EPJ Web of Conferences **214**, 07027 (2019) https://doi.org/10.1051/epjconf/201921407027
CHEP 2018

Exploiting private and commercial clouds to generate on-demand CMS computing facilities with DODAS

Daniele Spiga[1,1], *Marica* Antonacci[2], *Tommaso* Boccali[3], *Andrea* Ceccanti[4], *Diego* Ciangottini[1], *Riccardo* Di Maria[5], *Giacinto* Donvito[2], *Cristina* Duma[4], *Luciano* Gaido[6], *Álvaro* López García[7], *Aida* Palacio Hoz[7], *Davide* Salomoni[4], *Mirco* Tracolli[1]

[1] Istituto Nazionale di Fisica Nucleare, 06123 Perugia, Italy
[2] Istituto Nazionale di Fisica Nucleare, 70126 Bari, Italy
[3] Istituto Nazionale di Fisica Nucleare, 56127 Pisa, Italy
[4] Istituto Nazionale di Fisica Nucleare CNAF, 40127 Bologna, Italy
[5] Imperial College London, South Kensington, London SW7 2AZ, UK
[6] Istituto Nazionale di Fisica Nucleare, 10125 Torino, Italy
[7] Instituto de Física de Cantabria (CSIC-UC), 39005 Santander, Cantabria, Spain

Abstract. Minimising time and cost is key to exploit private or commercial clouds. This can be achieved by increasing setup and operational efficiencies. The success and sustainability are thus obtained reducing the learning curve, as well as the operational cost of managing community-specific services running on distributed environments. The greater beneficiaries of this approach are communities willing to exploit opportunistic cloud resources. DODAS builds on several EOSC-hub services developed by the INDIGO-DataCloud project and allows to instantiate on-demand container-based clusters. These execute software applications to benefit of potentially "any cloud provider", generating sites on demand with almost zero effort. DODAS provides ready-to-use solutions to implement a "Batch System as a Service" as well as a BigData platform for a "Machine Learning as a Service", offering a high level of customization to integrate specific scenarios. A description of the DODAS architecture will be given, including the CMS integration strategy adopted to connect it with the experiment's HTCondor Global Pool. Performance and scalability results of DODAS-generated tiers processing real CMS analysis jobs will be presented. The Instituto de Física de Cantabria and Imperial College London use cases will be sketched. Finally a high level strategy overview for optimizing data ingestion in DODAS will be described.

[1] Corresponding author: spiga@infn.it

1 Introduction

The Dynamic On Demand Analysis Services (DODAS) [1] is an open-source Platform-as-a-Service tool, developed and maintained by INFN, which allows to deploy software applications over heterogeneous and hybrid clouds. DODAS is one of the so-called Thematic Services of the EOSC-hub project [2] and it instantiates on-demand container-based clusters through Apache Mesos [3]. It offers a high level of abstraction to users, allowing to exploit any cloud infrastructure with almost zero effort since it requires a very limited knowledge of the underlying technologies.
DODAS completely automates the process of provisioning by creating, managing, and accessing a pool of heterogeneous computing and storage resources. As a consequence, it drastically reduces the learning curve as well as the operational cost of managing community-specific services running on distributed clouds.
Currently DODAS provides support to deploy:

- HTCondor-based batch system as a Service;
- Big Data platform, storage and distributed processing frameworks, for ML as a Service.

From the user perspective, this is a simple solution for the creation of a complex set-up on any cloud based environment. Users should experience the process of generating complex setups as easily as creating a virtual machine on any IaaS: a simple one-click solution. The DODAS added value can be summarized as follows:

- provides a simple but complete abstraction of hybrid cloud infrastructures;
- automates both virtual hardware provisioning and its configuration;
- provides a cluster platform with a high level of self-healing and scalability;
- guarantees both set-up and service customization to cope with specific scientific requirements.

