An integrated IaaS and PaaS architecture for scientific computing

Giacinto Donvito – INFN-Bari
On behalf of INDIGO Collaboration
INDIGO: The consortium

- 26 partners from 11 countries
- The exagon contains the partners involved in development activities
- 11M€ of total founding by EC
- 4 Big private companies

From the EC Evaluation Summary Report: “The consortium is of exceptional quality, and complementary, and with good relevant experience and skills”
INDIGO Topic of interest

• Since the very beginning we identified key issues with both Grid and Cloud technologies that prevented scientific communities an easy and optimal exploitation of data and compute resources.

• We therefore decided to propose the development of a software platform centered around two of the EINFRA-1-2014 pillars:
  • Large scale virtualization of data/compute center resources. This became the focus of INDIGO WP4.
  • Development and adoption of a standards-based computing platform (with an open software stack). This became the focus of INDIGO WP5.
How do we see distributed computing in the future

1. Ease of access and use for small and big collaborations alike.

2. Software and economic sustainability.

3. Robustness (no single points of failure).

4. Modular, scalable architecture.

5. Open source software, vendor independence, hybrid infrastructures.
## Users communities

<table>
<thead>
<tr>
<th>SIMPLIFIED IMPACT TABLE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELECTED OBJECTIVES versus REQUESTS/POTENTIAL IMPACT FOR COMMUNITIES</td>
</tr>
<tr>
<td>Q1: Development of the INDIGO Platform based on open software without restrictions on the e-Infrastructure</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Research Communities &amp; Initiatives, including ESFRIs</th>
</tr>
</thead>
<tbody>
<tr>
<td>ELIXIR INSTRUCT/ WeNMR EuroBioImaging</td>
</tr>
<tr>
<td>CTA LBT WLCG</td>
</tr>
<tr>
<td>DARIAN DCH-RP</td>
</tr>
<tr>
<td>EMSO LIFEWATCH ENES</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Examples of Applications</th>
</tr>
</thead>
<tbody>
<tr>
<td>HADDOCK GROMACS AMBER GALAXY</td>
</tr>
<tr>
<td>MIDAS, IRAF, IDL, Geant4 ROOT/PROOF Geant4</td>
</tr>
<tr>
<td>Fedora Digital Libraries</td>
</tr>
<tr>
<td>Delft3D R-Studio TRUFA MATLAB</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Life Sciences</th>
<th>Physical Sciences &amp; Astronomy</th>
<th>Social Sciences &amp; Humanities</th>
<th>Environmental Sciences</th>
</tr>
</thead>
<tbody>
<tr>
<td>RELEVANT</td>
<td>CRITICAL</td>
<td>RELEVANT</td>
<td>CRITICAL</td>
</tr>
<tr>
<td>CRITICAL</td>
<td>RELEVANT</td>
<td>MINOR</td>
<td>CRITICAL</td>
</tr>
<tr>
<td>CRITICAL</td>
<td>CRITICAL</td>
<td>RELEVANT</td>
<td>RELEVANT</td>
</tr>
<tr>
<td>CRITICAL</td>
<td>RELEVANT</td>
<td>MINOR</td>
<td>RELEVANT</td>
</tr>
<tr>
<td>CRITICAL</td>
<td>RELEVANT</td>
<td>CRITICAL</td>
<td>RELEVANT</td>
</tr>
</tbody>
</table>

- Design and development of a Platform providing advanced users and community developers a powerful and modern environment for development work. This includes programming and scripting tools, and composition of custom applications and software deployment.
- Developing a framework to enable the transparent execution on remote e-infrastructures of existing popular applications like MATLAB / OCTAVE, ROOT, MATHEMATICA, or R-Studio.
- Provide the services and tools needed to enable a secure composition of services from multiple providers in support of scientific applications.
- Develop and implement a solution that is able to deploy in a transparent and powerful way both services and applications in a distributed and heterogeneous environment made by several different infrastructures (EGI Grid and Federated Cloud, IaaS Cloud, Helix Nebula, HPC clusters).
- Develop the capability in the PaaS to provide unified data access despite geographical location of data, including APIs access, based on existing standards, or virtually mount like a POSIX device to worker node, cloud virtual machines, personal computer etc.
INDIGO WPs

