PROCEEDINGS OF SCIENCE

DODAS: How to effectively exploit heterogeneous clouds for scientific computations

Daniele Spiga[a]*, Marica Antonacci[b], Tommaso Boccali[c], Alessandro Costantini[d], Diego Ciangottini[a], Giacinto Donvito[b], Cristina Duma[d], Matteo Duranti[a], Valerio Formato[a], Luciano Gaido[e],Diego Michelotto[d], Davide Salomoni[d], Mirco Tracolli[a]

[a]*INFN sezione di Perugia*
[b]*INFN sezione di Bari*
[c]*INFN sezione di Pisa*
[d]*INFN CNAF*
[e]*INFN sezione di Torino*

E-mail: daniele.spiga@pg.infn.it

Dynamic On Demand Analysis Service (DODAS) is a Platform as a Service tool built combining several solutions and products developed by the INDIGO-DataCloud H2020 project. DODAS allows to instantiate on-demand container-based clusters. Both HTCondor batch system and platform for the Big Data analysis based on Spark, Hadoop etc, can be deployed on any cloud-based infrastructures with almost zero effort. DODAS acts as cloud enabler designed for scientists seeking to easily exploit distributed and heterogeneous clouds to process data. Aiming to reduce the learning curve as well as the operational cost of managing community specific services running on distributed cloud, DODAS completely automates the process of provisioning, creating, managing and accessing a pool of heterogeneous computing and storage resources. DODAS was selected as one of the Thematic Services that will provide multi-disciplinary solutions in the EOSC-hub project, an integration and management system of the European Open Science Cloud starting in January 2018. The main goals of this contribution are to provide a comprehensive overview of the overall technical implementation of DODAS, as well as to illustrate two distinct real examples of usage: the integration within the CMS Workload Management System and the extension of the AMS computing model.

International Symposium on Grids and Clouds (ISGC) 2018 in conjunction with Frontiers in Computational Drug Discovery
16-23 March 2018
Academia Sinica, Taipei, Taiwan

**Speaker*

PoS(ISGC 2018 & FCDD)024

1.Introduction

Nowadays the demands for computing capacity, flexibility and portability is increasing in almost all the scientific domains. The Big Data analysis is one of the most important trends from the perspectives of both industry and research, spanning from medicine to social sciences, information security, passing through the high energy physics.

The Large Hadron Collider LHC [1] represents an important use case: as its scientific programme advances, will requires a huge increase of the power and the intensity. The computing requirements are expected to go beyond what can be supported by general technology trends. In this context we foresee that new technologies, such as, virtualization and cloud computing represent a possible contribution to such important challenges. Virtualization techniques, in fact, have the potential to make available unprecedented amount of resources and related services to researchers, addressing new complex requirements coming from diverse scientific communities.

However, the integration of Cloud computing together with well established computing infrastructures of the experiments is not always an easy task to accomplish and, sometimes, it might require too much effort, mostly for small communities.

The INDIGO - DataCloud [2] project, in the context of HORIZON 2020, developed services and platforms, based on open source solutions, in order to address exactly these kinds of challenges. More in detail INDIGO provides technical solutions, generic services as well as guidelines to support interoperability across hybrid Cloud infrastructures at the IaaS, PaaS and SaaS levels involving different aspects in the cloud compute, storage and network areas. In this respect it is important to highlight that INDIGO was not intended to develop a one-size-fits-all solution, but rather a set of building blocks to allow composition and integration with specific use cases, scientific needs and requirements.

The next section describes the principal scientific motivation to develop the DODAS service. Section 3 provides an overview of the overall architecture of DODAS while section 4 shows the integration with two scientific communities in the context of High Energy Physics. The early results from using DODAS are given in section 5.

