Architecture and basic principles

Juan Antonio Lopez Perez, CERN
December 8, 2005

Contents

- Introduction
- Basic structure of BOINC
- The clients
- The server
- BOINCification of applications

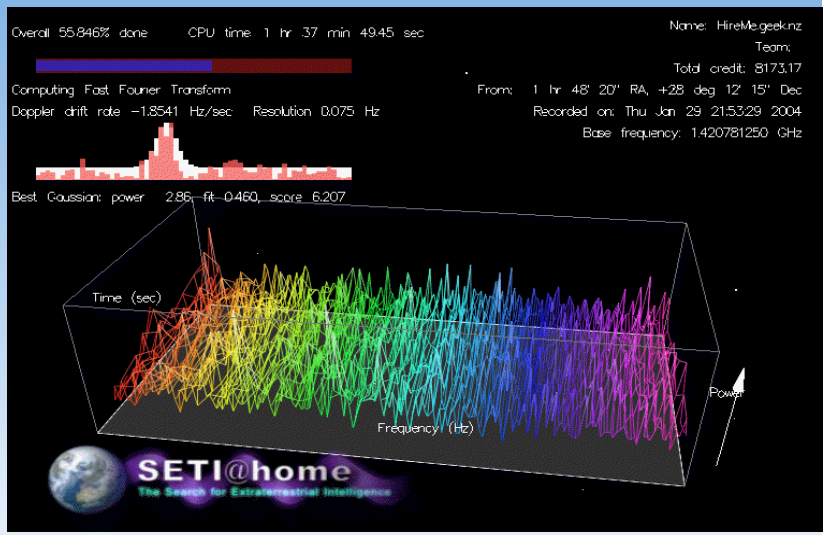
Introduction

- Brief history
  - 1967: Jocelyn Bell discover astronomical radio signals pulsating at regular intervals.
    - Extra-terrestrial signals? Finally just pulsars

  - 1992: MOP (Microwave Observing Program). First SETI project by NASA. Cancelled by U.S. Congress 1 year after.
  - 1995: Project Phoenix by SETI program. Private funding.
  - Then, David Gedye proposed to analyse those signals using a virtual supercomputer composed of large numbers of volunteered Internet-connected computers → SETI@home

Introduction

- 1999: Launch of the SETI@home project
- 2004: Next generation of SETI@home software using the BOINC platform
- Currently SETI@home has
  - > 9 million years of aggregate computing time
  - > 5 millions of clients



Introduction

- Other BOINC-powered projects:

climateprediction.net

> 10.000 CPU years processed
> 115 000 registered CPUs

> 6.800 CPU years processed
> 137 000 registered CPUs

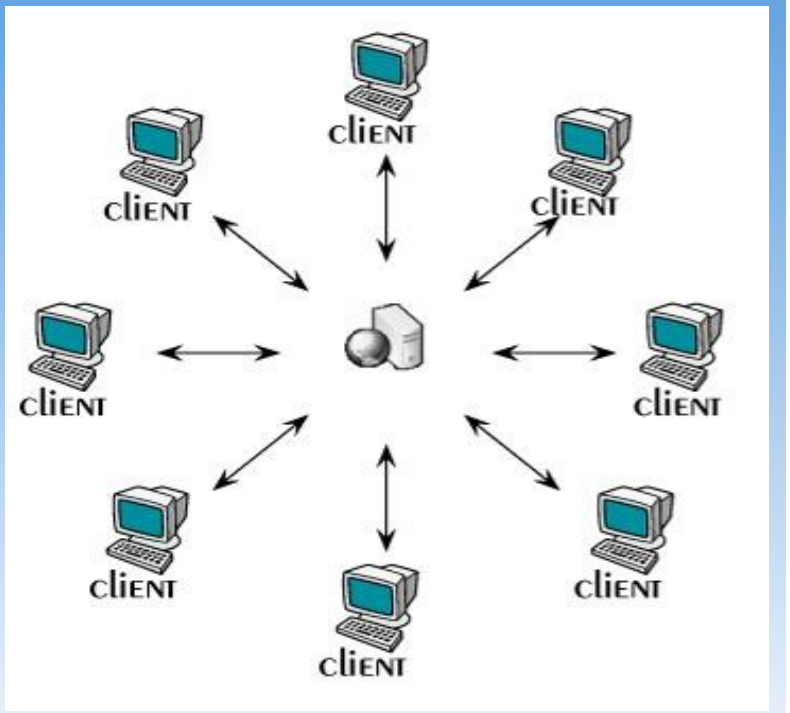
> 1.700 CPU years processed
> 84 000 registered CPUs

