

Packaging research artefacts with RO-Crate

This manuscript ([permalink](#)) was automatically generated from [stain/ro-crate-paper@3cc917d](#) on August 13, 2021.

Authors

- **Stian Soiland-Reyes**

 [0000-0001-9842-9718](#) ·  [stain](#) ·  [soilandreyes](#)

Department of Computer Science, The University of Manchester, UK; Informatics Institute, Faculty of Science, University of Amsterdam, NL · Funded by [BioExcel-2](#) (European Commission [H2020-INFRAEDI-2018-1 823830](#)); [IBISBA](#) (H2020-INFRAIA-2017-1-two-stage 730976)

- **Peter Sefton**

 [0000-0002-3545-944X](#) ·  [ptsefton](#)

Faculty of Science, University Technology Sydney, AU

- **Mercè Crosas**

 [0000-0003-1304-1939](#) ·  [mercecrosas](#)

Institute for Quantitative Social Science, Harvard University, Cambridge, MA, US; Harvard Data Commons is supported by an award from [Harvard University Information Technology](#) (HUIT).

- **Leyla Jael Castro**

 [0000-0003-3986-0510](#) ·  [ljgarcia](#) ·  [lj_garcia](#)

ZB MED Information Centre for Life Sciences, Cologne, DE

- **Frederik Coppens**

 [0000-0001-6565-5145](#) ·  [frederikcoppens](#) ·  [FrederikCoppens](#)

VIB-UGent Center for Plant Systems Biology, Gent, BE

- **José María Fernández**

 [0000-0002-4806-5140](#) ·  [JMFernand3z](#)


Barcelona Supercomputing Center, Barcelona, ES

- **Daniel Garijo**

 [0000-0003-0454-7145](#) ·  [dgarijov](#)

Ontology Engineering Group, Universidad Politécnica de Madrid, Madrid, ES

- **Björn Grüning**

 [0000-0002-3079-6586](#) ·  [bg](#)

Bioinformatics Group, Department of Computer Science, Albert-Ludwigs-University Freiburg, Freiburg, DE · Funded by [EOSC-Life](#) (European Commission [H2020-INFRAEOSC-2018-2 824087](#)); [DataPLANT](#) ([NFDI 7/1 – 42077441](#)), part of the German [National Research Data Infrastructure](#) (NFDI), funded by the [Deutsche Forschungsgemeinschaft](#) (DFG).

- **Marco La Rosa**

 [0000-0001-5383-6993](#)

- **Simone Leo**

 [0000-0001-8271-5429](#) ·  [simleo](#) ·  [simleo](#)

Center for Advanced Studies, Research, and Development in Sardinia (CRS4), Pula (CA), Italy · Funded by [EOSC-Life](#) (European Commission [H2020-INFRAEOSC-2018-2 824087](#))

- **Eoghan Ó Carragáin**

 [0000-0001-8131-2150](https://orcid.org/0000-0001-8131-2150) ·  [eocarragain](https://github.com/eocarragain) ·  [eocarragain](https://twitter.com/eocarragain)

University College Cork, IE

- **Marc Portier**

 [0000-0002-9648-6484](https://orcid.org/0000-0002-9648-6484) ·  [mportier](https://twitter.com/mportier)

Vlaams Instituut voor de Zee, Oostende, BE

- **Ana Trisovic**

 [0000-0003-1991-0533](https://orcid.org/0000-0003-1991-0533) ·  [atrisovic](https://twitter.com/atrisovic)

Institute for Quantitative Social Science, Harvard University, Cambridge, MA, US · Funded by [Alfred P. Sloan Foundation](https://www.alfredpsloan.org/) (grant number [P-2020-13988](https://www.alfredpsloan.org/grant-number/P-2020-13988)).

- **RO-Crate Community**

<https://www.researchobject.org/ro-crate/community> ·  [researchobject](https://www.researchobject.org/)

- **Paul Groth**

 [0000-0003-0183-6910](https://orcid.org/0000-0003-0183-6910) ·  [pgroth](https://github.com/pgroth) ·  [pgroth](https://twitter.com/pgroth)

Informatics Institute, Faculty of Science, University of Amsterdam, NL

- **Carole Goble**

 [0000-0003-1219-2137](https://orcid.org/0000-0003-1219-2137) ·  [carolegoble](https://github.com/carolegoble) ·  [CaroleAnneGoble](https://twitter.com/CaroleAnneGoble)

Department of Computer Science, The University of Manchester, UK · Funded by [BioExcel-2](https://ec.europa.eu/info/funding-opportunities-and-programmes/erc/erc-grants_en) (European Commission [H2020-INFRAEDI-2018-1 823830](https://ec.europa.eu/info/funding-opportunities-and-programmes/erc/erc-grants_en)); [EOSC-Life](https://ec.europa.eu/info/funding-opportunities-and-programmes/erc/erc-grants_en) (European Commission [H2020-INFRAEOSC-2018-2 824087](https://ec.europa.eu/info/funding-opportunities-and-programmes/erc/erc-grants_en)); [IBISBA](https://ec.europa.eu/info/funding-opportunities-and-programmes/erc/erc-grants_en) (European Commission [H2020-INFRAIA-2017-1-two-stage 730976](https://ec.europa.eu/info/funding-opportunities-and-programmes/erc/erc-grants_en), [H2020-INFRADEV-2019-2 871118](https://ec.europa.eu/info/funding-opportunities-and-programmes/erc/erc-grants_en)); [SyntheSys+](https://ec.europa.eu/info/funding-opportunities-and-programmes/erc/erc-grants_en) (European Commission [H2020-INFRAIA-2018-1 823827](https://ec.europa.eu/info/funding-opportunities-and-programmes/erc/erc-grants_en))

An increasing number of researchers support reproducibility by including pointers to and descriptions of datasets, software and methods in their publications. However, scientific articles may be ambiguous, incomplete and difficult to process by automated systems. In this paper we introduce RO-Crate, an open, community-driven, and lightweight approach to packaging research artefacts along with their metadata in a machine readable manner. RO-Crate is based on Schema.org annotations in JSON-LD, aiming to establish best practices to formally describe metadata in an accessible and practical way for their use in a wide variety of situations.

An RO-Crate is a structured archive of all the items that contributed to a research outcome, including their identifiers, provenance, relations and annotations. As a general purpose packaging approach for data and their metadata, RO-Crate is used across multiple areas, including bioinformatics, digital humanities and regulatory sciences. By applying “just enough” Linked Data standards, RO-Crate simplifies the process of making research outputs FAIR while also enhancing research reproducibility.

Introduction

The move towards open science has increased the need and demand for the publication of artefacts of the research process [1]. This is particularly apparent in domains that rely on computational experiments; for example, the publication of software, datasets and records of the dependencies that such experiments rely on [2].

It is often argued that the publication of these assets, and specifically software [3], workflows [4] and data, should follow the FAIR principles [5]; namely, that they are Findable, Accessible, Interoperable and Reusable. These principles are agnostic to the *implementation* strategy needed to comply with them. Hence, there has been an increasing amount of work in the development of platforms and specifications that aim to fulfil these goals [6].

Important examples include data publication with rich metadata (e.g. Zenodo [7]), domain-specific data deposition (e.g., PDB [8]) and following practices for reproducible research software [9] (e.g. use of containers). While these platforms are useful, experience has shown that it is important to put greater emphasis on the interconnection of the multiple artefacts that make up the research process [10].

The notion of **Research Objects** [11] were introduced to address this connectivity as semantically rich aggregations of (potentially distributed) resources that provide a layer of structure over a research study and are delivered in a machine-readable format. Research Object combines the ability to bundle multiple types of artefacts together, such as spreadsheets, code, examples, and figures; augmented by relationships that describe their context (e.g. a CSV being used by a script, a figure being a result of a workflow, etc.). This provides a compelling vision as an approach for implementing FAIR data. However, existing Research Object implementations require a large technology stack [12], are typically tailored to a particular platform and are also not easily usable by end-users.

To address this gap, a new community came together [13] to develop **RO-Crate** — an *approach to package and aggregate research artefacts with their metadata and relationships*. The aim of this paper is to introduce RO-Crate and assess it as a strategy for making multiple types of research artefacts FAIR. Specifically, the contributions of this paper are as follows:

1. an introduction to RO-Crate, its purpose and context;
2. a guide to the RO-Crate community and tooling;
3. an exemplar usage of RO-Crate demonstrating its value as connective tissue for different artefacts from different communities.

The rest of this paper is organised as follows. We first describe RO-Crate, the assumptions underlying it, and define RO-Crate technically and formally. We then proceed to introduce the community and tooling. We move to analyse RO-Crate with respect to usage in a diverse set of domains. Finally, we present related work and conclude with some remarks including RO-Crate highlights and future work.

RO-Crate

RO-Crate aims to provide an approach to packaging research artefacts with their metadata that can be easily adopted. To illustrate this, let us imagine a research paper reporting on the sequence analysis of proteins obtained from an experiment on mice. The sequence output files, sequence analysis code, resulting data and reports summarising statistical measures or outputs are all important and inter-related research outputs, and consequently would ideally all be co-located in a directory and accompanied with their corresponding metadata. In reality, some of the artefacts (e.g. data or software) will be recorded as external references, not necessarily captured in a FAIR way. This directory, along with the relationships between its constituent digital artefacts, is what the RO-Crate model aims to represent, linking together all the elements pertaining to an experiment and required for its reproducibility.

The question then arises as to how the directory with all this material should be packaged in a manner that is accessible and usable by others. This means programmatically and automatically accessible by machines and human readable. A de facto approach to sharing collections of resources is through compressed archives (e.g. a zip file). This solves the problem of “packaging”, but it does not guarantee downstream access to all artefacts in a programmatic fashion, or the role of each file in that particular research. Both features, the ability to automatically access and reason about an object, are crucial and lead to the need for explicit metadata about the contents of the folder, describing each and linking them together.

Examples of metadata descriptions across a [wide range of domains](#) abound within the literature, both in research data management (?cite) and within library and information systems (?cite). However, many of these approaches require knowledge of metadata schemas, particular annotation systems, or the use of complex software stacks.

Indeed, particularly within research, these requirements have led to a lack of adoption and growing frustration with current tooling and specifications [14].

RO-Crate seeks to address this complexity by:

1. being conceptually simple and easy to understand for developers;
2. strong, easy tooling and integration into community projects
3. providing a strong and opinionated guide regarding current best practices;
4. adopting de-facto standards that are widely used on the Web.

In the following sections we demonstrate how the RO-Crate specification and ecosystem achieves these goals.

Conceptual Definition

A key premise of RO-Crate is the existence of a wide variety of resources on the Web that can help describe research. As such, RO-Crate relies on the Linked Data principles [15]. Figure 1 shows the main conceptual elements involved in an RO-Crate; an RO-Metadata File (top) describes the research object using structured metadata including external references, coupled with the contained artefacts (bottom) bundled and described by the RO-Crate.