2. Motivations

Although the developed architecture is completely experiment agnostic, the genesis of DODAS is in the field of High Energy Physics and there main motivations for its implementation were to provide:

- A comprehensive approach to the opportunistic computing. It offers the possibility to orchestrate multiple centers, which might include for example general purposes campus facilities, public or private clouds, to gather all available computing and storage resources.
- A simple solution for elastic computing site extensions, e.g. extension of allocated resources in order to absorb peaks of usage.
- An easy and controlled procedure to dynamically instantiate a spot ‘Data Analysis Facility’, for example a mission specific site (e.g. to address data analysis of a group before a conference). From the physicist perspectives this is meant as the generation of an ephemeral site as a Services to share computing and data resources with collaborators.

Moreover since the beginning DODAS aims to provide a service fully oriented to the end users. Therefore, it does not require any deep expertise in system administration since scientists are not supposed to have advanced IT skills. In other words, DODAS has been designed to provide a tool at Platform as a Service tool that allows the setup of complex services and applications such as Web service, Batch systems, databases, Big Data platforms etc, as friendly as it is creating a single VM in any cloud provider.

3. Architectural Overview

As anticipated the development of DODAS started in the context of INDIGO project and thus the architecture is heavily based on INDIGO building blocks. Guided by the motivations summarized in section 2, the architectural pillars have been identified as follow:

- Automation: this is a primary pillar. The service must automate the whole flow of resources provisioning, services deployment, configuration, monitoring and auto-scaling.
- Application agnostic: the service must not be tight to any specific technology and/ or to a specific platform and/or to a resources provisioning paradigm.
- Portability: there is not a unique provider of resources nor a unique customer and thus not a single use case to cope with. The system must be agnostic both of clouds and of customer's application.
- Security: users must delegate the service to authenticate both with resource providers and with computing infrastructure of communities that might even expose to the dynamic cluster, their proprietary services (e.g. distributed storages, metadata-services, etc).

Moreover, one of the main architectural goals of DODAS, has been to provide a high level of modularity, a key to a generic applicability. Being modular, the architecture provides the ability to easily customize the workflow depending on the community computational requirements.

Starting from these requirements, a match with INDIGO products, available in the service catalog has been done in order to identify the most relevant products, suitable for the DODAS core system. As result several services have been identified:

- The PaaS Orchestrator [3] which has the role of taking the requests related to application or service deployment coming from the user expressed using TOSCA [4], the OASIS standard to specify the topology of services provisioned in IT infrastructures. Based on the user requirements (typically expressed in the TOSCA template), the Orchestrator has the role the best infrastructure (IaaS) for the deployment taking into account information about user's SLAs the availability and the health status of the IaaS services. The actual interaction with the infrastructure is delegated to the Infrastructure Manager (IM) [5].
- The IM is in charge to deploy complex and customized virtual infrastructures on different IaaS Cloud deployment, providing an abstraction layer to define and provision resources in different clouds and virtualization platforms. IM enables computing resource orchestration using TOSCA protocol. Moreover, it eases the access and the usability of IaaS clouds by automating the VMI (Virtual Machine Image) selection, deployment, configuration, software installation, monitoring and update of the virtual infrastructure,
- The glue of the implemented flow is the Identity and Access Management service (IAM) [6]. IAM provides a layer where identities, enrolment, group membership,

attributes and policies to access distributed resources and services can be managed in a homogeneous and interoperable way. It supports the federated authentication mechanisms behind the INDIGO AAI. The IAM service provides user identity and policy information to services so that consistent authorization decisions can be enforced across distributed services. Identity and Access Management is provided through multiple methods (SAML [7], OpenID Connect [8] and X.509 [9]) by leveraging on the credentials provided by the existing Identity Federations (i.e. IDEM [10], eduGAIN [11], etc). The support to Distributed Authorization Policies and Token Translation Service will guarantee selected access to the resources as well as data protection and privacy.

