Hello, I'm Matthieu and I use Grid'5000

Matthieu Simonin

October 12, 2018

Hello web application

- Get an MVC framework (Rails, Django, ExpressJS, . . .)
- Create a skeleton project
- Start the application
- Connect to `http://localhost:<someport>`

Code example

```
gem install rails
rails new hello
bin/rails server
```

LAMP stack

Web developpers have LAMP stacks
Originally

- Linux (OS)
- Apache (Web server)
- MySQL (Database)
- PHP (High level language)

Benefits:

- Higher level of abstraction
- Developper focus on the application logic

LAMP stack

Questions

- Is there a LAMP stack for experimenters ?
  Team Objectives:
  - Comfortable execution environments for large scale application
  - Comfortable developpements environments for programmers
  - As an experimenter I'd like to have :
    Comfortable experimentation environment

Grid'5000 is a partial answer ...

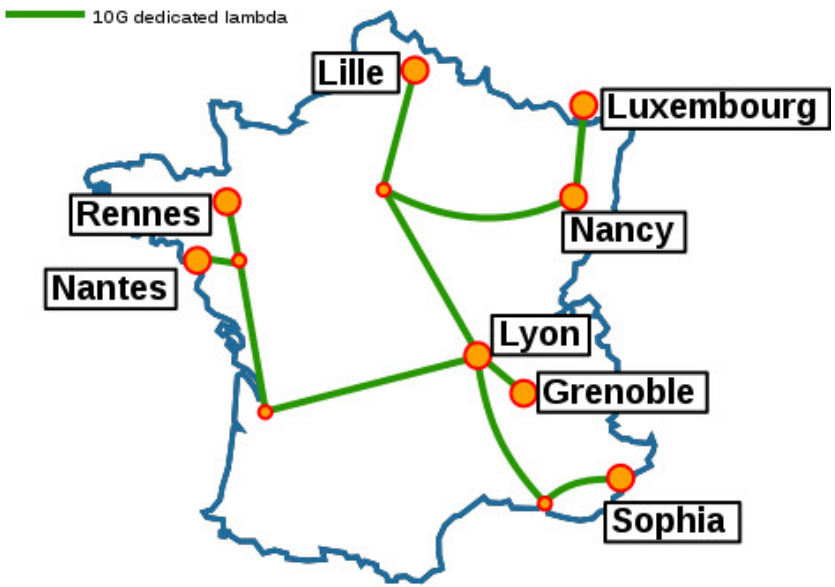
What is Grid'5000 anyway ?

What is Grid'5000 anyway ?

What is Grid'5000 anyway ?

Keywords definition

Grid'5000 is a unique platform that provides versatile Bare-metal as a service, fully reconfigurable infrastructure targetting reproducible research on distributed systems.



What is Grid'5000 anyway ?

Practical definition

- A large-scale testbed for distributed computing
  - 8 sites, 30 clusters, 850 physical machines, 8500 cores
  - Dedicated 10-Gpbs backbone network
  - Specific hardware (GPU, high speed network, storage)
- To meet your requirements:
  - You can find your special resources on G5k (GPU, highspeed network, storage . . .)
  - You can reconfigure some layers of G5K (OS, network, virtualization technologies ..)
    - Control your environment

What is Grid'5000 anyway ?

- 600 users and 100 publications per year
- Top 5 #publications from 2014 to early 2018 leveraging Grid'5000:
  - Avalon 77
  - Myriads 69
  - Ascola/Stack 45
  - Multispeech 44
  - Kerdata 36

Myriads on Grid'5000

- Simulations
- Model validation / invalidation
  - e.g energy
- Middleware validation
  - Workflow / Stream processing
  - IaaS / PaaS / FaaS (performance / security / green measures)
  - Fog/Edge emulation
- Complex stack deployment (e.g Openstack, Kubernetes, Kubernetes over Openstack over Kubernetes . . .)

