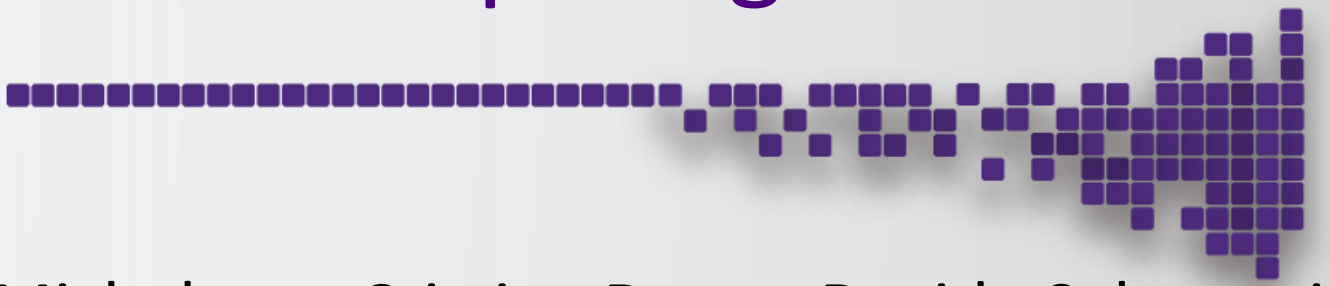
Speed up research by leveraging INDIGO-DataCloud solutions: containers in user-land and on-demand computing clusters



Diego Michelotto, Cristina Duma, Davide Salomoni
INFN-CNAF
diego.michelotto@cnaf.infn.it

INDIGO - DataCloud

RIA-653549

INDIGO-DataCloud is co-founded by the Horizon 2020Framework Programme

Outline

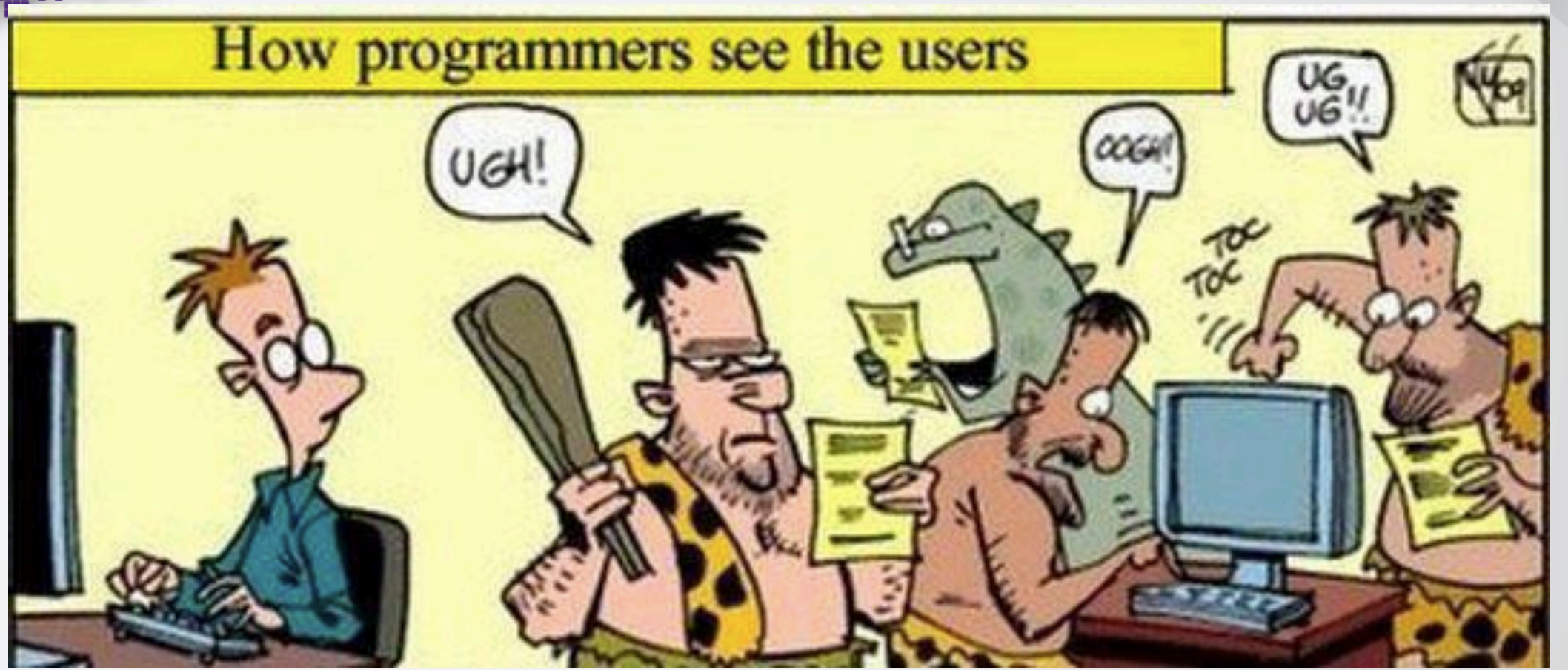
- Brief introduction to INDIGO-DataCloud project

- Solutions for HEP Community:
  - Udocker – containers In user-land
  - DoDAS - Dynamic On Demand Analysis Service

INDIGO-DataCloud

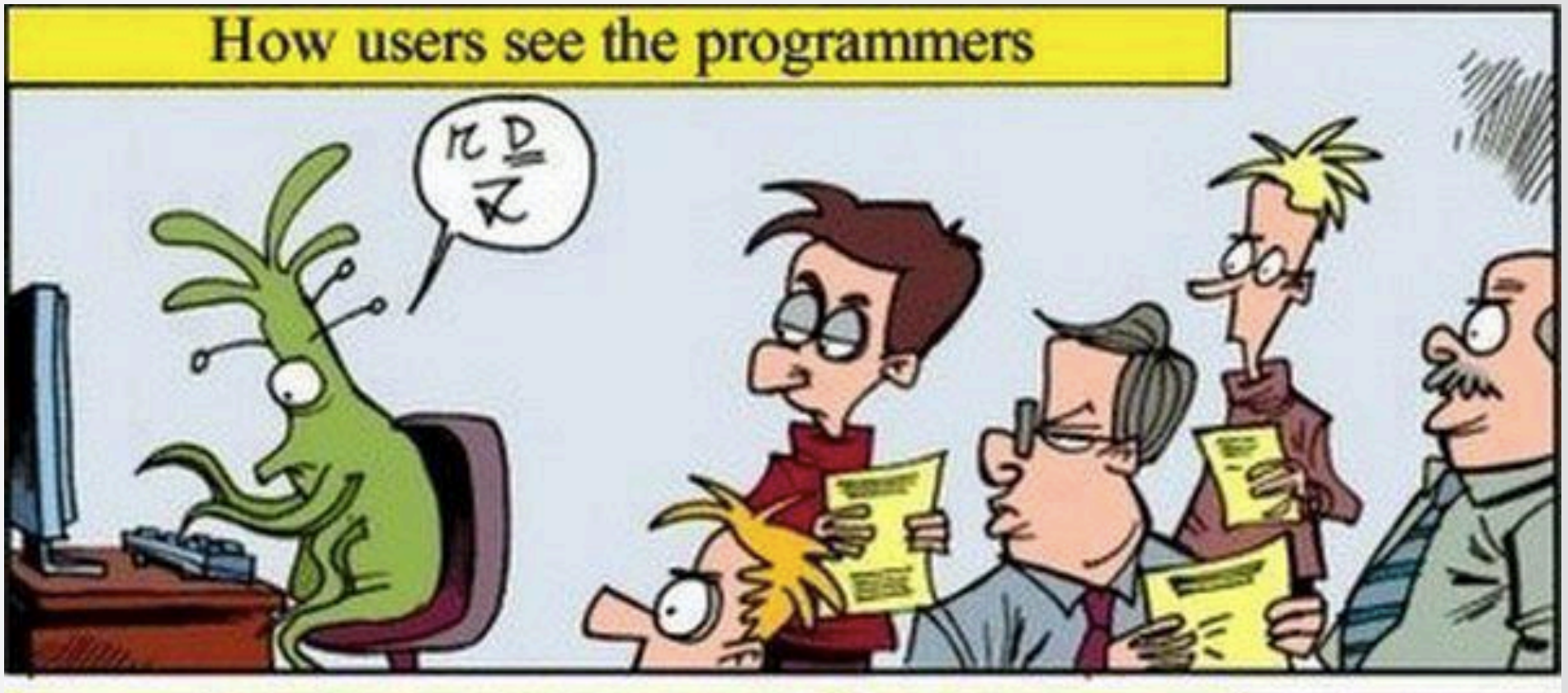
- **An H2020 project** approved in January 2015 in the EINFRA-1-2014 call
  - 11.1M€, 30 months (**from April 2015 to September 2017**)
- **Who**: **26 European partners** in 11 European countries
  - Coordination by the Italian National Institute for Nuclear Physics (INFN)
  - Including developers of distributed software, industrial partners, research institutes, universities, e-infrastructures
- **What**: **develop an open source Cloud platform** for computing and data ("DataCloud") tailored to science.
- **For**: **multi-disciplinary scientific communities**
  - E.g. structural biology, earth science, physics, bioinformatics, cultural heritage, astrophysics, life science, climatology
- **Where**: deployable on **hybrid (public or private) Cloud infrastructures**
  - INDIGO = **IN**tegrating **D**istributed data **I**nfrastructures for **G**lobal Expl**O**itation
- **Why**: answer to the technological **needs of scientists** seeking to easily exploit distributed Cloud/Grid compute and data resources.