> 700 CPU years processed
> 44 000 registered CPUs

1 CPU = 1 KSfp2K = 2.8 GHz Xeon

Introduction

- Berkeley Open Infrastructure for Network Computing
- "software platform for **distributed computing** using volunteered computer resources"
- http://boinc.berkeley.edu
- it uses the unused CPU cycles to analyse scientific data
- Structure of type **client-server**
- Free and open source



Contents

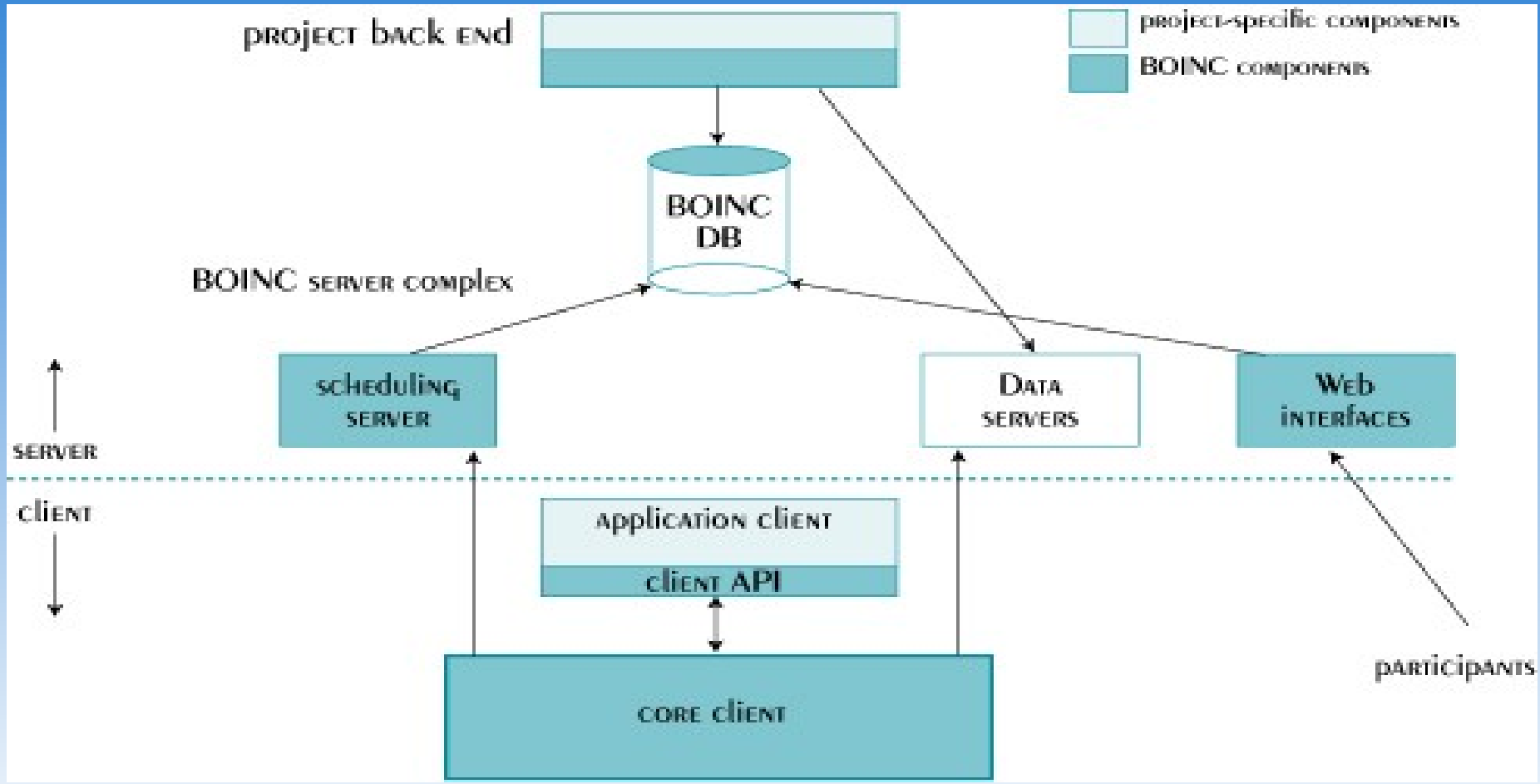
- Introduction
- Basic structure of BOINC
- The clients
- The server
- BOINCification of applications