Most of the Myriads' experimentation fits into Grid'5000

Caveats

- What Myriads do on Grid'5000 can be hard
  - uniqueness of the experiment
  - degree of control that is required
  - scaling the experiment
- But sometimes it's artificially hard
  - lack of general tools for doing experimentation
  - lack of common practice
  - experimenter solitude

My advice

Experimentation is a code

- Versionned (code + parameters . . .)
- High level language (e.g Python)
- Iteratively built
- Shareable (e.g pypi package)
- Used by someone else
  - Even your supervisor should be able to use it !
- Reasonnably over-engineered

LAMP stack for the experimenter

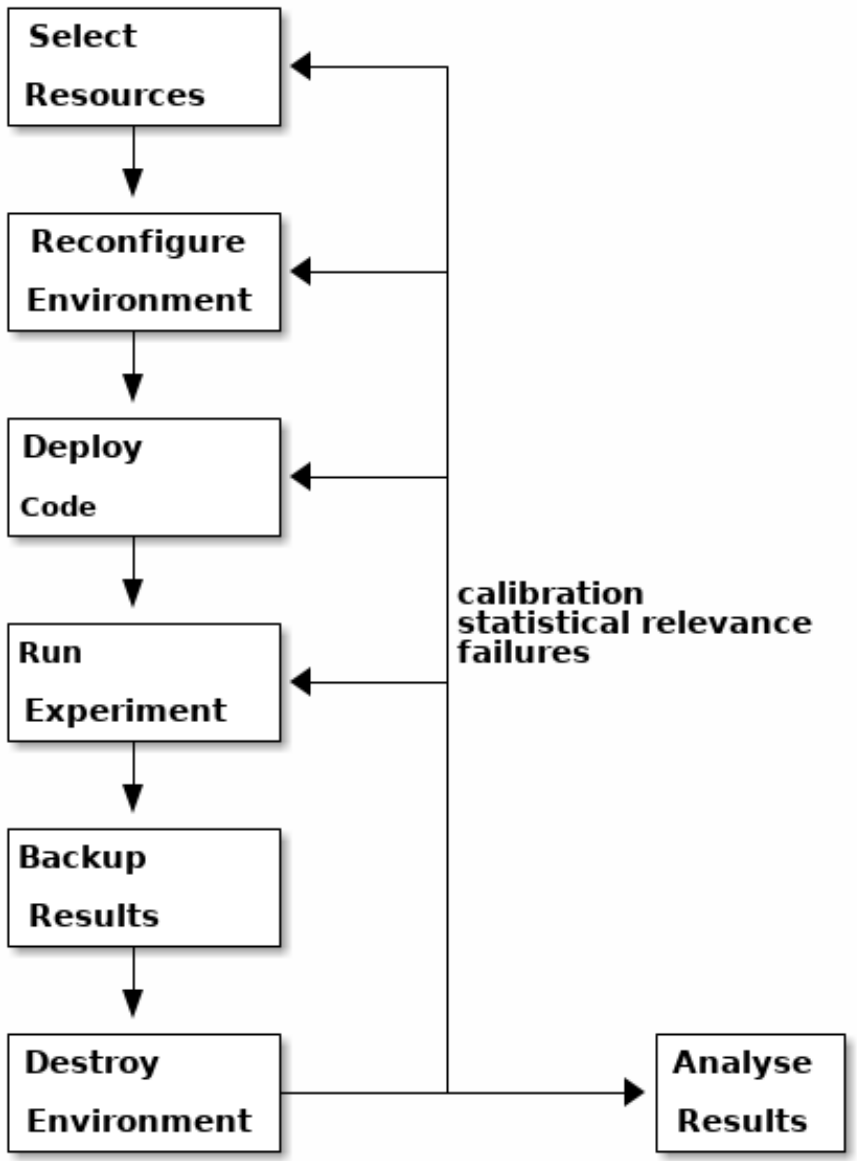
Questions

- Is there a LAMP stack for experimenters ?
  - What is the "Hello World" for Grid'5000 ?
  - What does / should it look like ?
- ~~What is Grid'5000 anyway ?~~

Grid'5000 is a partial answer . . .

