EGEE Application Case Study: Distributed Drug Analysis Platform

Hurng-Chun Lee

Hurng-Chun.Lee@cern.ch

Academia Sinica Grid Computing Centre (ASGC)

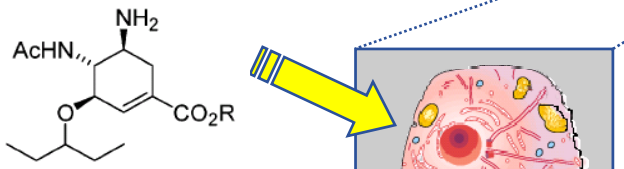
Coordinator of Drug discovery in EGEE:

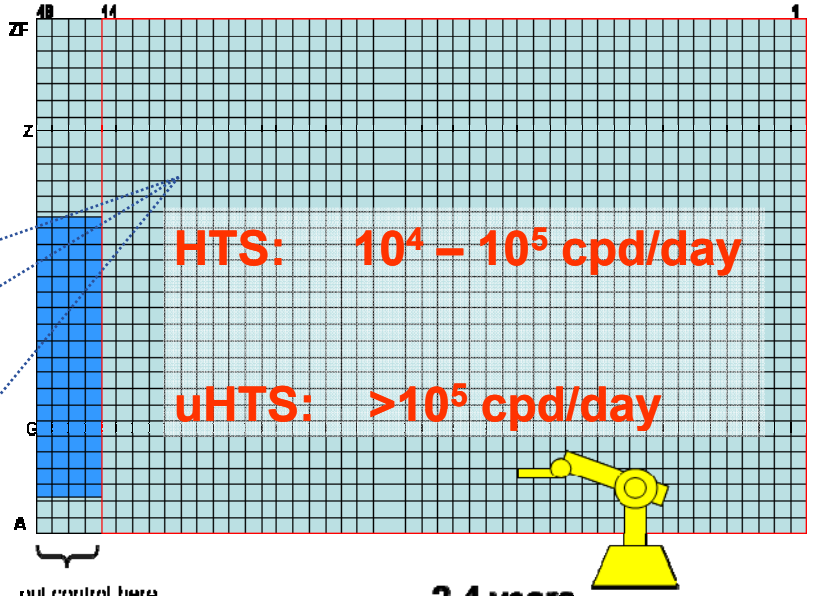
Vincent Breton

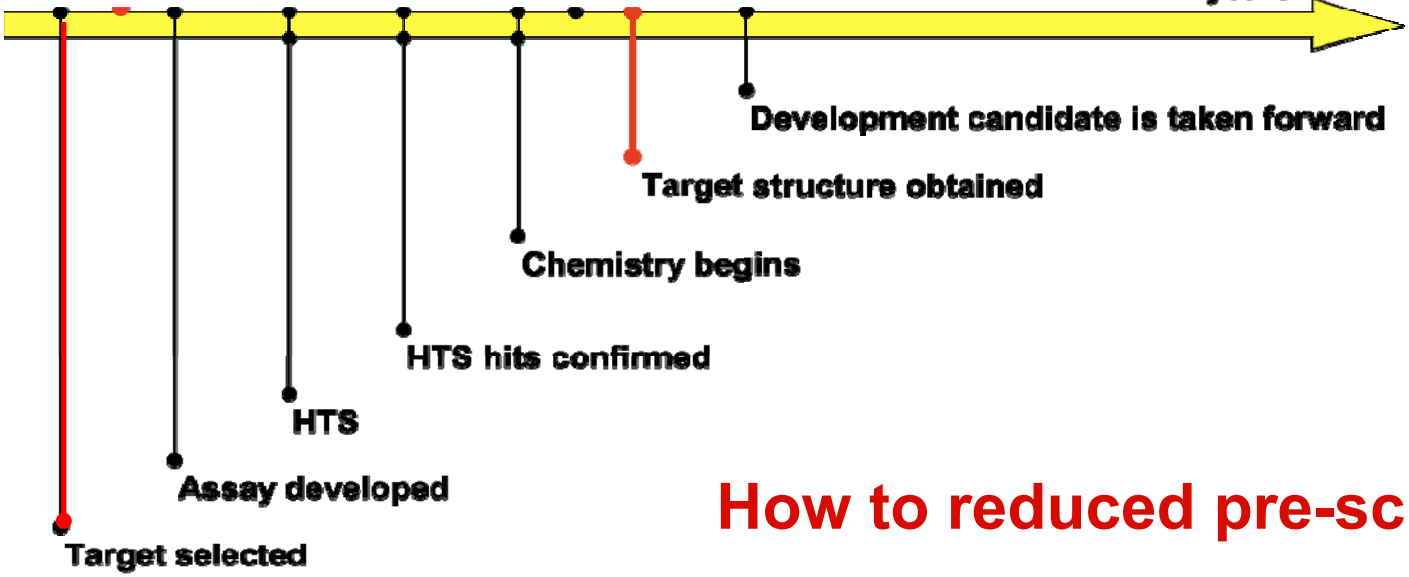
General introduction

"A needle in a haystack"

Screening is the first measure to take for the biological activity of each compound in a large compound collection against an disease target.