DODAS is currently adopted as a solution for several use cases:

- exploitation of opportunistic computing, intended as resources not necessarily or permanently dedicated to a specific experiment and/or activity;
- elastic extension of existing facilities, to absorb peaks of resource usage;
- generation of on-demand batch systems for data processing.

DODAS has already been integrated into the submission infrastructure of the Compact Muon Solenoid (CMS) [4], one of the two general purpose experiments at the CERN Large Hadron Collider (LHC) [5], and into the data analysis workflow of the Alpha Magnetic Spectrometer (AMS-02) [6], an experiment hosted on the International Space Station. Other communities are also actively investigating DODAS features for a possible integration into their own workflows.
Infrastructure-wise, DODAS has been successfully adopted to use and connect to resources provided by several Cloud providers, both private (based on Openstack or OpenNebula) and public (such as Microsoft Azure, Amazon Web Services, and T-Systems' Open Telekom Cloud - OTC).
In Sec. 2 we focus on the DODAS architectural pillars. Sec. 3 addresses its integration in the CMS computing infrastructure, namely HTCondor Global Pool [7,8]. After reporting on the results obtained exploiting OTC cloud infrastructure, two additional use cases are described: one from Instituto de Física de Cantabria (IFCA) and one from Imperial College London (ICL). A brief architectural description on a data caching mechanism as a solution

for optimizing data ingestion in DODAS is given in Sec. 4. Sec. 5 provides summary and conclusions.

2 DODAS architectural pillars

DODAS has a highly modular architecture and its workflows are highly customisable. For this reason, it is very extensible, spanning from software dependencies up to the integration of external services, including also user-tailored code management. There are four main pillars in the DODAS architecture which can be summarized as:

- **abstraction**: both in terms of software application and dependency description, and of underlying IaaS level cloud infrastructures;
- **automation**: it refers to software and application setup in order to manage resources and orchestrate software applications;
- **multi-cloud support**: to deal with multiple heterogeneous Cloud infrastructures.;
- **flexible AAI**: authentication, authorization, delegation, and credential translation primitives providing a secure composition of the various services participating in the DODAS workflow.

Regarding the technical implementation Fig.1, these pillars have been realised mostly using building blocks originally developed by the INDIGO-DataCloud [9] project and currently part of the EOSC-hub's service portfolio. These are namely: Identity and Access Management (IAM) and Token Translation Service (TTS), PaaS Orchestrator and Infrastructure Manager (IM). The composition of these services represents the so-called PaaS Core Service of DODAS.

IAM is the OpenID Connect (OIDC) Authorization Server, supporting the eduGAIN [10] federation, which has the crucial role to authenticate users. It also provides them with an access token which is then used to authorize services to act on the users' behalf. The PaaS

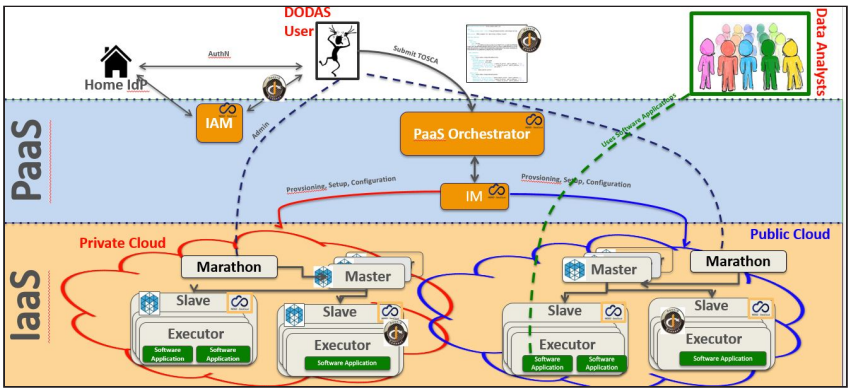

Fig. 1. A high level schema of the DODAS architecture. Three different colors identify User Domain (white), DODAS PaaS core service layer (light blue), and resource layer (dark yellow).

