EPiC Series in Computing

Volume 105, 2025, Pages 126–135

Proceedings of EUNIS 2024 annual congress in Athens

An Update on the Münster University Cloud - technical architecture and user adoption

Raimund Vogl[1] and Markus Blank-Burian[1]

Center for Information Technology (CIT), University of Münster, Germany
{rvogl,blankburian}@uni-muenster.de

Abstract

Initially starting with a now widely adopted but in scope quite narrowly focused sync & share cloud storage service (named sciebo.nrw) in 2015, Münster University expanded the scope of on-premises higher education and research cloud services to Infrastructure as a Service (IaaS) in 2016 with the plan for an open-source based platform for research data storage and processing using state-of-the-art cloud technology. After forming a multi-university consortium, developing a project plan, acquiring funding from the North Rhine-Westphalia (NRW) state ministry of science (MKW) and conducting the procurement, the Uni Cloud took off in 2019 and has since gained a very wide adoption amongst university researchers at the University of Münster, with 5.5 Petabyte of user data stored. It is now a core element of the e-Science services of the university and the platform of choice for large scale research undertakings (especially in collaborative research centers) and innovative IT services (like Gitlab, GitOps, UniGPT, JupyterHub, etc...).

1 Introduction

The implementation of cloud technology in the context of on-premises self operated (multi-university) services started for Münster University in 2014 with the sync & share NRW project, which became well known and widely adopted (with currently 230.000 users) as sciebo.nrw (the science box) [8]. From an IT architecture point of view, sciebo.nrw was offering cloud services (using the owncloud open-source solution), but relying on traditional infrastructure technologies usually used in the HPC context (especially a parallel file system). After the sciebo project, we started investigating into the applicability of cloud native technologies (like Ceph storage, OpenStack and Kubernetes) and soon realized that this was the perfect match for the rising demands for research data management.

With research data management becoming a focus of attention in the digital transformation activities at universities and research institutions, discussions in NRW higher education IT circles started around 2016 on what approaches to take for an infrastructure that was able to cope with the foreseeable deluge of data and the cost of storage hardware connected with this, the long retention periods required, and the manifest problem of storage hardware obsolescence and the connected disruptive effects of migration requirements.

Amongst the NRW universities, two main consortia formed with different approaches to the problem. One consortium focused on procuring storage platforms (integrated NAS and object storage components) with elaborate requirements on interface standards to prevent vendor lock-in.

The other consortium (lead by Münster University) adopted an approach via an Open Source based on-premises Infrastructure as a Service (IaaS) cloud with the following rationale:

- Low cost: Cloud hyper-scalers provide their high-quality tools in open-source and thus eliminate the cost of software procurement and maintenance.

- High scalability: Cloud technology (like Ceph or Kubernetes) is designed for large scale deployments and incremental seamless expansion. This fits well for the unfathomable storage demands of digital research data.

- Long term maintainability: The large scale community efforts backed by cloud hyper-scalers promise long term support, and if everything fails, the source code is available so that it is possible to perform basic maintenance with university own staff.

On these premises, we started a multi-university consortium to establish research data infrastructures (RDI) at those five research universities, sharing a common open-source software stack in 2016 and acquired funding from the NRW state ministry of science (MKW) with a funding proposal citing various utilization scenarios (see [10] and [9]). This multi-university RDI, as an MKW funded project now referred to as datainfrastructure.nrw, was procured and made available to users in 2019. The approach was to have independent installations at the five universities of the RDI consortium, with only a basic agreement to be open to provide storage and compute resources to other consortium members to enable geo-redundancy. The charter of the project was to share the same technological and architectural concept, but maintain independence of installations for better flexibility and resilience. At Münster University, the RDI has been dubbed *Uni Cloud* and has since been expanded capacity-wise in multiple steps with various funding sources (university own, MKW funding dedicated for NFDI4BIOIMAGE or to implement a high availability two data center fail-over configuration to enhance cyber resilience - in total around 3 million Euros), now featuring 48 Petabyte of HDD space, 6.5 Petabyte of SSD space, 6.000 CPU cores and 90 Terabyte of RAM. Flexible expansion of the Uni Cloud in varying increments is made possible by the hyper-converged approach to the hardware platform, using large numbers of identically configured server systems that provide storage and compute resources.