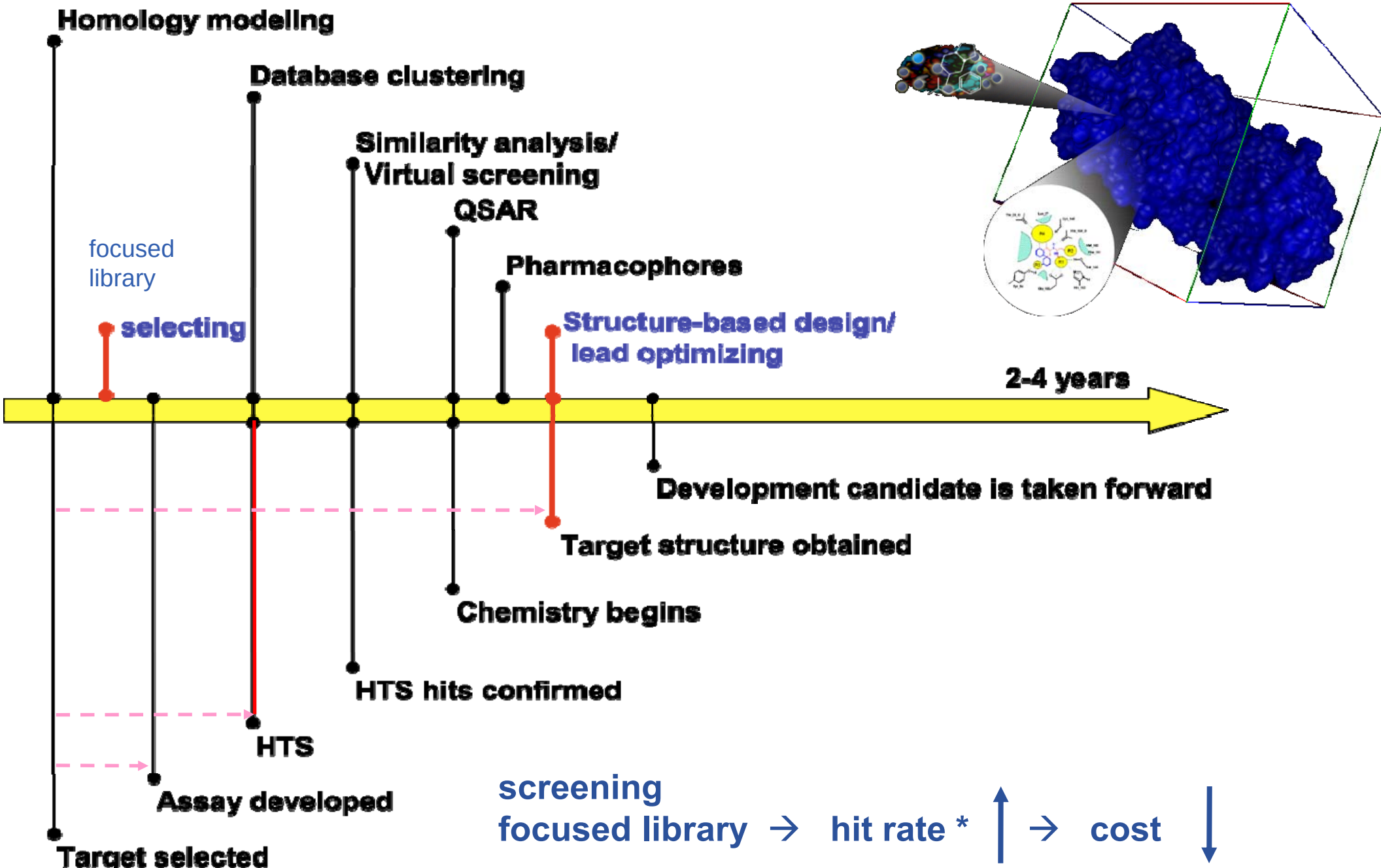


How to reduced pre-screening cost$?

The wet-lab cost

- more than 1 year for assay development
- US$ 2 for 1 compound

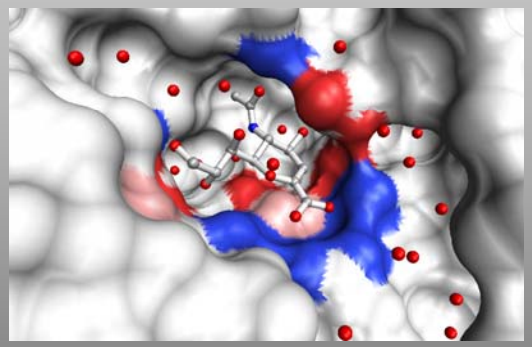
prove hit rate$ using the computer simulation



screening
focused library → hit rate * ↑ → cost ↓

Millions of chemical compounds available in laboratories

High Throughput Screening
$2/compound, nearly impossible

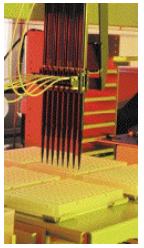


Target (PDB) :
Neuraminidase (8 structures)

