ISA API: An open platform for interoperable life science experimental metadata

David Johnson[1,2,†], Keeva Cochrane[3], Robert P. Davey[4], Anthony Etuk[4], Alejandra Gonzalez-Beltran[1,5], Kenneth Haug[3,6], Massimiliano Izzo[1], Martin Larralde[7], Thomas N. Lawson[8], Alice Minotto[4], Pablo Moreno[3], Venkata Chandrasekhar Nainala[3], Claire O'Donovan[3], Luca Pireddu[9], Pierrick Roger[10], Felix Shaw[4], Christoph Steinbeck[11], Ralf J. M. Weber[8,12], Susanna-Assunta Sansone[1*,†], Philippe Rocca-Serra[1*,†]

[1]Oxford e-Research Centre, Department of Engineering Science, University of Oxford, 7 Keble Road, OX1 3QG, Oxford, United Kingdom, [2]Department of Informatics and Media, Uppsala University, Box 513, 751 20 Uppsala, Sweden, [3]European Molecular Biology Laboratory, European Bioinformatics Institute (EMBL-EBI), Wellcome Genome Campus, Hinxton, Cambridge CB10 1SD, United Kingdom, [4]Earlham Institute, Norwich Research Park, Norwich NR4 7UZ, United Kingdom, [5]Science and Technology Facilities Council, Scientific Computing Department, Rutherford Appleton Laboratory, Harwell Campus, Didcot, OX11 0QX, United Kingdom, [6]Genome Research Limited, Wellcome Sanger Institute, Wellcome Trust Genome Campus, Hinxton, Saffron Walden CB10 1RQ, United Kingdom, [7]Structural and Computational Biology Unit, European Molecular Biology Laboratory (EMBL), Meyerhofstr. 1, 69117 Heidelberg, Germany, [8]School of Biosciences, University of Birmingham, Edgbaston, Birmingham, B15 2TT, United Kingdom, [9]Distributed Computing Group, CRS4: Center for Advanced Studies, Research & Development in Sardinia, Pula, Italy, [10]CEA, LIST, Laboratory for Data Analysis and Systems' Intelligence, MetaboHUB, Gif-Sur-Yvette F-91191, France, [11]Cheminformatics and Computational Metabolomics, Institute for Analytical Chemistry, Lessingstr. 8, 07743 Jena, Germany, [12]Phenome Centre Birmingham, University of Birmingham, Edgbaston, Birmingham, B15 2TT, United Kingdom

*Corresponding authors: susanna-assunta.sansone@oerc.ox.ac.uk, philippe.rocca-serra@oerc.ox.ac.uk

†Contributed equally

Abstract

Background: The Investigation/Study/Assay (ISA) Metadata Framework is an established and widely used set of open-source community specifications and software tools for enabling discovery, exchange and publication of metadata from experiments in the life sciences. The original ISA software suite provided a set of user-facing Java tools for creating and manipulating the information structured in ISA-Tab – a now widely used tabular format. To make the ISA framework more accessible to machines and enable programmatic manipulation of experiment metadata, a JSON serialization ISA-JSON was developed. **Results:** In this work, we present the ISA API, a Python library for the creation, editing, parsing, and validating of ISA-Tab and ISA-JSON formats by using a common data model engineered as Python class objects. We describe the ISA API feature set, early adopters and its growing user community. **Conclusions:** The ISA API provides users with rich programmatic metadata handling functionality to support automation, a common interface and an interoperable medium between the two ISA formats, as well as with other life science data formats required for depositing data in public databases.

Keywords

metadata, standards, reproducibility, open-source software, life science, API

Findings

Background

There are many data models and frameworks for describing entities and artefacts of scientific research.The life sciences have pioneered the development and application of ontologies, data standards, and minimum standards for reporting research results. More than ever, the importance of making scientific data FAIR (Findable, Accessible, Interoperable and Reusable) is at the forefront of discourse in the research community [1].

The ISA Metadata Framework, or simply ISA – so named after its constituent key concepts: 'Investigation' (the project context), 'Study' (a unit of research) and 'Assay' (analytical measurement), was born out of initial efforts to create an ‘exchange network test bed’ for a diverse set of digital resources in ‘omics studies. These included public and proprietary databases, as well as free and open source software tools and commercially developed systems. ISA specifies a data model and standard serialization formats for describing experiments in the life sciences.

At the heart of the ISA is a general-purpose data model; an extensible, hierarchically structured model that enables the representation of studies employing one or a combination of technologies, focusing on the description of their experimental metadata (i.e. sample characteristics, technology and measurement types, and sample-to-data relationships).

ISA was originally conceptualized and specified as the ISA-Tab format [2] for describing life science experiments. Refinements to the underlying data model were later captured in a new JSON-based serialization format, ISA-JSON [3]. A data model was then abstracted from both the tabular and JSON formats in order to formalize the core concepts and their relationships to one another (Figure 1). The ISA Data Model is formed around three core concepts: *Investigations*, *Studies* and *Assays*.

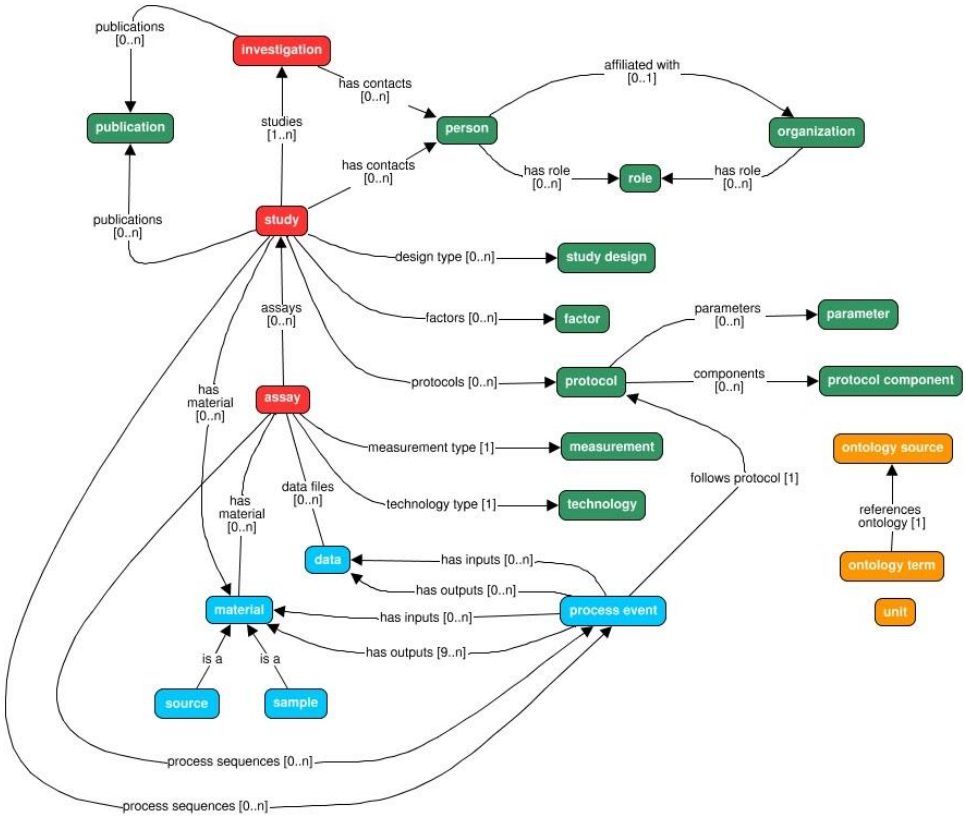

Figure 1. An overview of the ISA Data Model showing its key constituent classes and their relationships with one another. The model is structured around the concepts of investigation, study and assay (in red). Other model elements exist to qualify these core elements (in green), for example relating investigations and sub-studies with relevant contact persons or related publications; or information about the study design used and any experimental protocols implemented. Experimental workflows are modeled as sequences of process events with inputs and outputs that correspond to biological materials and data objects (in blue). Values can be made explicit by using term annotations or unit declarations that reference published ontologies (in orange).