The main rationale behind Uni Cloud and the Rector's policy on research data management is to provide the storage infrastructure needed to implement proper standards of research data management according to the FAIR (findable, accessible, interoperable, reusable) principles free of charge for all university researchers. To write the initial funding proposal for the RDI project, a wide survey amongst the research groups at all five universities was conducted to come up with numbers for capacity demands to base a sizing of the systems on. These initial estimates had to be adopted, since some proposed projects were not realized, but many more unexpected project materialized - some with previously unimaginable capacity demands (e.g. cryo electron microscopy). The close interaction of CIT with the members of the research groups through support channels (email and chat services) and the involvement of CIT in several large collaborative research centers (CRC funding line of DFG) provides the input needed to develop the Uni Cloud according to the user demands.

Here, we want to report on the details of the technical architecture and the state of user adoption and give an outlook on the next steps planned with the Münster Uni Cloud.

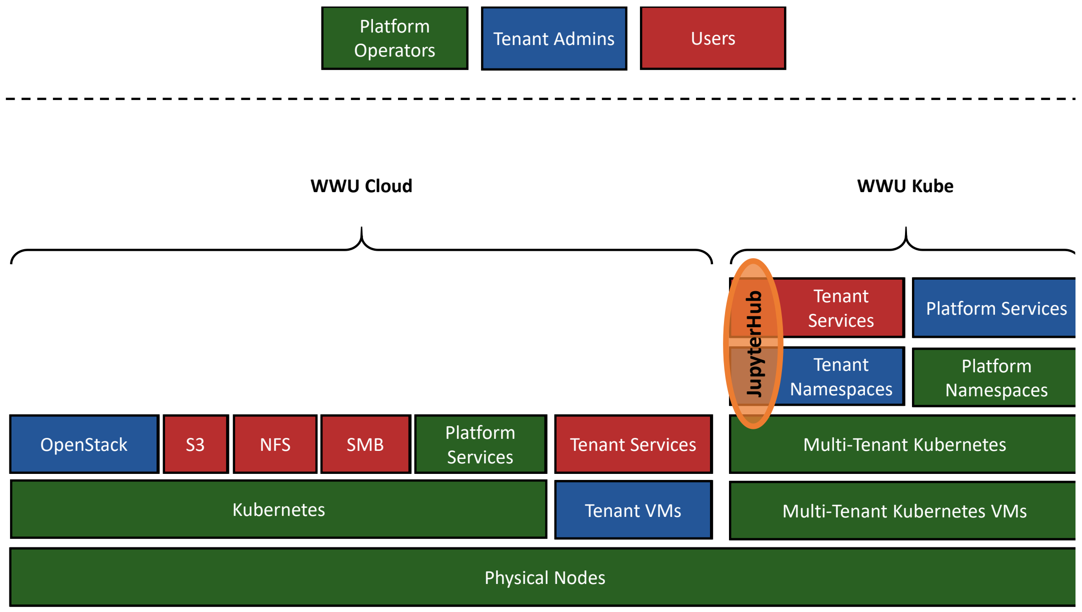

Figure 1: Graphical representation of the software architecture for Uni Cloud Münster OpenStack and Kubernetes. As a consequence of our hyper-converged design, all cloud platform components including storage and virtual machines run on the same hardware. JupyterHub runs as a service in Uni Cloud Kubernetes, which is a Kubernetes cluster running in virtual machines on Uni Cloud Münster.