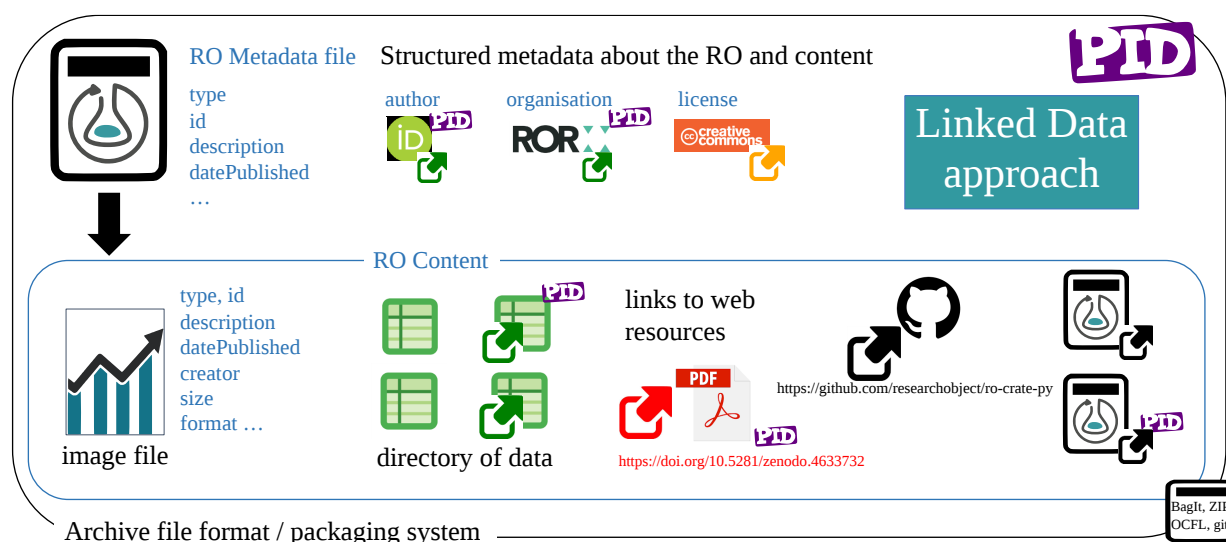


Figure 1: Conceptual overview of RO-Crate. A Persistent Identifier (PID) [16] points to a Research Object (RO), which may be archived using different packaging approaches like BagIt [17], OCFL [18], git or ZIP. The RO is described within a , providing identifiers for using ORCID, using ROR and licences such as Creative Commons using SPDX identifiers. The RO-Crate content is further described with additional metadata. Data can be embedded files and directories, as well as links to external web resources, PIDs [16] and nested RO-Crates.

Linked Data as a foundation

The **Linked Data** principles [19] (use of IRIs to identify resources (i.e. artefacts), resolvable via HTTP, enriched with metadata and linked to each other) are core to RO-Crate; therefore **IRIs**¹ are used to identify an RO-Crate, its constituent parts and metadata descriptions, and the properties and classes used in the metadata.

RO-Crates are *self-described*; and follow the Linked Data principles to describe all of their resources in both human and machine readable manner. Hence, resources are identified using global identifiers where possible; and relationships between two resources are defined with links.

The foundation of Linked Data and shared vocabularies also means multiple RO-Crates and other Linked Data resources can be indexed, combined, queried or transformed using existing Semantic Web technologies such as

[SPARQL](#) and knowledge graph triple stores.

Even though an RO-Crate is not required to be published on the Web, this use of mature web technologies means its developers and consumers are not restricted to the Research Object aspects that have already been specified by the RO-Crate community, but can extend and integrate in multiple standardized ways.

RO-Crate is a self-described container

An [RO-Crate is defined](#) as a self-described **Root Data Entity** that describes and contains *data entities*, which are further described using *contextual entities*. A **data entity** is either a *file* (i.e. a set of bytes stored on disk somewhere) or a *directory* (i.e. dataset of named files and other directories). A file does not need to be stored inside the RO-Crate root, it can be referenced via a PID/IRI. A **contextual entity** exists outside the information system (e.g. a Person, a workflow language) and is defined by its metadata. The representation of a **data entity** as a set of bytes makes it possible to store a variety of research artefacts including not only data but also, for instance, software and text.

Any particular IRI might appear as a contextual entity in one RO-Crate and as a data entity in another; their distinction lies in the fact that data entities can be considered to be *contained* or captured by the RO-Crate (*RO Content* in [1](#)), while contextual entities mainly *explain* the RO-Crate, although this distinction is not a formal requirement.

The Root Data Entity is a directory, the RO-Crate root, identified by the presence of the **RO-Crate Metadata File** (`ro-crate-metadata.json`) (Figure top). This is a JSON-LD file that describes the RO-Crate, its content and related metadata using Linked Data. JSON-LD is a W3C standard RDF serialisation that has become popular as it is easy to read by humans while also offers some advantages for data exchange on the Internet. JSON-LD is the preferred and widely supported format by RO-Crate tools and community.

The minimal [requirements for the root data entity metadata](#) are `name`, `description` and `datePublished`, as well as a contextual entity identifying its `license` — additional metadata are frequently added to entities depending on the purpose of the particular RO-Crate.

RO-Crate can be stored, transferred or published in multiple ways, e.g. BagIt [\[17\]](#), Oxford Common File Layout [\[18\]](#) (OCFL), downloadable ZIP archives in Zenodo or through dedicated online repositories, as well as published directly on the Web, e.g. using [GitHub Pages](#). Combined with Linked Data identifiers, this caters for a diverse set of storage and access requirements across different scientific domains, from metagenomics workflows producing hundreds of gigabytes of genome data to cultural heritage records with access restrictions for personally identifiable data. Specific [RO-Crate profiles](#) may constrain serialization and publication expectations, and require additional contextual types and properties.

Data Entities are described using Contextual Entities

RO-Crate distinguishes between [data and contextual entities](#) in a similar way to HTTP terminology's early attempt to separate *information* (data) and *non-information* (contextual) resources [\[21\]](#). Data entities are usually files and directories located by relative IRI references within the RO-Crate Root, but they can also be Web resources or restricted data identified with absolute IRIs.

As both types of entities are identified by IRIs, their distinction is allowed to be blurry; data entities can be located anywhere and be complex, while contextual entities can have a Web presence beyond their description inside the RO-Crate. For instance `https://orcid.org/0000-0002-1825-0097` is primarily an identifier for a person, but secondarily also a web page and their academic work. It follows that an RO-Crate should include a contextual entity that describes that person.

Any particular IRI might appear as a contextual entity in one RO-Crate and as a data entity in another; their distinction lies in the fact that data entities can be considered to be *contained* or captured by the RO-Crate, while contextual entities mainly *explain* the RO-Crate and its entities.

Figure shows a UML view of RO-Crate, highlighting the different types of data entities and contextual entities that can be aggregated and related. While an RO-Crate would usually contain one or more data entities (`hasPart`), it may also be a pure aggregation of contextual entities (`mentions`).

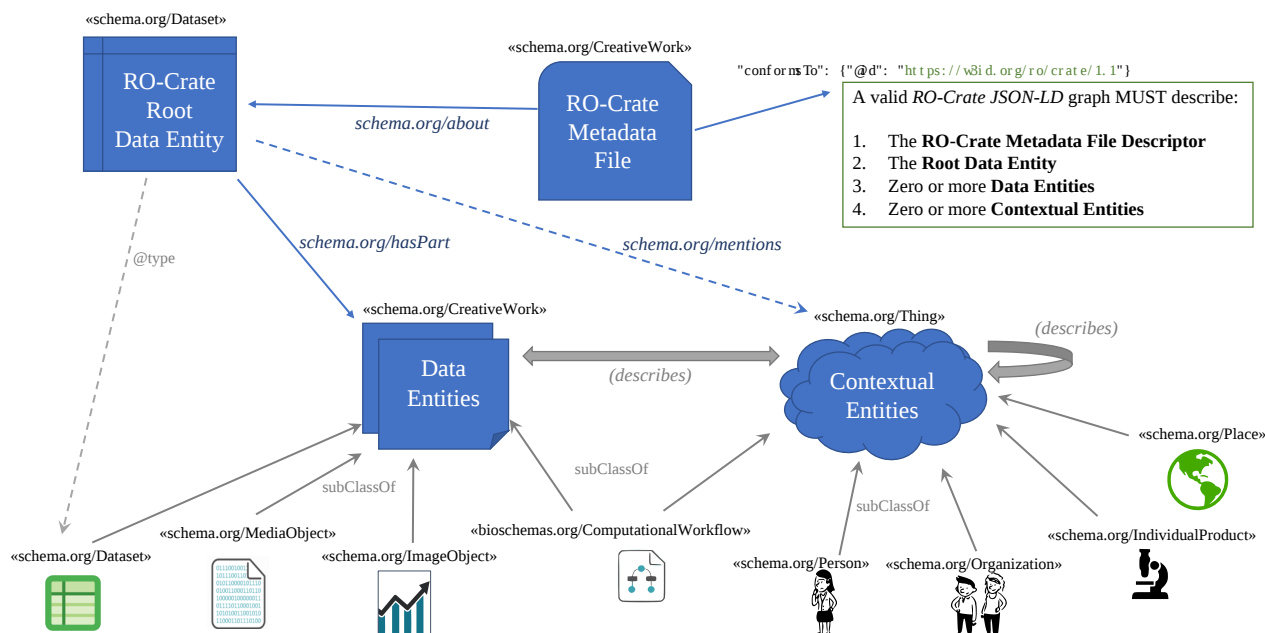


Figure 2: UML model view of RO-Crate. The RO-Crate Metadata File conforms to a version of the specification; and contains a JSON-LD graph that describes the entities that make up the RO-Crate. The RO-Crate Root Data Entity represent the Research Object as a dataset. The RO-Crate aggregates *data entities* (`hasPart`) which are further described using *contextual entities*. Multiple types and relations from Schema.org allow annotations to be more specific, including figures, nested datasets, computational workflows, people, organisations, instruments and places. Contextual entities not otherwise cross-referenced from other entities' properties (*describes*) can be grouped under the root entity (*mentions*).

Guide through Recommended Practices

RO-Crate as a specification aims to build a set of recommended practices on how to practically apply existing standards in a common way to describe research outputs and their provenance, without having to learn each of the underlying technologies in detail.

As such, the [RO-Crate 1.1](#) specification [22] can be seen as an opinionated and example-driven guide to writing [Schema.org](#) [23]) metadata as JSON-LD [24], which leaves it open for implementers to include additional metadata using other Schema.org types and properties, or even additional Linked Data vocabularies/ontologies or their own ad-hoc terms.

However the primary purpose of the RO-Crate specification is to assist developers in leveraging Linked Data principles for the focused purpose of describing Research Objects in a structured language, while reducing the steep learning curve otherwise involved in Semantic Web adaptation, like ontology development and selection, identifiers, namespaces, and RDF serialization choices.

Ensuring Simplicity

One aim of RO-Crate is to be conceptually simple. This simplicity has been repeatedly checked and confirmed through a community review process. For instance, in the discussion on supporting [ad-hoc vocabularies](#) in RO-Crate, the community explored potential Linked Data solutions. The conventional wisdom in [RDF best practice](#)

is to establish a vocabulary with a new URI namespace, formalised using [RDF Schema](#) or [OWL](#) ontologies. However, this may seem excessive learning curve for non-experts in semantic knowledge representation, and the RO-Crate community instead agreed on a dual lightweight approach: (i) [Document](#) how projects with their own web-presence can make a pure HTML-based vocabulary, and (ii) provide a community-wide PID namespace under <https://w3id.org/ro/terms/> that redirect to simple CSV files [maintained in GitHub](#).

To further verify this idea, we have formalised the RO-Crate definition (see *Appendix A*). An important result of this exercise is that the underlying data structure of RO-Crate, although conceptually a graph, is represented as a depth-limited tree. This formalisation also emphasises the *boundedness* of the structure; namely, the fact that elements are specifically identified as being either semantically *contained* by the RO-Crate (`hasPart`) or mainly referenced (`mentions`) and typed as *external* to the Research Object (Contextual Entities). It is worth pointing out that this semantic containment can extend beyond the physical containment of files residing within the RO-Crate Root directory on a given storage system, as the RO-Crate data entities may include any data resource globally identifiable using IRIs.