Basic structure of BOINC

- BOINC can be seen as a kind of **Grid**. Main features:
  - BOINC requires applications with some restrictions like small I/O ratio and high CPU requirements. Other Grids have no limitations
  - BOINC infrastructure is lighter (it just needs a server)
  - BOINC clients are in most cases volunteered machines
- It's possible to send BOINC jobs to other kinds of Grid environments and vice versa

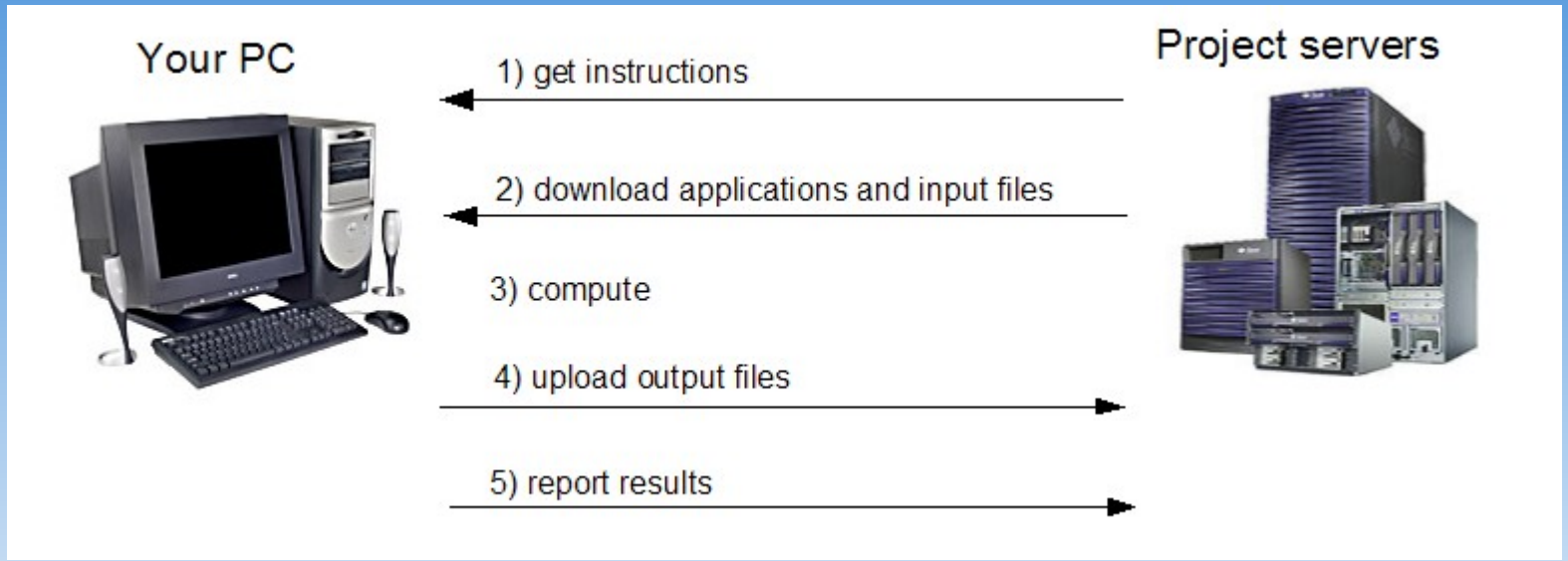
Basic structure of BOINC

- BOINC infrastructure



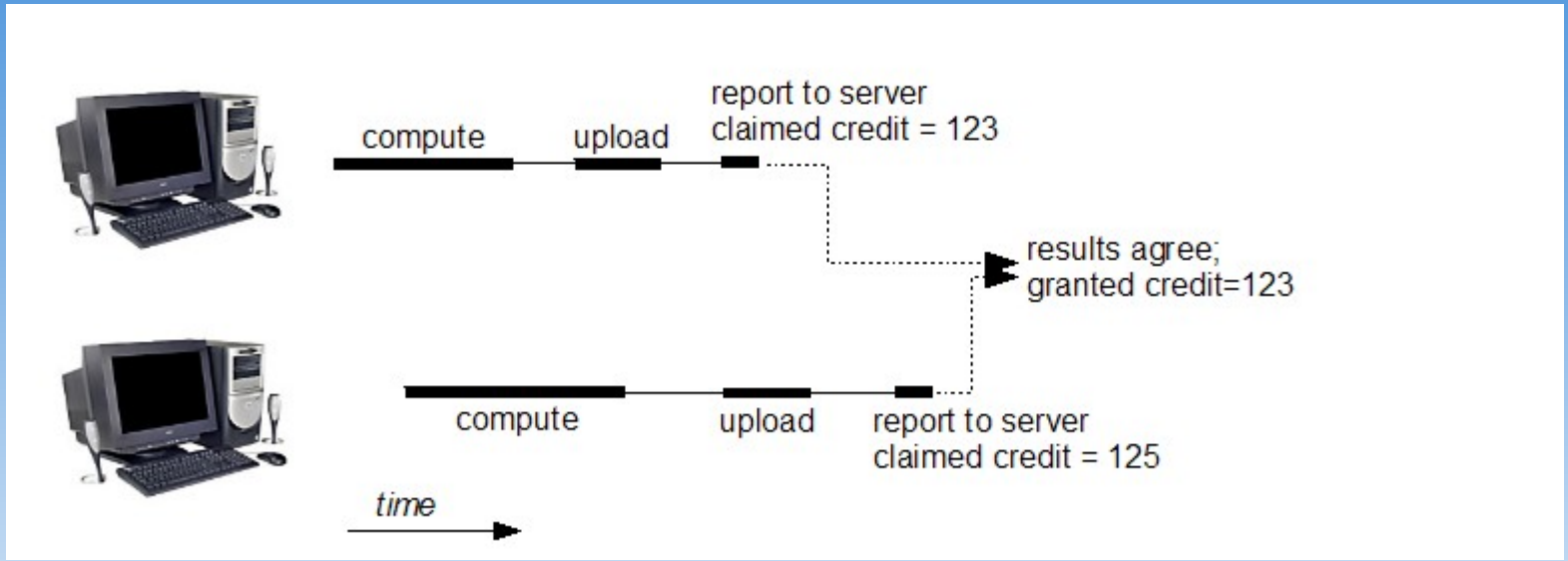
Basic structure of BOINC

- Interaction between **client** and **server**



Basic structure of BOINC

- Identifying erroneous results and obtaining credits



This process is called Redundant Computing

Basic structure of BOINC

- Identifying erroneous results and obtaining credits
  - Each result has a timeout. If it doesn't arrive on that time, it is re-sent to another client.
    - They can also be sent then to a dedicated farm (meta scheduling)
  - We can customize
    - the number of machines we send the same job
    - how many equal results we need to consider a result as good
    - the timeout

Basic structure of BOINC

- BOINC uses **code signing** to prevent **malicious executable distribution**
- All files associated with the applications are sent with digital signatures (they can also be created by BOINC)
- Each project has a key pair for code signing (private key in the server, public one in the clients)

Basic structure of BOINC

- The application and all the required files are placed in a secured **sandbox**.
  - Their physical names must be “resolved” using the BOINC API to be able to access them.
- BOINC also prevents **denial of service attacks** to the server, **result falsifications** and **credit falsifications**.

Contents

- Introduction
- Basic structure of BOINC
- The clients
- The server
- BOINCification of applications

The clients

- Steps to participate in a BOINC project
  1. Selection of the project/s in which the user will participate
  2. User registration
  3. Client software download
  4. Attachment of the client software to the selected project/s

The clients

1. Selection of the project/s in which the user will participate
   - Current available projects:
     - see http://boinc.berkeley.edu