Assays refer to specific tests performed on sample material taken from a subject, or performed on a whole subject, that produces some kind of qualitative or quantitative measurements, which correspond to response variables. Assay metadata describes sample-to-data relationships, grouped by analytical measurement types (e.g. metabolite profiling, DNA profiling) and technologies (e.g. Mass Spectrometry or Nuclear Magnetic Resonance, if the measurement type is metabolite profiling).

Study objects hold metadata about the subject(s) under study, including properties about the individual subject(s), any treatments (biological or statistical) applied, and the provenance of sample material back to the original source. Experimental factors relating to the subjects and

samples are stored here, allowing the definition of specific study groups in relation to the study independent variables.

Investigations contain all the information needed to understand the overall goals and means used in an experiment and are used as high-level objects to group multiple related Study objects. For each Investigation, there may be one or more Studies associated with it; for each Study there may be one or more Assays.

The ISA Data Model allows us to find scientific experiments of interest by having high-level metadata about the experiments such as what the studies are about and more concretely with descriptors of the Study Design type (a classifier for the study based on the overall study design, e.g. crossover design or parallel group design) and study factors used, e.g. independent variables manipulated by the experimentalist in the study. Furthermore, experiments can be searched by Assay types, on the type of measurements being carried out in the study and the technologies used to perform the measurements.

A key feature in the ISA Data Model is the use of ontology terms for certain fields in the metadata, supported with the Ontology Annotation and Ontology Source classes. Where appropriate, ISA Data Model objects can be qualified with ontology terms (Ontology Annotations) that are linked to a declared description of the source of the terms (Ontology Sources). Supporting software tools can therefore populate such annotations from established terminology services such as the NCBI BioPortal [4] or the EMBL-EBI Ontology Lookup Service [5]. By providing support for discretely identified terms, we enable the ability to search across ISA objects stored in different experiments.

The ISA Data Model enables experimentalists to describe metadata relating to the provenance of samples and data. This provenance is modeled in the form of directed acyclic graphs (i.e. vertices connected together by edges, but without any loops/cyclic dependencies), which describe the workflow undertaken from subject to sample to the production of data, including associated data acquisition files, analysis data matrices and discoveries being preserved in reusable form.

Full details about the ISA Data Model, as well as specifications for ISA-Tab and ISA-JSON are available from [6].

The supporting open-source ISA tools form a software suite designed to manage the experimental metadata using the aforementioned ISA formats, and more specifically to: (i) customise, describe and validate, following community reporting standards (with the ISAconfigurator, ISAcreator and ISAvalidator components [7] or OntoMaton [8]), (ii) store and browse, locally or remotely (BioInvestigation Index [9] and the ISAexplorer [10]); (iii) submit to public repositories (ISAconverter [7]); (iv) analyse with existing tools (Risa) [11]; (v) release, reason and generate Linked Data [12,13] , (ISA2RDF [14], LinkedISA [15]); and (vi) publish data alongside the article. The adoption of the Java-based [7] and web-based [9] software tools has helped grow the ISA community of users, characterised by those listed in the ISA Commons [16].

As the original ISA software suite does not provide a programmatic interface, we have developed the ISA API to position the ISA framework as an open and interoperable platform [17]. We also recognized a trend of growing enthusiasm by end users for writing software code. This trend is perhaps most evident in statistical programming platforms such as R, Python and MatLab, and also greatly helped with the uptake of interactive programming environments such as Project Jupyter [18].

The aim of the ISA API is to reproduce much of the user-facing functionality afforded by the ISA software suite and to additionally enable interoperation between software systems that produce and consume ISA formats and other machine-readable data formats, as illustrated in Figure 2. The ISA API is written in Python, which has a high uptake by non-programmers and a rich open source community ecosystem of supporting software packages, including many statistical and bioinformatics ones. The software code is available on GitHub [19] under the Common Public Attribution License Version 1.0 (CPAL-1.0) and the software package is available via the Python Package Index (PyPI) [20] and Bioconda [21] under the moniker: `isatools` [22,23].

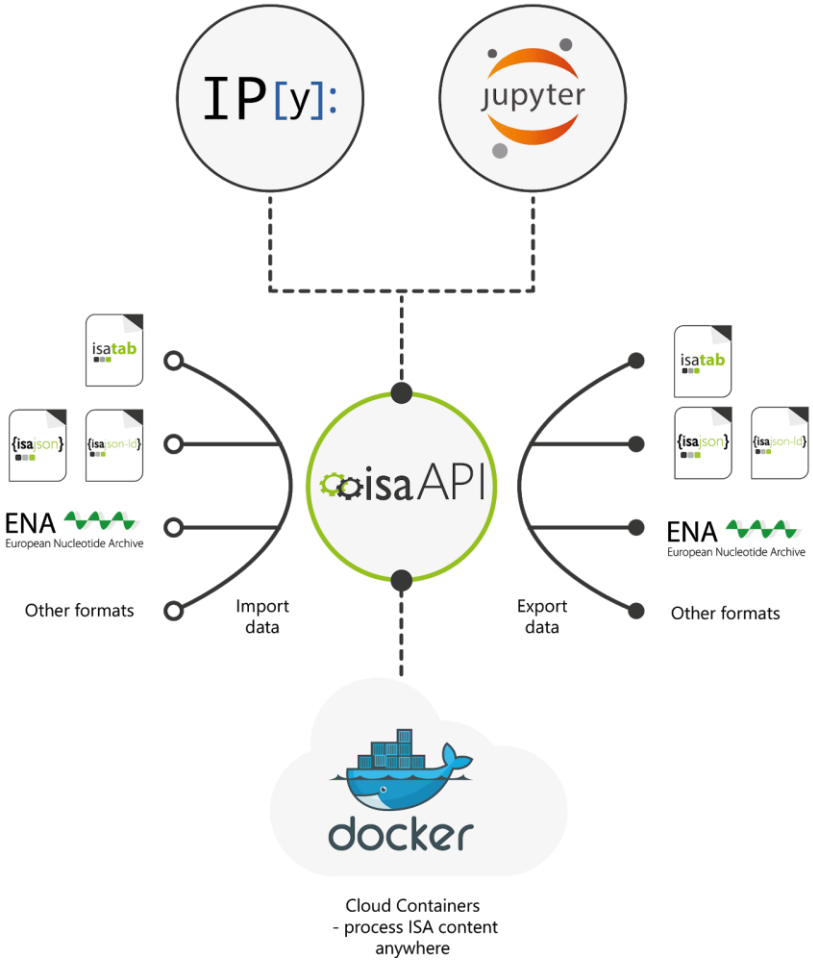

Figure 2. ISA API and its interactions with other software ecosystems and data formats. Apart from running in the Python interpreter, the ISA API can also be accessed through the iPython, Jupyter and Docker. It also supports interoperation with other systems through standardised machine-readable data formats.

