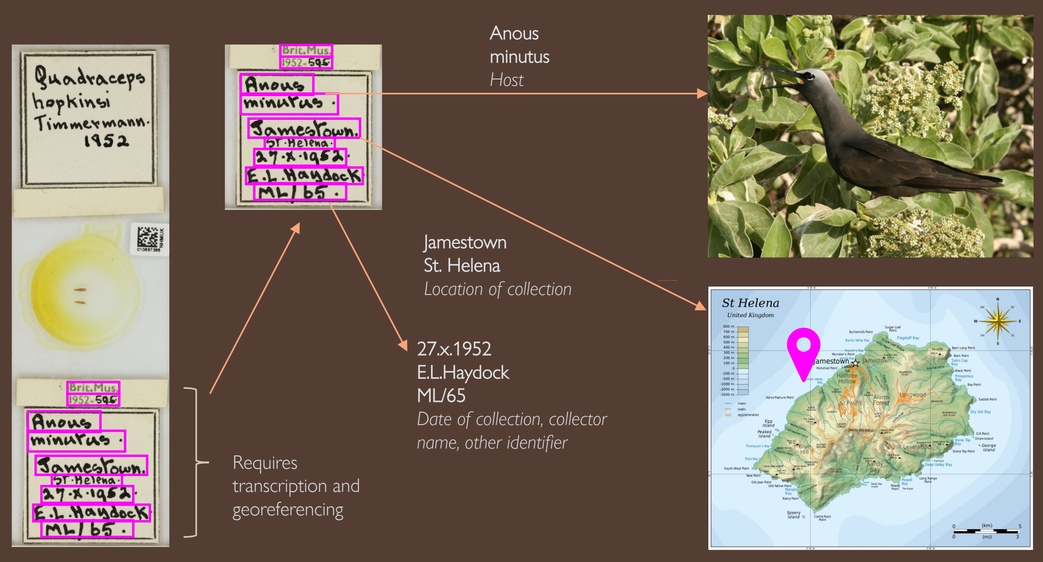
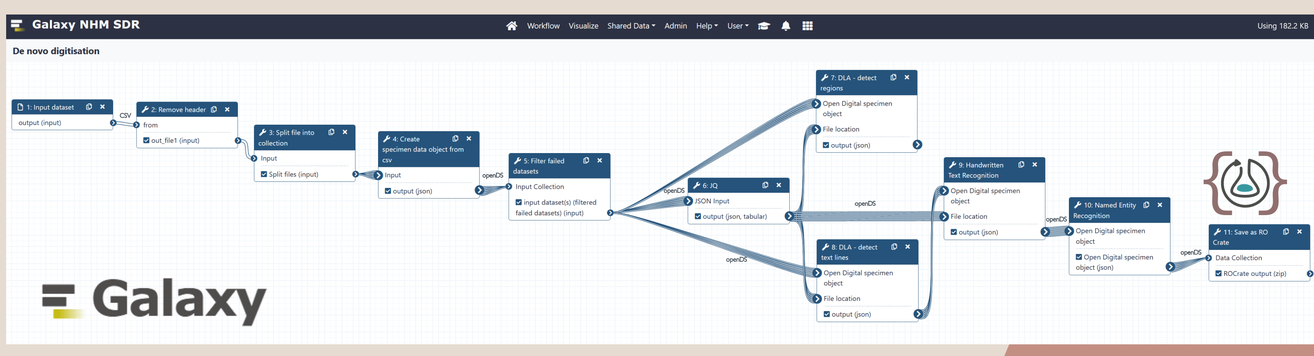
# Incrementally building FAIR Digital Objects with Specimen Data Refinery workflows

Oliver Woolland, Paul Brack, Stian Soiland-Reyes, Ben Scott, Laurence Livermore

The University of Manchester, Manchester, United Kingdom
Informatics Institute, University of Amsterdam, Amsterdam, Netherlands
The Natural History Museum, London, United Kingdom

## Creating FDOs from digitized specimen sheets




Galaxy workflow *De Novo digitization* shows propagation of partial Open Digital Specimen FDOs between individual canonical workflow building blocks.
First steps process a CSV file to create the initial openDS, where referenced images are analysed to detect text lines which are OCRed and then recognized as named entities.
Bands indicate flow of **collections of openDS**, processed concurrently by each step.
The final step bundles the collection of openDS FDOs as JSON files into an RO-Crate FDO.

Figure: https://doi.org/10.3897/rio.8.e94349.figure1
Workflow: https://doi.org/10.48546/workflowhub.workflow.373.1


Visualization of an **Open Digital Specimen** (openDS) FDO within Galaxy.
Text within regions of interest (specimen labels) have been digitized and incrementally added to the partial FDO.

## Incrementally building FDOs

In the *De novo* use case shown above, the workflow steps are exchanging **partial FDOs** – openDS objects which are not fully completed and not yet assigned persistent identifiers.

**openDS schemas** are still in development, therefore SDR uses a more flexible **JSON schema** where only the initial metadata (populated from CSV) are required. Each workflow step validates the partial FDO before passing it to the underlying command line tool.

