This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 777533.

PROviding Computing solutions for ExaScale ChallengeS

D5.1	Design of data infrastructure for extreme-large data sets		
Project:	PROCESS H2020 – 777533	**Start / Duration:**	01 November 2017 36 Months
Dissemination[1]:	Public	**Nature[2]:**	**R**
Due Date:	31st July 2018	**Work Package:**	**WP 5**
Filename[3]	PROCESS_D5.1_Design_of_data_infrastructure_v1.0.docx		

ABSTRACT

Deliverable D5.1 presents the different design options for the data infrastructure for extreme-large datasets (Delivery and Storage) and ranks them based on their ability to fulfil the requirements of the use cases of the project. The document builds upon the analysis summarized in D4.1. Based on the data requirements analysis of the five use cases described in Section 1 in D4.1, D5.1 provides a concrete software architecture including some potential technology solutions for the implementation of the first PROCESS prototype. For each technology choice, we will not only consider functional application related requirements but also the maturity of the technology (presented as an estimated TRL level of the solution) and the ease of integration with the other relevant European data/compute solutions. The outcomes of D5.1 will also serve as a starting point for the preparation of D4.2 and D6.1, which are due end M12.

This version is a draft of D5.1 and is under review.

[1] PU = Public; CO = Confidential, only for members of the Consortium (including the EC services).

[2] R = Report; R+O = Report plus Other. Note: all "O" deliverables must be accompanied by a deliverable report.

[3] eg DX.Y_name to the deliverable_v0xx. v1 corresponds to the final release submitted to the EC.

Deliverable Contributors:	Name	Organization	Role / Title
Deliverable Leader[4]	Adam Belloum, Reggie Cushing, Ali Rahmanian	UvA	Coordinators
Contributing Authors[5]	Martin Bobak, Ondrej Habala	UISAV	Writer
	Jan Meizner, Bartosz Wilk	AGH	Writer
Reviewer(s)[6]	Maximilian Höb	LMU	
	Jason Maasson	eScience Centre	
Final review and approval			

Document History

Release	Date	Reasons for Change	Status[7]	Distribution
0.0	2018-05-17	Structure of the deliverable fixed	Draft	
0.1	2018-05-24	Initial text to Section 1, 2, and Appendix 4.1	Draft	
0.2	2018-06-24	improved text Section 1, 2, 3 and Appendix 4.1	Draft	
0.3	2018-06-30	Section 1,2,3 and Appendix 4.1 complete draft	Draft	
0.4	2018-07-06	Draft completed for internal review	In Review	
0.5	2018-07-23	Reviewed Draft	In Review	
1.0	2018-07-30	Final Version	Released	

[4] Person from the lead beneficiary that is responsible for the deliverable.
[5] Person(s) from contributing partners for the deliverable.
[6] Typically, person(s) with appropriate expertise to assess the deliverable quality.
[7] Status = “Draft”; “In Review”; “Released”.

Table of Contents

Executive Summary

Due to energy limitation[8] and high operational costs, it is likely that exascale computing will not be achieved by one or two datacentres but will require many more. A simple calculation, which aggregates the computation power of the 2017 Top500 supercomputers[9], can only reach 418 Petaflops[10]. Companies like Rescale, which claims 1.4 exaflops of peak computing power, describes its infrastructure as composed of 8 million servers spread across 30 datacentres[10]. Any proposed solution to address exascale computing challenges has to take into consideration these facts and by design should aim to support the use of geographically distributed and likely independent datacentres. It should also consider whenever possible the co-allocation of the storage with the computation as it would take ~3 years to transfer 1 exabyte on a dedicated 100Gb connection on the GEANT network. This means we have to be smart about data and computation placement. As the natural settings of the PROCESS project is to operate within the European Research Infrastructure and serve the European research communities facing exascale challenges, it is important that PROCESS architecture and solutions are well positioned within the European computing and data management landscape namely PRACE, EGI, and EUDAT. Interoperability of the PROCESS architecture and solutions with these bodies and their infrastructures and services is of extreme importance for the achievement of PROCESS objective.

Deliverable D5.1 provides a concrete software architecture including some potential technology solutions for the implementation of the first prototype of the PROCESS data infrastructure. In this regard, D5.1 build upon the analysis summarized in D4.1 and especially upon the data requirements analysis of the five use cases described in Section 1 in D4.1. For each technology choice made in D5.1, we do not only take into consideration the requirements needed to support the five PROCESS use cases, but also the maturity of the technology (TRL level) and the potential integration with existing European solutions. The outcomes of D5.1 serve as a starting point for the preparation of D4.2 and D6.1, which are due end M12.

[8] With today's HDD capacity of 10TB, an exabyte storage means at least 104,857 HDDs without backup or RAID. Each HDD consumes ~7W which means ~0.7MW for just spinning HDDs

[9] https://www.top500.org/lists/2017/06

[10] https://www.top500.org/news/looking-for-an-exaflop-this-cloud-provider-has-it/

List of Figures

List of Tables

1 Data Requirements in PROCESS use cases and beyond

Within the PROCESS project we have defined 5 potential exascale data applications representing different scientific domains namely medical imaging, astronomy, Industrial (Airline domain), and Agricultural Observation and Prediction. All these applications are facing the data challenges either at this moment or will face data and compute challenges soon due to the expected increase of the data sets. Based on the use case description presented in Deliverable D4.1, we derived a number of data requirements that will guide the design of this PROCESS data infrastructure.

In Table 1, we summarize the characteristics of the data sets used the 5 different use cases in the light of the **NIST Big Data Interoperability Framework: Volume 1, Definitions**[11], which reference to the Volume, Variety, Velocity and Variability as the main characteristics of Big Data:

Table 1: Main data characteristics of the use cases

	UC#1: **Exascale learning on medical image data**	UC#2: **Square Kilometre Array/LOFAR**	UC#3**: Supporting innovation based on global disaster risk data**	UC#4: **Ancillary pricing for airline revenue management**	UC#5**: Agricultural analysis based on Copernicus data**
Volume	3.5TB	~ 28 PB	1.5 TB (minimum)	~3TB	10PB
Variety	files[12]	Files[13]	files	stream	Files
Velocity[14]	Low	Low	Low	Medium	Low
Variability[15]	Low	Low	Low	Low	Low
Growth	2TB/year[5]	5-7PB/year	1TB/year	1TB/year	1TB/year

Data management requirements are also reported for other exascale applications in the U.S DoE reports published in 2016 like the one for High-energy physics (HEP)[16], and Biology and environmental research[17]. Both reports mention **data access** and **movement** as a key element in dealing with growth of datasets. For the HEP community, the Large Hadron Collider (LHC) at CERN will continue to be the largest producer it is expected that in the future (roughly 2025-35) HL-LHC each experiment will transition from O(100) petabytes to O(1) exabyte of data. The Report states the infrastructure requirements for exascale of the HEP community for 2020 and 2025. The HEP community has developed data storage and movement services, ROCIO[18], to meet its needs in the 2020 timescale. In the Reports of the Biology and environmental research it is clearly stated that similar algorithmic barriers (lack of scalable solver algorithms and I/O) that challenged petascale performance will be faced again at exascale level, with the additional constraints introduced by accelerators and hierarchical memory.

[11] https://bigdatawg.nist.gov/_uploadfiles/NIST.SP.1500-1.pdf

[12] BIGTIFF Whole Slide Images. The growth is estimation based on these two datasets, taking into consideration the whole histopathology applications dataset sizes and growth rates are much larger.

[13] SKA/LOFAR use case the data is stored into geographically distributed archives (tapes)

[14] Velocity: is the rate of flow at which the data is created, stored, analysed, and visualized, Section 3.3.2 page 15, https://bigdatawg.nist.gov/_uploadfiles/NIST.SP.1500-1.pdf

[15] Variability: refers to any change in data over time, including the flow rate, the format, or the composition, Section 3.3.2 page 15, https://bigdatawg.nist.gov/_uploadfiles/NIST.SP.1500-1.pdf

[16] https://www.osti.gov/servlets/purl/1341722

[17] http://blogs.anl.gov/exascaleage/wp-content/uploads/sites/67/2017/05/DOE-ExascaleReport_BER_R27.pdf

[18] Rucio, the next-generation Data Management system in ATLAS, Nuclear and Particle Physics Proceedings Volumes 273–275, April–June 2016, Pages 969-975, https://doi.org/10.1016/j.nuclphysbps.2015.09.151

2 PROCESS Data infrastructure

When we think of storage in an exascale context, we must first consider what is exascale in storage terms. An exabyte of data is approximately 1 million terabytes which means around 100,000 10 terabyte hard disks without even considering any resiliency. Transport of an exabyte of data over dedicated 100Gb link will take around 3 years. At this scale, energy, MTBF, management, transport, redundancy and security become an even greater challenge. Furthermore, the high variety in data-oriented applications means that many applications have different data requirements. The European research landscape is fragmented over several countries and independent institutions. This further exacerbates the exascale challenge since scaling out storage and computing resources is rather an administrative challenge rather than, only, a technological one as we have experienced in the European Grid era. The path towards exascale in Europe is to loosely and smartly federate resources over independent, autonomous infrastructures while also accommodating the plethora of different data requirements stemming from different applications.

PROCESS data infrastructure has to be scalable, reliable, and easy to be installed and integrated which commonly used Data stack solutions by the scientific research community. A number of data storage and access technologies have been established and are used by different application developers ranging from commercial solution like Amazon S3, Dropbox, and Google drive to open source ones like ownCloud, nextCloud, Onedata or B2DROP EUDAT storage service. In order to achieve the goal of PROCESS project, which is to enable exascale data application, we propose a Data architecture, which is application centric whereby an application distributed infrastructure manages its view on the data – a file system or a database - while the underlying data system provides the raw storage. PROCESS Data architecture is not aimed at inventing yet another data management stack, it is targeted at re-using and developing further well-established open source technologies for data storage and access. In this deliverable we will focus describing in more detail the Data management sub-component of the initial PROCESS architecture described in Deliverable D4.1 page 54, (Figure 1). We will present in the following sections the technology selections and choices we are considering for the implementation of three main components of PROCESS data management infrastructure, namely: Data federated management, meta-data, and access to unsupported data storage types (compound data formats and databases).

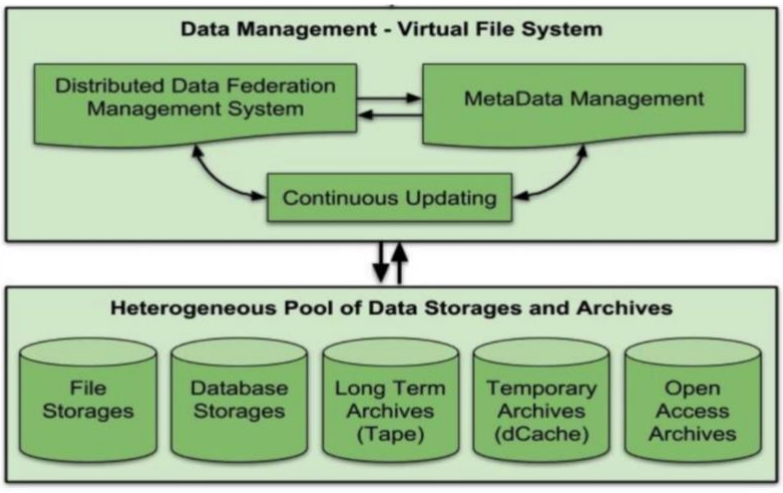

Figure 1: Data management sub-component of the initial PROCESS architecture described in Deliverable D4.1 page 54

2.1 Data federated management system

This deliverable describes an initial architecture of the PROCESS Data Federated Management Systems (PROCESS-VFS), which aims to provide the application developers within PROCESS with a quick solution which can enable them to start developing and testing their applications while benefiting from the storage provided by PROCESS data storage infrastructure owners. We will also describe the final Data federated management system, which will be developed during the course of the project and will substitute the initial implementation by the end of the project. With the Data requirements described in Deliverable D4.1 in mind, PROCESS-VFS should have the following characteristics: (1) Federated access to data storage, (2) Scalability, (3) Reliability (fault tolerance), (4) Standard interfaces, (5) Multiple backend (storage technology).

The PROCESS Data Federated Management Systems (PROCESS-VFS) should allow access and data sharing between de-centrally stored Data. It is likely that data used in exascale applications is stored across datacentres in various data storages. It is thus important that PROCESS-VFS abstracts the data storage technology allowing applications to work with data stored in various technologies. Recent achievements in data managements systems allow implementing such features.

