Data Science as a Service

Prototyping for an Enterprise Self-Service Platform for Reproducible Research

Steve Guhr
NetApp Deutschland GmbH
Berlin, Germany
e-mail: steve.guhr@netapp.com

Detlef Amendt
Center for Information Management
Hannover Medical School
Hannover, Germany
e-mail: amendt.detlef@mh-hannover.de

Jan-Hendrik Martenson
NetApp Deutschland GmbH
Hamburg, Germany
e-mail: jan-hendrik.martenson@netapp.com

Benjamin Schantze
NetApp Deutschland GmbH
Hamburg, Germany
e-mail: benjamin.schantze@netapp.com

Hans Laser
Center for Information Management
Hannover Medical School
Hannover, Germany
e-mail: laser.hans@mh-hannover.de

Svetlana Gerbel
Center for Information Management
Hannover Medical School
Hannover, Germany
e-mail: gerbel.svetlana@mh-hannover.de

Jannes Gless
Center for Information Management
Hannover Medical School
Hannover, Germany
e-mail: gless.jannes@mh-hannover.de

***Abstract*—A data scientific process (e.g., Obtain, Scrub, Explore, Model, and iNterpret (OSEMN)) usually consists of different steps and can be understood as an umbrella for the combination of different most modern techniques and tools for the extraction of information and knowledge. In this paper, we show a prototypical implementation for the efficient use of available compute center resources as a self-service platform on enterprise technology to support data-driven research. Scientific requirements for reproducibility and comprehensibility are to be taken into account.**

Keywords*—*Data Science as a Service; reproducible research; enterprise information technology; self-services; cloud infrastructure; data science platform.

I. INTRODUCTION

Technology should be available to everyone. That is one of the reasons why companies build services. One of the key aspects to consider when building service portfolios should be to make it as easy and consumable for the end-user as possible. The main challenge is to "carry" those known tools and solutions into a scalable and powerful platform that can be provided with enterprise technology to warrant Service-Level-Agreements (SLA) from a central enterprise Information Technology (IT).

A. The Data Science Process

To obtain information (e.g., based on patterns) for relevant business decisions from data of heterogeneous data sources, a classical multi-stage process for data preparation and analysis is used, the so-called data mining process [1]. Data science, on the other hand, can be understood as an umbrella for the combination of various state-of-the-art techniques for the extraction of information and knowledge (so-called insights) to develop data-based applications and, thus, automating processes. One approach to describe the individual steps for the data science process is **O**btain, **S**crub, **E**xplore, **M**odel and I**N**terprenting (OSEMN) [2]. In the Obtain step, for example, query languages are required for databases that can be extracted in various formats. Python [3] and R [4] encapsulate the otherwise heterogeneous data query tools (e.g. Structured Query Language (SQL), eXtensible Markup Language (XML), Application Programming Languages (API), Comma Separated Value (CSV) and Hybrid File System (HFS)). Classic database techniques such as Extract Transform Load (ETL) process can be used in the cleanup step (Scrub). Computer languages like Python and R or application suits like SAS Enterprise Miner [5] or OpenRefine [6] can also be used to transform data. To examine the data (Explore) languages like Python or R specialize in particular appropriate libraries (e.g. Pandas [7] or Scipy [8]). In this step, however, familiar players from the business

 ISBN: 978-1-61208-750-4

intelligence world (e.g. Rapid Miner [9] or KNIME [10]) can also be found for data-wrangling. To build a model, there are again specialized Python libraries like "Sci-kit learn" [11] or CARET [12] for R. Other tools like KNIME or Rapid Miner find reuse in this step as well. Finally, for interpreting the model and the data, as well as evaluating the generalization of the algorithm, tools for data visualization are reused (e.g. matplotlib [13], Tableau [14] or MS Power BI [15]). In summary, it means that, for the many single steps in OSEMN, many different tools can be necessary.

B. Reproducible Research

In the domain of computer-aided data-driven science, researchers today use common libraries and tools. Researchers often choose free and open source tools [16].

