

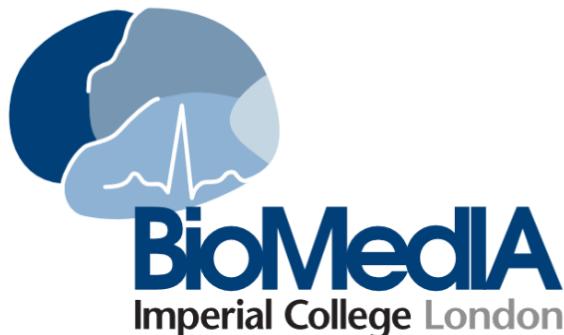


OPEN MODEL EXPERIMENT

Mathieu Leclaire

Romain Reuillon

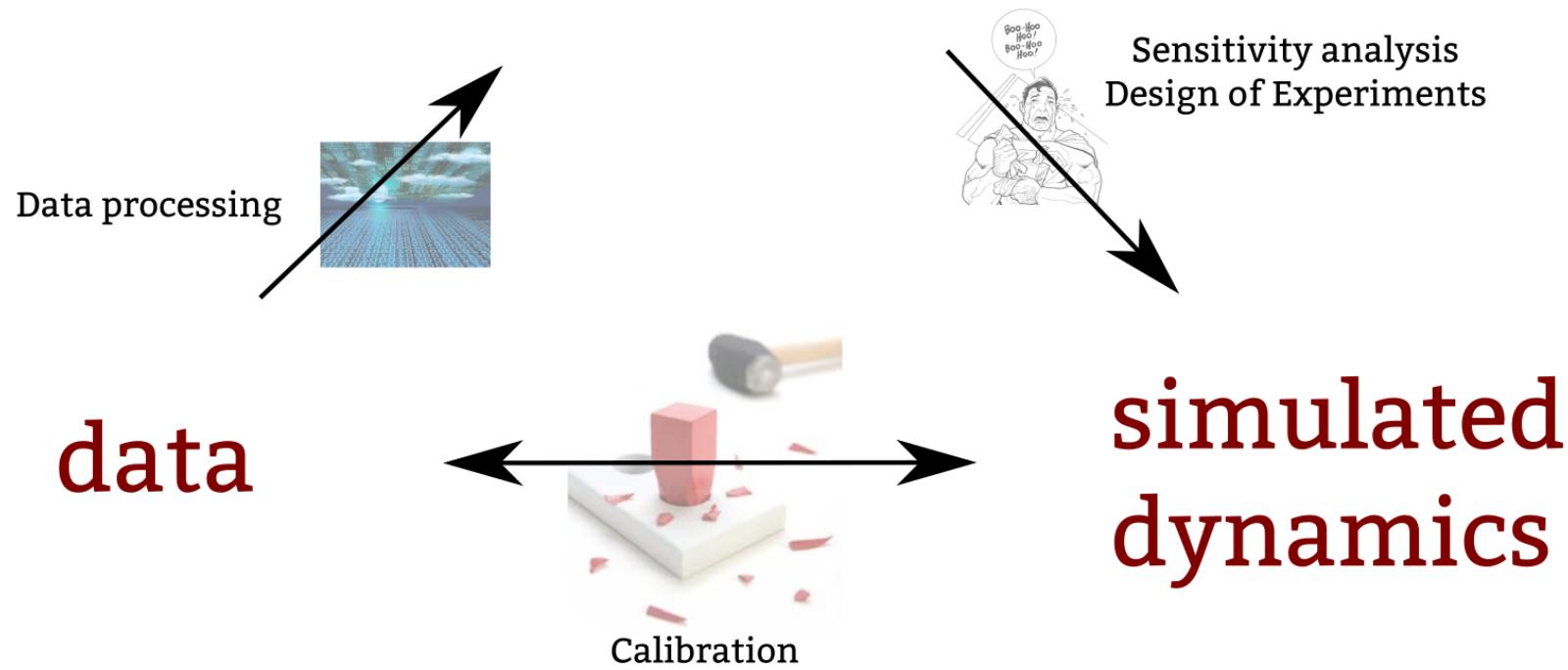
Jonathan Passerat-Palmbach



European Research Council
Established by the European Commission



model



Model construction steps

Many scientists use **naturally parallel methods** daily:

- Data reconstruction
- Parameter estimation
- Sensitivity analysis
- Optimisation
- Replication
- ...

Execution on the **same program** with **different parameters** and/or **datasets**.