Implementation

The ISA API is written in the Python 3 programming language, and therefore part of the wider Python ecosystem of software tooling which is very popular in the bioinformatics and data science communities [24]. The ISA API can immediately be imported and used alongside other Python packages in custom Python programs, or it can be used interactively via iPython's interactive shell [25] and in Jupyter Notebook web applications. It can be used in cloud infrastructure that supports Docker through the container image `isatools/isatools` published in DockerHub. The ISA API can be also exposed through RESTful APIs as demonstrated by the MetaboLights Labs Web service interface and the MetaboLights Online Editor Web service [26].

ISA API features

The ISA API provides a range of features that are summarized in this section.

Support for all ISA formats

The ISA API provides the solution to simplify access and generation of ISA metadata now that the ISA framework supports more than one on-disk data format – i.e., ISA-Tab and ISA-JSON. By using the ISA API, software can support full read, write and validation transparently on all current and future ISA formats without any additional complexity.
The ISA-Tab and ISA-JSON document validators included in the ISA API provide reference implementations by which to also validate documents generated by other systems that claim ISA compatibility [16].

Support for other bioinformatics formats

The ISA API also supports related domain formats used by bioinformatics communities and which are usually technology-centric (e.g. mass spectrometry, sequencing, DNA microarrays). This feature is a consequence of the ISA framework's support for multi-modality datasets (e.g. multi-omics datasets). The ISA API supports the import and export of MAGE-TAB [27] and SampleTab [28], import of metadata from mzML [29] and nmrML [30], and exports to SRA-XML [31] and Workflow4metabolomics [32].

Export metadata for submission to public repositories

The ability to export metadata to a range of formats means that software using the ISA API can prepare metadata submissions to public data repositories. The ISA API supports export of ISA metadata to submission-ready formats for public repositories such as the EMBL-EBI ArrayExpress [33], European Nucleotide Archive [34], and MetaboLights [35].

Multiple modes of handling ISA metadata

The ISA API is designed to support multiple modes of reading and writing ISA. Metadata reading and writing can be done natively with ISA-Tab folders, ISArchive zip files, and ISA-JSON files or strings. ISA metadata can be manipulated after loading using an object-oriented Python class model. Direct conversion between ISA formats is also supported.

Extensible and efficient object-oriented class model

Metadata objects are represented in the ISA API using an object-oriented class model, providing a format-agnostic in-memory representation. This in-memory model is based on the ISA Abstract Model, which was extracted from the original ISA-Tab specification to formalize the concepts used in the ISA framework. It describes the steps in an experimental workflow as a directed acyclic graph (DAG), which in the ISA API is expressed naturally as connected Python class objects. For example, metadata relating to the assay description is held in an `Assay` object that is in-turn contained by a `Study` object and its relevant attributes, which is finally contained by an `Investigation` object – thus reflecting the hierarchical nature of the ISA framework concepts. All ISA concepts are modelled as classes at a granular level giving software developers full control to extend the API's capabilities and build new features. Details of the class model used in the ISA API are published in the ISA Model and Serialization Specifications 1.0 [3], the latest version of which can be viewed online [6].

This functionally rich object-based representation completely avoids many of the complexities associated with handling ISA metadata as tables – e.g., joining data from the sample and assay tables, handling multiple table rows per edge of the DAG. While in memory, the experimental metadata can be programmatically processed and manipulated. Moreover, the ISA API's object-based representation is a powerful decoupling tool. First, it decouples the client application from the specific ISA file format being handled. Second, it decouples the input and output metadata formats. When metadata is read, the format-specific

parser creates the standardized in-memory representation, which can be edited and manipulated; when writing, a format-specific serialization routine traverses the objects to extract the data structure according to the target ISA format. Such decoupling between object creation and serialization is clearly visible from the ISA API documentation, for instance when invoking the `isatab2json` converter [36]. Writing the ISA API object model to ISA-Tab requires converting experimental workflow graphs to tabular formats. Such work is described in the `graph2tab` library by Brandizi et al [37]. However, in that work, the authors only describe cases of unidirectional transformations from graphs to tables; ISA API implements bidirectional transformations between each of the two ISA formats and ISA content expressed as Python objects that includes a representation of DAGs. ISA content can also be authored by directly creating Python objects, a brief example of which is shown in Figures 3 and 4.

```python
#!/usr/bin/env python3

from isatools.model import *

def create_descriptor():
    """Returns a simple ISA-Tab 1.0 descriptor for demonstration."""
    investigation = Investigation()
    investigation.identifier = "i1"
    investigation.title = "My Simple ISA Investigation"

    study = Study(filename="s_study.txt")
    study.identifier = "s1"
    study.title = "My ISA Study"

    source = Source(name='source_material')
    study.sources.append(source)

    ncbitaxon = OntologySource(name='NCBITaxon', description="NCBI Taxonomy")
    characteristic_organism = Characteristic(category=OntologyAnnotation(term="Organism"),
                                             value=OntologyAnnotation(term="Homo Sapiens", term_source=ncbitaxon,
                                             term_accession="http://purl.bioontology.org/ontology/NCBITAXON/9606"))

    prototype_sample = Sample(name='sample_material', derives_from=[source])
    prototype_sample.characteristics.append(characteristic_organism)
    study.samples = batch_create_materials(prototype_sample, n=3)  # creates a batch of 3 samples

    sample_collection_protocol = Protocol(name="sample collection",
                                          protocol_type=OntologyAnnotation(term="sample collection"))
    study.protocols.append(sample_collection_protocol)

    sample_collection_process = Process(executes_protocol=sample_collection_protocol)
    sample_collection_process.inputs = study.sources
    sample_collection_process.outputs = study.samples

    study.process_sequence.append(sample_collection_process)

    investigation.studies.append(study)

    # Serialize the model that we built above to an ISA-Tab sent to stdout
    from isatools import isatab
    return isatab.dumps(investigation)

if __name__ == '__main__':
    print(create_descriptor())
```

Figure 3. Example of a Python script using the ISA API's class model objects to programmatically construct metadata about a study and serialize it to ISA-Tab. This script first creates the Investigation and Study structures that store general metadata about the experiment being described. Next, a source material object is created, and three sample materials. These are connected as inputs and outputs respectively to a sample collection process,

forming a workflow from source to samples. This is then added to the study's 'process sequence': a container for experimental process event descriptions. Finally, the composition of the model objects is serialized as ISA-Tab to the standard output. Scripts such as these can form part of a larger Python software program, or executed directly from the command-line, to automate the construction of ISA metadata descriptors.

