Super Computer Simulation

Generated by Doxygen 1.8.11

Contents

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

Class Documentation

3.1 Configuration Class Reference

Configuration class.

`#include <configuration.h>`

Public Member Functions

- Configuration ()
  Configuration object default contructor.
- unsigned int get_users_nr ()
  Public method. Returns the number of users when this value is not generated randomly.
- unsigned int get_users_nr_min ()
  Public method. Returns the lower limit of randomly generated values of number of users.
- unsigned int get_users_nr_max ()
  Public method. Returns the upper limit of randomly generated values of number of users.
- bool is_users_nr_random ()
  Public method. Returns true if the number of users is generated randomly, and false if it's not.
- unsigned int get_jobs_nr ()
  Public method. Returns the number of jobs when this value is not generated randomly.
- unsigned int get_jobs_nr_min ()
  Public method. Returns the lower limit of randomly generated values of number of jobs.
- unsigned int get_jobs_nr_max ()
  Public method. Returns the upper limit of randomly generated values of number of jobs.
- bool is_jobs_nr_random ()
  Public method. Returns true if the number of jobs is generated randomly, and false if it's not.
- unsigned int get_cores_nr ()
  Public method. Returns the number of cores per node.
- unsigned int get_nodes_nr ()
  Public method. Returns the total number of system nodes.
- double get_usage_price ()
  Public method. Returns the usage price of the system per core second.
- double get_operational_cost ()
  Public method. Returns the operational cost of the system per second.

- double get_student_budget ()

  Public method. Returns the student budget.
- double get_student_budget_min ()

  Public method. Returns the lower limit of randomly generated values of student budget.
- double get_student_budget_max ()

  Public method. Returns the upper limit of randomly generated values of student budget.
- bool is_student_budget_random ()

  Public method. Returns true if the student budget is generated randomly, and false if it's not.
- double get_researcher_budget ()

  Public method. Returns the researcher budget.
- double get_researcher_budget_min ()

  Public method. Returns the lower limit of randomly generated values of researcher budget.
- double get_researcher_budget_max ()

  Public method. Returns the upper limit of randomly generated values of researcher budget.
- bool is_researcher_budget_random ()

  Public method. Returns true if the researcher budget is generated randomly, and false if it's not.
- time_t get_time ()

  Public method. Returns the starting time of the simulation in UNIX timestamp format.
- unsigned long long int get_requests_span ()

  Public method. Returns the requests time span in seconds.
- void set_student_random (bool random)

  Public method. Changes the way the student budget is defined.
- void set_researcher_random (bool random)

  Public method. Changes the way the researcher budget is defined.
- void set_users_random (bool random)

  Public method. Changes the way the number of users is defined.
- void set_jobs_random (bool random)

  Public method. Changes the way the number of jobs is defined.
- void set_now (bool now)

  Public method. Changes whether the simulation starting time is the present date or not.
- void set_time (time_t time)

  Public method. Defines a date value for non-present starting dates simulations.
- void set_usage_price (double usage_price)

  Public method. Defines the usage price of the system per core second.
- void set_operational_cost (double operational_cost)

  Public method. Defines the operational cost of the system per core second.
- void set_nodes_nr (unsigned int nodes_nr)

  Public method. Defines the total number of system nodes.
- void set_cores_nr (unsigned int cores_nr)

  Public method. Defines the number of cores per node.
- void set_student_budget (double budget)

  Public method. Defines the student budget of a new simulation.
- void set_student_budget_min (double min)

  Public method. Defines the lower limit student budgets when this value is randomly generated.
- void set_student_budget_max (double max)

  Public method. Defines the upper limit student budgets when this value is randomly generated.
- void set_researcher_budget (double budget)

  Public method. Defines the researcher budget of a new simulation.
- void set_researcher_budget_min (double min)

  Public method. Defines the lower limit researcher budgets when this value is randomly generated.
- void set_researcher_budget_max (double max)

Public method. Defines the upper limit researcher budgets when this value is randomly generated.

- void set_jobs_nr (unsigned int nr)

  Public method. Defines the number of jobs of a new simulation.
- void set_jobs_nr_min (unsigned int min)

  Public method. Defines the lower limit of jobs when this value is randomly generated.
- void set_jobs_nr_max (unsigned int max)

  Public method. Defines the upper limit of jobs when this value is randomly generated.
- void set_users_nr (unsigned int nr)

  Public method. Defines the number of users of a new simulation.
- void set_users_nr_min (unsigned int min)

  Public method. Defines the lower limit of users when this value is randomly generated.
- void set_users_nr_max (unsigned int max)

  Public method. Defines the upper limit of users when this value is randomly generated.
- void set_requests_span (unsigned long long int span)

  Public method. Defines the requests time span.

