MrData: An iRODS Based Human Research Data Management System

Blake G. Fitch
Max Planck Institute for Biological Cybernetics[1]
blake.fitch@tuebingen.mpg.de

Sebastian Müller*
Max Planck Institute for Biological Cybernetics[1]
Eberhard Karls University Tuebingen[2]
sebastian.mueller@tuebingen.mpg.de

Dario Bosch
Max Planck Institute for Biological Cybernetics[1]
Eberhard Karls University Tuebingen[2]
dario.bosch@tuebingen.mpg.de

ABSTRACT

MrData is an iRODS based archival system for human subject research producing medical image data. MrData was designed to automate collection and archival of data flowing from a Siemens 9.4 Tesla MRI system. Of particular importance to this project was managing metadata related to human subject recruiting in a GDPR compliant manner. We chose Castellum, a Max Planck developed open source system specifically designed for managing human subject data securely, and we worked with that team to integrate it with the MrData system. An additional requirement for us was "mixed use" metadata, that is information necessary for both subject recruiting and scientific processing. Mixed use metadata, such as handedness, is managed by Castellum but also passed to MrData for scientific and archival purposes securely, and without manual transcription. Our system never records any personally identifying information at the MRI scanner, so the resulting image files are never contaminated with a subject name, date of birth, etc. MrData is based on the iRODS ecosystem, GitLab, Flask, and Python processes, and deployed as a set of Docker micro-services. We will present an overview of this project, including current production status and future directions. We welcome feedback on whether some or all of this system would be usefully open-sourced.

Keywords

iRODS, MRI, medical imaging, data management, Python, GDPR, DICOM, Flask, Castellum.

INTRODUCTION

The MrData system was created to automate, and make GDPR (General Data Protection Regulation)[1] compliant, the handling of human subject medical image data for research at the Max Planck Institute for Biological Cybernetics. The initial imaging system we focused on is a Siemens 9.4 Tesla MRI system. MrData is built based on the iRODS[2] ecosystem.

GDPR compliance informed several aspects of the MrData system architecture. Early in the project, we made a decision to use the Castellum[3] system for human subject recruiting and personal information data management. The alternative, a system where Personal Health Information (PHI)[4] and Personal Identifiable Information (PII) are managed by the same system that manages the scientific data archive, has been done in earlier work. Our objective was a separation of concerns. PHI and PII would be kept in one security domain managed by Castellum, and scientific information would be kept in another security domain managed by MrData. A challenge was ensuring that mixed use metadata, needed in both domains, would have a single root source in Castellum but be available for scientific data archival structure and search.

The MrData system is implemented as a set of micro-services deployed in Docker[5] containers. We use Ansible[6][7] to build and deploy these containers into production on a bare metal Linux server, as well as into Linux virtual machines for testing. The containers are a mix of services we developed in-house using Python, and services such as

iRODS UGM 2022 July 5-8, 2022, Leuven, Belgium

[1] High-field Magnetic Resonance Center, Max Planck Institute for Biological Cybernetics, Tuebingen, Germany
[2] Department of Biomedical Magnetic Resonance, Eberhard Karls University Tuebingen, Tuebingen, Germany

iRODS, Davrods[8], Metalnx[9], and Nginx[10], developed externally. This mix of services is integrated using Docker infrastructure, managed by Ansible, yielding a single cohesive application deployed on a single server.

The remainder of this paper will be structured as follows: we describe our use of Castellum, management of mixed use metadata, the workflow a scientist experiences using MrData, the infrastructure environment, review each micro service component, compare our solution with other options, and conclude with future research directions.

CASTELLUM AND MIXED USE METADATA

Castellum provides a web interface for investigators and administrators to recruit human research subjects. It manages all relevant personal health information and contact information for subjects in a secure, private fashion. Castellum also manages additional configurable subject attributes which may be required for a recruiting search. A user of Castellum first defines a study and then uses Castellum to recruit subjects for that study. A key feature of Castellum is that each subject in a study is given a unique, randomized pseudonym which is used to refer to that subject in all research documents and data. Subjects are only referred to using pseudonyms in the MrData archival system.

