NHR-NORD@GÖTTINGEN

Data Management Concepts for Efficient and User-Friendly HPC

Hendrik Nolte
hendrik.nolte@gwdg.de

Matthias Eulert
matthias.eulert@gwdg.de

5. November 2024

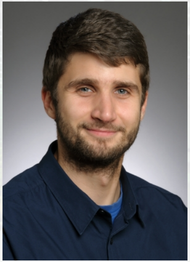
hpc@gwdg.de
GWDG – Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen

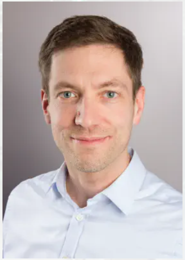
Learning Objectives

- At the end of the two blocks today you should be able to:
  - Use tiered storage systems employed by HPC systems
    - to optimize performance and prevent data loss
  - Organize your HPC workloads within workflows
    - Implement user isolation and data sharing in projects using UNIX permissions
  - Design your own HPC workflows
    - Describe independent tasks and data sets with standardized annotations
    - Classify these steps and data sets using annotations
    - Identify the correct storage tier/system based on these annotations
  - Understand UNIX permissions for data sharing and user isolation
  - Identify scenarios where a data catalog is useful
  - Recognize opportunities to ask for dedicated data management support
    - E.g. via our Ticket system

Your Trainers

- **Hendrik Nolte**
  - Part of the HPC-Team
    - Research, Sys-Admin, User Support
  - Responsible for
    - Data Management
    - Confidential Computing
  - hendrik.nolte@gwdg.de
- **Matthias Eulert**
  - Part of the HPC-Team
    - Professional trainer, User Support
  - Responsible for
    - Trainings and PR
    - Research into standardized Trainings
  - matthias.eulert@gwdg.de





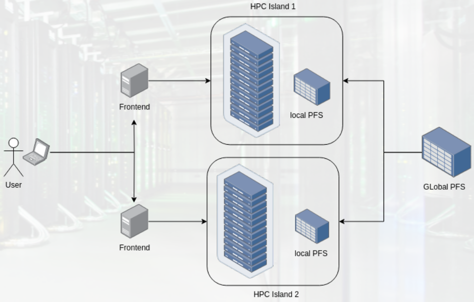
Motivation for Data and Life Cycle Management

- Modern HPC systems are really, really complicated
- It is unbelievably easy to lose oversight
- This makes collaborations close to impossible
- The only solution is careful planning
- And rigorously execution of the plan
- This plan has to become the single source of truth

General Overview HPC Architetcure

NHR-NORD@GÖTTINGEN

- The different clusters form compute islands
  - For example, Emmy, Grete
- Each of these islands has its own fast storage
- And there are more centralized storage systems
- Access to each island via centralized Slurm
- Not GWDG specific, see Dragenfly topologies

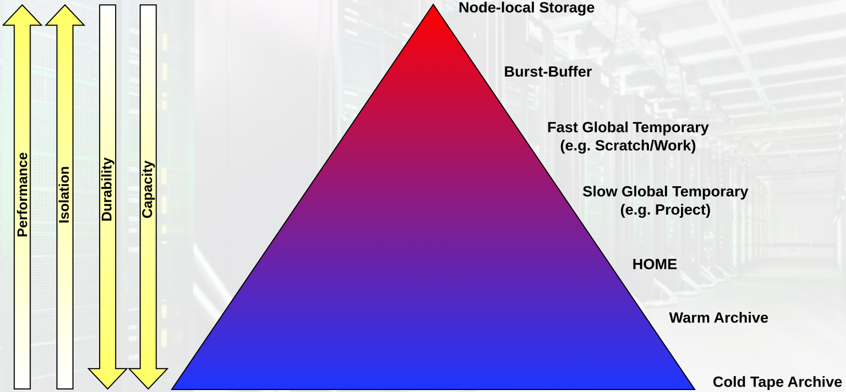


Why Storage Tiering?

- In the previous example, two storage tiers were shown
- Users have different requirements depending on the type of data
  - Think of Software as compared to hot data
- The different storage systems differ in many attributes, e.g.
  - Size
  - Speed
  - Data Durability
  - Backups
  - **Locality**
  - Lifetime, e.g. only available during job runtime, certain TTL, etc.