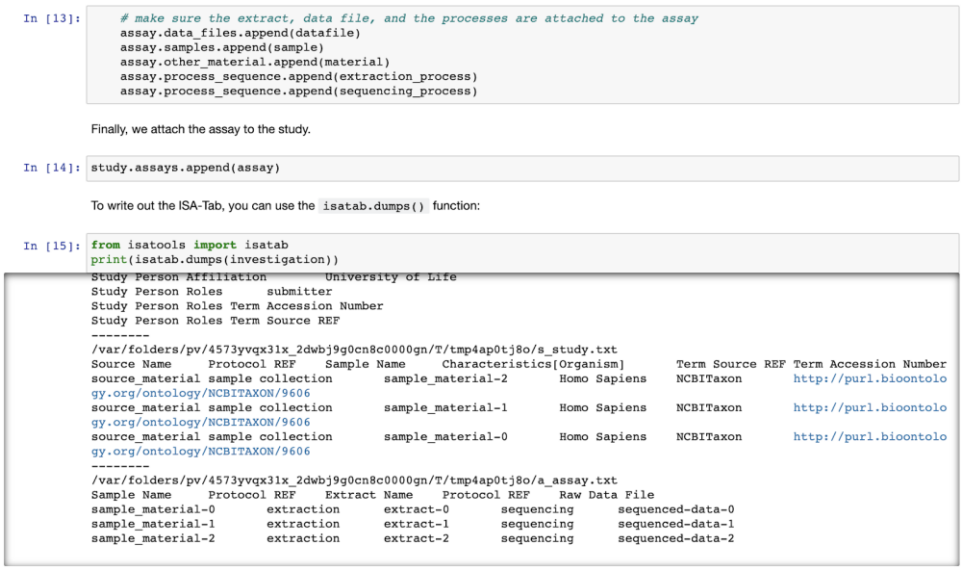

Figure 4. Example Jupyter Notebook using the ISA API to use ISA class objects to programmatically construct metadata about a study, using similar code to that shown in Figure 3. Being Python-based, ISA API can integrate with any notebook environment that supports Python kernels including Jupyter Notebook, JupyterLab and JupyterHub, and proprietary notebook environments such as Google Colab [38]*, Microsoft Azure Notebooks* [39]*, and Amazon's SageMaker* [40]*. A set of Jupyter notebooks detailing how to use ISA-API key functionalities is available on GitHub* [41]*.*

Querying over ISA content

Common use cases for ISA metadata involve gathering samples and data files produced by assays that satisfy certain selection criteria. The criteria are objects based on sample characteristics, processing parameters, and study factor combinations (treatment groups). When ISA metadata is loaded as in-memory model objects, this representation can be queried using native Python code, for example by filtering using list comprehensions with conditional logic. The ISA API also provides helper functions, in the `isatools.utils` package, to query and fix possible encoding problems in ISA content. For example in ISA-Tab files, a common issue is when ISA content is encoded in such a way that some

branches in experimental graphs converge unexpectedly which represents the intended workflow incorrectly. The ISA API provides specific functions to detect such anomalies (in the `detect_isatab_process_pooling` function) and to fix them (in the `insert_distinct_parameter` function). Details of how to query over ISA content can be found in the ISA API online documentation [42].

Semantic markup

An important feature that is embedded into the class model is the support for ontology annotation objects to better qualify metadata elements, contributing to making them Findable, Accessible, Interoperable, and Reusable (FAIR) [43]. The ISA API implements the `OntologyAnnotation` class that is used extensively throughout. To express quantitative values, units are implemented with the `Unit` class, which itself can be qualified with an `OntologyAnnotation` object. Populating annotations is aided by a network package in the ISA API for connecting to and querying the Ontology Lookup Service (OLS) [5]. JSON-LD context files complementing the ISA-JSON schemas [44] mapping into schema.org [45] or OBO Foundry based resources [46] are also available.

Assisted curation of study metadata

Studies published using ISA formats are increasingly commonplace, many of which are produced in ISA-Tab using the ISAcreator (for example, NASA GeneLab [47] and MetaboLights [35]), and many others produced either by-hand (using a spreadsheet or text editor) or output by third-party systems (for example, COPO [48], FAIRDOM/SEEK [49] and SCDE [50]). There are sometimes cases of invalid ISA-formatted content appearing in public databases or reported via bug reports to the ISA development team. To help with this, the ISA API includes features to assist with metadata curation of ISA-formatted studies. Beyond using the validators contained in the API, additional features include checking for possible structural problems in the DAGs that describe the provenance of samples and data files (for example, where converging edges are detected but are not expected) and correcting them;

re-situating incorrectly placed attributes (e.g. recasting incorrectly labelled `Factors` as `Parameters` or `Characteristics`); and batch fixing such issues. This work has resulted in the creation of curation functions, which have been applied to the content of the MetaboLights database as a means to improve the quality and consistency of ISA archives. These validators also form the basis of the current MetaboLights extensive online validations coupled to their submission infrastructure to validate the metadata for missing values, consistency and sufficiency.

Assisted creation of study metadata

The ISA API *create* module contains a set of functions and methods exploiting study design information to bootstrap the creation of ISA content [51]. Initially created to support factorial designs, this component is currently being extended to provide support for longitudinal studies and repeated treatment designs. This work in progress will be refined in future versions of the ISA API.

For more information about the features listed here, interested readers are invited to view the ISA tools API documentation [42] for detailed and up-to-date information.

Early adopters

The ISA API has a number of early adopters that demonstrate its value as an open platform to the ISA framework. The following section details how these groups rely on our python library in their data management tasks.

The MetaboLights metabolomics database [35] is using the ISA API to load, edit, and save ISA-Tab as part of its next-generation data curation interface. MetaboLights has redeveloped its user and data curator interface building on the latest Web technologies and using the ISA API to convert from ISA-Tab to ISA-JSON for further processing. The developers take a service-oriented approach exposing various features for editing and updating ISA content through a RESTful API [26].

The PhenoMeNal (Phenome and Metabolome aNalysis) e-infrastructure for molecular phenotype data analysis has developed a set of containerised microservices (using Docker) that utilise ISA API's converters, validators, and study creation features [52]. The PhenoMenal Virtual Research Environment (VRE) is based on the Galaxy workflow platform [45] through its integration with Kubernetes [53]. Through PhenoMeNal, a suite of tools has been developed to integrate the ISA API with Galaxy [54]. The tools have been published as a Galaxy workflow with the Dalcotidine release of PhenoMeNal to demonstrate a study metadata preparation and pre-submission pipeline to the MetaboLights database (see Figure 5).