Extensibility and RO-Crate profiles

The RO-Crate specification provides a core set of conventions to describe research outputs using types and properties applicable across scientific domains. However we have found that domain-specific use of RO-Crate will, implicitly or explicitly, form a specialized **profile** of RO-Crate; *a set of conventions, types and properties that one minimally can require and expect to be present in that subset of RO-Crates*. For instance, RO-Crates used for exchange of workflows will have to contain a data entity of type `ComputationalWorkflow`, or cultural heritage records should have a `contentLocation`.

These profiles allow further reliable programmatic consumption and generation of RO-Crates, Following the RO-Crate mantra of guidance over strictness, profiles are mainly *duck-typing* rather than strict semantic types, but may also have corresponding machine-readable schemas at multiple levels (file formats, JSON, RDF shapes, RDFS/OWL semantics).

The next version of the RO-Crate specification 1.2 will define a [formalization](#) for publishing and declaring conformance to RO-Crate profiles, and optionally define a machine-readable profile as a *Profile Crate* — another RO-Crate that describe the profile and in addition can list schemas for validation, compatible software, accepting repositories, serialization/packaging formats, extension vocabularies, custom JSON-LD contexts and examples. (See for example the [Workflow RO-Crate profile](#))

In addition, there are sometimes existing domain-specific metadata formats already exist, but they are either not RDF-based (and thus challenging to add terms for in JSON-LD) or are at a different granularity level that might become overwhelming if represented directly in the RO-Crate Metadata file (e.g. W3C PROV bundle detailing a workflow run [25]). RO-Crate allow such alternative metadata files to co-exist, and be described as data entities with references to the standards and vocabularies they conform to, enabling their programmatic consumption even where no filename or file extension conventions have emerged for those metadata formats.

Section 4 *Profiles of RO-Crate in use* examines the observed specialization of RO-Crate use in several domains and their emerging profiles.

Technical implementation of the RO-Crate model

The RO-Crate conceptual model has been realised using JSON-LD and Schema.org in a prescriptive form as discussed in the previous sections. These technical choices were made to caters for simplicity.

[JSON-LD](#) [24] provides a way to express Linked Data as a JSON structure, where a *context* provides mapping to RDF properties and classes. While JSON-LD cannot map arbitrary JSON structures to RDF, we found it does

lower the barrier as it follows a JSON structure, nowadays a common and popular format for data exchange on the Web.

However, JSON-LD alone has too many degrees of freedom and hidden complexities for software developers to reliably produce and consume without specialised expertise or large RDF software frameworks. A large part of the RO-Crate specification is therefore dedicated to describing JSON structures.

RO-Crate JSON-LD

RO-Crate mandates the use of [flattened, compacted JSON-LD](#) in the RO-Crate Metadata file `ro-crate-metadata.json`² where a single `@graph` array contains all the data and contextual entities in a flat list. An example can be seen in the JSON-LD snippet in listing 1 below, describing a simple RO-Crate containing two datasets (data1.txt and data2.txt):


```

{ "@context": "https://w3id.org/ro/crate/1.1/context",
  "@graph": [
    { "@id": "ro-crate-metadata.json",
      "@type": "CreativeWork",
      "conformsTo": {"@id": "https://w3id.org/ro/crate/1.1"},
      "about": {"@id": "./"}
    },
    { "@id": "./",
      "@type": "Dataset",
      "name": "A simplified RO-Crate",
      "author": {"@id": "#alice"},
      "license": {"@id": "https://spdx.org/licenses/CC-BY-4.0"},
      "hasPart": [
        {"@id": "survey-responses-2019.csv"},
        {"@id": "https://example.com/pics/5707039334816454031_o.jpg"}
      ]
    },
    { "@id": "survey-responses-2019.csv",
      "@type": "File",
      "author": {"@id": "#alice"}
    },
    { "@id": "https://example.com/pics/5707039334816454031_o.jpg",
      "@type": ["File", "ImageObject"],
      "contentLocation": {"@id": "http://sws.geonames.org/8152662"},
      "author": {"@id": "https://orcid.org/0000-0002-1825-0097"}
    },
    { "@id": "#alice",
      "@type": "Person",
      "name": "Alice"
    },
    { "@id": "https://orcid.org/0000-0002-1825-0097",
      "@type": "Person",
      "name": "Josiah Carberry"
    },
    { "@id": "http://sws.geonames.org/8152662/",
      "@type": "Place",
      "name": "Catalina Park"
    },
    { "@id": "https://spdx.org/licenses/CC-BY-4.0",
      "@type": "CreativeWork",
      "name": "Creative Commons Attribution 4.0"
    }
  ]
}

```

Listing 1: Simplified³ RO-Crate metadata file showing the flattened compacted JSON-LD `@graph` array containing the data entities and contextual entities, cross-referenced using [???]. The `ro-crate-metadata.json` entity

declares conformance with the RO-Crate specification using a versioned persistent identifier, further RO-Crate descriptions are on the root data entity ./ or any of the referenced data or contextual entities, as exemplified by the ImageObject referencing contextual entities for contentLocation and author that differs from that of the overall RO-Crate. While Person entities ideally are identified with ORCID PIDs as for Josiah, in contrast #alice is here an RO-Crate local identifier, highlighting the pragmatic “just enough” Linked Data approach

In this flattened profile of JSON-LD, each `{entity}` under `@graph` represents the RDF triples with a common subject (`@id`), mapped properties like `hasPart`, and objects — as either literal `"string"` values, referenced `{objects}` (which properties are listed in its own entity), or a JSON `[list]` of these. If processed as JSON-LD, this forms an RDF graph by matching the `@id` IRIs and applying the `@context` mapping to schema.org terms.

Flattened JSON-LD

When JSON-LD 1.0 [24] was proposed, one of the motivations was to seamlessly apply an RDF nature on top of regular JSON as frequently used by Web APIs. JSON objects in APIs are frequently nested with objects at multiple levels, and the perhaps most common form of JSON-LD is the [compacted form](#) which follows this expectation ([JSON-LD 1.1](#) further expands these capabilities, e.g. allowing nested `@context` definitions).

While this feature of JSON-LD can be seen as a way to “hide” its RDF nature, we found that the use of nested trees (e.g. a `Person` entity appearing as `author` of a `File` which nests under a `Dataset` with `hasPart`) counter-intuitively forces consumers to consider the JSON-LD as an RDF Graph, since an identified `Person` entity can appear at multiple and repeated points of the tree (e.g. author of multiple files), necessitating node merging or duplication, which can become complicated as this approach also invites the use of *blank nodes* (entities missing `@id`).

By comparison, a single flat `@graph` array approach as preferred by RO-Crate means that applications can process and edit each entity as pure JSON by a simple lookup based on `@id`. At the same time, lifting all entities to the same level emphasises the principle that describing the context and provenance is just as important as describing the data, and the requirement of `@id` of every entity forces RO-Crate generators to consciously [consider existing IRIs and identifiers](#).

JSON-LD context

In JSON-LD, the `@context` is a reference to another JSON-LD document that provides mapping from JSON keys to Linked Data term IRIs, and can enable various JSON-LD directives to cater for customized JSON structures for translating to RDF.

RO-Crate reuses Schema.org vocabulary terms and IRIs, but provides its own versioned [JSON-LD context](#), which has a flat list with the mapping from JSON-LD keys to their URI equivalents (e.g. `author` maps to <http://schema.org/author>).

The rationale behind this decision is to support JSON-based RO-Crate applications that are largely unaware of JSON-LD, that still may want to process the `@context` to find or add Linked Data definitions of otherwise unknown properties and types. Not reusing the official Schema.org context means RO-Crate is also able to map in additional vocabularies where needed, namely the *Portland Common Data Model* (PCDM) [26] for repositories and Bioschemas [27] for describing computational workflows. RO-Crate profiles may [extend](#) the `@context` to re-use additional domain-specific ontologies.

Similarly, while the schema.org context has `"@type": "@id"` annotations for object properties, RO-Crate JSON-LD distinguishes explicitly between references to other entities (`{"@id": "#alice"}`) and string values (`"Alice"`) — meaning RO-Crate applications can find references for corresponding entities and IRIs

without parsing the `@context` to understand a particular property. Notably this is exploited by the `ro-crate-html-js` [28] tool to provide reliable HTML rendering for otherwise unknown properties and types.

RO-Crate Community

The RO-Crate conceptual model, implementation and best practices are developed by a growing community of researchers, developers and publishers. The RO-Crate community is a key aspect of its effectiveness in making research artefacts FAIR. Fundamentally, the community provides the overall context of the implementation and model and ensures its interoperability.

The RO-Crate community consists of:

1. a diverse set of people representing a variety of stakeholders;
2. a set of collective norms;
3. an open platform that facilitates communication (GitHub, Google Docs, monthly teleconferences).

People

The initial concept of RO-Crate was formed at the first Workshop on Research Objects ([RO2018](#)), held as part of the IEEE conference on eScience. This workshop followed up on considerations made at a [Research Data Alliance \(RDA\) meeting on Research Data Packaging](#) that found similar goals across multiple data packaging efforts [13]: simplicity, structured metadata and the use of JSON-LD.

An important outcome of discussions that took place at RO2018 was the conclusion that the original Wf4Ever Research Object ontologies [12], in principle sufficient for packaging research artefacts with rich descriptions, were, in practice, considered inaccessible for regular programmers (e.g. web developers) and in danger of being incomprehensible for domain scientists due to their reliance on Semantic Web technologies and other ontologies.

DataCrate [29] was presented at RO2018 as a promising lightweight alternative approach, and an agreement was made by a group of volunteers to attempt building “RO Lite” as a combination of DataCrate’s implementation and Research Object’s principles.

This group, originally made up of library and Semantic Web experts, has subsequently grown to include domain scientists, developers, publishers and more. This perspective of multiple views led to the specification being used in a variety of domains, from bioinformatics and regulatory submissions to humanities and cultural heritage preservation.

The RO-Crate community is strongly engaged with the European-wide biology/bioinformatics collaborative e-Infrastructure, ELIXIR, [30], along with European Open Science Cloud (EOSC) projects including EOSC-Life and FAIRplus. RO-Crate has also established collaborations with Bioschemas, GA4GH, OpenAIRE and multiple H2020 projects.

A key set of stakeholders are developers; the RO-Crate community has made a point of attracting developers who can implement the specifications but, importantly, keeps “developer user experience” in mind. This means that the specifications are straightforward to implement and thus do not require expertise in technologies that are not widely deployed.

This notion of catering to “developer user experience” is an example of the set of norms that have developed and now define the community.

Norms

The RO-Crate community is driven by conventions or notions that are prevalent within it but not formalised. Here, we distil what we as authors believe are the critical set of norms that have facilitated the development of RO-Crate and contributed to the ability for RO-Crate research packages to be FAIR. This is not to say that there are no other norms within the community or that everyone in the community holds these uniformly. Instead, what we emphasise is that these norms are helpful and also shaped by community practices.