The clients

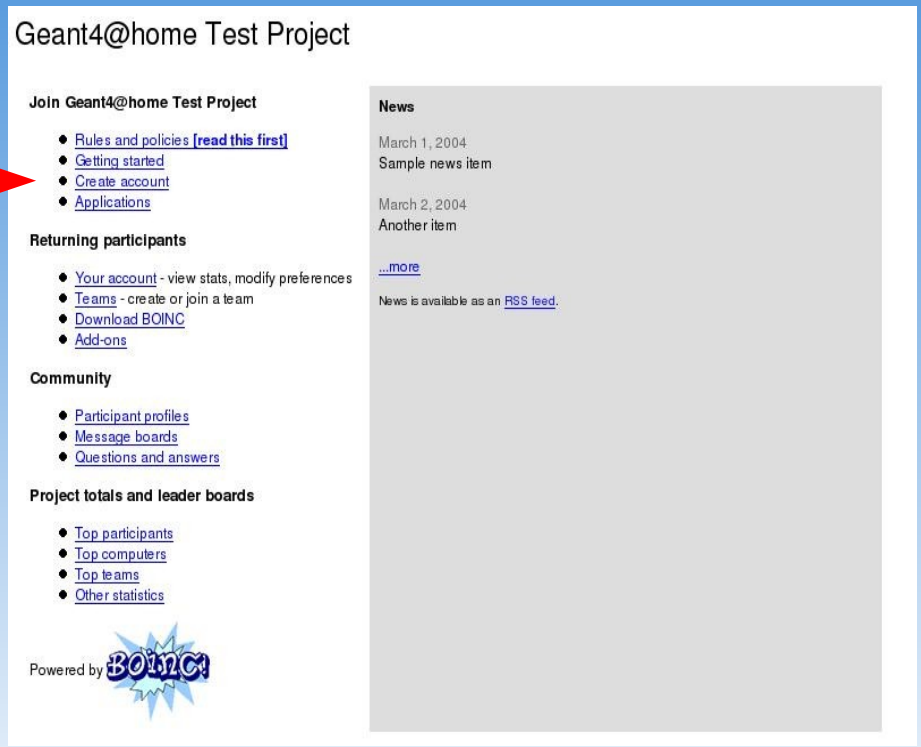
1. Selection of the project/s in which the user will participate
   - It's very important for the project to attract the volunteers
     - spending some time having a nice web
     - explaining the relevance of the project in which they will collaborate

The clients

2. User registration

- Standard web interface for all BOINC projects

create account →



The clients

2. User registration

- Registration process:
  - Fill the web form
  - Receive the confirmation mail with the account key
  - Repeat the registration for other possible projects

The clients

3. Download of the client software
   - From the BOINC web site or from each project's web site
   - Available for different platforms
     - Windows
     - Linux / 86
     - Mac OS X
     - Solaris / Sparc

The clients

3. Download of the client software

- In Windows the software is an .exe file to be executed and installed in the usual Windows way
  - It works as a screen saver application.
  - Works when the computer is idle.
  - Shows nice images/animations while the computation is being performed.
  - The screen savers depend on the application and have to be provided by each one (there is a BOINC default one).

The clients

3. Download of the client software

- In Linux (and the other UNIX-like systems) the software is mainly a **binary** which you copy and execute were you wish.
  - There is also a **GUI** manager

The clients

3. Download of the client software

- No RPM or DEB currently available at BOINC web page but one developed by us at CERN
- It's recommended to run the client as a daemon and executed by a special user
  - The process involves the creation of the user, the installation of the program and the execution of a startup script.
  - Our RPM does that in our lxboinc cluster attaching BOINC to LHC@home.

The clients

4. Attachment of the client software to the selected project/s
   - You have to tell to the client the project/s URL and their account key/s
   - In Windows (now also in the Linux/UNIX versions) you use a GUI
   - In Linux/UNIX versions you can also execute a command
   - You can de-attach from a given project or join another ones at any moment using the same ways
   - You can also join different groups to compete for credits

The clients

- Execution of the client
  - In Windows is a screen saver.
  - In Linux/UNIX you just execute the binary, which runs in low priority mode.
    - It shows the log information directly to the screen so you usually redirect it to log files.
    - Of course, then you have to manage those log files as they increase without limit.