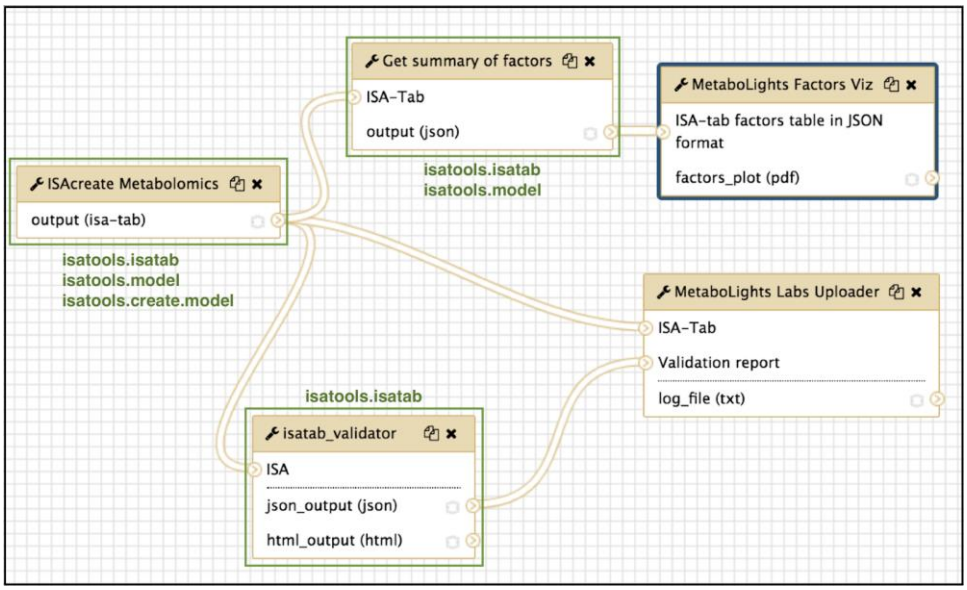

Figure 5. The ISA Create-Validate-Upload workflow, published as part of the PhenoMeNal platform Dalcotidine release. The workflow takes a user-configured study plan and creates an ISA-Tab template ready for the experimentalist to use in their study. The ISA-Tab then goes through two paths of processing: (1) a summary of study factors according to the study design is extracted and then visualized as a parallel sets plot; and (2) the ISA-Tab is validated, and if valid a pre-submission request is made by uploading the ISA-Tab to MetaboLights Labs. A preparatory study accession ID is then issued by MetaboLights if accepted. Parts of the workflow that directly use the ISA API are highlighted in green along with the packages used.

The Galaxy workflow platform includes native support for ISA formats by using the ISA API in its datatype implementations of ISA-Tab and ISA-JSON, paving the way for Galaxy tools to consume and produce ISA content as first-class Galaxy datasets.

The Collaborative Open Omics (COPO) platform [55] supports ISA-JSON as one of its metadata formats and uses ISA API's SRA exporter to allow COPO users to publish studies directly to the European Nucleotide Archive. COPO configures its back end and front end for brokering various data types, including omics data, through consuming ISA configurations. The ISA configurations inform how COPO should present the data preparation wizard to the user, and how it stores it in its database in a variation of ISA-JSON. COPO stores study metadata as JSON where it conforms to the ISA-JSON schemas and adds extra metadata relating to its operation within the COPO database. By supporting ISA-JSON, COPO lends itself to easily leveraging the ISA API to export data to formats supported by the API.
Still within the plant science community, a conversion from BrAPI [56] to ISA has been developed along with MIAPPE compliant ISA configurations [57]. The functionality will be integrated along with other community conversion components in future releases of the ISA API.

A stable and growing user community

Since its first release, ISA API has grown its user base steadily with active contributions from the community as an open source project as evidenced by bug reports, feature requests, and code contributions. While it is difficult to measure the uptake of free software, ISA API has been available since 2016 via PyPI, which collects download statistics for all packages it hosts [58]. There are documented drawbacks to using these PyPI download statistics, such as historical data issues and systemic irregularities such as caching of downloads that can cause undercounting, so we view these statistics as indicative of how the ISA API project has progressed rather than as an accurate measure. These statistics also do not include installations directly from the source code repository on GitHub or from Bioconda.

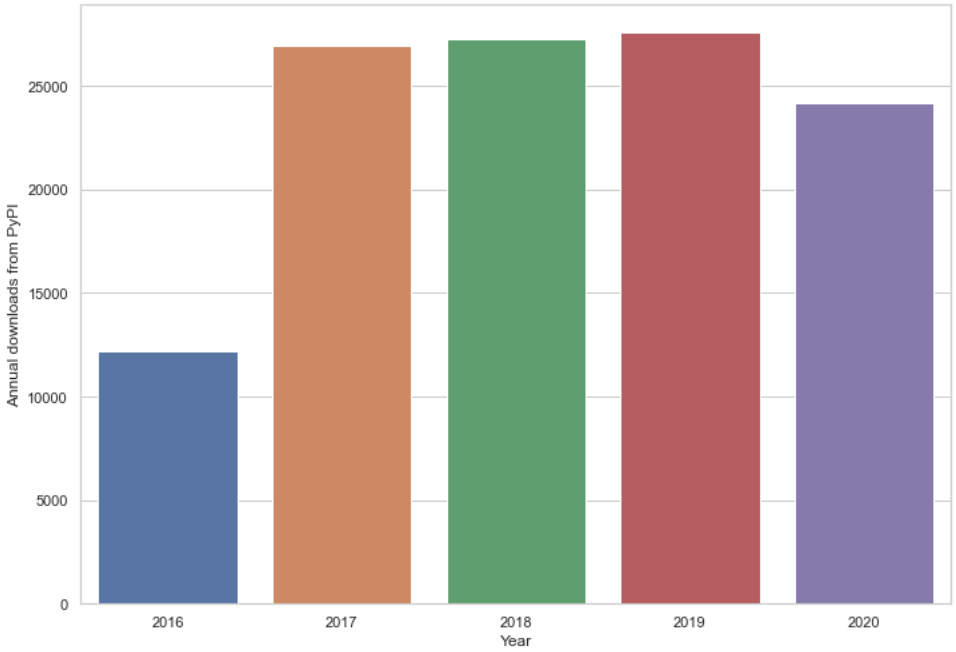

Figure 6. Bar chart showing the yearly total number of downloads of the `isatools` Python package from PyPI from 21 February 2016 (first release of ISA API, v0.1) up to 24 October 2020 (after latest release v0.11). Note that the total number of downloads for 2020 is incomplete at the time of writing.

After its initial release year, we observed a maintained and slight year-on-year growth from 2017 to 2019 of around 27,000 annual downloads via PyPI (Figure 6), which we believe demonstrates that the ISA API has built up a stable and active user base.

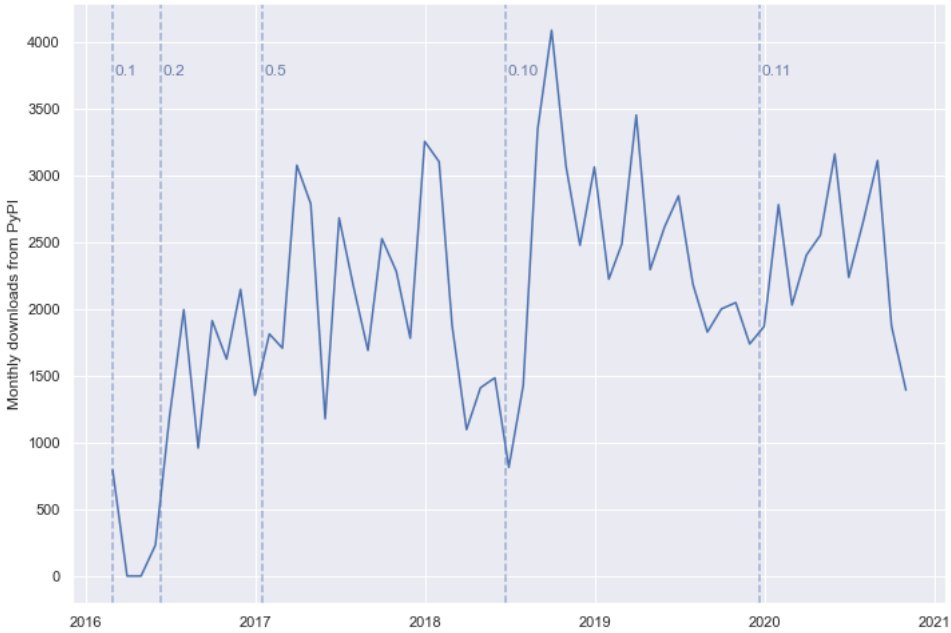

Figure 7. Line chart showing the monthly total number of downloads of the `isatools` Python package from PyPI from 21 February 2016 (first release of ISA API, v0.1) up to 24 October 2020 (v0.11). Major releases of the ISA API are indicated on the chart with a dashed line.