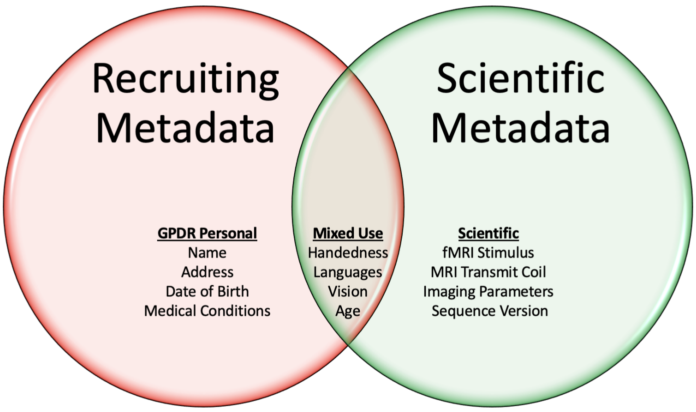

Figure 1. Mixed use metadata.

A challenge to using separate systems for subject recruiting and scientific data archival is mixed use metadata. Mixed use metadata is comprised of subject attributes that are required for both recruiting and for scientific data processing. For MRI related neuroscience, examples of mixed use metadata attributes include handedness, languages spoken, year of birth. These attributes may be used in Castellum to select subjects for a study and also for using, or reusing, scientific data. An important requirement is that any given piece of metadata have a single source and that metadata not be manually transferred between systems. For this reason we worked with the Castellum team to create a one-way API by which the MrData system can extract mixed use metadata to use in archiving and searching scientific data.

Figure 2. Mixed use metadata moves one-way from Castellum to MrData.

The mixed use metadata must be explicitly authorized for export on an attribute by attribute basis. Only attributes that are approved by our data security coordinator can be authorized, and only by an administrative action. For these attributes, the MrData system keeps a full history, even if a subject is deleted from Castellum or a study is deactivated. This is done to maintain the ability to access the archived scientific data which we are obligated to do for a minimum of ten years. It is however possible to remove a given subject from search or user data access as required.

TERMS and DEFINITIONS

This section will review terms and definitions useful in understanding the MrData system.

Study and StudyID

Generally, a study is a project where the scientific investigator will work on a specific research question by running experiments on a group of subjects. Studies may also be defined for calibration or other projects where we wish to archive the resulting data but will not have actual human subjects. A study is defined in Castellum where it is given an StudyID, and may also be given a text descriptor. Castellum can then be used to recruit subjects who will participate in the study.

Subject and Pseudonym

A subject is a person who is recruited to participate in a study using Castellum. Once a subject is recruited for a study, that subject will be given a study-specific pseudonym by Castellum. If the subject participates in multiple studies, the subject will be assigned a different pseudonym in each study. The pseudonym is a randomized, alphanumeric character sequence, and is used to represent the subject everywhere outside of Castellum. We do not record the subject's name, which is PII, at the MRI scanner as would be traditional in a healthcare context.

Experiment and ExperimentID

An experiment is a single data collection session with a subject who undergoes the protocols of the study. In the case of the 9.4T MRI scanner, the subject would meet the MRI operator who would perform a scan that might last a few hours. A scan may produce several types of data that need to be archived. A single subject can participate in several experiments, in one or more studies.

An experiment is identified by an alphanumeric ExperimentID. This is done using MrData Experiment Registration web graphical user interface (GUI). At the web page, a StudyID, subject Pseudonym, investigator userid, and type of experiment, must be entered to acquire an ExperimentID. The ExperimentID is recorded by the MrData system and provided to the investigator who will need to enter it at the MRI scanner console in the "Patient Name" field.

Expanded Data

The data types collected by MrData can be expanded beyond the file types produced directly by the scanner. In the context of an experiment, the scientific investigator can collect arbitrary additional data. Examples are subject questionnaires, experiment parameters, MRI scanner log files, fMRI stimulus parameters, source code, and freely formulated experiment descriptions. MrData provides a directory where investigators can deposit any relevant files. This directory, including its structure, will be archived together with the recorded MRI data to help provide context for later data processing and analysis. The Expanded Data directory also provides a convenient way to preserve digital artifacts needed to facilitate reproducibility.

SCIENTIFIC WORKFLOW

The MrData project's highest objective, after complying with privacy and security requirements, is making the scientific investigator's workflow as convenient, automated, and understandable as possible. Although greater automation may be available for a given use-case, we view it as important that the minimal use-case be as simple as possible. Once the investigator realizes benefit from the minimal use-case, we hope to encourage greater use of coded pipelines, for example via NextFlow[11], in support of reproducible research.

