JobsList**

The documentation for this class was generated from the following file:

- a-rex/grid-manager/jobs/job.h

6.41 ARex2::JobDescription Class Reference

`#include <job_descr.h>`

Data Structures

- class **InputFile**
- class **Notification**
- class **OutputFile**

Public Member Functions

- std::string & **JobName** (void)

6.41.1 Detailed Description

Internal representation of **Job** (p. 56) described by JSDL

6.41.2 Member Function Documentation

6.41.2.1 std::string& ARex2::JobDescription::JobName (void) [inline]

Interface methods to access stored values.

The documentation for this class was generated from the following file:

- job_descr.h

6.42 Paul::JobList Class Reference

The documentation for this class was generated from the following file:

- job_list.h

6.43 JobLocalDescription Class Reference

The documentation for this class was generated from the following file:

- info_types.h

6.44 JobLog Class Reference

`#include <job_log.h>`

6.44.1 Detailed Description

Put short information into log when every job starts/finishes. And store more detailed information for Reporter.

The documentation for this class was generated from the following file:

- job_log.h

6.45 ARex2::JobLRMSInfo Class Reference

`#include <lrms.h>`

6.45.1 Detailed Description

Class represents the information about job in LRMS

The documentation for this class was generated from the following file:

- lrms.h

6.46 Arc::JobNotFoundException Class Reference

The documentation for this class was generated from the following file:

- sched/job_queue.h

6.47 Paul::JobQueue Class Reference

The documentation for this class was generated from the following file:

- paul/job_queue.h

6.48 Arc::JobQueue Class Reference

The documentation for this class was generated from the following file:

- sched/job_queue.h

6.49 Arc::JobQueueIterator Class Reference

The documentation for this class was generated from the following file:

- sched/job_queue.h

6.50 ARex::JobRecord Class Reference

The documentation for this class was generated from the following file:

- JobRecord.h

6.51 Paul::JobRequest Class Reference

The documentation for this class was generated from the following file:

- paul/job_request.h

6.52 Arc::JobRequest Class Reference

The documentation for this class was generated from the following file:

- sched/job_request.h

6.53 Arc::JobSchedMetaData Class Reference

The documentation for this class was generated from the following file:

- sched/job_sched_meta.h

6.54 Paul::JobSchedMetaData Class Reference

The documentation for this class was generated from the following file:

- paul/job_sched_meta.h

6.55 Arc::JobSelector Class Reference

The documentation for this class was generated from the following file:

- sched/job_queue.h

6.56 JobsList Class Reference

The documentation for this class was generated from the following file:

- states.h

6.57 ARex2::JobState Class Reference

`#include <job_state.h>`

6.57.1 Detailed Description

Represents the state of job. It includes error messages as well

The documentation for this class was generated from the following file:

- job_state.h

6.58 ARex2::JobUser Class Reference

`#include <job_user.h>`

6.58.1 Detailed Description

Job (p. 56) run under the privilages of one of the system user. This class collects information related to this user

The documentation for this class was generated from the following file:

- job_user.h

6.59 JobUser Class Reference

The documentation for this class was generated from the following file:

- users.h

6.60 JobUserHelper Class Reference

The documentation for this class was generated from the following file:

- users.h

6.61 JobUsers Class Reference

The documentation for this class was generated from the following file:

- users.h

6.62 RunPlugin::lib_plugin_t Union Reference

The documentation for this union was generated from the following file:

- run_plugin.h

6.63 ARex::LoggerClient Class Reference

The documentation for this class was generated from the following file:

- client.h

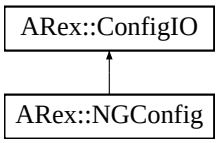
6.64 LRMSResult Class Reference

The documentation for this class was generated from the following file:

- info_types.h

6.65 ARex::NGConfig Class Reference

`#include <ngconfig.h>`Inheritance diagram for ARex::NGConfig::



Public Member Functions

- **Config Read** (std::istream &is)
- void **Write** (const **Config** &config, std::ostream &os)

6.65.1 Detailed Description

Configuration class used for reading configuration files ARC-style.