Molecular docking (Autodock)
~137 CPU years, 600 GB data

Data challenge on EGEE, Auvergrid, TWGrid
~6 weeks on ~2000 computers

Hits sorting and refining



In vitro screening of 100 hits

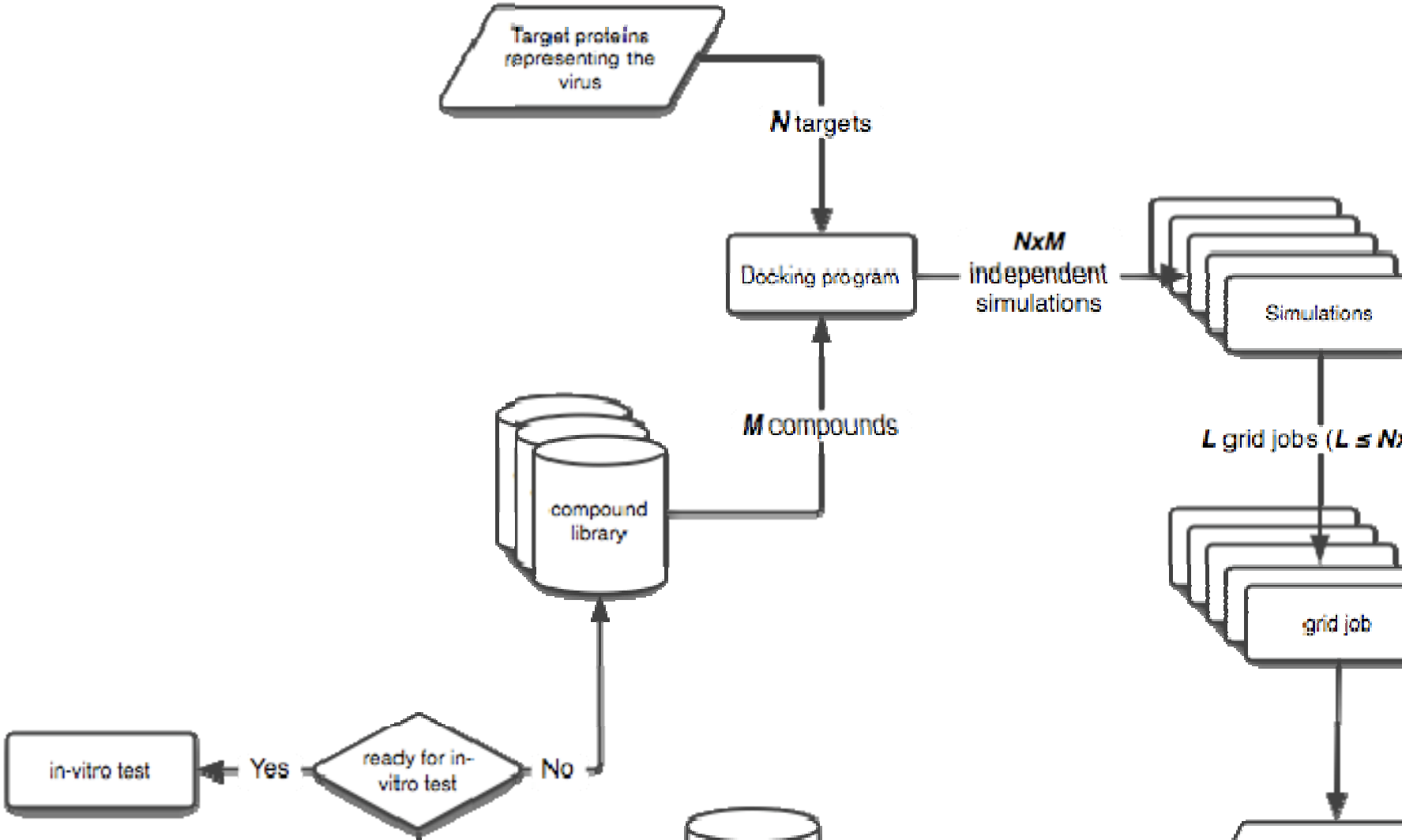
High-throughput screening using the Grid

- large-scale and on-demand resources: shorten the response time to emergency disease

Fully distributable grid jobs (parametric-sweep type)

Collective analysis on distributed results

Interactive pipeline (run → analysis → run)