2.1.1 Initial Architecture PROCESS-VFS

Data stacks have evolved over the past two decades. This change has been driven mainly by virtualization and the dynamicity that this provides. For example, the grid in Europe was a static infrastructure with homogenous middleware and strong collaborations between site administrators. This means that from raw storage on disks to applications several controllers for the same community and applications were developed to fit the infrastructure, in the grid case, applications were mostly batch style. With virtualization and new application paradigms, this static infrastructure was not adequate. A paradigm shift in computing also meant that users and scientists can compute anywhere, resources were made available, which could be of any type like public cloud, private infrastructure or simple laptops, while data could be fragmented over different infrastructures. Data management in this paradigm becomes a problem especially with huge amounts of data, which begs the question: how do we consolidate and compute on such data? Here we present our solution for this challenge.

In Deliverable D4.1, we extensively described the LOBCDER solution, developed in the context of the EU project VPH-share[19] (2011-2016), which has been used in the VPH production environment between for more than three years. The LOBCDER solution helped to federate access to data stored across multiple datacentre composing the VPH-share Platform[20]. Parts of the LOBCDER solution like authorization and the metadata management are specific to the VPH-share project and need to be replaced in the PROCESS by news solutions. In the meantime, a number of similar solutions emerged and gained popularity like ownCloud (2012), nextCloud (2016), and Onedata (2016). Mainly these technologies, including LOBCDER, help synchronize data (files) between end-users (desktop) and one or more remote data servers geographically and administratively distributed, while allowing an easy sharing of the files among users or groups of users. These technologies, except for Onedata, focus mainly on users' access through WebDAV protocol and are not designed to be used for staging in and out data at runtime due to overhead that can be incurred by the WebDAV. Onedata provide extra capabilities to help staging in and out data to compute nodes transparently and efficiently. More mature technologies enable high-throughput transfers also

[19] http://www.vph-share.eu

[20] M. Koehler, R. Knight, S. Benkner, Y. Kaniovskyi and S. Wood, "The VPH-Share Data Management Platform: Enabling Collaborative Data Management for the Virtual Physiological Human Community," *2012 Eighth International Conference on Semantics, Knowledge and Grids*, Beijing, 2012, pp. 80-87. doi: 10.1109/SKG.2012.51

exist and need to be investigated like EUDAT B2Stage[21], dCache[22], GlusterFS[23], and IRODS[24]. It is worth to note here that most of these high-throughput oriented data transfers are not trying to serve the users access directly rather they are integrating common user access centred systems like ownCloud or nextCloud. For instance, EUDAT provides two different services B2Stage and B2Drop, the first is a reliable, and easy-to-use service to transfer research data sets between EUDAT storage resources and high-performance computing centres, and B2Drop for secure data exchange and synchronization with other researchers. B2Drop is based on ownCloud technology. A Similar approach is followed by dCache, which announced in 2016, that dcache will be operating with ownCloud[25] to enable user access.

For the first implementation of the PROCESS-VFS, we will follow a similar approach, which can guarantee that we can have a working prototype by the end of the first year of the project. To keep the implementation simple, we consider architecture with two separate data management stacks one dedicated to user access the second dedicated to application access (Figure 2).

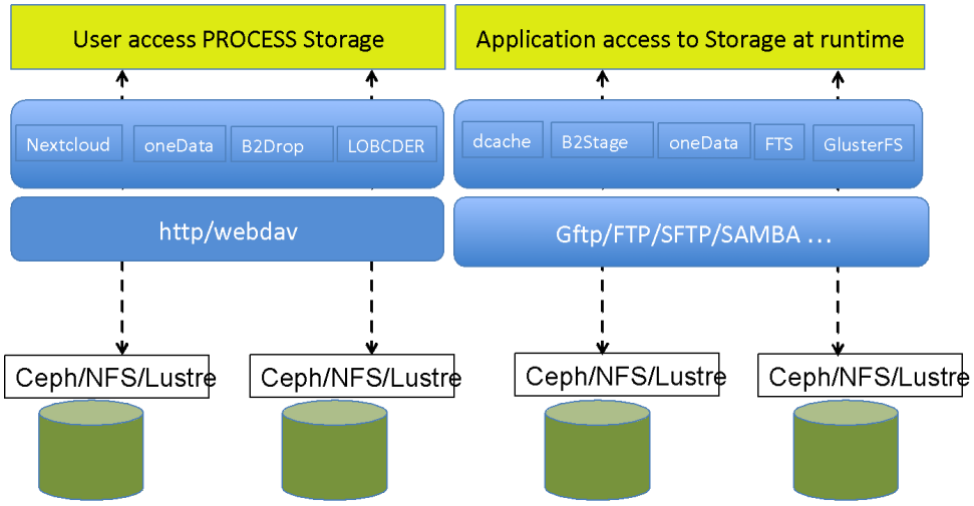

Figure 2: Common approach to data management, two separated data stacks one for user access the second for application access at runtime

In the following paragraphs, we describe three alternatives to implement the architecture described in Figure 2. In Appendix 4.1 we present the review of the various emerging technology and compare them in order to select the one that can serve exascale data applications.

2.1.1.1 PROCESS-VFS based on Onedata

Among the technologies selected, we have found out that Onedata has most of the features to support immediately the PROCESS application. Onedata system has been designed specially to satisfy requirements of data globalization and high-performance access[26]. Using the Onezone concept we can create a PROCESS federation and attach multiple backends (Figure 3). The Onedata Global Registry allows to offer users globally unified space over

[21] https://www.eudat.eu/services/b2stage
[22] https://www.dcache.org
[23] https://www.gluster.org
[24] https://iRODS.org
[25] https://www.dcache.org/manuals/2016/presentations/20161006-PM-dCache.pdf
[26] Dutka, Ł., Wrzeszcz, M., Lichoń, T., Słota, R., Zemek, K., Trzepla, K., Opioła, Ł., Słota, R. and Kitowski, J., 2015. Onedata–a step forward towards globalization of data access for computing infrastructures. Procedia Computer Science, 51, pp.2843-2847

geographically distributed and independent data centres[27]. Onedata Client (Oneclient), by querying the global registry, can identify the most appropriate provider service the user requests. The users have access directly to the selected Oneprovider service to access the data, which can be stored locally, or on any other Oneprovider service part of the configured configuration[28].

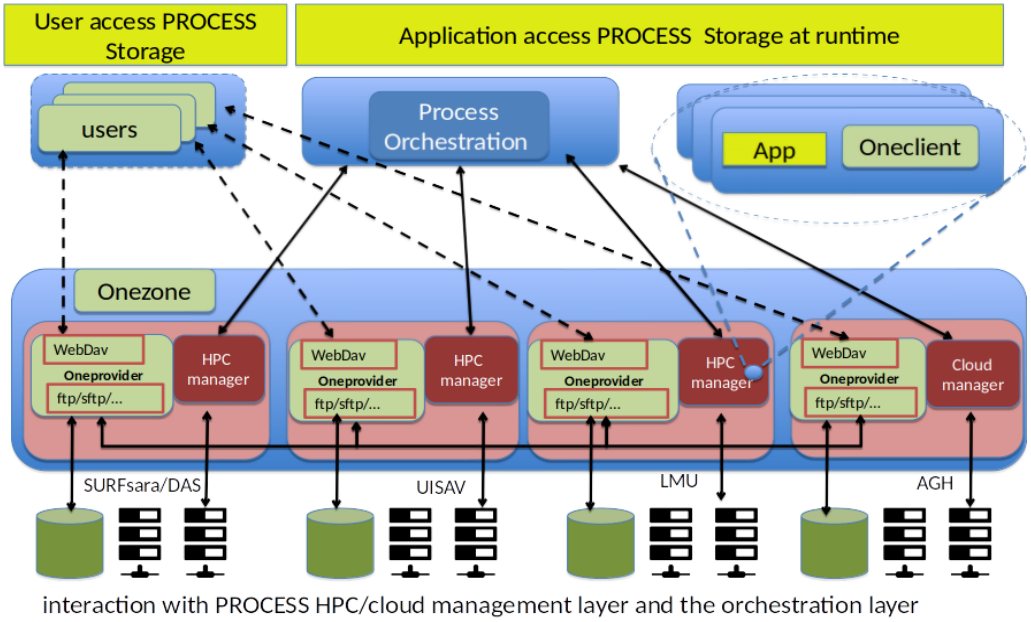

Figure 3: Implementation of the initial PROCESS-VFS using Onedata technology

2.1.1.2 PROCESS-VFS based on nextCloud

Since May 2017, nextCloud announced[29] the "global Scale architecture" as part of nextCloud 12. The global scale architecture was introduced to overcome the limitation on the number of users of a nextCloud installation, storage costs and global distribution. The new architecture helps to solve the need for data distributed over multiple datacentres (Figure 4). Global scale architecture uses multiple independent application servers, running on commodity hardware. The architecture is composed of a global site selector, a lookup server, and a balancer. A global site selector redirects users to the right data location. Users are authenticated through a central directory; their information is retrieved and they are redirected to the right storage provider. The balancer monitors the nodes (storage, network traffic, CPU and RAM) and initiates a migration of user accounts if needed. To avoid access to files at runtime via WebDAV, we can use other more suited protocols and periodically instruct nextCloud servers to scan for new files[30].

[27] Wrzeszcz, M., Trzepla, K., Słota, R., Zemek, K., Lichoń, T., Opioła, Ł., Nikolow, D., Dutka, Ł., Słota, R. and Kitowski, J., 2015, September. Metadata Organization and Management for Globalization of Data Access with Onedata. In International Conference on Parallel Processing and Applied Mathematics (pp 312-321). Springer, Cham

[28] Wrzeszcz, M., Opioła, Ł., Zemek, K., Kryza, B., Dutka, Ł., Słota, R. and Kitowski, J., 2017. Effective and Scalable Data Access Control in Onedata Large Scale Distributed Virtual File System. Procedia

Computer Science, 108, pp.445-454.

[29] https://nextcloud.com/blog/nextcloud-announces-global-scale-architecture-as-part-of-nextcloud-12/

[30] https://ownyourbits.com/2017/04/18/different-ways-to-access-your-nextcloud-files/

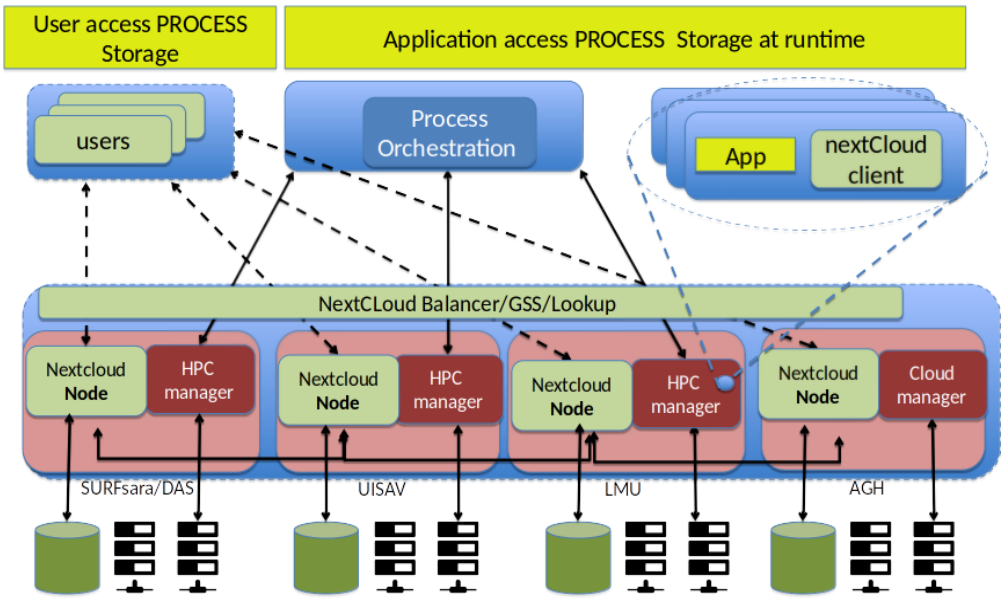

Figure 4: Implementation of the initial PROCESS-VFS using nextCloud technology

2.1.1.3 PROCESS-VFS based on EUDAT services

EUDAT offer a set of basic services we can use to implement the PROCESS-VFS namely B2Drop (for user access) which uses ownCloud and WebDAV protocol and B2stage (high-throughput data transfers) which uses iRODS. Other services like B2Find and B2authen can be used to locate and authenticate data on PROCESS Storage.

