MyTardis Ingestion

Release 0.1

Chris Seal

Dec 10, 2021

CONTENTS

The University of Auckland (UoA) has instantiated a customised version of MyTardis as a repository to hold data generated by scientific instruments. As part of the project developing this capability, we have created a set of ingestion scripts that can be used to automate the process of adding data to the repository.

Our focus in developing these scripts has been to ensure the greatest possible flexibility so we can integrate as seamlessly as possible into existing workflows that are used by facilities.

CHAPTER ONE

SCOPE

This document is intended to describe in detail the processes developed at UoA, by which a new instrument is onboarded into the system. As part of this discussion, the ingestion scripts need tailoring to fit the new instrument in order to gain the greatest possible benefit from automation that can be achieved.

To facilitate the changes necessary to tailor the ingestion scripts, the core Python classes developed are described in more detail and an example ingestion script is included.

CHAPTER

TWO

ONBOARDING A NEW INSTRUMENT

The process of onboarding a new instrument has developed organically as work has continued customising and implementing the UoA specific version of MyTardis. The codebase used at UoA has diverged significantly from the upstream but has, at the time of writing this document, been substantially refactored to be compatible with upstream.

During this refactoring process most of the customisation applied at UoA have been moved into 'apps', or microservices. The current list of 'apps' that are used (currently not available upstream but future pull requests should make them more widely available for the MyTardis community) are:

- project
- instrument_profile
- identifiers
- search
- facility_profile

The ingestion scripts developed employ these apps and are therefore incompatible with 'vanilla' MyTardis. Where possible specific code that relies on one of these apps has been identified.

2.1 Onboarding Steps

There are several stages to the onboarding process and it is, by its nature, iterative. Broadly speaking the steps involved consist of the following, which are discussed in more detail

- *Preliminary discussion with researchers and facility managers*
- Co-developing the ingestion process[1]
  - *Gathering necessary data for creating the instrument (and facility if necessary) and metadata schema*
  - *Co-developing an ingestion strategy with researchers and facility managers*
  - *Customising the ingestion scripts to fit with the strategy devised*
- *Testing the ingestion pipeline*
- *Migrating to a live service*

[1] These processes are iterative and happen concurrently in a typical onboarding.

2.2 Preliminary Discussions

Most researchers and facility managers are not aware of MyTardis and what the instrument data repository offers. In order to ensure that both the facility managers and the researchers are fully engaged with the co-design process it is important to provide them with context. This is typically done by meeting, either individually or as a group and going through the following:

- What are we trying to achieve with our repository?
  - Protect data by ensuring that it is backed up in the data centre.
  - Ensure that a set of raw data is collected as ‘close’ to the instrument as possible, allowing for repetition of different analyses for reproducible research.
  - Provide search functionality allowing researchers to find their own, and UoA shared data, quickly and easily.
  - Provide secured access to data that is in need of protection through robust access control.
  - Automate, as much as is possible, the hand-over of data from a facility to researchers
- What is MyTardis and how does it help us achieve these aims?
  - Normally a live demonstration with sample users/data.
- What are our next steps?
  - Introduce the need for minimum metadata.
  - Discuss the concept of metadata schema and what additional data would be useful to collate.
  - Discuss co-design process and agree to the approach that will be taken.
  - Check that there is still interest in pursuing the use of the repository.

After holding preliminary discussions, if there is still interest in pursuing the use of MyTardis, the co-design process begins.

2.3 Gathering Required Data

When onboarding a new instrument we have made the choice to implement instrument persistent identifiers (PIDs). Current best practice is to use the *schema <https://github.com/rdawg-pidinst/schema/blob/master/schema.rst>* defined by the Research Data Alliance (RDA) working group on persistent identifiers for instruments (PIDInst). The *instrument_profile* app developed for the UoA instance of MyTardis provides a database model that captures this schema, while retaining the flexibility for facilities to define their own metadata.

The data required to mint an instrument PID is summarised here:

- **Landing Page**: A URL that the identifier resolves to.
- **Name**: The instrument name
- **Owner**: The institution(s) responsible for the management of the instrument
  - **Owner Name**: The full name of the owner
- **Manufacturer**: The manufacturer or developer of the instrument
  - **Manufacturer Name**: The full name of the manufacturer

Other recommended metadata fields defined in the schema include:

  - **Owner**:
    - **Owner Contact**: Contact email for the instrument owner

  - **Owner Identifier**: Persistent identifier (PID) for the instrument owner
  - **Owner Identifier Type**: The type of PID included.
- **Manufacturer**:
  - **Manufacturer Identifier** PID for the manufacturer
  - **Manufacturer Identifier Type**: The type of PID included
- **Model**: Name or model of the instrument as attributed by the manufacturer
  - **Model Name**: Full name of the Model
  - **Model Identifier**: PID for the model
  - **Model Identifier Type**: The type of PID included
- **Description**: Technical description of the instrument and its capabilities
- **Instrument Type**: Classification of the type of instrument
- **Measured Variable**: What the instrument measures or observes
- **Date**: Key dates include commissioning/decommissioning, calibration etc.