Application architecture

integrating EGEE grid services

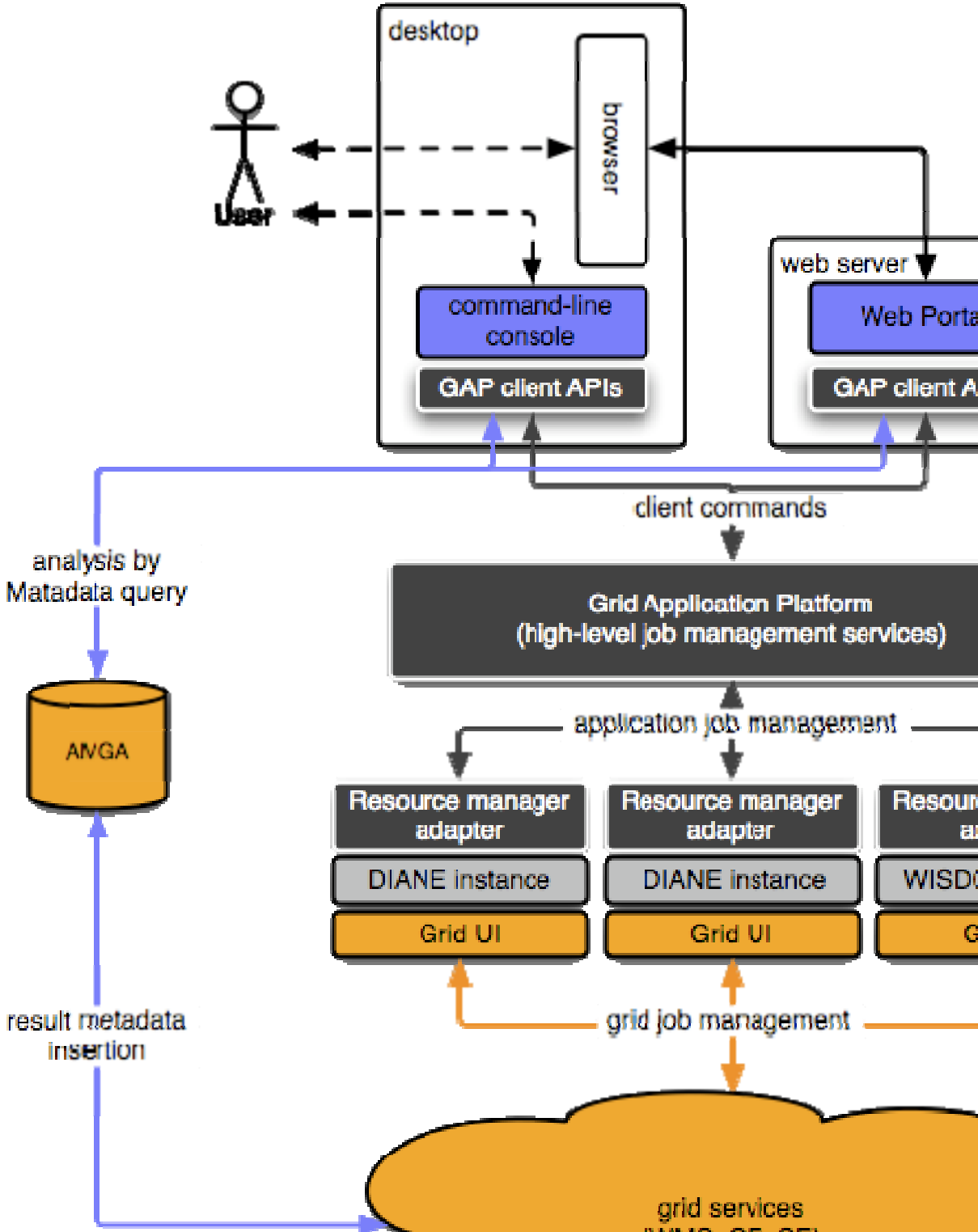
- AMGA, UI, WMS, SE, CE

Client

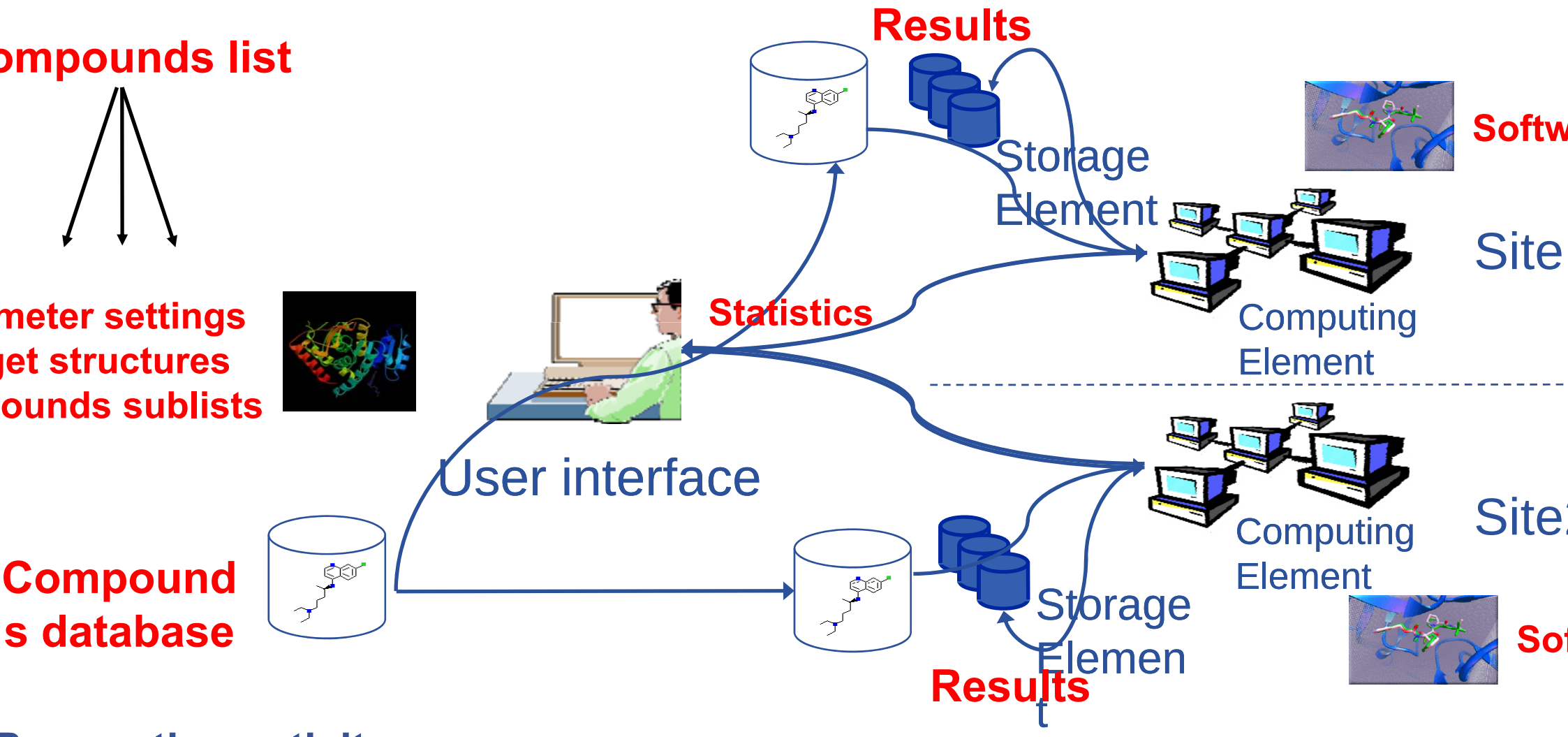
- light-weight, portable and application-oriented interfaces built on top of a same set of Java APIs

application oriented job management

- integrating existing approaches (DIANE, WISDOM)
- application platform for team