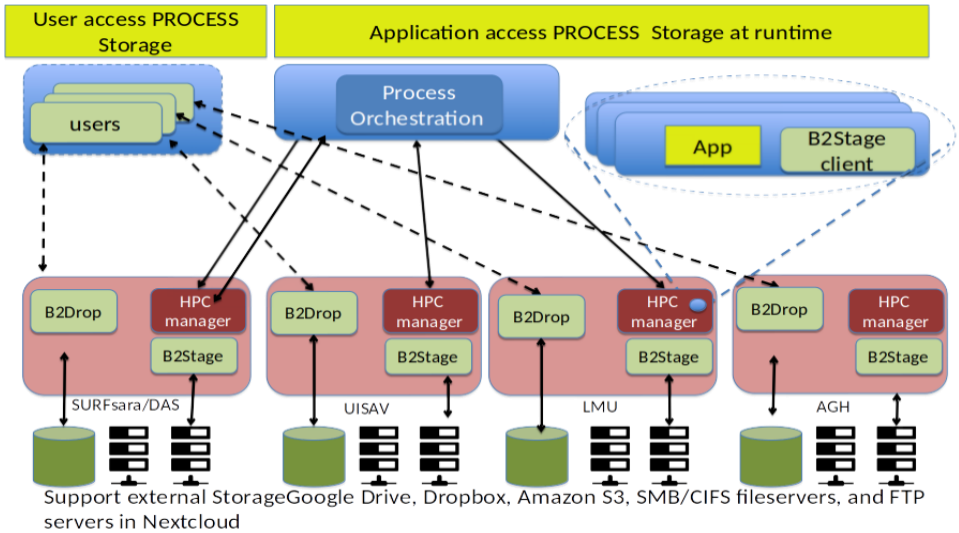

Figure 5: Implementation of the initial PROCESS-VFS using EUDAT service

Table 2 summarises the characteristics of selected technologies of interest for the PROCESS project.

Table 2: Summary of the features of the technologies selected to be used as a starting point in the PROCESS Project

	LOBCDER	ownCloud	nextCloud	Onedata	dCache	iRODS
File access control	No	No	Yes	Yes	Yes	Yes
Resource monitoring	No	No	Yes	No	MonAMI dCache plugin -dCache web interface -Storage monitoring can be provided by SRM tool	Yes
Storage back-ends/ API	WebDAV	Amazon S3, Drobox, FTP/FTPS, Googledrive, Local, Open stack object storage, ownCloud, SFTP, SMB/CIFS, WebDAV	Amazon S3, Drobox, FTP/FTPS, Googledrive, Local, Open stack object storage, ownCloud, SFTP, SMB/CIFS, SMB/CIFS using OC login, WebDAV	Amazon S3, NFS, Lustre, Ceph, Openstack SWIFT	NFS v4.1, HTTP and WebDAV, GridFTP, xrootd, SRM, dCap and GSIdCap	FUSE
User access	WebDAV	WebDAV, ownCloud Web Interface	WebDAV, nextCloud web interface	WebDAV	NFS, Http, WebDAV, GridFTP, xrootd, SRM, dCap, GSIdCap	username/password, LDAP, GSI, Kerberos
Anonymous upload **Open Source**	No	Yes	Yes	No	Yes	No
Authentication and Authorization	username/ password	User: IMAP, SMB, FTP, LDAP Server: CA certificate	User: IMAP, SMB, FTP, LDAP Server: CA certificate	username/ password	User: user certificate from CA Group of users: short-lived VOMS Proxy Certificate Server: X.509 certificates	User: username/password, LDAP, GSI, Kerberos out of the box: Pluggable Authentication Modules (PAM), SSL
File version control	No	Yes	Yes	No	No	No
Federation	No	-Certified SSL is needed for ownCloud servers +Sharing is encrypted via SSL/TLS	-Certified SSL is needed for ownCloud servers +Sharing is encrypted via SSL/TLS	Yes	A unified virtual file system is provided	co-ordinate & share in zones: catalogue provider hosts, negotiation key, zone_key, zonename

	LOBCDER	ownCloud	nextCloud	Onedata	dCache	iRODS
Extend-ability	Limited	Limited	Limited (Global scale architecture is provided)	Limited	Yes	Yes
Built-in redundancy	No	No	No	No	Yes	Yes
Metadata management	No	No	Metadata app is provided	No	No	Yes
Programming language	Java	PHP	PHP	Erlang/C	Java	C++
Community	1820 commits, 5 branches, 2 contributors	37000 commits, 278 branches, 450 contributors in 8 years	43000 commit, 78 branches, and 530 contributors in 2 years	7500 commits, 190 branches, 19 contributors	9000 commits, 61 branches, 35 contributors	6200 commits, 4 BRANCHES, 36 CONTRIBUTORS

2.1.2 PROCESS Data Federated Management Systems (PROCESS-VFS)

When studying the exascale data storage landscape, one can see that a significant research effort is put into designing new hardware infrastructures at the data centre level to optimize the HPC I/O stack[31,32,33,34,35,36]. The DOE report[37] on the system requirements expected archival storage describes an emerging trend to embed more data management features directly into HPSS and thus acting as the storage level itself. Simulation tools are being developed to support the design of exascale systems and better understand the features and design constraints[38]. However, it is clear from many analytical studies[39,40], which try to estimate current and future expenses in terms of energy consumption, predict that one single exascale datacentre is not realistic and thus the current effort of optimizing the HPC I/O stack has to be complemented with an effort to create a data management layer which can scale across data centres.

To be able to claim that a data infrastructure is exascale enabled, it should be able to easily scale across geographically and institutionally distributed datacentres. This implies that the targeted data infrastructure is able to operate as a multi-system, on multiple data centres, multiple providers, multiple domains/types (MS-MDC-MP-MD). Current approaches try to propose a data federation layer, which directly interacts with the backend storage, and for each backend they develop a specific driver in a plugin-like architectural style. They have been designed to operate in a specific setting that could be divided in 4 categories:

[31] J. Bent et al., "Jitter-free co-processing on a prototype exascale storage stack," 012 IEEE 28th Symposium on Mass Storage Systems and Technologies (MSST), San Diego, CA, 2012, pp. 1-5. doi: 10.1109/MSST.2012.6232382

[32] Christos Filippidis, Parallel Storage Systems for Large Scale Machines, http://sc16.supercomputing.org/sc-archive/doctoral_showcase/doc_files/drs104s2-file2.pdf

[33] DAOS and Friends: A Proposal for an Exascale Storage System http://pages.cs.wisc.edu/~johnbent/Pubs/lofstead_sc16.pdf

[34] Exascale storage gets a GPU boost DFAF
https://www.nextplatform.com/2018/02/12/exascale-storage-gets-gpu-boost/

[35] An exascale Timeline For Storage and I/O system
https://www.nextplatform.com/2017/08/16/exascale-timeline-storage-io-systems/

[36] infinite Memory Engine: The exascale –ear storage architecture https://www.hpcwire.com/2017/08/21/infinite-memory-engine-exascale-era-storage-architecture/

[37] Hick, J., Watson, D., Cook, D., Minton, J., Newman, H., Preston, T., … White, V. (2010). HPSS in the Extreme Scale Era: Report to DOE Office of Science on HPSS in 2018-2022. - Report Number: LBNL-3877E

[38] Jason Cope et al. CODES: Enabling Co-Design of Multi-Layer Exascale Storage Architectures, https://pdfs.semanticscholar.org/159d/bd0a8c18e2df895b131e33499e2d529210e0.pdf

[39] J. Mair, Z. Huang, D. Eyers and Y. Chen, "Quantifying the Energy Efficiency Challenges of Achieving Exascale Computing," 2015 15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, Shenzhen, 2015, pp. 943-950. doi: 10.1109/CCGrid.2015.130

[40] K. W. Cameron, "Energy efficiency in the wild: Why datacenters fear power management", Computer, vol. 47, no. 11, pp. 89-92, 2014.

1. One system, one data centre, one provider, one domain/type like LHC
2. One system, multiple data centres, multiple providers, one domain/type like WLGC, Astron, Globus
3. One system, multiple data centres, multiple providers, one domain/type like cloud storage providers
4. One system, multiple data centres, multiple providers, multiple domain/type like EUDAT

On the contrary the cloud approach has proven that scalability can only be achieved if we introduce a virtualization layer, which abstracts completely the details of the "hardware" infrastructure. New approaches based on Named Data Networking try to reduce the overhead in data transmission and will likely improve communication within and across data centres[41,42,43,44] and finding scattered across data centres could be completely agnostic of its location. Following the cloud virtualization approach, we propose a data micro-infrastructure which is based on two basic widely accepted concepts IaaS and the fact that most secondary storage are accessed for read and write through a simple mount action regardless of the operating system or the storage type.

2.1.3 Micro-infrastructures for distributed data management and compute

As the variety of applications and collaborations between researchers increases, so does their dependencies and requirements. Every group may have unique requirements and dependencies for their applications. These different environments might require different data management, distribution and processing. Clearly, a one size fits all distributed system that tries to encompass all these different requirements beforehand will not perform well. Such an approach entails that the system needs to continuously resolve new dependencies and requirements while also maintaining scalability. Furthermore, any smart data management is oftentimes very application or domain specific due to storage means (DB, files, etc), different data access patterns, algorithm complexity, provenance, value, etc. This implies that the common data storage denominator between application is, most often, raw block storage and a monolith system would need to handle all the different applications.

A different approach that can handle the multitude of different data models, applications, distribution and management is through virtualization, by encompassing all these requirements in a data micro-infrastructure with specific nodes for handling the different aspects e.g. a nextCloud node for sharing data within the group, and HDFS file system for computing, GridFTP for accessing remote files etc. The whole infrastructure then becomes an ensemble of micro-infrastructures each with its own full stack encapsulated in a virtual infrastructure.

Figure 6 illustrates the notion of a micro-infrastructure. Site providers provide raw resources through virtualization middleware such as OpenStack. They also provide raw storage that is accessible through the virtual machines. Through templating, micro-infrastructures can be booted up that will satisfy the groups' requirements for data processing. Cross provider data, process distribution and management are handled from within the micro-infrastructure. Cross group collaboration is also easily manageable e.g. a group could give access to another group through their ownCloud node inside the micro-infrastructure. Scalability is improved since one data management system will have difficulty managing exascale data, but many micro-infrastructures can better manage their own pool of data which is, most often, a few orders of magnitude less than an exabyte.

[41] Susmit Shannigrahi, Chengyu Fan, Christos Papadopoulos, Named Data Networking Strategies for Improving Large Scientific Data Transfers In Proceedings of the IEEE International Conference on Communications, May 2018

[42] Schi Chen et al, NDSS: A Named Data Storage System http://www.mit.edu/~caoj/pub/doc/jcao_c_ndss.pdf

[43] S. Zhu, M. Yuan and K. Lei, "Ndynamo: An ndn DHT-based distributed storage system over named data networking," 2016 5th International Conference on Computer Science and Network Technology (ICCSNT), Changchun, 2016, pp. 148-152. doi: 10.1109/ICCSNT.2016.8070137

[44] Y. Rao, D. Gao, H. Zhang and C. H. Foh, "Mobility Support for the User in NDN-Based Cloud Storage Service," 2015 IEEE Globecom Workshops (GC Wkshps), San Diego, CA, 2015, pp. 1-6. doi: 10.1109/GLOCOMW.2015.7414159

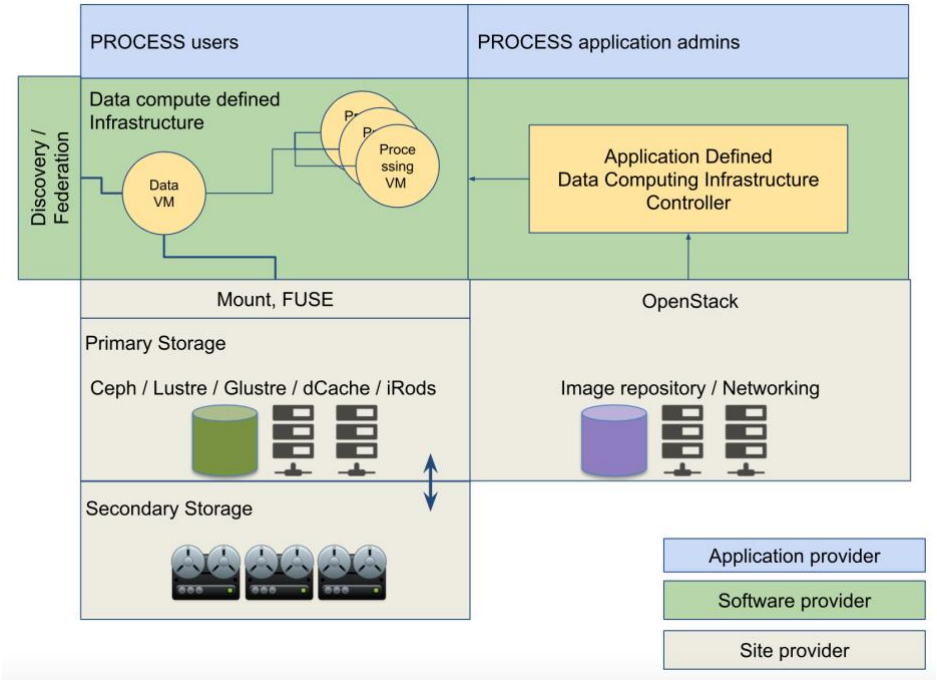

Figure 6: PROCESS proposed micro data architecture

2.2 Access to Complex Use Case-specific Data Sets

In this section we'll review the needs of the use cases regarding access to specific data types and storage technologies. Based on these, we will describe in detail the design and integration of distributed storage components connecting to the data infrastructure of PROCESS and allowing the use cases to obtain the access to currently not supported compound data types and storage systems.