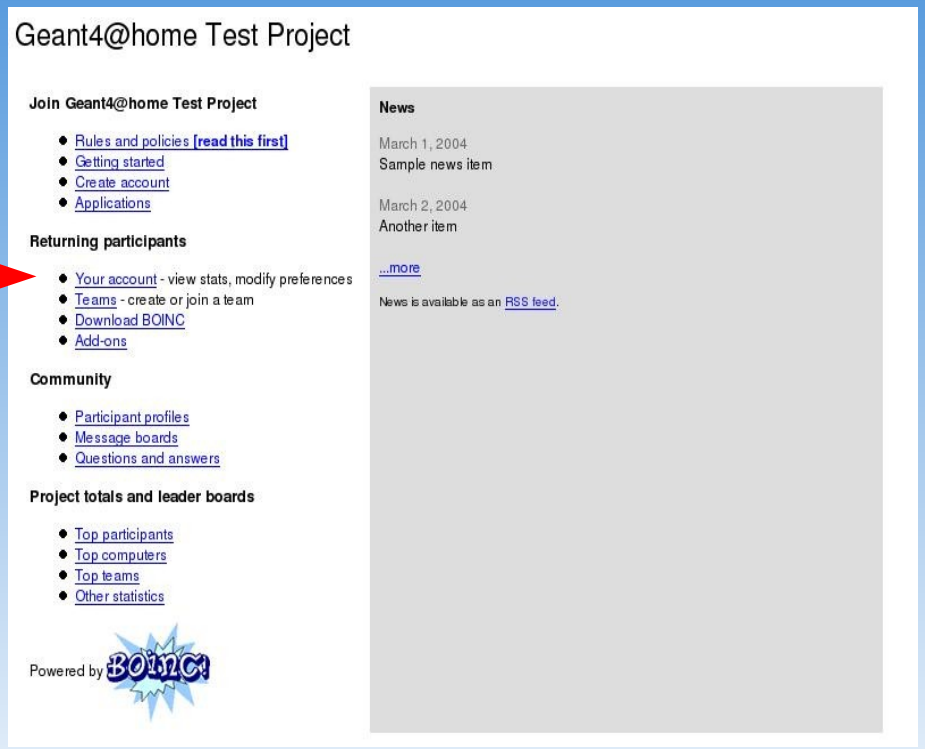
The clients

- Monitoring of the work done
  - You have a graphical interface (installed with the client in Windows and Linux/UNIX) which gives some basic information regarding the performed work
  - In Linux/UNIX you also have the standard output / error of the application (usually redirected to plain text log files).
  - In the project's public web interface, the users can log in and see their current and claimed credits.
  - You also have different public web services to monitor the current status of a given client and its evolution (groups).

The clients

- User web interface

log in to see your credits & options



Contents

- Introduction
- Basic structure of BOINC
- The clients
- The server
- BOINCification of applications

The server

- Compilation / installation
  - Currently it does not exist an official BOINC server RPM to automate the installation of servers
    - We have done one at CERN and we are testing it
    - The creation of that RPM required changes in the code
  - Some bugs found. In example, “make install” doesn't work so there is not a standard installation method.
  - Bug corrections and spec file will be sent back to BOINC
  - There is a lack of documentation. We have made some at https://uimon.cern.ch/twiki/bin/view/LHCAtHome/LinksAndDocs

The server

- Compilation / installation
  - An example of installation in Linux is
    - Download the code

```
cvs -d :pserver:anonymous:@alien.ssl.berkeley.edu:/home/cvs/cvsroot \
      checkout -r stable boinc
```

    - Build the executables

```
./_autosetup
./configure
make
```

The server

- Configuration
  1. Configuration of MySQL
  2. Creation of a project's space (directories/files hierarchy)
  3. Configuration of Apache
  4. Configuration of project web interfaces (user and management ones)

The server

- Configuration
  1. Configuration of MySQL
     - Start of the server
     - Creation of the BOINC MySQL user
     - Modification of permissions for the BOINC and Apache users

```
/etc/rc.d/init.d/mysqld start
mysql -u root
grant all on *.* to boincadm@localhost identified by 'passwd1';
grant all on *.* to boincadm identified by 'passwd1';
grant all on *.* to apache@localhost identified by 'passwd2';
grant all on *.* to apache identified by 'passwd2';
```

The server

- Configuration

  2. Creation of a project's space

     - Example command

```
tools/make_project    --user_name=boincadm                          \
                      --project_root /opt/boinc/project/geant01     \
                      --key_dir /opt/boinc/project/geant01/keys     \
                      --db_user boincadm                            \
                      --db_passwd 'passwd1'                         \
                      --delete_prev_inst                            \
                      --drop_db_first                               \
                      geant01                                       \
                      'Geant4@home'
```

The server

- Configuration

  2. Creation of a project's space

     - Example of directories/files hierarchy

project_root directory

apps : applications directory
bin : management executables
download : files to download by the client
templates : work unit and result templates
upload : upload files by the client
...