Orchestrator and IM [11] are responsible to take care of the users' requests (in form of a TOSCA [12] template) and to prepare a cluster for containers orchestration over the IaaS. DODAS relies on Mesos for resource management and Marathon [13] for the container orchestration. The container orchestrator is the layer responsible for the execution of end-users' services.

In principle, any framework and/or software application can run on such cluster. However,

DODAS provides two principal baseline recipes ready to be used and which can be easily extended: a HTCondor batch system and a Spark [14] cluster.

3 DODAS integration into the CMS computing infrastructure

The integration of DODAS into the CMS Submission Infrastructure has been designed to allow the generation of an ephemeral site (a Tier) fully compliant with the WLCG [15] requirements. As a matter of fact, DODAS has the objective of generating a lightweight Tier-N on demand.

From the resource provisioning perspective, DODAS implements the vacuum model,which means Worker Nodes (WN) are produced spontaneously instead of having to be requested from third party, with the consequence of any Computing Element exposed.WNs, namely HTCondor startd processes, start up as a Docker [16] container whose lifecycle is managed by the Marathon orchestrator. WN containers are dynamically configured as Marathon applications and this allows to dynamically setup HTCondor startd process in order to auto-join the CMS HTCondor Global Pool. The Token Translation Service (TTS) integration is necessary to enable the auto-join mechanism. TTS has been integrated through ProxyCache microservice managed by Marathon. The ProxyCache is responsible for interacting with TTS to periodically obtain an X.509 proxy certificate. This is derived from the original user access token and it is then cached and passed to all the WNs running on the cluster whenever needed. The X.509 proxy certificate is the credential used by every startd HTCondor process for the authentication with HTCondor Global Pool.

The global pool implements then a mapping system based on two parameters: the certificate Distinguished Name (DN) and the site name. This is a further authorisation layer, managed at the level of HTCondor central manager, which guarantees a given DN to be allowed to run the authorised workflows only.

Squids proxies [17] also run as containers and are managed by Marathon, while CVMFS [18] client is mounted on the host. All these services are automatically installed and configured without any extra requirements for the end-user who has only to configure a TOSCA template. This is what allows to drastically reduce the DODAS learning curve.

Data ingestion is based on the CMS XRootD [19] federation, with an additional layer recently included in the DODAS service: the dynamic Xcache. This is meant to be a storage layer for input data caching between CPUs and remote data storage endpoints. The output data are expected to be written on a remote storage system as defined by the users.

3.1 Ephemeral Tier3 on Open Telekom Cloud

DODAS has been recently deployed as Tier3 on T-System Open Telekom Cloud (OTC)resources. Beside the other cloud provider exploitations done through DODAS, OTC has been so far the most important in terms of scaling. As shown in Fig. 2, a DODAS ephemeral T3 (T3_IT_Opportunistic_hnsci) was one of the top 6 Tier sites running CMS jobs for 10 days. Job success rate and CPU efficiency were analyzed to ensure DODAS was not introducing systematic inefficiency. Measured CPU efficiency results above 85%, which is comparable with the CMS provided baseline for the same type of workflow.