Something ~~is~~ was still missing in the Cloud world…



Source: http://goo.gl/wT8XEq

Something ~~is~~ was still missing in the Cloud world…



Source: http://goo.gl/wT8XEq

Gap analysis

- What is missing:
  - Open **interoperation** / federation across (proprietary) CLOUD solutions at
    - IaaS,
    - PaaS,
    - and SaaS levels
  - Managing **multitenancy**
    - At large scale…
    - … and in heterogeneous environments
  - Dynamic and seamless **elasticity**
    - For both private and public cloud…
    - … and for complex or infrequent requirements
  - **Data management** in a Cloud environment
    - Due to technical…
    - … as well as to legal problems

Filling these gaps should lead to:

- Interoperable PaaS/SaaS solutions addressing both public and private Cloud infrastructures
- Migration of legacy applications to the Cloud

User (Scientist) first

Users first: from here…

Use-Cases from
LifeWatch
EuroBioImaging
INSTRUCT
LBT
CTA
WeNMR
ENES
eCulture
ELIXIR
EMSO
Dariah
WLCG

100 distinct requirements

Converted to concrete activities in the Project DoW

Computational

- Software as a Service
- Execution of Workflows
- Cloud Bursting
- X-Site Execution
- Improved Scheduling
- Access to GP-GPU's

Storage

- Distributed Storage, accessible via POSIX
- Persistent Data Storage

Infrastructure

- Global Level AAI
- Software Defined Networks

… to here … Community Case Studies

INDIGO - DataCloud

Report on how several scientific communities are implementing their own requirements into concrete applications using INDIGO-DataCloud components.

- Monitoring and Modelling Algae Bloom in a Water Reservoir
- TRUFA (Transcriptomes UserFriendly Analysis)
- Medial Imaging Biobanks
- Molecular Dynamics Simulations
- Astronomical Data Archives
- Archive System for the Cherenkov Telescope Array (CTA)
- HADDOCK Portal
- DisVis
- PowerFit
- Climate models inter comparison data analysis
- eCulture Science Gateway
- EGI FedCloud Community Requirements
- ELIXIR-ITA: Galaxy as a Cloud Service
- MOIST – Multidisciplinary Oceanic Information System
- Data Repository platform for DARIAH

https://www.indigo-datacloud.eu/documents-deliverables

… to here … Community Case Studies

Using “Champion” approach :

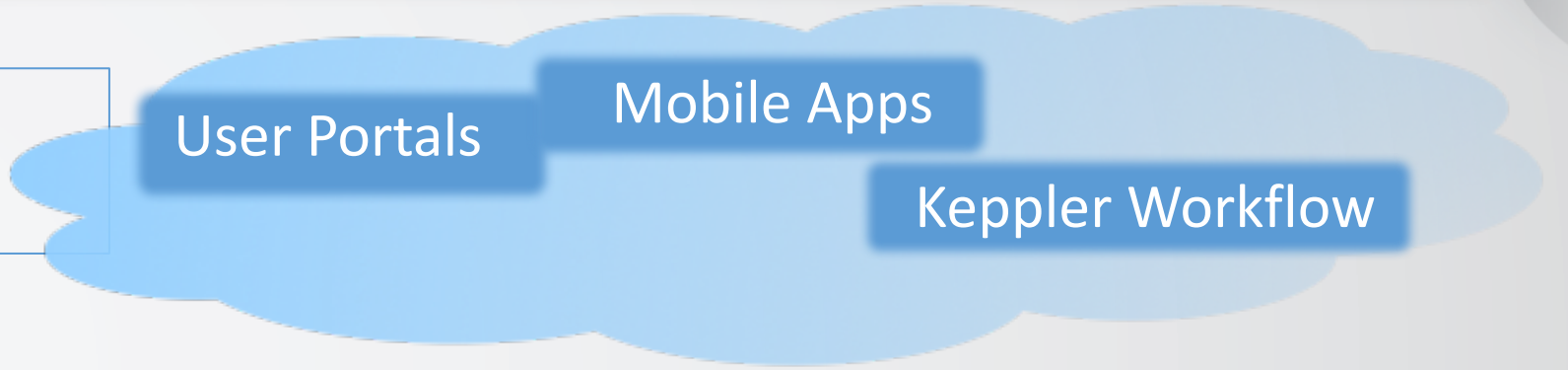
Communities have to provide a scientist, becoming an expert in computing and INDIGO terminology.

