2018 IEEE 14th International Conference on e-Science

Curation of Image Data for Medical Research

Lasse Wollatz*, Mark Scott*, Steven J. Johnston*, Peter M. Lackie†, and Simon J. Cox*
*Faculty of Engineering and Physical Sciences, University of Southampton, Southampton, UK
†Faculty of Medicine, University of Southampton, Southampton, UK
{L.Wollatz, Mark.Scott, S.J.Johnston, P.M.Lackie, S.J.Cox}@soton.ac.uk

***Abstract*—Microfocus X-ray computed tomography (µCT) and 3D microscopy scanning create scientific data in the form images. These images are each several tens of gigabytes in size. E-Scientists in medicine require a user-friendly way of storing the data and related metadata and accessing it. Existing management systems allow computer scientists to create automatic image workflows through the use of application programming interfaces (APIs) but do not offer an easy alternative for users less familiar with programming. We present a new approach to the management and curation of biomedical image data and related metadata. Our system, Mata, uses a network file share to give users direct access to their data and also provides access to metadata. Mata also enables a variety of visualization options as required by e-Scientists in medicine.**

***Index Terms*—data curation, image management, metadata, medical research**

I. Introduction

Data curation is an essential part of medical research. New imaging techniques and the fast-changing nature of research lead to quick changes in the image analysis tools required by e-Scientists in medicine.

Current systems, such as the Open Microscopy Environment Remote Object (OMERO) platform [1] and the Bio-Image Semantic Query User Environment (BisQue) [2], rely on custom plug-ins and internal software to allow users to interact with the data stored by them. Additional steps are needed for the integration of new software into the image management workflow. Either the users export data to their local file store, to then import the data into the new software, or a plug-in is programmed to allow the software to retrieve data through the systems' application programming interfaces (APIs).

Many researchers and technicians are working to ensure that e-Scientists in medicine can conduct research using the latest advances in computer science. The medical research itself has to be undertaken by medical specialists. It is vital that e-Scientists in medicine do not have to consult a computer scientist to test and experiment with the vast number of software tools and algorithms included in those tools. We consider easy access to data to be more critical than programmatic access in the form of an API [3].

Heterogeneous Data Centre (HDC) [4] is a tool that manages users' data without imposing such restrictions. A network share enables users to access their data directly through their operating system's (OS) file manager application. A file-system monitor keeps track of the data and synchronizes it with the metadata in a database.

In this paper, we adapt HDC for the use of biomedical research images. The resulting software is called Mata[1] (a combination of medicine and data) and provides additional functionalities for working with metadata as well as previewing capabilities targeted to e-Scientists in medicine.

This paper makes the following contributions:

- It proposes an alternative approach to the curation of research data in medicine;
- It shows how HDC can be configured to medical research and the steps taken to create Mata;
- It designs distinct use cases to show how Mata can benefit medical researchers.

The rest of this paper is structured as follows. An overview of HDC and reasons for choosing it are given in section II. Section III discusses Mata and how it adapts HDC to medical research. Section IV presents different use cases and shows how Mata can benefit medical researchers. We give a conclusion in section V.

II. Motivation

In [5], we tested a website-based system for image management. The upload of several tens of gigabytes of data through a web browser is slow and unreliable due to browser limitations [6], [7]. The upload speed is mainly dependent on the internet performance, which cannot be controlled by the system. There are two common solutions to implement the upload of large files. Either a webbrowser application is used, or a separate software connects to both the local file store and the remote one. We used Plupload[2] in [5] to overcome the issue of file-size when uploading large datasets, but it was not able to increase reliability compared to that offered by software-based file upload as used by other management systems [1], [2], [8]. The alternative approach, as used by [1], [2], [8], is a software running on the e-Scientists computer that interacts with an API on the management server. The downside of a system-specific software for image upload and download is that it reduces user-friendliness since it requires users to learn the operation of the image management software. Providing access to data only via an API introduces an additional step to connect other software. Any new software will require a computer scientist to integrate

Funding: This work was supported by the Engineering and Physical Sciences Research Council [grant number 1511465]

[1]Published as: doi:10.5258/SOTON/D0430
[2]Official site: http://www.plupload.com/ (last accessed 22/05/2018)

978-1-5386-9156-4/18/$31.00 ©2018 IEEE
DOI 10.1109/eScience.2018.00026

it with the management system via an API. The APIs used by existing management systems are not standardized. The existing clinical standard for data management and transfer, Digital Imaging and Communications in Medicine (DICOM), relies on detailed patient information and cannot be used in a research environment, where the patient needs to be anonymous [9], [10].

The reason for choosing HDC [4] was that it gives users direct access to the server file store. Using a network file share gives users a familiar environment for accessing their data while ensuring robust file transfer and storage methods that evolve with the file store.

HDC has a file-system monitor, which will be referred to as the "file watcher" for the rest of this paper. Its task is to compare the state of the file system to that of the database storing the metadata and modify each of them to reach a consistent state. The file watcher is triggered by file-system events to analyze a particular folder and checks every folder in the system at regular intervals. All folders on a defined level are considered datasets in the database. Information about the datasets' files is stored in hidden sub-directories of the data folder. With the help of this information, HDC can determine and differentiate between additions, removals, renames and copies on a folder and file level [4], [11]. Another functionality provided by the file watcher is the automatic execution of plug-ins if specific file-types are added. This allows extending the capabilities of the file watcher further by interfacing with other external software [4].