When looking at a more granular picture of the download statistics for the `isatools` package, on a month-by-month basis, we observe some interesting patterns. The initial release of version 0.1 understandably has far fewer downloads than later releases, however after each major release we note a significant jump in downloads in the following months. This can be explained by users updating the versions they might be using to incorporate bug fixes, and also that users are accessing the newly released features in the ISA API.

The major releases highlighted in Figure 7 each made significant new functionality available to the user community. At version 0.2, the ISA-Tab and ISA-JSON validators were released. For version 0.5, a major performance enhancement to the ISA-Tab parser was implemented in response to user community feedback. Version 0.10 saw the release of the ISA API *create* module for assisted study metadata creation. The latest version at the time of writing, version 0.11, consists of upgrades to the create module and to ISA-JSON support.

We believe that the spikes in downloads observed on PyPI after each major release indicate that the ISA API user community is actively engaged with and continuing to use the platform by migrating to its newest updates soon after they are made available.

Conclusions

The development of the ISA API represents a major step forward in making the ISA framework open and interoperable, enabling better handling of experimental metadata, and supporting a diverse user community. By providing a programming interface, rather than a graphical user interface, the ISA API provides a platform for simplified and consistent integration of the ISA framework in new and existing software, promoting the production of structured metadata by agents and the development of data management solutions which can be FAIR by design. The availability of an ISA implementation that is easily used by other software also reduces the likelihood of independent implementations that may not strictly adhere to the ISA format specifications. The object-oriented class model provides a format-agnostic interface for client software allowing software that uses the ISA API to automatically

support all ISA file formats, while also providing a common basis on which to build data format parsers and serializers more easily. Should external implementations of the ISA formats be required, by providing reference implementations of ISA-Tab and ISA-JSON parsers, serializers and validators, software developers have the infrastructure in which to extend the API or build interoperation of ISA with other software systems. As Python code, it can also easily integrate into existing bioinformatics and data science platforms such as Galaxy and Jupyter. Importantly, the object-oriented DAG model presented by the ISA API offers a more intuitive way to reason about and process study metadata than as ISA-Tab tables, simplifying the work of developers and enabling the development of richer metadata processing applications. The ISA API is available as free and open-source software which will improve the dissemination and application of the ISA metadata framework.

Methods

The download statistics for the `isatools` package hosted on PyPI are collected by the Linehaul statistics daemon [59] that logs downloads of all Python packages on PyPI and pushes the data to a public dataset [58] on Google BigQuery [60]. The all-time statistics about the `isatools` package were retrieved from this public dataset on 24 October 2020 at 14:20 UTC+2 with the following SQL query via the BigQuery interface:

```
SELECT TIMESTAMP_TRUNC(timestamp, WEEK) week
, COUNT(*) downloads
FROM `the-psf.pypi.downloads*`
WHERE file.project='isatools'
GROUP BY week, file.version
ORDER BY week
```

The result of this query was downloaded as a CSV file containing a timestamp of the week-ending each weekly period of downloads, the `isatools` version number, and the total number of downloads for each period and version. This data was then aggregated by month and by year before being visualized using the Seaborn statistical data visualization package [61].

Data about the version release dates of ISA API were collected from the ISA API GitHub repository [19].

Availability of supporting data

The all-time weekly download statistics data for the `isatools` package on PyPI up to 24 October 2020 14:20 UTC+2 and the code used to plot the charts used in Figures 6 and 7 in this manuscript are available in a Code Ocean capsule [62]. The data and code is available under CC BY 4.0 and MIT License respectively.

Availability of source code and requirements

- Project name: ISA API
- Project home page: http://github.com/ISA-tools/isa-api/
- Operating system(s): Platform independent
- Programming language: Python
- Other requirements: Python 3.6+
- License: CPAL-1.0

Declarations

List of abbreviations

API: Application Programming Interface

BBSRC: Biotechnology and Biological Sciences Research Council

BrAPI: Breeding API

COPO: Collaborative Open Plant Omics

DAG: Directed Acyclic Graph

EMBL: European Molecular Biology Laboratory

EMBL-EBI: European Bioinformatics Institute

GUI: Graphical User Interface

H2020: Horizon 2020

ISA: Investigation, Study, Assay

ISA-JSON: ISA JavaScript Object Notation format

ISA-Tab: ISA Tabular format

JSON: JavaScript Object Notation

JSON-LD: JavaScript Object Notation for Linked Data

MAGE-TAB: MicroArray Gene Expression-Tabular format

MIAPPE: Minimum Information About a Plant Phenotyping Experiment

mzML: Mass spectrometry Markup Language

NASA: National Aeronautics and Space Administration

NERC: Natural Environment Research Council

NCBI: National Center for Biotechnology Information

nmrML: Nuclear magnetic resonance Markup Language

OLS: Ontology Lookup Service

PhenoMeNal: Phenome and Metabolome aNalysis

PyPI: Python Package Index

REST: Representational state transfer

RDF: Resource Description Framework

SCDE: Stem Cell Discovery Engine

SQL: Structured Query Language

SRA-XML: Sequence Read Archive-eXtensible Markup Language

VRE: Virtual Research Environment.

Consent for publication

Not applicable.

Competing Interests

All authors declare that they have no other competing interests.

Funding

This work has been supported in part by: European Commission Horizon 2020 programme *PhenoMeNal* project (grant agreement no. EC654241); UK BBSRC *COPO* project (Bioinformatics and Biological Resources Fund (BBR) grant: BB/L021390/1 [BB/L024055/1, BB/L024071/1, BB/L024101/1]); UK BBSRC *Establishing common standards and curation practices: towards real world biosharing* grant (BB/J020265/1); UK BBSRC *Sharing of metabolomics data and their analyses as Galaxy workflows through a UK-China collaboration* grant (BB/M027635/1); and a UK NERC CASE Ph.D. studentship in collaboration with GigaScience: (NE/L002493/1).

A.G-B., K.H., M.I., D.J., M.L., T.N.L., P.M., L.P., P.R-S, P.R., S-A.S., C.S. and R.J.M.W. received funding from the EC H2020 PhenoMeNal project grant. A.E., R.P.D., A.G-B., D.J., P.R-S, S-A.S., and F.S. received funding from the UK BBSRC COPO project grant. T.N.L. received funding from the NERC CASE Ph.D. studentship.