2.2.1 Description of the use cases' needs regarding access to their data - database queries, extraction of data from compound formats

The use cases of PROCESS require access to several specialized data sets, stored either in files or relational databases, in various formats. Based on D4.1 we have compiled a list of these sets in Table 3.

Table 3: List of specific data sets used by PROCESS use case scenarios

Dataset Name	Estimated size	Description	Format
Camelyon17	>3TB	1000 WSI, 100 patients	BIGTIFF
Camelyon16	>1TB	400 WSI	BIGTIFF
TUPAC16	>3TB	WSI	BIGTIFF
TCGA	>3TB	WSI	BIGTIFF
PubMed Central	~ 5 million images	Low resolution	Multiple formats
SKIPOGH	>30TB	WSI	BIGTIFF

Dataset Name	Estimated size	Description	Format
LOFAR	16TB	Radio astronomical observations	CASA MeasurementSet[45]
GAR		Hazard models	AME[46]
PNR	>1TB	Airline booking	RDBMS
EMD	>1TB	Ancillaries to booking	RDBMS
COPERNICUS	>100PB	Earth observations	SAFE (SENTINEL-specific)[47]

Table 3 contains references to specific data formats, which need to be accessed. In an exa-scale environment, it may often be impossible, or very expensive, to transfer a complete data set to the place of its processing, only to extract from it a small part which is actually needed. Therefore, as part of the pre-processing of PROCESS use case data, we will provide tools for accessing the data formats and extracting from them the relevant parts in or near the place where they are stored. Only the extracted, relevant part will be then transferred over the network to the place of processing. Following we list the data formats we have been able to identify, their description and scenarios of access to them.

BIGTIFF

BIGTIFF[48] is an extension of the well-known TIFF (Tagget Image File Format) image format. The format has been modified so that it can encode files larger than 4GB (2^{32}, the maximum given by TIFF encoding offsets as 32-bit integers). This has been achieved by new number encoding methods and a new BIGTIFF-capable version of the libtiff library, which shields user from having to develop the necessary decoding methods.

Apart from providing access to large images, BIGTIFF supports a “pyramid TIFF” paradigm. The image is encoded in several different resolutions, all stored as a “pyramid” in the same (BIG)TIFF file.

BIGTIFF file format is used in the data sets of the UC #1: Exascale learning on medical image data[49].

The role of data pre-processing in PROCESS will be to

- extract the selected resolution of the pyramid TIFF file
- extract the selected rectangle of the selected resolution layer of a TIFF file

CASA MeasurementSet

The MeasurementSet is a relational database-like file format used to hold radio astronomical data to be calibrated following the MeasurementEquation approach[50]. It stores data split into several tables, with relations between them, in order to reduce the redundancy of the data. The data set is not stored in a single file, but the tables are organized in a directory structure.

[45] https://casa.nrao.edu/casadocs-devel/stable/reference-material/measurement-set

[46] https://pos.sissa.it/239/030/pdf

[47] https://sentinel.esa.int/ web/sentinel/user-guides/sentinel-1-sar/data-formats/safe-specification

[48] http://bigtiff.org/#Overview

[49] PROCESS deliverable D4.1: Initial state of the art and requirement analysis, initial PROCESS architecture.

[50] https://casa.nrao.edu/casadocs-devel/stable/reference-material/measurement-set

Transport of the data is usually facilitated by packing (tar-ing) the whole directory structure into a single file, moving the file over the network, and unpacking it at the destination.

The MeasurementSet format is used in PROCESS UC#2: Square Kilometre Array/LOFAR (PROCESS D4.1).

The role of data pre-processing in PROCESS will be to

- extract a subset of the MeasurementSet data, based on a SQL-like query
- pack the extracted subset of data into a single file, deliver it to the processing destination, unpack it there

AME

The AME file format is a specific data format used to represent hazard results in the CAPRA software package[51]. The CAPRA platform specializes in probabilistic risk assessment and is provided with an open source licence. It contains several tools and modules:

- AMExploit – tool for accessing AME files
- GridExploit – tool for accessing gridded data
- SHPConverter – tool for converting SHP (Shape) files[52]
- DATEditor – tool for accessing vulnerability representation in a custom .dat format
- CRISIS 2007 - Program for calculating seismic and tsunami hazard
- ERN-Hurricane - Program for calculating hurricane hazard
- ERN-NH Rainfall - Program for calculating non-hurricane rainfall hazard
- ERN-Landslides - Program for calculating landslide hazard
- ERN-Flood - Program for calculating flood hazard
- VHAST - Program for calculating volcanic hazard
- Data gathering tool (web) - Google Earth-based data gathering tool
- ERN-Vulnerability - Desktop Program for calculating and editing vulnerability functions
- CAPRA-GIS - CAPRA risk calculation and visualization system
- FileCAT - CAPRA data classification and previsualization system

The PROCESS UC#3: Supporting innovation based on global disaster risk data relies heavily on the CAPRA platform. Therefore, it will be better not to develop new versions of tools for access to the AME data sets, but to allow the user to deploy parts the CAPRA framework remotely, pre-process the data in-situ, extract the results of the pre-processing, and transfer them for further processing.

In the first phase of the project, the role of data pre-processing in PROCESS with respect to UC#3 will be to

- deploy selected parts of the CAPRA platform to the remote data storage
- control the deployed parts via command-line-style commands
- transfer the outputs of the remotely deployed tools over the network

In this phase, we will deploy the AMExploit, GridExploit, SHPConverter tools.

PNR and EMD, stored in a RDBMS

The reservation and ancillaries' records used in airline bookings in UC#4: pricing for airline revenue management are currently described as the structure of a relational DBMS. Since these data contain sensitive information, we may not be able to access them remotely as is usual with RDBMS. Therefore, we will develop a PNR and EMD-specific data extraction

[51] http://siteresources.worldbank.org/ INTLACREGTOPURBDEV/Images/840342-1264721236030/ CAPRA_Salgado_M.pdf
[52] https://en.wikipedia.org/wiki/Shapefile

tools, which will allow only certain, pre-defined operations to be done on the PNR and EMD records. These extraction tools will be deployed to the place where the data is stored (or near it), the required parts of the data will be extracted and transferred over the network for further processing. Since the data also contains personal information, which is not necessary for analysis, we will also provide anonymization tools (generic, not PNR and EMD-specific).

The role of data pre-processing in PROCESS regarding UC#4 will be to:

- define a set of allowed operations on PNR and EMD records
- develop a remotely deployed tool capable to perform these operations
- develop a configurable data anonymization tool

SAFE

The “Standard Archive Format for Europe” is an ESA-developed format for Earth observation data storage[53]. The version used in PROCESS UC#5: Agricultural analysis based on Copernicus data is so-called “SENTINEL-SAFE” format, specific to the operation of the SENTINEL range of EO satellites[54]. The SENTINEL-SAFE format contains[55]:

- a 'manifest.safe' file which holds the general product information in XML
- subfolders for measurement datasets containing image data in various binary formats
- a preview folder containing 'quicklooks' in PNG format, Google Earth overlays in KML format and HTML preview files
- an annotation folder containing the product metadata in XML as well as calibration data
- a support folder containing the XML schemes describing the product XML.

From this data set, UC#5 needs to extract the general product information, the metadata, the schema, and – based on these – parts of the binary image data. Therefore, the role of PROCESS pre-processing services in UC#5 will be to:

- provide service for extraction of SENTINEL-SAFE manifest information
- provide service for extraction of SENTINEL-SAFE metadata
- provide service for extraction of SENTINEL-SAFE schema
- provide service for extraction of selected parts of the binary image data

2.2.2 Description of Use Cases' Data Pre-processing Requirements

The new data pre-processing services required by the use cases are designed from three sources of requirements:

- the workflows of the use cases, as described in D4.1[56]
- the non-functional requirements of use-cases, also as described in D4.1
- the requirements to access and process use-case-specific data formats, described in the previous section.

2.2.3 Summary of UC requirements

UC#1:

- make local copies of data in each processing centre
- provide streaming server able to handle file locations and data transfer
- handling of bottlenecks during I/O and data decoding

[53] http://earth.esa.int/SAFE/

[54] https://www.esa.int/Our_Activities/Observing_the_Earth/Copernicus/Sentinel-2

[55] https://sentinel.esa.int/ web/sentinel/user-guides/sentinel-1-sar/data-formats/safe-specification

[56] PROCESS deliverable D4.1: Initial state of the art and requirement analysis, initial PROCESS architecture.

- extraction of thousands of hi-res patches from Whole Slide Images (WSIs)
    - creation of normal tissue and tumour masks
    - random sampling of hi-res patches
    - creation of a database of patches
    - learning on the extracted data, both locally and remotely
- possibility of moving the training to hospitals
- possibility of moving the training to the data centres which hold the WSIs
- need for an efficient data storage system considering data formats
- BIGTIFF, XML, CSV, TXT, JPEG, MPEG
- extract the selected resolution of the pyramid TIFF file
- extract the selected rectangle of the selected resolution layer of a TIFF file

UC#2:

- support for current data environment
    - uberftp client[57]
    - globus-url-copy client[58] (for dCache access)
    - voms-client[59] (for LOFAR VO support)
    - CernVM File System[60] support
    - Access to CASA MeasurementSet data
- An efficient data management system that is capable of efficiently transporting the MeasurementSets from the archive locations in Amsterdam, Juelich and Poznan to the processing locations
- Extract a subset of the MeasurementSet data, based on a SQL-like query
- Compress the extracted subset of data, deliver it to the processing destination, uncompress it there

UC#3:

- replace the current data transfer model (usually shipping of physical HDDs between locations)
- support integration with heterogeneous software solutions as the workflows may differ based on the place of data origin
- support for CAPRA-GIS toolset as the most used platform
    - AMExploit – tool for accessing AME files
    - GridExploit – tool for accessing gridded data
    - SHPConverter – tool for converting SHP (Shape) files61
    - DATEditor – tool for accessing vulnerability representation in a custom .dat format
- Support for AME files manipulation
- deploy selected parts of the CAPRA platform to the remote data storage
    - AMExploit
    - GridExploit
    - SHPConverter
- control the deployed parts via command-line-style commands

[57] http://www.lofar.org/wiki/doku.php?id=public:grid_srm_software_installation
[58] http://www.lofar.org/wiki/doku.php?id=public:grid_srm_software_installation
[59] http://www.lofar.org/wiki/doku.php?id=public:grid_srm_software_installation
[60] https://cernvm.cern.ch/portal/filesystem
[61] https://en.wikipedia.org/wiki/Shapefile

- transfer the outputs of the remotely deployed tools over the network

UC#4:

- Access to RDBMS-stored PNR and EMD records
- Handle large amount of data
- Handle data from different source
- Handle high volume of requests per day
- Establish a consolidated data structure on which further statistical processing can be performed
- Process ongoing data streams to keep the consolidated data structure up-to-date
- Protect personal information of passengers
- define a set of allowed operations on PNR and EMD records
- develop a remotely deployed tool capable to perform these operations
- develop a configurable data anonymization tool

UC#5:

- Access data from the Copernicus archive
- Pre-process data for the PROMET software
- provide service for extraction of SENTINEL-SAFE manifest information
- provide service for extraction of SENTINEL-SAFE metadata
- provide service for extraction of SENTINEL-SAFE schema
- provide service for extraction of selected parts of the binary image data

2.2.4 Data Pre-processing Services to be Developed and Deployed in the Initial Stage of PROCESS

Based on the above-stated UC requirements for data processing, we have selected the following data pre-processing services for design, development and deployment in the initial stage of PROCESS implementation (up to MS3 in M24, with a basic set available at MS2 – Integration, M12 of the project).