The reproducibility and repeatability of the research results and the description of the specific runtime environment in which the results were created are described in the respective publications either not at all or only textually [16][17]. An important factor in the publication of the scientific work is the reproducibility of the research results [17][18].

The data principles published in 2016 define fundamentals that research data and research data infrastructures should meet in order to ensure sustainability and reusability [19]. The scientific data should therefore be **f**indable, **a**ccessible, **i**nteroperable and **r**eusable (FAIR data principles). In a highly simplified way, data and services should be stored in central data repositories using suitable metadata (F), taking into account aspects of long-term archiving (A), and should be able to be exchanged and interpreted (semi-)automatically (I) and thus be comparable and reusable (R).

Results of systems research show that open source tools in particular are suitable for reproducibility requirements. Although Docker was introduced primarily for enterprise needs and Web application delivery, it provides solutions for virtualization, platform portability, reuse, sharing, archiving, and versioning [17] for the scientific community.

The use of tools such as Jupyter Notebooks (jupyter.org) enables semantically interoperable publication of program code, including through the use of the IPYNB format [16]. Jupyter Notebooks supports workflows in the fields of scientific computing, astrophysics, geology, genetics and computer science [16]. Various applications and programming languages (e.g. Python, R) offer interfaces to Jupyter Notebooks [17][20]. Jupyter gathers many valuable tools that are needed in the steps of the OSEMN process model.

C. Aims of the project

NetApp is one of the leading independent data management providers [21] and has been helping organizations store, manage, secure, and leverage their most mission-critical data assets for more than 25 years.

The Center for Information Management (ZIMt) of the Hannover Medical School (MHH) centralizes operative systems and is a service provider especially for the areas of research and teaching, clinic and administration. The ZIMt operates a class tier 3 [22] computing center at the MHH and is ISO 9001:2015 certified. The centralization of applications to support the scientific field is a strategic goal of the MHH.

The goal of this project is to establish an easy to maintain and cost-efficient infrastructure to support the data-driven research and the implementation in the existing data center infrastructure. Thereby, the requirements according to FAIR and the operation of enterprise-level applications will be considered.

The rest of the paper is structured as follows. In Section II, we describe the methods used to address the above mentioned challenges. In Section III, we describe the results achieved referring to the main issues. Section IV concludes this work addressing open issues and next steps.

II. METHODS / APPROACH

For the operation of Jupyter Notebooks in the data center, the open-source environment JupyterHub Notebook Server is used. It enables users to access computing environments and resources without bothering users with installation and maintenance tasks.

Docker is a technology that abstracts applications from the underlying operating system to gain portability for software solutions wrapped in so called "Containers". With a standardized environment (the Docker software/binaries running on many different operating systems today) built for containers one can use individual application manifests on different sites (e.g. on-premises as well as in a public cloud infrastructure provided by Google, Amazon Web Services (AWS) or Microsoft Azure) without the need to change any code.

JupyterHub uses Docker as the foundation for deploying Jupyter Notebooks. The Jupyter Docker Stacks project [23] will provide standardized Jupyter Notebooks environments for various applications using Docker images, including pre-configured environments for data science use. For special requirements of the development environment, it is possible to offer further images that can be individually modified to the needs of the user.

Docker containers are more convenient to deploy, easier to manage, minimize overhead in resource usage, and are therefore more efficient than traditional virtual infrastructures [24]. The provisioning of environments (e.g., through containers) in the academic field can be very extensive, thus increasing the burden on the operators and maintainers of the environment. Automating the provisioning and orchestration of the environment is highly recommended.

Running applications in containers also does not automatically solve the challenge of protecting those applications against failures (e.g. hardware outages or resource bottlenecks on this one server we are working on). Even though containers are encapsulated on top of an operating system, there might be issues with the underlying host running the container - therefore, an additional software layer taking care of resource scheduling and availability of our Jupyter Notebooks is needed.

 ISBN: 978-1-61208-750-4

Kubernetes has been used as an open-source solution to orchestrate, automate and meet those high availability requirements for container-based infrastructure [25]-[27]. It is a widely used and proven technology for delivering services like Jupyter. Because Kubernetes is designed to host an enormous number of applications with minimal overhead, it is perfect for many Jupyter Notebooks and other potential applications within the science ecosystem [24][28].