Tiering Pyramid

NHR-NORD@GÖTTINGEN



- This concept is often visualized in such a pyramid

An Example at GWDG

NHR-NORD@GÖTTINGEN

Project Origin	Name	Storage Kind	Storage Type	Clusters	Path	Disk Kind	Filesystem	Backed Up	Description
all	Project Directory	MAP	Filesystem	Emmy, Grete	`/projects/PROJECTPATH`	SSD	VAST NFS	yes+snapshot	Symlink farm pointing to all the data stores
NHR	NHR Archive	ARCHIVE	Filesystem	Emmy, Grete	`/perm/projects/PROJECT`	Tape	Stornext	yes	Archival storage (very robust, very slow)
NHR (legacy)	Legacy Project HOME	HOME	Filesystem	Emmy, Grete	`/home/projects/PROJECT`	HDD	GPFS	yes+snapshot	HOME storage for the project (robust, but slow and small)
NHR	Project HOME	HOME	Filesystem	Emmy, Grete	/mnt/ddn-gpfs/projects/PROJECT	HDD	GPFS	yes+snapshot	HOME storage for the project (robust, but slow and small)
NHR	Lustre Emmy HDD	SCRATCH	Filesystem	Emmy, Grete	`/mnt/lustre-emmy-hdd/projects/PROJECT`	HDD	Lustre	no	Large and reasonably fast storage optimized for Emmy
NHR	Lustre Emmy SSD	SCRATCH	Filesystem	Emmy, Grete	`/mnt/lustre-emmy-ssd/projects/PROJECT`	SSD	Lustre	no	Small and fast storage optimized for Emmy
NHR	Lustre Grete	SCRATCH	Filesystem	Grete	`/mnt/lustre-grete/projects/PROJECT`	SSD	Lustre	no	Small and fast storage optimized for Grete
NHR (legacy)	scratch-emmy	SCRATCH	Filesystem	Emmy, Grete	`/scratch-emmy/projects/PROJECT`	HDD	Lustre	no	Large and reasonably fast storage optimized for Emmy
NHR (legacy)	scratch-grete	SCRATCH	Filesystem	Grete	`/scratch-grete/projects/PROJECT`	SSD	Lustre	no	Small and fast storage optimized for Grete

- Already 9 storage tiers, and the local `tmpfs` and SSD's are even neglected

Resulting Problem

- People are overwhelmed and do wrong data placement
  - Hot data sits on cold storage
  - Cold data sits in hot storage
  - Standard datasets sit on expensive backed up storage
  - Important results are on fragile storage
  - The wrong storage system for the wrong cluster island is used
    - GWDG might be an edge case here, but also think of a Dragonfly Topology
- Many storage tiers quickly lead to a loss of oversight
  - Data is not cleaned up
  - Data is not reproducible, unclear where it belongs to
  - Data loss, hard to find
  - How can I select all data with a certain property?

Interactive Q&A

NHR-NORD@GÖTTINGEN

- Were you aware of these tiered storage systems?
- How is it at your center?
- Did you experience any problems with the (storage) systems?
- What were your solutions?

Experimental Planing I

NHR-NORD@GÖTTINGEN

- The experimental planning is defined within a data management plan (DMP)
  - It defines properties and policies of the measured or computed data
  - It is a living, and in this case light-weight, document
  - It should be designed at the beginning of a project and then continuously updated
  - **Goal:** Conceptually find the most suitable storage tier and define the data flow
- It starts at the data source(s)
  - Where is what data generated by whom?
  - How much data is generated per day/week/month?
  - Does the data need to be checked for validity?
  - How can the data be transferred from the source and to where?
  - Usually raw data is extremely valuable, how is it being backed up
  - Who needs access to it?
  - What are the semantic metadata to describe this data?

Experimental Planing II

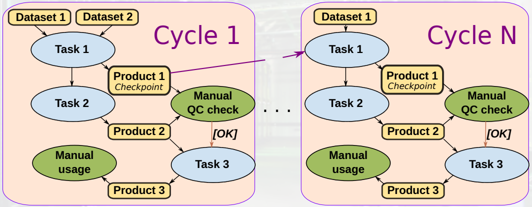
NHR-NORD@GÖTTINGEN

- Define the first processing step
  - Is it automatic or manual/interactive?
  - Who does it?
  - Is it compute intensive?
  - Is it I/O intensive?
  - What are the data products?
- First Data product
  - Only temporary/intermediate result?
  - Should it be retained?
  - Manual check?
  - Semantic Metadata
  - How will it be processed?
  - Will it be used IO intensively?

Thinking within Structured Workflows

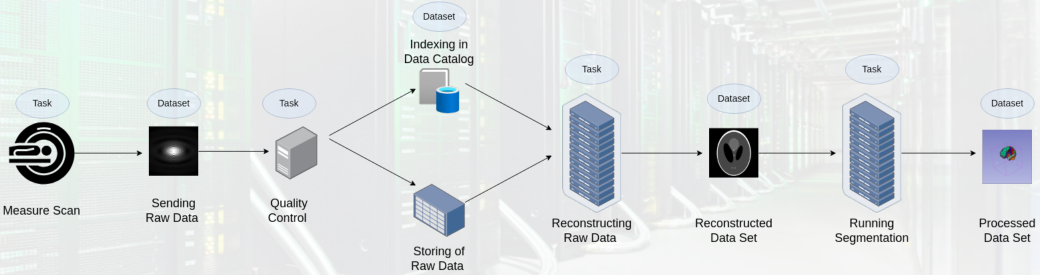
NHR-NORD@GÖTTINGEN

- It is useful to organize an HPC experiment within an overarching workflow
- In the first step, data sets and tasks are linked
- Then, these can be further annotated, e.g.
  - Who needs access
  - Semantic Metadata
  - Temporary or permanent result
  - ...
- **What do you think is important as well?**