1. Simplicity
2. Developer friendliness
3. Focus on examples and best practices rather than rigorous specification
4. Reuse “just enough” Web standards

A core norm of RO-Crate is that of **simplicity**, which sets the scene for how we guide developers to structure metadata with RO-Crate. We focus mainly on documenting simple approaches to the most common use cases, such as authors having an affiliation. This norm also influences our take on **developer friendliness**; for instance, we are using the Web-native JSON format, allowing only a few of JSON-LD’s flexible Linked Data features. Moreover, the RO-Crate documentation is largely built up by **examples** showcasing **best practices**, rather than rigorous specifications. We build on existing **Web standards** that themselves are defined rigorously, which we utilise “**just enough**” in order to benefit from the advantages of Linked Data (e.g. extensions by namespaced vocabularies), without imposing too many developer choices or uncertainties (e.g. having to choose between the many RDF syntaxes).

While the above norms alone could easily lead to the creation of “yet another” JSON format, we keep the goal of **FAIR interoperability** of the captured metadata, and therefore follow closely FAIR best practices and current developments such as data citations, PIDs, open repositories and recommendations for sharing research outputs and software.

Open Platforms

The critical infrastructure that enables the community around RO-Crate is the use of open development platforms. This underpins the importance of open community access to supporting FAIR. Specifically, it is difficult to build and consume FAIR research artefacts without being able to access the specifications, understand how they are developed, know about any potential implementation issues, and discuss usage to evolve best practices.

The development of RO-Crate was driven by capturing documentation of real-life examples and best practices rather than creating a rigorous specification. At the same time, we agreed to be opinionated on the syntactic form to reduce the jungle of implementation choices; we wanted to keep the important aspects of Linked Data to adhere to the FAIR principles while retaining the option of combining and extending the structured metadata using the existing Semantic Web stack, not just build “yet another” standalone JSON format.

Further work during 2019 started adapting the DataCrate documentation through a more collaborative and exploratory *RO-Lite* phase, initially using Google Docs for review and discussion, then moving to GitHub as a collaboration space for developing what is now the RO-Crate specification, [maintained as Markdown](#) in GitHub Pages and published through Zenodo.

In addition to the typical Open Source-style development with GitHub issues and pull requests, the RO-Crate Community now has two regular monthly calls, a Slack channel and a mailing list for coordinating the project, and many of its participants collaborate on RO-Crate at multiple conferences and coding events such as the ELIXIR BioHackathon. The community is jointly developing the RO-Crate specification and Open Source tools,

as well as providing support and considering new use cases. The [RO-Crate Community](#) is open for anyone to join, to equally participate under a code of conduct, and currently has more than 40 members.

RO-Crate Tooling

The work of the community led to the development of a number of tools for creating and using RO-Crates. Table 1 shows the current set of implementations. Reviewing this list, one can see that tools support commonly used programming languages, including Python, JavaScript, and Ruby. Additionally, these tools can be integrated into commonly used research environments; in particular, the command line (*ro-crate-html-js*). Furthermore, there are tools that cater to the end-user (*Describe*, *Workflow Hub*). For example, Describe was developed to help researchers of the Australian [Criminal Characters project](#) annotate historical prisoner records to gain greater insight into the history of Australia [31].

While the development of these tools is promising, our analysis of their maturity status shows that the majority of them are in the Beta stage. This is partly due to the fact that the RO-Crate specification itself only recently reached 1.0 status, in November 2019 [32]. Now that there is a fixed point of reference, and RO-Crate 1.1 (October 2020) [33] has stabilised based on feedback from application development, we expect to see a further increase in the maturity of these tools, along with the creation of new ones.

Given the stage of the specification, these tools have been primarily targeted to developers, essentially providing them with the core libraries for working with RO-Crates. Another target has been that of research data managers who need to manage and curate large amounts of data.

Tool Name	Targets	Language / Platform	Status	Brief Description
Describe [34]	Research Data Managers	NodeJS (Desktop)	RC	Interactive desktop application to create, update and export RO-Crates for different profiles
Describe Online [35]	Platform developers	NodeJS (Web)	Alpha	Web-based application to create RO-Crates using cloud storage
ro-crate-excel [36]	Data managers	JavaScript	Beta	Command-line tool to help create RO-Crates and HTML-readable rendering
ro-crate-html-js [28]	Developers	JavaScript	Beta	HTML rendering of RO-Crate
ro-crate-js [37]	Research Data Managers	JavaScript	Alpha	Library for creating/manipulating crates; basic validation code
ro-crate-ruby [38]	Developers	Ruby	Beta	Ruby library for reading/writing RO-Crate, with workflow support
ro-crate-py [39]	Developers	Python	Alpha	Object-oriented Python library for reading/writing RO-Crate
WorkflowHub [40]	Workflow users	Ruby	Beta	Workflow repository; imports and exports Workflow RO-Crate
Life Monitor [41]	Workflow developers	Python	Alpha	Workflow testing and monitoring service; Workflow Testing profile of RO-Crate
SCHeMa [42]	Workflow users	PHP	Alpha	Workflow execution using RO-Crate as exchange mechanism [43]
galaxy2cwl [44]	Workflow developers	Python	Alpha	Wraps Galaxy workflow as Workflow RO-Crate

Tool Name	Targets	Language / Platform	Status	Brief Description
Modern PARADISEC [45]	Repository managers	Platform	Beta	Cultural Heritage portal based on OCFL and RO-Crate
ONI express [46]	Repository managers	Platform	Beta	Platform for publishing data and documents stored in an OCFL repository via a web interface
ocfl-tools [47]	Developers	JavaScript (CLI)	Beta	Tools for managing RO-Crates in an OCFL repository
RO Composer [48]	Repository developers	Java	Alpha	REST API for gradually building ROs for given profile.
RDA maDMP Mapper [49]	Data Management Plan users	Python	Beta	Mapping between machine-actionable data management plans (maDMP) and RO-Crate [50]
Ro-Crate_2_maDMP [51]	Data Management Plan users	Python	Beta	Convert between machine-actionable data management plans (maDMP) and RO-Crate
CheckMyCrate [52]	Developers	Python (CLI)	Alpha	Validation according to Workflow RO-Crate profile

Table 1: Applications and libraries implementing RO-Crate, targeting different types of users across multiple programming languages. Status is indicative as assessed by this work (Alpha < Beta < Release Candidate (RC) < Release).

Profiles of RO-Crate in use

RO-Crate is fundamentally an infrastructure to help build FAIR research artefacts. In other words, the key question is whether RO-Crate can be used to share and (re)use research artefacts. Here we look at three research domains where RO-Crate is being applied: Bioinformatics, Regulatory Science and Cultural Heritages. In addition, we note how RO-Crate may have an important role as part of machine-actionable data management plans and institutional repositories.

From these varied uses of RO-Crate we observe a natural differences in their detail level and the type of entities described by the RO-Crate. For instance, on submission of an RO-Crate to a workflow repository, it is reasonable to expect the RO-Crate to contain at least one workflow, ideally with a declared licence and workflow language. Specific additional recommendations such as on identifiers is also needed to meet the emerging requirements of [FAIR Digital Objects](#). [Work has now begun](#) to formalise these different *profiles* of RO-Crates, which may impose additional constraints based on the needs of a specific domain or use case.

Bioinformatics workflows

[WorkflowHub.eu](#) is a European cross-domain registry of computational workflows, supported by European Open Science Cloud projects, e.g. [EOSC-Life](#), and research infrastructures including the pan-European bioinformatics network [ELIXIR](#) [30]. As part of promoting workflows as reusable tools, WorkflowHub includes documentation and high-level rendering of the workflow structure independent of its native workflow definition format. The rationale is that a domain scientist can browse all relevant workflows for their domain, before narrowing down their workflow engine requirements. As such, the WorkflowHub is intended largely as a registry of workflows already deposited in repositories specific to particular workflow languages and domains, such as UseGalaxy.eu [53] and Nextflow nf-core [54].

We here describe three different RO-Crate profiles developed for use with WorkflowHub.

Profile for describing workflows

Being cross-domain, WorkflowHub has to cater for many different workflow systems. Many of these, for instance Nextflow [55] and Snakemake [56], by virtue of their script-like nature, reference multiple neighbouring files typically maintained in a GitHub repository. This calls for a data exchange method that allows keeping related files together. WorkflowHub has tackled this problem by adopting RO-Crate as the packaging mechanism [57], typing and annotating the constituent files of a workflow and — crucially — marking up the workflow language, as many workflow engines use common file extensions like `*.xml` and `*.json`. Workflows are further described with authors, licence, diagram previews and a listing of their inputs and outputs. RO-Crates can thus be used for interoperable deposition of workflows to WorkflowHub, but are also used as an archive for downloading workflows, embedding metadata registered with the WorkflowHub entry and translated workflow files such as abstract Common Workflow Language (CWL) [58] definitions and diagrams [59].

RO-Crate acts therefore as an interoperability layer between registries, repositories and users in WorkflowHub. The iterative development between WorkflowHub developers and the RO-Crate community heavily informed the creation of the Bioschemas [27] profile for [Computational Workflows](#), which again informed the [RO-Crate 1.1 specification on workflows](#) and led to the RO-Crate Python library [39] and WorkflowHub's [Workflow RO-Crate profile](#), which, in a similar fashion to RO-Crate itself, recommends which workflow resources and descriptions are required. This co-development across project boundaries exemplifies the drive for simplicity and for establishing best practices.

Profile for recording workflow runs

RO-Crates in WorkflowHub have so far been focused on workflows that are ready to be run, and development of WorkflowHub is now creating a **Workflow Run RO-Crate profile** for the purposes of benchmarking, testing and executing workflows. As such, RO-Crate serves as a container of both a *workflow definition* that may be executed and of a particular *workflow execution with test results*.

This workflow run profile is a continuation of our previous work with capturing workflow provenance in a Research Object in CWLProv [25] and TavernaPROV [60]. In both cases, we used the PROV Ontology [59], including details of every task execution with all the intermediate data, which required significant workflow engine integration⁴.

To simplify from that approach, for this Workflow Run RO-Crate profile we will use a higher level [schema.org provenance](#) for the input/output boundary of the overall workflow execution. This *Level 1 workflow provenance* [25] can be expressed generally across workflow languages with minimal engine changes, with the option of more detailed provenance traces as separate PROV resources in the RO-Crate.

WorkflowHub has recently enabled minting of Digital Object Identifiers (DOIs), a PID commonly used for scholarly artefacts, for registered workflows, e.g. `10.48546/workflowhub.workflow.56.1` [62], lowering the barrier for citing workflows as computational methods along with their FAIR metadata – captured within an RO-Crate. While it is not an aim for WorkflowHub to be a repository of workflow runs and their data, RO-Crates of *exemplar workflow runs* serve as useful workflow documentation, as well as being an exchange mechanism that preserve FAIR metadata in a diverse workflow execution environment.

Profile for testing workflows

The value of computational workflows, however, is potentially undermined by the “collapse” over time of the software and services they depend upon: for instance, software dependencies can change in a non-backwards-compatible manner, or active maintenance may cease; an external resource, such as a reference index or a

database query service, could shift to a different URL or modify its access protocol; or the workflow itself may develop hard-to-find bugs as it is updated. This can take a big toll on the workflow's reusability and on the reproducibility of any processes it evokes.

For this reason, WorkflowHub is complemented by a monitoring and testing service called LifeMonitor[41], also supported by EOSC-Life. LifeMonitor's main goal is to assist in the creation, periodic execution and monitoring of workflow tests, enabling the early detection of software collapse in order to minimise its detrimental effects. The communication of metadata related to workflow testing is achieved through the adoption of a [Workflow Testing RO-Crate profile](#) stacked on top of the *Workflow RO-Crate* profile. This further specialisation of Workflow RO-Crate allows to specify additional testing-related entities (test suites, instances, services, etc.), leveraging [RO-Crate's extension mechanism](#) through the addition of terms from custom namespaces.