2 Technical Architecture of Uni Cloud Münster

Uni Cloud Münster was designed using multiple abstraction layers, with each layer containing a specific set of services and providing a unique view onto the cloud system. For such a complex cloud system, clear abstraction layers facilitate updates, maintenance and bug fixing. Also users of the cloud system most often interact only with a single layer, limiting the required knowledge of the overall system. In addition, using well-known standards, like virtual machines and containers as well as widely used software for these layers, like OpenStack [7] or Kubernetes [3], users as well as admins benefit from extensive documentation on the internet.

2.1 Kubernetes on Bare Metal

At the bottom layer, we have a Kubernetes cluster running on bare metal, which is used to orchestrate Ceph storage services, Prometheus monitoring services, all OpenStack services and OVN/OpenVSwitch virtual network services. Using Kubernetes helped immensely in creating a dynamic system with high availability, that can scale all services on demand, balance requests among identical services, migrate services easily between hosts and distribute services for resilience against node failures.

We decided to use image-based deployment using Cluster-API [1] and Ironic. Images are built automatically within a CI pipeline and uploaded to a central location for deployment. Rollout of new images is done sequentially on the nodes of our clusters, starting on a development cluster and then progressing from least used to the most used. Using this method, we can find problems early and keep as many storage replicas as possible to prevent data unavailability in case of hardware failures during updates. As major advantages of image based deployment, we can easily roll back to a previous image in case of errors, redeploy a node quickly after an error condition/after testing and eliminate configuration drift. During updates, before a node is redeployed, virtual machines running on the nodes are automatically migrated, so there is no visible service interruption for users of VMs. Other services are migrated automatically by Kubernetes and sometimes require special configuration to prevent user-visible aborted connections. To deploy services, we use a GitOps [5] approach based on some custom tooling including helmfile with kustomize and ArgoCD. We pregenerate all Kubernetes resource files using this toolchain and commit the final resources to git, once we have checked against gatekeeper security policies, verified resource correctness and tested for deprecations.

Starting at the base layer, we implement tight security measures, to prevent attackers from gaining access to critical services or to slow them down, so we can quickly detect possible attacks and shutdown the affected parts. For example, after completing an image build, we use multiple scanners to detect vulnerabilities in our images. We also create and upload SBOMs, so we can periodically check for new vulnerabilities. Security updates are quickly built using the CI pipelines and can be deployed without user intervention using Cluster-API, allowing for a fast response time to newly found CVEs. As special hardening measures, we use signing/verification of all components during the deployment and boot process as well as LUKS encryption for all storage devices. We further use IMA signing for all non-modifiable files in the operating system and SELinux enforcement. In Kubernetes, we make use of network policies for all services and enforce strict security policies on all resources. As a last line of defense, we use eBPF based intrusion detection software Falco to report unusual activity on our systems. Kernel Audit Logs, Kubernetes Audit Logs and other logs are gathered in our central logging system, distributed over multiple locations, so that a local attacker cannot tamper with them.

2.2 OpenStack for Virtual Machines

The next layer is OpenStack, which provides users with virtualized hardware and the possibility to create virtual machines within some previously set quotas for their project. For many users, this is a familiar abstraction interface, allowing deployment of all operating systems supporting cloud-init on virtual hard disks and using virtual network adapters, attached to a virtual network. However, we also use OpenStack to manage network shares, which can be mounted in virtual machines but can also be used stand-alone. The latter is commonly used by researchers, which only require a large storage space but no VMs.

We provide two kinds of storage options within OpenStack: Block storage via OpenStack Cinder and filesystem storage via OpenStack Manila. Both kinds are backed by Ceph storage services in the base layer.

Block storage is provided through Ceph RBD in five-fold replicated Ceph pools on hdd or nvme disks. This massive overhead is necessary, as Ceph always requires at least $n/2+1$ replicas for data consistency during disk writes. During updates, where there is always a single node unavailable, we want to tolerate another node failure without service interruption. This has proven not to be an edge case, but has already been observed. We advertise block storage for small to medium size data, e.g. operating system or database volumes.