6.65.2 Member Function Documentation

6.65.2.1 Config ARex::NGConfig::Read (std::istream & *is*) `[virtual]`

Read old arc.conf style configuration.

Implements **ARex::ConfigIO** (p. 35).

6.65.2.2 void ARex::NGConfig::Write (const Config & *config*, std::ostream & *os*) `[virtual]`

Write configuration to named file.

Implements **ARex::ConfigIO** (p. 35).

The documentation for this class was generated from the following file:

- ngconfig.h

6.66 ARex2::JobDescription::Notification Class Reference

`#include <job_descr.h>`

6.66.1 Detailed Description

Class represents notification requiest

The documentation for this class was generated from the following file:

- job_descr.h

6.67 numvalue_for_shell Class Reference

The documentation for this class was generated from the following file:

- job_desc.h

6.68 ARex::Option Class Reference

The documentation for this class was generated from the following file:

- **configcore.h**

6.69 ARex2::JobDescription::OutputFile Class Reference

`#include <job_descr.h>`

6.69.1 Detailed Description

Class represents the one of the output file of the job

The documentation for this class was generated from the following file:

- job_descr.h

6.70 Paul::PaulService Class Reference

The documentation for this class was generated from the following file:

- paul.h

6.71 Hopi::PayloadBigFile Class Reference

Public Member Functions

- **PayloadBigFile** (const char ∗filename)
- **PayloadBigFile** (const char ∗filename, Size_t size)

6.71.1 Constructor & Destructor Documentation

6.71.1.1 Hopi::PayloadBigFile::PayloadBigFile (const char ∗ *filename*)

Creates object associated with file for reading from it

6.71.1.2 Hopi::PayloadBigFile::PayloadBigFile (const char ∗ *filename*, Size_t *size*)

Creates object associated with file for writing into it. Use size=-1 for undefined size.

The documentation for this class was generated from the following file:

- hopi/PayloadFile.h

6.72 ARex::PayloadFile Class Reference

`#include <PayloadFile.h>`

Public Member Functions

- **PayloadFile** (const char *filename, size_t start=0, size_t end=(size_t)(-1))
- **PayloadFile** (int handle, size_t start=0, size_t end=(size_t)(-1))
- **PayloadFile** (const char *filename, int size)

6.72.1 Detailed Description

Implementation of PayloadRawInterface which provides access to ordinary file. Currently only read-only mode is supported.

6.72.2 Constructor & Destructor Documentation

6.72.2.1 ARex::PayloadFile::PayloadFile (const char * *filename*, size_t *start* = `0`, size_t *end* = `(size_t)(-1)`)

Creates object associated with file for reading from it

6.72.2.2 ARex::PayloadFile::PayloadFile (int *handle*, size_t *start* = `0`, size_t *end* = `(size_t)(-1)`)

Creates object associated with file handle for reading or writing to it

6.72.2.3 ARex::PayloadFile::PayloadFile (const char * *filename*, int *size*)

Creates object associated with file for writing into it. Use size=-1 for undefined size.

The documentation for this class was generated from the following file:

- a-rex/PayloadFile.h

6.73 Hopi::PayloadFile Class Reference

`#include <PayloadFile.h>`

Public Member Functions

- **PayloadFile** (const char *filename)
- **PayloadFile** (const char *filename, Size_t size)

6.73.1 Detailed Description

Implementation of PayloadRawInterface which provides access to ordinary file. Currently only read-only mode is supported.

6.73.2 Constructor & Destructor Documentation

6.73.2.1 Hopi::PayloadFile::PayloadFile (const char * *filename*)

Creates object associated with file for reading from it

6.73.2.2 Hopi::PayloadFile::PayloadFile (const char * *filename*, Size_t *size*)

Creates object associated with file for writing into it. Use size=-1 for undefined size.