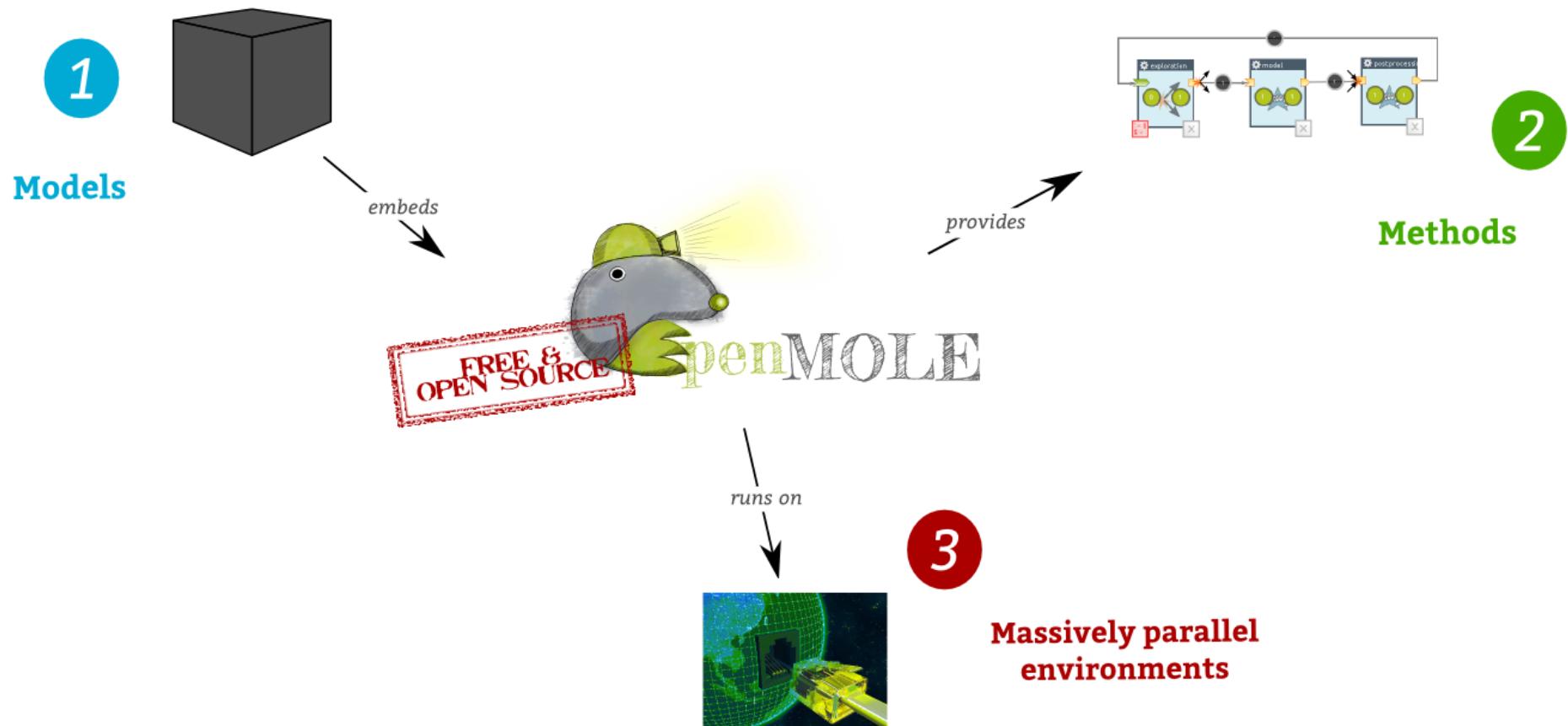


These methods are **time consuming**, but generally executed in a sequential manner.



OpenMOLE fills the gap. It provides an easy way to **describe** and **distribute** naturally parallel processes.

OPENMOLE IN 3 STEPS



EMBED YOUR MODEL AS A BLACK BOX

C
R
C++
Java
Scala
Scilab
Octave
Python
Netlogo
...



ZERO DEPLOYMENT APPROACH

- The code is the user's code, not a web service
- User code is automatically deployed at runtime
- Ships to remote environment
- No prior knowledge of remote environment needed
- No installation required on any machine

PORt (ALMOSt) ANY PROGRAM TO THE GRID IN 3 SIMPLE STEPS

- Archive it with CARE* <=> execute it on linux
- Write your OpenMOLE workflow
- Click the run button

* <http://reproducible.io>

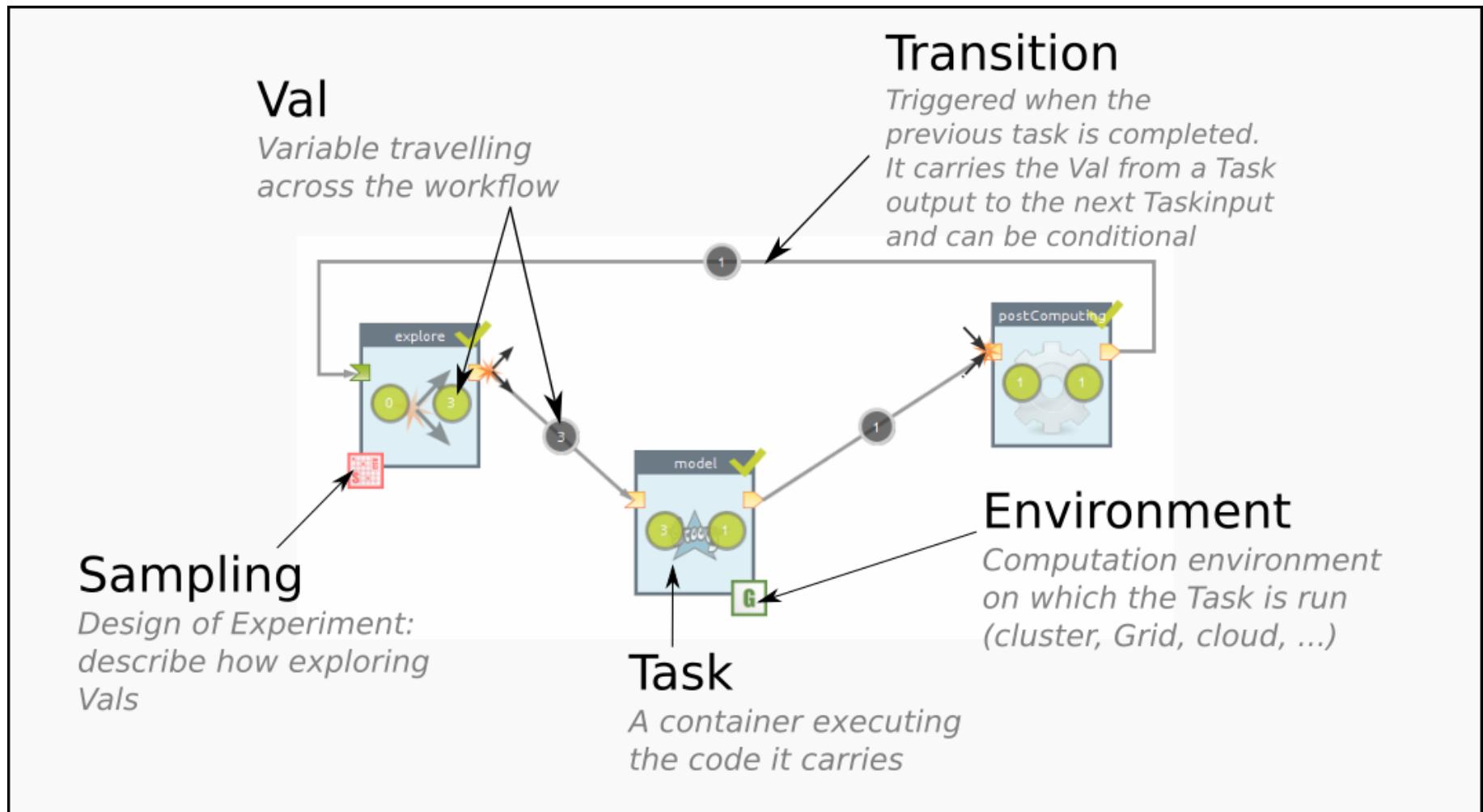
PACKAGING AN APPLICATION WITH CARE

Applications have dependencies:

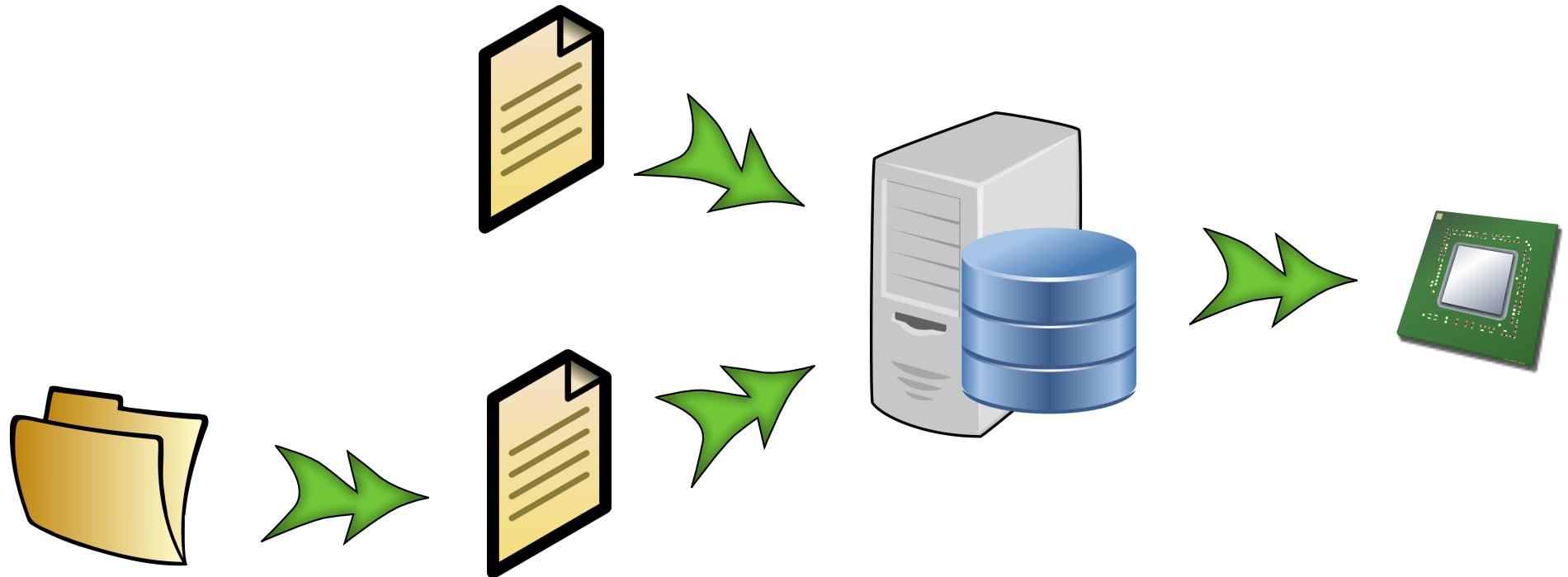
- Shared libraries
- Packages (Python, R, ...)
- Low level system calls
- Environment variables
- ...

Capture these dependencies and **transfer** along with the application from Linux to Linux