extract the selected resolution of the pyramid TIFF file

Operation: extract the selected resolution from a multi-resolution BIGTIFF file
Inputs:
- BIGTIFF file
- selected layer (resolution)

Outputs:
- one-layer BIGTIFF file

extract the selected rectangle of a TIFF file

Operation: extract the selected rectangle from an input BIGTIFF file into a new BIGTIFF file
Inputs:
- BIGTIFF file
- coordinates of the upper-left corner of the extracted portion
- coordinates of the lower-right corner of the extracted portion

Outputs:
- BIGTIFF file

extract part of JPEG file

Operation: extract the selected rectangle from an input JPEG file into a new JPEG file
Inputs:
- JPEG file
- coordinates of the upper-left corner of the extracted portion
- coordinates of the lower-right corner of the extracted portion

Outputs:
- JPEG file

globus-url-copy as a service

Operation: perform a globus-url-copy operation on one file, deliver the file
Inputs:
- Globus credentials
- URL of a file

Outputs:
- A file, based on the given URL

uberftp as a service

Operation: perform FTP GET operation on one file, deliver the file
Inputs:
- UberFTP credentials (name, password)
- URL of a file

Outputs:
- A file, based on the given URL

Extract a subset of the MeasurementSet data, based on a SQL-like query

Operation: based on a SQL-like query, extract data from a MeasurementSet data structure
Inputs:
- Identifier of a MeasurementSet data set (directory name, local ID…)
- Query

Outputs:
- Data stream

Compress data

Operation: apply a statistical compression algorithm (ZIP, GZIP, BZ2…) to a data package
Inputs:
- Data stream

Outputs:
- Compressed data stream

Decompress data

Operation: apply a decompression algorithm to a data package
Inputs:
- Compressed data stream

Outputs:
- Data stream

amexploit as a service

Operation: remotely provide command-line functionality of amexploit

Inputs:
- amexploit command line

Outputs:
- Data stream

gridexploit as a service

Operation: remotely provide command-line functionality of gridexploit
Inputs:
- gridexploit command line

Outputs:
- Data stream

shpconverter as a service

Operation: remotely provide command-line functionality of shpconverter
Inputs:
- shpconverter command line

Outputs:
- Data stream

perform a pre-selected parameterized SQL query

Operation: similar to a stored procedure, perform an SQL query with defined variables
Inputs:
- ID of selected SQL query
- variable number of parameters

Outputs:
- Data stream

anonymization of records

Operation: replace selected parts of tuples in a data stream with anonymous IDs
Inputs:
- data stream of tuples
- identification of items in the tuples to replace with anonymous IDs

Outputs:
- data stream of tuples

extraction of SENTINEL-SAFE manifest information

Operation: extract the manifest from a SENTINEL-SAFE data set
Inputs:
- ID of a SENTINEL-SAFE data set (file name)

Outputs:
- Data stream containing that SENTINEL-SAFE data set's manifest file

extraction of SENTINEL-SAFE metadata

Operation: extract the metadata from a SENTINEL-SAFE data set
Inputs:
- ID of a SENTINEL-SAFE data set (file name)

Outputs:
- Data stream containing that SENTINEL-SAFE data set's metadata

extraction of SENTINEL-SAFE schema

Operation: extract the schema from a SENTINEL-SAFE data set
Inputs:
- ID of a SENTINEL-SAFE data set (file name)

Outputs:
- Data stream containing that SENTINEL-SAFE data set's schema

extraction of selected parts of the binary image data

Operation: extract the selected part from a SENTINEL-SAFE data set
Inputs:
- ID of a SENTINEL-SAFE data set (file name)
- ID of the image to extract
- Optional coordinates of the upper-left and lower-right corners of a part of the image to extract

Outputs:
- Data stream containing extracted image data

2.2.5 Integration of PROCESS data infrastructure with data processing (DISPEL)

Requirements

The use cases of PROCESS all require access to large amounts of data, pre-processing these data, extracting portions of it and performing operations on these sub-sets. The process of transfer of data from the place where it is stored to the place where it is to be processed gets increasingly difficult and expensive with the data size increasing. For exa-scale data sets, transfer over the network is usually impossible. Therefore, the data management subsystem of PROCESS needs to solve the problem of accessing data sets without having to transfer over the network unnecessary volumes of data, which will be discarded by further pre-processing in the use case workflow.

The details of the data stream pre-processing platform which will be used in PROCESS work package 7 are explained in PROCESS D4.1. Here we will just summarize:

- The ADMIRE platform is a mature (TRL > 6) software suite.
- The parts reused and extended in PROCESS are called the *DISPEL Gate.*
- It creates and manages data processes which handle streams of data.
- It is highly distributed, scalable and extensible.
- It is based on the SOA paradigm.
- The data processes are described by a high-level data process description language called *DISPEL.*
- It supports separation of concerns, allowing data processes to be designed (described in DISPEL) by domain and data experts, and later used by application users who do not need to understand the details of the process.

The goal of integration of the DISPEL Gate from WP7 into the data management suite of WP5 is

- to allow PROCESS use cases to access data created by DISPEL-driven processes in the same manner as any other data available in PROCESS,
- to insulate use cases and their users from the complexities of data streaming and data process management, and

- to provide use cases with the option to pre-process their input data sets in (or near) the place where they are stored, reducing the amount of data transferred over the network.

Design

The integration of the Dispel Gate from WP7 with the main data management component from WP5 – LOBCDER – will be handled by a custom Virtual File System (VFS) driver for LOBCDER. LOBCDER architecture supports extension of its file access capabilities by custom VFS drivers, which handle data retrieval.
The process of accessing on-demand, DISPEL Gate-produced data will consist of these steps:

1. a user (use case program) accesses the LOBCDER WebDAV interface, providing a URL of a file it wishes to access;
2. a filter within LOBCDER WebDAV interface will, based on the URL structure, direct the request to the DISPEL VFS driver in LOBCDER's data access layer;
3. the DISPEL VFS driver will parse the URL and based on its structure will
    a. select the correct DISPEL data process document,
    b. insert parameters into the selected DISPEL document, and
    c. enact the parameterized DISPEL document on a pre-configured DISPEL Gate;
4. the enactment, creation of a data process and production of a stream of output data are part of WP7; and
5. the result of the DISPEL data process will be fed back to the client who has initiated the original request.

This way, user will have access to the distributed data processing capabilities of the ADMIRE platform – a.k.a. DISPEL Gate in PROCESS – without having to implement an additional client interface. At the same time, data processes will be pre-created by data and domain experts.

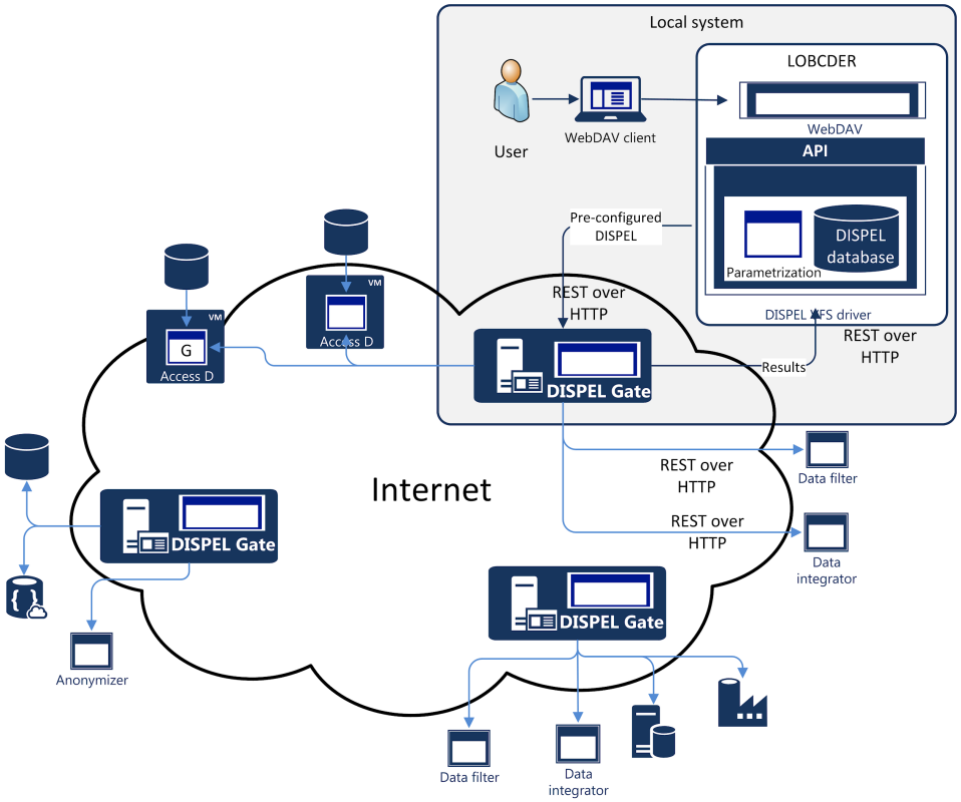

Figure 7: The architecture and method of integration of WP7 data processing components (DISPEL Gate) and WP5 data management components (LOBCDER)

An example of the URL structure for DISPEL enactment is:
https://lobcder.process-project.eu/dispel/meteo/grid3855/layer015/time20180812060000
The components are interpreted as:

- **dispel** – filtered by LOBCDER WebDAV interface and fed to DISPEL VFS driver
- **meteo** – selector of a DISPEL document from the DISPEL database of DISPEL VFS driver (see Figure 7)
- **grid3855** – a parameter (grid selection in a map) for the DISPEL document parameterization module (see Figure 7)
- **layer05** – another parameter (layer selection in a map) for the DISPEL document parameterization module
- **time20180812060000** – another parameter (time selection) for the DISPEL document parameterization module

2.3 Meta-Data

In this section we're going to describe details regarding the Metadata component that is going to supplement the PROCESS-VFS - responsible for storage of BLOB data - by allowing management of the tabular data, including basic operations such as saving, retrieving and searching.

Motivation behind the Metadata component (DataNet)

Computations in the PROCESS project are going to be performed on the blocks of raw data specific for the given use case. However regular data-sets (such as medical images in the use case #1) may be accompanied by additional information (even after a full anonymization) such as imaging technology, condition during the imaging process. That information should be easily searchable as they may be required for proper setup of the computations or post-

processing of the results. Due to this reasons metadata should be separated from regular data (unless they're already stored in separate sets) and stored into specialized service such as DataNet component described here.

DataNet Requirements

We do not assume that the amount of meta-data would require DataNet service to utilize exascale system by itself, however it still must be designed in a way that would support computations of this scale. Due to this fact, and the need to use multiple sites at the same time in attempt to build an exascale system (as shown earlier in this Deliverable), the final version of DataNet would need to be highly scalable and geographically distributed.

DataNet Functionalities

The platform needs to support following functionalities:

- Web UI for metadata repository management
  - Creation
  - scaling
  - Monitoring
- REST interface for metadata management
  - metadata entry creation
  - metadata entry retrieval
  - querying metadata collection
    - Filtering
    - Sorting
    - Counting
- Metadata - plain JSON or HAL+json (relationships) representation format
- JSON schema validation or schema-less metadata storage
- Web-based metadata browser

The DataNet Architecture

To meet already mentioned requirements we're going to build this platform in 3 iterative steps during the course of the PROCESS project. This would allow to ship proper solution to use case providers quickly, and then work on improving scalability and redundancy of the system as required to meet further requirements of the prospective exascale solution.

General overview of the final version of the metadata component DataNet alongside related tooling is shown in Fig. 8.