The following is a list of the steps an investigator will follow from defining a study through accessing and processing archived data.

- Use the Castellum web GUI to define a Study and acquire a StudyID.
- Use the Castellum web GUI to recruit Subjects to participate in a Study.
- Use the Castellum web GUI to acquire a Pseudonym for an individual Subject in a Study.
- Use the MrData web GUI to Register an Experiment using the Pseudonym and StudyID, get an ExperimentID.
- At the MRI scanner console, the operator enters their userid and an ExperimentID in the Patient Name field.
- The operator performs the experiment and image data automatically streams into MrData/iRODS.
- The MRI experiment data is then accessed via iRODS (as a network share or via Python API, etc.).

MRDATA SYSTEM

This section will describe the technical details of the MrData system. This includes describing the surrounding infrastructure that enables MrData, as well as the implementation of core MrData application.

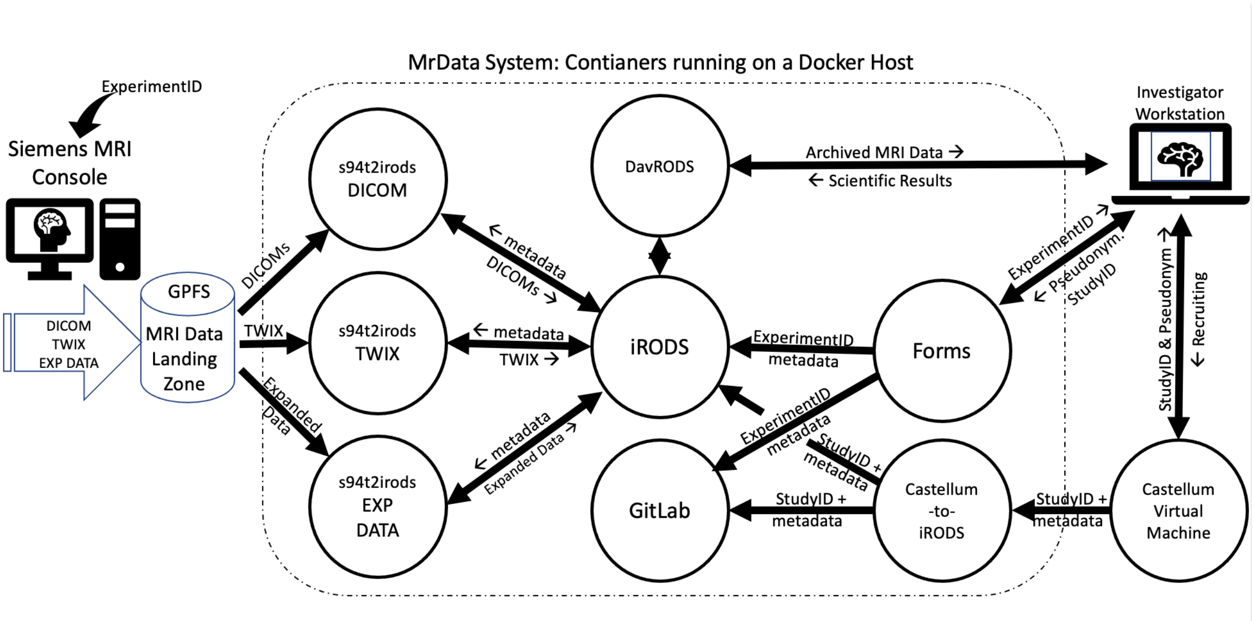

Figure 3. Overview of the MrData environment with core MrData application containers outlined.

Infrastructure Enabling MrData

This section reviews the major pieces of hardware and software infrastructure that enable the MrData system but which are not part of the core application.

Castellum Deployment

Castellum is deployed in a virtual machine maintained by a system administrator. IT staff maintains the highest level of administrative control over Castellum, however scientific investigators have several other, more limited administrative and functional roles. Castellum is the root source of information for study definition, subject assignment to studies, and all mixed use metadata. This information is made available to the investigators using the Castellum web interfaces and to the MrData system via a secure REST[12] API.