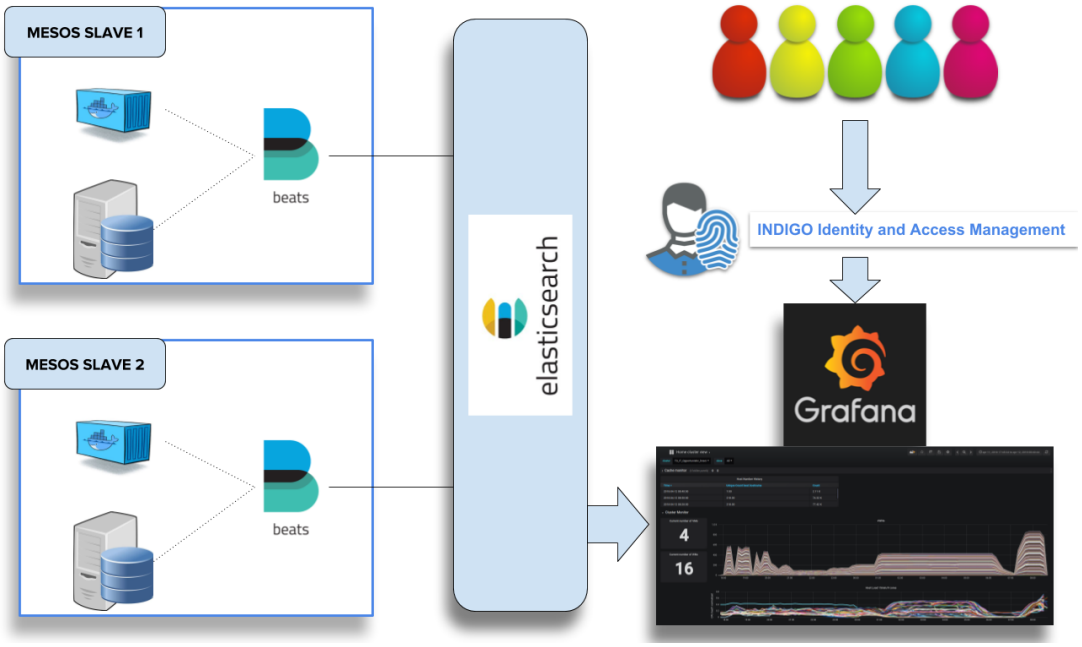

Fig.1: DODAS monitoring stack schema, from metrics collection to dashboard results. The access permissions to these dashboards has been also integrated with INDIGO Identity and Access Management credentials, so that whoever creates the cluster has also the possibility to manage its own dashboard seamlessly.

The resource abstraction and the full automation, one of the pillar stone of DODAS, are thus implemented combining together the PaaS Orchestrator and the IM. The latter provides the support for several providers like, for example, OpenStack [12], OpenNebula [13], Amazon AWS [14] and Microsoft Azure [15], while the PaaS Orchestrator represents the DODAS entrypoint. As such it is directly exposed to the end user who is required to provide properly configured TOSCA templates. The cluster setup and the services configuration are automated using Ansible [16] recipes. The TOSCA and Ansible combination guarantees an easy procedure to describe complex computing infrastructures. TOSCA templates are made available through git repository.

Thanks to the modularity of the described architecture, DODAS project extended the INDIGO provided building blocks with a monitoring system. As shown in Fig 1. the monitoring is based on Metricbeat [17], Elasticsearch [18] and Grafana [19]. Metricbeat is a lightweight

agent that is installed on the resource layer in order to collect metrics from the operating system and from processes running on the host. Standalone modules are used to configure the polling of different metrics categories. The agents can be configured to send information to different backends although they were mostly designed to use elasticsearch as backend, that, indeed, is the current default in DODAS.

4. Integrating scientific computing with DODAS

Although originally designed for the Compact Muon Solenoid (CMS) [20] Experiment at LHC, DODAS is currently being adopted by the Alpha Magnetic Spectrometer (AMS) [21] astroparticle physics experiment mounted on the ISS. This adoption is happening in the context of EOSC-hub [22] project where DODAS is participating as one of the nine Thematic Services [23]. From the AMS community perspectives DODAS represents a solution to transparently exploit Cloud computing with almost zero effort.

Aiming at a generic design and, also, for the sake of conceptual simplicity, the strategy adopted to deploy the software components as well as the applications, is based on a complete factorization between the baseline Mesos, Marathon [24] and the experiment specific packages. Each one is also managed separately as to scale based on specific needs.