Ronan-Alexandre Cherrueau, Matthieu Simonin, Alexandre van Kempen: EnosStack: A LAMP-like stack for the experimenter. INFOCOM Workshops 2018: 336-341

The experimenter workflow



The experimenter workflow

The workflow is repeated over and over for each combination of the input parameters

Exemple:

```
# 72 combinations (6 non zipped * 2 call_type * 6 bus)
campaign:
  test_case_1:
    nbr_clients: [1000, 2000, 4000, 6000, 8000, 10000]
    nbr_servers: [20  , 20  , 20  , 20  , 20  , 20]
    pause      : [0.1 , 0.2 , 0.4 , 0.6 , 0.8 , 1.0]
    nbr_calls  : [3000, 1500, 750 , 500 , 375 , 300]
    call_type  : ["rpc-call", "rpc-cast"]
    driver     : ["1-rabbitmq", "1-router", "3-rabbitmq",\
 "3-router", "5-rabbitmq", "5-router"]
```

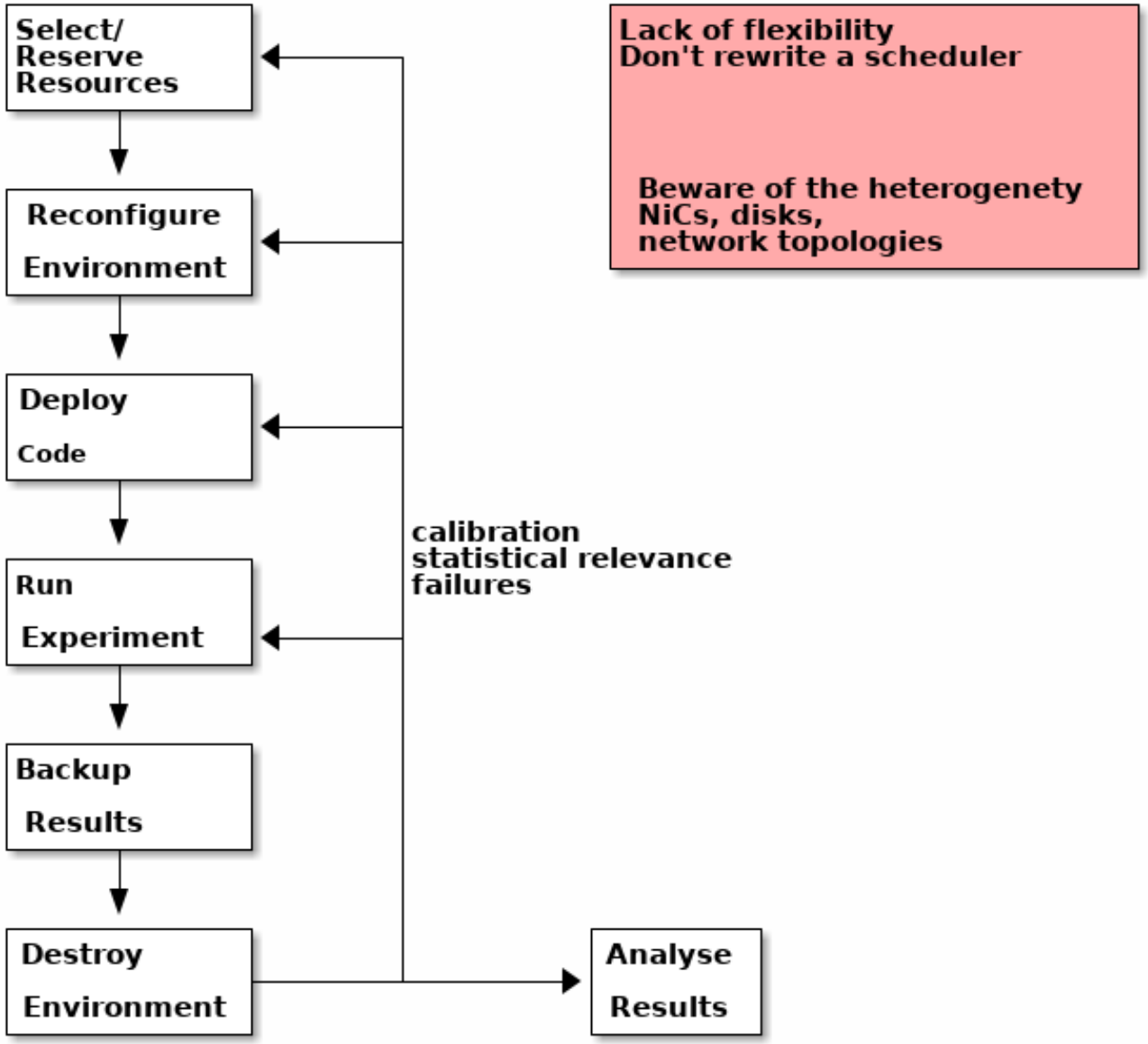
Javier Rojas Balderrama, Matthieu Simonin. Scalability and Locality Awareness of Remote Procedure Calls: An Experimental Study in Edge Infrastructures. CloudCom, Dec 2018, Nicosia, Cyprus. hal-01891567