III. The Image Management System

In this section, we will explain how we created Mata by modifying HDC. HDC implements a variety of features for the management of data, most of which were deemed useful for medical research. For Mata, we retained the Windows server to integrate with companies' Windows networks, as well as the Structured Query Language (SQL) database used by HDC for metadata storage. The layout of the database was copied since it already implemented the versatility required for this project.

Other parts of HDC were considered unnecessary for this particular implementation. Specifically, the Microsoft SharePoint[3] interface was removed and replaced with a more lightweight PHP (PHP: Hypertext Preprocessor) website. SharePoint integrates well with Microsoft environments since it is part of the Microsoft Office suite. The number of third-party applications for SharePoint is limited and integration of custom plug-ins, though possible, requires programming them in .NET or finding another way of implementing them. The original interface was built with SharePoint 2010 and .NET 3.5. That version of SharePoint is out of date, and we found it to be slow. We did not use many of the SharePoint features that were implemented with the HDC front-end, either. Instead of reimplementing the SharePoint front-end in a more recent release, we created a PHP site for Mata that was easier to prototype and is fully open source. The use of PHP also allowed easy reimplementation of essential features, such as the editing of user permissions and dataset metadata.

[3]Official site: www.sharepoint.com/ (last accessed: 22/05/2018)

The rest of this section will address various parts of the management system and the corresponding solutions we have developed. We will briefly discuss feature re-implementations of HDC and then introduce any additions made to the system for each part. We distinguish between the following modules of the management system: editing of metadata, visualization and searching of metadata, visualization of datasets, and system security.

A. Editing of Metadata

Metadata here refers to key-value pairs [4], which we refer to as tags in order to match users' expectations. Tags allow users to assign metadata to datasets. Tags can help maintain information about data that is important to medical researchers but cannot be stored as part of the image file. For example, a tag can describe the species seen in the image, which may be a human, a sheep, or a zebrafish. As with HDC, editors of the dataset are allowed to sort tags in a particular order and create hierarchies for them. These hierarchies allow users to structure the tags according to their needs.

A challenge faced when allowing users to edit the metadata is the lack of consistency, which makes a comparison of tags more challenging [12]. We used two solutions to overcome this problem.

One solution we adapted from HDC to avoid users using different terms to describe the same thing involved the ability to import tags from parents. It assumes that users will tend to use this functionality to save the effort of copying common tags. When importing tags from a parent, all tags from that parent, where the key has not yet been assigned any value for the dataset, are copied over to the dataset. This approach ensures consistency among tags between different datasets, users, and sites. Not copying values of keys which have already been defined avoids overwriting data or having duplicate keys.

In addition, Mata implements a dictionary to guide users into using the same tags for describing the same features [13]. The advantage is that synonyms can be avoided. In this project, a dictionary has been implemented for guidance to avoid typographical errors and reduce the number of grammatical variations of the same term. The system suggests existing tags to users based on the user's input. Tags that have been used more often have been ranked higher amongst the suggested tags.

B. Visualization and Searching of Metadata

One of the main requirements of a data management system is the ability to look up data in an efficient manner. Our solution provides three different options for finding datasets: a basic search, tag clouds, and relation networks.

1) Basic Search: One functionality re-implemented is a basic search function which looks for the exact phrase in the datasets title, description and tags. It returns any matches found in the order of the number of matches of the search term

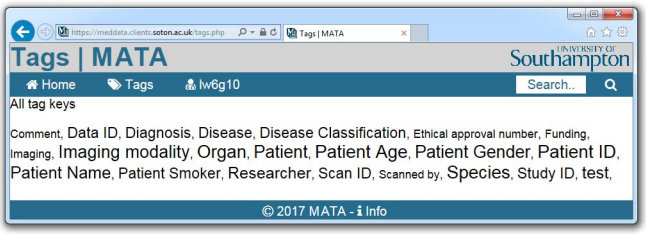

Fig. 1. Screenshot showing the rectangular tag cloud for a small number of tags. Entries are in alphabetical order. More common tags have a larger font-size than less common tags.

found in the database. This search function can be expanded to be more sophisticated if needed [14]–[17].

2) Tag Cloud: A page with all of the different tags was created, to enable users to browse the tags. Tags were arranged in an elementary rectangular tag cloud as shown in Fig. 1, to aid users in finding important tags quicker. The implementation of more sophisticated tag clouds is possible and has been discussed in many papers [18]–[20]. We implemented our own tag cloud here and will outline it in the following paragraphs.

In order to prioritize tags in a tag cloud, they are represented in different font sizes, colours, or layouts. Tags that appear more often are larger, such that specific, unusual tags or typographical errors that only appear for a few datasets are smaller and less noticeable.

We assume that more frequently used tags are more likely to be of interest to a common user and therefore should be larger and easier to find. Meanwhile, to help users searching for specific tags, we minimized the variation of font size between tags. We used Zipf's Law [21] to map the occurrence of specific tags to a linear distribution. We also maintained an alphabetical ordering of tags to help users find specific tags [19].