NATURALLY PARALLEL FORMALISM TO DESIGN EXPERIMENTS: A WORKFLOW

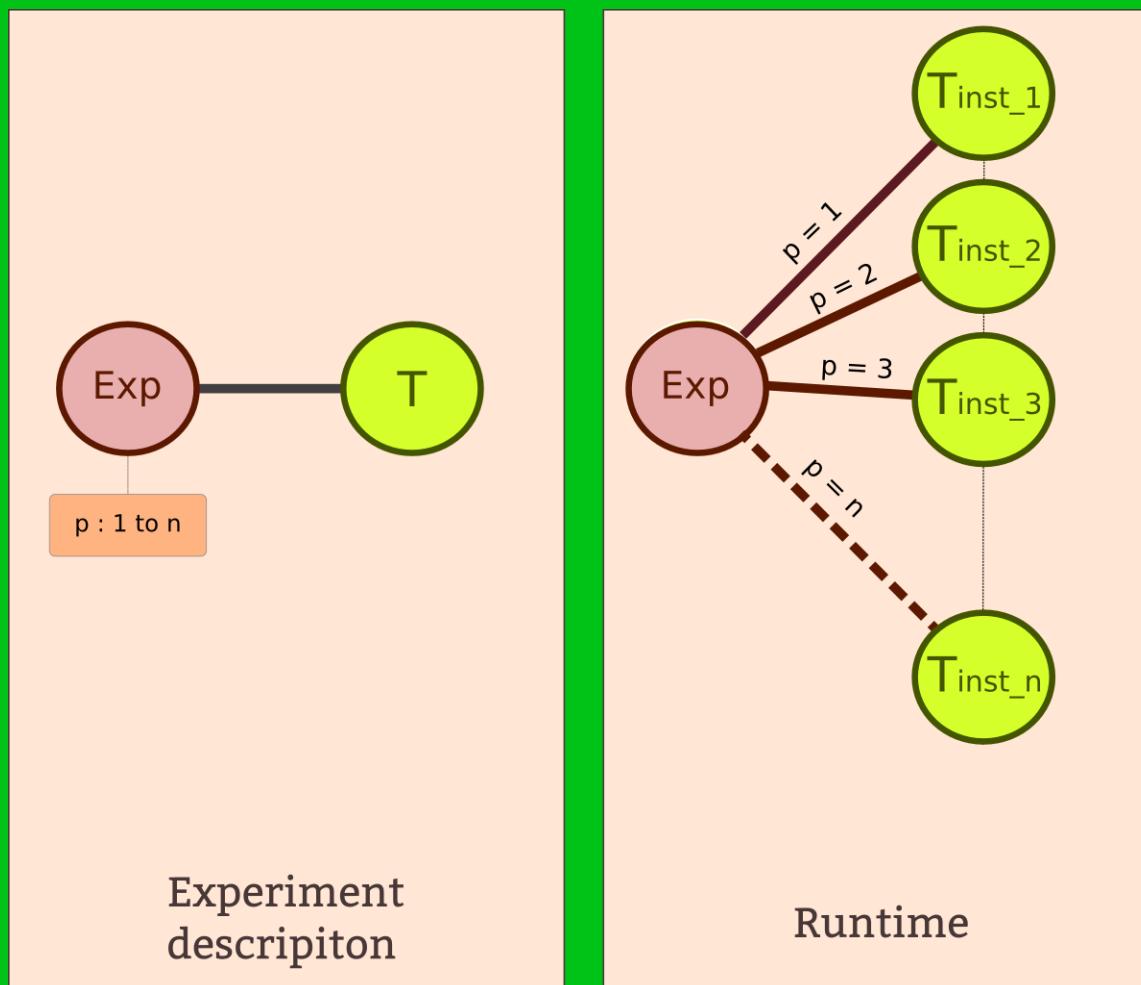


AUTOMATIC DATA TRANSFERS AND REPLICA MANAGEMENT



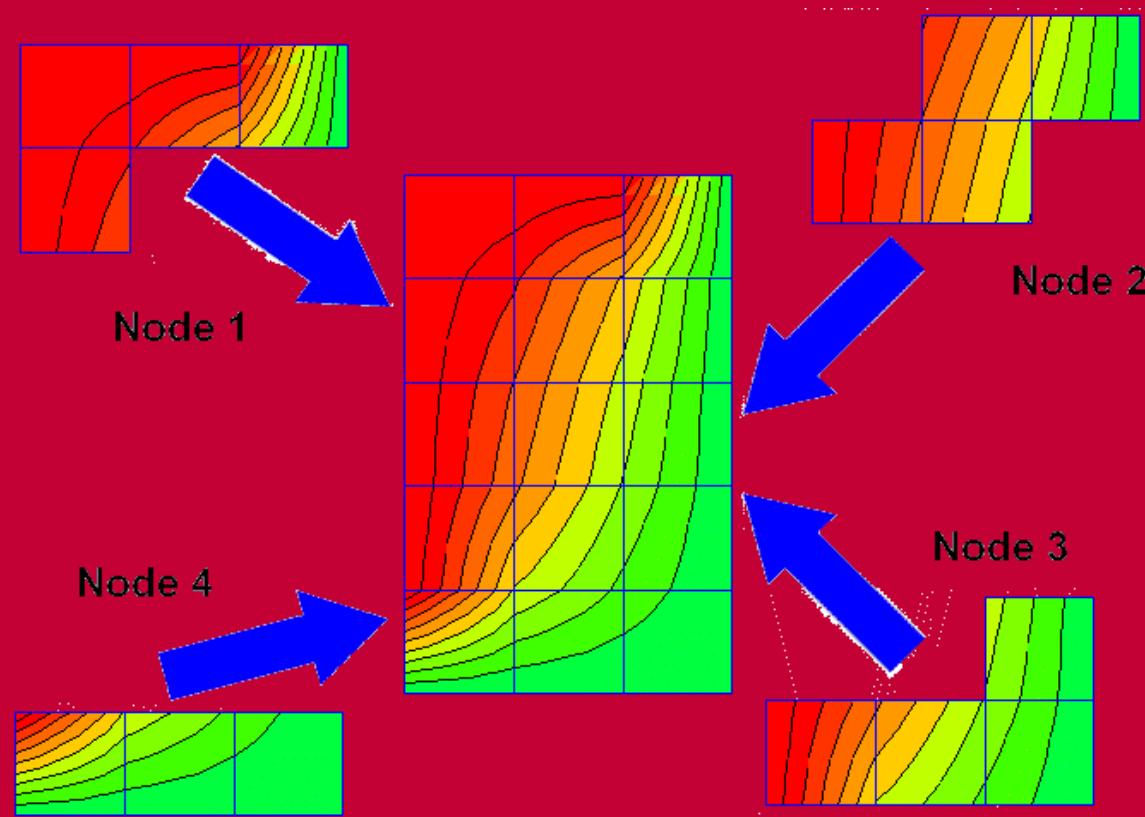
Files and folders transfers are **handled transparently** by
OpenMOLE