**WP1 – Administrative and Technical Management**
Leader Partners: INFN, CSIC

**WP2 – Definition of Support to Research Communities**
Leader Partners: LifeWatch, EGI.eu

**WP3 – Software Management and Pilot services**
Leader Partner: LIP, CEA

**WP4 – Resource Virtualization**
Leader Partner: DESY, KIT

**WP5 – PaaS Platform**
Leader Partner: INFN, Cyfronet

**WP6 – Portal, Workflows and User Interfaces**
Leader Partners: PSNC, INFN

Final product/Deployed App

Feedback/Revision
INDIGO approach

• Based on **Open Source** solutions

• widely **supported** by big communities

• whenever possible exploit **general solutions** instead of specific tools/services
  • or put effort in **increasing the generality** of tools developed in a given community
  • this will be important for **sustainability** of the architecture

• ensure that the framework offered to final users, as well as to developers, will have a **low learning curve**
  • **existing software suites** like ROOT, OCTAVE/MATLAB, MATHEMATICA or R-STUDIO, **will be supported** and offered in a transparent way
Scientific Computational Portal “as a Service”

- LBaaS
- Use Case Portal / Access services
- Use Case Portal / Access services
- LRMS front-end
- LRMS WN
- LRMS WN
- Storage
- Site A
- Site-level Scheduling
- VPNaaS
- LRMS WN
- LRMS WN
- Storage
- Site B
- TOSCA end-point
- TOSCA end-point
- VMI repo
- Container repo
- End user
- AAI
- App dev
- TOSCA Spec
- Scheduling

- ELIXIR
- Haddock
- CIRMMP
- FedCloud
- DARIAH
- INAF-LBT
- CMCC – ENES
- CTA
- ALGAE-BLOSSOM
- INGV - MOIST
Software as a Service Use Case

Interactive console

Data Analytics Services

WN

Data

Site A

TOSCA end-point

External application

End user

AAI

App dev

TOSCA Spec

Scheduling

VMI repo

Container repo

Site B

TOSCA end-point

Data

WN

Data Analytics Services

Data

CIRMMP

INAF-LBT

CMCC – ENES

ALGAE-BLOSSOM

INGV - MOIST

INDIGO-DataCloud RIA-653549
Example: R-Studio

- R-Studio Server provides a browser-based interface (RStudio IDE) to a version of R running on a remote Linux server.
  - How to deploy it using INDIGO?

  - The user selects the template for R-studio and provides further requirements (optional)
    - CPU, RAM, etc.
**INDIGO overall architecture**

**Color codes:**

**Yellow:** implementation based on already available solution to be improved/changed;

**Orange:** Completely new services to be implemented
Interfaces among WPs

This will be described in this talk.
WP5: PaaS Platform Development

- **PaaS** layer is provided by WP5
  - It is realized by means of a high-available and scalable µServices cluster
    - Hosted by Kubernetes
  - It is able to provide two main approach:
    - High-Level automatized IaaS
    - Real PaaS services and application execution
Deployment of Customized Virtual Infrastructure using INDIGO

1. Access
2. Authenticate
3. Select / Customize
4. Deploy
5. Status
6. Prioritize
7. Agree
8.a. Transfer (Optional)
8.b. Transfer (Optional)
9.a. Delegate Deploy
9.b.1. Delegate Deploy
9.b.2. Provision & Configure
10. Deliver VMs
11. Access Service/Infrastructure
12. Scale out

Service / Applications (MSA) Service
Infrastructure Manager
Managed Services/
Applications (MSA) Service
Repository
TOSCA-compliant Templates

Deployment of User Application / Service

External Cloud

Virtual Infrastructure for
User Application / Service

INDIGO - DataCloud

15
On Standards

- **OASIS TOSCA** (Topology and Orchestration Specification for Cloud Applications) 1.0 (11/2013)
  - Interoperable description of application and infrastructure cloud services, the relationships between parts of the service, and the operational behavior of these services (e.g., deploy, patch, shutdown) independent of the supplier creating the service, and any particular cloud provider or hosting technology.

More on TOSCA (I)

What is TOSCA?

A language for defining Service Templates...

...including a Topology Template describing the structure of a service

...including the definition of plans for orchestrating the application

Packaging format (CSAR) for packaging models and all related artifacts.