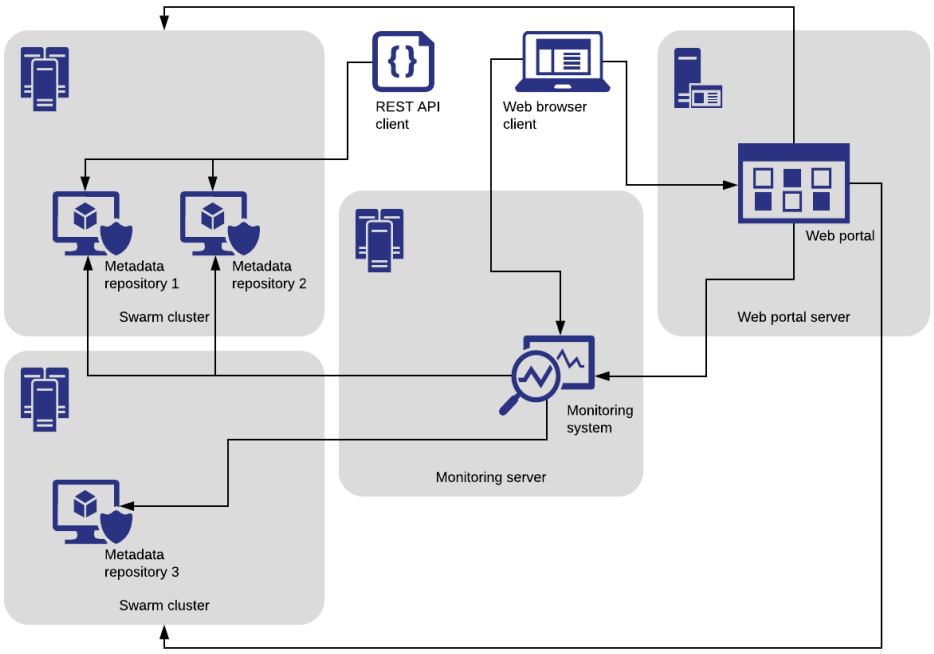

Figure 8: Final Architecture of the DataNet Component

The Metadata repository is going to be deployed using a RESTHeart solution providing REST-full API for the MongoDB storage.
DataNet platform development and integration steps:

- First prototype (project M12) - single repository (as shown in Fig. 9 deployed, integrated with VFS and available to the Use Case providers
- Second prototype - multiple repositories running on a single container cluster (e.g. Docker Swarm)
- Final version - multiple Swarm Clusters (different sites) running multiple repositories each (see previous slide)

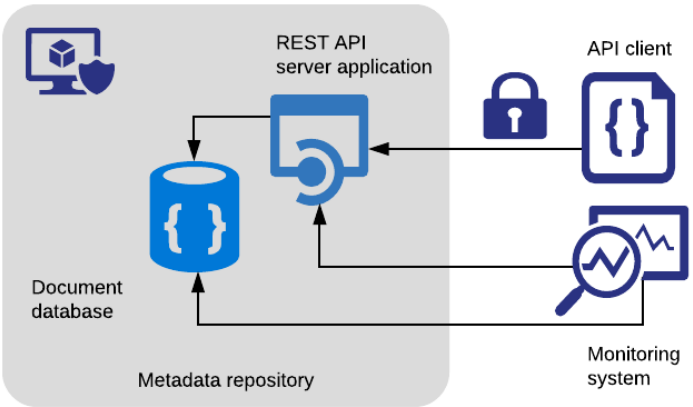

Figure 9: Single Metadata Repository (part of the DataNet System)

2.4 Interaction between the Data and Metadata components

Requirements

At least part of the use cases in the PROCESS project require both regular data as well as metadata describing them. For example, in the use case #1 we have both typical data sourced from the BIGTIFF or JPEG files, as well as textual data in form of the TXT or CSV files parametrizing them. This prompts the requirement to provide mechanism for metadata extraction as well as later access to it for operations such as querying.

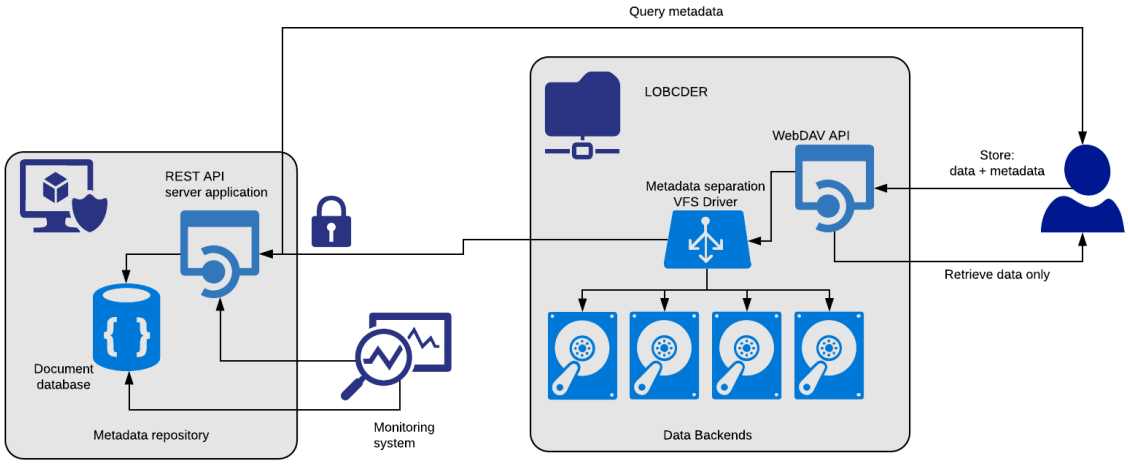

Figure 10: Interaction between Data and Metadata components

Architecture Overview of planned interaction between LOBCDER and DataNet is shown in Fig. 10.

As shown there we intend to provide the mechanism allowing:

- Automatic extraction of metadata from the uploaded files using the dedicated VFS Driver
- Storage of Data on appropriate backend and passing metadata to DataNet via the REST API to be stored in the document database
- Allow users querying the stored metadata directly via the REST API provided by the DataNet component

3 PROCESS Data infrastructure in the context of existing EU Research Infrastructure

Likely in the European computing and data management infrastructure a number of steps have been taken to enable an EU-wide collaboration which can use data centres across EU countries and operating under the PRACE or EGI umbrella.

3.1 PRACE HPC resources

Access to PRACE resources is described in details on PRACE website **Interactive Access to HPC Resources**[62]. In short, the PRACE resources are organized as Tiers (Tier-0 and Tier-1). Tier-0 is composed HPC resources distributed on 5 data centres located in Spain, France and Germany; one of these datacentres is associated to LMU, which is leading the PROCESS project. Tiers is composed of computing resources distributed over 10 data centres located in Italy, Finland, United Kingdom, Sweden, Hungary, Poland, Germany and The Netherlands.

Access to these HPC resources is granted as part of a PRACE allocation of Tier-0 resources and DECI calls. Three basic concepts are used to organize the access to PRACE HPC resources: *Execution Site* (resource offering computational resources), *Home Site* (taking care of the setup of the allocated project) and *Door Node* (PRACE service for DECI/Tier1 providing access to the PRACE resources via the public Internet network). Currently there are two PRACE sites acting as door nodes: CINECA and SURFSARA.

The PRACE interactive access service is based on two technologies: SSH (Secure Shell Access) and GSI SSH (Grid Security Infrastructure enabled SSH). Only some sites offer SSH, mainly PRACE Tier-0, GSI enabled SSH represent the default interactive service for DECI sites. Its single sign on feature makes it convenient in an inherently distributed infrastructure such as PRACE Tier-1. It is thus important that PROCESS solution relays on this technology as way to authenticate use across the PRACE datacentres.

3.2 EGI computing resources

Access to EGI service and computing resources are possible through the 3 access policies[63]: Policy-based, Wide access, and Market-driven. EGI provide a variety of computing and storage resources namely: HTC[64], Cloud compute[65], online storage[66], archive storage[67], and data transfer[68]. The EGI access service is based on Check-in service, which operates as a central hub to connect federated Identity Providers (IdPs) with EGI service providers[69]

3.3 EUDAT computing resources

EUDAT a pan-European data infrastructure which provides solutions for managing data for various research communities in Europe. The EUDAT service catalogue[70] covers: (1) Data Hosting, Registration & Management & Sharing (2) Data Discovery, (3) Data Access, Interface & Movement, (4) Identity and Authorization. In PROCESS we will develop solution which are very similar to EUDAT category (2) and (3) and thus it is important to consider interoperability issues with EUDAT services. EUDAT offers a federated cross-infrastructure authorisation and

[62] http://www.prace-ri.eu/interactive-access-to-hpc/
[63] https://www.egi.eu/access-policy/
[64] https://www.egi.eu/services/high-throughput-compute/
[65] https://www.egi.eu/services/cloud-compute/
[66] https://www.egi.eu/services/online-storage/
[67] https://www.egi.eu/services/archive-storage/
[68] https://www.egi.eu/services/data-transfer/
[69] https://www.egi.eu/services/check-in/
[70] https://www.eudat.eu/catalogue

authentication framework B2ACCESS supports several methods of authentication via the users' primary identity providers (OpenID, SAML, x.509).

4 Appendixes

4.1 Data storage, Delivery, and Sharing, platforms

In this section, we review the alternative data infrastructures for the PROCESS project.

4.1.1 OwnCloud/nextCloud

nextCloud is a new development line based on the ownCloud code. In 2016 Karlitschek forked ownCloud to create nextCloud. Up to nextCloud 11, the two systems presented similar features. In 2017, nextCloud announced[71] the “global Scale architecture” as part of nextCloud 12, which does seem to have a counterpart on the ownCloud system. In the following paragraphs we focus on the features which are common to the two systems and which are of interest of the PROCESS project.

Command line interface: ownCloud/nextCloud have a comprehensive command line interface *occ* that allows perform many common server operations. *occ* is a PHP script residing on ownCloud / it must be run as **HTTP user** to ensure that the correct permissions are maintained on your ownCloud files and directories such as installing and upgrading ownCloud, managing users and groups, encryption, passwords, LDAP setting, and more

Federation of storage: ownCloud/nextCloud federation concept allows multiple user to share file stored on different ownCloud servers. When the ownCloud Federation app is enabled it is possible to easily and securely link file shares between ownCloud servers. A list of create a list of trusted ownCloud servers can be created for Federation sharing. This allows your linked ownCloud servers to share user directories, and to autofill user names in share dialogs.

External storage: ownCloud/nextCloud server support mounting external storage services and devices as secondary ownCloud storage devices. ownCloud/nextCloud users may be allowed to mount their own external storage services. ownCloud support a variety of external storage (Amazon S3, Dropbox, gDrive, SFTP, SMB/CIFS, WebDAV). Different backends support different authentication mechanisms such as passwords, OAuth, or token-based, to name a few examples. External storage can be configured either through the GUI interface or through a set of *occ commands*.

Authenticate: ownCloud/nextCloud support two-factor authentication, when enable the ownCloud TOTP app generates a one-time authentication password TOTP which can be used to login again to the ownCloud server within 30 second. The two-factor authentication has been introduced since version 9.1. ownCloud offers user authentication with IMAP, SMB, and FTP, which allow to attach new external user storage to the ownCloud Server.

Security: ownCloud/nextCloud allow data encryption at rest and when transferring data to and from the ownCloud/nextCloud server. Encrypting data in movement is done using ‘TLS‘, a secure communication protocol for the Internet. When enabling ownCloud Encryption app, ownCloud will encrypt all data with a strong, randomly generated key, which is then protected with your log-in password[72].

Connect to ownCloud server: There are ownCloud clients for various desktops and mobile platform. The ownCloud client always automatically synchronized between the local storage with one or more ownCloud/nextCloud server. In 2018 ownCloud announced the implementation of Delta sync for the ownCloud Server and Desktop Client. This speed up the synchronization of uncompressed files, instead of the complete file, the sync client only uploads or downloads the corresponding modified parts when files are changed. The main implementation was the integration of the ZSync algorithm into the ownCloud server and the

[71] https://nextcloud.com/blog/nextcloud-announces-global-scale-architecture-as-part-of-nextcloud-12/

[72] https://doc.owncloud.org/server/8.2/user_manual/files/encrypting_files.html

desktop client. The speed up of the synchronization support many file formats for Disc images, audio, images, video, VM images, tex file and tar.

Scaling across multiple machines: since nextCloud 12[73], a global scale architecture was introduced to overcome the limitation on the number of users of a nextCloud installation, Storage cost, and global distribution. With this approach, nextCloud become more reliable and support fault tolerance (single, failover, Cluster). nextCloud Global Scale works by effectively removing the need for shared components in the existing architecture like the load balancers, hosting centre uplink, database, storage, and cache. It uses multiple independent application servers, called nodes, each running on standard, inexpensive commodity hardware. Storage, database, and cache are running local on the application servers and no longer have to be kept in sync. Nodes can be located in different data centres and be as small or large as any current nextCloud instance. A sensible scale would be at least 2 machines, providing redundancy in case of hardware failure. The Global Site Selector (GSS) acts as a central instance that is accessed by the user during the first login, accessing it via the Web, WebDAV or REST[74].