WHAT OPENMOLE DOES



Data parallelism

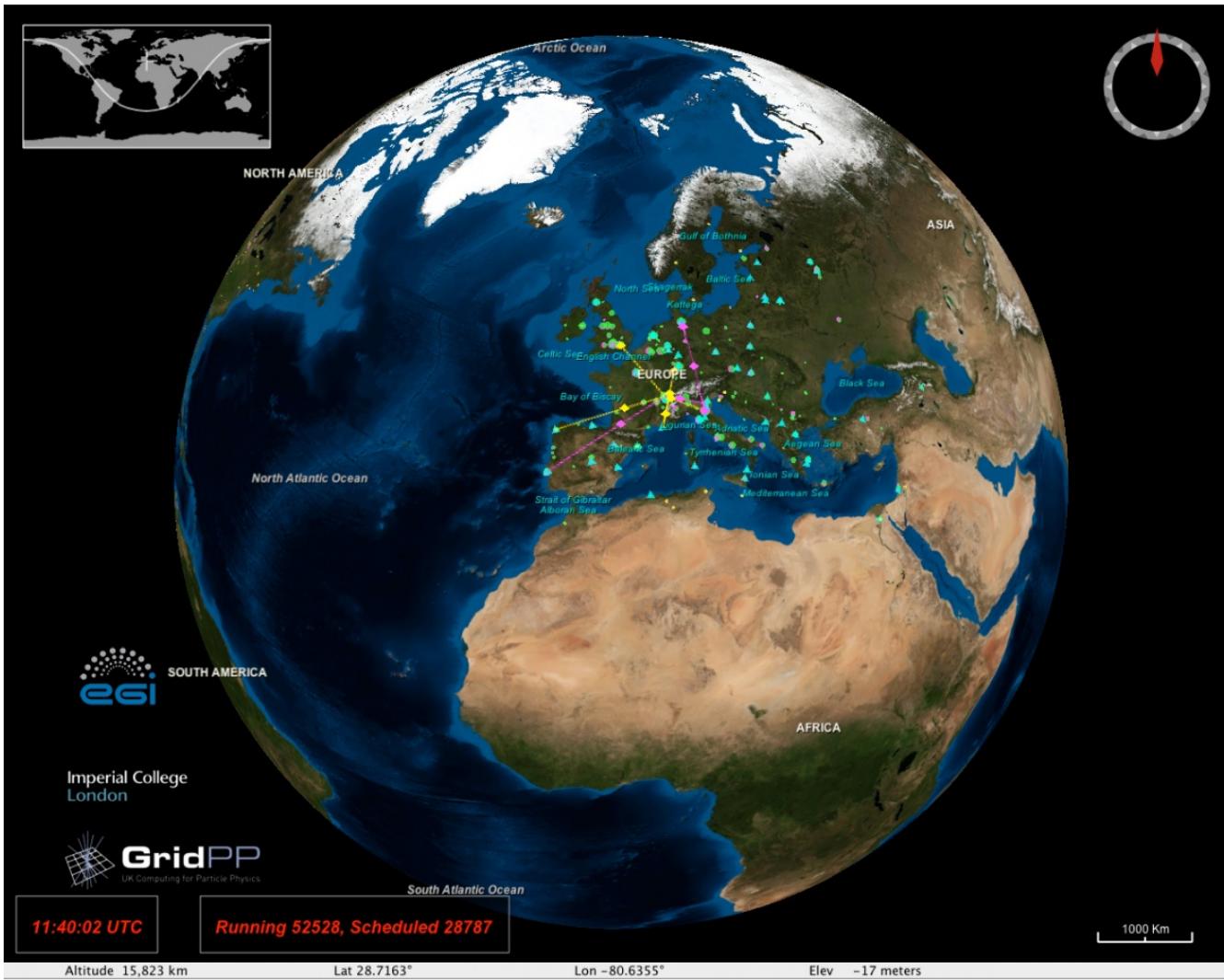
WHAT OPENMOLE DOES NOT



Parallelisation by message / Task Parallelism

Example: Spark, MPI, ...

POWERED BY EGI (AND OTHERS)



A WORKFLOW

```
val i    = Val[Double]
val res = Val[Double]

val exploration = ExplorationTask ( i in (0.0 to 10.0 by 1.0) )

val model =
  ScalaTask ("val res = i * 2") set (
    inputs  += i,
    outputs += (i, res)
  )

val env = LocalEnvironment(5)

val ex = exploration -< (model on env) start
```

THE SAME WORKFLOW ON THE GRID !

```
val i    = Val[Double]
val res = Val[Double]

val exploration = ExplorationTask ( i in (0.0 to 10.0 by 1.0) )

val model =
  ScalaTask ("val res = i * 2") set (
    inputs  += i,
    outputs += (i, res)
  )

val env = EGIEvironment("biomed")

val ex = exploration -< (model on env) start
```

Switching to the **Grid Environment** is so easy !

WEB APPLICATION

The screenshot shows a web-based interface for managing and executing simulations. At the top, there's a browser header with tabs for 'about:sessionrestore' and 'https://localhost:46447/'. The main area has a dark theme with blue and white UI elements.

File Browser: On the left, there's a sidebar with a 'File' dropdown menu and buttons for 'Home', 'Fire in NetLogo', 'Explore.oms', 'Fire.nlogo', and 'README.md'. A modal window is open over the sidebar, showing the contents of 'Explore.oms'.