  -**Date Type**: What the date represents
- **Related Identifier**: PIDs that are related to the instrument. For example a complex instrument might contain sensors that can be considered to be instruments in their own right. These could have PIDInst minted for them and they would list the other sensors in the instrument as related identifiers
  - **Related Identifier Type**: The type of PID included.
  - **Relation Type**: Description of the relationship
- **Alternate Identifier**: Other Identifiers that the instrument has
  - **Alternate Identifier Type**: The type of identifier used as an alternate

In addition to the instrument metadata if the facility is also being onboarded then we need metadata associated with this. Currently there are no internationally recognised PIDs for facility or department scale institutions that lie under a larger research institution, such as a university. Work in this area is ongoing and there is some indication that the Research Organisation Registry (ROR) PID will be extended to include sub-units. As such, there is no agreed upon minimum metadata, other than the facility name. We have developed a skeleton facility profile to hold this data once consensus about what the minimum metadata standards are has been reached.

MyTardis has been developed with a very flexible metadata model that allows for nearly complete customisation of the metadata fields stored and presented by the system. This flexibility comes with the additional overhead, however, of needing to establish metadata schema for the different objects stored in the database. During the data gathering stage of the onboarding process, we work with researchers and facility managers to discover:

1. What metadata exists for the data generated?
   - What data can be automatically obtained from the instrument or the output files from the instrument?
2. What metadata should be stored with the data?[2]
   - What is useful for researchers to be able to search on?
   - Are there any examples of international best practice that we can leverage?

[2] Since metadata is trivially small compared to the data attached, and in the absence of a good reason not to, a 'greedy' approach has been decided upon, where we collect more, rather than less, metadata.

3. What is the priority of the metadata, in other words, what is the best order to display it in?[3]
4. At what level in the object hierarchy of MyTardis should the metadata sit?

Once this information has been gathered, appropriate metadata schema can be prepared within MyTardis and their identifier recorded.

2.4 Determining an Ingestion Strategy

Concurrently with information gathering, onboarding a new instrument requires the development of an ingestion strategy. At it's heart such a strategy needs to unambiguously assign metadata to the data files that are generated by an instrument.

This process is complicated by the presence of an object hierarchy within MyTardis and metadata needs to be sorted into it's appropriate place within the object hierarchy.

With the **project** 'app' installed the object hierarchy in MyTardis is as follows:

1. Project.
2. Experiment (It is often convenient to think of this as being a sample on which multiple assays, from different instruments, are being made).
3. Dataset (This represents a suite of measurement(s) taken using a single instrument for a related sample or samples).
4. Datafiles generated by the instrument (It is envisaged that little metadata will sit at this level of the object hierarchy, but it does provide a location for quality assurance data or similar to sit).

For each level of the hierarchy the minimum metadata is similar, but has different names due to historical reasons.

At the Project level the minimum metadata is:

- **name**: The project name
- **description**: A short project description
- **identifier**: A unique project identifier, can be multiple in use but all need to be globally unique within MyTardis. To help with global uniqueness we are recommending that facilities prepend their own identifiers with a 3 letter code representing the facility. This provides some protection against namespace collision.
- **principal_investigator**: A username for the lead researcher in the project. This user will get admin access at all levels of the project and it's child objects. It should be noted that the UoA version of MyTardis authenticates against Active Directory and the API may need reworking for OAuth authentication.
- **schema**: A schema name as defined within MyTardis for the Project level schema. This will include the metadata fields and short names associated with them.

For Experiments the minimum metadata required for them to be created in MyTardis is:

- **title**: The experiment name
- **identifier**: A unique experiment identifier. See Project identifier field for notes.
- **description**: A short description of the experiment.
- **project**: A **list** of project identifiers (i.e. the **identifier** field from the project object in question) for the parent projects. *Note:* Many-To-Many relationships are established between the Project and the Experiment objects, allowing for the ready re-use of data, as appropriate.

[3] This has yet to be implemented/undertaken. Current developments with one of our research facilities has yielded the potential to include literally hundreds of metadata fields and it is important, with such a number of possible items of metadata, that some for of sorting be carried out when defining the associated schema.

- **schema**: A schema name as defined within MyTardis for the Experiment level schema. This will include the metadata fields and short names associated with them.

For Datasets the minimum metatdata is:

- **description**: The dataset name (see experiment **title** above)
- **identifier**: A unique dataset identifier, could also be Dataset DOIs. See Project::**identifer** for notes.
- **experiments**: A **list** of experiment identifiers associated with the Experiment **identifier**. *Note:* Many-To-Many relationships are established between the Experiment and the Dataset objects, allowing for the ready re-use of data, as appropriate.
- **instrument_id**: A unique identifier to the instrument that the data was generated on.
- **schema**: A schema name as defined within MyTardis for the Dataset level schema. This will include the metadata fields and short names associated with them.

