Cloud Computing for HPC Simulations

Diego Montes <kabute@uvigo.es>

Goals

- Insights of HPC over Cloud Computing.

- Give an starting point to start your own experiments in the Cloud.

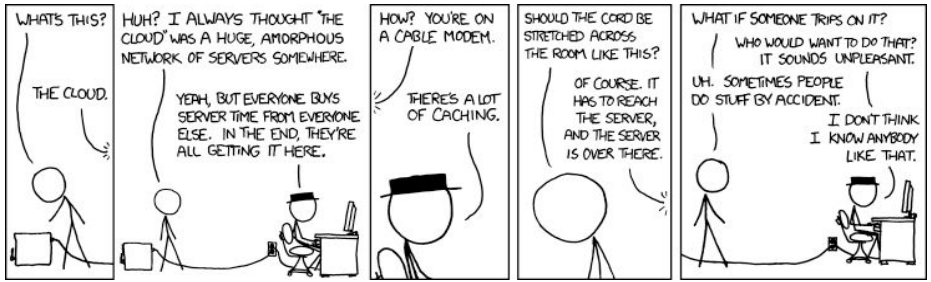
What is the Cloud Anyway?

As defined by the NIST:

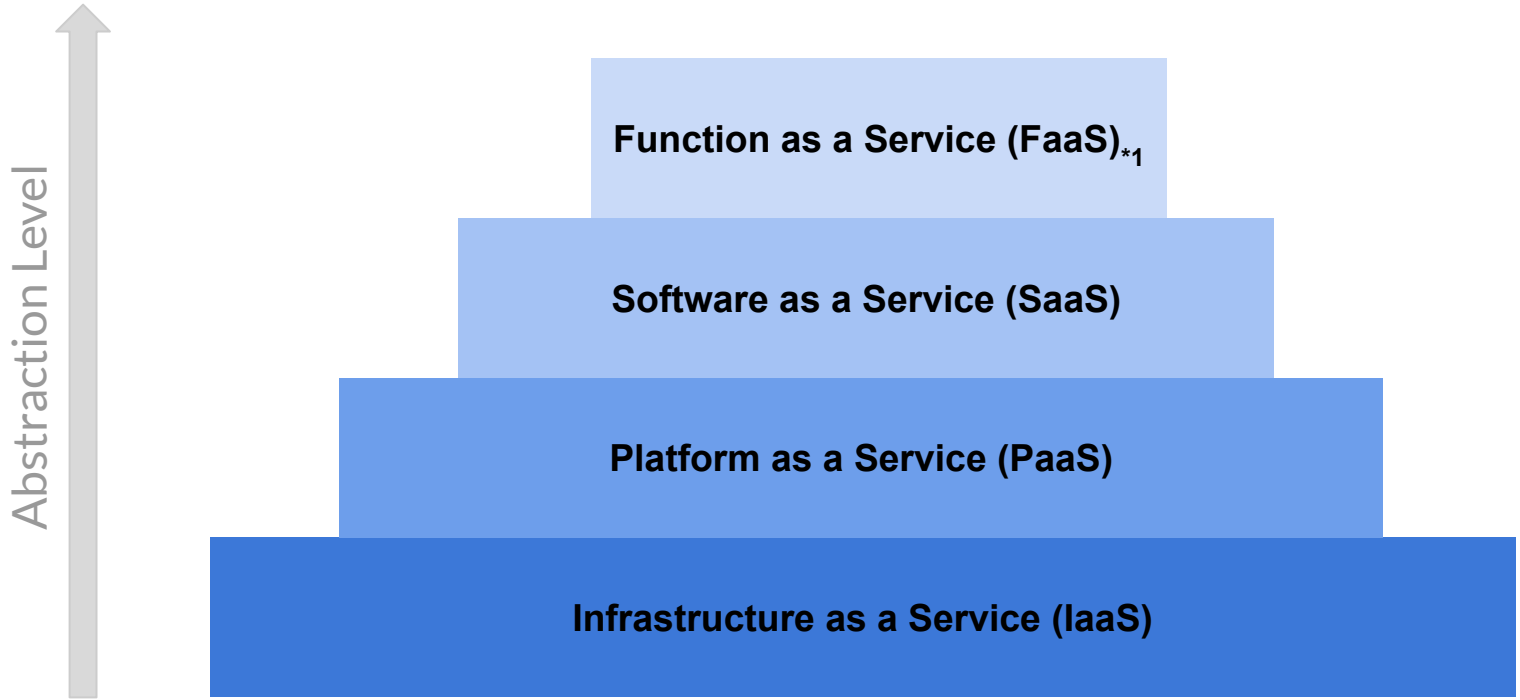
- On-demand Self-service
- Broad Network Access
- Rapid Elasticity
- Measured Service

What is the Cloud Anyway?