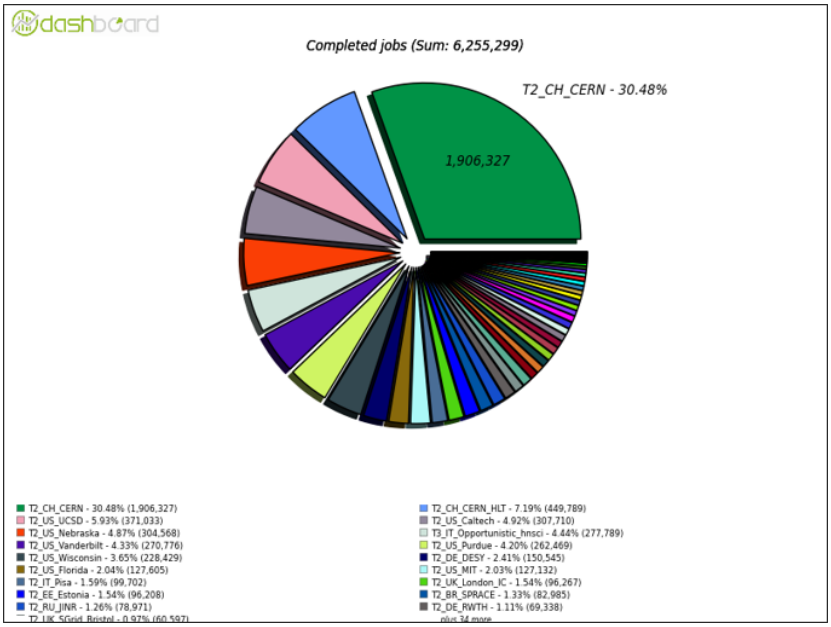

Fig. 2. The pie chart shows the CMS sites running CMS jobs over 10 days, from July 26th to August 5th. Each color represents a distinct site.

Fig.3 shows the very same information from the resource monitoring perspective. There were up to 2k concurrent worker nodes orchestrated by Marathon. The picture allows to appreciate two major added values of a DODAS-based system, namely elasticity and self-healing.

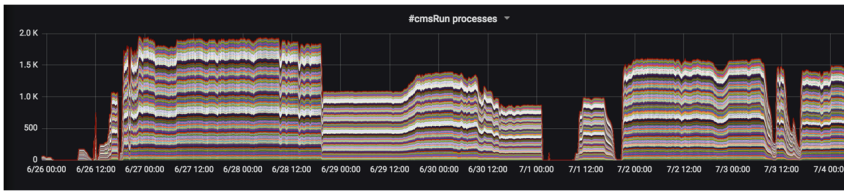

Fig. 3. Number of CMSSW processes running on the DODAS cluster. Each color is a distinct process. The number of WNs are adjusted depending on the memory consumption as required by jobs.

3.2 CMS tier3 on private cloud

In this use case, DODAS has been exploited to create a CMS Grid Tier-3 site using resources hosted at Imperial College London (ICL), UK.

The primary objective of this activity has been to perform a functional test of DODAS to run requirement-specific workflows. For the functional test, a small amount of quota has been reserved on ICL public OpenStack [20]. The new Tier3 site has been called T3_UK_Opportunistic_dodas, and it relies on the Tier2 running at ILC (T2_UK_London_IC) as target storage to copy the produced output data.

The test has been carried out using two different configurations of the Mesos Cluster. The first configuration (C1) had Slave Servers with 8 CPUs and 16 GB RAM. C1 was used to create WNs with 1 CPU and 2 GB RAM, to run more than 1000 jobs with the goal of producing CMS data in NanoAOD format for ICL standard model physics analyses from MiniAOD [21]. This use case included a CMS tree slimming stage and the production of flat trees from EDM trees. The second configuration (C2) had Slave Servers with 16 CPUs and 32 GB RAM. C2 was used to create WNs with 8 CPU and 16 GB RAM, to run requirement-specific jobs of high energy physics interest. A first use case aimed to perform the AOD [21] step of the offline reconstruction sequence for gluon-fusion (cp-mixed) and ggH+2j events generated using Madgraph5 aMC@NLO [22,23] (more than 3k jobs). A second use case produced Gen-Sim (Generation and Simulation workflow) from LHE files [24], performing Pythia [25] parton shower, hadronisation and particle decays, and Geant4 [26] detector simulation (roughly 150 jobs). A third use case is being proven using 7 WNs with 15 CPU and 32 GB RAM, creating LHE files for Madgraph5 aMC@NLO V+j(j) processes generated from the Matrix Element only.
All the use cases were successfully run at the T3, completing 100% of the jobs. Even though the statistic of the test is not sufficient to perform an accurate and exhaustive study for the general performance of the new infrastructure, the results show a total average above 90% in CPU efficiency, demonstrating the validity of the DODAS-based solution.