Report
scientif
are imp
own re
concre
usi
DataClo

https://www.indigo-datacloud.eu/documents-deliverables

The long road to the release, from the architecture…

INDIGO-DataCloud
General Architecture*

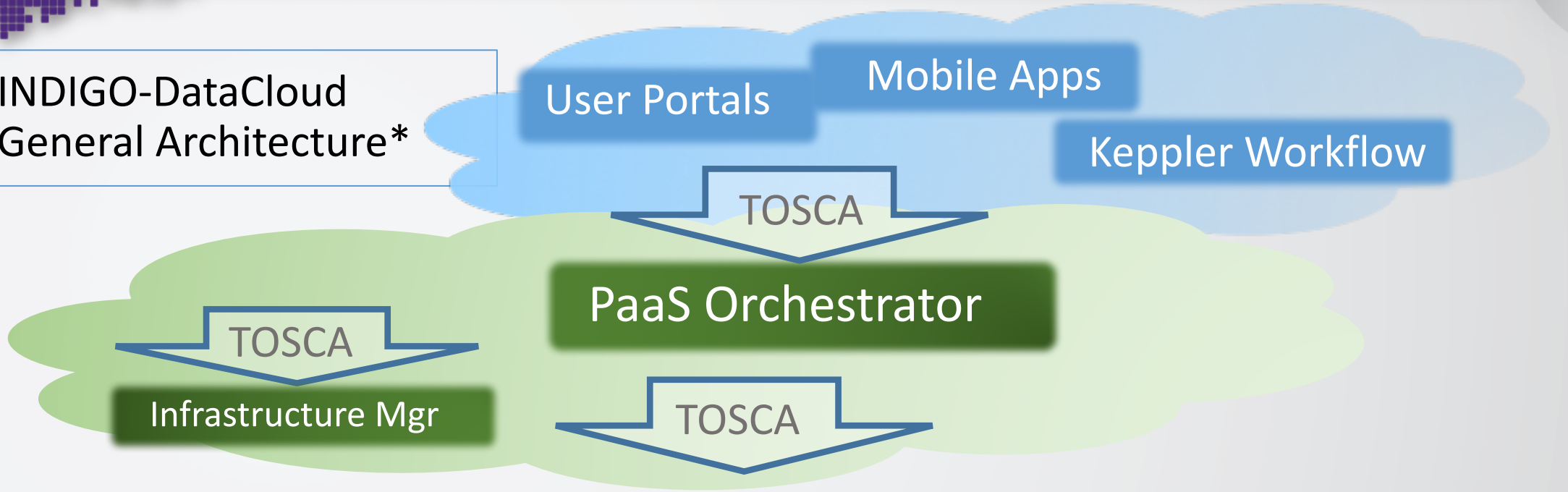


Portals

The long road to the release, from the architecture…

INDIGO - DataCloud

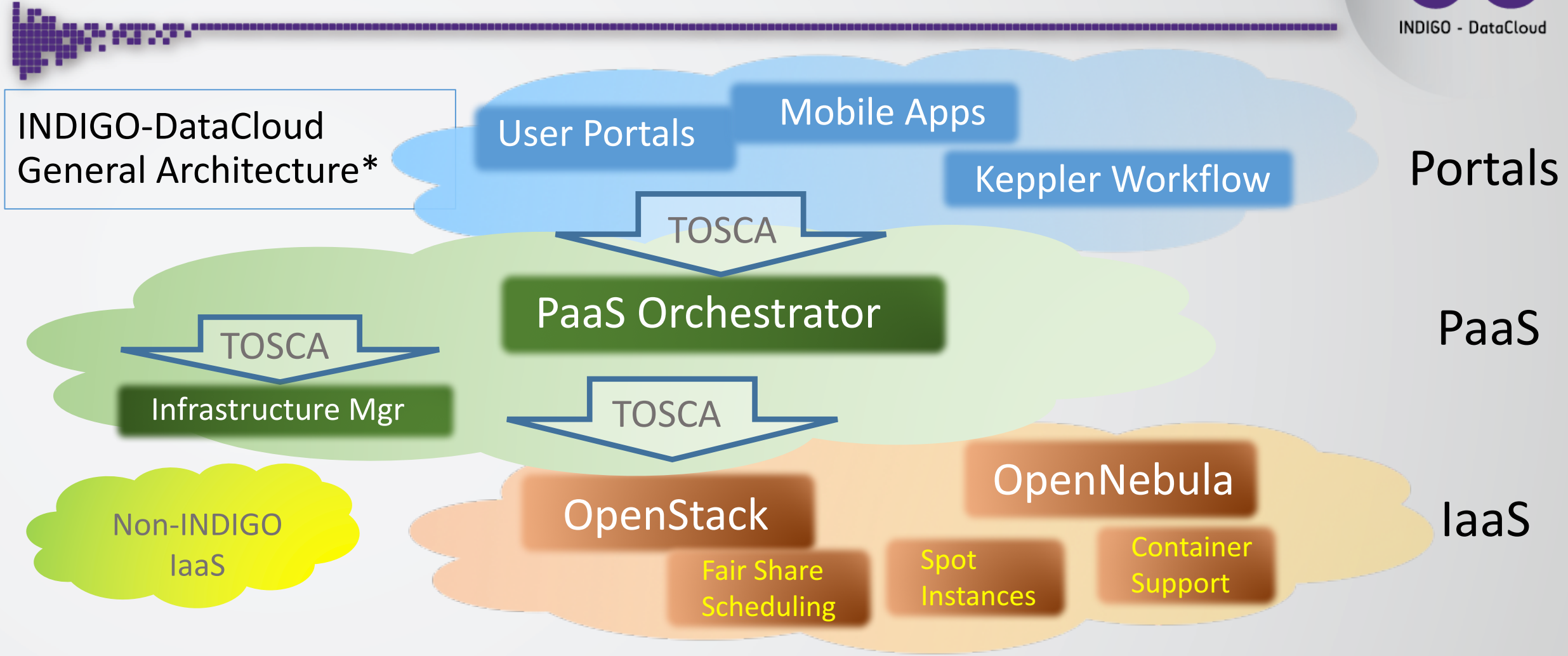
INDIGO-DataCloud
General Architecture*



Portals

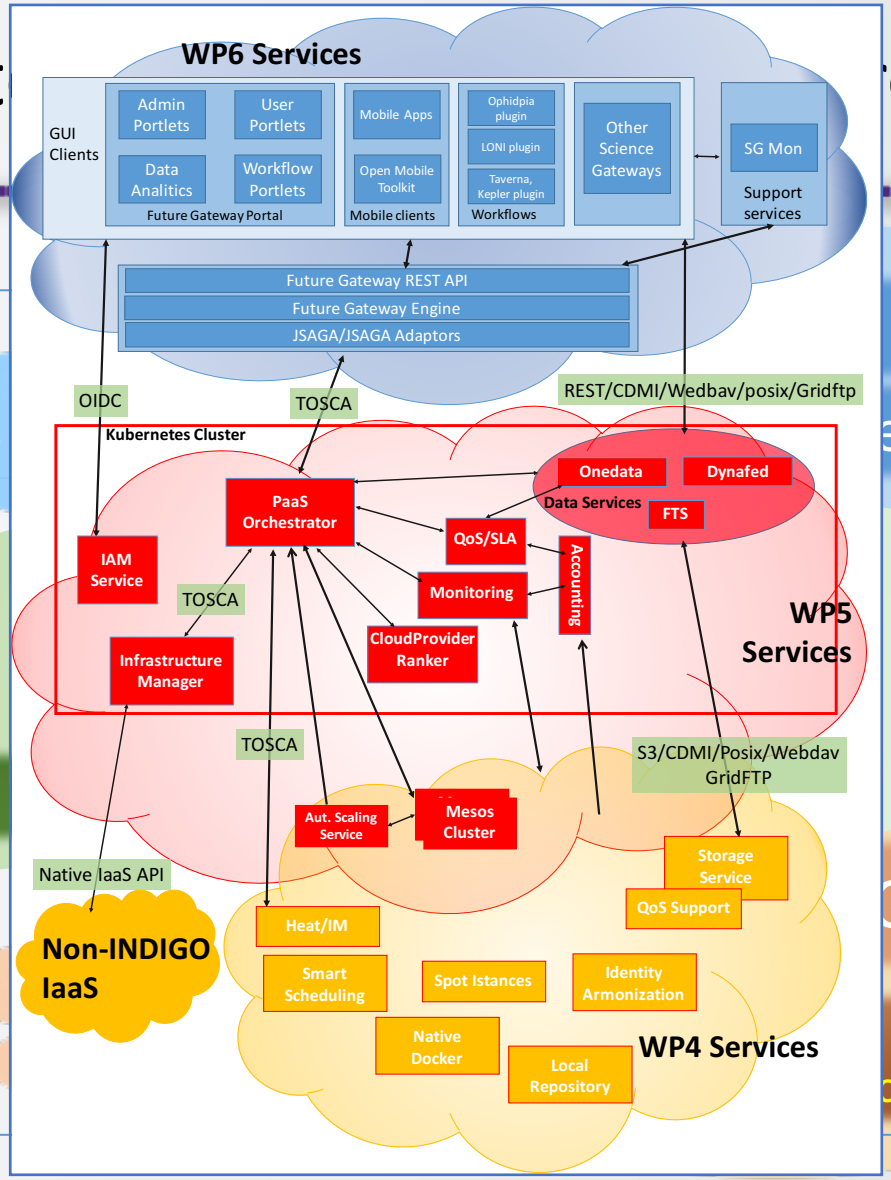
PaaS

The long road to the release, from the architecture…



The long road t… rchitecture…

INDIGO-DataCloud
General Architecture*



Portals

PaaS

IaaS

TOSCA

Infrastructure Mgr

Non-INDIGO IaaS

eppler Workflow

OpenNebula

Container Support

*: see details in http://arxiv.org/abs/1603.09536 or in https://www.indigo-datacloud.eu/documents-deliverables

The first INDIGO-DataCloud Software Release

August 8, 2016

INDIGO-DATACLOUD FIRST PUBLIC RELEASE IS OUT!

INDIGO MIDNIGHTBLUE

On August 8, 2016 INDIGO-DataCloud project announced the general availability of its first public software release, codenamed MidnightBlue. The release comes after an initial phase of requirement gatherings which involved several European scientific collaborations in areas as diverse as structural biology, earth sciences, physics, bioinformatics, cultural heritage, astrophysics, life sciences, climatology, etc. This resulted in the development of many software components addressing existing technical gaps linked to easy and optimal usage of distributed data and compute resources.

INDIGO - DataCloud
Better Software for Better Science

INDIGO MidnightBlue Service Catalogue

Updates and new releases of the INDIGO services are expected to come in the forthcoming months.
The first scientific applications and use cases adopting this first INDIGO release are expected starting from September 2016.

https://www.indigo-datacloud.eu/communication-kit

INDIGO-DATACLOUD FIRST PUBLIC RELEASE IS OUT!