Modal Window (Explore.oms): This window is titled 'Executions' and lists a single entry: 'biomed'. It shows various metrics: 22 (80,87MB) inputs, 35 outputs, 401 errors, 69 warnings, 54 errors, and 0 errors. There are buttons for 'Play' and 'Close'.

Code Editor: Below the modal, a large code editor window displays the 'Explore.oms' file content. The code is as follows:

```
12  List(
13    "random-seed ${seed}",
14    "setup",
15    "while [any? turtles] [go]"
16  )
17
18 val fireTask =
19   NetLogo5Task(workDirectory / "Fire.nlogo", cmds) set (
20     inputs += seed,
21     outputs += (seed, density),
22     netLogoInputs += (density, "density"),
23     netLogoOutputs += ("burned-trees", burned)
24   )
25
26 val csvHook = AppendToCSVFileHook(workDirectory / "result.csv", density, burned, seed)
27
28 val grid = EGIEnvironment("biomed", name = "biomed")
29
30 exploration -> (fireTask on grid hook csvHook)
31
```

CONSOLE INTERFACE

```
[reuillon:~] $ openmole -c
Picked up JAVA_TOOL_OPTIONS: -javaagent:/usr/share/java/jayatanaag.jar

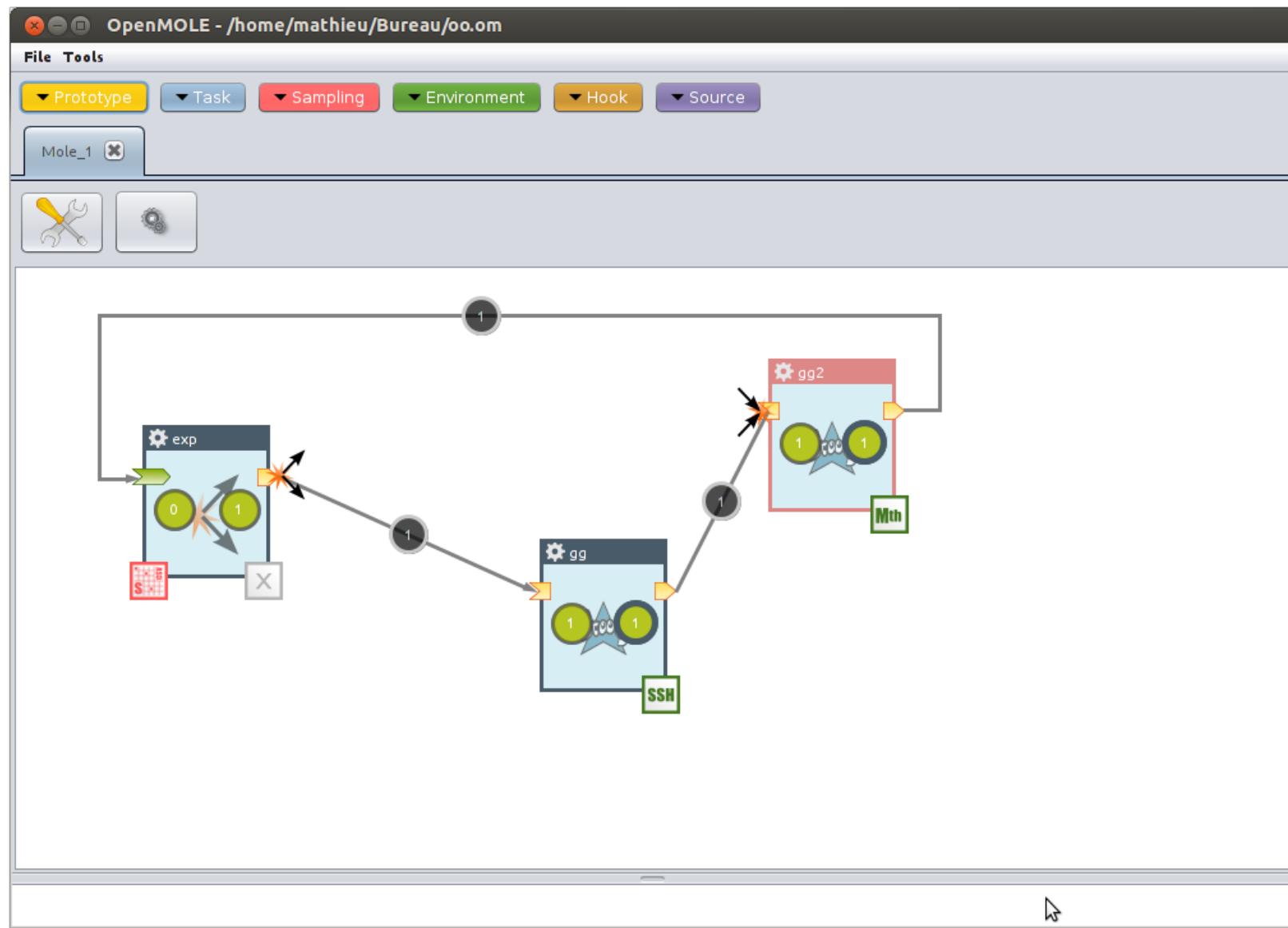
/ _\ \_ _ / _\ \_ _ / _\ \_ / _\ \_ / _\ \_ / _\ \_ / _\ \_ / _\ \_
| [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] |
| [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] | [ ] |
\ __/| .__/ \ __| | | | | | | | | | | | | | | | | | | | | |
|_|

(Type :q to quit)
Enter your OpenMOLE password (for preferences encryption): *****
OpenMOLE>
```