Author's Contributions

Conceptualization: P.R-S, S-A.S.; software-lead: D.J.; software-supporting: M.I., P.R-S, A.G-B, K.C., A.E., K.H., M.L., T.N.L., A.M., P.M., V.C.N., L.P., P.R., F.S., R.J.M.W.; investigation-usage: D.J.; visualization-usage: D.J.; writing-original draft: D.J.; writing-review and editing: D.J., P.R-S., K.C., R.P.D., A.E., A.G-B, K.H., M.I., M.L., T.N.L., P.M., V.C.N., C.O., L.P., P.R., F.S., C.S., R.J.M.W., S-A.S.; funding acquisition: R.P.D., C.O., S-A.S, C.S.

Acknowledgements

We thank members of the ISA Working Group, ISA Commons, H2020 PhenoMeNal project, UK BBSRC COPO project, attendees of the 2016 “Hack-the-Spec - ISA as a FAIR research

object" workshop hosted by the Oxford e-Research Centre, UK, attendees of the 2016 "China-UK Data Dissemination in Metabolomics (CUDDEL)" workshop hosted by GigaScience, Hong Kong, and the EMBL-EBI MetaboLights team for guidance and feedback on the development of the ISA API.

References

1. McQuilton P, Batista D, Beyan O, Granell R, Coles S, Izzo M, et al. Helping the consumers and producers of standards, repositories and policies to enable FAIR Data. Data Intell. 2019; doi:10.1162/dint_a_00037.

2. Rocca-Serra P, Sansone S-A, Brandizi M. Specification documentation: ISA-TAB 1.0. Zenodo. 2009; doi:10.5281/zenodo.161355.

3. Sansone S-A, Rocca-Serra P, Gonzalez-Beltran A, Johnson D, ISA Community. ISA model and serialization specifications 1.0. Zenodo. 2016; doi:10.5281/zenodo.163640

4. Whetzel PL, Noy NF, Shah NH, Alexander PR, Nyulas C, Tudorache T, et al. BioPortal: enhanced functionality via new Web services from the National Center for Biomedical Ontology to access and use ontologies in software applications. Nucleic Acids Res. 2011; doi:10.1093/nar/gkr469.

5. Jupp S, Burdett T, Leroy C, Parkinson HE. A new ontology lookup service at EMBL-EBI. In: Malone J, Stevens R, Forsberg K, Splendiani Al, editors. Proceedings of the 8th International Conference on Semantic Web Applications and Tools for Life Sciences (SWAT4LS 2015). Aachen: CEUR-WS.org; 2015.

6. ISA Model and Serialization Specifications. https://isa-specs.readthedocs.io/en/latest/isamodel.html. Accessed 9 Oct 2020.

7. Rocca-Serra P, Brandizi M, Maguire E, Sklyar N, Taylor C, Begley K, et al. ISA software suite: supporting standards-compliant experimental annotation and enabling curation at the community level. Bioinformatics. 2010. doi:10.1093/bioinformatics/btq415.

8. Maguire E, Gonzalez-Beltran A, Whetzel PL, Sansone S-A, Rocca-Serra P. OntoMaton: a

BioPortal powered ontology widget for Google Spreadsheets. Bioinformatics. 2013;29:525-527.

9. Gonzalez-Beltran A, Maguire E, Georgiou P, Sansone S-A, Rocca-Serra P. Bio-GraphIIn: a graph-based, integrative and semantically-enabled repository for life science experimental data. EMBnet J. 2013;19:46–50.

10. Gonzalez-Beltran A. ISA-explorer: A demo tool for discovering and exploring Scientific Data's ISA-tab metadata. http://blogs.nature.com/scientificdata/2015/12/17/isa-explorer/. Accessed 11 Nov 2020.

11. Gonzalez-Beltran A, Neumann S, Maguire E, Sansone S-A, Rocca-Serra P. The Risa R/Bioconductor package: integrative data analysis from experimental metadata and back again. BMC Bioinformatics. 2014; doi:10.1186/1471-2105-15-S1-S11.

12. Mons B, Velterop J. Nano-publication in the e-science era. In: Clark T, Luciano JS, Marshall MS, Prud'hommeaux E, Stephens S, editors. Proceedings of the Workshop on Semantic Web Applications in Scientific Discourse (SWASD 2009), collocated with the 8th International Semantic Web Conference (ISWC 2009). Aachen: CEUR-WS.org; 2009.

13. Groth P, Gibson A, Velterop J. The anatomy of a nanopublication. Inf Serv Use. 2010;30:51-56.

14. Kohonen P, Benfenati E, Bower D, Ceder R, Crump M, Cross K, et al. The ToxBank data warehouse: Supporting the replacement of in vivo repeated dose systemic toxicity testing. Mol Inform. 2013;32:47-63.

15. Gonzalez-Beltran A, Maguire E, Sansone S-A, Rocca-Serra P. linkedISA: semantic representation of ISA-Tab experimental metadata. BMC Bioinformatics. 2014;15:1-15

16. ISA commons. https://www.isacommons.org/. (2018). Accessed 9 Oct 2020.

17. Eisenmann TR, Parker G, Van Alstyne M. Opening platforms: how, when and why? In: Gawer A, editor. Platforms, markets and innovation. Cheltenham: Edward Elgar Publishing; 2009. p. 131-162.

18. Kluyver T, Ragan-Kelley B, Pérez F, Granger BE, Bussonnier M, Frederic J, et al. Jupyter Notebooks - a publishing format for reproducible computational workflows. In:

Proceedings of the 20th International Conference on Electronic Publishing (ELPUB 2016). Amsterdam: IOS Press; 2016. p. 87-90.

19. ISA API on GitHub. https://github.com/ISA-tools/isa-api/. Accessed 9 Oct 2020.

20. The Python Package Index. https://pypi.org/. Accessed 29 Oct 2020.

21. Grüning B, Dale R, Sjödin A, Chapman BA, Rowe J, Tomkins-Tinch CH, et al. Bioconda: sustainable and comprehensive software distribution for the life sciences. Nat Methods.; 2018; doi:10.1038/s41592-018-0046-7.

22. ISA API (isatools) on PyPI. https://pypi.org/project/isatools/. Accessed 9 Oct 2020.

23. ISA API (isatools) on Bioconda. https://anaconda.org/bioconda/isatools/. Accessed 9 Oct 2020.

24. Russell PH, Johnson RL, Ananthan S, Harnke B, Carlson NE. A large-scale analysis of bioinformatics code on GitHub. PLoS ONE. 2018; doi:10.1371/journal.pone.0205898.

25. Pérez F, Granger BE. IPython: a system for interactive scientific computing. Comput Sci Eng. 2007:9:21-29.

26. MetaboLights RESTful WebService API specification. https://www.ebi.ac.uk/metabolights/ws/api/spec.html. Accessed 9 Oct 2020.

27. Rayner TF, Rocca-Serra P, Spellman PT, Causton HC, Farne A, Holloway E, et al. A simple spreadsheet-based, MIAME-supportive format for microarray data: MAGE-TAB. BMC Bioinformatics. 2006;7:489.

28. Courtot M, Cherubin L, Faulconbridge A, Vaughan D, Green M, Richardson D, et al. BioSamples database: an updated sample metadata hub. Nucleic Acids Res. 2019;47:D1172–D1178.