3.3 IFCA experience

Nowadays, the Instituto de Física de Cantabria (IFCA) participates actively in the CMS experiment sharing computing resources both within Grid computing infrastructure and with local resources, like HPC or local clusters, where national or international researchers, as well as CMS users, run their analysis.
In recent years, the IFCA CMS users used to run their jobs using other ways besides the local infrastructure.
For this reason, DODAS has been considered as a service where the IFCA-CMS analyses could be run on. This service has been installed on an OpenStack based cloud infrastructure, implemented by the IFCA Advanced Computing and e-Science research group. To integrate DODAS at the IFCA cloud infrastructure, the Indigo PaaS orchestrator was replaced by the Openstack orchestration service named Heat which main functions are to manage the infrastructure lifecycle: from the initial Mesos cluster deployment to node escalation and configuration of the environment.
Even though it is considered as a proof of concept, the service could enter the production stage as next step and test site may be turned into a local site in the HTCondor Global Pool.

4 Data ingestion strategy

In the DODAS architecture, a possible source of inefficiency might come from the latency during read operations of data hosted outside the cloud provider. To address this issue, a possible solution is to bring up a set of services acting as a proxy between the computing resources and the remote storage, and in addition to cache the served data. The technology used for this solution is called XCache and is based on XRootD, a software largely adopted in the CMS computing model under the “Anydata, Anytime, Anywhere” (AAA) project. A deployment recipe for an XCache cluster federated under a common namespace (Fig. 4) has been integrated with the standard DODAS recipe and it is available within the DODAS deployment procedure. In this design, the federator server (called “redirector”) is

responsible for redirecting request to the correct proxy server avoiding file duplication across the federated servers. The logic is the following: if the file requested by the client is already stored in one of the cache servers, the client is then put directly in contact with the corresponding server; otherwise an available cache server is chosen according to a round-robin mechanism. The file is then provided to the client using the "proxying while caching" function. A clear advantage of such a solution is of course to mitigate network latencies in remote data access as much as possible.

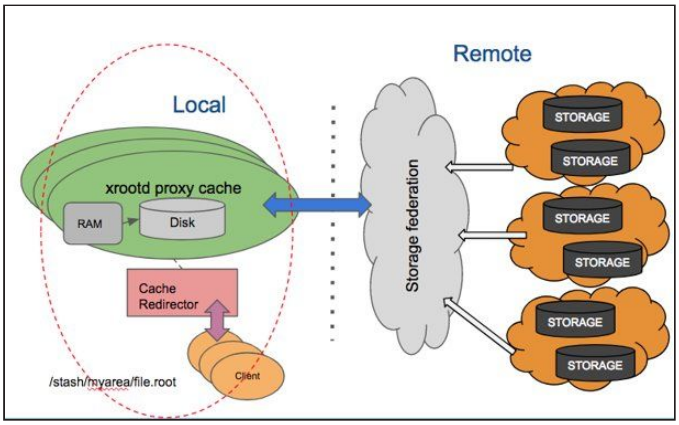

Fig. 4. Data Cache architectural schema. The AAA federation of CMS (right) and the dynamic cache system (left) are shown.

5 Conclusions

DODAS has been developed under the EC-funded INDIGO-DataCloud project and it is currently part of the EC-funded EOSC-hub project service catalogue. It has demonstrated to be very promising, being successfully integrated in the CMS submission infrastructure, the HTCondor Global Pool, and currently representing a working solution for this experiment to exploit opportunistic resources and to elastically extend pledged sites. Moreover, several communities are evaluating the DODAS adoption in the context of CMS, such as IFCA and ICL. Besides CMS, the AMS-02 experiment is integrating DODAS into their data analysis workflow.

Regarding the ICL case, a possible usage of DODAS for the exploitation of AWS resources is under evaluation.