3) Relation Network: When editing a dataset, users can define parent datasets, meaning datasets which the current dataset was derived from [4]. Examples are a segmentation of a scan or follow-up data, which can be linked to the original image dataset. By default, a dataset is expected to have precisely one parent unless it is an original scan, in which case it does not have any parent. In some cases, several images may be combined into one. An example is the correlation of a single photon emission computed tomography (SPECT) image and a computed tomography (CT) image. Areas of abnormality found by a SPECT can be located relative to internal organs, detected by the CT [22]. In such cases, more than one parent may exist. Having the origin of an image defined ensures that they can be traced back. A network graph (see Fig. 2) is generated, using recursion to search for all related datasets in the database and vis.js[4] for display. In this graph, each node represents a dataset and arrows connecting the nodes indicate their relation, as seen in Fig. 2. This graph helps users finding other datasets derived from the same scan with just a few clicks.

This view has also been combined with the tags explained in the previous section. As a result, it can display a hierarchy of tags for a folksonomy [12], [23] by assuming that two tags are equal, if their keys, their values, and child tags are all the same. The order of the children is disregarded.

As seen in Fig. 2, the datasets "*Sample_CT002*" and "*test-Merge of twins CT*" are tagged with the same patient, as all the child tags related to "*Patient*" are the same. However, they are different from the patient of "*Sample_CT001*", as they only share a single attribute.

Fig. 2 also illustrates the limitation of this strict definition of equality, since the "*Patient*" in "*Sample_CT001*" and "*Sample_CT001-Copy*" are considered different even though they describe the same person. Not all child tags of "*Patient*" are the same, because the researcher in one dataset listed the species as part of the "*Patient*" information. The gravitational physics model behind vis.js solves this issue by ensuring that such tags, which are almost equal, will remain close together since they share many children.

The rather strict definition avoids false positives. In medicine, critical metadata, such as the patient, have multiple identifiers. This reduces the chance of tags falsely being identified as equal.

The comparison of tags and their children, as mentioned before, occurs over several levels. As a result, the tag "*Imaging*" may include subcategories, for example, "*Scan settings*", containing a list of settings such as the exposure rate and projections. In the web interface, every tag (for example a single patient) links to a page that shows all datasets containing that tag.

C. Visualization of Datasets

Microfocus X-ray computed tomography (µCT) data and resulting processed data need to be available to medical researchers in a secure, accessible, and meaningful way. One of the critical demands identified was fast previewing of images. It allows users to decide which images are worth analyzing before retrieving the full image from a remote storage onto their systems for processing. In order to display files over the web, we used the Multiresolution Computed Tomography Viewer (MCTV) [5] and Viewstl and integrated them as shown in Fig. 3.

1) Adding a 2D Volumetric Image Viewer: MCTV allows the viewing of extra-large 3D CT files in a web browser without requiring plug-ins [5]. It can also present 2D images as a special case of a 3D image stack. On the presentation layer, it can be directly embedded into Mata without additional installations. On the back end, a script had to be implemented to automatically create previews and metadata required by MCTV. The HDC file watcher already had a functionality to allow a script to be executed when a dataset changes. This was used by Mata to trigger a script creating a preview for a dataset each time it changes, as shown in Fig. 3.

[4]Accessible at http://visjs.org (last accessed 22/05/2018)