The documentation for this class was generated from the following file:

- hopi/PayloadFile.h

6.74 DREService::PerlProcessor Class Reference

Data Structures

- struct **ThreadInterface**

Public Member Functions

- **PerlProcessor** (int threadNumber, **TaskQueue** *pTaskQueue, **TaskSet** *pTaskSet)
- virtual ~**PerlProcessor** (void)

6.74.1 Constructor & Destructor Documentation

6.74.1.1 DREService::PerlProcessor::PerlProcessor (int *threadNumber*, TaskQueue * *pTaskQueue*, TaskSet * *pTaskSet*)

Constructor which is capable to extract prefix and suffix for the echo service.

6.74.1.2 virtual DREService::PerlProcessor::~PerlProcessor (void) [virtual]

Destructor.

The documentation for this class was generated from the following file:

- PerlProcessor.h

6.75 GridScheduler::Resource Class Reference

The documentation for this class was generated from the following file:

- resource.h

6.76 GridScheduler::ResourcesHandling Class Reference

The documentation for this class was generated from the following file:

- resources_handling.h

6.77 RTE Class Reference

`#include <rte.h>`

Public Member Functions

- **RTE** (const std::string &re)
- ~**RTE** ()
- std::string **str** () const
- std::string **Name** () const
- std::string **Version** () const
- bool **operator==** (const **RTE** &other) const
- bool **operator!=** (const **RTE** &other) const
- bool **operator**> (const **RTE** &other) const
- bool **operator**< (const **RTE** &other) const
- bool **operator**>**=** (const **RTE** &other) const
- bool **operator**<**=** (const **RTE** &other) const

6.77.1 Detailed Description

Interface for handling runtime environments. **RTE** (p. 100) class. It represents a runtime environment, and provides functionality for getting information about them.

6.77.2 Constructor & Destructor Documentation

6.77.2.1 RTE::RTE (const std::string & *re*)

Constructs a new runtime environemt. String should in general be of the type: STRING-VERSION. Where version consists of numbers with . between them.

6.77.2.2 RTE::~RTE ()

Destructor. Not that much to say.

6.77.3 Member Function Documentation

6.77.3.1 std::string RTE::Name () const

Returns the name of the runtime environment.

6.77.3.2 bool RTE::operator!= (const RTE & *other*) const

Inequility operator. Return the opsite of ==

6.77.3.3 bool RTE::operator< (const RTE & *other*) const

Less than operator. Returns false if the other is equal, otherwise it returns the opposite of >

6.77.3.4 bool RTE::operator<= (const RTE & *other*) const

Less than or equal operator. Returns the oppsite of >

6.77.3.5 bool RTE::operator== (const RTE & *other*) const

Equliaty operator. Returns true if the runtime environments have the string representation.

6.77.3.6 bool RTE::operator> (const RTE & *other*) const

Greater than operator. Returns true if the compared runtime environment is greater than the current.

6.77.3.7 bool RTE::operator>= (const RTE & *other*) const

Greater or equal operator. Returns the opposite of <

6.77.3.8 std::string RTE::str () const

Returns a string representation of the runtime environment. This is usually the same as given in the constructor.

6.77.3.9 std::string RTE::Version () const

Returns the version of the runtime environment.

The documentation for this class was generated from the following file:

- rte.h

6.78 RunFunction Class Reference

The documentation for this class was generated from the following file:

- run_function.h

6.79 RunParallel Class Reference

The documentation for this class was generated from the following file:

- run_parallel.h

6.80 RunPlugin Class Reference

Data Structures

- union **lib_plugin_t**

The documentation for this class was generated from the following file:

- run_plugin.h

6.81 RunPlugins Class Reference

The documentation for this class was generated from the following file:

- run_plugin.h

6.82 RunRedirected Class Reference

The documentation for this class was generated from the following file:

- run_redirected.h

6.83 ArcSec::Service_AA Class Reference

`#include <aaservice.h>`

6.83.1 Detailed Description

A Service which includes the AttributeAuthority functionality; it accepts the <samlp:AttributeQuery> which includes the <Subject> of the principal from the request and <Attribute> which the request would get; it access some local attribute database and returns <samlp:Assertion> which includes the <Attribute>

The documentation for this class was generated from the following file:

- aaservice.h

6.84 Compiler::Service_Compiler Class Reference

`#include <compiler.h>`

Public Member Functions

- **Service_Compiler** (Arc::Config *cfg)
- virtual Arc::MCC_Status **process** (Arc::Message &, Arc::Message &)

6.84.1 Detailed Description

This service need in the server config a "compiler:scriptfile_url" element. It is the scriptfile place, where-from it will be download the JSDL.