Docker Host

The Docker Host is the server on which all the MrData micro-services are run. The server is a bare-metal install of Rocky Linux[13] version 8 with Docker and little else installed. This is a single purpose server and only administrator login is permitted. It mounts an external storage system and has a local 100 terabyte (TB) disk array with a ZFS[14] filesystem. Users access this system via iRODS and HTTPS network protocols. The server is located in the same data center as our compute cluster and storage systems. It is connected to a 25 gigabit Ethernet network via a two port ether-bond. The Docker Host server has the following specs:

```
Memory:              380GB
Disk:                100TB
lModel name:         Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz
Socket(s):           2
Core(s) per socket:  16
Thread(s) per core:  2
L1d cache:           32K
L1i cache:           32K
L2 cache:            1024K
L3 cache:            22528K
```

Siemens 9.4 Tesla MRI System

We have focused on a Siemens Magnetom Plus 9.4 Tesla Magnetic Resonance Imaging scanner (Siemens Healthineers, Erlangen, Germany) running Syngo VE12U software. The 9.4T system produces research data of several types such as anatomical MRI, functional MRI, and raw MRI data that has not been processed into an image. The system can produce hundreds of gigabytes in a single session of a few hours.

The 9.4T MRI scanner produces data stored in two file formats, DICOM[15] format for processed images and "TWIX", which is a Siemens proprietary format for "raw", unprocessed, k-space MRI data.

The DICOM file format is a standard image data format produced by MRI and other medical imagining systems. The Siemens 9.4T operating software allows for setting up an automated DICOM export to an SMB network path, where files will be written to directly after creation. These files are automatically picked up and archived as they are completed.

TWIX files are not normally accessed in a healthcare environment but are required for some MRI research projects. Exporting TWIX files from the scanner interferes with any running scan, so automated exports will only be triggered in the night. However, scanner operators can choose to start an export manually at any time, which can take from minutes up to an hour, depending on the amount of data recorded. Exporting raw data is performed by using the RDS tool provided by the Yarra[16] project. The tool automatically determines which files have yet to be exported and writes them onto a preconfigured network drive. In our case, this is the MrData landing zone defined below.

MrData Landing Zone

The storage where data is stored for import to an archival system is often called a "landing zone", or LZ. MrData uses storage hosted by an IBM Spectrum Scale[17] scalable filesystem, commonly called "GPFS", exported via both Server Message Block (SMB) and Network File System (NFS) protocols. This network file system is used as an LZ for data written from the Siemens console and from the Yarra RDS software and read by MrData. The Siemens console is a Windows system which is allowed a narrow network interface to mount the LZ as a SMB share. The Docker Host mounts the LZ as an NFS share and that mounted filesystem is then made available inside relevant containers as a Docker volume.

As DICOM, TWIX, and Expanded Data files land in the LZ, they are processed by the 9.4T "uploaders" implemented as Docker containers, one uploader for each type of data. The uploaders locate the ExperimentID in the files, extract metadata, index archival information, and put the data into the iRODS archive.

MrData Micro-services

In this section we describe each of the micro-services that make up the MrData system in detail. Each service is deployed as a Docker container. All containers run on a single Docker host.

As mentioned above, we use the Ansible Docker Module to build and start these containers. The reasoning behind this decision warrants elaboration. We first implemented all MrData services as individual virtual machines (VMs), built using Ansible. However, we could not realize enough I/O bandwidth using VMs and tests indicated Docker containers would solve this problem. Since we were targeting a single server environment, Docker Compose[18] seemed like the right container orchestrator. However, as the Docker Compose version came together, we ran into several challenges. First, Docker Compose isn't as flexible as Ansible in configuring the Docker Host, for example ensuring an NFS filesystem is mounted before starting a container that requires it. Second, Docker Compose needs to run on the Docker Host or, at least, with remote access the Docker Host's Docker instance. Ansible more naturally controls a remote host, Docker or otherwise, with ssh and no remote agent. In summary, Ansible enabled using a single "infrastructure as code" tool to configure, build, and deploy the MrData production system on a bare-metal server, while enabling the same capability for test systems in VMs by changing a single file. Ultimately, we are targeting a fully automated continuous integration environment driven by GitLab[19].