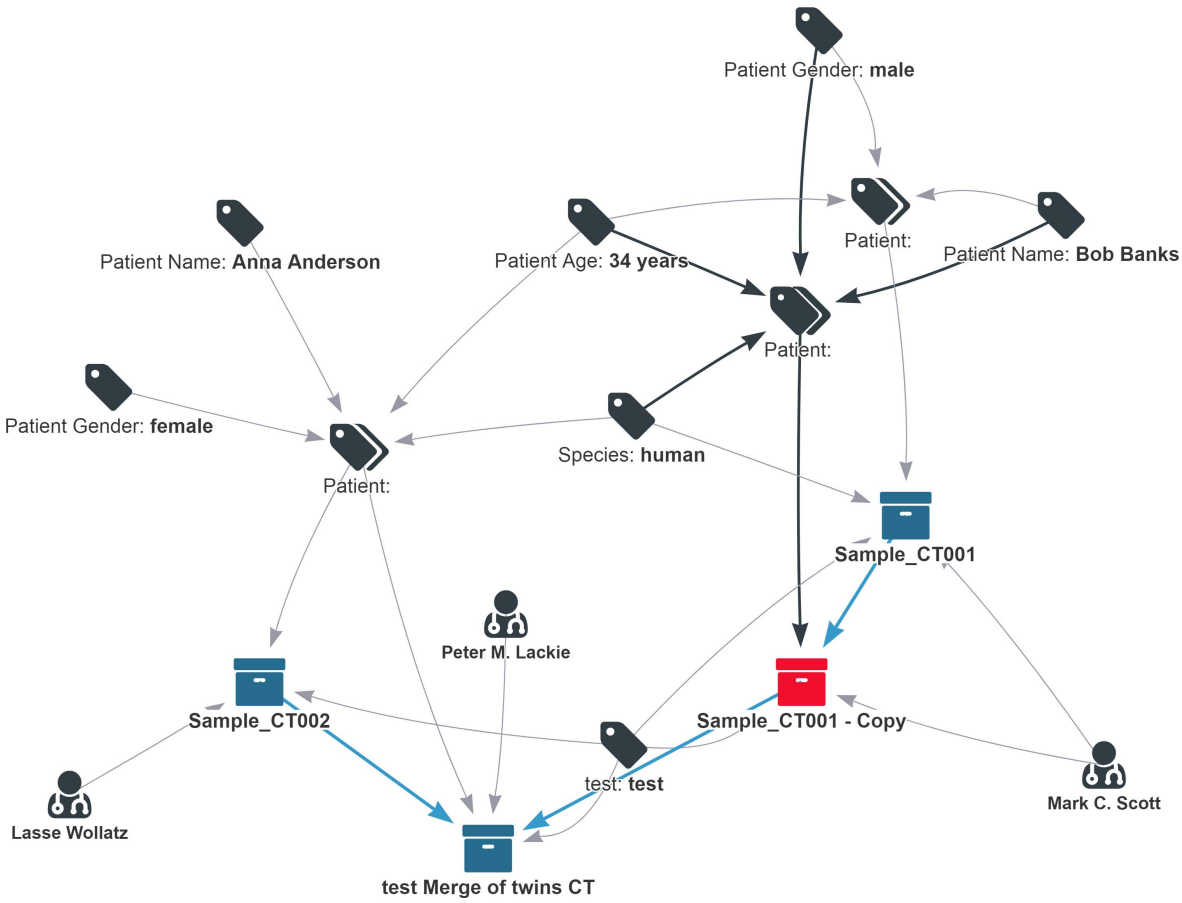

Fig. 2. Screenshot of a dataset relation network graph using fictional datasets for demonstration purposes. It allows to find all datasets (blue) related to the currently selected dataset (red). We invented patient names and details to make it easier to follow the example.

The process of creating tiles requires a large number of read and write operations and is therefore relatively slow. This means that it is likely that a dataset is being changed while a preview of the old version is still being created. For example, when a dataset comprising a stack of 2D image files is first uploaded, the file watcher triggers the plug-in for each file upload. When the first file is being uploaded, the file watcher starts the tile-creating script (the tiler). By the time the second file reaches the Mata file store, the tiler has not yet finished, but the file watcher starts another instance of it. For a 3D image with 2000 slices, this would mean that millions of duplicate tiles would be created.

Therefore the script needed to cope with processing large images. The approach we took was to split the task of creating tiles into several subtasks which can then be canceled remotely. To implement this, a local queue was set up to enable local deployment, as shown in Fig. 4. Celery version 3.1.25 was used for the queue as it is the latest version compatible with Windows. Scripts interacting with the queue were implemented in Python.

HDC executes a script that sends a job request to "readdir" to the queue. The "readdir" function first checks if the command has been executed for the same directory before and revoke any outstanding jobs. As this version of Celery does not support accessing information about tasks in the queue, it was necessary to save the job identifications (IDs) in a separate file. The Python code will then read any metadata files it finds and attempts extracting remaining information such as the global minimum and maximum pixel values from the image files. For extracting the minimum and maximum pixel values, only a selection of the image files is read (or a random set of points taken from a raw file) and evaluated to reduce the overall time it takes to execute the script. Finally, one tiling-job request is created for each image slice, and the job ID is written to a file. The tiler then splits the image into smaller tiles and saves the tiles to disk [5]. The tiler jobs are started with a lower priority than that of the job reading the directory, to ensure that revoking jobs gets priority over working on other jobs. This reduces the overall queue length and avoids images being tiled more often than necessary.

It is important to make images accessible in a way meaningful to the medical researchers who are used to 2D images. With the script for creating the tiles in place and having it linked as shown in Figs. 3 and 4, MCTV is able to read the

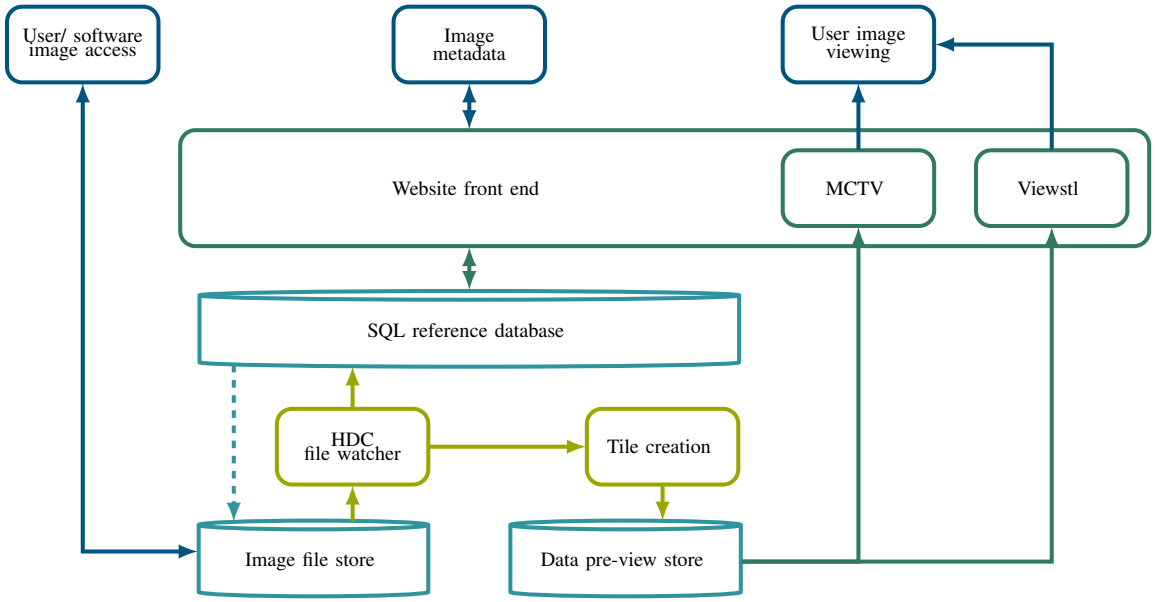

Fig. 3. The new proposed system using HDC. The image file store, HDC file watcher and SQL reference database are taken from the original HDC implementation. The website front-end, the implementation of visualization tools and related data pre-processing scripts are newly created for this paper.