INDIGO MIDNIGHTBLUE

INDIGO - DataCloud

Common Solutions	
Identity and Access Management	14
Data Center Solutions	
Fairshare Scheduler for OpenStack	15
Partition Director Service for Batch and Cloud resources	16
Cloud Provider Ranker	17
Infrastructure Manager	23
OCCI support for OpenStack and OpenNebula	24
Extended OpenStack and OpenNebula Functionalities	25
Data Solutions	
Global Data Access	18
Storage Quality of Service and Data Lifecycle support	27
Automated Solutions	
PaaS Orchestrator	20
Core PaaS	26
QoS/SLA Management Service	30

User-oriented Solutions	
Userspace Container Support	21
Data Mining and Analytics for eScience Server	22
Future Gateway (Programmable Scientific Portal)	28
INDIGO Plug-ins for scientific workflow systems	29
INDIGO Mobile Toolkit	31

INDIGO - DataCloud

Selected use cases

Selected use cases:

1. Running Docker containers without Docker
2. An application to CMS

Selected use cases:

1. Running Docker containers without Docker
2. An application to CMS

Running Docker containers without Docker

- Imagine that Computing & Storage resources are made available as a *pool, in which:*

  1. You cannot count on having your very specific software/libraries installed,

  2. There is no system software available to run your application encapsulated as a container

We show how to address this scenario by using:

- **the INDIGO developed tool *udocker***

Containers – a few facts

“Linux Containers” is a technology provided by the Linux kernel, to “contain” a group of processes in an independent execution environment: this is called a “container”.

Advantages of Containers versus classical Virtual Machines:

- Containers are much more light-weighted than a Virtual Machine
- They provide an enormous simplification of the software deployment processes

✓ The most extended software to build containers is called **Docker**

✓ Docker is optimized for the **deployment of applications as containers**

One can generate a completely tailored “docker image” of any Linux Operating System, with all the required libraries, compilers, source codes,… which later on can be run as a container.

Containers in multi-user environment

Adoption of docker is being very slow in computing farms or interactive linux system shared by many users

- Particularly so in large infrastructures operated for many users (HPC systems)
- The typical situation is that docker is not installed, and one cannot run containers without support from the system software.
- The main issue is that docker needs root permissions to run a container.
- Even though the user, within the context of the container is completely isolated from the rest of the machine, it raises all the alarms among security people
- A user with access to docker can own the hosting system

Containers in multi-user environment

Adoption of docker is being very slow in computing farms or interactive linux system shared by many users

INDIGO has developed udocker

- Particularly so in large infrastructures operated for many users (HPC systems)
- The typical situation is that docker is not installed, and one cannot run containers without support from the system software.
- The main issue is that docker needs root permissions to run a container.
- Even though the user, within the context of the container is completely isolated from the rest of the machine, it raises all the alarms among security people
- A user with access to docker can own the hosting system

INDIGO udocker

- **A tool to execute content of docker containers in user space** when docker is not available
  - enables download of docker containers from dockerhub
  - enables execution of docker containers by non-privileged users
- **It can be used to execute the content of docker containers in Linux batch systems and interactive clusters managed by others**
- **A wrapper around other tools to mimic docker capabilities**
  - current version uses **proot** to provide a chroot like environment without privileges (it runs on CentOS 6, CentOS 7, Fedora, Ubuntu)
- **More info and downloads at:**
  - https://www.gitbook.com/book/indigo-dc/udocker/details
  - https://indigo-dc.gitbooks.io/udocker/content/doc/user_manual.html

INDIGO udocker

- Everything is stored in the user **home dir** or some other location
- Container layers are download to the user home
- Directory trees can be created/extracted from these container layers
- proot uses the debugger ptrace mechanism to change pathnames and execute transparently inside a directory tree
- No impact on read/write or execution, only impact on system calls using pathnames (ex. open, chdir, etc)

- Does not require installation of software in the host system:
  - udocker is a python script
  - proot is statically compiled

Architecture & Limitations

Architecture:

- Single-file python script
- Fetches public images from Docker Hub
  - Can also import image tarballs exported via docker save
- Creates container filesystem hierarchy in $HOME/.udocker
- Uses PRoot for (limited) sandboxing
  - Almost no CPU overhead
  - Negligible data I/O overhead
  - Sensible metadata I/O overhead

Limitations:

- Images cannot be created by udocker
  - Use Docker on another system to build images
- Privileged OS operations are not possible
- Debugging inside containers does not work
- Private repositories are not supported
  - … but you can use docker save and udocker load

Workflow

Docker:

```
# docker search ubuntu
# docker pull ubuntu:14.04
# docker run --name=mycontainer ubuntu:14.04 lsb_release -a
# docker rm mycontainer
```

udocker:

```
$ udocker search ubuntu
$ udocker pull ubuntu:14.04
$ udocker create --name=mycontainer ubuntu:14.04
$ udocker run mycontainer 'lsb_release -a'
$ udocker rm mycontainer
```

INDIGO Added Value: udocker is a crucial tool

INDIGO - DataCloud

For researchers, often, **the difference between being able to work, or not.**

- **Increasing accesibility to e-infrastructures**
  - By providing a **complete encapsulation** to the software, without impacting performance.
  - Crucial when **targeting generic resources elsewhere**, when there is no way to know what system software is pre-installed.
- **Providing the means for users to be independent** of the sysadmin layer
  - Independent of the System Software

Selected use cases:

1. Running Docker containers without Docker
2. An application to CMS

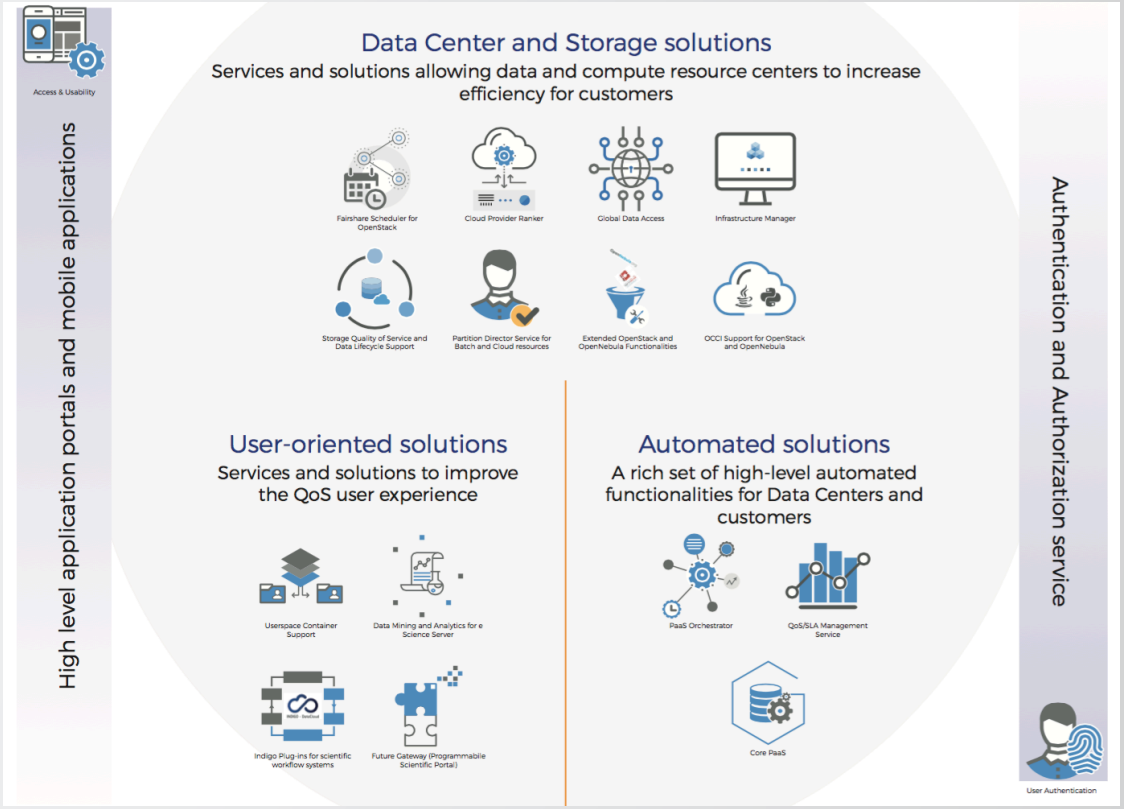
An application to LHC/CMS - enhancing CMS analysis workflows