TOWARDS A MULTI-USER WEB PLATFORM



TOWARDS GRAPHICAL WORKFLOWS



MARKET PLACE

- Full workflows describing various experiments from **real cases**.
- It contains scripts and all the required resources. Just **download** and **run** it!
- **Community:** Users can **upload** their own workflow through **github**

MARKET PLACE

Market place

Filter

Pi Computation

Stochastic Simulation

Random Forest

Stochastic Machine Learning Native Code Data Python

Hello World in R

R Data Native Code

Fire in NetLogo

NetLogo Stochastic Simulation

Hello World in Java

Java

Calibration of Ants

Download

NetLogo Genetic Algorithm Simulation Calibration

Calibration of a NetLogo model

This market entry calibrates the [Ants](#) NetLogo model with an Evolutionary/Genetic Algorithm (EA/GA) in OpenMOLE. It proposes 2 aproach, the first one uses a generational NSGA2 which is very classical. The second one is better fit for distributed computing and uses an [island model](#) distribution strategy.

Hello with OpenMOLE plugin

Scala Java Plugin

SimpopLocal

Stochastic Simulation Genetic Algorithm Scala Calibration

Metamimetic Networks

Stochastic Simulation NetLogo

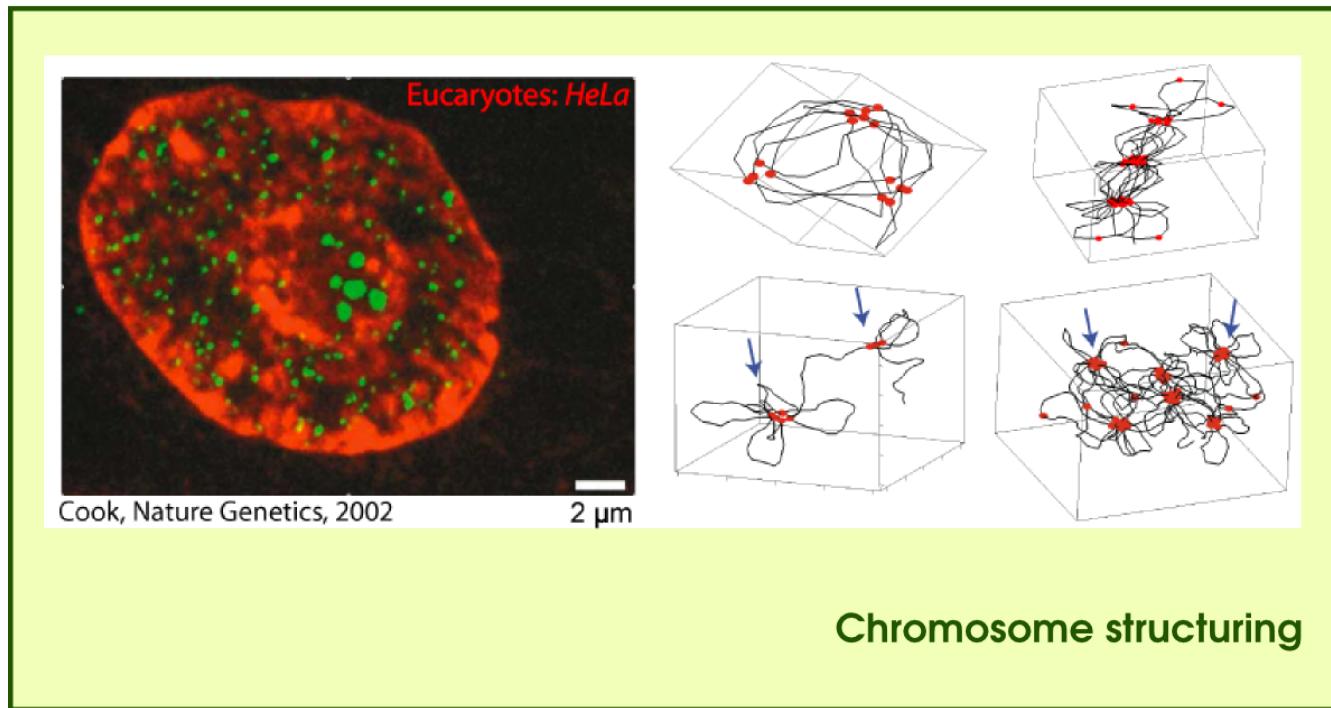
Close

OPENMOLE THROUGH 4 EXAMPLES

OpenMOLE is neither dedicated to a **scientific field** nor to a **language**

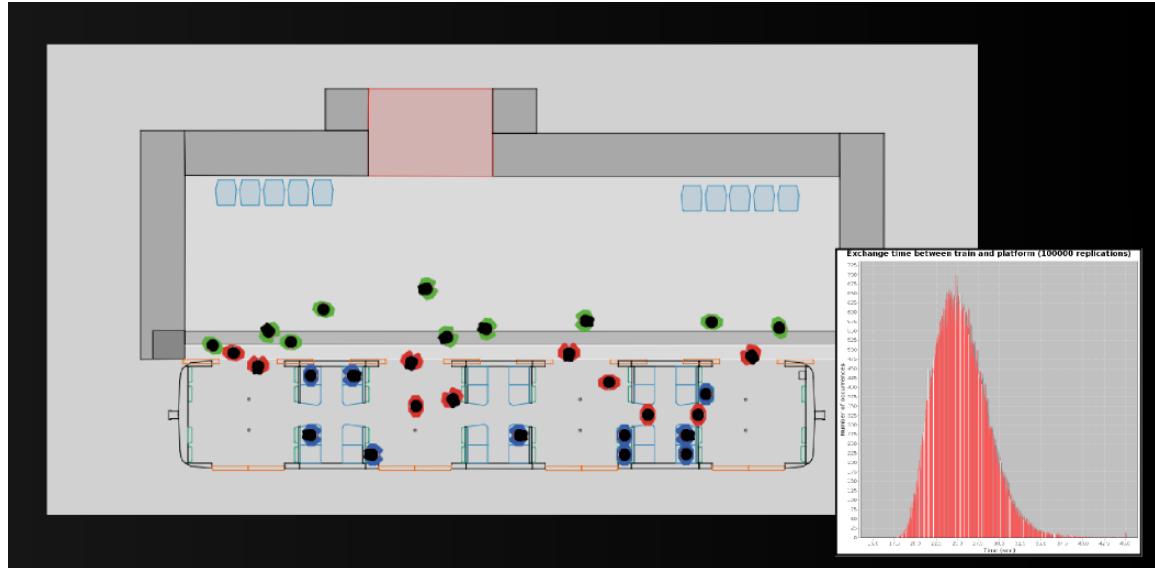
- Chromosome structuring: **Neuro Sciences, C++**
- The SimTRAP project: **Social Sciences, Netlogo**
- The SimPOP project: **Geography, Scala**
- The BioEmergence project: **Biology, C**

CHROMOSOME STRUCTURING



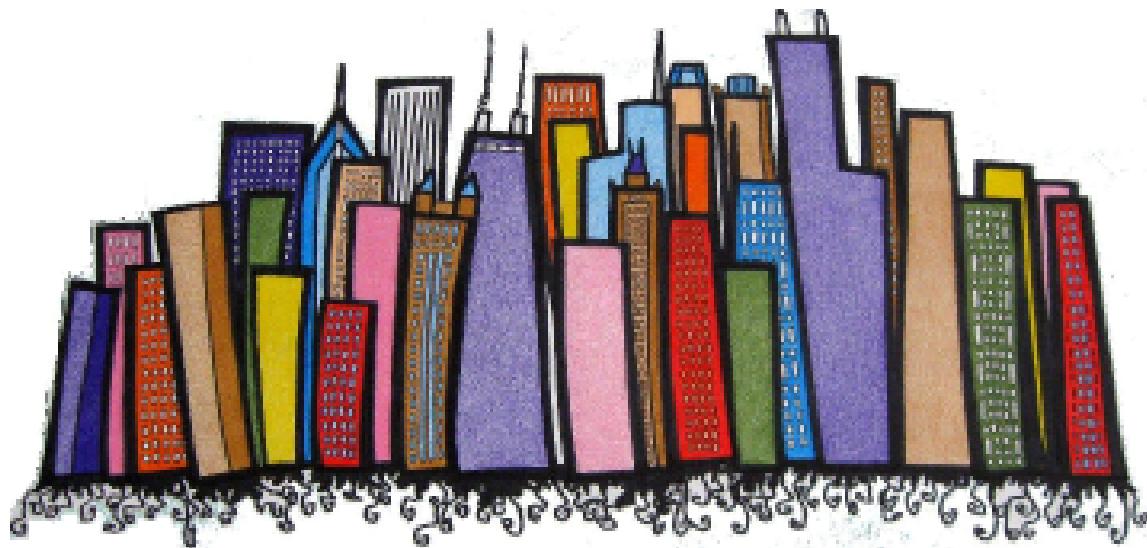
C++ model
2 days per
simulation for
1,600 simulations
= **8.5 years CPU**
time

THE SIMTRAP PROJECT



Netlogo model
5 mins per simulation
for 100,000 simulations
= 1 year CPU time

THE SIMPOP PROJECT



Scala model
2 secs per simulation
for
500,000,000 simulations
= **30 years CPU time**

THE BIOEMERGENCES PROJECT

081104a

BIOEMERGENCES
EMMANUEL FAURE (LOG OUT)

Wishlist

Experiment

081104a SP5
22
Nadine Peyrieras
DA.RE
transgenesis
512x512x151 px, 260 t
670x670x198 μ every 02 m 25 s
from 4 h 00 m to 14 h 31 m