A broad concept that encompasses the abstraction of computer systems (including their resources) and presents them in a ubiquitous way.



Cloud - Service Levels



*1: Including the concept of “Serveless”

Cloud - Service Levels

Rule of Thumb, choose wisely:

Higher Abstraction ⇒ Less Operational Load ⇒ More Constraints ⇒ More Out-of-the-Box Features ⇒ Higher Vendor Dependency

Is the Cloud Right for my Experiment(s)?

Maybe… Start by answering yourself:

If Legacy Code:

- Does to code run on commodity hardware?

- How fault tolerant is the code?

Is the Cloud Right for my Experiment(s)?

If New Development:

- Can it take advantage (do I understand?) of the Cloud Paradigm? (APIs, ephemeral environments…)

Is the Cloud Right for my Experiment(s)?

In any case:

- What is the budget.
- How many resources do you really need.
- What are acceptable thresholds (e.g. failure/errors)

Developing HPC for the Cloud

Think **Cloud Native**:

A cloud native app is architected specifically to run in the elastic and distributed nature required by modern cloud computing platforms.

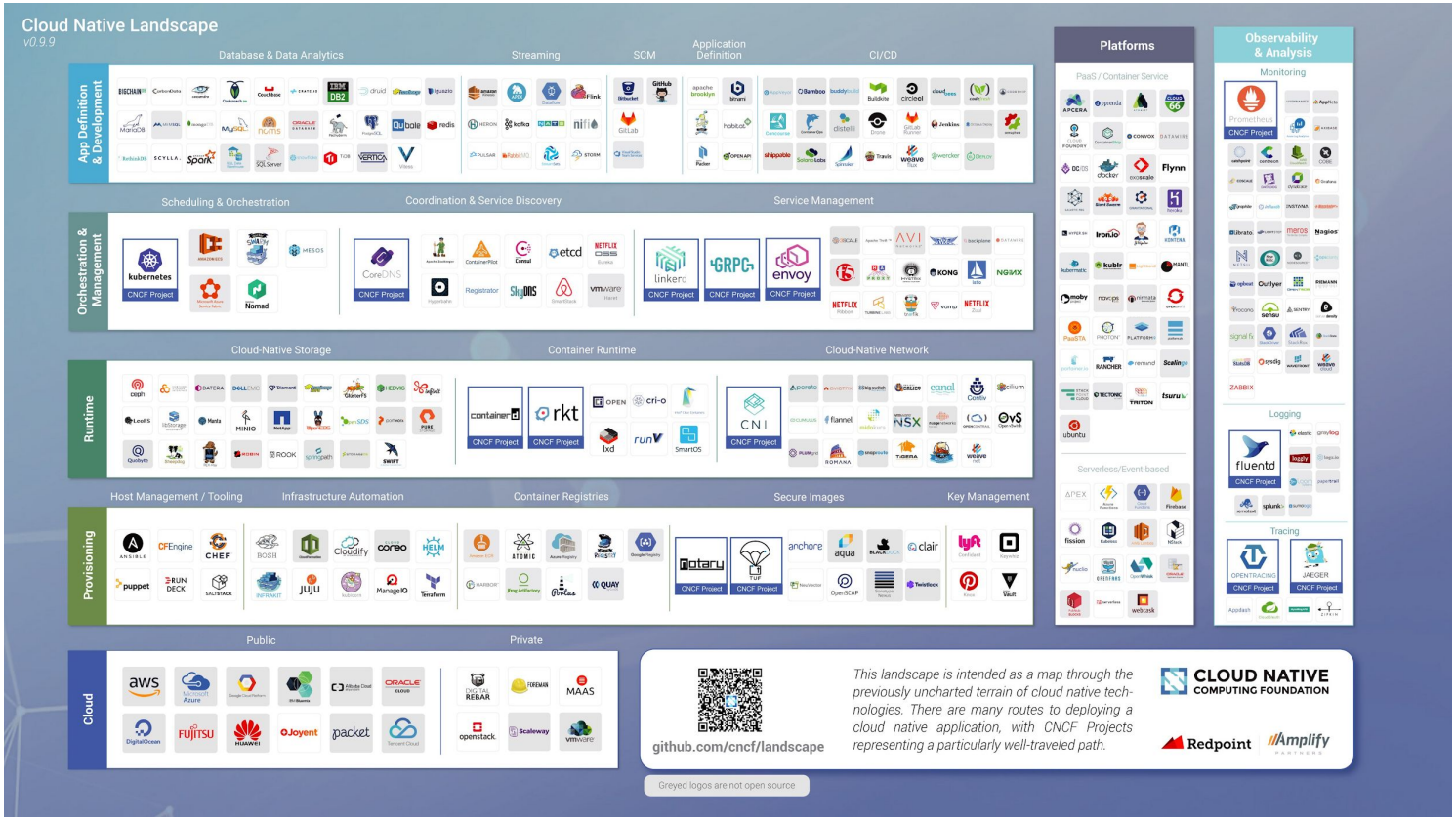
Developing HPC for the Cloud

- Immutable infrastructure, idempotence and containerization are not just a trend.