Snapshot of the MrData Docker containers running the Docker Host as a production system.

```
IMAGE                  COMMAND                     CREATED        STATUS         NAMES
s94t2irods_image_prod  "/home/mradmin/mr2ir..."    3 weeks ago    Up 3 weeks     s94t2irods_EXP_DATA_prod
s94t2irods_image_prod  "/home/mradmin/mr2ir..."    3 weeks ago    Up 3 weeks     s94t2irods_TWIX_prod
s94t2irods_image_prod  "/home/mradmin/mr2ir..."    3 weeks ago    Up 3 weeks     s94t2irods_DICOM_prod
irods_image_prod       "/docker-entrypoint...."    2 months ago   Up 2 months    irods_prod
cast2irods_image_prod  "/home/mradmin/cast2..."    3 months ago   Up 3 weeks     cast2irods_prod
forms_image_prod       "/home/mradmin/mrfor..."    3 months ago   Up 3 months    forms_prod
nginx:latest           "/docker-entrypoint...."    4 months ago   Up 4 months    nginx
davrods_image_prod     "/bin/sh -c 'dockeri..."    4 months ago   Up 4 months    davrods_prod
```

Castellum to iRODS

The cast2irods service is responsible for fetching exportable Study and Subject metadata from Castellum and making it available in iRODS for the MrData system. This is required to structure the archive and for scientific data search and processing. The service also records a history of the exportable metadata in a private, local GitLab repository.

cast2irods periodically polls a Castellum REST API, retrieving the exportable metadata for all subjects in each active study. This metadata is then organized into a canonical directory structure. The metadata is stored in this directory as YAML[20] files with key value pairs in sorted order. This directory is git diff'd with a reference repository cloned from our local GitLab. The resulting diffs represent changes in the metadata in Castellum relative to what MrData has recorded already. The diffs are merged into the cloned git repository with the exception that we do not delete files so basic archival information about deactivated studies remains available. This allows the git repository to track all exportable metadata and it's history. The modified version of the repository is then committed and pushed to GitLab. These diffs are also used to store the same information in iRODS for use by MrData services such as those that upload and archive data. When a study becomes inactive, its data is no longer exported by Castellum and there will be no further modifications to the corresponding files in GitLab or iRODS.

MrData Forms

The MrData forms service is a small Python Flask[21] application responsible for presenting a web GUI to scientific staff enabling them to register experiments. An experiment is a single session on a scanner with a single subject and may last from tens of minutes to a few hours. A registered experiment is given an ExperimentID.