The experimenter workflow and pitfalls

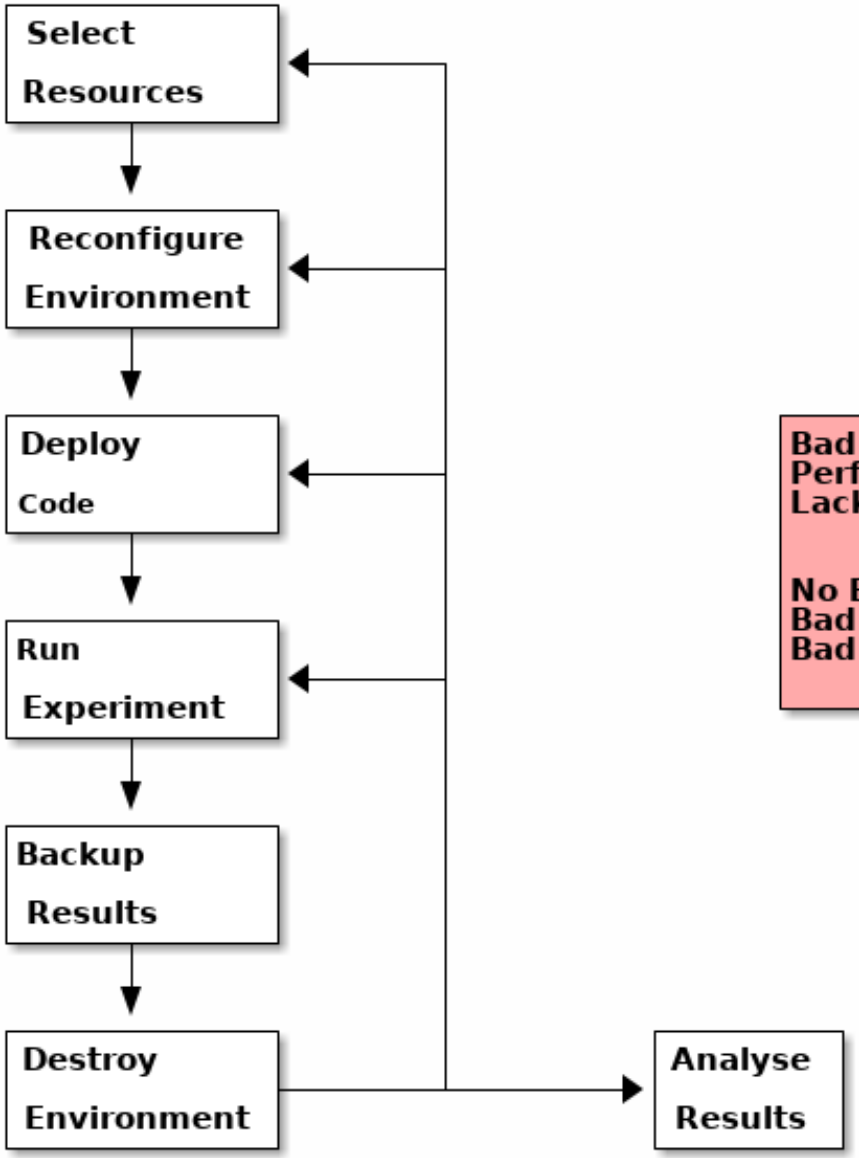
General remarks on the workflow

- It has re-entrant tasks
  - You need idempotency
- You ususally cannot fully unit-test a task (e.g deploy)
  - At design time: you want to restart from this specific task upon failure
  - You need a mechanism to adapt a task implementation