3.1.1 Detailed Description

Configuration class.

This object includes the input values of the simulation.

3.1.2 Constructor & Destructor Documentation

3.1.2.1 Configuration::Configuration ( )

Configuration object default contructor.

Private time_t. Start time of the simulation.

Initializes a Configuration object.

Default contructor of configuration. The default input values are defined in the "utils.h" header file.

3.1.3 Member Function Documentation

3.1.3.1 unsigned int Configuration::get_cores_nr ( )

Public method. Returns the number of cores per node.

Returns

unsigned int. The number of cores per node.

Ŕeturns the number of cores per node.

3.1.3.2 unsigned int Configuration::get_jobs_nr ( )

Public method. Returns the number of jobs when this value is not generated randomly.

Returns

unsigned int. The number of jobs.

Ŕeturns a constant value if the variable "jobs_random" is false, and a random value between two established limits if not.

3.1.3.3 unsigned int Configuration::get_jobs_nr_max ( )

Public method. Returns the upper limit of randomly generated values of number of jobs.

Returns

unsigned int. The upper limit.

Ŕeturns the upper limit of randomly generated values of jobs numbers.

3.1.3.4 unsigned int Configuration::get_jobs_nr_min ( )

Public method. Returns the lower limit of randomly generated values of number of jobs.

Returns

unsigned int. The lower limit.

Ŕeturns the lower limit of randomly generated values of jobs numbers.

3.1.3.5 unsigned int Configuration::get_nodes_nr ( )

Public method. Returns the total number of system nodes.

Returns

unsigned int. The number of nodes.

Ŕeturns the total number of nodes.

3.1.3.6 double Configuration::get_operational_cost ( )

Public method. Returns the operational cost of the system per second.

Returns

double. Operational cost of the system per second.

Ŕeturns the operational cost of the system per second.

3.1.3.7 unsigned long long int Configuration::get_requests_span ( )

Public method. Returns the requests time span in seconds.

Returns

time_t. Requests time span.

Ŕeturns the requests time span in seconds.

3.1.3.8 double Configuration::get_researcher_budget ( )

Public method. Returns the researcher budget.

Returns

double. Student budget value.

Ŕeturns a constant value if the variable "student_random" is false, and a random value between two established limits if not.

3.1.3.9 double Configuration::get_researcher_budget_max ( )

Public method. Returns the upper limit of randomly generated values of researcher budget.

Returns

double. The upper limit of researcher budget.

Ŕeturns the upper limit of a randomly generated researcher budget.

3.1.3.10 double Configuration::get_researcher_budget_min ( )

Public method. Returns the lower limit of randomly generated values of researcher budget.

Returns

double. The lower limit of researcher budget.

Ŕeturns the lower limit of a randomly generated researcher budget.

3.1.3.11 double Configuration::get_student_budget ( )

Public method. Returns the student budget.

Returns

double. Student budget value.

Ŕeturns a constant value if the variable "student_random" is false, and a random value between two established limits if not.

3.1.3.12 double Configuration::get_student_budget_max ( )

Public method. Returns the upper limit of randomly generated values of student budget.

Returns

double. The upper limit of student budget.

Ŕeturns the upper limit of a randomly generated student budget.

3.1.3.13 double Configuration::get_student_budget_min ( )

Public method. Returns the lower limit of randomly generated values of student budget.

Returns

double. The lower limit of student budget.

Ŕeturns the lower limit of a randomly generated student budget.

3.1.3.14 time_t Configuration::get_time ( )

Public method. Returns the starting time of the simulation in UNIX timestamp format.

Returns

time_t. Starting time of the simulation.

Returns the simulation starting date, the current UNIX time stamp if the "now" is true, a defined date if it's false.

3.1.3.15 double Configuration::get_usage_price ( )

Public method. Returns the usage price of the system per core second.

Returns

double. Usage price of the system per core second.

Ŕeturns the usage price of the system per core second.

3.1.3.16 unsigned int Configuration::get_users_nr ( )

Public method. Returns the number of users when this value is not generated randomly.

Returns

unsigned int. The number of users.

Ŕeturns a constant value if the variable "users_random" is false, and a random value between two established limits if not.

3.1.3.17 unsigned int Configuration::get_users_nr_max ( )

Public method. Returns the upper limit of randomly generated values of number of users.

Returns

unsigned int. The upper limit.

Ŕeturns the upper limit of randomly generated values of users numbers.

