

API DIET batch management system

November 11, 2010

1 Presentation

The *DIET batch management system* is composed of a development library usable from C/C++ codes and several binaries that can be used on the command line.

The compilation of the library will add *diet-bms-lib* to the *lib* repository and also the corresponding header files *diet-bms-lib.h* and *JobStructures.h* in the *include* directory describing the functions of the API and the structures used to represent a job and a list of jobs.

2 Data structures

Two data structures representing a job and a list of jobs are available: `job_t` and `listOfJobs_t`, and their definitions are :

```
// structure of a job
typedef struct {
char* id;          // id of the job
char* name;        // name of the job
char* user;        // owner of the job
char* subTime;     // submission time
char* status;      // status of the job
char* priority;    // priority of the job
char* jobClass;    // class of the job
char* walltime;    // walltime of the job
char* nbNodes;     // number of nodes of the job
} job_t;

// structure of a list of jobs
typedef struct {
unsigned long long nbJobs;          // number of jobs
unsigned long long nbRunningJobs;   // number of running jobs
unsigned long long nbWaitingJobs;   // number of waiting jobs
job_t* jobs;                        // jobs
} listOfJobs_t;
```

3 Job submission

A function of the API can be used to submit a job with a script

```
int dietsubmit(const char* hostname, const char* scriptPath, char* jobID,
              const size_t maxsize, char* error_message, const size_t error_maxsize);
```

Function allowing the submission of a job in a machine called **hostname** with a script defined in **scriptPath**. The function returns the **jobID** of the submitted jobs if successful or an **errorMessage** generated by the batch scheduler can be retrieved. This function returns negative value if the operation failed.

4 Job deletion

A function of the API can be used to delete a job based on its id:

```
int dietcancel(const char* hostname, const char* jobId, char* outputMessage,
              const size_t maxsize, char* errorMessage, const size_t errorSize);
```

The first parameter is the machine where the job is located, the **jobId** is the id of the job to delete, **outputMessage** is the message sent by the batch scheduler after the submission, and **errorMessage** stores any error message sent by the batch scheduler. This function returns a negative value if the operation fails and zero otherwise.

5 Information on existing jobs

Four functions can be used to get information about the existing jobs on the different machines:

```
int dietlist(const char* hostname, listOfJobs_t* listOfJobs, char* errorMessage,
            const size_t errorSize);
```

```
int dietNbJobs(const char* hostname, char* errorMessage, const size_t errorSize);
```

```
int dietNbRunningJobs(const char* hostname, char* errorMessage, const size_t errorSize);
```

```
int dietNbWaitingJobs(const char* hostname, char* errorMessage, const size_t errorSize);
```

The **dietlist** retrieves the list of jobs in **listOfJobs** existing on the machine defined by **hostname**. This function returns a negative value if the operation fails and zero otherwise.

The function **dietNbJobs** retrieves the total number of jobs on **hostname**. This function returns -1 if the operation fails.

The function **dietNbRunningJobs** retrieves the number of *running* jobs on **hostname**. This function returns -1 if the operation fails.

The function **dietNbWaitingJobs** retrieves the number of *waiting* jobs on **hostname**. This function returns -1 if the operation fails.

For all these function, it's also possible to get the **errorMessage** that could have been generated by the batch scheduler if the operation fails.

6 Utility functions

Two other functions are available in the API for displaying a job, and for a list of jobs:

```
int printListOfJobs(const listOfJobs_t* listOfJobs);  
  
int printJob(const job_t* job);
```

The function `printListOfJobs` displays the `listOfJobs` in argument.

The function `printJob` displays the `job` in argument.

These functions return a value different from zero if the operation fails.

7 Example of API usage

An example code is present in the distribution in the `APITest.c` file in the *src/api* directory. This program uses the API to perform the following operations:

- Get a list of the jobs on the platform,
- Make a submission of a job based on a script,
- Get a list of the jobs (the submitted job should be visible),
- Delete the submitted job,
- Get a list of the jobs (the deleted job is no more visible).