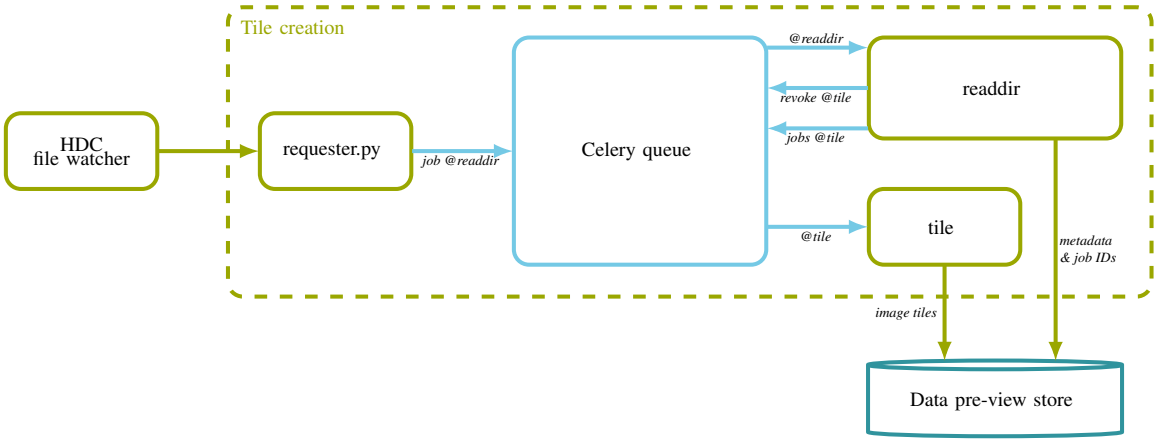

Fig. 4. Schematic of the queue implemented for the tiler in Fig. 3

tiles from the server file store and present them to the user as shown in Fig. 5. Users' access to the tiles is restricted through the permissions on the network file share.

2) Adding a 3D Surface Image Viewer: The viewer presented in the previous section enables the viewing of volumetric images in the form of 3D image stacks and 2D images. It cannot deal with 3D viewing or with the viewing of surface files. The data which are inherently 3D also requires a 3D visualization in order to enable the analysis of 3D structures not represented in a 2D viewer. It was therefore decided to add Viewstl[5] to display stereo-lithography (STL) and object (OBJ) files.

As with MCTV, Viewstl does not require special plug-ins but works with JavaScript. Files are loaded into client-side memory and are not processed by any third party.

In order to use the encrypted version of the Hypertext Transfer Protocol (HTTP) for all resources, a local copy of Viewstl was created and modified to use the encrypted protocol.

To integrate the viewer, an HDC file-watcher plug-in has been created, which is executed for STL and OBJ files. It calls a Python script which uses the same queue as the MCTV tiler but only creates a file with the name of the first STL file in

[5]Official site: http://www.viewstl.com/ (last accessed 22/05/2018)

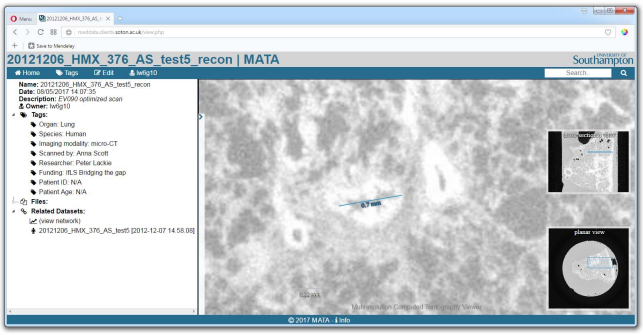

Fig. 5. Screenshot of a dataset containing a TIF-stack viewed in Mata. The panel on the left shows the tags and related datasets. On the right a slice of a µCT scan of a lung biopsy can be seen with a planar and cross-sectional view as explained in [5].

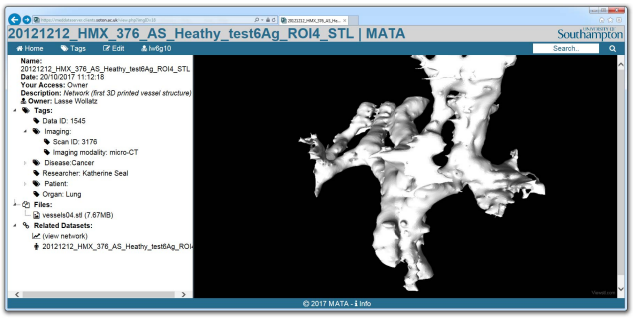

Fig. 6. Screenshot of a dataset containing an STL file, viewed in Mata. The layout is the same as in Fig. 5 but MCTV is replaced with Viewstl showing the surface of a vascular structure from a lung biopsy.

the datasets folder. On the web page for viewing a dataset, PHP is used to check if the dataset contains a 3D surface file and if the user has read permission. If this is the case, Viewstl is loaded and is passed the file name, which is then displayed as shown in Fig. 6.