INDIGO - DataCloud

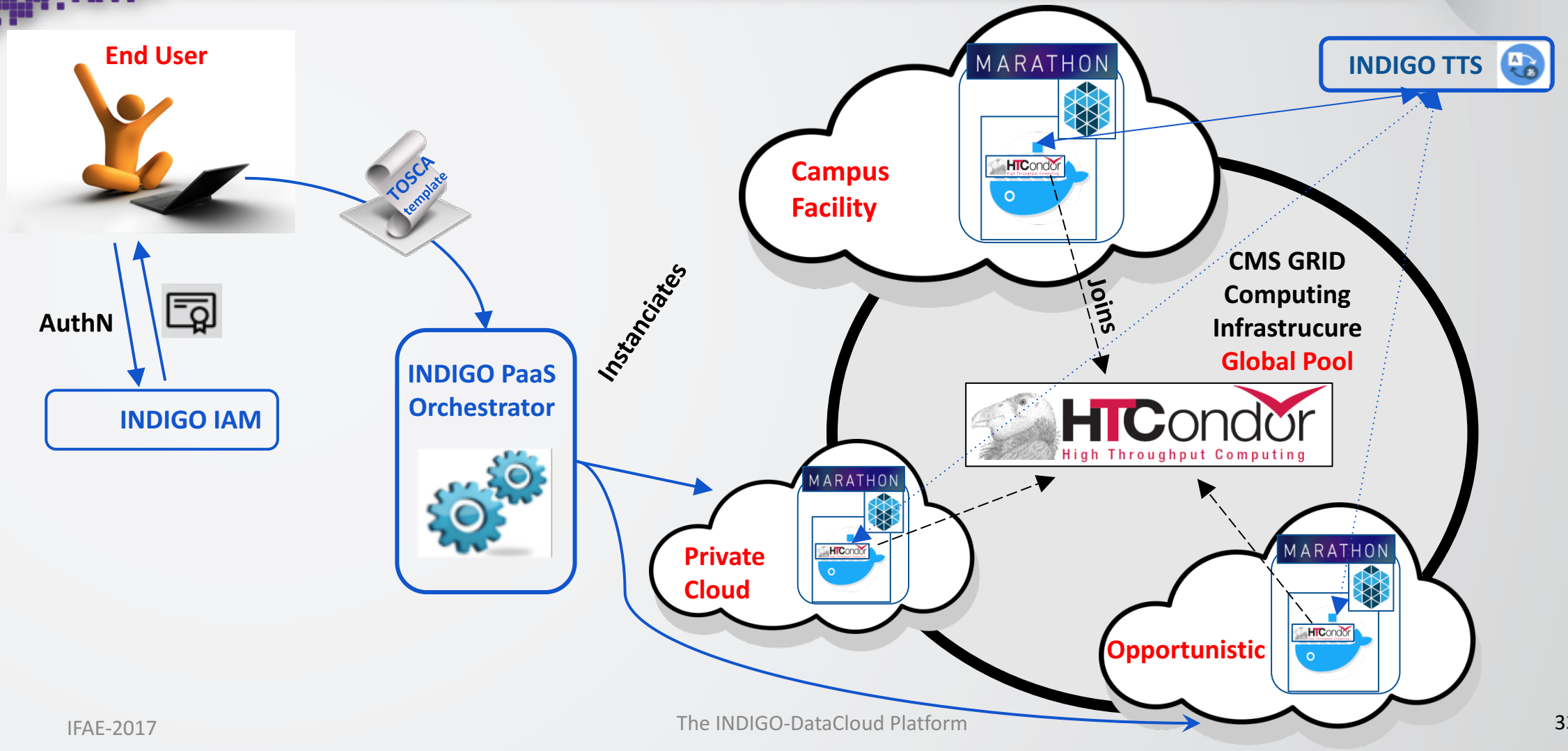
- The **goal** is to develop a solution for generating automatically an on-demand, container-based cluster for CMS in order to allow:
  - The effective use of **opportunistic resources**, such as general purposes campus facilities.
  - The **dynamic extension** of an already existing dedicated facility.
- By simplifying and automating the process of creating, managing and accessing a pool of computing resources the project aims to **improve**:
  - **Sites management:**
    - A simple solution for dynamic/elastic T2 extensions on “opportunistic”/stable resources
    - A friendly procedure to dynamically instantiate a spot “Tier3-like resource center”
  - **Users experience:**
    - Generation of an ephemeral on-demand T3 seen by the Experiment computing infrastructure as a personal WLCG-type facility, in order to serve a group of collaborators. The system must allow the use of standard/regular CMS Tools such as CRAB.
  - **Experiment-Collaboration resources:**
    - A comprehensive approach to opportunistic computing. A solution to access and orchestrate e.g. multiple campus centers, harvesting all the free CPU cycles without major deployment efforts.

INDIGO Services and solutions adopted

- **Data Center Solutions**
  - Mesos, Marathon, CLUES
- **Data/Storage Solution:**
  - Dynafed, FTS , Onedata
- **Automated Solutions**
  - TOSCA templates, Orchestrator
- **Common Solution**
  - Identity Access Management (IAM), Token Translation Service (TTS)