6.84.2 Constructor & Destructor Documentation

6.84.2.1 Compiler::Service_Compiler::Service_Compiler (Arc::Config * *cfg*)

Constructor accepts configuration describing content of scriptfile_url

6.84.3 Member Function Documentation

6.84.3.1 virtual Arc::MCC_Status Compiler::Service_Compiler::process (Arc::Message &, Arc::Message &) `[virtual]`

Service request processing routine

The documentation for this class was generated from the following file:

- compiler.h

6.85 ArcSec::Service_Delegation Class Reference

A Service which launches the proxy certificate request; it accepts the request from.

`#include <delegation.h>`

6.85.1 Detailed Description

A Service which launches the proxy certificate request; it accepts the request from.

The documentation for this class was generated from the following file:

- delegation.h

6.86 Arc::Service_JavaWrapper Class Reference

Public Member Functions

- virtual Arc::MCC_Status **process** (Arc::Message &, Arc::Message &)

6.86.1 Member Function Documentation

6.86.1.1 virtual Arc::MCC_Status Arc::Service_JavaWrapper::process (Arc::Message &, Arc::Message &) [virtual]

Service request processing routine

The documentation for this class was generated from the following file:

- javawrapper.h

6.87 Arc::Service_PythonWrapper Class Reference

Public Member Functions

- virtual Arc::MCC_Status **process** (Arc::Message &, Arc::Message &)

6.87.1 Member Function Documentation

6.87.1.1 virtual Arc::MCC_Status Arc::Service_PythonWrapper::process (Arc::Message &, Arc::Message &) [virtual]

Service request processing routine

The documentation for this class was generated from the following file:

- pythonwrapper.h

6.88 ArcSec::Service_SLCS Class Reference

`#include <slcs.h>`

6.88.1 Detailed Description

A Service which signs the short-lived certificate; it accepts the certificate signing request (CSR) from from client side through soap, signs a short-lived certificate and sends back through soap. This service is supposed to be deployed together with the SPService and saml2sso.serviceprovider handler, in order to sign certificate based on the authentication result from saml2sso profile. Also the saml attribute (inside the saml assertion from saml2sso profile) will be put into the signed short-lived certificate. By deploying this service together with SPService and saml2sso.serviceprovider handler, we can get the convertion from username/password ------> x509 certificate.

The documentation for this class was generated from the following file:

- slcs.h

6.89 SPService::Service_SP Class Reference

`#include <SPService.h>`

Public Member Functions

- **Service_SP** (Arc::Config *cfg)
- virtual Arc::MCC_Status **process** (Arc::Message &, Arc::Message &)

6.89.1 Detailed Description

This is service which accepts HTTP request from user agent (web browser) in the client side and processes the functionality of Service Provider in SAML2 SSO profile --- composing <AuthnRequest> Note: the IdP name is provided by the user agent directely when it gives a request, instead of the WRYF(where are you from) or Discovery Service in other implementation

6.89.2 Constructor & Destructor Documentation

6.89.2.1 SPService::Service_SP::Service_SP (Arc::Config * *cfg*)

Constructor

6.89.3 Member Function Documentation

6.89.3.1 virtual Arc::MCC_Status SPService::Service_SP::process (Arc::Message &, Arc::Message &) [virtual]

Service request processing routine

The documentation for this class was generated from the following file:

- SPService.h

6.90 Paul::SysInfo Class Reference

The documentation for this class was generated from the following file:

- sysinfo.h

6.91 DREService::Task Class Reference

Public Member Functions

- **Task** (int taskID, Arc::Message ∗request, Arc::Message ∗response)
- virtual ∼**Task** (void)

6.91.1 Constructor & Destructor Documentation

6.91.1.1 DREService::Task::Task (int *taskID*, Arc::Message ∗ *request*, Arc::Message ∗ *response*)

Constructor which is capable to extract prefix and suffix for the echo service.