Datafiles have the following minimum metadata requirements:

- **filename**: The file name of the data file to be ingested
- **md5sum**: The MD5 checksum of the original data file
- **storage_box**: The MyTardis storage box defined for the facility
- **local_path**: The full path to the local instance of the data file to be ingested
- **remote_path**: The relative path to the remote instance of the data file for the purposes of maintaining the local directory structure. This is in place to accommodate analysis packages that expect a specific directory structure.
- **full_path**: The full path to the remote instance of the data file (normally constructed from the **remote_path** by the parser.
- **schema**: A schema name as defined within MyTardis for the Datafile level schema. This will include the metadata fields and short names associated with them.

While the above represents the minimum metadata required to generate objects within MyTardis, these do not necessarily need to be explicitly defined and there are a wide range of approaches that can be taken in order to match metadata to data. Working with researchers and facility managers to integrate the ingestion process into their existing workflow is the goal of the co-design process. We have found, however, that the flexibility associated with the wide range of potential solutions can lead to a state of '*analysis paralysis*', which has led to the development of a default ingestion strategy.

The default ingestion strategy detailed here, uses a directory structure to implicitly set the parent/child relationships between Projects -> Experiments -> Datasets -> Datafiles and a metadata file defined within each directory that contains the metadata for the associated object. This is limited to a One-To-Many relationship and where a Many-To-Many relationship is desired, this will need to be explicitly defined in the metadata file.

The ingestion scripts also automate the generation of the file path and checksum metadata fields for datafiles, unless explicitly defined in the associated metadata file.

This strategy is intended as an example as to how the object metadata may be captured in a robust manner without excessively burdening researchers and facility staff with administration.

2.5 Developing Customised Ingestion Scripts

Once an ingestion strategy has been defined (even if it is only an interim one) a set of custom ingestion scripts are needed to process the metadata, in whatever form it takes, into a set of Python Dictionaries that can be handed off to the core ingestion scripts.

It is worth explaining the approach that has been taken with the ingestion scripts developed. As development of MyTardis continues, it is likely that there will be incremental improvements to the backend. Given the wide application that we expect to see for the instrument data repository across UoA, we did not want to be in a situation where an update to the backend API required reconstruction of all of the various ingestion scripts that were in use. As a result we have chosen to separate the scripts into two areas, a backend interface and a suite of customised ingestion scripts that are developed on a facility basis.

The ingestion scripts are able to process different data types from different facilities into a standardised format that is presented to the backend interface. This interface acts as a wrapper around the various API calls that are needed to create objects within MyTardis and accepts a standardised format, thus if, at some point in the future, the API calls used to generate objects within MyTardis are updated (such as the move from a REST API to a GraphQL API), the existing ingestion scripts will still work without any changes necessary, as they will continue to present the standardised metadata format.

Similarly, as new facilities are brought onboard, there is no need to develop scripts that make the API calls to MyTardis, instead, these scripts need to prepare the metadata into a standard format, which also should allow for code reuse where similar ingestion strategies are employed.

The standard metadata dictionaries for ingestion are described in detail here

2.6 Testing the Ingestion Pipeline

The UoA instance of MyTardis consists of three separate instances:

- A development instance
- A test instance
- A live instance

Once a set of ingestion scripts has been developed and an ingestion strategy employed, the process needs end-to-end testing using data that is as similar as possible to that to be ingested. Testing of ingestion scripts should be undertaken on the test instance and the primary copy of the data will not be removed from its existing storage until the ingestion process moves into ‘production’. It’s also important to ensure that the researchers and facility managers are aware that, during the testing phase of ingestion, we will need to verify that everything is progressing as expected, which in turn means that we need access to the data. For sensitive data this may not be possible and in these cases we will need to work with the researchers to get data that is similar to the data generated by the instrument, with similar metadata for the purposes of testing.

2.7 Going Live

When all parties are happy that the ingestion process is working as expected, then a phased transition into production can be implemented. In the initial stages of productionisation of a new ingestion workflow, it is recommended that the primary data be moved to a 'holding' directory rather than deleted. This provides an opportunity to ensure that there are no issues associated with scaling up the ingestion process.

During this transition period researchers and facility managers should be encouraged to migrate their workflows over to using MyTardis and there will likely need to be additional training provided to ensure that they are familiar with how to use MyTardis.

Once a majority of users have integrated MyTardis into their workflows, and provided there are no scale-up issues, discussions should be held with the facility managers about removing the primary copy of the data stored locally, once a verfied copy of the data have been ingested into MyTardis. At this point the transition to production can be considered complete and ingestion into MyTardis business as usual.

- Functionality provided by the Ingestion Classes
  - Class overview
  - How it all links together
  - Minimum metadata

CHAPTER

THREE

INDICES AND TABLES

- genindex
- modindex
- search