In addition to showcasing RO-Crate's extensibility, the testing profile is an example of the format's flexibility and adaptability to the different needs of the research community. Though ultimately related to a computational workflow, in fact, most of the testing-specific entities are more about describing a protocol for interacting with a monitoring service than a set of research outputs and its associated metadata. Indeed, one of LifeMonitor's main functionalities is monitoring and reporting on test suites running on existing Continuous Integration (CI) services, which is described in terms of service URLs and job identifiers in the testing profile. In principle, in this context, data could disappear altogether, leading to an RO-Crate consisting entirely of contextual entities. Such an RO-Crate acts more as an exchange format for communication between services (WorkflowHub and LifeMonitor) than as an aggregator for research data and metadata, providing a good example of the format's high versatility.

Regulatory Sciences

[BioCompute Objects](#) (BCO) [63] is a community-led effort to standardise submissions of computational workflows to biomedical regulators. For instance, a genomics sequencing pipeline, as part of a personalised cancer treatment study, can be submitted to the US Food and Drugs Administration (FDA) for approval. BCOs are formalised in the standard IEEE 2791-2020 [64] as a combination of [JSON Schemas](#) that define the structure of JSON metadata files describing exemplar workflow runs in detail, covering aspects such as the usability and error domain of the workflow, its runtime requirements, the reference datasets used and representative output data produced.

BCOs provide a structured view over a particular workflow, informing regulators about its workings independently of the underlying workflow definition language. However, BCOs have only limited support for additional metadata⁵. For instance, while the BCO itself can indicate authors and contributors, and in particular regulators and their review decisions, it cannot describe the provenance of individual data files or workflow definitions.

As a custom JSON format, BCOs cannot be extended with Linked Data concepts, except by adding an additional top-level JSON object formalised in another JSON Schema. A BCO and workflow submitted by upload to a regulator will also frequently consist of multiple cross-related files. Crucially, there is no way to tell whether a given `*.json` file is a BCO file, except by reading its content and check for its `spec_version`.

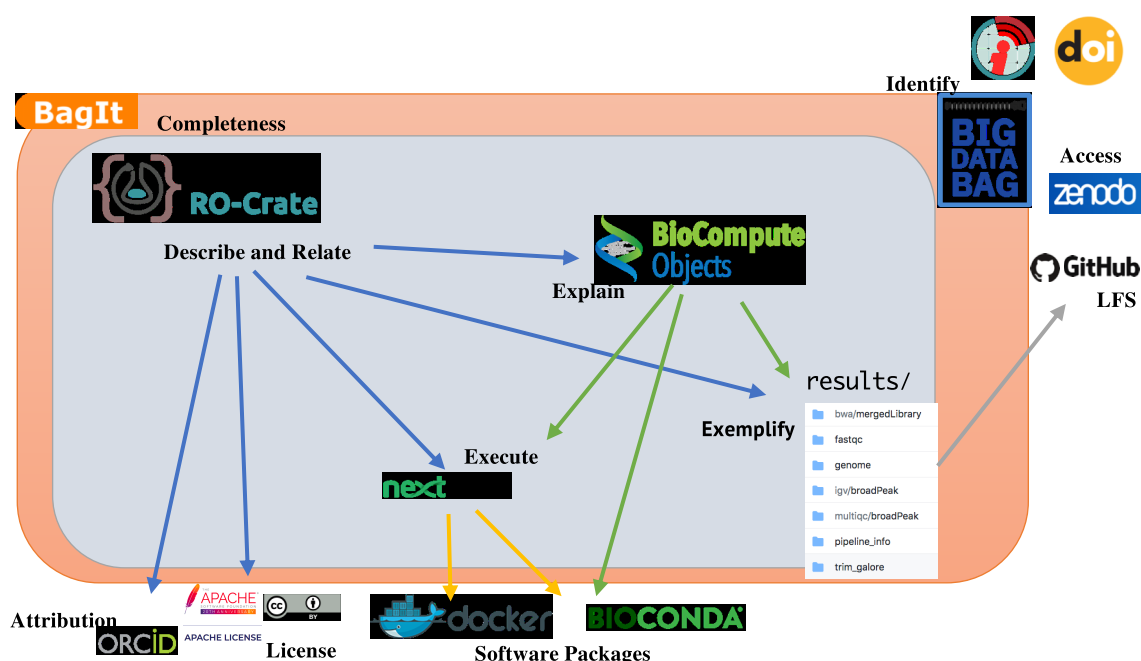
We can then consider how a BCO and its referenced artefacts can be packaged and transferred following FAIR principles. [BCO RO-Crate](#)[65], part of the BioCompute Object user guides, defines a set of best practices for wrapping a BCO with a workflow, together with its exemplar outputs in an RO-Crate, which then provides typing and additional provenance metadata of the individual files, workflow definition, referenced data and the BCO metadata itself.

Here the BCO is responsible for describing the *purpose* of a workflow and its run at an abstraction level suitable for a domain scientist, while the more open-ended RO-Crate describes the surroundings of the workflow,

classifying and relating its resources and providing provenance of their existence beyond the BCO. This emerging *separation of concerns* highlight how RO-Crate is used side-by-side of existing standards, even where there are apparent partial overlaps.

A similar separation of concerns can be found if considering the RO-Crate as a set of files, where the *transport-level* metadata, such as checksum of files, are [delegated to BagIt](#) manifests, a standard focusing on the preservation challenges of digital libraries[17]. As such, RO-Crates are not required to iterate all the files in their folder hierarchy, only those that benefit from being described.

Specifically, a BCO alone is insufficient for reliable re-execution of a workflow, which would need a compatible workflow engine depending on the workflow definition language, so IEEE 2791 recommends using Common Workflow Language [58] for interoperable pipeline execution. CWL itself relies on tool packaging in software containers using [Docker](#) or [Conda](#). Thus, we can consider BCO RO-Crate as a stack: transport-level manifests of files (BagIt), provenance, typing and context of those files (RO-Crate), workflow overview and purpose (BCO), interoperable workflow definition (CWL) and tool distribution (Docker).



Separation of Concerns in BCO RO-Crate. BioCompute Object (IEEE2791) is a JSON file that structurally explains the purpose and implementation of a computational workflow, for instance implemented in Nextflow, that installs the workflow’s software dependencies as Docker containers or BioConda packages. An example execution of the workflow shows the different kinds of result outputs, which may be external, using GitHub LFS to support larger data. RO-Crate gathers all these local and external resources, relating them and giving individual descriptions, for instance permanent DOI identifiers for reused datasets accessed from Zenodo, but also adding external identifiers to attribute authors using ORCID or to identify which licences apply to individual resources. The RO-Crate and its local files are captured in a BagIt whose checksum ensures completeness, combined with Big Data Bag [66] features to “complete” the bag with large external files such as the workflow outputs

Digital Humanities: Cultural Heritage

[PARADISEC](#) (the Pacific And Regional Archive for Digital Sources in Endangered Cultures) maintains a repository of more than 500,000 files documenting endangered languages across more than 16,000 items, collected over many years by researchers interviewing and recording native speakers across the region. As a proposed update of the 18 year old infrastructure, the [Modern PARADISEC demonstrator](#) has been [developed](#) to also help digitally preserve these artefacts using the [Oxford Common File Layout](#) (OCFL) for file consistency and RO-Crate for structuring and capturing the metadata of each item. The existing PARADISEC data collection has been ported and captured as RO-Crates. A web portal then exposes the repository and its entries by indexing the RO-Crate metadata files using Elasticsearch as a “NoSQL” object database, presenting a domain-specific view of the items — the RO-Crate is “hidden” and does not change the user interface.

This use case takes advantage of several RO-Crate features and principles. Firstly, the transcribed metadata are now independent of the PARADISEC platform and can be archived, preserved and processed in its own right, using Schema.org vocabularies augmented with PARADISEC-specific terms. The lightweight infrastructure with RO-Crate as the holder of itemised metadata in regular files (organised using OCFL[18], with checksums for integrity checking and versioning) also gives flexibility for future developments and maintenance; for example, potentially using Linked Data software such as a graph database, queried using SPARQL triple patterns across RO-Crates, or a “last resort” fallback to the generic RO-Crate HTML preview [28], which can be hosted as static files by any web server, in line with the approach taken by the Endings Project⁶.

Machine-actionable Data Management Plans

Machine-actionable Data Management Plans (maDMPs) have been proposed as an improvement to automate FAIR data management tasks in research [67], e.g. by using PIDs and controlled vocabularies to describe what happens to data over the research life cycle [68]. The Research Data Alliance’s *DMP Common Standard* for maDMPs [69] is one such formalisation for expressing maDMPs, which can be expressed as Linked Data using the DMP Common Standard Ontology [70], a specialisation of the W3C Data Catalog Vocabulary (DCAT) [71]. RDA maDMPs are usually expressed using regular JSON, conforming to the DMP JSON Schema.

A mapping has been produced between Research Object Crates and Machine-actionable Data Management Plans [50], implemented by the RO-Crate {RDA maDMP Mapper [49]. A similar mapping has been implemented by `RO-Crate_2_ma-DMP` [51]. In both cases, a maDMP can be converted to a RO-Crate, or vice versa. In [50] this functionality caters for two use cases:

1. Start a skeleton data management plan based on an existing RO-Crate dataset, e.g. from an RO-Crate from WorkflowHub.
2. Instantiate an RO-Crate based on a data management plan.

An important difference here is that data management plans are (ideally) written in advance of data production, while RO-Crates are typically created to describe data after it has been generated. This approach shows the importance of *templating* to make both tasks more automatable and achievable, and how RO-Crate can fit into earlier stages of the research life cycle.

Institutional data repositories – Harvard Data Commons

The concept of a Data Commons for research collaboration was originally defined as “cyber-infrastructure that co-locates data, storage, and computing infrastructure with commonly used tools for analysing and sharing data to create an interoperable resource for the research community” [72]. More recently, it was established to integrate active data-intensive research with data management and archival best practices. It facilitates research by providing computational infrastructure where researchers can use, share and store data, software, workflows and other digital artefacts used in their studies. Furthermore, the Commons feature tools and services, such as computation clusters and storage for scalability, data repositories for disseminating and preserving regular, but also large or sensitive datasets, and other research assets. Multiple initiatives were undertaken to create Data Commons on national, research, and institutional levels. For example, the [Australian Research Data Commons \(ARDC\)](#) [73] is a national initiative that enables local researchers and industries to access computing infrastructure, training, and curated datasets for data-intensive research. NCI’s [Genomic Data Commons](#) (GDC) [74] provides the cancer research community with access to a vast volume of genomic and clinical data. Initiatives such as [Research Data Alliance \(RDA\) Global Open Research Commons](#) propose standards on the implementation of Data Commons to avoid them becoming “data silos” and enable interoperability from one Data Commons to another.

Harvard Data Commons [75] aims to address data access and reuse challenges of cross-disciplinary research within a research institution. It brings together multiple institutional schools, libraries, computing centres and the [Harvard Dataverse data repository](#). [Dataverse](#) [76] is a free and open-source software platform to archive, share and cite research data. The Harvard Dataverse repository is the largest of 70 installations worldwide, containing over 100K datasets with about 1M data files. Toward the goal of facilitating collaboration and data discoverability and management within the university, Harvard Data Commons has the following primary objectives:

1. integrating Harvard Research Computing with Harvard Dataverse by leveraging Globus endpoints [77] that will allow an automatic transfer of large datasets to the repository. In some cases, only the metadata will be transferred while the data stays stored in remote storage;
2. supporting advanced research workflows and providing packaging options for assets such as code and workflows in the Harvard Dataverse repository to enable reproducibility and reuse, and
3. integrating repositories supported by Harvard, which are [DASH](#), the open access institutional repository, the Digital Repository Services (DRS) for preserving digital asset collections, and the Harvard Dataverse.