3.1.3.18 unsigned int Configuration::get_users_nr_min ( )

Public method. Returns the lower limit of randomly generated values of number of users.

Returns

unsigned int. The lower limit.

Ŕeturns the lower limit of randomly generated values of users numbers.

3.1.3.19 bool Configuration::is_jobs_nr_random ( )

Public method. Returns true if the number of jobs is generated randomly, and false if it's not.

Returns

bool. Number of jobs randomness.

Ŕeturns whether the number of jobs is randomly generated or not.

3.1.3.20 bool Configuration::is_researcher_budget_random ( )

Public method. Returns true if the researcher budget is generated randomly, and false if it's not.

Returns

bool. Researcher budget randomness.

Ŕeturns whether the researcher budget is randomly generated or not.

3.1.3.21 bool Configuration::is_student_budget_random ( )

Public method. Returns true if the student budget is generated randomly, and false if it's not.

Returns

bool. Student budget randomness.

Ŕeturns whether the student budget is randomly generated or not.

3.1.3.22 bool Configuration::is_users_nr_random ( )

Public method. Returns true if the number of users is generated randomly, and false if it's not.

Returns

bool. Number of users randomness.

Ŕeturns whether the number of users is randomly generated or not.

3.1.3.23 void Configuration::set_cores_nr (unsigned int *cores_nr*)

Public method. Defines the number of cores per node.

Parameters

unsigned	int nodes_nr. New value of number of cores per node.

Defines number of cores per node.

3.1.3.24 void Configuration::set_jobs_nr (unsigned int *nr*)

Public method. Defines the number of jobs of a new simulation.

Parameters

unsigned	int budget. New value of jobs number.

Defines number of jobs.

3.1.3.25 void Configuration::set_jobs_nr_max (unsigned int *max*)

Public method. Defines the upper limit of jobs when this value is randomly generated.

Parameters

unsigned	int max. New value of upper limit.

Defines the upper limit of randomly generated number of jobs, never letting this value being smaller than the upper limit.

3.1.3.26 void Configuration::set_jobs_nr_min (unsigned int *min*)

Public method. Defines the lower limit of jobs when this value is randomly generated.

Parameters

unsigned	int min. New value of lower limit.

Defines the lower limit of randomly generated number of jobs, never letting this value being higher than the upper limit.

3.1.3.27 void Configuration::set_jobs_random (bool *random*)

Public method. Changes the way the number of jobs is defined.

Parameters

bool	random. True if random generated, false if constant.

Defines whether the number of jobs is randomly generated or not.

3.1.3.28 void Configuration::set_nodes_nr (unsigned int *nodes_nr*)

Public method. Defines the total number of system nodes.

Parameters

unsigned	int nodes_nr. New value of total number of system nodes.

Defines number of nodes of the system.

3.1.3.29 void Configuration::set_now (bool *now*)

Public method. Changes whether the simulation starting time is the present date or not.

Parameters

bool	now. True if the date is the present, false if not.

Defines whether the simulation starting date is the present or not.

3.1.3.30 void Configuration::set_operational_cost (double *operational_cost*)

Public method. Defines the operational cost of the system per core second.

Parameters

double	usage_price. New value of operational cost.

Defines the value of the operational cost of the system.

3.1.3.31 void Configuration::set_requests_span (unsigned long long int *span*)

Public method. Defines the requests time span.

Parameters

unsigned	long long int span. New value of requests time span.

Defines the upper limit of requests time span in seconds.

3.1.3.32 void Configuration::set_researcher_budget (double *budget*)

Public method. Defines the researcher budget of a new simulation.

Parameters

double	budget. New value of researcher budget.

Defines value of a researcher budget.

3.1.3.33 void Configuration::set_researcher_budget_max (double *max*)

Public method. Defines the upper limit researcher budgets when this value is randomly generated.

Parameters

double	max. New value of upper limit.

Defines the upper limit of randomly generated researcher budgets, never letting this value being smaller than the upper limit.

3.1.3.34 void Configuration::set_researcher_budget_min (double *min*)

Public method. Defines the lower limit researcher budgets when this value is randomly generated.

Parameters

double	min. New value of lower limit.

Defines the lower limit of randomly generated researcher budgets, never letting this value being higher than the upper limit.

3.1.3.35 void Configuration::set_researcher_random (bool *random*)

Public method. Changes the way the researcher budget is defined.

Parameters

bool	random. True if random generated, false if constant.

Defines whether the researcher budget is randomly generated or not.

3.1.3.36 void Configuration::set_student_budget (double *budget*)

Public method. Defines the student budget of a new simulation.