4.1.2 Onedata

Onedata offers a global (unified) data access solution for scientific applications. It abstracts both the location and the underlying storage of the data and enables various types of sharing data team-sharing, cross-community, and instant or ad-hoc[75]. Onedata has three basic concepts: (1) **Spaces** - distributed virtual volumes which can belong to multiple users, multiple user groups, and support by multiple providers (2) **Providers** - entities which provides actual storage (3) **Zones** - federations of providers which enable creation of closed or interconnected communities. Onedata provides advanced functionality including replica and transfer management for high performance scientific applications[76]. Files are divided into equal size blocks, which can be independently replicated between storage resources. There are different ways of replication strategy (manual, automatic, policy based) Onedata support high throughput transfers using a Distributed Priority Queue for cluster-to-cluster transfers. Published results show a 55 Gbit/s on a single node 5 parallel stream.

Command line interface: Onedata provides "**oneclient**" a command-line based client based on Fuse that is able to mount a remote Onedata space in a local file system tree[77]. oneclient supports major Linux platforms and macOS (Sierra and higher). The authentication to the remote Onedata server uses access token generated by through a web interface of the remote Onedata server. oneclient is part of the official Docker image.

Federation of storage: Files stored in Onedata server are organized into *Spaces*. Spaces are Onedata concepts, which abstracts storage volume distributed across various providers. Spaces are assigned a quota by the providers. Onedata spaces are organized into zones, forming a federation of providers, which enable to share files across geographically data centre. Onedata zones are enabled through the **Onezone** service, which is responsible for **authentication** and **authorization** of users. Sharing data is possible within and across Onedata zones.

Authenticate: Onedata offers a pluggable method of authentication per zone, it exposes multiple levels of access control, ACL on files and directories, group management, token-based authentication, and X.509. Only Onezone is needed to manage multiple zones distributed over different storage providers. Onezone is a sort of gateway for users as it manages the authentication and authorization across data. Onezone supports Open ID services - GitHub, Facebook, Google, Dropbox & INDIGO IAM. Onezone also generates space

[73] https://www.youtube.com/watch?v=1kGF4Pa5kdg

[74] https://nextcloud.com/blog/nextcloud-announces-global-scale-architecture-as-part-of-nextcloud-12/

[75] The Onedata platform - https://www.youtube.com/watch?v=wLGZiP0LujI

[76] https://onedata.org/docs/doc/using_onedata/replication_management.html

[77] https://onedata.org/#/home/documentation/doc/using_onedata/oneclient.html

support tokens, monitors availability of storage providers, sees the geographical distribution of storage providers, and chooses storage provider for spaces.

External storage: Onedata provider[78] installs ***Oneprovider* service** and attaches storage and registering it in a particular **Onezone service** through the Onedata web interface. Oneprovider implements drivers for storages such as NFS, Lustre, Ceph, Openstack SWIFT and S3. Onedata is in principle easy to integrate with external tools through a rich set of API interfaces (POSIX, CDMI, FTP/SFTP, WebDAV)

4.1.3 dCache

dCache provides a single virtual file system for geographically distributed heterogeneous servers. This is specifically provided for scientific purposes that need to store and retrieve huge amount of data with different data transfer protocols. The main features provided in dCache are data exchange backend storages, space management, data replication, tape support, fault recovery in case of node failure, and more.

dCache **namespace**, called **Chimera**, maps files to unique identifiers. dCache works based on the **cell** and **domain** concepts. Cell is the smallest block in dCache that is a set of threads and has a specific task to do. In general, a cell may interact with other cells to perform its task. Conceptually, all features provided by dCache are executed by cells.

dCache includes a set of domains that are Java Virtual Machine (JVM) containers. Each domain is host of multiple cells and each node can also run multiple domains. For scalability, a domain can be started, stopped, or migrated. This may happen to 1) solve the problem of the overloaded nodes; 2) provide domains for new nodes; or 3) provide load balancing. Cells may also be migrated to other domains. According to their tasks, cells are grouped into different types: **pool** and **door** cells.

Door cells are responsible for communication between end-clients and dCache instance while pool cells are responsible to access and manage stored file. There are different door types such as GSICap door, GridFtp door, etc. In order to connect the client to the data storage, a door port needs to be opened[79].

Security: dCache supports certificate-based authentication through the Grid Security Infrastructure used in GSI-FTP, GSIdCap transfer protocols and the SRM management protocol. Certificate authentication is also available for HTTP and WebDAV. dCache uses the Location Manager to discover the network topology of the internal communication: to which domains this domain should connect. The domain contacts a specific host and queries the information using UDP port. The response describes how the domain should react: whether it should allow incoming connections and whether it should contact any other domains. Once the topology is understood, dCache domains connect to each other to build a network topology. Messages will flow over this topology, enabling the distributed system to function correctly[80].

Authenticate: In order to provide a secure data access, a specific interface called gPlazma2 is provided in dCache. The authorization can be done with x.509 certificates, username/password, or Kerberos authentication in general. Users want to access data through doors with one of these authorization techniques. Then, the users' requests are sent to gPlazma and it allows data access to the users. There are different plugins provided to access by a variety of mentioned authorization techniques such as AUTH plug-ins, map plugins, account plugins, session plug-ins, and identity plug-ins. Take, for instance, user can access data with username/password with kpwd plug-in that maps users to UID and GID. There are also a variety of certificate to verify the validity such as CA certificates, user certificates, host certificate, and voms-proxy certificates (for a group of users).

[78] https://onedata.org/docs/doc/administering_onedata/provider_overview.html

[79] https://www.dcache.org/manuals/Book-3.2/index-fhs.shtml

[80] https://www.dcache.org/manuals/Book-3.2/Book-fhs.shtml#in-securing

To provide storage authorization in general, there is a two-step process: 1) obtaining username from user's DN and role; 2) mapping of username to UID and GID, which is called storage-authzdb.

Replication for resiliency: dCache provides a service that controls the number of replicas of a file on the pools. This resilience service creates and manages multiple permanent copies of a file. Hence, if a higher rate of security and/or availability is required, the resilience feature of dCache can be used. When a file is transferred into the dCache, its replica is copied into one of the pools. Since this is the only replica and normally the required range is higher, this file will be replicated to other pools. When some pool goes down the replica count for the files in that pool may fall below the valid range and these files will be replicated. Replicas of the file with replica count below the valid range and which need replication are called deficient replicas.

Later on, some of the failed pools can come up and bring online more valid replicas. If there are too many replicas for some files, these extra replicas are called redundant replicas and they will be "reduced". Extra replicas will be deleted from pools.

Resilience Manager (RM) counts number of replicas for each file in the pools which can be used online and keeps number of replicas within the valid range (min, max). RM keeps information about pool state, list of the replicas (file ID, pool) and current copy/delete operations in persistent database (in upcoming dCache release, persistent database will not be needed). For each replica RM keeps list of pools where it can be found. For the pools, pool state is kept in DB. There is a table which keeps ongoing operations (replication, deletion) for replica[81].

4.1.4 iRODS

This is an open source data management software used by research organizations and government agencies worldwide. iRODS strives to serve as the glue that can tie together many existing storage technologies with a unified namespace, discovery mechanism, and policy engine[82]. Usually, when an organization decides they need to incorporate iRODS into their infrastructure, they already have a significant amount of data, usually in disparate physical systems. To manage data at scales of hundreds of petabytes, billions of files, and time periods of decades, iRODS implements four main functions: data visualization, data discovery, workflow automation, secure collaboration.

Data visualization iRODS provides a logical representation of files stored in physical storage locations. We call this logical view a virtual file system and the capabilities it provides, Data Virtualization. Data stored in iRODS is typically accessed through an iRODS client. iRODS clients present files as data objects organized into collections. For the most part, there is little difference between data objects and files, and between collections and subdirectories.

Each iRODS deployment—or Zone—is composed of an iRODS Metadata Catalogic (iCAT) database, a catalogue Provider, and optional catalogue Consumers. The iCAT is a relational database that holds all the information about your data, users, and zone that the iRODS servers need to facilitate the management and sharing of your data.

Data objects and collections are stored in storage resources in an *iRODS Zone*. Each Storage Resource has a name — the Resource's logical representation — and a hostname and path. The hostname is the network name of the device that serves the data, and the path is the local file system path or object storage bucket that holds the data.

In iRODS, the term Data Object refers to the logical representation of data that maps to one or more physical instances of the data at rest in storage resources, such as Amazon's S3. Data objects are organized into hierarchical Collections—the logical representations of physical containers, similar to directories or folders that are found in a file system. As with file system

[81] Millar, A. P., et al. "dCache, Sync-and-Share for Big Data." Journal of Physics: Conference Series. Vol. 664. No. 4. IOP Publishing, 2015.

[82] https://iRODS.org/roadmap/

directories and folders, iRODS denotes levels of hierarchy with slashes (/) in the pathname. The complete pathname of an iRODS data object includes the Zone (i.e., iRODS deployment) name and the full pathname within that zone, e.g., /tempZone/home/alice/sciproject/results.txt. [83].

Data discovery the metadata catalogue contains information about a Zone's Data Objects, Collections, Users, Storage Resources, as well as information about the Zone itself. The provided metadata in iRODS is used for data discovery and locating relevant data within large data sets. iRODS metadata can include whatever descriptors you choose to apply to your data. Metadata can also be applied to Collections, Users, Resources, and other

iRODS Zones. The entire iRODS catalogue for a Zone is contained in a relational database. The database for this purpose must be hosted in a PostgreSQL, MySQL, or Oracle database management system.

Workflow iRODS automates data workflows, with a rule engine that permits any action to be initiated by any trigger on any server or client in the Zone[84].

Secure collaboration iRODS enables secure collaboration, so users only need to log in to their home Zone to access data hosted on a remote Zone. With iRODS, organizations can share data, or federate, by simply adding a few bits of networking information to their iRODS configuration. Organizations are not required to coordinate the configuration of their respective iRODS zones. Each organization in a collaborative partnership retains autonomous control over its data collections, including maintaining security and data management policies distinct from fellow collaborators.

iRODS virtual file system iRODS contains a virtual file system which maps logical directory paths stored in the iCAT to actual physical storage (e.g., Ceph cluster) that contains the logical data objects[85]. A coordinating resource has built-in logic that defines how it determines, or coordinates, the flow of data to and from its children. Coordinating resources exist solely in the iCAT and exist virtually across all iRODS servers in a particular Zone. A storage resource has a Vault (physical) path and knows how to speak to a specific type of storage medium (disk, tape, etc.). The encapsulation of resources into a plugin architecture allows iRODS to have a consistent interface to all resources, whether they represent coordination or storage. This virtualization enables the coordinating resources to manage both the placement and the retrieval of Data Objects independent from the types of resources that are connected as children resources. When iRODS tries to retrieve data, each child resource will "vote", indicating whether it can provide the requested data. Coordinating resources will then decide which particular storage resource (e.g. physical location) the read should come from. The specific manner of this vote is specific to the logic of the coordinating resource. A coordinating resource may lean toward a particular vote based on the type of optimization it deems best. For instance, a coordinating resource could decide between child votes by opting for the child that will reduce the number of requests made against each storage resource within a particular time frame or opting for the child that reduces latency in expected data retrieval times. We expect a wide variety of useful optimizations to be developed by the community[86].

iRODS Clients and APIs Users access the data and the metadata in iRODS through iRODS clients. Clients communicate with iRODS through an API, and some clients use a stack of multiple APIs. iRODS clients may be customized for an organization's particular needs, and anyone is free to develop a new iRODS client, open source or otherwise[87].

[83] https://github.com/iRODS/iRODS_training/blob/master/beginner/iRODS_beginner_training_2018.pdf

[84] https://github.com/iRODS/iRODS

[85] https://github.com/iRODS/iRODS

[86] https://docs.iRODS.org/4.2.3/plugins/composable_resources/#virtualization

[87] https://iRODS.org/uploads/2016/06/technical-overview-2016-web.pdf

4.1.5 Rucio

Rucio is a system developed at CERN with the aim to scale in term of search (billions of files), transfer petabytes of data (largest installation is responsible for more than 350 Petabytes of data, stored in a billion files, and distributed over 120 data centres globally)[88]. Rucio is composed of a set of front end servers that interact with a Database back-end. There are two kinds of servers: the *authentication nodes* that get the user credentials and generate tokens and the *Rucio servers* that allow to create/list/modify files, rules, replicas, meta-data, subscriptions etc. *Rucio servers provides a* RESTful interface, on the other hand users interact with Rucio using Rucio client, which first contact the authentication servers to get an access token. Rucio servers runs a set of lightweight agents for specific tasks like replication rule evaluations, file transfers, recovering corrupted or lost files. These agent (demons) interact directly with the Rucio Storage Elements or external service via the File Transfer Service (FTS).