File storage is realized using the CephFS in 8+3 erasure coded Ceph pools on hdd disks. As these pools are used for large data, this is a good compromise between disk usage and redundancies, as repair times after disk failures are typically less than 6 hours. The file shares can be mounted via NFS or CephFS as required. We opted to implement an additional special kind of file shares called ”usershares”, where data is stored using centrally managed POSIX ids. These shares are mounted on our HPC cluster and JupyterHub and exported via SAMBA servers and Ganesha NFS (with Kerberos), so they can be accessed directly from all workstations. This provides an easy and flexible way to store and process large datasets for university members.

Our virtual network is based on OVN, which configures OpenFlow Rules in OpenVSwitch and configures overlay tunnels based on configuration data from OpenStack Neutron. This is done by generating a logical representation of network hardware based on the Neutron data. This logical configuration is then converted into a flow-like representation by a central OVN daemon. On the individual hosts, OpenFlow rules are generated from the latter data. Using the provided tooling, each packets traversal through the network stack, from external interfaces, over virtual routers, NAT and ACLs to virtual machines and reverse can be analyzed, providing helpful insights in case of complex network problems.

2.3 Kubernetes for Containerized Services

At the uppermost layer, we run a multi-tenant Kubernetes cluster within OpenStack. Kubernetes, developed and offered as a platform by all big cloud providers, provides users with a standardized API, for users to run their own services. We operate a single big cluster, which is centrally managed and updated. Namespaces along with quotas are assigned to users on demand. This has the main advantage, that updates and security configuration is centrally managed, which is especially important as service operators at the university typically have no strong IT background.

Kubernetes requires only few components which have a dependency on the actual cloud provider, like interfaces for deployment of virtual machines or managing storage via CNI plugins, making the deployment (at least in principle) portable to public cloud providers like Google, Amazon or Microsoft. We use Cluster-API as in the base layer for automatic deployment, giving strong administrative synergies and similar advantages. Cluster-API has backends for all big cloud providers and is therefore an ideal choice for future platform independence.

For increased reliability of services, we operate multiple Kubernetes clusters on independent OpenStack sites. These clusters are connected via Cilium and Istio, so requests can be automatically loadbalanced and failed over in case of network or power failures. LoadBalancing is currently achieved using global DNS with health checking but may soon be replaced by using Cilium BGP for better stability. We discussed having a single OpenStack deployment with multiple availability zones, but decided against that for increased resilience to errors on software updates.

Security and configuration policies are enforced via Gatekeeper/OPA to harden the overall system and prevent accidental misconfiguration. For more sensitive services and basically all centrally operated services, we also designed a more strict hardened policy, enforcing things like a read-only root filesystem or running as non-root. Like in the baremetal cluster, we enabled SELinux enforcement, collect audit logs and use Falco for intrusion detection. Due to using GitOps security updates are quickly deployed and also rolled back in case of problems. This also applies to image updates, as the rollout of new virtual machines takes only around 2 minutes.

Authentication for web services is optionally provided by using SATOSA as an OIDC wrapper to DFN AAI / edugain, authenticating users if possible at the edge and authorizing internally via JWT identity. Istio mTLS together with AuthorizationPolicies and Cilium NetworkPolicies provide authorization and security between the central ingressgateway and services or between services.

To facilitate monitoring, we rolled out a central multi-tenant Grafana, Mimir, Loki, Tempo

Stack. We also configured multi-tenant alerting, so that alerts can be generated based on metrics and logs. All local observability services like Prometheus or Vector are automatically configured to push data into this central system, so service operators can readily see metrics and logs of their services. For redundancy, the monitoring stack is distributed over all our cluster sites.

3 User Adoption

User adoption was very strong from the very start and is still gaining momentum. There has been substantial churn in the projects using the Uni Cloud data infrastructure from the original funding proposal to the current situation - a few projects dropping out or not showing substantial resource usage, but mostly new, unexpected projects joining in, driving the resource usage far beyond the originally expected limits.

There are currently approx. 250 projects and 5.5 Petabyte of user data from basically all faculties of Münster University, with the medical faculty having developed into the most demanding user group with 51 projects and 2.2 Petabyte of data.