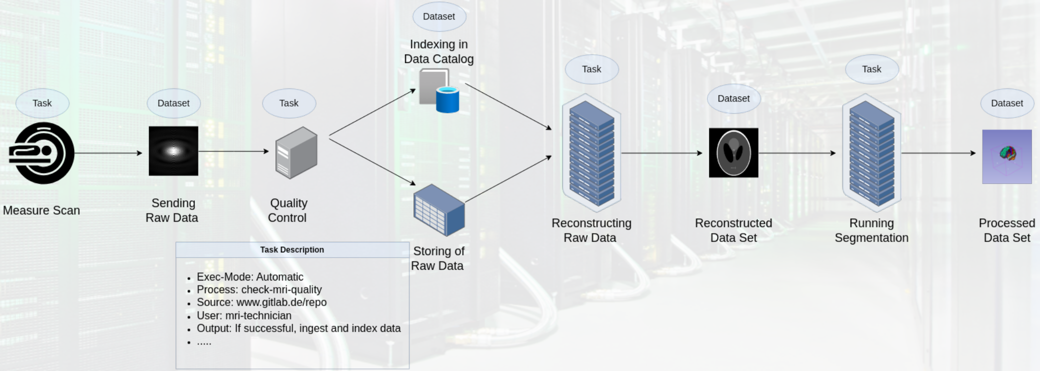
Experimental Planning Example: MRI Workflow Overview

NHR-NORD@GÖTTINGEN



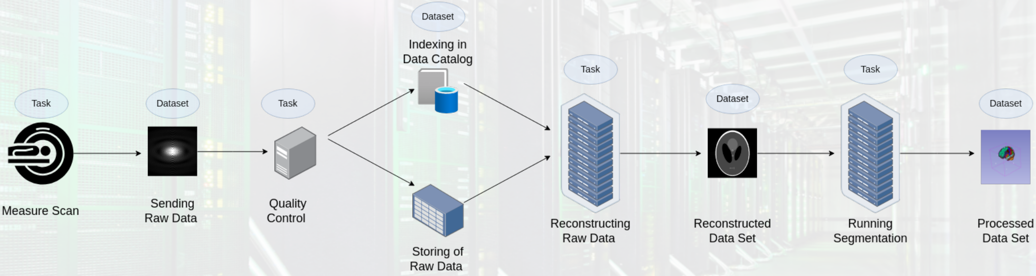
Experimental Planning Example: Workflow Annotations I

NHR-NORD@GÖTTINGEN



Experimental Planning Example: Workflow Annotations II

NHR-NORD@GÖTTINGEN

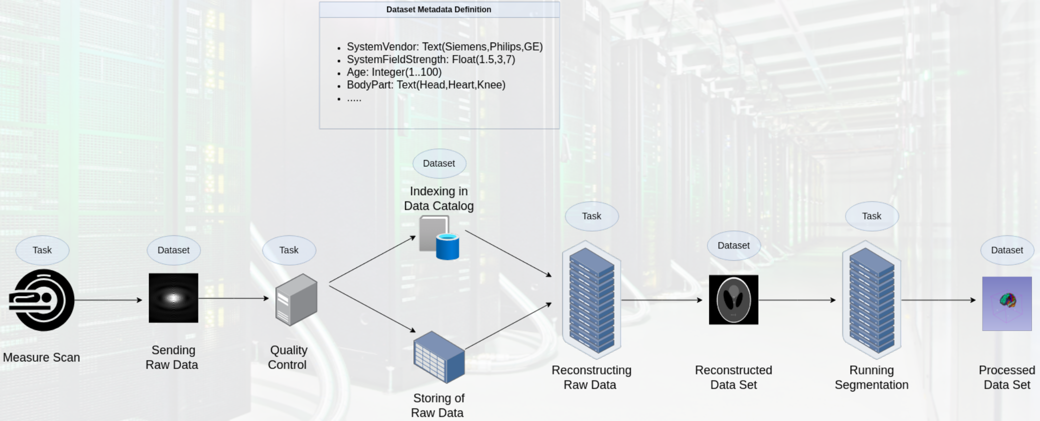


Dataset Description

- Durability: High
- Don't allow delete/overwrite
- Index Metadata
- Access: Reco-User-1

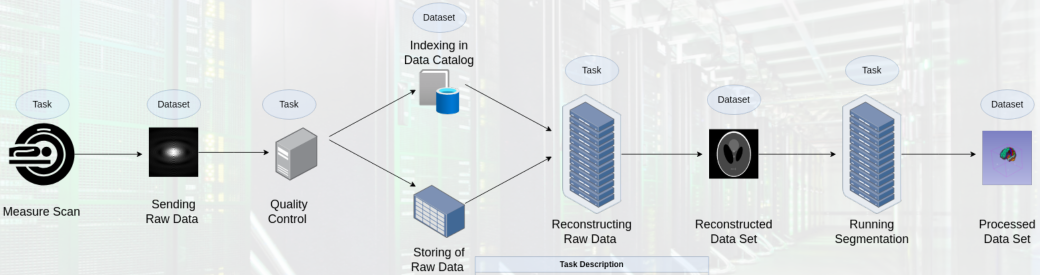
Experimental Planning Example: Workflow Annotations III