In terms of future developments, DODAS is actively evolving the data ingestion mechanism towards a smart use of a data caching mechanism and is extending the BigData platforms support to enable machine learning as a service.

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

References

1. DODAS Project: https://dodas-ts.github.io/dodas-doc/
2. EOSC-hub project: https://eosc-hub.eu/

3. Apache Software Foundation: Apache Mesos. http://mesos.apache.org/ (2018)
4. S. Chatrchyan et al. CMS Collaboration 2008 The CMS experiment at the CERN LHC J. Inst. 3 S08004
5. L. Evans and P. Bryant (editors), "LHC Machine", JINST 3 (2008) S08001, doi:10.1088/1748-0221/3/08/S08001
6. The Alpha Magnetic Spectrometer (AMS) on the International Space Station: Part I – results from the test flight on the space shuttle.Physics Reports, 366(6):331 – 405,2002
7. 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
8. Thain D, Tannenbaum T and Livny M 2004 Distributed computing in practice: the condor experience Concurrency: Pract. Exper. 17 323-356
9. Salomoni, D., Campos, I., Gaido, L. et al. J Grid Computing (2018) 16: 381. https://doi.org/10.1007/s10723-018-9453-3
10. eduGAIN interfederation http://www.geant.org/Services/Trust_identity_and_security/eduGAIN
11. Caballer, M., Zala, S., García, Á. et al. J Grid Computing (2018) 16: 3. https://doi.org/10.1007/s10723-017-9418-y
12. Palma, D., Rutkowski, M., Spatzier T.: TOSCA Simple Profile in YAML Version 1.1. Tech. rep., OASIS Standard. http://docs.oasis-open.org/tosca/TOSCA-Simple-Profile-YAML/v1.1/TOSCA-Simple-Profile-YAML-v1.1.html (2016)
13. Marathon. https://mesosphere.github.io/marathon/ (2018)
14. Matei Zaharia, Reynold S. Xin, et al. 2016. Apache Spark: a unified engine for big data processing. 59, 11 (October 2016), 56-65. DOI: https://doi.org/10.1145/2934664
15. WLCG - Worldwide LHC Computing Grid, http://wlcg.web.cern.ch.
16. Dirk Merkel. Docker: lightweight linux containers for consistent development and deployment. Linux Journal, 2014(239):2, 2014
17. Dave Dykstra and Lee Lueking 2010 Greatly improved cache update times for conditions data with Frontier/Squid *J. Phys.: Conf. Ser.* **219** 072034
18. P Buncic et al 2010 J. Phys.: Conf. Ser. 219 042003
19. 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
20. OpenStack Foundation: OpenStack. https://www.openstack.org (2018)
21. CMS Collaboration (Petrucciani G., Rizzi, A., and Vuosalo C.), Mini-AOD: A New Analysis Data Format for CMS, Journal of Physics: Conference Series 664 (2015) 072052, DOI: 10.1088/1742-6596/664/7/072052
22. Alwall J. et al. MadGraph 5 : Going Beyond, JHEP 1106 (2011) 128, DOI: 10.1007/JHEP06(2011)128
23. Alwall J. et al., The automated computation of tree-level and next-to-leading order differential cross sections, and their matching to parton shower simulations, JHEP 1407 (2014) 079, DOI: 10.1007/JHEP07(2014)079
24. Alwall J. et al. A Standard format for Les Houches event files, Comput.Phys.Commun. 176 (2007) 300-304, DOI: 10.1016/j.cpc.2006.11.010
25. Torbjörn Sjöstrand et al. An Introduction to PYTHIA 8.2, Comput.Phys.Commun. 191 (2015) 159-177, DOI: 10.1016/j.cpc.2015.01.024
26. GEANT4 Collaboration (S. Agostinelli et al.), GEANT4: A Simulation toolkit, Nucl.Instrum.Meth. A506 (2003) 250-303, DOI: 10.1016/S0168-9002(03)01368-8