id 20520

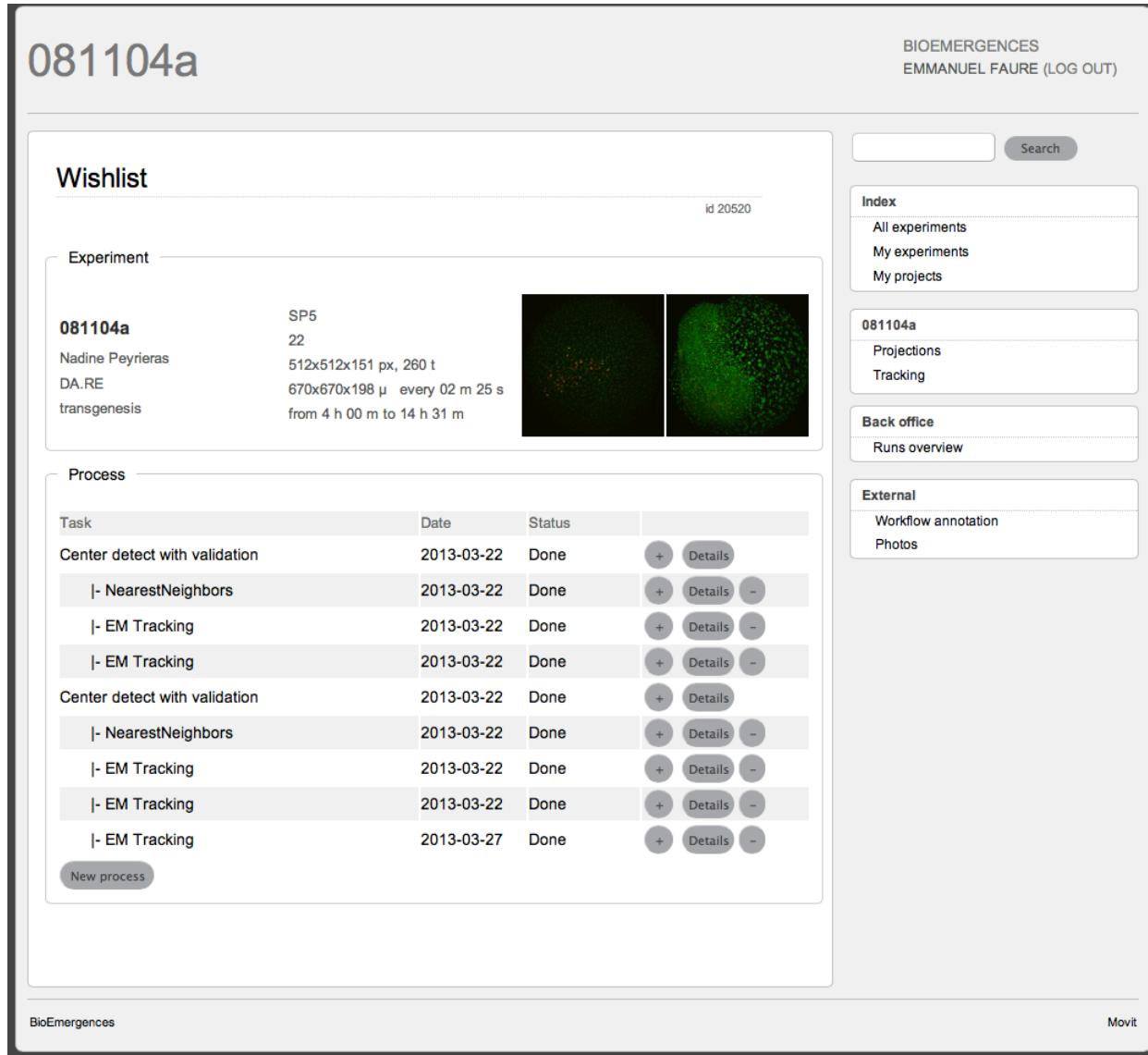
Process

Task	Date	Status	Actions
Center detect with validation	2013-03-22	Done	+ Details
- NearestNeighbors	2013-03-22	Done	+ Details -
- EM Tracking	2013-03-22	Done	+ Details -
- EM Tracking	2013-03-22	Done	+ Details -
Center detect with validation	2013-03-22	Done	+ Details
- NearestNeighbors	2013-03-22	Done	+ Details -
- EM Tracking	2013-03-22	Done	+ Details -
- EM Tracking	2013-03-22	Done	+ Details -
- EM Tracking	2013-03-27	Done	+ Details -

New process

BioEmergences

Movit



C model - Portal access
Daily production
10,000 hours / day

USEFUL LINKS

Documentation

www.openmole.org

Mailing-list

list.openmole.org

Development version

next.openmole.org

Source code

github.com/openmole

Market place

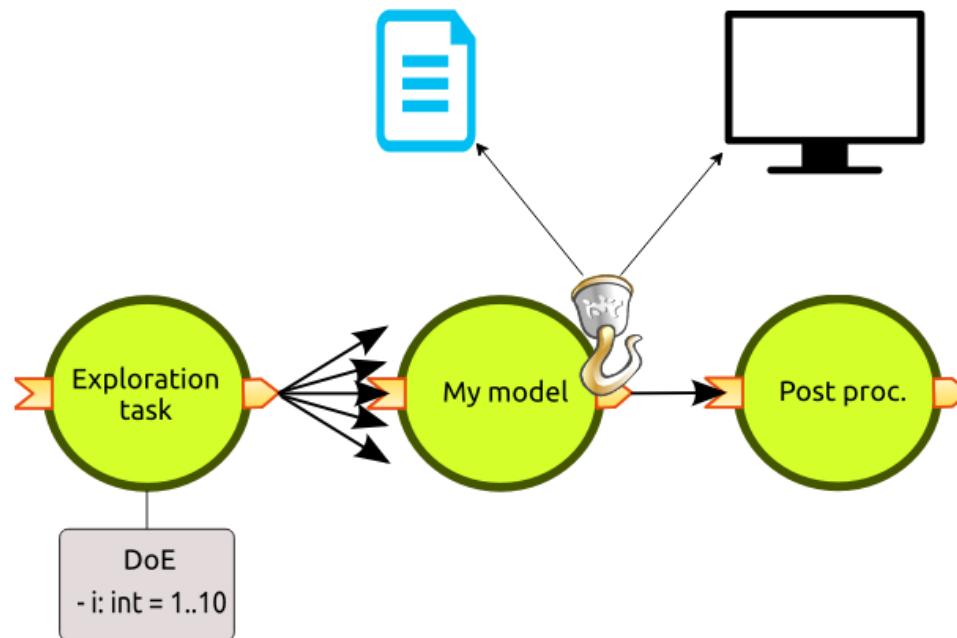
github.com/openmole-market

EXTRA NOTIONS

HOOKS

Tasks are mute

Hooks **extract content** from the dataflow



```
exploration -< (model on env) >- (average hook ToStringHook())
```

GROUPING

Short jobs don't cope well with distributed computing
Group **multiple tasks** in the **same batch job**

```
exploration -< (model on env by 10) >- (average hook ToStringHook())
```

THANKS!



romain.reuillon@iscpif.fr



mathieu.leclaire@iscpif.fr



j.passerat-palmbach@imperial.ac.uk