Through the use of Viewstl, the ability to view 3D surface files over the web via Mata was demonstrated. The secure implementation of an external viewer also shows that, due to its modularity, Mata can be extended very easily.

D. System Security

The importance of security for medical databases has been highlighted through recent incidents [24], [25]. For the security of a service, various functional components have to be considered and evaluated. This is particularly important when dealing with sensitive medical data. Components that could be vulnerable include the user authentication and the database. Most of these components have been discussed in [4], [11]. The specific permissions in Mata were kept the same as in the case of HDC. Everyone can list folder content but access to file content is limited to certain users. The owner (the creator) of the dataset can decide whom to give access to, and people with access can preview and edit the data apart from their permission in a browser. HDC also forwards the permissions as read and write permissions to the server file store and the connected network file share.The file access on the network file share is secured through Windows Authentication. Applications, such as the PHP interpreter, running on the server responsible for the website need elevated permissions in order to access the database and other data. Further, the website authentication has to be re-implemented with PHP.

When webpage content is generated, it is essential that confidentiality of the data is maintained. Accordingly, when rendering the content of a webpage, PHP needs to know who is allowed to view or edit individual content and who is accessing the website. HDC synchronizes folder and file permissions between the database and network file share. As PHP has access to the database, it can look up if a user has permission to access specific data. Additionally, Windows Server and Microsoft Internet Information Service (IIS) enable the use of the same authentication for websites and file access. This enables single sign-on (SSO), so that users only need one account and password to access both, the network file share and the website of Mata as well as other systems connected to the domain. PHP can read the session's details from the server and therefore knows who is accessing the webpage. The user identification is the same as the one HDC synchronized between the servers' file store and database. As a result, PHP knows the users' permissions.

The primary reasons for choosing SQL as the main database type was the well-established security it offers in comparison to not only SQL (NoSQL) databases [26]–[28]. The use of well-established systems, like Windows Server, Windows authentication, SQL and PHP enables system administrators to set up a secure implementation throughout the whole data management system.

E. Summary

In this section, we introduced the medical image management system Mata. We discussed the underlying software and presented various parts of the management system. This included the editing of metadata, visualization and searching of metadata, visualization of datasets and system security. Metadata is created by users and the editing of it is guided through the use of a dictionary. A tag cloud and dataset relation networks allow additional navigation options apart from a basic search. We also showed the implementation of a 2D and a 3D viewer, and explained how we secured the system. A comparison of HDC and Mata is given in Table I.

IV. Use Cases

Though it is applicable to any sort of data, Mata was tested using a scenario modelled after a research project on lung diseases, at the University of Southampton. This project evaluates the use of µCT scanners for the analysis of biopsies in a non-destructive way.

This section presents the use cases for this scenario to show how the tools discussed can integrate with an existing image workflow. The use cases are illustrated in Fig. 7.

TABLE I
COMPARISON OF HDC AND MATA.

	HDC	Mata
Operating system	Windows Server	Windows Server
Server	Microsoft IIS	Microsoft IIS
Database	SQL Server	SQL Server
Middleware	Sharepoint 2010/ .NET 3.5	PHP 7.1
Data access	Network file share	Network file share
Synchronisation	File watcher	File watcher (from HDC)
SSO authentication	✓	✓ (partially reimplemented)
Metadata editing	✓	✓ (reimplemented)
Metadata import	✓	✓ (reimplemented)
Metadata suggestion	✗	✓ (ranked dictionary)
Basic search	✓	✓ (reimplemented)
Tag cloud	✗	✓
List of direct relatives	✓	✓
Relation network	✗	✓ (graph of all relatives)
CT slice viewer	✗ (but 2D viewer)	✓ (MCTV)
3D surface viewer	✗	✓ (Viewstl)

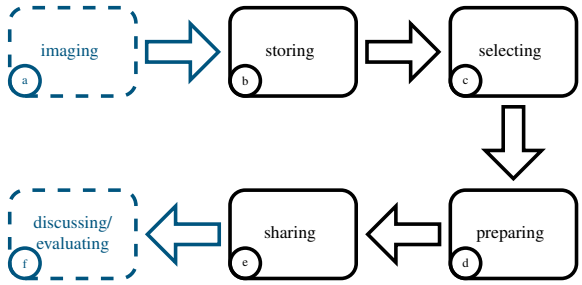

Fig. 7. Simplified overview of the use cases. Dashed parts are important to the project and the image workflow but not part of the presented system. Each of the steps (a) to (f) are described in more detail in sections IV-A-IV-E.

As part of this scenario, a large number of images needs to be created using a newly developed µCT scanner. Matching microscopy images are retrieved, and images are processed to enhance the contrast. µCT and microscopy images are then aligned in 2D and 3D using point-based registration based on the scan conditions. Matching image areas are presented to experts to assess both biological content and image quality. Images of microscopy slices and µCT slices are then assessed for technical quality or biological data content by image experts or histopathologists.

A. Storing New Scans

Use case: **An image has been scanned on the µCT, and the scan operator wants to store the image. (Fig. 7 a, b)**
By using Mata developed in section III, it is possible to store the images generated by the newly developed medical µCT scanner by simply copying the image files into the network file share.