NHR-NORD@GÖTTINGEN



Experimental Planning Example: Workflow Annotations IV

NHR-NORD@GÖTTINGEN

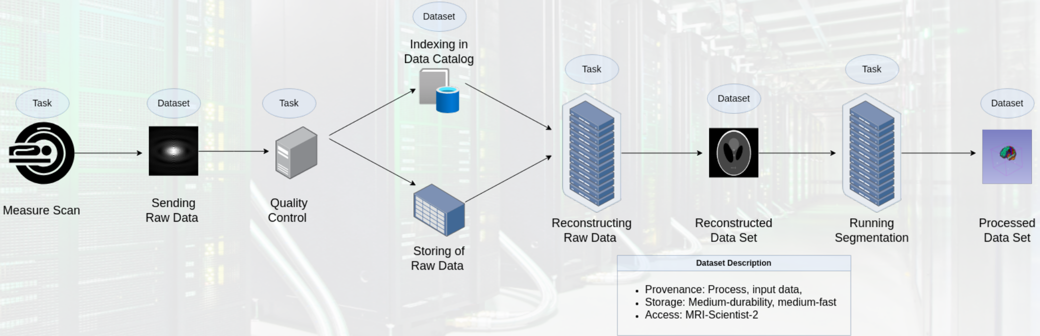


Task Description

- Exec-Mode: Manually Triggered
- Process: reconstruct-mri-image
- Source: www.gitlab.de/reconstruct-repo
- User: mri-reco-scientist
- Input: MRI Raw Data
- Storage: Fast
- Output: MRI Images
- .....

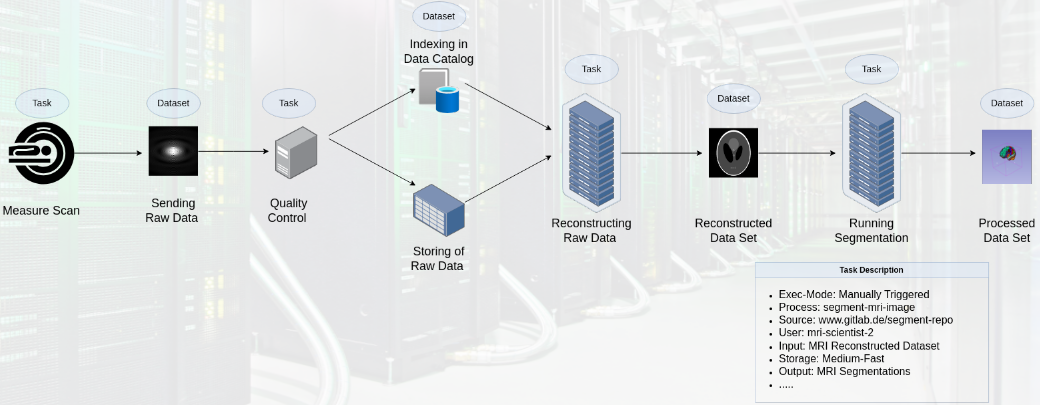
Experimental Planning Example: Workflow Annotations V

NHR-NORD@GÖTTINGEN

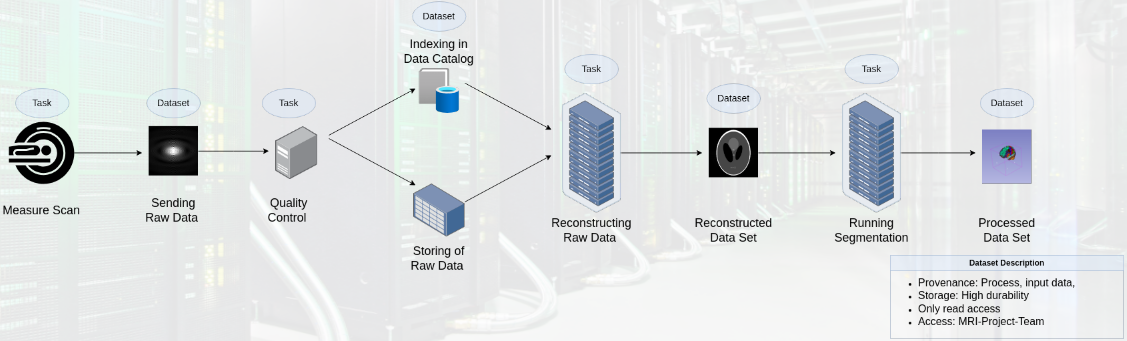


Experimental Planning Example: Workflow Annotations VI

NHR-NORD@GÖTTINGEN



Experimental Planning Example: Workflow Annotations VII



Experimental Planning: Interactive Q&A

NHR-NORD@GÖTTINGEN

- Before coming back to the storage tiering, I have a few questions:
  - Who of you has done something similar in the past?
    - More complicated, or way simplified?
    - What were the differences/similarities?
    - What was your use case?
  - If not, how do you organize your experiments instead?
  - Do you think such a planning is useful?
    - Or just a waste of time?