Regarding the experiment specifications, we distinguish among dependencies to insulate as containers and those better suitable for a deployment on virtual hosts. Concrete examples are GRID Middleware dependencies, embedded at the level of Dockers [25], with respect to CVMFS [26] which is mounted on Virtual Machine while user own libraries are shipped through HTCondor [27] within Docker containers running condor_master daemon. All the components, including HTCondor, squid proxy, security cache, are containerized as Dockers.

4.1 CMS Use Case

The CMS Workload Management system is built over a “Global Pool” by meaning a single HTCondor pool covering all GRID computing processing resources, ~150,000 dedicated CPUs. CMS Global pool has over 30 schedds connected to it in total, and over a dozen of them are active at the same time. Resource provisioning is based on GlideinWMS [28] which has the knowledge of all the sites supporting CMS computing. DODAS implements a model which de facto bypasses the resource provisioning glidein based and, upon user request, creates a CMS ephemeral WLCG [29] site which auto-registers to the Global Pool. As a regular CMS site, it includes CVMFS setup, squid proxy, Trivial File Catalog, X.509 certificates, credential caching, renewal mechanisms and, of course properly configured condor. The resource auto-registration is based on X.509 certificate obtained from TTS. From that point on, the flow has the very same pattern of a regular HTCondor based workflow: once requirements of users’ jobs match at least a resource offer (job slot), payloads start their execution. It is important to note how everything is transparent to CMS users. They just submit analysis jobs using regular CRAB [30] toolkit. Finally data input is managed through the XrootD [31] federation of CMS. The described mechanism has been tested using several cloud providers. Real analysis jobs have been used to validate the described integration. Figure 2 shows how DODAS ephemeral site seamlessly joins the HTCondor Global Pool.

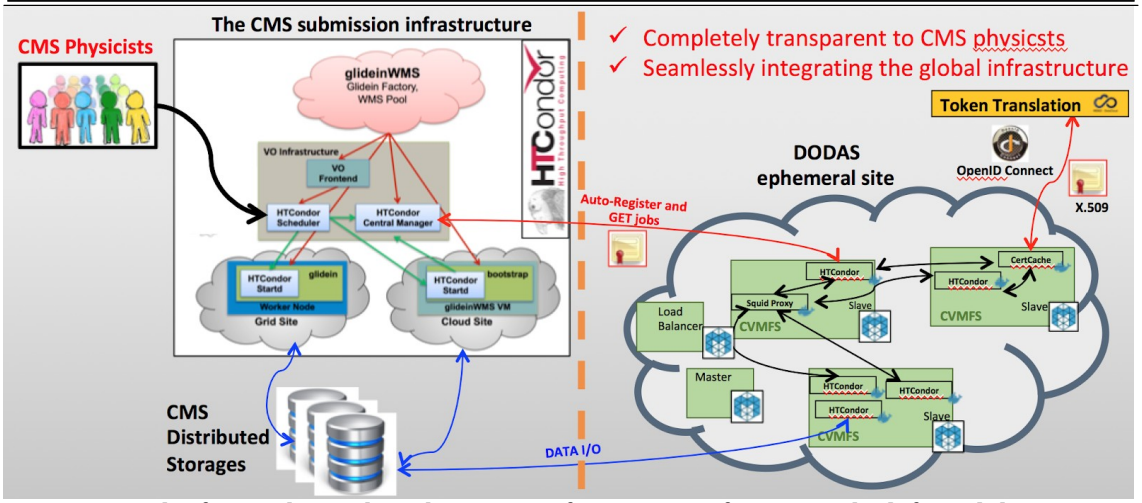

Fig.2: The figure shows the submission infrastructure of CMS on the left, and the DODAS ephemeral site on the right. Red arrow represent the communication channel among HTCondor services. Blue colored arrow represent the data ingestion flow.