The file watcher detects the new files and creates an entry in the database. It also triggers the code for creating a preview of the image. The operator can add file permissions and metadata as appropriate. In many practical scenarios, it will be beneficial to add an HDC plug-in which reads out metadata from files and adds it automatically as described in [4].

If metadata are created in a consistent way by the scanner, a plug-in for the file watcher can be written, to automatically add this metadata to the database as soon as the image gets uploaded. DICOM devices can also be supported by adding a DICOM receiver. An administrator is required to set permissions for any received data since DICOM does not transfer user information during file exchange. This is due to DICOM being patient centred.

B. Finding Relevant Images

Use case: **A medical researcher tries to find and get hold of images relevant to the project. For that, they want to find images of a lung taken by µCT and preview them to confirm that they align with the projects requirements. (Fig. 7 c)**
The researcher has various options of finding relevant images, as discussed in section III-B. A basic search for a relevant keyword or browsing through the tags can help obtaining a list of relevant datasets. If images are not in the desired format or processing stage, it is possible to find related datasets through the network graph. With the help of MCTV and stlviewer (section III-C), it is possible to preview images before downloading them, even if their size of several tens of GB exceeds the machine's Random Access Memory (RAM). Download of images can be conveniently achieved through the file explorer.

C. Finding Related Images

Use case: **The researcher wants to retrieve the microscopy scan of the same sample. (Fig. 7 c)**
The researcher has various options to find a related dataset. The two images are likely to share a tag, like the sample identification number. The researcher can therefore find the data by searching for this tag. A more direct approach is to look at the relationship network graph. Any linked dataset can be found and accessed in this way.

D. Processing Images

Use case: **The researcher found a relevant image and tries to process it for presentation to the experts. (Fig. 7 d)**
After relevant images have been found, they can be processed using any suitable processing software.

Since the images are accessible via network file share, any processing software the researcher wants to use can read the files directly from the remote storage.

After the processing, the new data are stored in a new folder on the network file share to create a new dataset in Mata and share them with other, specified researchers. Newly found metadata can also be added to Mata, and a custom HDC plug-in for importing this metadata can be developed if suitable.

E. Sharing Images

Use case: **A set of images is to be shared with an expert for evaluation. (Fig. 7 e, f)**

In order to share a dataset with an expert, the web link to that dataset can be sent to the other person. Provided that the other person is a user of Mata, they can access the dataset as soon as the owner has added them to the list of viewers/editors of the dataset through the web interface.

F. Summary

In this section, we introduced a possible scenario for the use of an image management system in medical research. We showed how Mata can be used as a management system for this scenario by discussing different use cases based on the scenario. We demonstrated that Mata can tackle the various steps involved for image management in medical research. All of the steps can be achieved without programming scripts or plug-ins to link external software to the system.

V. Conclusion

It is essential to provide access in an user-friendly way, to enable e-Scientists in medicine to access data without requiring programming skills. Current image management systems are restrictive in the way they incorporate other software.

We took a data curation system that gives users direct access to their data through a network file share, eliminating the need for special software to access the data. It also simplifies the use of new software for image processing and visualisation with the existing system, without requiring custom plug-ins to load image data via a non-standard API. Since most software can read files from a file store, no additional programming is required to allow them to use image data stored in our management system.

We added a website front-end to the system, which allows medical researchers to fulfill common use cases. This includes the management of metadata through the web-interface as well as different ways of visualizing and searching metadata and datasets. We showed how additional data visualization tools can be implemented. We used well established standards all along the way to ensure that the system is easy to transfer to and secure in a different environment.

Future work involves the addition of an API to allow scripts to access metadata through code. It would be beneficial for the biomedical image community to develop a standard for research images similar to DICOM for clinical, medical images. Declaring an existing API, like the one used by OMERO or BisQue, as a standard may well be an option for this. We will also show how this management system fits into the bigger workflow of images in medical research.

Acknowledgment

We would like to thank Anna Scott for providing her dataset as an example figure for this paper. All data supporting this study are openly available from the University of Southampton repository at https://doi.org/10.5258/SOTON/D0430.

References

[1] C. Allan, J. M. Burel, J. Moore, C. Blackburn, M. Linkert *et al.*, "OMERO: flexible, model-driven data management for experimental biology," *Nat. Methods*, vol. 9, pp. 245–253, 2012.

[2] K. Kvilekval, D. Fedorov, B. Obara, A. Singh, and B. S. Manjunath, "BisQue: a platform for bioimage analysis and management," *Bioinformatics*, vol. 26, no. 4, pp. 544–552, Feb 2010.

[3] J. Dryndos, A. S. S. Kazi, D. Langenberg, H. Löh, and R. Stark, "Collaborative virtual engineering for smes: Technical architecture," in *IEEE Int. Technol. Manag. Conf.* Lisbon, Portugal: IEEE, Jun 2008, pp. 1–8.

[4] M. Scott, R. P. Boardman, P. A. Reed, and S. J. Cox, "Managing heterogeneous datasets," *Inf. Syst.*, vol. 44, pp. 34–53, 2014.

[5] L. Wollatz, S. J. Cox, and S. J. Johnston, "Web-based manipulation of multiresolution micro-CT images," in *Proc. 11th Int. Conf. eScience*. Munich, Bavaria, Germany: IEEE, Sep 2015, pp. 308–311.