Implementation details



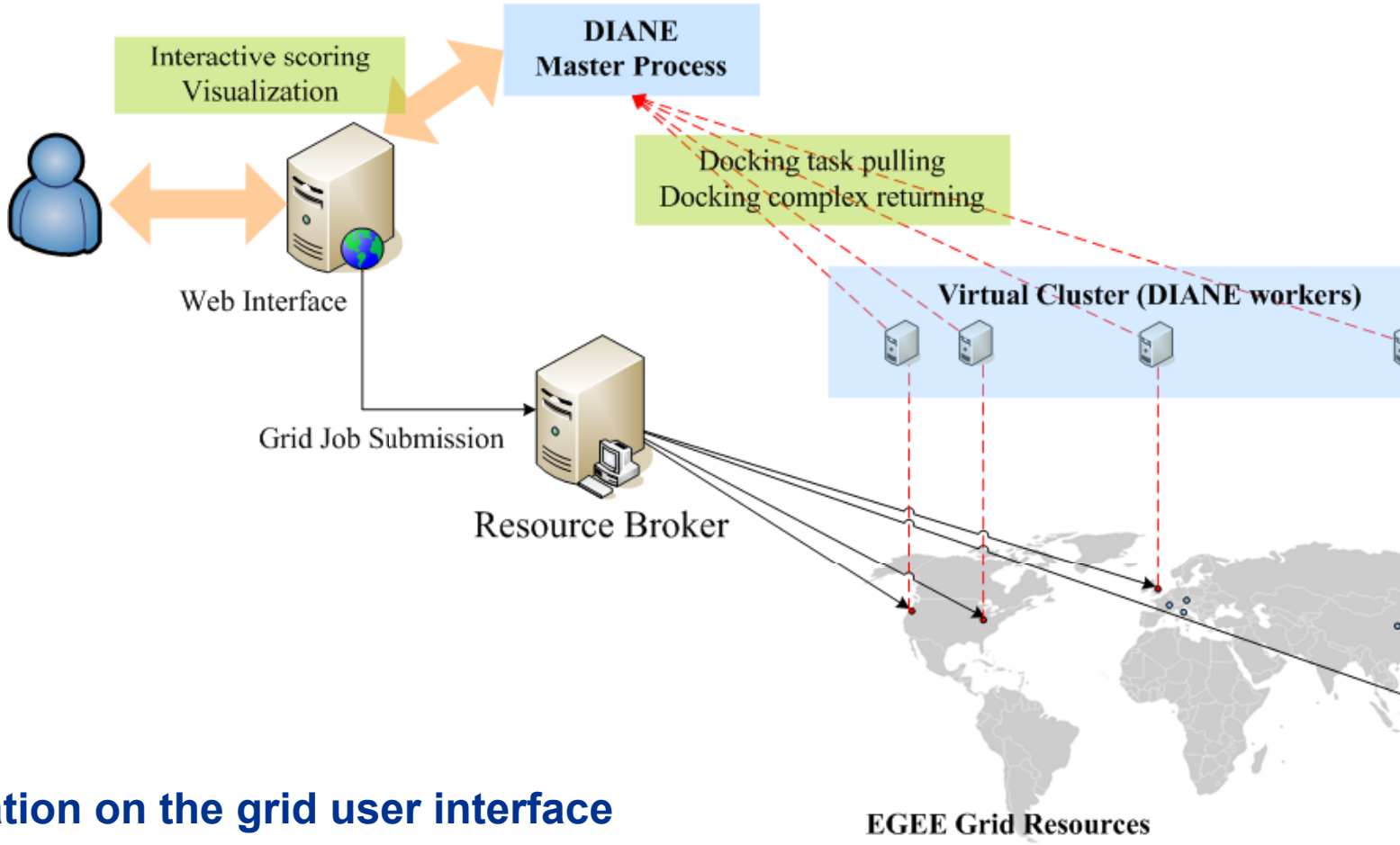
Preparation activity

1. docking program deployment on grid computing elements
2. compound library deployment on grid storage elements

Runtime activity

1. splitting and wrapping simulations in grid jobs
2. job management automation (submission, book-keeping through the grid WMS)

http://cern.ch/diane



Preparation activity

1. DIANE framework installation on the grid user interface

Runtime activity

1. a task queue holding the docking simulations (the DIANE master runs on the grid user interfa
2. grid job submission through the grid WMS (jobs for installing DIANE workers on the grid work node)
3. establishing extra communication channel between DIANE workers and the DIANE master
4. DIANE workers run the following cycle until the task queue is emptied

simulation from the task queue ⇒ run simulation ⇒ return simulation result to the master

	WISDOM	DIANE
Total number of completed dockings	$2 * 10^6$	308,585
Estimated duration on 1 CPU	88.3 years	16.7 years
Duration of the experience	6 weeks	4 weeks
Cumulative number of the Grid jobs	54,000	2580
Max. number of concurrent CPUs	2,000	240
Crunching rate	912	203
Approximated distribution efficiency	46%	84%
Approximated throughput	2 sec/docking	10 sec./ docking

First data challenge for avian flu drug analysis

- **8 targets vs. 300,000 compounds (2,400,000 simulations)**

WISDOM wins in the throughput by submitting huge amount o grid jobs

DIANE wins in the efficiency by controlling the utilization of th

User's analysis model

- giving me the compounds with docking energy lower than -9.0 eV

with simulation results distributed on the Grid

- user has to download all results from the SEs (~ 10^{12} bytes)
- parsing the results one by one

just a simple query on metadata:

- SELECT compound_name IN simulation_results WHERE docking_energy < -9.0

AMGA aims to host the metadata of the simulation results

- essential attributes of simulation results are extracted at simulation runtime and inserted into AMGA
- user performs data analysis by AMGA queries

GAP: Grid Application Platform developed by ASGC

Historically it intended to integrate distributed resources and services to form a large bioinformatic computing facility in Taiwan.