Parameters

double	budget. New value of student budget.

Defines value of a student budget.

3.1.3.37 void Configuration::set_student_budget_max (double *max*)

Public method. Defines the upper limit student budgets when this value is randomly generated.

Parameters

double	max. New value of upper limit.

Defines the upper limit of randomly generated student budgets, never letting this value being smaller than the upper limit.

3.1.3.38 void Configuration::set_student_budget_min (double *min*)

Public method. Defines the lower limit student budgets when this value is randomly generated.

Parameters

double	min. New value of lower limit.

Defines the lower limit of randomly generated student budgets, never letting this value being higher than the upper limit.

3.1.3.39 void Configuration::set_student_random (bool *random*)

Public method. Changes the way the student budget is defined.

Parameters

bool	random. True if random generated, false if constant.

Defines whether the student budget is randomly generated or not.

3.1.3.40 void Configuration::set_time (time_t *time*)

Public method. Defines a date value for non-present starting dates simulations.

Parameters

time_t	time. Value of date in UNIX timestamp format.

Defines the simulation starting date.

3.1.3.41 void Configuration::set_usage_price (double *usage_price*)

Public method. Defines the usage price of the system per core second.

Parameters

double	usage_price. New value of usage price.

Defines the value of the usage price of the system.

3.1.3.42 void Configuration::set_users_nr (unsigned int *nr*)

Public method. Defines the number of users of a new simulation.

Parameters

unsigned	int budget. New value of users number.

Defines number of users.

3.1.3.43 void Configuration::set_users_nr_max (unsigned int *max*)

Public method. Defines the upper limit of users when this value is randomly generated.

Parameters

unsigned	int max. New value of upper limit.

Defines the upper limit of randomly generated number of users, never letting this value being smaller than the upper limit.

3.1.3.44 void Configuration::set_users_nr_min (unsigned int *min*)

Public method. Defines the lower limit of users when this value is randomly generated.

Parameters

unsigned	int min. New value of lower limit.

Defines the lower limit of randomly generated number of users, never letting this value being higher than the upper limit.

3.1.3.45 void Configuration::set_users_random (bool *random*)

Public method. Changes the way the number of users is defined.

Parameters

bool	random. True if random generated, false if constant.

Defines whether the number of users is randomly generated or not.

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

- supercomputer/src/configuration/configuration.h
- supercomputer/src/configuration/configuration.cpp

3.2 Job Class Reference

Job class.

`#include <job.h>`

Public Member Functions

- Job (Configuration *config, time_t time, unsigned long long int duration)
  Job object default contructor.
- time_t get_time ()
  Public method. Returns the submission time of a job.
- unsigned long long int get_duration ()
  Public method. Returns the duration of a job.
- int get_cores ()
  Public method. Returns the number of cores taken to run this job.
- double get_price ()

Public method. Returns the price to be paid to run this job.

- void set_user (User *user)

  Public method. Defines the user who submits the job.

- bool is_short ()

  Public method. Return whether a job is Short or not.

- bool is_medium ()

  Public method. Return whether a job is Medium or not.

- bool is_large ()

  Public method. Return whether a job is Large or not.

- bool is_huge ()

  Public method. Return whether a job is Huge or not.

Friends

- bool operator< (Job const &a, Job const &b)

  Operator overload. Overloads the < operator according to time of submission.

3.2.1 Detailed Description

Job class.

This object represents a job submited by an user.

3.2.2 Constructor & Destructor Documentation

3.2.2.1 Job::Job (Configuration * *config*, time_t *time*, unsigned long long int *duration*)

Job object default contructor.

Private int. Computational resources taken by this job in cores.

Initializes a Job object.

Parameters

*Configuration**	config. Defines which configuration this job should follow.
time_t	time. Submission date of job.
unsigned	long long int duration. Job duration in seconds.

Contructor of Job object. Defines what is the type of a job according to its duration, generating its number of cores randomly, following a linear distribution.

3.2.3 Member Function Documentation

3.2.3.1 int Job::get_cores ( )

Public method. Returns the number of cores taken to run this job.

Returns

int. Number of cores.

Ŕeturns the number of cores taken by a job.

3.2.3.2 unsigned long long int Job::get_duration ( )

Public method. Returns the duration of a job.

Returns

unsigned long long int. Duration of job.

Returns the job duration in seconds.

3.2.3.3 double Job::get_price ( )

Public method. Returns the price to be paid to run this job.

Returns

double. Price to be paid by a simulated user.

Ŕeturns the price to pay to run a job.

3.2.3.4 time_t Job::get_time ( )

Public method. Returns the submission time of a job.