For more flexibility in design, a hypervisor (VMware) was used to deploy the hosts for container orchestration based on Docker and Kubernetes. Terraform and Ansible are fully automated - Terraform creates the virtual machines within the hypervisor, Ansible handles the installation of the packages and configuration of the hosts (configuration management). To avoid inconsistencies in the configuration, the DevOps (software Development and information technology Operations) paradigm applies to be an "immutable" infrastructure, where every change in the ecosystem leads to a completely new deployment of the entire stack [29].

Docker containers are ephemeral, which means that they do not persist data after termination of their life cycle. Data storage must be ensured and is therefore required to secure the data beyond the life cycle of the container.

In terms of the reproducibility of the results, the collected data becomes the most important asset in the process chain - since a robust and highly available architecture is to serve as the basis, a NetApp storage system is used to store the data. The central storage system consolidates data in the data center. This prevents, on the one hand, the storage of data on terminals and on the other hand, enables a more effective backup.

The Network File System (NFS) was chosen as a protocol for the connection to the storage. This decision is based on being able to make the stored data available outside of the data science platform. NFS can be used by several clients at the same time with write access. This integration into conventional infrastructures simultaneously allows the usual access via Explorer or Shell.

Within Kubernetes, so-called storage classes can be used to provide storage from an external source - in this case, a container gets persistent storage space to store data beyond its own life cycle, thereby making the data reusable. Trident [30] is an open-source storage orchestrator for Kubernetes provided by NetApp and can be used to add persistence to the Jupyter Notebooks. Since each notebook is to be provisioned individually for a particular user, each user also receives an exclusive area for data storage. This is made possible by the use of authentication within the ecosystem, so each user must provide credentials to be accepted.

Even though we are in a “proof of concept” phase of the project, we wanted to integrate authentication methods from the very beginning to control access to the platform while obtaining compliance in terms of security. At the MHH, a local security area for managing objects (e.g. usernames, computers, printers, etc.) is implemented as a domain via Microsoft Windows Active Directory. For the centralization of user IDs administration using Role-Based Access Control (RBAC), the authentication to the Active Directory using Lightweight Directory Access Protocol (LDAP) implementation of JupyterHub was accessed.

III. RESULTS

This prototypical implemented infrastructure allows the end-user (students, scientists) to easily use Jupyter Notebooks. Using the Docker-based approach, the description of the runtime environment required for the research approach can be fixed using the Docker-specific tagging option and stored in a manifest in a comprehensible and interoperable manner for publication [17].

If the researcher chooses a work environment based on Jupyter Notebooks, necessary work steps and results can be saved together with the notebook [16][20]. The basic condition to support requirements as described in FAIR could be taken care of.

By default, configurations of Jupyter Notebooks are offered and further evaluated via the JupyterHub Spawner (Basic and Data Science). Additional libraries or tools, which may not be included in the standardized environment, can be installed flexibly to the current runtime environment in their own separate area. Jupyter Notebooks, therefore, offer the possibility to use multiple tools without having to change the environment. Requirements for various tools, such as those required in processes such as OSEMN, can be supported by Jupyter Notebooks.

The isolation of the specific workspace of a researcher can be solved by using Docker. Regardless, central compute resources can be shared and used efficiently across multiple environments.

The operator of the infrastructure (ZIMt) achieves a work facilitation through the selected reference architecture (see Figure 1) by automating the provision of resources for the users (researchers). Because JupyterHub is deployed through Docker, the branding requirements for maintaining the Corporate Identity can be easily met. By using the existing Active Directory, user access can be controlled centrally. Authentication via LDAP simplifies logging on to the system, since no separate access data needs to be maintained.

By provisioning the required storage area at runtime, resources can be provided centrally and efficiently. The persistence of research data outside the Docker container runtime on the existing enterprise storage system could be efficiently solved by using Trident (see Figure 1).