Now it's grid enabled and supporting the EGEE grid

Features:

- 3-tier architecture implementing the Model-View-Controller pattern in developing grid applications
  - with the flexibility to adopt new technology in each tier
- service oriented approach fully implemented in Java

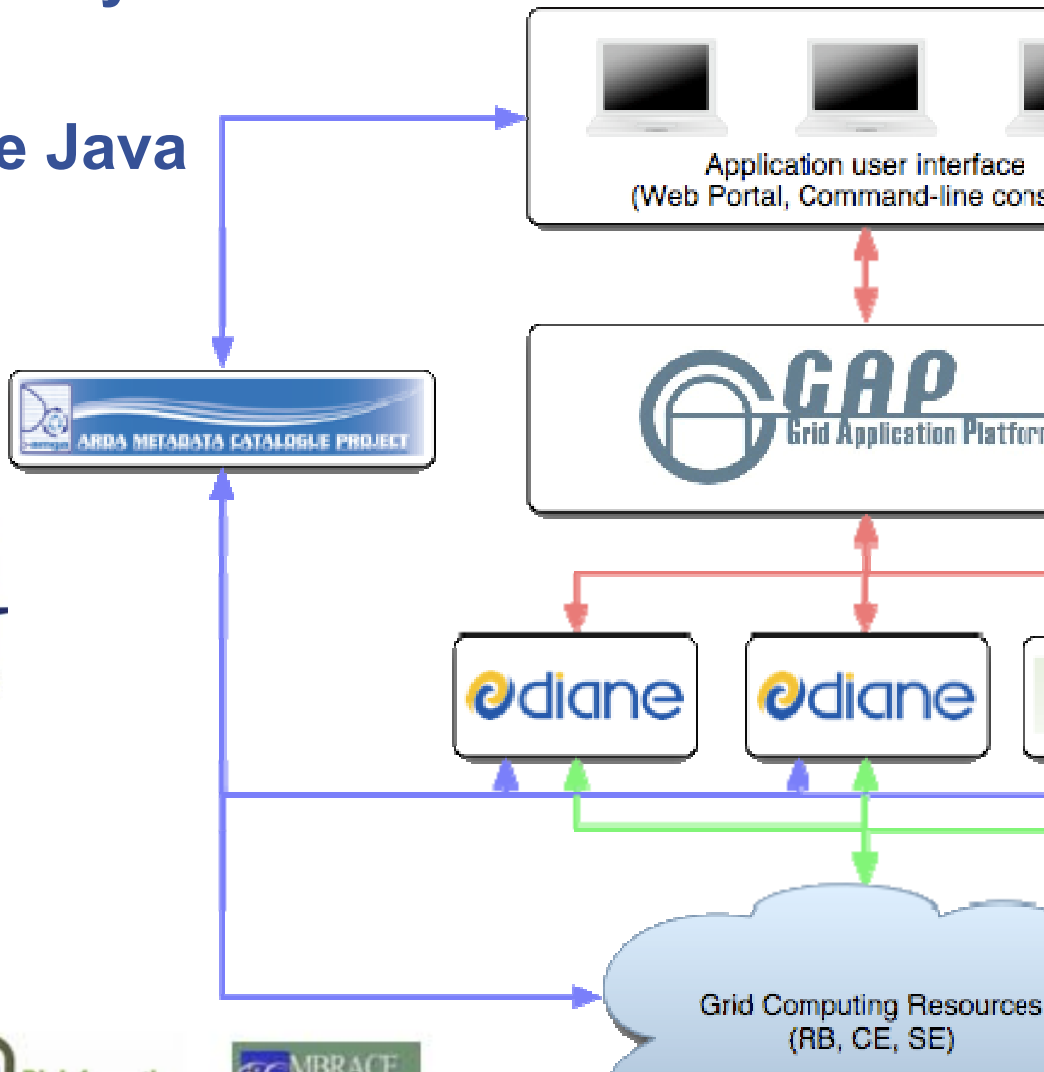
multiple *DIANE* instances deployed on multiple grid UIs (distributed on As sites)

new *WebServices* interface of *WISDOM*

AMGA stores the essential data from the simulation results

GAP manages the simulations performed by distributed *DIANE* instances a *WISDOM*

GAP provides a set of client APIs in pure Java



KiSTi Korea Institute of Science and Technology Information www.kisti.re.kr

THE UNIVERSITY OF MELBOURNE

National Grid NG SINGAPORE

Academia Sinica Genomics Research Center

IN2P3 eGee

User interface

the main user interface

based on BeanShell Java interpreter

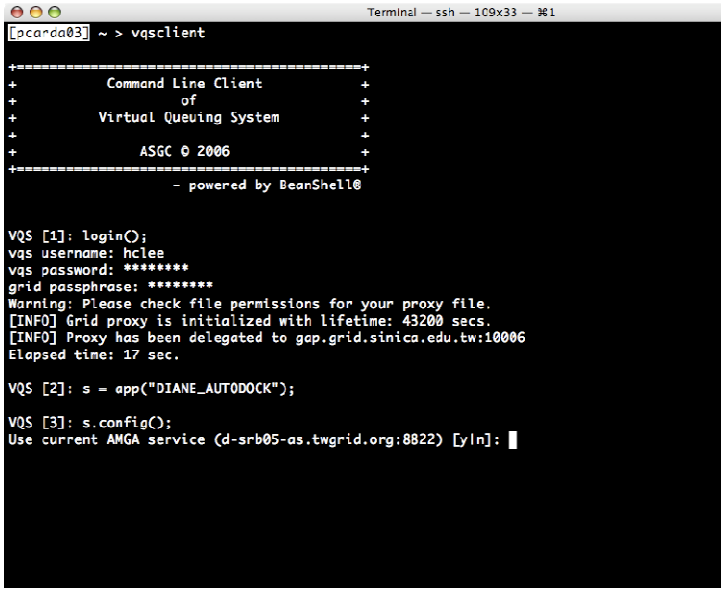
a set of BeanShell scripts wraps the client APIs

user benefits by

- application-oriented commands in configure and run applications
- an interactive environment for building/testing development with the client APIs

```
login();
```