Returns

time_t. Date represented in UNIX timestamp.

Ŕeturns the submission time of a job.

3.2.3.5 bool Job::is_huge ( )

Public method. Return whether a job is Huge or not.

Returns

bool. True if yes, false if it's not.

Ŕeturns whether the job is huge or not.

3.2.3.6 bool Job::is_large ( )

Public method. Return whether a job is Large or not.

Returns

bool. True if yes, false if it's not.

Ŕeturns whether the job is large or not.

3.2.3.7 bool Job::is_medium ( )

Public method. Return whether a job is Medium or not.

Returns

bool. True if yes, false if it's not.

Ŕeturns whether the job is medium or not.

3.2.3.8 bool Job::is_short ( )

Public method. Return whether a job is Short or not.

Returns

bool. True if yes, false if it's not.

Ŕeturns whether the job is short or not.

3.2.3.9 void Job::set_user (User * *user*)

Public method. Defines the user who submits the job.

Parameters

*User**	user. User who submits the job.

Defines the user who submited the job.

3.2.4 Friends And Related Function Documentation

3.2.4.1 bool operator< (Job const & *a*, Job const & *b*) [friend]

Operator overload. Overloads the < operator according to time of submission.

< Operator overload.

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

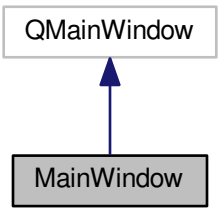
- supercomputer/src/jobs/job.h
- supercomputer/src/jobs/job.cpp

3.3 MainWindow Class Reference

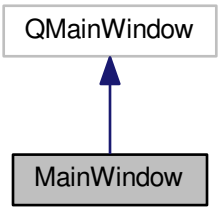
MainWindow class.

`#include <mainwindow.h>`

Inheritance diagram for MainWindow:



Collaboration diagram for MainWindow:



Public Member Functions

- MainWindow (QWidget *parent=0)

  MainWindow object default contructor.
- ~MainWindow ()

  MainWindow object default destructor.

3.3.1 Detailed Description

MainWindow class.

This object is the GUI.

3.3.2 Constructor & Destructor Documentation

3.3.2.1 MainWindow::MainWindow (QWidget * *parent* = 0) [explicit]

MainWindow object default contructor.

Initializes a MainWindow object.

3.3.2.2 MainWindow::~MainWindow ( )

MainWindow object default destructor.

Destructs a MainWindow object.

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

- supercomputer/mainwindow.h

3.4 State Class Reference

State class.

`#include <state.h>`

Public Member Functions

- State (long long int total_cores, time_t time, StateType state_type)
  State object default contructor.
- State (State state, time_t time, StateType state_type)
  State object alternative contructor.
- void set_period (time_t start, time_t end)
  Public method. Defines the period of life of a state.
- void insert_job (Job job)
  Public method. Decreases number of available cores according to job type.
- bool can_insert_job (Job job)
  Public method. Indicates whether a job can be inserted in this state or not.
- time_t get_time ()
  Public method. Returns the state time of occurence.
- string get_name ()
  Public method. Returns the state time of occurence.
- StateType get_type ()

Public method. Returns type of state.

- long long int get_short_cores ()

  Public method. Returns available number of cores reserved for short jobs.
- long long int get_medium_cores ()

  Public method. Returns available number of cores reserved for medium jobs.
- long long int get_large_cores ()

  Public method. Returns available number of cores reserved for large jobs.
- long long int get_total_cores ()

  Public method. Returns total number of cores.
- long long int get_used_cores ()

  Public method. Returns total number of used cores.

Friends

- bool operator< (State const &a, State const &b)

  Operator overload. Overloads the < operator according to time of ocurence.

3.4.1 Detailed Description

State class.

This object represents a the number of cores available in each queue at every start time and end time of a job.

3.4.2 Constructor & Destructor Documentation

3.4.2.1 State::State (long long int *total_cores*, time_t *time*, StateType *state_type*)

State object default contructor.

Private time_t. Time which this state occurs reprensented in UNIX timestamp.

Initializes a Job object.

Parameters

long	long int total_cores. Total number of system cores.
time_t	time. Date of occurence.
StateType	state_type. StateType indicating whether a job starts or ends.

Default contructor of State.

3.4.2.2 State::State (State *state*, time_t *time*, StateType *state_type*)

State object alternative contructor.

Initializes a Job object.

Parameters

State	state. State to copy cat.
time_t	time. Date of occurence.
StateType	state_type. StateType indicating whether a job starts or ends.

Alternative contructor of State.