Particularly relevant to this paper is the second objective of the Harvard Data Commons, which aims to support the deposit of research artefacts to Harvard Dataverse with sufficient information in the metadata to allow their future reuse (Figure~). Considering the requirements of incorporating data, code, and other artefacts from various institutional infrastructures, Harvard Data Commons is currently working on RO-Crate adaptation. The RO-Crate metadata provides the necessary structure to make all research artefacts FAIR. The Dataverse software already has extensive support for metadata, including the Data Documentation Initiative (DDI), Dublin Core, DataCite, and Schema.org. Incorporating RO-Crate, which has the flexibility to describe a wide range of research resources, will facilitate their seamless transition from one infrastructure to the other within the Harvard Data Commons.

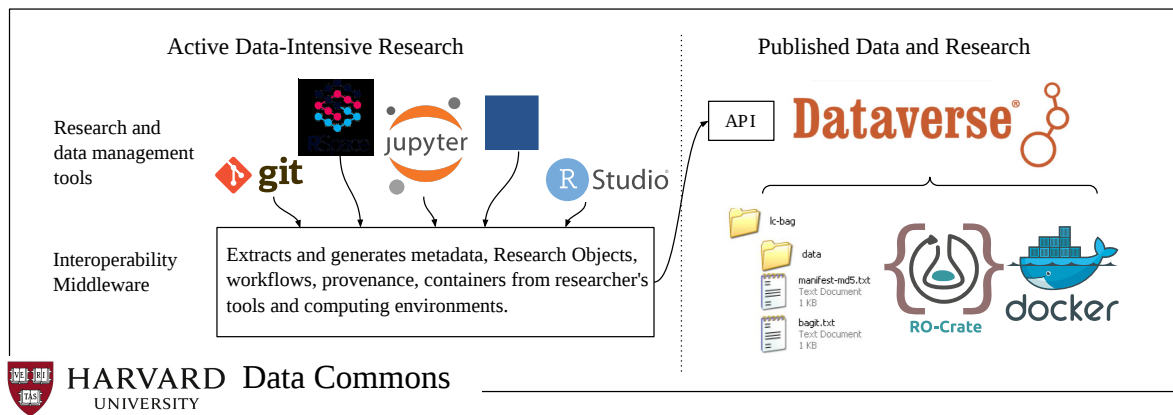


Figure 3: One aspect of Harvard Data Commons. Automatic encapsulation and deposit of artefacts from data management tools used during active research at the Harvard Dataverse repository.

Even though the Harvard Data Commons is specific to Harvard University, the overall vision and the three objectives can be abstracted and applied to other universities or research organisations. The Commons will be designed and implemented using standards and commonly-used approaches to make it interoperable and reusable by others.

Related Work

With the increasing digitisation of research processes, there has been a significant call for the wider adoption of interoperable sharing of data and its associated metadata. For a comprehensive overview and recommendations, in particular for data, we refer to [78], which highlights the wide variety of metadata and documentation that the literature prescribes for enabling data reuse.

Here we focus on approaches for bundling research artefacts along with their metadata. This notion of publishing compound objects for scholarly communication has a long history behind it [79] [80], but recent approaches have followed three main strands: 1) publishing to centralised repositories; 2) packaging approaches similar to RO-Crate; and 3) bundling the computational workflow around a scientific experiment.

Bundling and Packaging Digital Research Artefacts

The challenge of describing computational workflows was one of the main motivations for the early proposal of *Research Objects* (RO) [11] as first-class citizens for sharing and publishing. The RO approach involves bundling datasets, workflows, scripts and results along with traditional dissemination materials like journal articles and presentations, forming a single package. Crucially, these resources are not just gathered, but also individually typed, described and related to each other using semantic vocabularies. As pointed out in [11] an open-ended *Linked Data* approach is not sufficient for scholarly communication: a common data model is also needed in addition to common and best practices for managing and annotating lifecycle, ownership, versioning and attributions.

Considering the FAIR principles [5], we can say with hindsight that the initial RO approaches strongly targeted *Interoperability*, with a particular focus on the reproducibility of *in-silico experiments* involving computational workflows and the reuse of existing RDF vocabularies.

The first implementation of Research Objects for sharing workflows in myExperiment [81] was based on RDF ontologies [82], building on Dublin Core, FOAF, SIOC, Creative Commons and OAI-ORE to form myExperiment ontologies for describing social networking, attribution and credit, annotations, aggregation packs, experiments, view statistics, contributions, and workflow components [83].

This initially workflow-centric approach was further formalized as the Wf4Ever Research Object Model [12], which is a general-purpose research artefact description framework, based on existing ontologies (FOAF, Dublin Core Terms, OAI-ORE and AO/OAC precursors to the W3C Web Annotation Model [84]), adding specializations for workflow models and executions based on W3C PROV-O [85]. The Research Object statements are saved in a *manifest* (the OAI-ORE *resource map*), with additional annotation resources containing user-provided details such as title and description.

We can claim that one barrier for adoption of the Wf4Eer Research Object model for general packaging digital research artefacts was exactly this re-use of multiple existing vocabularies (FAIR principle I2: *Meta)data use vocabularies that follow FAIR principles*), itself a challenge [86], as developers had to navigate documentation of multiple overlapping ontologies in addition to facing the usual Semantic Web choices for RDF serialization formats, identifier minting and publishing resources on the Web.

Several later developments for Research Objects improved on this situation, such as ROHub used by Earth Sciences [87], which provides a interactive user-interface for making research objects, along with Research Object Bundle [61] (RO Bundle), which is a ZIP-archive embedding data files and a JSON-LD serialization of the manifest has mapping for a limited set of terms and was used for storing workflow run provenance (TavernaPROV [60]).

RO-Bundle evolved to [Research Object BagIt archives](#), a variant of RO Bundle as a BagIt archive [17], used by Big Data Bags [66], CWLProv [25] and WholeTale [88].

FAIR Digital Objects

FAIR Digital Objects (FDO) [89] have been proposed as a conceptual framework for making digital resources available in a Digital Objects (DO) architecture that encourages active use of the objects and their metadata. In

particular, an FDO has five parts: (i) The FDO *content*, bit sequences stored in an accessible repository; (ii) a *Persistent Identifier* (PID) such as a DOI that identifies the FDO and can resolve these parts; (iii) Associated rich *metadata*, as separate FDOs; (iv) Type definitions, also separate FDOs; (v) Associated *operations* for the given types. A Digital Object typed as a Collection aggregates other DOs by reference.

As an “[abstract protocol](#)”, DOs could be implemented in multiple ways. One suggested implementation is the [FAIR Digital Object Framework](#), based on HTTP and the Linked Data Principles. While there is agreement on using PIDs based on DOIs, consensus on how to represent common metadata, core types and collections as FDOs has not yet been reached. We argue that RO-Crate can play an important role for FDOs:

1. By providing a predictable and extensible serialisation of structured metadata.
2. By formalising how to aggregate digital objects as collections (and adding their context).
3. By providing a natural Metadata FDO in the form of the RO-Crate Metadata File.
4. By being based on Linked Data and schema.org vocabulary, meaning that PIDs already exist for common types and properties.

At the same time, it is clear that the goal of FDO is broader than that of RO-Crate; namely, FDOs are active objects with distributed operations, and add further constraints such as PIDs for every element. These features improve FAIR features of digital objects and are also useful for RO-Crate, but they also severely restrict the infrastructure that needs to be implemented and maintained in order for FDOs to remain available. RO-Crate, on the other hand, is more flexible: it can minimally be used within any file system structure, or ideally exposed through a range of Web-based scenarios. A *FAIR profile of RO-Crate* (e.g. enforcing PID usage) will fit well within a FAIR Digital Object ecosystem.

Packaging Workflows

The use of computational workflows, typically combining a chain of open source tools in an analytical pipeline, has gained prominence, in particular in the life sciences. Workflows may have initially been used to improve computational scalability, but they also assist in making computed data results FAIR [4], for instance by improving reproducibility [90], but also because programmatic data usage help propagate their metadata and provenance [91]. At the same time, workflows raise additional FAIR challenges, since they can be considered important research artefacts themselves, posing the problem of capturing and explaining the computational methods behind the analysis they perform [3].

Even when researchers follow current best practices for workflow reproducibility, [92] [90] the communication of outcomes through traditional academic publishing routes relying on a textual representation adds barriers that hinder reproducibility and FAIR use of the knowledge previously captured in the workflow.

As a real-life example, let us look at a metagenomics article [93] where the authors have gone to extraordinary efforts to document the individual tools that have been reused, including their citations, versions, settings, parameters and combinations. The *Methods* section is 2 pages in tight double-columns with 24 additional references, supported by the availability of data on an FTP server (60 GB) [94] and of open source code in GitHub [Finn-Lab/MGS-gut](#) [95], including the pipeline as shell scripts and associated analysis scripts in R and Python.

This attention to reporting detail for computational workflows is unfortunately not yet the norm, and although bioinformatics journals have strong *data availability* requirements, they frequently do not require authors to include or cite *software, scripts and pipelines* used for analysing and producing results [96] – rather, authors might be penalised for doing so [cite?] as it would detrimentally count against arbitrary limits on number of pages and references.

However detailed this additional information might be, another researcher who wants to reuse a particular computational method may first want to assess if the described tool or workflow is Re-runnable (executable at all), Repeatable (same results for original inputs on same platform), Reproducible (same results for original inputs with different platform or newer tools) and ultimately Reusable (similar results for different input data), Repurposable (reusing parts of the method for making a new method) or Replicable (rewriting the workflow following the method description). [97][98]

Following the textual description alone, researchers would be forced to jump straight to evaluate “Replicable” by rewriting the pipeline from scratch. This can be expensive and error-prone. They would firstly need to install all the software dependencies and download reference datasets. This can be a daunting task in and of itself, which may have to be repeated multiple times as workflows typically are developed at small scale on desktop computers, scaled up to local clusters, and potentially put into production using cloud instances, each of which will have different requirements for software installations.

In recent years the situation has been greatly improved by software packaging and container technologies like Docker and Conda, which have seen increased adoption in life sciences [99] thanks to collaborative efforts such as BioConda [100] and BioContainers [101], and support by Linux distributions (e.g. Debian Med [102]). As of May 2021, more than 7000 software packages are available [in BioConda alone] (<https://anaconda.org/bioconda/>), and 9000 containers [in BioContainers](#). Docker and Conda have gained integration in workflow systems such as Snakemake [56], Galaxy [103] and Nextflow [55], meaning a downloaded workflow definition can now be executed on a “blank” machine (except for the workflow engine) with the underlying analytical tools installed on demand – but even here there is a reproducibility challenge, for instance [Docker Hub’s retention policy will expire container images after 6 months](#), or lack of recording versions of transitive dependencies of Conda packages could cause incompatibilities if the packages are subsequently updated. Except for brief metadata in their repositories, these containers and packages do not capture any semantic relationships of their content – rather their opaqueness and wrapping of arbitrary binary tools makes such relationships harder to find.

From this we see that computational workflows are themselves complex digital objects that needs to be recorded not just as files, but in the context of their execution environment, dependencies and analytical purpose in research – as well as other metadata (e.g. version, license, attribution and identifiers).