[6] E. Lawrence, "File upload and download limits," Mar 2011, accessed on 06/12/2017. [Online]. Available: https://blogs.msdn.microsoft.com/ieinternals/2011/03/10/file-upload-and-download-limits/

[7] Ephox, "Plupload: Multi-runtime file-uploader frequently asked questions," 2017, accessed on 06/12/2017. [Online]. Available: http://www.plupload.com/docs/v2/Frequently-Asked-Questions#when-to-use-chunking-and-when-not

[8] D. S. Marcus, T. R. Olsen, M. Ramaratnam, and R. L. Buckner, "The extensible neuroimaging archive toolkit," *Neuroinformatics*, vol. 5, no. 1, pp. 11–33, Mar 2007.

[9] NEMA, *Digital Imaging and Communications in Medicine (DICOM) Standard*, National Electrical Manufacturers Association Standard 2018a, 2018. [Online]. Available: ftp://medical.nema.org/medical/dicom/2018a/

[10] M. J. Warnock, C. Toland, D. Evans, B. Wallace, and P. Nagy, "Benefits of using the dcm4che DICOM archive," *J. Digit. Imaging*, vol. 20, no. Suppl. 1, pp. 125–129, Nov 2007.

[11] M. Scott, "Research data management," Thesis, University of Southampton, May 2014.

[12] S. A. Golder and B. A. Huberman, "Usage patterns of collaborative tagging systems," *J. Inf. Sci.*, vol. 32, no. 2, pp. 198–208, 2006.

[13] A. Noruzi, "Folksonomies - why do we need controlled vocabulary?" *Webology*, vol. 4, no. 2, Jun 2007.

[14] S. Dessloch and N. Mattos, "Integrating SQL databases with content-specific search engines," in *Proc. 23rd VLDB Conf.*, Athens, Greece, 1997, pp. 528–536.

[15] W. Kießling and G. Köstler, "Preference SQL: design, implementation, experiences," in *Proc. 28th Int. Conf. Very Large Data Bases (VLDB '02)*. Hong Kong, China: VLDB Endowment, 2002, pp. 990–1001.

[16] Y. Tsuruoka, J. Tsujii, and S. Ananiadou, "FACTA: a text search engine for finding associated biomedical concepts," *Bioinformatics*, vol. 24, no. 21, pp. 2559–2560, Nov 2008.

[17] S. Brin and L. Page, "Reprint of: The anatomy of a large-scale hypertextual web search engine," *Comput. Networks*, vol. 56, no. 18, pp. 3825–3833, Dec 2012.

[18] B. Y.-L. Kuo, T. Hentrich, B. M. Good, and M. D. Wilkinson, "Tag clouds for summarizing web search results," in *Proc. 16th Int. Conf. World Wide Web (WWW '07)*. New York, New York, USA: Association for Computing Machinery (ACM), 2007, pp. 1203–1204.

[19] M. J. Halvey and M. T. Keane, "An assessment of tag presentation techniques," in *Proc. 16th Int. Conf. World Wide Web (WWW '07)*. New York, New York, USA: Association for Computing Machinery (ACM), 2007, pp. 1313–1314.

[20] C. Seifert, B. Kump, W. Kienreich, G. Granitzer, and M. Granitzer, "On the beauty and usability of tag clouds," in *Proc. 12th Int. Conf. Inf. Vis. (IV 2008)*. London, UK: IEEE, Jul 2008, pp. 17–25.

[21] G. K. Zipf, *Human Behaviour and the Principle of Least-Effort*. Cambridge, MA: Addison-Wesley, 1949.

[22] G. Soo and T. Cain, "SPECT-CT scan," Jul 2017, accessed on 25/05/2018. [Online]. Available: https://www.insideradiology.com.au/spect-ct-scan/

[23] A. Mathes, "Folksonomies - cooperative classification and communication through shared metadata," 2004, accessed on 25/05/2018. [Online]. Available: http://www.adammathes.com/academic/computer-mediated-communication/folksonomies.html

[24] TrapX Research Labs, "MEDJACK.2 hospitals under siege," TrapX Research Labs, Tech. Rep., 2016, accessed on 25/05/2018. [Online]. Available: https://media.scmagazine.com/documents/242/trapx_medjack2_60312.pdf

[25] T. Fox-Brewster, "Medical devices hit by ransomware for the first time in US hospitals," May 2017, accessed on 04/07/2017. [Online]. Available: https://www.forbes.com/sites/thomasbrewster/2017/05/17/wannacry-ransomware-hit-real-medical-devices/#741ce643425c

[26] L. Okman, N. Gal-Oz, Y. Gonen, E. Gudes, and J. Abramov, "Security issues in NoSQL databases," in *Proc. Int. Jt. Conf. IEEE Trust. ICESS-11/FCST-11*, IEEE. IEEE, 2011, pp. 541–547.

[27] A. Ron, A. Shulman-Peleg, and A. Puzanov, "Analysis and mitigation of NoSQL injections," *IEEE Secur. Priv.*, vol. 14, no. 2, pp. 30–39, 2016.

[28] D. Chahal, L. Kharb, and M. Gupta, "Challenges and security issues of NoSQL databases," *Int. J. Sci. Res. Comput. Sci. Eng. Inf. Technol.*, vol. 2, no. 5, pp. 976–982, 2017.