Each time a user logs on to the home page, the system checks to see if the user already has a Jupyter notebook created in the past. In this case, he or she will be redirected to this existing environment. Otherwise, a new notebook is created (in the form of a new container) and new storage space is provided (because that user did not exist before).

 ISBN: 978-1-61208-750-4

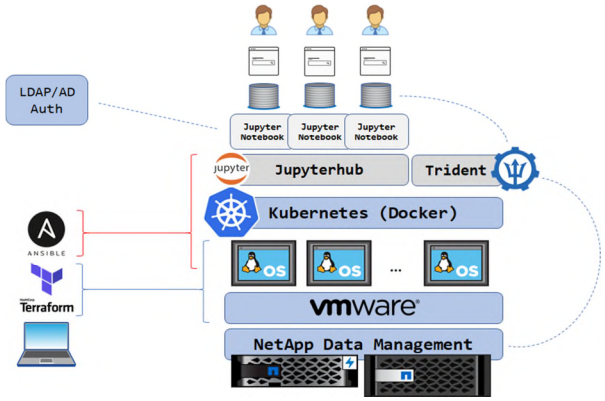

Figure 1. Prototyped Architecture to provision Jupyter Notebooks on Enterprise Technology

IV. CONCLUSION

In this paper, we showed a prototypical implementation for the efficient use of available compute center resources as a self-service platform on enterprise technology to support data-driven research.

The use of the predefined configurations of the Jupyter Notebooks is initially limited by the images. It will show with prolonged use of the service if the provision of additional images would be useful.

For the prototype implementation of this infrastructure, one Kubernetes master with two Kubernetes workers was deployed. For productive operation, at least two Kubernetes masters should be used to meet the requirements for failure safety. In the event of a disaster recovery scenario and the loss of the complete Kubernetes cluster, the storage volumes that are deployed through Trident must be manually reconnected. An automatism for restore procedures would still have to be created. In an emergency, the administrator can migrate the contents of the corresponding volume through NFS.

Existing and established methodologies like OAuth [31] or OpenID [32] would have required additional components to be available. However, those security concepts will be considered in later phases of the project after the initial thesis is validated (if this software stack is suitable for the use case at all).

If the JupyterHub internal database is lost, the link to the pod and, thus, to the individual runtime environment is lost and needs to be restored. To use Jupyter notebooks on existing HPC infrastructures, the use of the Jupyter Enterprise Gateway [33] needs to be evaluated.

As already mentioned before, this project implemented only a proof of concept to demonstrate the feasibility of IT operations by combining common data science tools with enterprise architecture. Beyond Jupyter Notebooks related to machine learning, tools such as Airflow [34] or Pachyderm [35] (as a platform solution) could be used for pipelining and automation in the next development stages. These tools could support process models such as OSEMN, as well as aspects of reproducibility and re-usability.

REFERENCES

[1] C. Li, “Preprocessing Methods and Pipelines of Data Mining: An Overview”, CoRR, 2019, arXiv:1906.08510

[2] H. Mason and C. Wiggins, “A Taxonomy of Data Science”, dataists.com, 2010, http://www.dataists.com/2010/09/a-taxonomy-of-data-science/, last accessed 2019/07/22

[3] Python Programming Language, 2019, https://www.python.org, last accessed 2019/10/23

[4] The R Project for Statistical Computing, https://www.r-project.org, last accessed 2019/10/23

[5] SAS® Enterprise Miner™, 2019, https://www.sas.com/en_us/software/enterprise-miner.html, last accessed 2019/10/23

[6] Open Refine, http://openrefine.org/, last accessed 2019/10/23

[7] Pandas Python Data Analysis Library, https://pandas.pydata.org/, last accessed 2019/10/23

[8] Scipy, 2019, https://www.scipy.org/, last accessed 2019/10/23

[9] Rapid Miner, 2019, https://rapidminer.com/, last accessed 2019/10/23

[10] KNIME End to End Data Science, 2019, https://www.knime.com/, last accessed 2019/10/23

[11] scikit-learn: machine learning in Python, https://scikit-learn.org/stable/, last accessed 2019/10/23

[12] A Short Introduction to the caret Package, https://cran.r-project.org/web/packages/caret/vignettes/caret.html, last accessed 2019/10/23