Developing HPC for the Cloud

- APIs are first class citizens.
- Commodity hardware (reliable over unreliable).
- Be prepared for failure: checking points, incremental backoff, different AZs...

Developing HPC for the Cloud



Running HPC in the Cloud

Compared with a more traditional environment, **by default**, the Cloud is not very HPC “friendly”:

- No low latency networks.
- Prone to different (points of) failure.
- No limits***

Running HPC in the Cloud

- Scheduling is now transparent, (initially) no need for queues.
- You might need to give a heads up to be sure resources are allocated.

Is the Cloud Right for my Experiment(s)?

But...

- Yes, there is HPC-friendly in the cloud (low latency, affinity, location…) ...
- … But it is not the standard.
- Do you really need it?

1 Minute Example: Serverless Monte Carlo

Simple case (Monte Carlo, not my code) that runs on Google Cloud Functions (FaaS) and returns the value of the calculation.

Up and running in less than 1 minute.

Code:*1

```
# main.py: Numerical Integration using Monte Carlo method
# FB - 201006137
import math
import random

# define any function here!
def f(x):
        return math.sin(x)

# define any xmin-xmax interval here! (xmin < xmax)
def montecarlo(request):
 xmin = 0.0
 xmax = 2.0 * math.pi

 # find ymin-ymax
 numSteps = 1000000 # bigger the better but slower!
 ymin = f(xmin)
 ymax = ymin
 for i in range(numSteps):
        x = xmin + (xmax - xmin) * float(i) / numSteps
        y = f(x)
        if y < ymin: ymin = y
        if y > ymax: ymax = y

 # Monte Carlo
 rectArea = (xmax - xmin) * (ymax - ymin)
 numPoints = 1000000 # bigger the better but slower!
 ctr = 0
 for j in range(numPoints):
        x = xmin + (xmax - xmin) * random.random()
        y = ymin + (ymax - ymin) * random.random()
        if math.fabs(y) <= math.fabs(f(x)):
        if f(x) > 0 and y > 0 and y <= f(x):
        ctr += 1 # area over x-axis is positive
        if f(x) < 0 and y < 0 and y >= f(x):
        ctr -= 1 # area under x-axis is negative

 fnArea = rectArea * float(ctr) / numPoints
 return "Numerical integration = " + str(fnArea)
```

*1: Not my code, numerical recipe from the Internet.

1 Minute Example: Deploying and Running

```
# Deploying (I am logged in google cloud, no other steps needed)
gcloud functions deploy montecarlo --runtime python37 --trigger-http

# Running (triggering) the code
curl "https://us-central1-myexperiment.cloudfunctions.net/montecarlo"
Numerical integration = 0.0024630086404143978
```

HPC for the Cloud - Pitfalls

- Embrace the Order in Chaos: Do NOT fight the nature of the Cloud.
- Plan, architect and plan again.
- Be prepared for failure: checking points, incremental backoff, different AZs...

Choosing a Provider

- Know your needs, requirements and budget.
- Follow a methodology (e.g. RACI).
- Public, Private or Hybrid?

Money...

- The Cloud can be expensive… but also cheap… (as well as DCs).
- Many waivers and programs available for funding (e.g. EU Horizon 2020). Talk to your provider.

Takeaways

- The Cloud is not a silver bullet.
- Understand the paradigm first: read a lot or reach Engineers with expertise.
- Design and architect are the most important stages.

Thank You! - Questions?

References

- *The NIST Definition of Cloud Computing*. P Mell, et al.NIST, 2011.
- *Designing Data-Intensive Applications*, Chapter 8. M Kleppmann. O'Reilly, 2017.
- *Views on the potential of Cloud computing and IaaS/HPCaaS for meteorology and climatology*. D Montes, et al. EMS Annual Meeting Abstracts 14 (EMS2017), 639, 2017.
- *CNCF Cloud Native Interactive Landscape*. Cloud Native Computing Foundation, <https://landscape.cncf.io> . October 2017.
- Design and Implementation of a Cloud Computing Adoption Decision Tool: Generating a Cloud Road. I Bidosola, et al. PLoS ONE, 2017.