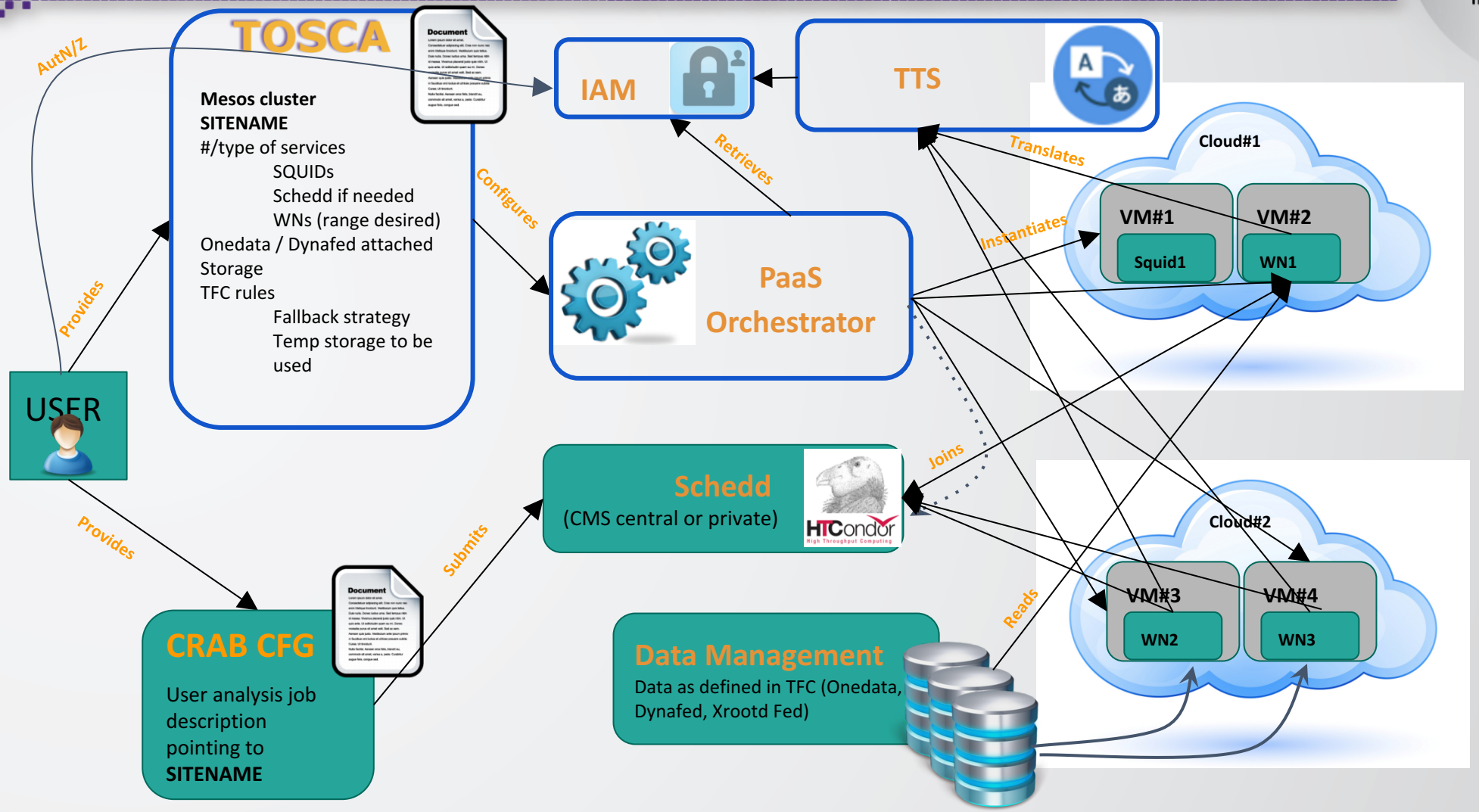
Solution Developed



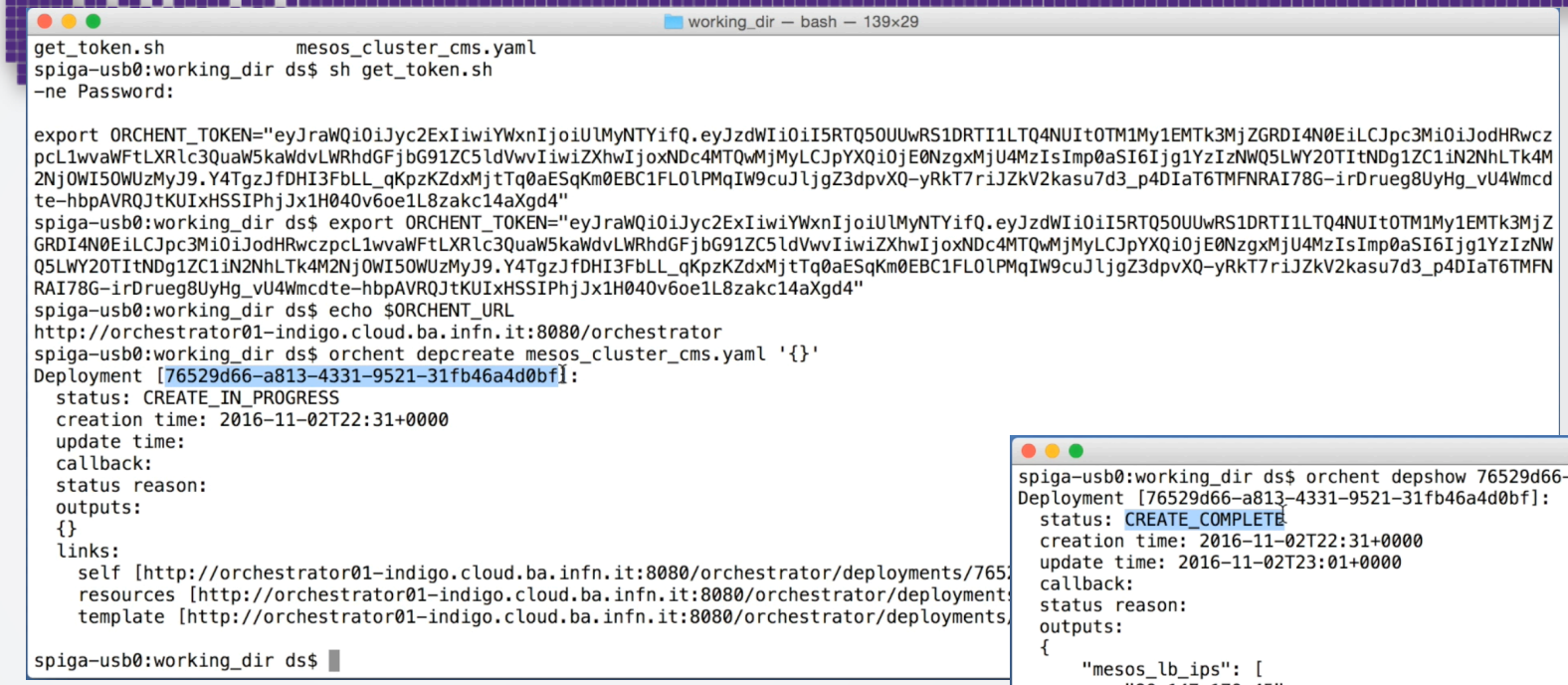
Application to CMS, four pillars:

- **Cluster Management:**
  - Mesos clusters as a solution in order to execute docker for all the services required by a regular CMS site (Worker Nodes, HTCondor Schedd and squids).
  - Marathon guarantees us the dynamic scaling up and down of resources, a key point.
- **AuthN/Z & Credential Management:**
  - The INDIGO Identity Access Management (IAM) service is responsible for AuthN/Z to the cluster generation.
  - The Token Translation Service (TTS) enables the conversion of IAM tokens into an X.509 certificate
    - NOTE: This allows Mesos slaves (running HTCondor_startd daemon) to join the CMS central queue (HTCondor_schedd) as a regular Grid WN
- **Data Management:**
  - Dynafed is the approach currently followed by the project. A further possibility we will investigate is Oneclient (from Onedata) as a tool allowing to mount remote Posix filesystems.
- **Automation:**
  - TOSCA templates, meant to be managed by INDIGO PaaS Orchestrator, allow the automation of the overall setup.
    - The aim is to produce a single YAML file describing the setup of all required services and deps.

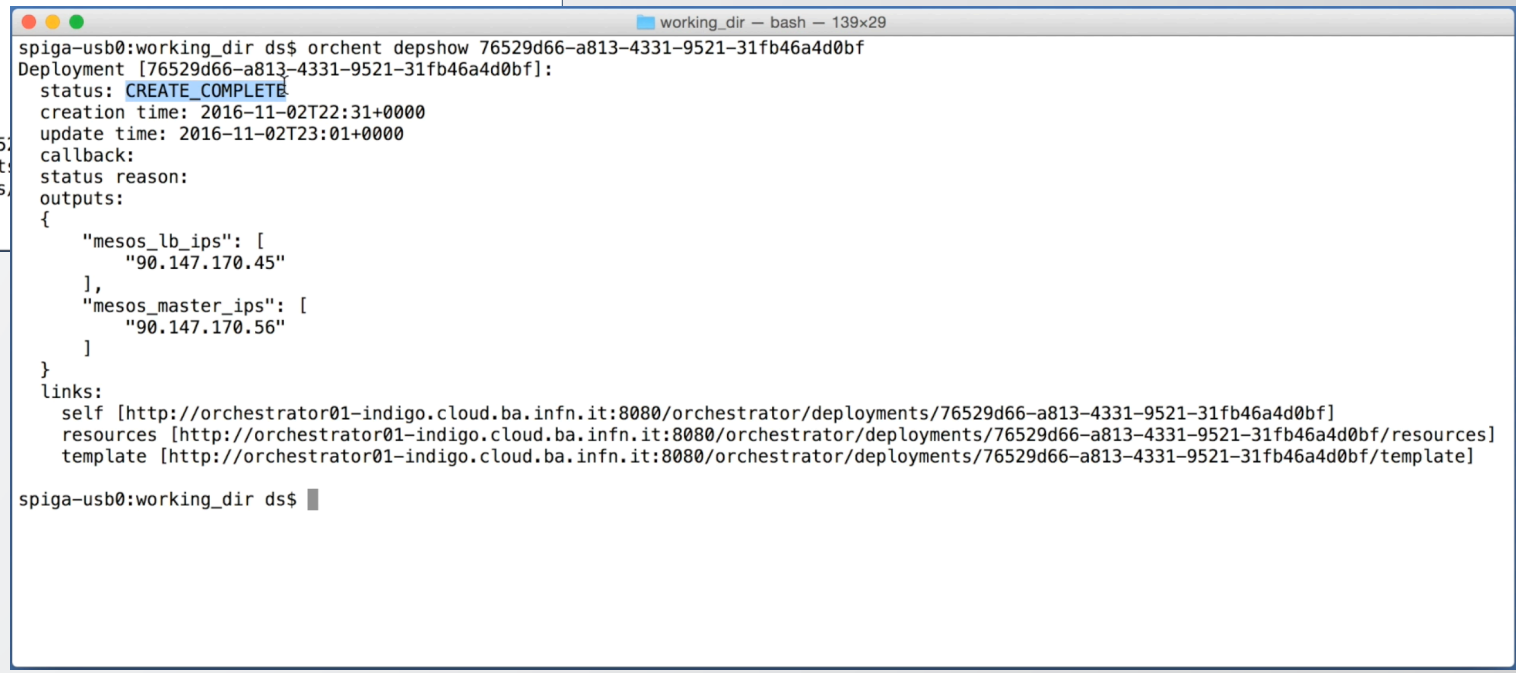
Application to CMS, architecture of the solution