[13] matplotlib, 2019, https://matplotlib.org/, last accessed 2019/10/23

[14] Tableau, 2019, https://www.tableau.com, last accessed 2019/10/23

[15] Microsoft Power BI, 2019, https://powerbi.microsoft.com, last accessed 2019/10/23

[16] T. Kluyver et al., Jupyter Development Team, “Jupyter Notebooks – a publishing format for reproducible computational workflows”, IOS Press. pp. 87-90, 2016, DOI: 10.3233/978-1-61499-649-1-87

[17] C. Boettiger, “An introduction to Docker for reproducible research, with examples from the R environment”, ACM SIGOPS Oper. Syst. Rev.. 49. 10.1145/2723872.2723882. https://arxiv.org/pdf/1410.0846.pdf

[18] Nature Editors 2012. “Must try harder”. Nature. 483,7391 Mar. 2012, 509–509.

[19] M. D. Wilkinson et al., “The FAIR Guiding Principles for scientific data management and stewardship”, Scientific Data 3, doi : 10.1038/sdata.2016.18, 2016.

[20] E. Cirillo, “TranSMART data exploration and analysis using Python Client and Jupyter Notebook”, 2018, http://blog.thehyve.nl/blog/transmart-data-exploration-and-analysis-using-python-client-and-jupyter-notebook,last accessed 2019/07/22

[21] K. Kerr, “Gartner Named NetApp a Leader in Magic Quadrant for 2019 Primary Storage”, 2019, https://blog.netapp.com/netapp-gartner-magic-quadrant-2019-primary-storage/, last accessed 2019/10/23

[22] OVH SAS. 2018 “Understanding Tier 3 and Tier 4”. https://www.ovh.com/world/dedicated-servers/understanding-t3-t4.xml, last accessed 2018/09/15.

[23] Jupyter Docker Stacks, 2018, https://jupyter-docker-stacks.readthedocs.io/en/latest/, last accessed 2019/10/23

[24] Q. Zhang, L. Liu, C. Pu, Q. Dou, L. Wu and W. Zhou, “A comparative study of Containers and Virtual Machines in Big Data environment”, arXiv:1807.01842v1

 ISBN: 978-1-61208-750-4

[25] L. Hecht, “What the data says about Kubernetes deployment patterns”, 2018, https://thenewstack.io/data-says-kubernetes-deployment-patterns/, last accessed 2019/07/22

[26] Project Jupyter Contributors, “Zero to JupyterHub with Kubernetes”, 2019, https://zero-to-jupyterhub.readthedocs.io/en/latest/, last accessed 2019/07/22

[27] S. Conway, “Survey shows Kubernetes leading as orchestration Platform”, 2018, https://www.cncf.io/blog/2017/06/28/survey-shows-kubernetes-leading-orchestration-platform/, last accessed 2019/07/22

[28] S. Talari, “Why Kubernetes is a great choice for Data Scientists”, https://towardsdatascience.com/why-kubernetes-is-a-great-choice-for-data-scientists-e130603b9b2d, last accessed 2019/07/22

[29] P. Debois and J. Humble, “The DevOps Handbook: how to create World-Class agility, Reliability, and Security in Technology Organizations”, IT Revolution Press, 2016, ISBN: 978-1942788003

[30] NetApp Trident, 2019, https://netapp-trident.readthedocs.io/en/latest/introduction.html, last accessed 2019/10/23

[31] OAuth community site, https://oauth.net/, last accessed 2019/10/23

[32] OpenID Foundation, 2019, https://openid.net, last accessed 2019/10/23

[33] Project Jupyter Team, “Jupyter Enterprise Gateway”, 2016, https://jupyter-enterprise-gateway.readthedocs.io/en/latest/, last accessed 2019/07/22

[34] Apache Airflow Documentation, https://airflow.apache.org/, last accessed 2019/10/23

[35] Pachyderm - Reproducible Data Science that Scales!, 2019, https://www.pachyderm.io/, last accessed 2019/10/23

 ISBN: 978-1-61208-750-4