The experimenter workflow and pitfalls



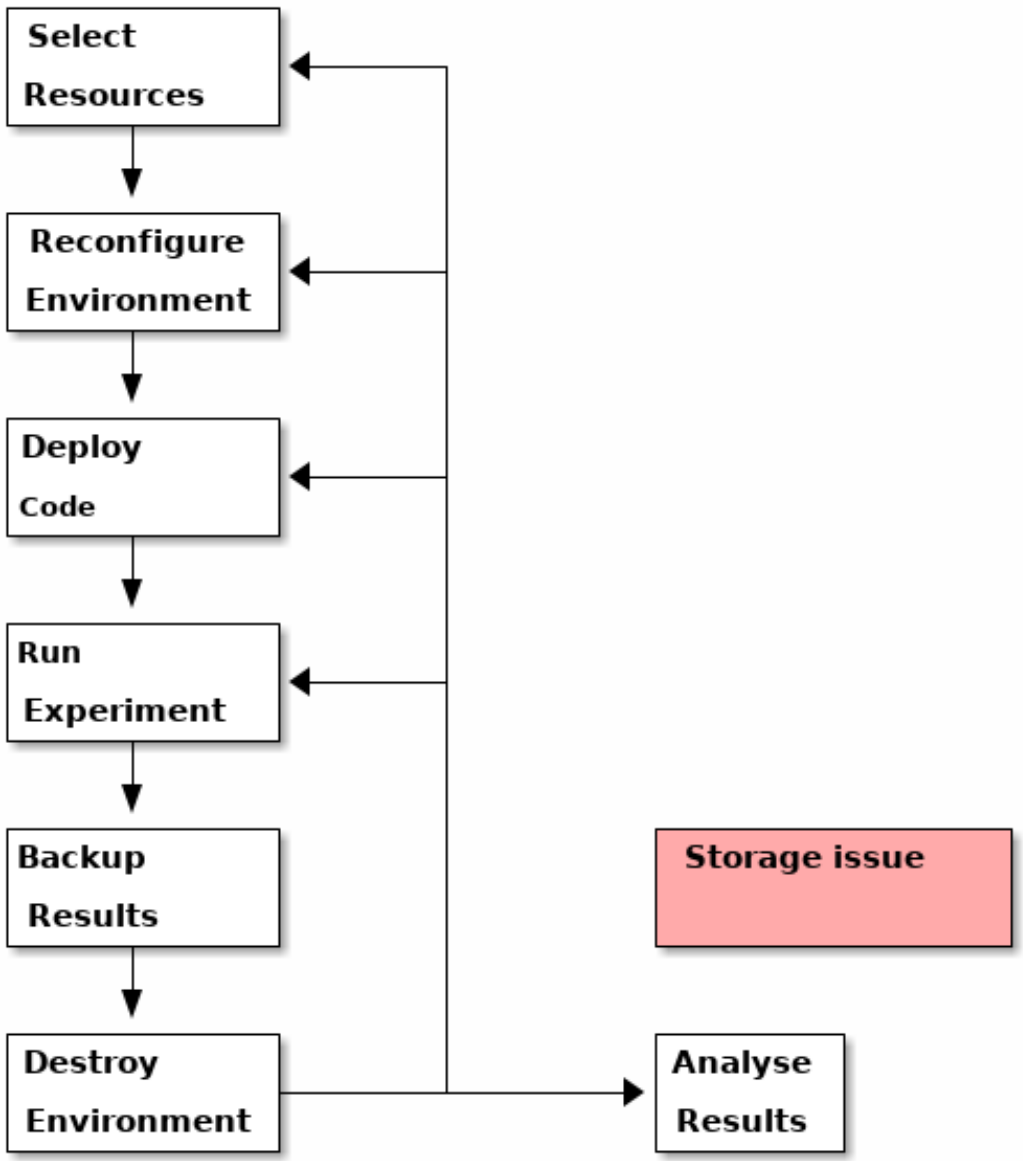
The experimenter workflow and pitfalls



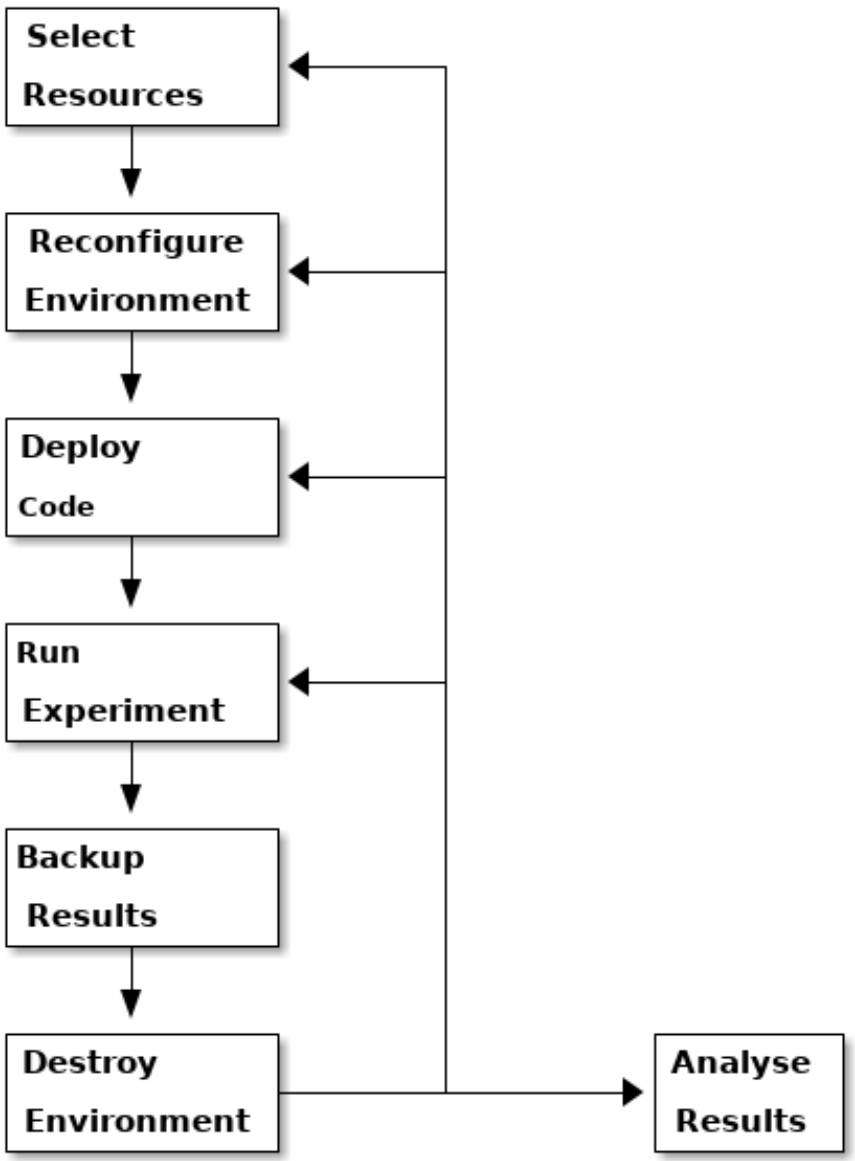
Bad abstraction
Performance hit
Lack of flexibility

No Error Handling
Bad monitoring
Bad calibration

The experimenter workflow and pitfalls



The experimenter workflow and pitfalls



lack of tooling
loss of intermediate datas
iteration impossible on analysis

Conclusion

To overcome all those pitfalls:

- Consider your experimentation code as a regular programming project
- Stand on other people shoulders as much as possible
- Talk of your experiment to other people