Status: The prototype… It is working

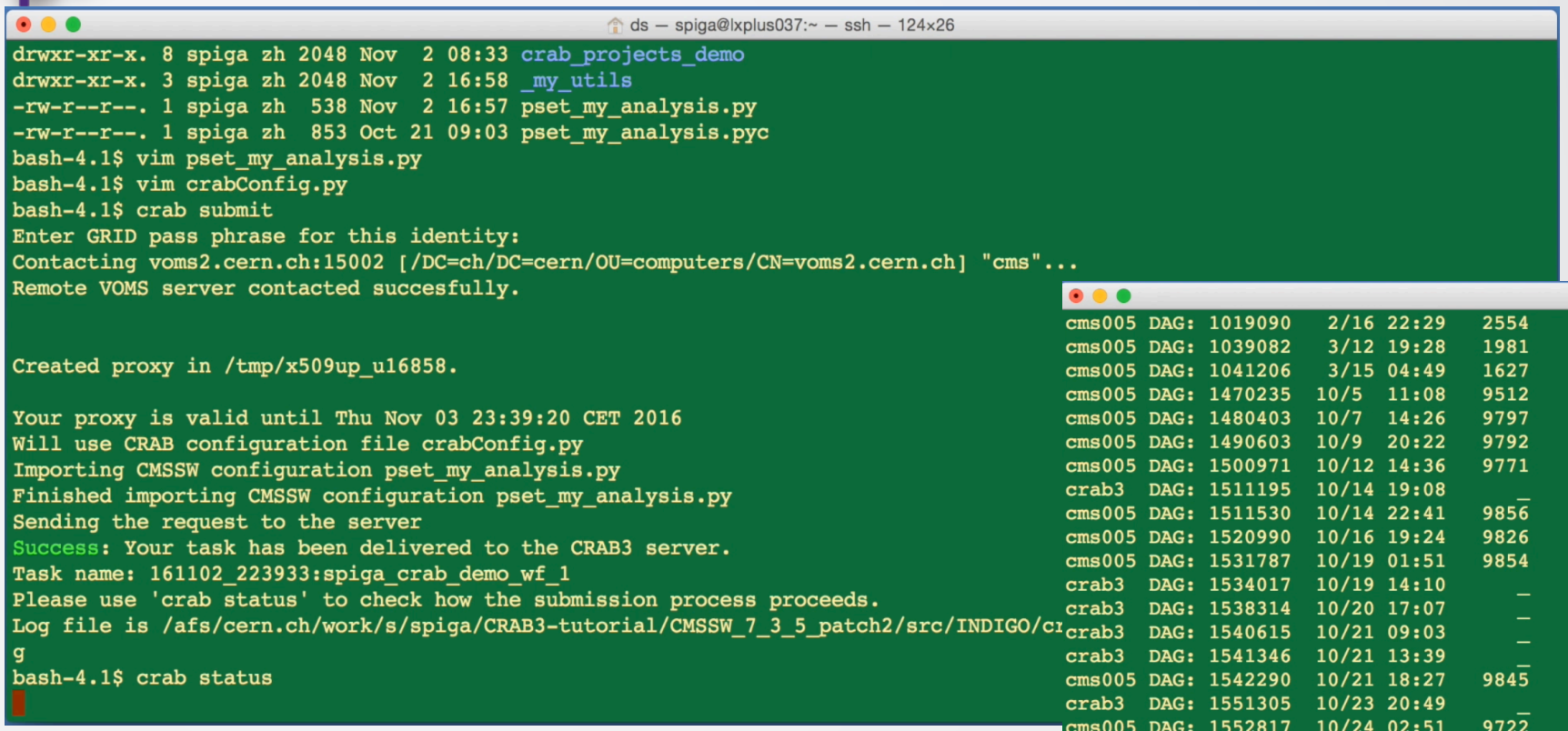


Deploying a CMS Mesos cluster through INDIGO PaaS Orchestrator



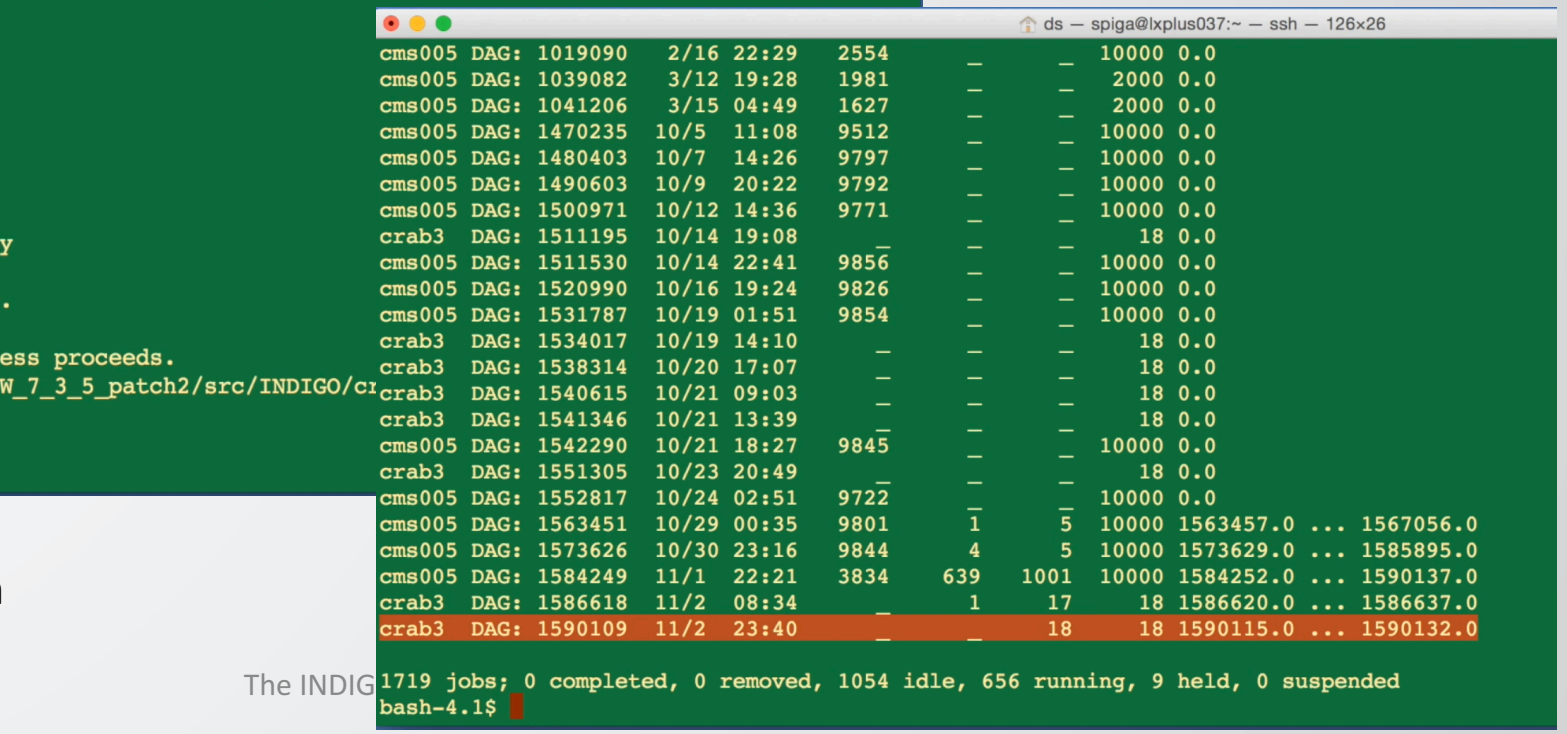
- The Mesos Cluster generation has been fully automated
- TOSCA (+Ansible) INDIGO template for a Mesos cluster:
  - Squid proxy (docker), CVMFS setup, WN (docker), proxy manager service (docker)