29. Martens L, Chambers M, Sturm M, Kessner D, Levander F, Shofstahl J, et al. mzML-a community standard for mass spectrometry data. Mol Cell Proteomics. 2011;10.

30. Schober D, Jacob D, Wilson M, Cruz JA, Marcu A, Grant JR, et al. nmrML: a community supported open data standard for the description, storage, and exchange of NMR data. Anal Chem. 2018; doi:10.1021/acs.analchem.7b02795.

31. Kodama Y, Shumway M, Leinonen R. The Sequence Read Archive: explosive growth of

sequencing data. Nucleic Acids Res. 2012;40:D54–D56 .

32. Giacomoni F, Le Corguillé G, Monsoor M, Landi M, Pericard P, Pétéra M, et al. Workflow4Metabolomics: a collaborative research infrastructure for computational metabolomics. Bioinformatics. 2015; doi:10.1093/bioinformatics/btu813.

33. Athar A, Füllgrabe A, George N, Iqbal H, Huerta L, Ali A, et al. ArrayExpress update–from bulk to single-cell expression data. Nucleic Acids Res. 2019; 47:D711–D715.

34. Amid C, Alako BT, Balavenkataraman Kadhirvelu V, Burdett T, Burgin J, Fan J, et al. The European Nucleotide Archive in 2019. Nucleic Acids Res. 2020;48:D70–D76.

35. Haug K, Cochrane K, Nainala VC, Williams M, Chang J, Jayaseelan KV, et al. MetaboLights: a resource evolving in response to the needs of its scientific community. Nucleic Acids Res. 2020;48:D440–D444.

36. ISA tools API documentation - ISA Conversions. https://isatools.readthedocs.io/en/latest/conversions.html. Accessed 9 Oct 2020.

37. Brandizi M, Kurbatova N, Sarkans U, Rocca-Serra P. graph2tab, a library to convert experimental workflow graphs into tabular formats. Bioinformatics. Oxford University Press; 28:1665–16672012;

38. Google Colaboratory. https://colab.research.google.com/. Accessed 22 Oct 2020.

39. Microsoft Azure Notebooks. https://notebooks.azure.com/. Accessed 22 Oct 2020.

40. Amazon SageMaker. https://aws.amazon.com/sagemaker/. Accessed 22 Oct 2020.

41. Example Jupyter notebooks using the ISA-API on GitHub. https://github.com/ISA-tools/isatools-notebooks/. Accessed 28 Oct 2020.

42. ISA tools API documentation. https://isatools.readthedocs.io/en/latest/. Accessed 9 Oct 2020.

43. Wilkinson MD, Dumontier M, Aalbersberg IjJ, Appleton G, Axton M, Baak A, et al. The FAIR Guiding Principles for scientific data management and stewardship. Sci Data. 2016;3:1-9.

44. Pezoa F, Reutter JL, Suarez F, Ugarte M, Vrgoč D. Foundations of JSON Schema. In: Bourdeau J, editor. Proceedings of the 25th International Conference on World Wide Web

(WWW '16). New York: ACM; 2016. p. 263-273.

45. Guha RV, Brickley D, MacBeth S. Schema.org: Evolution of Structured Data on the Web: Big data makes common schemas even more necessary. ACM Queue. 2015; doi:10.1145/2857274.2857276.

46. Smith B, Ashburner M, Rosse C, Bard J, Bug W, Ceusters W, et al. The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. Nat Biotechnol. 2007; doi:10.1038/nbt1346.

47. Ray S, Gebre S, Fogle H, Berrios DC, Tran PB, Galazka JM, et al. GeneLab: Omics database for spaceflight experiments. Bioinformatics. 2019;35:1753–1759.

48. Shaw F, Etuk A, Gonzalez-Beltran A, Rocca-Serra P, Kersey PJ, Bastow R, et al. COPO-Linked Open Infrastructure for Plant Data. In: Malone J, Stevens R, Forsberg K, Splendiani Al, editors. Proceedings of the 8th International Conference on Semantic Web Applications and Tools for Life Sciences (SWAT4LS 2015). Aachen: CEUR-WS.org; 2015.

49. Wolstencroft K, Owen S, Krebs O, Nguyen Q, Stanford NJ, Golebiewski M, et al. SEEK: a systems biology data and model management platform. BMC Syst Biol. 2015;9:1-12.

50. Ho Sui SJ, Begley K, Reilly D, Chapman B, McGovern R, Rocca-Serra P, et al. The Stem Cell Discovery Engine: an integrated repository and analysis system for cancer stem cell comparisons. Nucleic Acids Res. 2012; doi:10.1093/nar/gkr1051.

51. Rocca-Serra P, Johnson D, Weber RJM, Pireddu L, Roger P, Gonzalez-Beltran A, et al. ISAcreate Galaxy tool for prospective data management with ISA format support - application to metabolomics datasets (poster). F1000Research. 2018; doi:10.7490/f1000research.1115757.1.

52. Peters K, Bradbury J, Bergmann S, Capuccini M, Cascante M, de Atauri P, et al. PhenoMeNal: processing and analysis of metabolomics data in the cloud. Gigascience. 2019;8:giy149.

53. Moreno P, Pireddu L, Roger P, Goonasekera N, Afgan E, Van Den Beek M, et al. Galaxy-Kubernetes integration: scaling bioinformatics workflows in the cloud. bioRxiv. 2018;bioRxiv.488643.

54. ISA Galaxy tools, tours, and other enhancements on GitHub. https://github.com/ISA-tools/isatools-galaxy/. Accessed 9 Oct 2020.

55. Shaw F, Etuk A, Minotto A, Gonzalez-Beltran A, Johnson D, Rocca-Serra P, et al. COPO: a metadata platform for brokering FAIR data in the life sciences. F1000Research. 2020; doi:10.12688/f1000research.23889.1.

56. Selby P, Abbeloos R, Backlund JE, Basterrechea Salido M, Bauchet G, Benites-Alfaro OE, et al. BrAPI—an application programming interface for plant breeding applications. Bioinformatics. 2019; doi:10.1093/bioinformatics/btz190.

57. Ćwiek-Kupczyńska H, Altmann T, Arend D, Arnaud E, Chen D, Cornut G, et al. Measures for interoperability of phenotypic data: minimum information requirements and formatting. Plant Methods. 2016; doi:10.1186/s13007-016-0144-4.

58. Analyzing PyPI package downloads - Python Packaging User Guide. https://packaging.python.org/guides/analyzing-pypi-package-downloads/. Accessed 28 Oct 2020.

59. The Linehaul Statistics Daemon. https://github.com/pypa/linehaul/. Accessed 28 Oct 2020.

60. BigQuery: Cloud Data Warehouse. https://cloud.google.com/bigquery/. Accessed 28 Oct 2020.

61. Michael Waskom, Olga Botvinnik, Maoz Gelbart, Joel Ostblom, Paul Hobson, Saulius Lukauskas, et al. mwaskom/seaborn: v0.11.0. Zenodo. 2020; doi:10.5281/zenodo.4019146

62. Johnson D. Code for the ISA API download statistics visualizations in “ISA API: An open platform for interoperable life science experimental metadata” [Source Code]. Code Ocean. 2020. https://doi.org/10.24433/CO.8813991.v1