Some interesting usage scenarios of Uni Cloud include the following:

- Collaborative Research Centers (CRCs): In this prestigious funding line of DFG (german research fund) for topical large scale research endeavours, it is possible to propose special INF projects for data infrastructures. CIT is currently providing the INF projects for two CRCs (with further participations in progress), which utilize the data storage of Uni Cloud, but also the remote visualization functionality for collaborative access to common cloud stored research data for dislocated researcher. With JupyterLab [4] and Horizon VDI, a web browser based access to centrally provided graphical data analysis and visualization software packages that can directly work on the cloud stored data sets is possible, creating new opportunities for collaboration. Community outreach will also be improved by allowing public access to selected data sets in the form of Atlas applications.
- Medical faculty research data: The medical faculty is located on premises of the university hospital and usually receives IT support by their clinical IT support. Security standards and pricing are adapted to clinical services, so that for research data the Uni Cloud has become a key option, documented by the dominant share in resource usage. The largest user is now the cryo electron microscopy facility (cryo-EM) which only ramped up their operations in early 2023 and has already stored some 900 Terabyte of data, with a plan to use 2 Petabyte of storage for retention of 2 years of data production. Up to now, only medical faculty research data that is not patient related may be stored on the Uni Cloud - but we are currently conducting risk assessments with medical projects that should open up the possibility to also store pseudonymized patient medical data.
- High Performance Computing (HPC) and Artificial Intelligence (AI): The cloud was designed with high bandwidth interconnection to the university HPC system - this makes it well suited to offload cold data from the expensive parallel HPC storage (using spectrum scales information lifescycle management (ILM) features), but also to quickly copy data sets stored in the cloud over to HPC for processing (synchronous volume mounts did not prove performant enough). This is especially well received by the various groups researching in AI, who are using the GPU resources of the central HPC system.
- JupyterHub: Many AI groups also rely on the JuypterHub [6] that has been set up on the Uni Cloud with direct access to the cloud storage as well as the HPC parallel file system.

The Uni Cloud JupyterHub [2] is not only used for research (in AI and life sciences - like gene sequence analysis with R packages), but also for teaching of programming courses.

- Research data management: The research data management support group of the university library has implemented (together with CIT and in the context of the sciebo.RDS project) RDM tools, that are based on the the Uni Cloud - like a data repository (invenio RDM), and a long term data retention service to comply with funding requirements (10 years data safe). The latest addition is an electronic lab notebook service using eLabFTW. The Münster imaging network, coordinating the use of visible light microscopy systems, is also hosting their OMERO image data base service on the Uni Cloud.
- NFDI: For two years now, a federal funding line for a National Research Data Infrastructure (NFDI) is established in Germany, that develops RDM tools in the context of around 30 research discipline specific consortia. CIT is part of PUNCH4NFDI (particle, astro-particle, nuclear physics) and NFDI4BIOIMAGE (microscopy) and is pledging cloud resources (storage and compute) for these NFDI communities, enabled by MKW funding through the project nfdi4bioimage.nrw which allowed to add some 6 Petabyte of storage and 800 CPU cores to the datainfrastructure.nrw specifically to provide physical infrastructure for NFDI.
- GITlab: The CIT GitLab uses the Uni Cloud to host the runners and is the key platform for GitOps.
- OpenCast: The opencast platform for storage and streaming of lecture recordings has proven to be very helpful in the pandemic and benefited from the elasticity of resources of the Uni Cloud in the high demand period. In the educast.nrw project, this platform was also made available to multiple other universities. It benefits from the high scalability of cloud storage and compute (for video transcoding applications).
- Video conferencing: As on-premises alternatives to commercial products, we host cloud scalable deployments of Jitsi and BigBlueButton on the Uni Cloud.

