**On-demand large-scale data analytics platform with INDIGO-DataCloud PaaS**

Big data processing and analytics are presently among the most important trends in the research and industry landscape: the growing availability of data collected from a multitude of sources is a real treasure in many fields (health care, bioinformatics, human and social sciences, high-energy physics, etc.), but it comes with a series of critical challenges as well. Building reliable algorithms that are able to extract precious information from data is a crucial task. Fortunately, researchers can exploit data analysis platforms designed to simplify the development of these algorithms. Apache Spark [1] has rapidly emerged as the platform of common use and the natural successor to Hadoop.

The development of tools allowing the deployment of data analytics platforms in the most straightforward and automated way is the next big challenge.

One of the main achievements of the H2020 project "INDIGO-DataCloud" [2] is to enable scientific communities to access data and computing resources provided by distributed e-infrastructures in an easy and transparent way, removing operational complexities for users.

The Platform-as-Service (PaaS) layer of the INDIGO-Datacloud software stack provides advanced features for orchestrating the deployment of complex virtual infrastructures, dockerized services or batch jobs on distributed cloud environments: it is able to provision the required resources automatically over heterogeneous and/or hybrid (public and private) cloud infrastructures.

The INDIGO Orchestrator, the core component of the PaaS layer, implements complex workflows to coordinate the deployment of the requested cloud services.

The deployment requests are expressed using TOSCA (Topology and Orchestration Specification for Cloud Applications [3]), an OASIS standard to specify the topology of services provisioned in IT infrastructures; a TOSCA template is submitted to the REST API endpoint of the INDIGO Orchestrator that coordinates resource deployment using information about the health status and capabilities of underlying IaaS, as well as their resource availability, QoS/SLA constraints, the status of the data files and of storage resources needed by the service/application. This process allows to achieve the best allocation of the desired resources among multiple IaaS sites.

The adoption of the TOSCA templating language for the description of the cloud services topology brings the big advantage of being inherently infrastructure-agnostic, thus ensuring portability across multi-cloud or hybrid cloud platforms.

The INDIGO Project has developed several TOSCA templates covering different use-cases with different levels of complexity: a special focus has been put on large-scale data clusters (like Hadoop [4]) and computing clusters (like Apache Mesos [5], Slurm [6], etc.) enhancing their capabilities. Self-provisioning of the resources, automated configuration, cluster elasticity are some of the cutting-edge features provided by the INDIGO PaaS.

Using the INDIGO PaaS it is possible to spin-up in a few minutes a complete big data infrastructure consisting of an automatically managed pool of cloud resources that are auto-configured for running Spark workloads on a Mesos cluster. Apache Mesos [5] is an open-source cluster manager that provides efficient resource isolation and sharing across distributed applications ensuring automated self-healing and scalability.

Running Spark on Mesos is not a trivial task. Some documentation is provided on the official web site [7] but it requires some knowledge on how to install, configure and tune both Mesos and Spark components.

Using the INDIGO PaaS, end-users need only to specify some input parameters that define the cluster size (number of master/slave nodes, cpus, memory, etc.); once the deployment is carried out, the endpoint of the cluster is returned to the user.

The cluster is installed and configured automatically using a set of ansible roles (shared on Ansible-Galaxy under the indigo-dc namespace [8]) that are referenced within the TOSCA template (artifacts); most of the services are started as containers from docker images published on the docker hub (under the indigodatacloud organization [9]).

The INDIGO Mesos cluster that gets automatically and dynamically instantiated by the INDIGO PaaS comes with some important features:

- High-availability of the cluster components:
    - Leader election among master nodes managed by Zookeeper [10];
    - HA Load-balancing;
- Configuration of the two main Mesos Frameworks: Marathon [11] for managing Long-Running Services and Chronos [12] for running scheduled jobs: it is possible to enable/disable the configuration of these frameworks setting some flags in the TOSCA template;
- SSL support and basic authentication (username/password) enabled for cluster endpoints;
- Service discovery through Consul [13] that provides also DNS functionality and health checks;
- Cluster elasticity through INDIGO CLUES plugin [14]: the cluster can automatically shrink or expand depending on the tasks queue;
- Persistent storage for long-running services through rex-ray plugin [15].

On top of this Mesos cluster, the Apache Spark framework is automatically configured and deployed.

The user can choose to deploy only the Spark Mesos Cluster Dispatcher and then submit jobs through the spark-submit utility or he can choose to deploy Apache Zeppelin [16], a web-based notebook platform that enables interactive data analytics with interactive data visualizations and notebook sharing. The Dispatcher and the web application (Zeppelin) are deployed and managed through Marathon (Mesos framework for long-running services) using the Spark docker image that we have prepared and pushed on Docker Hub, whereas the non-interactive Spark jobs can be easily submitted through Chronos (Mesos framework for batch-like tasks) using the same docker image.

Both HDFS [4] and Swift [17] storage options are supported by our TOSCA template: the user can specify an account on a Swift Object Store to be used by Spark applications whereas for the HDFS

he can use an already existing cluster (specifying its URL) or can request the creation of the distributed file system in parallel with the creation of the Mesos cluster using the service composition pattern supported by TOSCA.

With the same approach it is possible to include the automatic deployment of a monitoring system for aggregating the cluster logs and metrics: a monitoring node is automatically installed and configured with Elasticsearch [18] for storing the collected data and Grafana [19] for displaying them (the Elasticsearch datasource is automatically configured along with some useful dashboards). The service logs are sent to the database through Fluentd [20] whereas the metrics are collected through Metricbeat and then sent directly to Elasticsearch.

Leveraging the same technologies and components we have also developed a simple and automated solution for creating, managing and accessing a dynamic on-demand analysis cluster for LHC (CMS) data analysis workflow.

The outcomes of performance and scalability tests will be presented.

A recorded demo showing the automatic deployment of Spark on Mesos platform through the INDIGO PaaS is available at the following link: https://goo.gl/hQNSk8

**References:**


[1] https://spark.apache.org/
[2] https://www.indigo-datacloud.eu/
[3] https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=tosca
[4] http://hadoop.apache.org/
[5] http://mesos.apache.org/
[6] https://slurm.schedmd.com/
[7] https://spark.apache.org/docs/latest/running-on-mesos.html
[8] https://galaxy.ansible.com/indigo-dc/
[9] https://hub.docker.com/u/indigodatacloud/
[10] https://zookeeper.apache.org/
[11] https://mesosphere.github.io/marathon/
[12] https://mesos.github.io/chronos/
[13] https://www.consul.io/
[14] https://indigo-dc.gitbooks.io/clues-indigo/content/
[15] https://rexray.readthedocs.io/en/stable/
[16] https://zeppelin.apache.org/
[17] https://docs.openstack.org/swift/latest/
[18] https://www.elastic.co/products/elasticsearch
[19] https://grafana.com/
[20] https://www.fluentd.org/