Namespace: The file managed by Rucio are organised as datasets with associate metadata (called data IDentifiers) and distributed as containers by scope based on the metadata. A Data Identifier is unique within a scope but can be used in different scopes. Moreover, in Rucio the scopes are protected and are only writable. In Rucio the physical paths of the files can be obtained via a deterministic function of the scope and name of the file. Because of the path convention, all the files are locate based on the scope and name. The function has also been chosen to have a well-balanced distribution on the number of files per directories.

Storage: Rucio server support mounting external storage including tape storage. Rucio supports multiple protocols to interact with the Storage Elements; in particular WebDAV, or Amazon S3. Rucio also provides a central HTTP redirector that allows to federate all Storage Elements which simplifies the accesses to files and datasets by end-user. The logical abstraction of a storage endpoint is called a Rucio Storage Element (RSE) which can be tagged with key/values pairs. Different tags can be used together to build a RSE expression.

Replica management: One of the most important features in Rucio is the concept of replication rules and subscriptions. In Rucio users can write replication rules to describe how a Data Identifier can be replicated across a list of Rucio Storage Elements. Rucio create the minimum number of replicas that satisfy the rule to optimise the storage space, minimise the number of transfers and automate data distribution. The Replication policy is a subscription, based on metadata of Data Identifiers. When a Data Identifier that matches the parameters of the subscription is produced, Rucio will generate a rule for it and will create the replicas that satisfy the rule. Because the management of the replicas only the scope and name f the file and the site are needed, the space used by the replicas tables (around 700 GB for the LFC at CERN versus 300 GB for the Rucio replica table).

Authentication and authorisation: Rucio users can register either as individuals or as member of group or community. Rucio users can connect using either a certificate, a proxy certificate, a Kerberos token, or even a user/password.

4.2 TRL applied to software development

Considering that the TRLs were initially introduced to represents the evolution of an idea from a thought to a product in the marketplace[89], at a first sight it might not be directly able to draw conclusions about the quality of different aspects of software. Several studies have been proposed in order to interpret the original TRL to Software TRL[90] like the U.S. Army Workshop Report[91]. Table 4 describes the adapted definitions for software, which are numbered according to the corresponding TRL (level) as found on NASA's website[92].

[88] https://rucio.cern.ch

[89] https://www.nasa.gov/topics/aeronautics/features/trl_demystified.html

[90] Brian Sauser, et al. From TRL to SRL: The concept of systems readiness levels Conference on Systems Engineering Research Los Angeles, CA, April 7-8, 2006

[91] Beyond Technology Readiness Levels for Software
https://resources.sei.cmu.edu/asset_files/TechnicalReport/2010_005_001_15305.pdf

[92] https://goo.gl/XykPfq

Table 4: Technology Readiness Level Definitions (as defined on NASA website)

Level_1	Scientific knowledge generated underpinning basic properties of software architecture and mathematical formulation.
Level_2	Practical application is identified but is speculative, no experimental proof or detailed analysis is available to support the conjecture. Basic properties of algorithms, representations and concepts defined. Basic principles coded. Experiments performed with synthetic data.
Level_3	Development of limited functionality to validate critical properties and predictions using non-integrated software components.
Level_4	Key, functionally critical, software components are integrated, and functionally validated, to establish interoperability and begin architecture development. Relevant Environments defined and performance in this environment predicted.
Level_5	End-to-end software elements implemented and interfaced with existing systems/simulations conforming to target environment. End-to-end software system, tested in relevant environment, meeting predicted performance. Operational environment performance predicted. Prototype implementations developed.
Level_6	Prototype implementations of the software demonstrated on full-scale realistic problems. Partially integrate with existing hardware/software systems. Limited documentation available. Engineering feasibility fully demonstrated.
Level_7	Prototype software exists having all key functionality available for demonstration and test. Well integrated with operational hardware/software systems demonstrating operational feasibility. Most software bugs removed. Limited documentation available.
Level_8	All software has been thoroughly debugged and fully integrated with all operational hardware and software systems. All user documentation, training documentation, and maintenance documentation completed. All functionality successfully demonstrated in simulated operational scenarios. Verification and Validation (V&V) completed.
Level_9	All software has been thoroughly debugged and fully integrated with all operational hardware/software systems. All documentation has been completed. Sustaining software engineering support is in place. System has been successfully operated in the operational environment.

By examining the definitions above, we see that it is more than viable to evaluate a software system's TRL by investigating which points are being fulfilled. Although the application of the above is kind of abstracted, as there is no defined way of correlating a system to the TRL, except by maybe filling in a questionnaire or more like a spreadsheet (one of those available on the internet).

4.2.1 Software TRL calculation

Being able to determine the TRL of a software system can be really useful, not only by giving insight about the status of the system and its development stage, but -potentially- additionally can be used in order to simplify the communication between technical and non-technical parties of and organization or a third party, as well as act as a proof of the quality of the system when it's released in production (CS) or published to the community (OSS). Although for the last argument, we can argue regarding how dependable it is to take as proof of software quality its TRL, when that is calculated with a spreadsheet? That means that a (group of) human(s) is responsible for filling in the accurate response to the spreadsheet, and therefore we can't determine for sure whether the outcome of the calculation is true and valid, as the means by

which it was being calculated are not dependable (i.e. the person making the calculation could have false insight of the system or being biased). Of course, there are several publications[93,94] that propose calculations (equations), but usually they are too specific trying to address a specific type of software.

As the current procedure of calculating the TRL of software systems has several flaws, cannot be used as the dependable proof we described, and so we identify the need of calculating the TRL in such a way that can be valid, have minimized flaws and also act as a proof of the software quality of the system under examination. In order to do so, the procedure that is going to be used, needs to fulfil the same requirements on its turn.

In order to tackle the problem, we described, we argue that if actual metric measurements were involved to the TRL calculation, it would support the validity of the outcome. Software attribute measurements can be a difficult but valuable task, as it provides great insight of several aspects of the software system. At this point, we would like to discuss the proposed approach for calculating the TRL. It is really important to highlight that such procedure, is (most likely) impossible to be automated, as until now it is being done by the use of spreadsheets. Additionally, the proposal also gives space for fine-grained adjustment, precisely as it is desired for (different types of systems require different measurements and also same metrics can have different impact depending on the type as well). First comes the software quality measurement, which is complex and important, as it is the basis for the idea. Once we are able to actually make measurements, we need to be able to understand what they mean about software quality (maturity) as it expressed by the TRL scale, and therefore we need to approximately interpret and correlate the TRL levels with the software quality measurements, in order to be able to extract the information required, and even more, argue about the achieved TRL.

4.2.2 Software quality measurement

For this procedure we are making use of software analysis tools that provide the possibility of fine-grained measurements, tailored to project specific needs. We came across several tools that could be used, but we are going to include only 2 (or 3) of them for the means of the present report. The mentioned tools are the ones that we preferred to use for our research and investigation namely: SonarQube[95], Codacy[96], and Codefactor[97]. Some preliminary experiments with the three code evaluation tools provided us with some insight on the advantages and disadvantages of each tool (see Table 5)

Table 5: Summary of the Advantages and Disadvantages of the three selected code analysis tools

	Advantages	Disadvantages
SonarQube	**Customizable**: enable to define/select the rules to apply during analysis. Offers the option of tagging like; “false positive”, “removed” etc. **Issues grading**: identified issues are graded on based on their severity, making possible to outline the critical issues. **Team working**: gives the possibility to assign people to specific issues/tasks.	The tool requires some time to be set up, along with calibrating for the desired results. Requires project specific “properties” file, which is great as you can define the parts of the project you want analysed, but sometimes can be bothersome. The compiled build is required to run the analysis, which means that you are forced

[93] Taner Altunok, Tanyel Cakmak, A technology readiness levels (TRLs) calculator software for systems engineering and technology management tool, Advances in Engineering Software,
Volume 41, Issue 5, 2010, Pages 769-778, ISSN 0965-9978

[94] A. Parasuraman, Technology Readiness Index (Tri): A Multiple-Item Scale to Measure Readiness to Embrace New Technologie, Journal of Service Research, vol 2, no 4, pages 307-320, 2000, doi=10.1177/109467050024001

[95] https://www.sonarqube.org

[96] https://app.codacy.com

[97] https://www.codefactor.io

	Advantages	Disadvantages
	Reporting of Metrics: provides a breakdown containing the metrics used, their impact, and also an estimation of the effort required to fix them. **Visual summary** of the code's quality: by illustrating the relation between the effort required and the severity of the issues.	to deploy the project locally in order to produce all the required files.
Codacy	**Online code Analysis**: by providing projects github/bitbucket url. In depth analysis as well, along with explanation of why the specific issue counts as an issue and with examples of "good" and "bad" implementation of the issue in order to guide the user. **Classify issues**: The reported issues can be filtered according to "Language", "Category", "Level" and "Pattern". That way the tool allows the user to make really specific or more generic selections, in order to target specific aspects of the system in its current state. **Team Work**: offers the functionality of creating organization and assigning tasks to members, which improves cooperation in a team	None that was important to our investigation.
Codefactor	Importing new project: Easy to import projects and analyze them. **Grading**: More accurate grading compared to the other two tools, codefactor allow to add more severity levels to capture a have a finer grading.	Minimalistic user interface (a less user-friendly environment). **Customizing rules**: not available. **Code Analysis**: Less in depth analysis probably as there are no comments regarding vulnerabilities or security for example.

Table 6: Summary of the Analysis report after analysing the source code of the projects we are considering in the PROCESS project

	Sonarqube			Codacy			Codefactor		
Project	Overall problems identified*	Analysed	Score/ Grade*	Overall problems identified*	analysed	Score/ Grade*	Overall problems identified*	analysed	Score/ Grade*
Cookery		✓	A		✓	B		✓	C+
Pumpkin	1366	✓	B-	508	✓	C	710	✓	F
Weevilscout		✓	C+		✓	B		✓	D-
Sonarqube		×			✓	B		✓	A
Lobcder		×		3302	✓	B	945	✓	B-
OwnCloud	8669	✓	C	42416	✓	B	56556	✓	F
OneData		×		485	✓	B	1076	✓	C
NextCloud		×		6699	✓	A	4415	✓	B
dCache		×		7825	✓	B	2900	✓	B
iRODS		×		1253	✓	A	3252	✓	C-

(× analysis of the code was not possible using the open source version of the tool)

4.2.3 Mapping TRL to assessment metrics

The software analysis tools, presented in 4.2.1, are able to assess software starting from TRL 4 and above. Mapping which metrics can give insight on the TRL levels is needed. Table 7 describes the mapping we propose between TRL levels and the assessment metrics provided by the three selected software analysis tools.

Table 7: Attempt to map the TRL levels to concrete assessment metrics that could be used in with the code analysis tools

TRL levels	Assessment metrics
Level 4,5	Coding standards, style, and compatibility metrics.
Level 6	Minimizing/solving bugs & error prone issues.
Level 7	Solving all errors that are possibly produced, focusing security issues additionally.
Level 8,9	We continue by exterminating as many issues as possible, (striving towards better grades in the grading tools).

The more we move to higher levels; the more attention needs to be given to the severity of the issues. Errors should be kept as low as possible at all times, but of course if not possible, they need to be solved in order to reach level 7-8 as at that point we are talking about the final stages of the system development and possible deployment. Not to forget that level 8 was initial the top level and level 9 was added later so we can take it as a refinement level.

4.3 PROCESS Storage Resources

Table 8: Summary of the storage resources available and the respective access protocols available in PROCESS

Sites	Storage Size	Access - Mount point? - NAS? Others ...	VMs - Public IP - Access to storage from the VM
AGH	21 TB (permanent) 100 TB (temporary, 30 days max.)	Mounted on the UI and the Prometheus worker nodes (via Lustre) - direct access from the jobs; external access via SSH based solutions (incl. SCP/SFTP)	By default, private IP, unrestricted outgoing connections via SNAT, incoming connections via: - Reverse proxy (HTTP/S) - TCP/UDP port redirection (DNAT mechanism) - OpenVPN Optional public IP on justified request VM Storage: - Default 10 GB for / - Optional Volume (block device) upon request - Access to Prometheus storage via SSH (e.g. SSHFS)
UISAV	48 TB (permanent)	Access via a dedicated server, storage mounted via NFS; LOBCDER or other data management software may be installed	Public IP available Storage mounted to VMs via NFS
LMU	100 TB (permanent)	Access via dedicated VMs, there mounted via NFS	Public IP available Data Storage mounted via NFS
UvA	2TB for pilot experiments. For real experiment, we can apply for much more storage	Access via dedicated VMs, there mounted via NFS	Public IP available

Core PROCESS Storage node = storage + cloud virtualization (virtual Machine, which can mount the storage).