3.4.3 Member Function Documentation

3.4.3.1 bool State::can_insert_job (Job *job*)

Public method. Indicates whether a job can be inserted in this state or not.

Parameters

Job	job. Job object containing information about what amount of computational resources to be decreased.

Returns

bool. True if job can be inserted, false if can not.

Indicates whether a job can run with the available computational resources at this system state.

3.4.3.2 long long int State::get_large_cores ( )

Public method. Returns available number of cores reserved for large jobs.

Returns

ong long int. Number of cores.

Returns the amount of available computational resources for large jobs.

3.4.3.3 long long int State::get_medium_cores ( )

Public method. Returns available number of cores reserved for medium jobs.

Returns

ong long int. Number of cores.

Returns the amount of available computational resources for medium jobs.

3.4.3.4 string State::get_name ( )

Public method. Returns the state time of occurence.

Returns

time_t. Date in UNIX timestamp.

3.4.3.5 long long int State::get_short_cores ( )

Public method. Returns available number of cores reserved for short jobs.

Returns

ong long int. Number of cores.

Returns the amount of available computational resources for short jobs.

3.4.3.6 time_t State::get_time ( )

Public method. Returns the state time of occurence.

Returns

time_t. Date in UNIX timestamp.

Returns time of ocurrence.

3.4.3.7 long long int State::get_total_cores ( )

Public method. Returns total number of cores.

Returns

ong long int. Number of cores.

Returns the total number of cores.

3.4.3.8 StateType State::get_type ( )

Public method. Returns type of state.

Returns

StateType. Start or End.

Returns state type.

3.4.3.9 long long int State::get_used_cores ( )

Public method. Returns total number of used cores.

Returns

ong long int. Number of cores.

Returns the number of used cores.

3.4.3.10 void State::insert_job (Job *job*)

Public method. Decreases number of available cores according to job type.

Parameters

Job	job. Job object containing information about what amount of computational resources to be decreased.

Decreases the available number of a computational resources available in the system according to type of job.

3.4.3.11 void State::set_period (time_t *start*, time_t *end*)

Public method. Defines the period of life of a state.

Parameters

time_t	start. Starting time.
time_t	end. Ending time.

3.4.4 Friends And Related Function Documentation

3.4.4.1 bool operator< (State const & *a*, State const & *b*) [friend]

Operator overload. Overloads the < operator according to time of ocurence.

< Operator overload.

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

- supercomputer/src/system/state.h
- supercomputer/src/system/state.cpp

3.5 Statistics Class Reference

Statistics class.

```
#include <statistics.h>
```

Public Member Functions

- Statistics (Configuration *config)
  Statistics object default contructor.
- string get_usage_price ()
  Public method. Returns the total usage price as a string.
- string get_machine_time ()
  Public method. Returns the total machine time as a string.
- string get_operational_cost ()

Public method. Returns the total operational cost as a string.

- string get_economic_balance ()

  Public method. Returns the economic balance as a string.
- string get_weekly_usage ()

  Public method. Returns the weekly usage as a string.
- string get_short_ta ()

  Public method. Returns the average turn around ratio of short jobs.
- string get_medium_ta ()

  Public method. Returns the average turn around ratio of medium jobs.
- string get_large_ta ()

  Public method. Returns the average turn around ratio of large jobs.
- string get_huge_ta ()

  Public method. Returns the average turn around ratio of huge jobs.
- string get_short_wt ()

  Public method. Returns the average waiting time of short jobs.
- string get_medium_wt ()

  Public method. Returns the average waiting time of medium jobs.
- string get_large_wt ()

  Public method. Returns the average waiting time of large jobs.
- string get_huge_wt ()

  Public method. Returns the average waiting time of huge jobs.
- void add_usage_price (double price)

  Public method. Adds price to total usage price.
- void add_operational_cost (double cost)

  Public method. Adds cost to total operational cost.
- void add_machine_time (unsigned long long int time)

  Public method. Adds time to total machine time.
- void add_job (time_t start, Job job)

  Public method. Adds job to waiting time and turn around vectors.

3.5.1 Detailed Description

Statistics class.

This object keeps statistics information structured and organized.

3.5.2 Constructor & Destructor Documentation

3.5.2.1 Statistics::Statistics (Configuration * *config*)

Statistics object default contructor.

Private unsigned long long int. Total machine time of the simulation.

Initializes a Statistics object.

Parameters

*Configuration**	config. Defines which configuration this statistics object should follow.

Contructor of Statistics object.

3.5.3 Member Function Documentation

3.5.3.1 void Statistics::add_job (time_t *start*, Job *job*)

Public method. Adds job to waiting time and turn around vectors.