Conclusion

RO-Crate provides an approach to packaging digital research artefacts with structured metadata, assisting developers and researchers to produce and consume FAIR archives of their research.

As a set of best practice recommendations, developed by an open and broad community, RO-Crate shows how to use “just enough” Linked Data standards in a consistent way, with structured metadata using a rich base vocabulary that can cover general-purpose contextual relations, whilst retaining extensibility to domain- and application-specific uses.

The adoption of simple web technologies in the RO-Crate specification has helped a rapid development of a wide variety of supporting open source tools and libraries. RO-Crate fits into the larger landscape of open scholarly communication and FAIR Digital Object infrastructure, and can be integrated into data repository platforms. RO-Crate can be applied as a data/metadata exchange mechanism, assist in long-term archival preservation of metadata and data, or simply used at small-scale by individual researchers. Thanks to its strong community support, new and improved profiles and tools are continuously added to the RO-Crate tooling landscape, making it easier for adopters to find examples and support for their own use case.

Acknowledgements

This work has received funding from the European Commission's Horizon 2020 research and innovation programme for projects [BioExcel-2](#) (H2020-INFRAEDI-2018-1 823830), [IBISBA 1.0](#) (H2020-INFRAIA-2017-1-two-stage 730976), [PREP-IBISBA](#) (H2020-INFRADEV-2019-2 871118), [EOSC-Life](#) (H2020-INFRAEOSC-2018-2 824087), [SyntheSys+](#) (H2020-INFRAIA-2018-1 823827).

Björn Grüning is supported by [DataPLANT](#) (NFDI 7/1 – 42077441), part of the German National Research Data Infrastructure (NFDI), funded by the Deutsche Forschungsgemeinschaft (DFG).

Ana Trisovic is funded by the Alfred P. Sloan Foundation (grant number [P-2020-13988](#)). Harvard Data Commons is supported by an award from Harvard University Information Technology (HUIT).

Contributions

Author contributions to this article and the RO-Crate project according to the Contributor Roles Taxonomy [CASRAI CrEDiT](#) [104]:

Stian Soiland-Reyes

Conceptualization, Data curation, Formal Analysis, Funding acquisition, Investigation, Methodology, Project administration, Software, Visualization, Writing – original draft, Writing – review & editing

Peter Sefton

Conceptualization, Investigation, Methodology, Project administration, Resources, Software, Writing – review & editing

Mercè Crosas

Writing – review & editing

Leyla Jael Castro

Methodology, Writing – review & editing

Frederik Coppens

Writing – review & editing

José M. Fernández

Methodology, Software, Writing – review & editing

Daniel Garijo

Methodology, Writing – review & editing

Björn Grüning

Writing – review & editing

Marco La Rosa

Software, Methodology, Writing – review & editing

Simone Leo

Software, Methodology, Writing – review & editing

Eoghan Ó Carragáin

Investigation, Methodology, Project administration, Writing – review & editing

Marc Portier

Methodology, Writing – review & editing

Ana Trisovic

Software, Writing – review & editing

RO-Crate Community

Investigation, Software, Validation, Writing – review & editing

Paul Groth

Methodology, Supervision, Writing – original draft, Writing – review & editing

Carole Goble

Conceptualization, Funding acquisition, Methodology, Project administration, Supervision, Visualization, Writing – review & editing

We would also like to acknowledge contributions from:

Finn Bacall

Software, Methodology

Herbert Van de Sompel

Writing – review & editing

Ignacio Eguinoa

Software, Methodology

Nick Juty

Writing – review & editing

Oscar Corcho

Writing – review & editing

Stuart Owen

Writing – review & editing

Laura Rodríguez-Navas

Software, Visualization, Writing – review & editing

Formalizing RO-Crate in First Order Logic

Below is a formalization of the concept of RO-Crate as a set of relations using First Order Logic:

Language

Definition of language $\mathbb{L}_{\text{rocrate}}$:

```
 $\mathbb{L}_{\text{rocrate}} = \{ \text{Property}(p), \text{Class}(c), \text{Value}(x), \mathbb{R}, \mathbb{S} \}$   
 $\mathbb{D} = \mathbb{Iri}$   
 $\mathbb{Iri} \equiv \{ \text{IRIs as defined in RFC3987} \}$   
 $\mathbb{R} \equiv \{ \text{real or integer numbers} \}$   
 $\mathbb{S} \equiv \{ \text{literal strings} \}$ 
```

The domain of discourse is the set of \mathbb{Iri} identifiers [20] (notation `<http://example.com/>`), with additional descriptions using numbers \mathbb{R} (notation `13.37`) and literal strings \mathbb{S} (notation `"Hello"`).

From this formalized language $\mathbb{L}_{\text{rocrate}}$ we can interpret an RO-Crate in any representation that can gather these descriptions, their properties, classes, and literal attributes.

Minimal RO-Crate

Below we use $\mathbb{L}_{\text{rocrate}}$ to define a minimal⁸ RO-Crate:

$$\begin{aligned}
\text{ROCrate}(R) &\models \text{Root}(R) \wedge \text{Mentions}(R, R) \wedge \text{hasPart}(R, d) \wedge \\
&\quad \text{Mentions}(R, d) \wedge \text{DataEntity}(d) \wedge \\
&\quad \text{Mentions}(R, c) \wedge \text{ContextualEntity}(c) \\
\forall r \text{ Root}(r) &\Rightarrow \text{Dataset}(r) \wedge \text{name}(r, n) \wedge \text{description}(r, d) \wedge \\
&\quad \text{published}(r, \text{date}) \wedge \text{license}(e, l) \\
\forall e \forall n \text{ name}(e, n) &\Rightarrow \text{Value}(n) \\
\forall e \forall s \text{ description}(e, s) &\Rightarrow \text{Value}(s) \\
\forall e \forall d \text{ datePublished}(e, d) &\Rightarrow \text{Value}(d) \\
\forall e \forall l \text{ license}(e, l) &\Rightarrow \text{ContextualEntity}(l) \\
\text{DataEntity}(e) &\equiv \text{File}(e) \oplus \text{Dataset}(e) \\
\text{Entity}(e) &\equiv \text{DataEntity}(e) \vee \text{ContextualEntity}(e) \\
\forall e \text{ Entity}(e) &\Rightarrow \text{Class}(e) \\
\text{Mentions}(R, s) &\models \text{Relation}(s, p, e) \oplus \text{Attribute}(s, p, l) \\
\text{Relation}(s, p, o) &\models \text{Entity}(s) \wedge \text{Property}(p) \wedge \text{Entity}(o) \\
\text{Attribute}(s, p, x) &\models \text{Entity}(s) \wedge \text{Property}(p) \wedge \text{Value}(x) \\
\text{Value}(x) &\equiv x \in \mathbb{R} \oplus x \in \mathbb{S}
\end{aligned}$$

An `ROCrate(R)` is defined as a self-described *Root Data Entity*, which describes and contains parts (*data entities*), which are further described in *contextual entities*. These terms align with their use in the [RO-Crate 1.1 terminology](#).

The `Root(r)` is a type of `Dataset(r)`, and must have the metadata to literal attributes to provide a `name`, `description` and `datePublished`, as well as a contextual entity identifying its license. These predicates correspond to the RO-Crate 1.1 [requirements for the root data entity](#).

The concept of an `Entity(e)` is introduced as being either a `DataEntity(e)`, a `ContextualEntity(e)`, or [both](#); and must be typed with at least one `Class(e)`.

For simplicity in this formalization (and to assist production rules below) `R` is a constant representing a single RO-Crate, typically written to independent RO-Crate Metadata files. `R` is used by `Mentions(R, e)` to indicate that `e` is an Entity described by the RO-Crate and therefore its metadata (a set of Relation and Attribute predicates) form part of the RO-Crate serialization. `Relation(s, p, o)` and `Attribute(s, p, x)` are defined as a *subject-predicate-object* triple pattern from an `Entity(s)` using a `Property(p)` to either another `Entity(o)` or a `Literal(x)` value.

Example of formalized RO-Crate

The below is an example RO-Crate represented using the above formalization, assuming a base URI of `http://example.com/ro/123/`:

```

RO-Crate(<http://example.com/ro/123/>)
name(<http://example.com/ro/123/,
    "Data files associated with the manuscript:Effects of ...")
description(<http://example.com/ro/123/,
    "Palliative care planning for nursing home residents ...")
license(<http://example.com/ro/123/>,
    <https://spdx.org/licenses/CC-BY-4.0>
datePublished(<http://example.com/ro/123/>, "2017")
hasPart(<http://example.com/ro/123/>,
    <http://example.com/ro/123/survey.csv>)
hasPart(<http://example.com/ro/123/>,
    <http://example.com/ro/123/interviews/>)

ContextualEntity(<https://spdx.org/licenses/CC-BY-4.0>)
name(<https://spdx.org/licenses/CC-BY-4.0,
    "Creative Commons Attribution 4.0")

ContextualEntity(<https://spdx.org/licenses/CC-BY-NC-4.0>)
name(<https://spdx.org/licenses/CC-BY-NC-4.0,
    "Creative Commons Attribution Non Commercial 4.0")

File(<http://example.com/ro/123/survey.csv>)
name(<http://example.com/ro/123/survey.csv>, "Survey of care providers")

Dataset(<http://example.com/ro/123/interviews/>)
name(<http://example.com/ro/123/interviews/>,
    "Audio recordings of care provider interviews")
license(<http://example.com/ro/123/interviews/>,
    <https://spdx.org/licenses/CC-BY-NC-4.0>)

```

Notable from this triple-like formalization is that a RO-Crate R is fully represented as a tree at depth 2 helped by the use of `iri` nodes. For instance the aggregation from the root entity `hasPart(...interviews/>)` is at same level as the data entity's property `license(...CC-BY-NC-4.0>)` and that contextual entity's attribute name `(...Non Commercial 4.0")`. As shown in section RO-Crate JSON-LD, the RO-Crate Metadata File serialization is an equivalent shallow tree, although at depth 3 to cater for the JSON-LD preamble of `"@context"` and `"@graph"`.

In reality many additional attributes and contextual types from schema.org types like <http://schema.org/affiliation> and <http://schema.org/Organization> would be used to further describe the RO-Crate and its entities, but as these are optional (*SHOULD* requirements) they do not form part of this formalization.