4.2 AMS-02 Use Case

AMS doesn't operate a central service to manage workflows and thus it relies on batch systems and technology available on the sites where resources are made available to the collaboration. That said what AMS requires, at first stage, is to access a batch system in any Cloud to process remote data, and thus by adopting DODAS AMS will have an enabler for a Batch-system as a Service.

To accomplish this requirement a set of Ansible roles and TOSCA templates have been developed. TOSCA and Ansible allow to automate the deployment of a HTCondor batch system which in turn is executed as Marathon application over Mesos managed resources. In the case of AMS The HTCondor structure is made of a Central Manager, a submitter node and several worker nodes. The Central Manager is stateless and has the task to coordinate the jobs that the users want to do, so it connects the submitter node to the worker nodes. The submitter node is special because it has also the environment where each user can prepare his own jobs and of course it is similar to the environment present in the worker nodes. These ones have the task to execute the user jobs and return the results to the submitter. All the nodes are managed as Docker instances: there is a core image that is extended specifically for the HTCondor cluster. Both user compiled software and centrally managed libraries are distributed through CVMFS. Data ingestion is managed through XrootD, similarly to the CMS use case. The Figure 3 show how DODAS bring the cloud to the AMS users with almost zero effort and without any change to the AMS model.

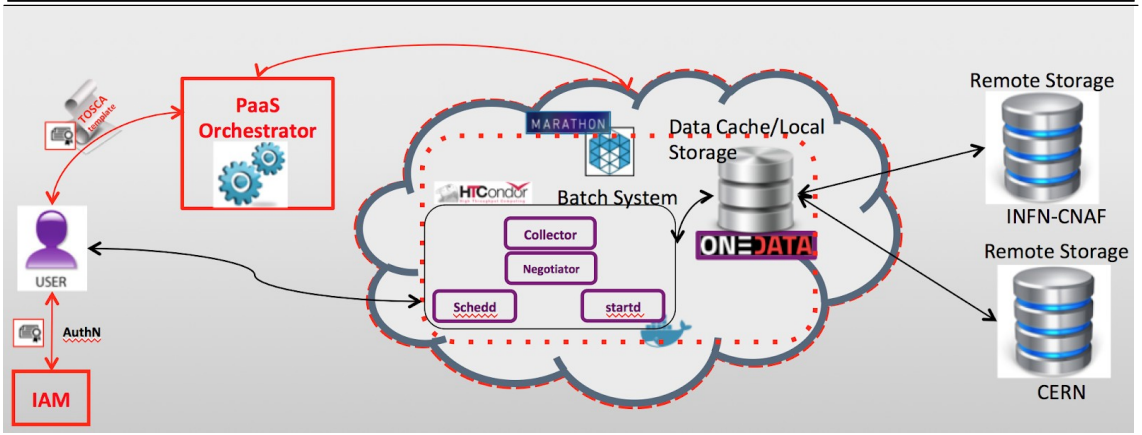

Fig.3 This figure shows how DODAS is being integrated with the AMS data processing workflow.

5. Conclusion and Outlook

DODAS service has been developed during the past 20 months. It is currently in production within the CMS experiment as well as part of EOSC-hub project as a Thematic Service. It is in this scope that a preliminary version of the DODAS integration with AMS-02 experiment has been developed.

CMS experiment successfully exploited both public (Microsoft Azure and OpenTelecomCloud) and private cloud (openstack based) using DODAS, as a solution for opportunistic computing. The early adoption shows an overall good stability and self-healing capabilities. Job success rate has been measured and compared with regular GRID based infrastructure and no differences have been observed.

