WMS APIs

Información para las prácticas

GRIDS & e-CIENCIA

Curso 2005 - IFIC Valencia - SPAIN

14 - 16 de junio de 2005

http://ific.uv.es/grid/cursogrid/

INTRODUCCION

- Las siguientes transparencias son obtenidas del WMS CERN Tutorial (Simone Campana)
- APIs útiles para las prácticas del WMS

WMS APIs

Enabling Grids for E-science in Europe

- The WMS makes C++ and Java APIs available for UI, LB consumer and client.
  - Python libraries also available
    - Created from C++ APIs using Swig
- In the following document:

  http://server11.infn.it/workload-grid/docs/DataGrid-01-TEN-0118-1_2.pdf

  details about the rpms containing the APIs are given.
- Correspondent doxygen documentation can be found in share/doc area. Ex.:
  - $EDG_LOCATION/share/doc/edg-wl-ui-api-cpp-lcg2.1.49/html
- Look also at:
  - http://grid-deployment.web.cern.ch/grid-deployment/eis/tutorial/edg-wl-ui-api/index.html
  - http://grid-deployment.web.cern.ch/grid-deployment/eis/tutorial/edg-wl-common-api/index.html

“JobAd” Class

Enabling Grids for E-science in Europe

Representation of the job description in the JDL language

String and Stream Constructor/Destructor

- void **fromStream** (istream &jdl_in)
  - *Update the JobAd object with the given input stream.*
- string **toString** ()
  - *Convert the JobAd Instance into a single line string representation.*
- string **toString** (const string &attr_name)
  - *Retrieve the string representation of an attribute.*
- string **toSubmissionString** ()
  - *Convert the JobAd Instance into a single line string representation ready for submission.*
- void **toFile** (const string &file_path)
  - *Put the JobAd Instance as a string into a file.*

Insertion Methods

- void **setAttribute** (const string &attr_name, const string &attr_value)
  - *Set an attribute key-value pair*
- void **addAttribute** (const string &attr_name,const, string &attr_value)
  - *Add an value to an attribute*
- void **setAttributeExpr** (const string &attr_name, const string &attr_value)
  - *Add The specified Expression Attribute to the jdl istance.*

“JobAd” Class

Enabling Grids for E-science in Europe

Retrieval Methods

- string **getString** (const string &attr_name)
  - *Retreive the value of the specified attribute.*
- int **getInt** (const string &attr_name)
  - *Retreive the value of the specified attribute.*
- double **getDouble** (const string &attr_name)
  - *Retreive the value of the specified attribute.*
- bool **getBool** (const string &attr_name)
  - *Retreive the value of the specified attribute.*

Miscellaneous Methods

- void **checkSyntax** (const string &attr_name, classad::ExprTree *attr_value)
  - *Check if the couple attribute/value is admitted.*
- void **checkMultiAttribute** (const vector< string > &multi)
  - *Check if the Member/isMember expression is properly used in rank and requirements attributes expressions.*
- classad::ExprTree * **delAttribute** (const string &attr_name)
  - *Delete an Attribute.*
- void **check** ()
  - *Check the JobAd instance for both syntax and semanthic errors.*

- Namespace: **edg::workload::common::requestad**

“JDL” class (attributes)

Enabling Grids for E-science in Europe

- **edg/workload/common/requestad/JDLAttributes.h**
- Namespace: **edg::workload::common::requestad**

```
class JDL {
public:
  static const string REQUIREMENTS;
  static const string FUZZY_RANK;
  static const string EXITCODE;
  static const string NODENUMB;
  static const string SHPORT;
  static const string RETRYCOUNT;
  static const string CE_MATCH;
  static const string CHKPT_STEPS;
  static const string CHKPT_CURRENTSTEP;
  static const string RANK;
  static const string NOTIFYTYPE;
  static const string JOBSTATUS;
[…]
}
```