Parameters

time_t	start. Time of start.
Job	job. Job to be added.

Adds job to waiting time and turn around ratio vectors according to its type. Increments the number of Short, Medium, Large or Huge jobs processed in the week of job start time.

3.5.3.2 void Statistics::add_machine_time (unsigned long long int *time*)

Public method. Adds time to total machine time.

Parameters

unsigned	long long int time. Time in seconds to be added.

Adds time to system total machine time.

3.5.3.3 void Statistics::add_operational_cost (double *cost*)

Public method. Adds cost to total operational cost.

Parameters

double	cost. Cost to be added.

Adds operational cost to system total operational cost.

3.5.3.4 void Statistics::add_usage_price (double *price*)

Public method. Adds price to total usage price.

Parameters

double	price. Price to be added.

Adds usage price to system total usage price.

3.5.3.5 string Statistics::get_economic_balance ( )

Public method. Returns the economic balance as a string.

Returns

string. Economic balance string.

Returns system economic balance as a string with a precision of 2 decimal places.

3.5.3.6 string Statistics::get_huge_ta ( )

Public method. Returns the average turn around ratio of huge jobs.

Returns

string. Turn around ratio of huge jobs string.

Returns average of turn around times of huge jobs as a string with a precision of 2 decimal places.

3.5.3.7 string Statistics::get_huge_wt ( )

Public method. Returns the average waiting time of huge jobs.

Returns

string. Average waiting time of huge jobs string.

Returns average of waiting times of huge jobs as a string with a precision of 2 decimal places.

3.5.3.8 string Statistics::get_large_ta ( )

Public method. Returns the average turn around ratio of large jobs.

Returns

string. Turn around ratio of large jobs string.

Returns average of turn around times of large jobs as a string with a precision of 2 decimal places.

3.5.3.9 string Statistics::get_large_wt ( )

Public method. Returns the average waiting time of large jobs.

Returns

string. Average waiting time of large jobs string.

Returns average of waiting times of large jobs as a string with a precision of 2 decimal places.

3.5.3.10 string Statistics::get_machine_time ( )

Public method. Returns the total machine time as a string.

Returns

string. Machine time string.

Returns the machine time as a string with the number of days, hours, minutes and seconds.

3.5.3.11 string Statistics::get_medium_ta ( )

Public method. Returns the average turn around ratio of medium jobs.

Returns

string. Turn around ratio of medium jobs string.

Returns average of turn around times of medium jobs as a string with a precision of 2 decimal places.

3.5.3.12 string Statistics::get_medium_wt ( )

Public method. Returns the average waiting time of medium jobs.

Returns

string. Average waiting time of medium jobs string.

Returns average of waiting times of medium jobs as a string with a precision of 2 decimal places.

3.5.3.13 string Statistics::get_operational_cost ( )

Public method. Returns the total operational cost as a string.

Returns

string. Operational cost string.

Returns system total operational cost as a string with a precision of 2 decimal places.

3.5.3.14 string Statistics::get_short_ta ( )

Public method. Returns the average turn around ratio of short jobs.

Returns

string. Turn around ratio of short jobs string.

Returns average of turn around times of short jobs as a string with a precision of 2 decimal places.

3.5.3.15 string Statistics::get_short_wt ( )

Public method. Returns the average waiting time of short jobs.

Returns

string. Average waiting time of short jobs string.

Returns average of waiting times of short jobs as a string with a precision of 2 decimal places.

3.5.3.16 string Statistics::get_usage_price ( )

Public method. Returns the total usage price as a string.

Returns

string. Usage price string.

Returns system total usage price as a string with a precision of 2 decimal places.

3.5.3.17 string Statistics::get_weekly_usage ( )

Public method. Returns the weekly usage as a string.

Returns

string. System weekly usage.

Returns system weekly usage as a string.

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

- supercomputer/src/statistics/statistics.h
- supercomputer/src/statistics/statistics.cpp

3.6 System Class Reference

System class.

```
#include <system.h>
```

Public Member Functions

- System (Configuration *config)
  System object default contructor.
- System (Configuration *config, vector< User * > users, vector< Job > jobs)
  System object contructor for custom tests.
- string get_results ()
  Public method. Fetchs results of statistics object, returning a string with a specific format.

3.6.1 Detailed Description

System class.

This object represents the computing system.

3.6.2 Constructor & Destructor Documentation

3.6.2.1 System::System (Configuration * *config*)

System object default contructor.

Initializes a System object.

Parameters

Configuration	* config. Configuration to be followed by the simulation.