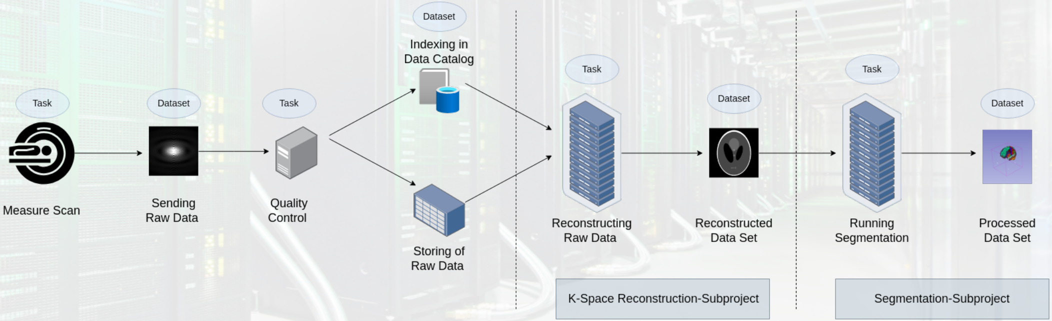
Experimental Planning Example: Project Decomposition I

NHR-NORD@GÖTTINGEN

- This overarching *MRI-Project* can be further divided into two sub-projects:
  - *K-Space Reconstruction*-Project
  - *Segmentation*-Project
- Advantages:
  - Every (physical/human) user can get a dedicated `uid`
  - Allows fine-granular permission control
    - We will cover this in the second block this afternoon
  - Fine-granular Quota enforcement

Experimental Planning Example: Project Decomposition II

NHR-NORD@GÖTTINGEN



Mapping to User Roles and File Paths: Step 1

NHR-NORD@GÖTTINGEN

- Now, the high-level experimental plan gets mapped on user roles and file paths
- User Roles:
  - MRI-Technician as a **Data Steward** to upload curated data
- File Paths:
  - Upload to `/projects/mri-project/raw/`
    - **rw** access for user *mri-technician*, **r** access for user *mri-reco-scientist*
    - located on durable, but slow cold/warm storage

Mapping to User Roles and File Paths: Step 2

NHR-NORD@GÖTTINGEN

- User Roles:
  - MRI-Reco-Scientist can read raw data from the *MRI-Project* (located at `/projects/mri-project/`) as a **ReadOnlyMember**
  - MRI-Reco-Scientist can write data to their sub-project *K-Space-Reconstruction-Subproject* (located at `/projects/k-space-reco-project/`) as **PI**
- File Paths:
  - Stage data from `/projects/mri-project/raw/` to `/projects/k-space-reco-project/input-data/`
    - **rw** access for user *mri-reco-scientist*
    - fast storage for GPU, input data has a limited, preconfigured lifetime, e.g., 6 months
    - Completely different filesystem, but same semantic

Mapping to User Roles and File Paths: Step 3

- User Roles:
  - MRI-Reco-Scientist can run arbitrarily their experiment on the *K-Space-Reconstruction-Subproject* (located at `/projects/k-space-reco-project/`) as **PI**
- File Paths:
  - Work on `/projects/k-space-reco-project/`

Mapping to User Roles and File Paths: Step 4

NHR-NORD@GÖTTINGEN

- User Roles:
  - MRI-Reco-Scientist puts their final artifacts into a staging area to share with *K-Space-Reconstruction-Subproject* (located at `/projects/k-space-reco-project/share`) as **PI**
  - MRI-Scientist-2 has read access to that staging area as **ReadOnlyMember**
  - A **Data Steward** can copy the staged data into the main *MRI-Project*
- File Paths:
  - Stage data from `/projects/k-space-reco-project/share`
    - **r** access for user *mri-scientist-2*
    - **rw** access for user *mri-reco-scientist*
    - fast storage for GPU, input data has a limited, preconfigured lifetime, e.g., 6 months
  - to `/projects/segmentation-project/input-data/`
    - **rw** access for user *mri-scientist-2*
    - fast storage for CPU, input data has a limited, preconfigured lifetime, e.g., 6 months
    - Different filesystem in a different island/partition

Mapping to User Roles and File Paths: Step 5

- User Roles:
  - MRI-Scientist-2 can run arbitrarily their experiment on the *Segmentation-Subproject* (located at `/projects/segmentation-project/input-data/`) as **PI**
- File Paths:
  - `/projects/segmentation-project/input-data/`
    - **rw** access for user *mri-scientist-2*
    - fast storage for CPU, input data has a limited, preconfigured lifetime, e.g., 6 months

Mapping to User Roles and File Paths: Step 6

NHR-NORD@GÖTTINGEN

- User Roles:
  - MRI-Scientist-2 stages their data product within the *Segmentation-Subproject* in a staging area (located at `/projects/segmentation-project/merge/`) as **PI**
  - **Data Steward** of overarching *MRI-Project* curates staged artifacts and merges them back to the main project (e.g. `/projects/mri-project/segmentations/`)
- File Paths:
  - `/projects/segmentation-project/merge/`
    - **rw** access for user *mri-scientist-2*
    - **r** access for **Data Steward** of parent *MRI-Project*
    - fast storage for CPU, input data has a limited, preconfigured lifetime, e.g., 6 months
  - `/projects/mri-project/segmentations/`
    - Only read access for most members
    - slower, but durable storage
    - long lifetime