The server

- Configuration

  3. Configuration of Apache (badly documented also)

  - Copy of .conf file generated in the creation of the project's space to the Apache httpd directory
  - Restart of Apache

```
cp  /opt/boinc/project/geant01/geant01.httpd.conf  /etc/httpd/conf.d/
/etc/init.d/httpd restart
```

The server

- Configuration
  4. Configuration of project web interfaces (user and management ones)
     - Copy of project.xml file from the build directory to the project
     - Modify some configuration files
     - Run **bin/xadd** to update the BOINC database
     - (possible problems found: SELinux, lack of documentation)

```
cp tools/project.xml /opt/boinc/project/geant01
vi /opt/boinc/project/geant01/project.xml
vi /opt/boinc/project/geant01/config.xml
vi /opt/boinc/project/geant01/html/project/project.inc
bin/xadd
```

The server

- With the BOINC management web interface you can monitor the work units sent and their results.

The server

- Example of successful job done

1 results

Server state	# results
Inactive	0
Unsent	0
Unsent (in work seq)	0
In Progress	0
Over	1

'Over' results

Outcome	# results
Init	0
Success	1
Couldn't send	0
Client error	0
No reply	0
Didn't need	0
Validate error	0

'Success' results

Validate state	# results
Initial	1
Valid	0
Invalid	0
Skipped	0
Inconclusive	0
Too late	0

File Delete state	# results
Initial	1
Ready to delete	0
Deleted	0
Delete Error	0

'Client error' results

Client state	# results
Downloading	0
Downloaded	0
Computing	0
Uploading	0
Done	0

The server

- Example of job exited with error

2 results

Server state	# results
Inactive	0
Unsent	1
Unsent (in work seq)	0
In Progress	0
Over	1

'Over' results

Outcome	# results
Init	0
Success	0
Couldn't send	0
Client error	1
No reply	0
Didn't need	0
Validate error	0

'Success' results

Validate state	# results
Initial	0
Valid	0
Invalid	0
Skipped	0
Inconclusive	0
Too late	0

File Delete state	# results
Initial	0
Ready to delete	0
Deleted	0
Delete Error	0

'Client error' results

Client state	# results
Downloading	0
Downloaded	0
Computing	1
Uploading	0
Done	0

The server

- You can also monitor the registered users, applications, hosts...

Geant4@home Test Project: Project Management

Browse database:

- Platforms
- Applications
- Application versions
- Users
- Teams
- Hosts
- Workunits
- Results

Maintain and Modify database:

- Screen user profiles for User of the Day
- Forum repair
- Team repair
- Repair a validator problem
- Update forum activities
- Update profile pages (Should run as cron-job)
- Update user of the day (Should run as cron-job)
- Create account
- Cancel workunits
- Manage special users
- Email user with misconfigured host

The server

- Regarding management, you can create accounts, cancel work units, update the forum...

Geant4@home Test Project: Project Management

Browse database:

- Platforms
- Applications
- Application versions
- Users
- Teams
- Hosts
- Workunits
- Results

Maintain and Modify database:

- Screen user profiles for User of the Day
- Forum repair
- Team repair
- Repair a validator problem
- Update forum activities
- Update profile pages (Should run as cron-job)
- Update user of the day (Should run as cron-job)
- Create account
- Cancel workunits
- Manage special users
- Email user with misconfigured host

Contents

- Introduction
- Basic structure of BOINC
- The clients
- The server
- BOINCification of applications

0. Preparation

- Suitable applications to port
  - High CPU requirements
  - Low data / compute ratio
- Desirable properties
  - Written in C, C++ or Fortran
  - Easy to find/substitute I/O calls
  - Checkpointable in case of long work units
  - Easier to implement if SPMD (single program/multiple data)

1. Porting of the application

- Changes to do in the code:
  - We include the relevant BOINC headers
  - We call **boinc_init** before calling BOINC functions and **boinc_finish** at the end.
  - We have to 'resolve' all the I/O file names (because of the sandbox) using **boinc_resolve_filename** before using them to get the physical names with the right path in the client.
  - There are a lot of other useful BOINC functions like **boinc_zip**, **boinc_fraction_done** or **boinc_fopen** .