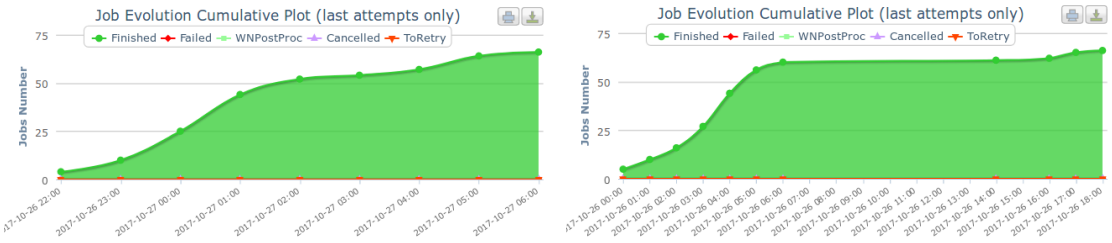

Fig.3: Comparison of CMS payload success rate over time using DODAS cluster (on the left) and regular CMS Tier2 on the right (T2_IT_Bari)

The key feature of such a service is to allow the exploitation of any cloud with almost zero effort, from the end user perspectives. In the context of EOSC-hub project there is an active process for attracting new scientific communities seeking to easily exploit Clouds

ACKNOWLEDGMENTS

The authors would like to thank European Union's Horizon 2020 research and innovation programme for financial support, under grant agreement RIA 777536.

References

[1] L. Evans and P. Bryant (editors), "LHC Machine", JINST 3 (2008) S08001, doi:10.1088/1748-0221/3/08/S08001

[2] D.Salomoni et al. INDIGO-Datacloud: foundations and architectural description of a Platform as a Service oriented to scientific computing. ArXiv:1603.09536.

[3] See https://indigo-dc.gitbooks.io/indigo-datacloud-releases/content/ indigo2/orchestrator2.html

[4] See http://docs.oasis-open.org/tosca/TOSCA/v1.0/os/TOSCA-v1.0- os.html

[5] See https://indigo-dc.gitbooks.io/indigo-datacloud-releases/ content/indigo2/im2.html

[6] See https://indigo-dc.gitbooks.io/ indigo-datacloud-releases/content/indigo2/iam2.html

[7] Security Assertion Markup Language. http://docs.oasis-open.org/security/saml/Post2.0/ sstc-saml-tech-overview-2.0.html

[8] See. openid.net/connect/

[9] See https://www.ietf.org/rfc/rfc5280.txt

[10] See https://www.idem.garr.it/

[11] See https://www.geant.org/Services/Trust_identity_and_security/ eduGAIN

[12] See https://www.openstack.org/

[13] See http://opennebula.org/

[14] See https://azure.microsoft.com/

[15] See https://aws.amazon.com

[16] See https://www.ansible.com/

[17] See https://www.elastic.co/products/beats/metricbeat

[18] Clinton Gormley and Zachary Tong. 2015. Elasticsearch: The Definitive Guide (1st ed.). O'Reilly Media, Inc..

[19] See http://grafana.org/

[20] S. Chatrchyan et al. CMS Collaboration 2008 The CMS experiment at the CERN LHC J. Inst. 3 S08004

[21] The Alpha Magnetic Spectrometer (AMS) on the International Space Station: Part I – results from the test flight on the space shuttle.

[22] http://www.eosc-hub.eu/

[23] https://confluence.egi.eu/display/EOSC/EOSC-hub+service+catalogue

[24] See https://open.mesosphere.com/

[25] Dirk Merkel. Docker: lightweight linux containers for consistent development and deployment. Linux Journal, 2014(239):2, 2014.

[26] P Buncic et al 2010 J. Phys.: Conf. Ser. 219 042003

[27] Thain D, Tannenbaum T and Livny M 2004 Distributed computing in practice: the condor experience Concurrency: Pract. Exper. 17 323-356

[28] Balcas J et al. 2015 Using the glideinWMS system as a common resource provisioning layer in CMS J. Phys. Conf. Ser 2015 J. Phys.: Conf. Ser. 664 062030

[29] WLCG - Worldwide LHC Computing Grid, http://wlcg.web.cern.ch.

[30] D Spiga, CRAB3: Establishing a new generation of services for distributed analysis at CMS. Journal of Physics Conference Series , 396(3):032026, December 2012.

[31] Dorigo A., Elmer P., Furano F., and Hanushevsky A. XROOTD - a highly scalable architecture for data access. WSEAS Transactions on Computers, 4(4):348–353, April 2005