Mapping to User Roles and File Paths: Step 7

NHR-NORD@GÖTTINGEN

- User Roles:
  - **Data Steward** prepares staging area of a dataset to be published (e.g., located at `/projects/mri-project/stage-to-data-pool/`)
  - **Data Steward** provides metadata about this dataset for other users in a sidecar file
- File Paths:
  - `/data/mri-dataset/`
    - **r** access for users
    - either for all or only upon invitation

Quick Recap

NHR-NORD@GÖTTINGEN

- We have created a workflow containing the experimental description
- The nodes of this workflow were annotated, containing information about
  - The users needing access
  - The storage requirement, e.g., fast, durable, ...
  - The processes that should run
  - Data Volume
  - Life Cycle
- Based on this description the overarching project was split into isolated subprojects
- Lastly, the involved users were mapped on roles and specific storage paths

Interactive Q&A: Advantages of this approach

- What advantages/disadvantages do you see with this approach?

Advantages of this approach

NHR-NORD@GÖTTINGEN

- A clear target state is defined
  - This allows to later on compare the is state to the ideal target state
- Different users can be isolated within a single overarching project
  - No data loss due to an undercoffinated `rm -rf *`
  - Later: Restricted Quota assignment
- Merging to a main project ensures data quality
  - Can be data products, software environments, etc.

Interactive Q&A

- What is your opinion about such an approach?
- Do you have a use case, that we could discuss?

Group Work: Breakout Room Session

NHR-NORD@GÖTTINGEN

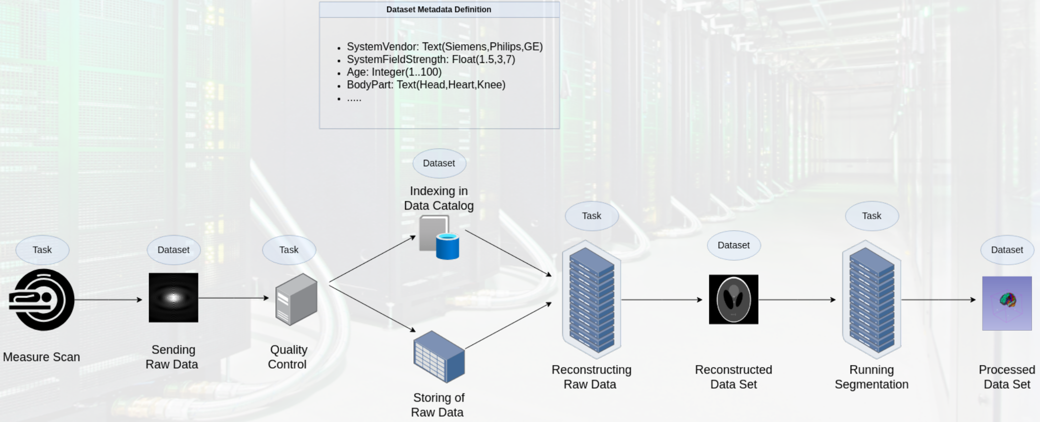
- Example: Experimental data (e.g. PIV)
  - Measurements are thousands of images
  - Uploaded to HPC and processed
  - Processing results are averaged into time-dependent field
  - Time-dependent results are averaged into low amount of numbers
- Example: Simulation results (e.g. CFD)
  - Full 3D field simulations
  - Fields stored in regular intervals for restart
  - Slices of 3D field stored
  - Time-dependent volume averages computed

Group Work: Breakout Room Session

- Get together in groups and think of another use case
- Define the workflow as a graph
  - What subprojects can be defined?
  - How can the datasets be annotated?
  - How can the tasks be annotated?
- Lets reconvene in **20** minutes and discuss your use cases

Data Catalogs I

- Quick digression into **Data Catalogs**
- You have seen the usage of a data catalog previously:



Data Catalogs II

NHR-NORD@GÖTTINGEN

- The idea of **Data Catalogs** is
  - To use domain-specific metadata to describe a dataset/file
    - Will cover automatic metadata extraction in the last session today
  - Instead of encoding this information within paths and file names
  - The file path becomes a simple handle to access a datum
- Advantages:
  - One does not have to remember paths
  - No confusion because `/scratch` are different file systems on different systems
  - One gets a global view of all data on all filesystems
    - Across all the different storage tiers
    - Can be used to orchestrate data staging and migration
  - Actually much faster data retrieval

Data Catalogs: Data Modeling

NHR-NORD@GÖTTINGEN

Excerpt from the MRI Model

```
{
"SystemVendor" : "Text('Siemens', 'Philips', 'GE')",
"SystemFieldStrength" : "Float(1.5,3,7)",
"NumberReceiverChannels" : "Integer(8,12,16,32,64)",
"Age": "Integer(1..100)",
"BrainAge" : "Integer(Age + NormDist(5))",
"BodyPart" : "Text('Head','Heart','Knee')",
...
}
```

- First the model is defined
  - MRI Example: 61 keys
- Using JSON to write it
- The left image defines the model
- Each scan (file) will become one concrete instance of this model