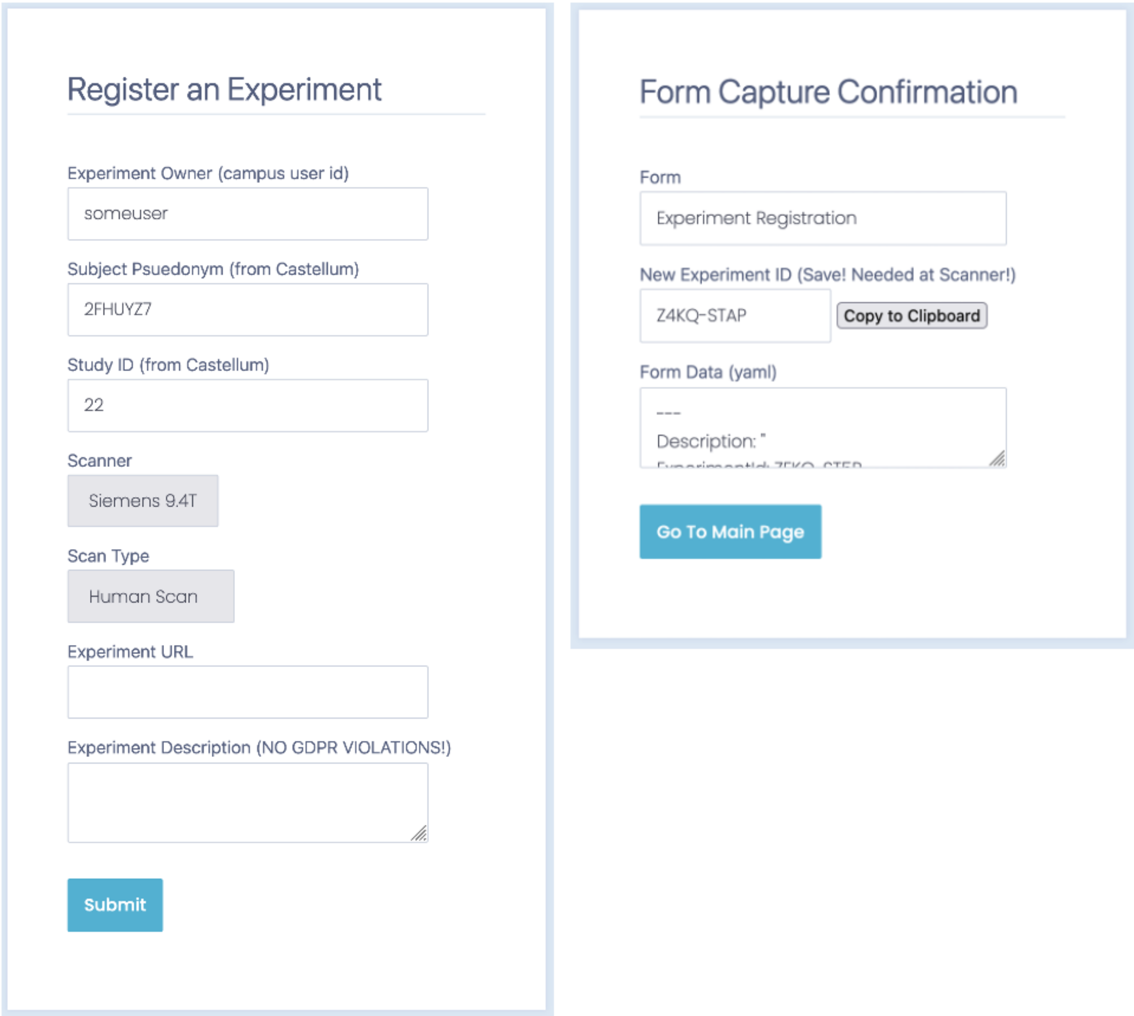

Figure 4. MrData Experiment Registration web page (left) and Confirmation web page (right).

To register an experiment and acquire an ExperimentID, one enters a Castellum StudyID, a Castellum Subject Pseudonym, and a userid into the web GUI. There are drop down tabs to select what kind of experiment will be done, which scanner will be used, etc. Once this input is validated, the Flask application returns a response page with an alphanumeric ExperimentID of 9 characters. The ExperimentID is encoded to enable detection of transcription errors. The ExperimentID with its metadata is committed to GitLab as a YAML file and also stored in iRODS for use by the upload services. The ExperimentID will later be entered into the Patient Name field on the MRI scanner user interface by the operator.

iRODS

iRODS is deployed as a Docker container which extends an existing, published PostgreSQL container. Backing store for the iRODS vault is currently provided by a 100TB ZFS raid array directly connected to the Docker Host. The iRODS container performs regular, incremental backups of the PostgreSQL database to the container backing store, which is in turn regularly backed up to tape. We have configured the iRODS deployment for Transport Layer Security (TLS)[22] only access.

TLS-only iRODS does result in some extra encryption/decryption processing for data exchanges on the Docker internal network, where TLS is arguably not required. Our current understanding is that iRODS cannot force use of encryption on one network interface but allow lack of encryption on another. This ability would be required to enable intra-container communications on the Docker Host to avoid encrypting traffic to iRODS.

Davrods

We deploy the Davrods software from Utrecht University as our main path to providing iRODS data to end-users. We have selected rclone[23] as our primary high bandwidth, data download tool which accesses iRODS through Davrods. Various other WebDAV[25] clients may be used for browsing archived data, including those that present a shared filesystem on the user's workstation.

Metalnx

Metalnx will be added as a browsing method in the near future.

NGINX Reverse Proxy

Nginx is used as a reverse proxy. All external access to MrData web based interfaces, in particular to MrData forms, Davrods, and Metalnx, go through Nginx via HTTPS. Nginx forwards traffic to the individual services on an internal Docker network using mere HTTP.