4 Outlook

The Uni Cloud has received a very quick adoption and this is gaining further momentum. The recent hardware procurements are being brought into production as resource demand grows - but there is still enough reserve for several years of even increasing data storage demand. The currently stored 5.5 Petabyte of user data consume approx. 9 Petabyte of raw HDD space and we currently have 48 Petabyte of raw storage space - but the data production rate is increasing, and for certain sensible data sets a replication at two data center location will become standard in the future, again increasing storage demand.

The IT staff for the Uni Cloud has grown considerably over the years - starting with one seminal person in 2018, the number of staff members working in the context of the Uni Cloud - with financing from research and development projects or grants from the university rectorate - has now reached a total of seven.

And there are new endeavours ahead, that will potentially also add to the Uni Cloud staff:

- JupyterHub.nrw: In this MKW NRW funded project, we will establish a JupyterHub infrastructure with the necessary storage and compute (CPU and GPU) resources for all

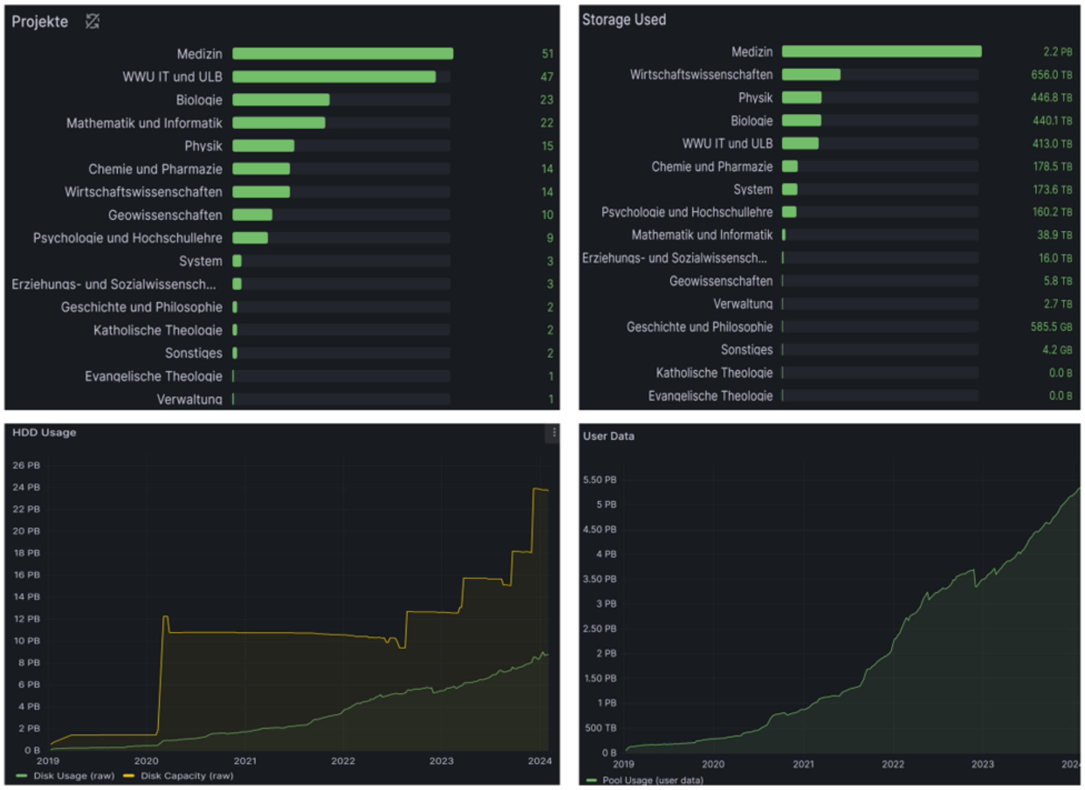

Figure 2: Grafana statistics on the number of OpenStack projects (total approx. 250) and Ceph user data storage volume by faculties. The development of storage disk capacity provided (with multiple expansion steps), disk raw usage and user data stored are shown over time since production start in 2019.

universities in NRW. This will mainly support teaching by providing the infrastructure for massive courses, but will also give young researchers low threshold access to compute resources e.g. for AI focused projects. An ambitious extension to this project is the evaluation of this cloud-based JupyterHub service for conducting online assessments and exams.