Default contructor of System object. Populates vector of users and jobs. Runs the scheduler algorithm and calculates the operating cost of the system.

3.6.2.2 System::System (Configuration * *config*, vector< User * > *users*, vector< Job > *jobs*)

System object contructor for custom tests.

Initializes a System object.

Parameters

Configuration	* config. Configuration to be followed by the simulation.
vector<User>*	users. Simulated users vector.
vector<Job>	jobs. Simulated jobs vector.

Alternative contructor of System object, indicated for testing purposes. Runs the scheduler algorithm and calculates the operating cost of the system.

3.6.3 Member Function Documentation

3.6.3.1 string System::get_results ( )

Public method. Fetchs results of statistics object, returning a string with a specific format.

Returns

string. String with information about the outputs of the simulation.

Returns string with informations about the outputs of the simulation.

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

- supercomputer/src/system/system.h
- supercomputer/src/system/system.cpp

3.7 User Class Reference

System class.

`#include <user.h>`

Public Member Functions

- User (Configuration *config, int id, bool support)
  User object default contructor.
- bool can_afford (Job *job)
  Public method. Method to check if an User can afford to run a given job.
- void pay (Job *job)
  Public method. Decreases the user budget according to the price of a job.

3.7.1 Detailed Description

System class.

This object represents a simulated user.

3.7.2 Constructor & Destructor Documentation

3.7.2.1 User::User (Configuration * *config,* int *id,* bool *support*)

User object default contructor.

Initializes a User object.

Parameters

Configuration	∗ config. Configuration to be followed by the simulation.
int	id. User id, this value is unique between users.
bool	support. The user is part of the IT Support if true.

Default contructor of User.

3.7.3 Member Function Documentation

3.7.3.1 bool User::can_afford (Job ∗ *job*)

Public method. Method to check if an User can afford to run a given job.

Parameters

Job	∗ job. Job to be paid.

Returns

bool. True if user can afford the job, false if not.

Method to check if an user can afford for a given job.

3.7.3.2 void User::pay (Job ∗ *job*)

Public method. Decreases the user budget according to the price of a job.

Parameters

Job	∗ job. Job to be paid.

Method to decrease the user budget, according to the price of a given job.

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

- supercomputer/src/users/user.h
- supercomputer/src/users/user.cpp

3.8 Week Class Reference

Week class.

`#include <week.h>`

Public Member Functions

- Week (time_t start, time_t end)

  Week object default contructor.
- time_t get_start ()

  Public method. Returns the starting date of a job.
- time_t get_end ()

  Public method. Returns the ending date of a job.
- void set_start (time_t start)

  Public method. Defines new starting date of a week.
- void add_job (Job job)

  Public method. Defines new starting date of a week.

Friends

- ostream & operator<< (ostream &os, const Week &week)

  Operator overload. Overloads the << operator of a Week object.

3.8.1 Detailed Description

Week class.

This object keeps information about number of jobs processed in a week by each queue.

3.8.2 Constructor & Destructor Documentation

3.8.2.1 Week::Week (time_t *start*, time_t *end*)

Week object default contructor.

Private unsigned int. Number of huge jobs processed.

Initializes a Week object.

Parameters

time_t	start. Defines the stating date of the week.
time_t	end. Defines the ending date of the week.

Contructor of Week object.

3.8.3 Member Function Documentation

3.8.3.1 void Week::add_job (Job *job*)

Public method. Defines new starting date of a week.

Parameters

time_t	start. Starting date represented in UNIX timestamp.

Increments the number of jobs processed this week according to its type.

3.8.3.2 time_t Week::get_end ( )

Public method. Returns the ending date of a job.

Returns

time_t. Date represented in UNIX timestamp.

Ŕeturns the ending time of a week.

3.8.3.3 time_t Week::get_start ( )

Public method. Returns the starting date of a job.

Returns

time_t. Date represented in UNIX timestamp.

Ŕeturns the starting time of a week.

3.8.3.4 void Week::set_start (time_t *start*)

Public method. Defines new starting date of a week.

Parameters

time_t	start. Starting date represented in UNIX timestamp.

Defines a start time of a week.

3.8.4 Friends And Related Function Documentation

3.8.4.1 ostream& operator<< (ostream & *os,* const Week & *week*) [friend]

Operator overload. Overloads the << operator of a Week object.

Parameters

ostream&	os. Ostream.
const	Week& week. Week to be converted.

Returns

ostream&. Week object converted to ostream.

Converts the week object to a specific ostream output format.

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

- supercomputer/src/statistics/week.h
- supercomputer/src/statistics/week.cpp