Although workflow steps exchange openDS objects, they cannot be combined in any order. For instance, named entity recognition requires digitised text in the FDO. We can consider these intermediate steps as **sub-profiles of an FDO Type**.

Unlike hierarchical subclasses, these FDO profiles are more like **ducktyping**.

For instance a text detection step may only require the regions key, but semantically there is no requirement for an **"OpenDSWithText"** to be a subclass of **"OpenDSWithRegion"**, as text also can be transcribed manually without regions.

Similarly, we found that some steps can be executed in parallel, but this requires **merging of partial FDOs**. This can be achieved by combining JSON queries and JSON Schemas, but indicates that it may be more beneficial to have FDO fragments as separate objects. Adding explicit shims for *openDS fragments* would however complicate workflows.

Several of our tools process the **referenced images**, currently https URLs in openDS. We added a **caching layer** to avoid repeated image downloading, coupled with local file-paths wiring in the workflow. A similar challenge occurs if accessing image data using DOIP, which unlike HTTP, has no built-in caching mechanisms.

## Packaging workflow run as RO-Crate

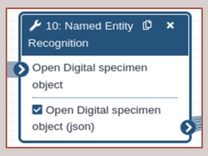
Galaxy is developing support for importing and exporting **Workflow Run Crates**, a profile of **RO-Crate** to captures execution history of a workflow, including its definition and intermediate data (FDO poster by *De Geest*). SDR is adopting this support to combine openDS FDOs with workflow provenance.

Our workflow returns results as a ZIP file of openDS objects. End-users should also get copies of the referenced images and generated visualisations, along with workflow execution metadata, without needing to dereference FDOs. We are now embedding the Galaxy workflow history before the final step, so that this result can be an enriched and self-described RO-Crate FDO.

## Challenges of incremental FDOs

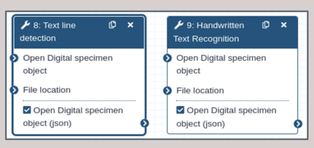
Obscure the data type tools need / produce.

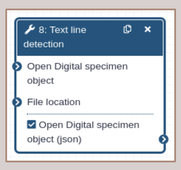
*What does this tool need?*
*What does it output?*




Obscure the ordering of tools.

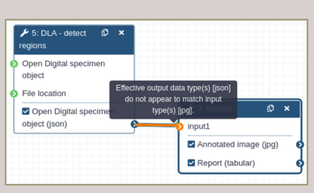
*Which tool goes where?*
*Do they work in any order?*




Create **conflicts** with Workflow Manager philosophies.

*If FDO points to file on disc, how can WFM control data?*
Workaround: flow of data is **outside** workflow.

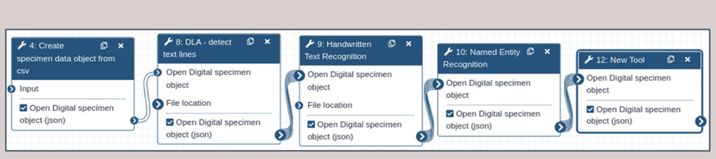



Make it **challenging** to integrate third party tools

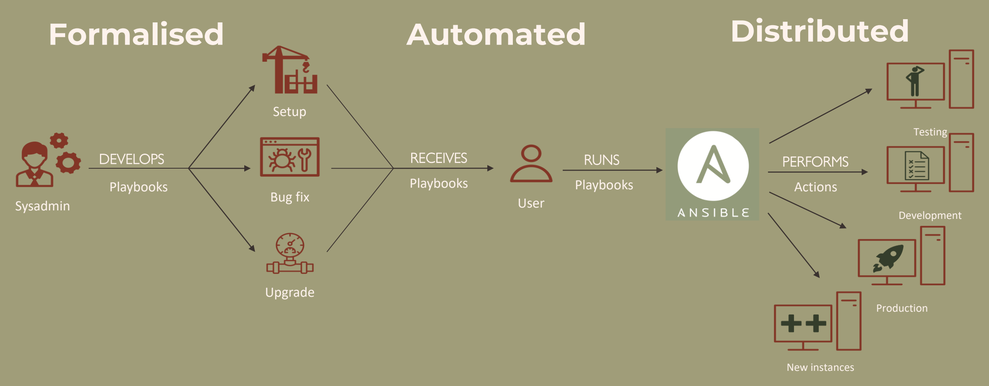
How do I connect inputs?
Should I create a wrapper?

Composite FDOs can make **debugging challenging.**

*How do I feed testing data to New Tool?*
*What if New Tool only needs trivial input to test?*




## Ansible Deployment of Galaxy server




Processes are documented
Processes are repeatable
Processes can be shared
Processes can be run on multiple targets

Funding acknowledgements:

Synthesys+ https://doi.org/10.3030/823827
BioExcel-2 https://doi.org/10.3030/823830
DiSSCo Prepare https://doi.org/10.3030/871043
EOSC-Life https://doi.org/10.3030/824087



RIO abstract: https://doi.org/10.3897/rio.8.e94349
Poster: https://doi.org/10.5281/zenodo.7224964