Source: https://wiki.openstack.org/w/images/a/a1/TOSCA_in_Heat_-_20130415.pdf
On Mesos / Marathon /Chronos

- Marathon/Chronos run on top of a Mesos cluster
- Enables scale-out of services by deploying additional containers on spare nodes.
- Marathon will be used to instantiate the Long Running Services and keep them monitored and running
  - It will take care of recover in case of failure
- Chronos will be used in order to deal with application execution:
  - It is able to deal with dependency/re-executing the application in case of failure
- The Mesos cluster will be used in order to share the resources among different use-cases
Enhanced IaaS
IaaS on steroids

• Features missing in (some) OpenSource IaaS Cloud Management Frameworks
  • Lack or insufficient support for running containers as lightweight resources
  • Naïve scheduling policies.
  • Lack of support for standards-based orchestration

• Objectives
  • Improve IaaS to accommodate PaaS workloads
  • Aiming for features interesting for the broader community, not only for the project
  • → integration with upstream tools.
Lack of container support

• Containers can be used as lightweight virtual machines.
  • Offer better portability.
  • Lower performance overhead.

• Support in OpenStack in place, but features missing.

• Support in ONE not available.
  • Develop a new driver to promote containers as first-class resources.

• Not aiming on container orchestration (e.g. Kubernetes).
  • This will be done by other components and areas
Naïve scheduling

• CMF developments driven by industry needs.
  • Normally scheduling is done on a FIFO basis.

• Scientific applications have unique resource provisioning requirements.

• User challenges:
  • Static quotas, difficult to overcoming.
  • Opportunistic usage not possible.

• Resource provider challenges:
  • Lack of proper dynamic partitioning → not possible to get a good utilization
  • Not possible to prioritize different workloads.

• Two different approaches: fair-share scheduling and spot instances support
Fair-share scheduling

• VMs do not have a duration.
  • They cannot be terminated

• No queuing mechanism.
  • Difficult to prioritize tasks.

• Synergy is being developed.
  • External service that intercepts user requests.
  • Instances launched through Synergy have are associated with a WALL time.
  • They queued according to previous usage (fair-sharing).
  • They are terminated after the WALL time is expired.
Enhanced IaaS

![Diagram of Enhanced IaaS system]

- RESTful API
- Synergy
  - Queue M.
  - FairShare M.
  - Quota M.
- P. Queue
- Nova DB
- AMQP
- Keystone
- Nova-scheduler
- Nova-compute
Spot-instances support

- Implemented in commercial cloud offerings: AWS and Google Cloud Engine.
- Instances that can be terminated by higher priority requests → priority can be determined by the bid prize of by another mechanism.
- They can be always terminated by non-spot instances.
- It makes possible opportunistic usage of the cloud
- Users can fill the idle resources, but higher priority task can enter.
Standards-based orchestration

• For OpenStack, Heat uses the Heat Orchestration Template (HOT).
• For ONE, the Infrastructure Manager (IM) uses the Resource and Application description Language (RADL).
• Support for using TOSCA in both orchestrators is being developed:
  • Approach: translation between TOSCA templates and the native languages → initial translator already available for OpenStack
  • Initial support at the CLI level
  • Studying the possibility enabling APIs to receive TOSCA documents.
Progress so far (IaaS)

- Container support
- Initial driver for ONE has been developed.
- OpenStack testbed being deployed.
- OCCI support ongoing.
- Fair-share scheduling
  - Preparing code for initial publishing.
  - Testbed deployed and currently under testing.
  - Problems with SLURM Priority Multi-factor, adapting to a new algorithm.
Progress so far (IaaS)

• Standards-based orchestration
  • Contacts with OpenStack developers to include TOSCA on Heat
  • Improved TOSCA-parsers to support INDIGO use cases

• Spot-instances support
  • Initial blueprint presented to OpenStack, but discussion still ongoing.
  • Initial prototype working.
  • Studying how to expose functionality trough OCCI.
Progress so far (IaaS)

• Standards-based orchestration
  • Contacts with OpenStack developers to include TOSCA on Heat
  • Improved TOSCA-parsers to support INDIGO use cases

• Spot-instances support
  • Initial blueprint presented to OpenStack, but discussion still ongoing.
  • Initial prototype working.
  • Studying how to expose functionality through OCCI.
Progress so far (PaaS)

- PaaS Architecture fully defined
- PaaS Orchestrator technology is already defined
  - Implementation of the first workflow is already going on
- Define additional non-normative node types for TOSCA
- Monitoring technology defined
- Technology for deploying of Long Running Services and Application is defined
- Hosting of μServices
Conclusions

• First official release will be: end of July next year
• We will start make available some services as soon as they are ready enough to be tested
• All the changes on the already available projects, will be pushed back to the official releases.
  • OpenStack, OpenNebula, dCache, Onedata, Mesos, Accounting, QoS/SLA, etc
Questions?

Giacinto DONVITO

giacinto.donvito@ba.infn.it