Siemens 9.4T Data Uploaders

We deploy three containers for uploading data from the Siemens 9.4T MRI scanner. They each handle one of the following categories of data:

- DICOM data – Siemens provided DICOM image files – large number of small files
- TWIX data – Siemens proprietary raw data format files – small number of large files
- Expanded Data – Files not produced directly by the MRI scanner – a user defined file tree replicated to iRODS

Each of the upload services is implemented as a Python program using the Python iRODS Client[24] library. Each service polls a particular area of the MrData LZ filesystem for data uploaded from the MRI scanner. As files flow in to the LZ areas, the processes extract metadata in a content dependent way and recover an ExperimentID. The ExperimentID is used to look up the experiment metadata in iRODS, in particular locating the StudyID. Using the ExperimentID and the StudyID, an iRODS archive path for the files to upload is determined. The headers of the DICOM and TWIX files are also processed to extract other metadata useful for searching and processing the archived data. The data and metadata are then placed in the iRODS archive as collections and objects, with metadata in Attribute/Value/Units (AVUs). The Expanded Data area is uploaded as an unmodified directory hierarchy to iRODS. In the future, we will add the ability to codify metadata to be attached to objects in the Expanded Data.

An MrData experiment data archive iRODS path is formed on the following template:

```
/MRDataZone/home/mrdata/echtdata/studies/<studyID>/experiments/<experimentID>/<DataType>/
```

End-to-end MRI data path through MrData

MrData automates the flow of data from an MRI scanner to the computational processes of the scientific investigators. Here, we follow the path of data from the MRI scanner, through the uploaders, and into the archive.

- The MRI scanner operator enters their userid and the ExperimentID in the Patient Name field on the console.

- The operator proceeds to scan the subject, potentially using stimulus required by the experimental protocol.
- During the scan, data flows from the MRI scanner to the MrData LZ mounted as a SMB network filesystem.
- The uploader processes poll the LZ mounted as an NFS filesystem, and begin processing the data as it arrives.
- The ExperimentID is extracted from each DICOM and/or TWIX file, and used to find the StudyID.
- The StudyID, ExperimentID, the file type, and metadata from file headers are used to form an iRODS path.
- The iRODS archive path for the ExperimentID is created and the DICOM and/or TWIX files put into iRODS.
- Additional metadata from the MRI file headers is added to the iRODS collections and objects as they are stored.
- The investigator (or automation) can then access the iRODS archive data for the experiment in soft real time.
- Viewing the image data may be done while the subject is still in the scanner so the experiment can be adjusted.
- Experiment Expanded Data in the LZ is copied to iRODS after 24 hours, or when the area is marked ”finished”.
- The experiment will be marked as finalized after given amount of time has passed or based on user input.
- Manual and automated process can now search and access all experiment data using normal iRODS methods.
- Investigators will be encouraged to store their computational results into iRODS under their own userid.

RELATED WORK

There are several systems available for medical image data handling which we explored before building MrData. We explored using XNAT [26] and Loris [27] in depth. These are both excellent systems but in each case we found ourselves extending them with significant amounts of local scripting. Further, we were not using a good deal of the available features since they overlapped with Castellum, which satisfied our high level objective of separating the human subject recruiting system from the data archival system. Finally, the modularity of the MrData system and the underlying iRODS ecosystem will ease adding additional capabilities for future data management projects.

CONCLUSION AND FUTURE WORK

We have implemented MrData, an iRODS based data management system that automates the archival of data streaming from a research MRI system. We carefully integrated with a GDPR compliant human subject recruiting system, Castellum. To avoid manual transcription of subject mixed use metadata, we worked with the Castellum team to establish a REST API to access this data via automation. The MrData system is deployed using Ansible and Docker containers in a micro-services architecture making it extensible as well as testable.

Future work includes:

- Collecting DICOM files sequence groups and archiving a tar and NiFTI[28] file for each sequence group.
- Extending MrData automated archival to additional MRI systems at our institute and beyond.
- Full automation of the continuous integration pipeline using GitLab, Ansible, and Virtual Machine targets.
- Where it makes sense, explore creating open source repositories for the MrData project.
- Trigger automatic workflow processing on the acquired data.

ACKNOWLEDGEMENTS

The authors want to thank Dr. Timo Göttel, from the Max Planck Institute for Human Development in Berlin, and the Castellum team for supporting the integration of Castellum in this project. We further want to thank Dr. Jonas Bause, from the Max Planck Institute for Biological Cybernetics in Tübingen, for his contribution to the design of the MrData project. Funding by the Deutsche Forschungsgemeinschaft (DFG German Research Foundation) under the Reinhart Koselleck Programme (DFG SCHE 658/12) and by the European Research Council (ERC Advanced Grant No 834940, SpreadMRI) is gratefully acknowledged.