- login the Grid Application Platform
- initiate and delegate the grid proxy




```
s = app("DIANE_AUTODOCK");
```

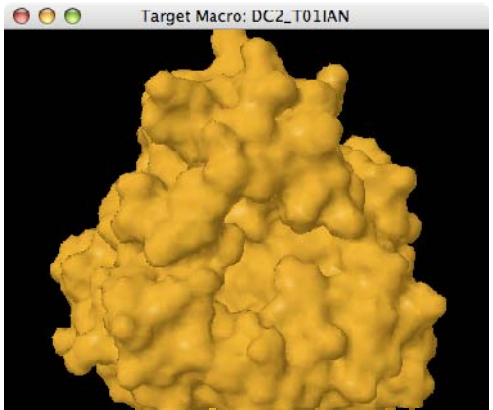
- load an application instance

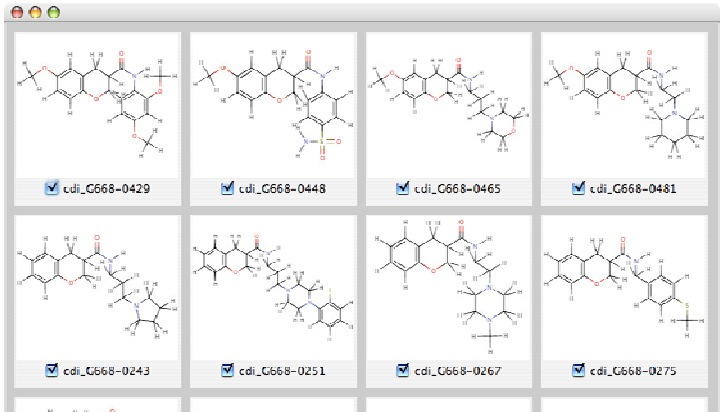
```
s.loadJobs(jobs("name","=","production"));
```

- bind historical production jobs with the application

```
s.config();
```

- configure the application in an interactive and graphic way



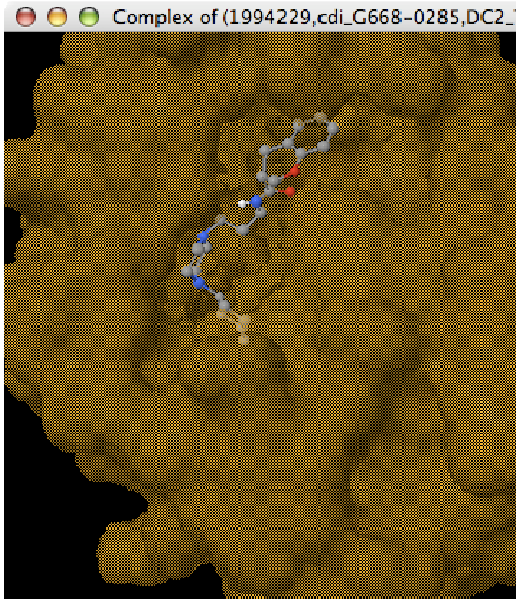



`s.run();`

- generate and submit application jobs (high-level jobs)

`s.progress();`

- check the overall progress of the simulation

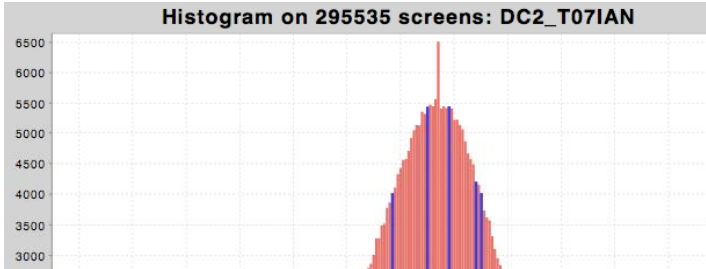


`s.listSimulations();`

- list all docking simulations

`s.viewComplex(<sim_id>);`

- online visualization of the docking result of the given simulation id



`s.histogram([controls]);`

```
s.downloadSimulations();
```

- **downloads all the simulation outputs (from many distributed storages) to local disk**

```
screens = s.getScreens([energy_threshold]);
```

- **gets the screen results with a given energy threshold**

used compound list

new simulation

configure

analysis

simula

`jobs = s.getJobs();`

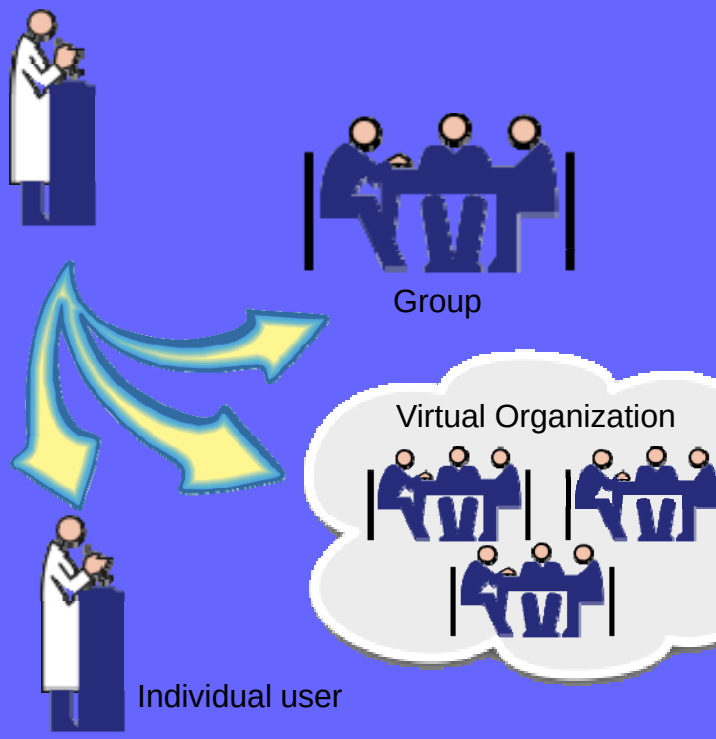
- gets the job list created by (or associated with) the applicati
- use `j = jobs.get(0);` to get the first job in the list