“Job” Class

Enabling Grids for E-science in Europe

Namespace: **edg::workload::userinterface**

- JobId * **getJobId**()
  - *Get the JobId instance*
- JobAd * **getJobAd**()
  - *Get the JobAd instance.*
- void **setCredPath(const** string cp)
  - *Set a different Proxy certificate from the default one.*
- void **unsetCredPath**()
  - *Set the Proxy certificate as default.*
- void **setLoggerLevel(int** level)
  - *Set the verbosity level for NS debug default value*
- void **setJobAd(const** JobAd &ad)
  - *Set the JobAd instance.*
- void **setJobId(const** JobId &id)
  - *Set the JobId instance.*

“Job” Class

Enabling Grids for E-science in Europe

- Status **getStatus** (bool ad=true)
  - *Retrieve the status of the job.*
- Event **getLogInfo** ()
  - *Retrieve the bookkeeping information of the job.*
- void **submit** (const string &nSHost, int nsPort, const string &lbHost, int lbPort,const string &ce_id="")
  - *Submit the job to the Network Server.*
- vector< pair< string,double > >**listMatchingCE** (const string &host, int port)
  - *Look for matching Computing Element available resources.*
- ResultCode **cancel** ()
  - *Cancel the job from the Network Server.*
- void **getOutput** (const string &dir_path)
  - *Retrieve output files of a submitted job (Success Done status has to be reached).*

“JobId” Class

Enabling Grids for E-science in Europe

- void **fromString** (const string &dg_JobId)
  - *sets the JobId instance from the JobId in string format given as input.*
- string **toString** () const
  - *Converts the jobId into a string.*
- void **clear** ()
  - *Unsets the JobId instance.*
- bool **isSet** ()
  - *Check wheater the jobId has been already created (true) or not (false).*
- void **setJobId** (const string &lb_server, int port=0, const string &unique="")
  - *Set the JobId instance according to the LB and RB server addresses and the unique string passed as input parameters.*

- Namespace: **edg::workload::common::jobid**

“JobStatus” class

Enabling Grids for E-science in Europe

- Namespace: **edg::workload::logging::client**

Public Member Values:

- enum **Code** {UNDEF = 0, SUBMITTED, WAITING, READY, SCHEDULED, RUNNING, DONE, CLEARED, ABORTED, CANCELLED, UNKNOWN, PURGED, CODE_MAX };
- enum **Attr** {[…]}
  - *Job attributes, like worker node where the job is executed (CE_NODE), return code (DONE_CODE), type of job (JOBTYPE) etc …*
- enum **AttrType** { INT_T,STRING_T,TIMEVAL_T,BOOL_T,JOBID_T, …};
  - *Type of attributes*
- Code **status**;
  - *Numeric status code*

Public Methods:

- const string & **name** (void) const;
  - *String representation of the status code*
- const vector<pair<Attr,AttrType> >& **getAttrs** (void) const;
  - *List of attributes and types valid for this instance*
- const std::string& **getAttrName** (Attr) const;
  - *Attribute name*

“JobStatus” class & “Event” class

Enabling Grids for E-science in Europe

Several retrieve methods, depending of the AttrType:

- int **getValInt** (Attr) const;
- string **getValString** (Attr) const;
- struct timeval **getValTime** (Attr) const;
- const JobId **getValJobId** (Attr) const;
- bool **getValBool** (Attr) const;
- const vector<string> **getValStringList** (Attr) const;
- const vector<pair<string,string> > **getValTagList** (Attr) const;
- const vector<JobStatus> **getValJobStatusList** (Attr) const;

- **The class “Event” has structure similar to “JobStatus”**
  - It is used to retrieve the Logging Info

“Exception” Class

Enabling Grids for E-science in Europe

- Namespace: **edg::workload::common::utilities**

  - **Pure virtual**
  - It inherits **std::exceptions**

Public Methods:

  - virtual string **dbgMessage** ();
    - *Return a string debug message containing information about Exception thrown*
  - virtual int **getCode** ();
    - *Return the Code number*
  - virtual const char* **what** () const throw ();
    - *return the Error Message associated to the Exception*
  - virtual void **log** (const string& logfile = "");
    - *Print Exception error information into a log file*
  - virtual string **getExceptionName** ();
    - *Return the name of the Exception raised*
  - virtual string **printStackTrace** () ;
    - *Return the list of methods that caused the Exception*