Data Catalogs: Metadata Example

NHR-NORD@GÖTTINGEN

Excerpt from one specific MRI Scan

```
{
"SystemVendor" : "Siemens",
"SystemFieldStrength" : 3,
"NumberReceiverChannels" : 16,
"Age": 68,
"BrainAge" : 75,
"BodyPart" : "Head",
...
}
```

- One concrete example of a single scan
- The model can be used for data quality control
- These `json` files can be used as a sidecar file
  - Have both in the same directory with the same name
    - `/data/scan_hnolte.nii`
    - `/data/scan_hnolte.json`
- The `json` file describes the `nii` file

Data Catalogs: Example

NHR-NORD@GÖTTINGEN

- For each file that should be cataloged the required metadata needs to be provided
  - For instance via a sidecar file as in the previous example
  - Once a data file has a sidecar file one can index it
  - E.g. `goedl --ingest-folder /path/to/data/`
- Afterwards you can check what data matches specific attributes
  - E.g. `goedl --list key1=val1,key2=val2`
    - Returns a list of matching files (handles)
    - Is therefore completely transparent for different/tiered storage systems
- Can be used to stage data on a certain subproject path before computing
  - E.g. `goedl --stage key1=val1 --target /path/to/staging-area`

Interactive Q&A

- Is working with semantic metadata and handles a new approach for you?
- Do you encode metadata in paths / filenames?
- Do you already use a data catalog?
- What do you think of such an example?
- What are the advantages/disadvantages?
- In what scenarios can you utilize a data catalog?
- Can you describe the datasets in your previous example in such a way?

Hands-On Session: Data Ingest

NHR-NORD@GÖTTINGEN

- **rsync**
  - Works via ssh and is terminal-based (For all Linux users)
  - Copy data from your system to the HPC:
    `rsync -avvH <source> <user>@transfer-scc.gwdg.de:/usr/users/<user>/<target>`
  - If `<source>` is a folder it will copy the content to `<target>`
- **Owncloud**
  - Go to `owncloud.gwdg.de` and create a folder
  - Click on `...`, via **Details** to **Sharing** and create a **Public Link** and **Set a Password!**
  - Extract the folder ID from the public link:
    `https://owncloud.gwdg.de/index.php/s/<folder_id>`
  - On the HPC system within the terminal run for downloading:
    `rclone sync --webdav-url=https://owncloud.gwdg.de/public.php/webdav --webdav-user=«folder_id>-webdav-pass="$(rclone obscure '<folder_password>')-webdav-vendor=owncloud :webdav: <local_dir>`
- You find all documentation here [1]

[1] https://docs.hpc.gwdg.de/usage_guide/data_transfer/index.html

Hands-On Session: Data Ingest

- What are the differences between the two Ingestion methods?
- In which scenario would you use each of these?

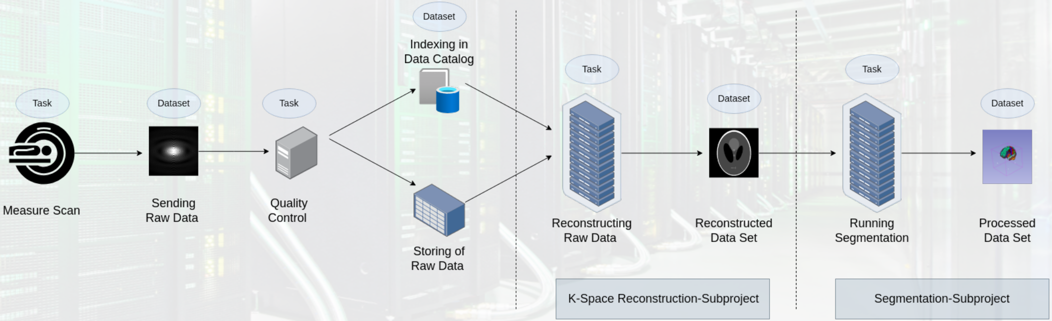
- **UNIX Permissions: Data Sharing and User Isolation**

Access Management

NHR-NORD@GÖTTINGEN

- You have developed and decomposed your own workflow before
  - We distinguished in the user roles between shared and private paths
  - Let's implement it!

Reminder



Task
Measure Scan
Dataset
Sending
Raw Data
Task
Quality
Control
Dataset
Indexing in
Data Catalog
Storing of
Raw Data
Task
Reconstructing
Raw Data
Dataset
Reconstructed
Data Set
Task
Running
Segmentation
Dataset
Processed
Data Set
K-Space Reconstruction-Subproject
Segmentation-Subproject

UNIX Permissions

NHR-NORD@GÖTTINGEN

- A key aspect of achieving the envisioned user isolation are UNIX permission:

```
glogin1:  $ ls -lah $WORK
drwx------ 5 gzadmhno gzadmhno 4,0K 2. Jun 2022 io500-isc
drwxr-x--- 2 gzadmhno gzadmhno 4,0K 19. Jun 2023 data
-rw-r----- 1 gzadmhno gzadmhno 9,8G 19. Jun 2023 file_1
```