- GIT.nrw: CIT is part of a MKW funded project GIT.nrw to establish a GITlab platform for all NRW universities and has pledged to host part of this infrastructure on the Uni Cloud.

- UniGPT: With growing demand for AI services by university faculty and students, especially for access to large language models (LLM), hosting such a service on-premises on the Uni Cloud, using recently made available open-source solutions like META's LLAMA 2 and Mistrals Mixtral 8x7B, has become an important goal for CIT and we report in another conference paper on this.

5 Acknowledgements

- We thank the Ministry for Art and Science (MKW NRW) for funding the sciebo.nrw, datainfrastructure.nrw, NFDI4BIOIMAGE.nrw and JupyterHub.nrw projects.

References

[1] Cluster api. `https://github.com/kubernetes-sigs/cluster-api`.

[2] Markus Blank-Burian, Jürgen Hölters, and Raimund Vogl. Jupyterhub on an on-premises cloud – a special focus on gpu accelerated machine learning and 3d visualization. In S Bolis, J-F Desnos, L Merakos, and R Vogl, editors, *Proceedings of the European University Information Systems Conference 2021*, pages 69–76. EasyChair, 2021.

[3] Brendan Burns, Joe Beda, and Kelsey Hightower. *Kubernetes: up and running: dive into the future of infrastructure*. O'Reilly Media, 2019.

[4] B Granger and J Grout. Jupyterlab: Building blocks for interactive computing. *Slides of presentation made at SciPy*, 2016.

[5] Thomas A Limoncelli. Gitops: a path to more self-service it. *Communications of the ACM*, 61(9):38–42, 2018.

[6] Jeffrey M Perkel. Why jupyter is data scientists' computational notebook of choice. *Nature*, 563(7732):145–147, 2018.

[7] Tiago Rosado and Jorge Bernardino. An overview of openstack architecture. In *Proceedings of the 18th International Database Engineering & Applications Symposium*, pages 366–367, 2014.

[8] Anne Thoring, Dominik Rudolph, and Raimund Vogl. David against goliath: How a university cloud succeeds in the niche of higher education – a user survey. In Jean-François Desnos, Ramin Yahyapour, and Raimund Vogl, editors, *Proceedings of the European University Information Systems Conference 2022*, pages 34–43. EasyChair, 2022.

[9] Raimund Vogl, Jürgen Hölters, Markus Ketteler-Eising, Dominik Rudolph, Markus Blank-Burian, Holger Angenent, Christian Schild, and Stefan Ost. Blueprint for an open source on-premise cloud infra-structure to serve as a research data infrastructure for universities. In *Proceedings of the European University Information Systems Conference 2019*. EUNIS, 2019.

[10] Raimund Vogl, Dominik Rudolph, and Anne Thoring. Bringing structure to research data management through a pervasive, scalable and sustainable research data infrastructure. In *The Art of Structuring*, pages 501–512. Springer, 2019.

6 Biographies

M. Blank-Burian is a research assistant at the it department of the University of Münster (CIT) and lead cloud architect of the Uni Cloud Münster. He graduated from University of Münster (Germany) in 2013 with degrees in both physics and computer sciences. In 2018, he received his Ph.D. in theoretical physics. His research focuses on private clouds and cloud native technology. More info: https://www.uni-muenster.de/forschungaz/person/17330

R. Vogl holds a Ph.D. in elementary particle physics from the University of Innsbruck (Austria). After completing his Ph.D. studies in 1995, he joined Innsbruck University Hospital as IT manager for medical image data solutions and moved on to be deputy head of IT. He served as a lecturer in medical informatics at UMIT (Hall, Austria) and as managing director for a medical image data management software company (icoserve, Innsbruck) and for a center of excellence in medical informatics (HITT, Innsbruck). Since 2007 he has been director of the it department of the University of Münster (CIT, Germany). His research interests focus on management of complex information systems and information infrastructures.