1. Porting of the application

- We have already ported three applications: SixTrack, SpinGlasses and the Geant4 Release Test Code
  - Our Geant4 Code had
    - > 9300 code files
    - > 1700 data files to be sent with the binary
    - about 100 lines of code modified/added
    - ported also to Windows using Cygwin and MS C++
- We also have been studying the possibility of using **CoLinux** to encapsulate the Linux BOINC client in Windows

1. Porting of the application

- Example from our Geant4 Release Test Code

```
#include "BOINC/boinc_api.h"
#include "BOINC/boinc_zip.h"
#include "BOINC/filesys.h"
...
int main() {
...
  boinc_init();
  retval = boinc_resolve_filename("PhotonEvaporation.zip", resolved_name_zip,
sizeof(resolved_name_zip));
  boinc_zip(UNZIP_IT, resolved_name_zip, ".");
...
  boinc_finish(0);
  return 0;
}
```

2. Registration in the server

- Remember the directory structure of the project

project_root directory

- **apps** : applications directory
- **bin** : management executables
- **download** : files to download by the client
- **templates :** work unit and result templates
- **upload** : upload files by the client
- **...**

just copy the application into the **apps** directory, following some rules on the naming of the files and directories

3. Creation of templates

- They must be into the **templates** directory

project_root directory

apps : applications directory
bin : management executables
download : files to download by the client
templates : work unit and result templates
upload : upload files by the client
...

3. Creation of templates

- **Work unit** templates tells the client which file/s the binary needs to do the work (and to be different from other jobs)
- **Result** templates tells the client which file/s need to be uploaded to the server (containing the results)
- Their XML syntax is badly documented and it's very difficult to debug those syntax errors.
- Here you have some **examples** :

3. Creation of templates

```
<file_info>
  <number>0</number>
</file_info>
<file_info>
  <number>1</number>
</file_info>
<workunit>
   <file_ref>
      <file_number>0</file_number>
      <open_name>run.g4</open_name>
   </file_ref>
   <file_ref>
      <file_number>1</file_number>
      <open_name>seed.txt</open_name>
   </file_ref>
</workunit>
```

Work unit template

File identifiers

Real file names
associated to the identifiers

3. Creation of templates

```
<file_info>
   <name><OUTFILE_0/></name>
    <generated_locally/>
    <upload_when_present/>
   <max_nbytes>102400</max_nbytes>
   <url><UPLOAD_URL/></url>
</file_info>
<result>
   <file_ref>
      <file_name><OUTFILE_0/></file_name>
      <open_name>my_stdout.txt</open_name>
   </file_ref>
</result>
```

Result template

File identifiers and other properties

Real file names associated to the identifiers

4. Generating jobs & results

- You use a command to generate a job

```
bin/create_work                                                    \
      -appname mainStatAccepTest                                   \
      -wu_name test_wu_1501                                        \
      -wu_template  templates/mainStatAccepTest_0.1_wu_template_zip_1304.xml   \
      -result_template templates/mainStatAccepTest_0.1_result_template_1301.xml  \
      -min_quorum 1                                                \
      -target_nresults 1                                           \
      run.g4 seed.txt
```

  - Each work unit (job) has a different name
  - You have to provide application name, work unit and result templates, needed files to download and other options

4. Generating jobs & results

- If the server is on (**bin/start**), it will send the job to the clients and they will compute and send back the results to the **upload** directory.
- The results are stored there in “random” directories and with the name of the job (work unit)
- Example:

```
upload/2e8/test_wu_1501_0_0
upload/2e8
upload/391/test_wu_1305_0_0
upload/391
upload/245/test_wu_1401_0_0
...
```

Thanks!

- More information:
  - LHC@home: http://lhcathome.cern.ch
  - CERN LHC@home Twiki web pages: https://uimon.cern.ch/twiki/bin/view/LHCAtHome/LinksAndDocs
  - Our BOINC presentation at CIEMAT in Madrid: https://uimon.cern.ch/twiki/pub/LHCAtHome/LinksAndDocs/boincciemat06.pdf
  - BOINC: http://boinc.berkeley.edu
  - BOINC Wiki: http://boinc-doc.net/boinc-wiki
  - SETI: http://setiathome.berkeley.edu/sah_about.php
  - Wikipedia: http://wikipedia.org, look for SETI, BOINC...