- The leading `d` indicates a directory, the `-` that it is a file

UNIX Permissions: Basics

NHR-NORD@GÖTTINGEN

- The next 9 positions are made up by 3 tuples:
  - **r:** read access
  - **w:** write access
  - **x:** execute permission
- The first tuple is for the user (owner): **u**
- The second for the group: **g**
- The last for others: **o**
- On files this is "straight forward"
- On directories it means:
  - **r:** list files within a directory
  - **w:** create and delete files
  - **x:** `cd` or traverse into that directory

UNIX Permissions: Interactive Basics Quiz

NHR-NORD@GÖTTINGEN

- Let's discuss what the following permissions allow:

  1: `drwxrwxr-x 2 hendrik hendrik folder1`

  2: `drwx--xr-- 2 hendrik hendrik folder2`

  3: `d--xrwx-w- 2 hendrik hendrik folder3`

  4: `-rw-rw-r-- 1 hendrik hendrik file1`

  5: `-r--r----x 1 hendrik hendrik file2`

  6: `-r--r--rwx 1 hendrik hendrik file3`

  7: `drwxrwxr-x 2 hendrik hendrik file4`

UNIX Permissions: S-Bit

- SGID Bit on a directory
  - Any files created in that directory will set group ownership to the directory owner
  - Special Bit **s** on **g** (group)
  - If set, an **s** is where the **x** is usually located (for the group)
    - If execute permissions are not set, then an uppercase **S** is displayed
- Sticky Bit on directory
  - Only the **owner** of a file can delete it
  - Special Bit **t** on **others**
  - If set, an **t** is where the x is usually located (for others)
    - If execute permissions are not set, then an uppercase **T** is displayed

UNIX Permissions: Interactive S-Bit Quiz

NHR-NORD@GÖTTINGEN

- Let's discuss what the following permissions enforce:

1: `drwxrwsr-x 2 hendrik hendrik folder1`
2: `drwxrwxr-T 2 hendrik hendrik folder2`
3: `drwxrwSr-x 2 hendrik hendrik folder3`
4: `drwxrwSr-t 2 hendrik hendrik folder4`

Setting/Changing UNIX Permissions

- `chmod`: Command for changing permission
  - `chmod (u|g|o|a)(+|-|=)(r||w||x|| ) TARGET`
- Few examples:
  - `chmod a+r test.txt` Gives everyone read permission
  - `chmod g= test.txt` Removes all permission for group
  - `chmod u+x test` Allows execution of test
  - `chmod u+X test-dir` Make directories (but not files) executable/ “cd able”
  - `chmod -R g+rwX test-dir` Makes test-dir and files and folders in it group readable and writable, -**R** flag makes it recursive
  - `chmod +t test-dir` Adds sticky-bit **T** to test-dir
  - `chmod g+s test` Add SGID bit

UNIX Permissions: Ownership

NHR-NORD@GÖTTINGEN

- Every file and directory is owned by a user and a group
- Very important concept since the previous permissions are using the ownership
- Usage:
  - `chown NEW_OWNER TARGET` Change the ownership of target
    - Usually **not possible** in user space
    - `chown hendrik.hendrik file`
    - `chown hendrik:hendrik file`
  - `chgrp NEW_GROUP TARGET` Change the group of target
    - **Possible** in user space
    - `chgrp cdrom file`
  - `whoami` Show own username

Group Work: Overview Data Management

- List all aspects discussed today for efficient data management
- State their individual contributions
- Describe how you would proceed to implement a real use case
- Lets compare in **7 minutes**

Group Work: Overview Data Management

- Key Aspects:
  - Write down your workflow as a graph
    - Differentiate between tasks and data sets
  - Annotate each node (Or even edge?)
    - Defining semantic metadata
    - Access Control Lists
    - ...
  - Setup project structure accordingly
  - Check that the definition and implementation match
    - Integrate quality control where defined
- Individual Contributions:
  - Comprehensibility
  - User isolation
  - Correct Data Placement (Findability, Performance, Prevent Data Loss)
  - Quality Control

Group Work: Setting Up Directory Structure

NHR-NORD@GÖTTINGEN

- You have previously defined and annotated your own workflow
- Implement a directory structure that
  - enables the defined data-sharing
  - restricts access as far as possible
- You have all gotten `u1234` training accounts, which are all in the same group
- Find yourself in groups and implement and test your workflow
  - Reuse the old groups in which you defined your use case
- If you don't have a suitable workflow defined, take my MRI example
- Lets meet again at **10:45 am**

Group Work: Setting Up Directory Structure

- Present your solution!
  - Maximum of 5 Minutes
- What are your impressions?

Recap

- Use tiered storage systems employed by HPC systems
  - to optimize performance and prevent data loss
- Organize your HPC workloads within workflows
  - Implement user isolation and data sharing in projects using UNIX permissions
- Design your own HPC workflows
  - Describe independent tasks and data sets with standardized annotations
  - Classify these steps and data sets using annotations
  - Identify the correct storage tier/system based on these annotations
- Identify scenarios where a data catalog is useful
- Recognize opportunities to ask for dedicated data management support
  - E.g. via our Ticket system