6.91.1.2 virtual DREService::Task::∼Task (void) [virtual]

Destructor.

The documentation for this class was generated from the following file:

- Task.h

6.92 DREService::TaskQueue Class Reference

Public Member Functions

- **TaskQueue** (int length)
- virtual ∼**TaskQueue** (void)
- int **pushTask** (**Task** ∗task)
- **Task** ∗ **shiftTask** ()

6.92.1 Constructor & Destructor Documentation

6.92.1.1 DREService::TaskQueue::TaskQueue (int *length*)

Constructor which is capable to extract prefix and suffix for the echo service.

6.92.1.2 virtual DREService::TaskQueue::∼TaskQueue (void) `[virtual]`

Destructor.

6.92.2 Member Function Documentation

6.92.2.1 int DREService::TaskQueue::pushTask (Task ∗ *task*)

Blocks, if taskqueue is full. If task is stored in the queue and had a taskID == -1 it gets a fresh taskID.

6.92.2.2 Task∗ DREService::TaskQueue::shiftTask ()

Shifts the first task from the queue (and removes it).

The documentation for this class was generated from the following file:

- TaskQueue.h

6.93 DREService::TaskSet Class Reference

Public Member Functions

- **TaskSet** (int size)
- virtual ∼**TaskSet** (void)
- **Task** ∗ **removeTask** (int)

6.93.1 Constructor & Destructor Documentation

6.93.1.1 DREService::TaskSet::TaskSet (int *size*)

Constructor which is capable to extract prefix and suffix for the echo service.

6.93.1.2 virtual DREService::TaskSet::∼TaskSet (void) [virtual]

Destructor.

6.93.2 Member Function Documentation

6.93.2.1 Task∗ DREService::TaskSet::removeTask (int)

Checks wheter there is a task in the queue having that taskID in order to return it. If such a taskID is not available, themethod blocks until such a taskID is available. The task will be removed from the stack in that case.

The documentation for this class was generated from the following file:

- TaskSet.h

6.94 UrlMapConfig Class Reference

The documentation for this class was generated from the following file:

- conf_map.h

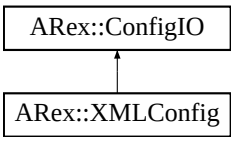
6.95 value_for_shell Class Reference

The documentation for this class was generated from the following file:

- job_desc.h

6.96 ARex::XMLConfig Class Reference

`#include <xmlconfig.h>`Inheritance diagram for ARex::XMLConfig::



Public Member Functions

- **Config Read** (std::istream &is)
- void **Write** (const **Config** &config, std::ostream &os)

6.96.1 Detailed Description

Class for reading in configuration files in xml-format. It uses libxml2 for xml-parsing.

6.96.2 Member Function Documentation

6.96.2.1 Config ARex::XMLConfig::Read (std::istream & *is*) `[virtual]`

Read configuration.

Implements **ARex::ConfigIO** (p. 35).

6.96.2.2 void ARex::XMLConfig::Write (const Config & *config*, std::ostream & *os*) `[virtual]`

Write configuration.

Implements **ARex::ConfigIO** (p. 35).

The documentation for this class was generated from the following file:

- xmlconfig.h

Chapter 7

File Documentation

7.1 configcore.h File Reference

```
#include <iostream>
#include <list>
#include <map>
#include <string>
#include <arc/Logger.h>
```

Data Structures

- class **ARex::ConfigError**
- class **ARex::Option**
- class **ARex::ConfGrp**
- class **ARex::Config**

Functions

- Config **ARex::ReadConfig** (std::istream &is)
- Config **ARex::ReadConfig** (const std::string &filename)

7.1.1 Detailed Description

This file describes the core configuration

Index