References

[1] General Data Protection Regulation (GDPR). 2018. General Data Protection Regulation (GDPR) – Final text neatly arranged. [online] Available at: <https://gdpr-info.eu/> [Accessed 29 June 2022].

[2] RODS: Open Source Data Management Software. `https://irods.org` [Accessed 2 July 2022]

[3] Castellum A Privacy-Compliant Subject Management for Scientific Research `https://www.mpib-berlin.mpg.de/research-data/castellum` [Accessed 1 July 2022]

[4] Protected Health Information vs Personal Identifiable Information `https://www.accountablehq.com/post/pii-vs-phi` [Accessed 2 July 2022]

[5] Merkel, D. (2014). Docker: lightweight linux containers for consistent development and deployment. Linux Journal ()2014)

[6] Ansible is an IT automation tool. It can configure systems, deploy software, and orchestrate more advanced IT tasks. `https://docs.ansible.com/ansible/latest/index.html` [Accessed 2 July 2022]

[7] Ansible Community Modules and plugins for working with Docker `https://galaxy.ansible.com/community/docker` [Accessed 2 July 2022]

[8] Davrods - An Apache WebDAV interface to iRODS `https://github.com/UtrechtUniversity/davrods` [Accessed 1 July 2022]

[9] Metalnx is a web application designed to work alongside iRODS. It is a graphical user interface and serves as a client that authenticates to an existing iRODS Zone. `https://github.com/irods-contrib/metalnx-web` [Accessed 3 July 2022]

[10] Nginx `https://nginx.org/en/` [Accessed 1 July 2022]

[11] Di Tommaso, P., Chatzou, M., Floden, E. W., Barja, P. P., Palumbo, E., Notredame, C.. Nextflow enables reproducible computational workflows. Nature biotechnology (2017)

[12] REST is an acronym for REpresentational State Transfer and an architectural style for distributed hypermedia systems `https://restfulapi.net/` [Accessed 2 July 2022]

[13] Rocky Linux is an open-source enterprise operating system designed to be 100% bug-for-bug compatible with Red Hat Enterprise Linux® `https://rockylinux.org/` [Accessed 2 July 2022]

[14] OpenZFS is an open-source storage platform. `https://openzfs.org/wiki/Main_Page` [Accessed 13 July 2022]

[15] DICOM® — Digital Imaging and Communications in Medicine — is the international standard for medical images and related information. `https://www.dicomstandard.org` [Accessed 1 July 2022]

[16] Wiggins,R., ,Block, K. T. Yarra Framework – Open-Source Toolkit for Clinical-Translational MRI Research. [online] `https://s3.amazonaws.com/download.yarraframework.com/doc/ISMRM19_Whitepaper.pdf` [Accessed 30 June 2022]

[17] IBM Spectrum Scale `https://www.ibm.com/products/spectrum-scale` [Accessed 2 July 2022]

[18] Compose is a tool for defining and running multi-container Docker applications. https://docs.docker.com/compose [Accessed 2 July 2022]

[19] DevOps platform build around git https://about.gitlab.com/ [Accessed 2 July 2022]

[20] YAML Ain't Markup Language™ https://yaml.org/ [Accessed 2 July 2022]

[21] Flask is a lightweight WSGI web application framework. https://flask.palletsprojects.com/en/2.1.x/ [Accessed 2 July 2022]

[22] The Transport Layer Security (TLS) Protocol Version 1.3 https://datatracker.ietf.org/doc/html/rfc8446 [Accessed 2 July 2022]

[23] Rclone syncs your files to cloud storage https://rclone.org/ [Accessed 2 July 2022]

[24] Python iRODS Client (PRC) https://github.com/irods/python-irodsclient [Accessed July 1 2022]

[25] WebDAV Protocol (rfc4918). https://datatracker.ietf.org/doc/html/rfc4918

[26] Marcus, D. S., Olsen, T. R., Ramaratnam, M., Buckner, R. L. The Extensible Neuroimaging Archive Toolkit: an informatics platform for managing, exploring, and sharing neuroimaging data. Neuroinformatics (2007)

[27] Das, S., Zijdenbos, A. P., Harlap, J., Vins, D., Evans, A. C. LORIS: a web-based data management system for multi-center studies. Frontiers in neuroinformatics (2012)

[28] Neuroimaging Informatics Technology Initiative https://nifti.nimh.nih.gov/ [Accessed 13 July 2022]