… A real CMS Analysis Workflow



A Regular CRAB Job Submission

condor_q

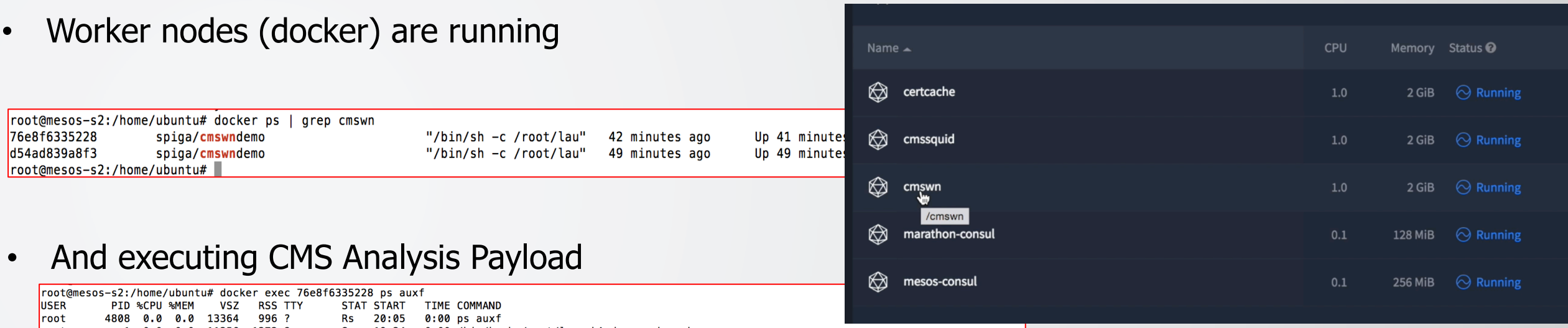


Behind the scenes

INDIGO - DataCloud

Marathon Apps Monitoring

- Worker nodes (docker) are running




```
root@mesos-s2:/home/ubuntu# docker ps | grep cmswn
76e8f6335228        spiga/cmswndemo                     "/bin/sh -c /root/lau"   42 minutes ago      Up 41 minutes
d54ad839a8f3        spiga/cmswndemo                     "/bin/sh -c /root/lau"   49 minutes ago      Up 49 minutes
root@mesos-s2:/home/ubuntu#
```

- And executing CMS Analysis Payload

```
root@mesos-s2:/home/ubuntu# docker exec 76e8f6335228 ps auxf
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root      4808  0.0  0.0  13364   996 ?        Rs   20:05   0:00 ps auxf
root         1  0.0  0.0  11356  1372 ?        Ss   19:34   0:00 /bin/bash /root/launchAndrew_spiga.sh
root       395  0.0  0.0  44072  1380 ?        S    19:35   0:00 su - glidein_pilot -c /home/glidein_pilot/runWithAndrew.sh
502        396  0.0  0.0 106112  1400 ?        Ss   19:35   0:00  \_ /bin/bash /home/glidein_pilot/runWithAndrew.sh
502        415  0.0  0.0 106508  1876 ?        S    19:35   0:00      \_ /bin/bash ./glidein_startup.sh -v std -name v3_2_11_2 -entry T3_IT_Opportunistic
fwz.cfg -descriptentry description.g6j8cz.cfg -dir . -param_GLIDEIN_Client frontend_service-v3_2_7.main -slotslayout partitionable -clientweb http://lcggw
uk:8319/vofrontend/stage/frontend_frontend_service-v3_2_7/group_main -clientsigngroup 988b368247cf8e1fd20c8bc9f8e5ffe47e1d4fea -clientdescriptgroup descri
OR_OS default -param_GLIDEIN_Collector lcggwms02.dot,gridpp.dot,rl.dot,ac.dot,uk.colon,9619.minus,9623
502       3927  0.0  0.0   9384  1476 ?        S    19:35   0:00          \_ /bin/bash /home/glidein_pilot/glide_Jwh0o2/main/condor_startup.sh glidein_cor
502       4509  0.0  0.1  95948  9608 ?        S    19:35   0:00              \_ /home/glidein_pilot/glide_Jwh0o2/main/condor/sbin/condor_master -f -pidfi
502       4511  0.0  0.0  20328  2124 ?        S    19:35   0:00                  \_ condor_procd -A /home/glidein_pilot/glide_Jwh0o2/log/procd_address -L
502       4512  0.0  0.1  96628 10372 ?        S    19:35   0:00                  \_ condor_startd -f
502       4524  0.0  0.1  96612  9292 ?        S    19:50   0:00                      \_ condor_starter -f lcggwms02.gridpp.rl.ac.uk
502       4528  0.0  0.0   9252  1396 ?        S    19:50   0:00                          \_ /bin/bash /home/glidein_pilot/glide_Jwh0o2/execute/dir_4524/c
alse --firstEvent=None --firstLumi=None --lastEvent=None --firstRun=None --seeding=AutomaticSeeding --scriptExe=None --eventsPerLumi=None --scriptArgs=[]
502       4561  0.0  0.0   9256  1428 ?        S    19:50   0:00                              \_ sh ./CMSRunAnalysis.sh -a sandbox.tar.gz --sourceURL=http
nt=None --firstRun=None --seeding=AutomaticSeeding --scriptExe=None --eventsPerLumi=None --scriptArgs=[] -o {} --oneEventMode=0
502       4587  0.0  0.1  51636 13324 ?        S    19:50   0:00                                  \_ python CMSRunAnalysis.py -r /home/glidein_pilot/glide
putFiles=False --firstEvent=None --firstLumi=None --lastEvent=None --firstRun=None --seeding=AutomaticSeeding --scriptExe=None --eventsPerLumi=None --scri
502       4730  0.0  0.0   9532  1592 ?        S    19:50   0:00                                      \_ /bin/bash /home/glidein_pilot/glide_Jwh0o2/execut
502       4769  7.9  4.9 721236 401252 ?       Sl   19:50   1:09                                          \_ cmsRun -j FrameworkJobReport.xml PSet.py
root@mesos-s2:/home/ubuntu#
```

Dynamic On Demand Analysis Service

An automated system that simplifies the process of provisioning, creating, managing and accessing a pool of heterogeneous (possibly opportunistic) computing resources.

1 **A batch system as a Service based on HTCondor** which in turn can:

a. Seamlessly be integrated in the existing HTCondor GlobalPool of CMS.

b. Deploy a standalone, auto-scaling HTCondor batch farm, also using different geographically distributed computing centers.

2 **A data analytics Infrastructure as a Service**, aiming at:

a. Extending the current standalone Big Data solutions based on Spark/Hadoop into the distributed on demand infrastructure suitable for large scale collaborations like CMS

b. Providing a platform for facilitating user access and supporting the testing and development of new Machine Learning applications

Key issues on which we have focused

- **Multi-backend support**
  - The solution is not developed for a specific backend.
    - PaaS Orchestrator; Infrastructure Manager
- **Automation**
  - Interaction with the IaaS is fully automated, from the provisioning of resources to monitoring
    - TOSCA templates; Autoscaling-self-healing
- **Authentication/authorization**
  - Identity harmonization and support integration with "proprietary" systems
    - Identity Access Management; Token Translation Service
- **software experiment encapsulation**
  - The implementation is completely experiment agnostic.
    - Ansible roles; Dockers

All that makes it basically developed a generic system, independent from the experiment and the use case.
A crucial aspect of the design development of the Thematic Service.

State of the art and experimentation

- Integration with HTCondor clusters is ready.
  - In particular the integration with *CMS-Submission Infrastructure(HTCondor Global Pool) is done*.
    - The AuthZ part , in colalboration with HTCondor team,is work in progress.
- Working on the BigData integration, in particular Spark

- The current solution is being tested in the following infrastructure:
  - **Cloud@CNAF, ReCaS@Bari, EGI FedCloud@PD** e **Perugia**

Conclusions

INDIGO - DataCloud

- The **first public INDIGO release** was issued at the beginning of August 2016.
- Its services are already available in several testbeds.
- Concrete use cases are currently being implemented by many scientific communities.
- A lot of important developments are being carried on in coordination with upstream developers, so that code maintenance is not only upon us.
- **Now looking** for early adopters / people willing to test and run INDIGO components with their applications or requirements. **If interested, please contact us**.
- We look forward to providing these components in a future **European Open Science Cloud through INFRADEV-4-2016 and EINFRA-12-2017 projects**.
  - And extending them through **EINFRA-21-2017** projects.

Thank you

INDIGO - DataCloud

https://www.indigo-datacloud.eu

Better Software for Better Science.