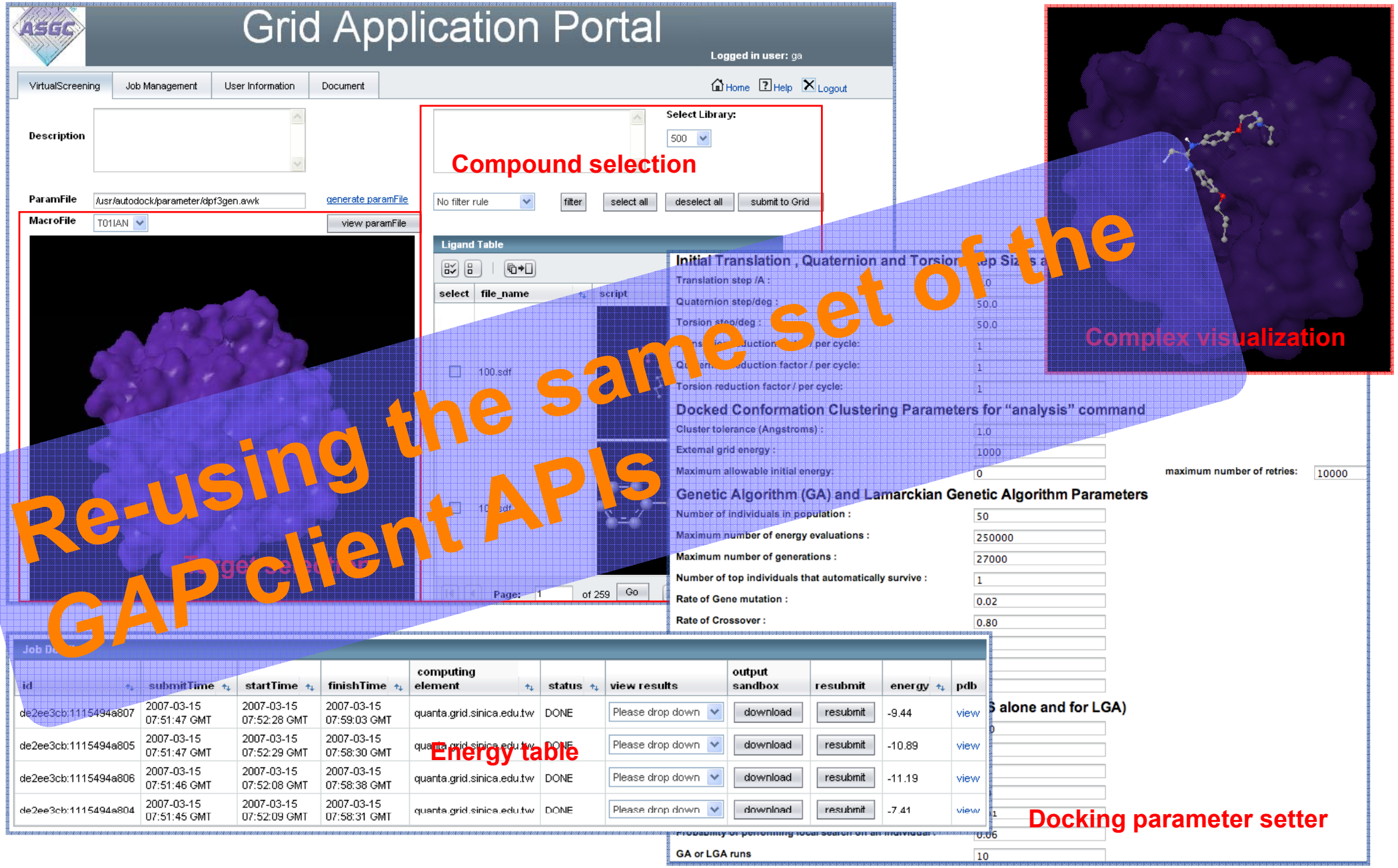
more Job operations available ... try online help:

`apidoc(j);`

ollaboration by sharing jobs

```
it = jobs.iterator();
while ( it.hasNext() ) {
    j = it.next();
    j.addShareWithUser("hclee");
    j.addShareWithVO("biomed");
}
```





id	submitTime	startTime	finishTime	computing element	status	view results	output sandbox	resubmit	energy	pdb
de2ee3cb:1115494a807	2007-03-15 07:51:47 GMT	2007-03-15 07:52:28 GMT	2007-03-15 07:59:03 GMT	quanta.grid.sinica.edu.tw	DONE	Please drop down	download	resubmit	-9.44	view
de2ee3cb:1115494a805	2007-03-15 07:51:47 GMT	2007-03-15 07:52:29 GMT	2007-03-15 07:58:30 GMT	quanta.grid.sinica.edu.tw	DONE	Please drop down	download	resubmit	-10.89	view
de2ee3cb:1115494a806	2007-03-15 07:51:46 GMT	2007-03-15 07:52:08 GMT	2007-03-15 07:58:38 GMT	quanta.grid.sinica.edu.tw	DONE	Please drop down	download	resubmit	-11.19	view
de2ee3cb:1115494a804	2007-03-15 07:51:45 GMT	2007-03-15 07:52:09 GMT	2007-03-15 07:58:31 GMT	quanta.grid.sinica.edu.tw	DONE	Please drop down	download	resubmit	-7.41	view