Mapping to RDF with schema.org

A formalized RO-Crate can be mapped to different serializations. Assume a simplified⁹ language `LRDF`

$$\begin{aligned}
\mathbb{L}r\mathbb{D}f &= \{ \text{Triple}(s,p,o), \text{IRI}(i), \text{BlankNode}(b), \text{Literal}(s), \\
&\quad \mathbb{I}ri, \mathbb{R}, \mathbb{S} \} \\
\mathbb{D}r\mathbb{D}f &= \mathbb{S} \\
\forall i \text{ IRI}(i) &\Rightarrow i \in \mathbb{I}ri \\
\forall s \forall p \forall o \text{ Triple}(s,p,o) &\Rightarrow (\text{IRI}(s) \vee \text{BlankNode}(s)) \wedge \\
&\quad \text{IRI}(p) \wedge \\
&\quad (\text{IRI}(o) \vee \text{BlankNode}(o) \vee \text{Literal}(o)) \\
\text{Literal}(v) &\models \text{Value}(v) \wedge \text{Datatype}(v,t) \wedge \text{IRI}(t) \\
\forall v \text{ Value}(v) &\Rightarrow v \in \mathbb{S} \\
\text{LanguageTag}(v,l) &\equiv \text{Datatype}(v, \\
&\quad \text{http://www.w3.org/1999/02/22-rdf-syntax-} \\
&\quad \text{ns\#langString})
\end{aligned}$$

Below follows a mapping from `Ⓐrocrate` to `Ⓐrdf` using schema.org.

$$\begin{aligned}
\text{Property}(p) &\Rightarrow \text{type}(p, \langle \text{http://www.w3.org/2000/01/rdf-} \\
&\quad \text{schema\#Property} \rangle) \\
\text{Class}(c) &\Rightarrow \text{type}(c, \langle \text{http://www.w3.org/2000/01/rdf-schema\#Class} \rangle) \\
\text{Dataset}(d) &\Rightarrow \text{type}(d, \langle \text{http://schema.org/Dataset} \rangle) \\
\text{File}(f) &\Rightarrow \text{type}(f, \langle \text{http://schema.org/MediaObject} \rangle) \\
\text{CreativeWork}(e) &\Rightarrow \text{ContextualEntity}(e) \wedge \\
&\quad \text{type}(e, \langle \text{http://schema.org/CreativeWork} \rangle) \\
\text{hasPart}(e, t) &\Rightarrow \text{Relation}(e, \langle \text{http://schema.org/hasPart} \rangle, t) \\
\text{name}(e, n) &\Rightarrow \text{Attribute}(e, \langle \text{http://schema.org/name} \rangle, n) \\
\text{description}(e, s) &\Rightarrow \text{Attribute}(e, \langle \text{http://schema.org/description} \rangle, s) \\
\text{datePublished}(e, d) &\Rightarrow \text{Attribute}(e, \langle \text{http://schema.org/datePublished} \rangle, d) \\
\text{license}(e, l) &\Rightarrow \text{Relation}(e, \langle \text{http://schema.org/license} \rangle, l) \wedge \\
&\quad \text{CreativeWork}(l) \\
\text{type}(e, t) &\Rightarrow \text{Relation}(e, \\
&\quad \langle \text{http://www.w3.org/1999/02/22-rdf-syntax-ns\#type} \rangle, \\
t) &\wedge \\
&\quad \text{Class}(t) \\
\text{String}(s) &\equiv \text{Value}(s) \wedge s \in \mathbb{S} \\
\text{String}(s) &\Rightarrow \text{Datatype}(s, \\
&\quad \langle \text{http://www.w3.org/2001/XMLSchema\#string} \rangle) \\
\text{Decimal}(d) &\equiv \text{Value}(d) \wedge d \in \mathbb{R} \\
\text{Decimal}(d) &\Rightarrow \text{Datatype}(d, \\
&\quad \langle \text{http://www.w3.org/2001/XMLSchema\#decimal} \rangle) \\
\text{Relation}(s,p,o) &\Rightarrow \text{Triple}(s,p,o) \wedge \text{IRI}(s) \wedge \text{IRI}(o) \\
\text{Attribute}(s,p,o) &\Rightarrow \text{Triple}(s,p,o) \wedge \text{IRI}(s) \wedge \text{Literal}(o)
\end{aligned}$$

Note that in the JSON-LD serialization of RO-Crate the expression of `Class` and `Property` is typically indirect: The JSON-LD `@context` maps to schema.org IRIs, which, when resolved as Linked Data, embeds their formal definition as RDFa. Extensions may however include such term definitions directly in the RO-Crate.

RO-Crate 1.1 Metadata File Descriptor

An important RO-Crate principle is that of being **self-described**. Therefore the serialization of the RO-Crate into a file should also describe itself in a [Metadata File Descriptor](#), indicating it is **about** (describing) the RO-Crate root data entity, and that it **conformsTo** a particular version of the RO-Crate specification:

```

about(s,o) ⇒ Relation(s, <http://schema.org/about>, o)
conformsTo(s,o) ⇒ Relation(s,
    <http://purl.org/dc/terms/conformsTo>, R)
MetadataFileDescriptor(m) ⇒ ( CreativeWork(m) ∧ about(m,R) ∧ ROCrate(R) ∧
    conformsTo(m,
        <https://w3id.org/ro/crate/1.1>) )

```

Note that although the metadata file necessarily is an *information resource* written to disk or served over the network (as JSON-LD), it is not considered to be a contained *part* of the RO-Crate in the form of a *data entity*, rather it is described only as a *contextual entity*.

In the conceptual model the *RO-Crate Metadata File* can be seen as the top-level node that describes the *RO-Crate Root*, however in the formal model (and the JSON-LD format) the metadata file descriptor is an additional contextual entity that is not affecting the depth-limit of the RO-Crate.

Forward-chained Production Rules for JSON-LD

Combining the above predicates and schema.org mapping with rudimentary JSON templates, these forward-chaining production rules can output JSON-LD according to the RO-Crate 1.1 specification^{[10](#)}:

```

Mentions(R, s) ∧ Relation(s, p, o) ⇒ Mentions(R, o)
    IRI(i) ⇒ "i"
    Decimal(d) ⇒ d
    String(s) ⇒ "s"
    ∀e∀t type(e,t) ⇒ { "@id": s,
        "@type": t }
    ∀s∀p∀o Relation(s,p,o) ⇒ { "@id": s,
        p: { "@id": o }
    }
    ∀s∀p∀v Attribute(s,p,v) ⇒ { "@id": s,
        p: v
    }
    ∀r∀c ROCrate(R) ⇒ { "@graph": [
        Mentions(r, c)*
    ]
    }
    R ⊨ <./>
    R ⇒ MetadataFileDescriptor(
        <ro-crate-metadata.json>)

```

This exposes the first order logic domain of discourse of IRIs, with rational numbers and strings as their corresponding JSON-LD representation. These production rules first grow the graph of **R** by adding a transitive rule that anything described in **R** which is related to **o** means that **o** is also considered mentioned by the RO-Crate **R**. For simplicity this rule is one-way; in theory the JSON-LD graph can also contain free-standing

contextual entities that have outgoing relations to data- and contextual entities, but these are proposed to be bound to the root data entity with schema.org relation <http://schema.org/mentions>.

RO-Crate Community

As of 2021-05-24, the *RO-Crate* Community members are:

- Peter Sefton (co-chair)
- Stian Soiland-Reyes (co-chair)
- Eoghan Ó Carragáin (emeritus chair)
- Oscar Corcho
- Daniel Garijo
- Raul Palma
- Frederik Coppens
- Carole Goble
- José María Fernández
- Kyle Chard
- Jose Manuel Gomez-Perez
- Michael R Crusoe
- Ignacio Eguinoa
- Nick Juty
- Kristi Holmes
- Jason A. Clark
- Salvador Capella-Gutierrez
- Alasdair J. G. Gray
- Stuart Owen
- Alan R Williams
- Giacomo Tartari
- Finn Bacall
- Thomas Thelen
- Hervé Ménager
- Laura Rodríguez-Navas
- Paul Walk
- brandon whitehead
- Mark Wilkinson
- Paul Groth
- Erich Bremer
- LJ Garcia Castro
- Karl Sebby
- Alexander Kanitz
- Ana Trisovic
- Gavin Kennedy
- Mark Graves
- Jasper Koehorst
- Simone Leo
- Marc Portier
- Paul Brack
- Milan Ojsteršek
- Bert Driesbeke
- Chenxu Niu
- Kosuke Tanabe
- Tomasz Miksa

- Marco La Rosa

References

1. FAIR Data Management; It's a lifestyle not a lifecycle - ptsefton.com

Peter Sefton

ptsefton.com (2021-04-07) <http://ptsefton.com/2021/04/07/rdmpic/>

2. Enhancing reproducibility for computational methods

Victoria Stodden, Marcia McNutt, David H Bailey, Ewa Deelman, Yolanda Gil, Brooks Hanson, Michael A Heroux, John PA Ioannidis, Michela Taufer

Science (2016-12-09)

DOI: [10.1126/science.aah6168](https://doi.org/10.1126/science.aah6168) · PMID: [27940837](https://pubmed.ncbi.nlm.nih.gov/27940837/)

3. Towards FAIR principles for research software

Anna-Lena Lamprecht, Leyla Garcia, Mateusz Kuzak, Carlos Martinez, Ricardo Arcila, Eva Martin Del Pico, Victoria Dominguez Del Angel, Stephanie van de Sandt, Jon Ison, Paula Andrea Martinez, ... Salvador Capella-Gutierrez

Déviance et société (2019-11-13)

DOI: [10.3233/ds-190026](https://doi.org/10.3233/ds-190026)

4. FAIR Computational Workflows

Carole Goble, Sarah Cohen-Boulakia, Stian Soiland-Reyes, Daniel Garijo, Yolanda Gil, Michael R. Crusoe, Kristian Peters, Daniel Schober

Data Intelligence (2019-11-01)

DOI: [10.1162/dint_a_00033](https://doi.org/10.1162/dint_a_00033)

5. The FAIR Guiding Principles for scientific data management and stewardship.

Mark D Wilkinson, Michel Dumontier, IJsbrand Jan Aalbersberg, Gabrielle Appleton, Myles Axton, Arie Baak, Niklas Blomberg, Jan-Willem Boiten, Luiz Bonino da Silva Santos, Philip E Bourne, ... Barend Mons

Scientific data (2016-03-15)

DOI: [10.1038/sdata.2016.18](https://doi.org/10.1038/sdata.2016.18) · PMID: [26978244](https://pubmed.ncbi.nlm.nih.gov/26978244/) · PMCID: [PMC4792175](https://pubmed.ncbi.nlm.nih.gov/PMC4792175/)

6. Data Stewardship for Open Science

Barend Mons

Taylor & Francis

ISBN: [9781315351148](https://doi.org/9781315351148)

7. Zenodo, an Archive and Publishing Repository: A tale of two herbarium specimen pilot projects

Mathias Dillen, Quentin Groom, Donat Agosti, Lars Nielsen

Biodiversity Information Science and Standards (2019-06-18)

DOI: [10.3897/biss.3.37080](https://doi.org/10.3897/biss.3.37080)

8. The worldwide Protein Data Bank (wwPDB): ensuring a single, uniform archive of PDB data.

Helen Berman, Kim Henrick, Haruki Nakamura, John L Markley

Nucleic Acids Research (2007-01)

DOI: [10.1093/nar/gkl971](https://doi.org/10.1093/nar/gkl971) · PMID: [17142228](https://pubmed.ncbi.nlm.nih.gov/17142228/) · PMCID: [PMC1669775](https://pubmed.ncbi.nlm.nih.gov/PMC1669775/)

9. Ten simple rules for reproducible computational research.

Geir Kjetil Sandve, Anton Nekrutenko, James Taylor, Eivind Hovig

PLoS Computational Biology (2013-10-24)

DOI: [10.1371/journal.pcbi.1003285](https://doi.org/10.1371/journal.pcbi.1003285) · PMID: [24204232](https://pubmed.ncbi.nlm.nih.gov/24204232/) · PMCID: [PMC3812051](https://pubmed.ncbi.nlm.nih.gov/PMC3812051/)

10. Talking datasets – Understanding data sensemaking behaviours

Laura Koesten, Kathleen Gregory, Paul Groth, Elena Simperl

International Journal of Human-Computer Studies (2021-02) <https://doi.org/gmghzh>

DOI: [10.1016/j.ijhcs.2020.102562](https://doi.org/10.1016/j.ijhcs.2020.102562)

11. Why linked data is not enough for scientists

Sean Bechhofer, Iain Buchan, David De Roure, Paolo Missier, John Ainsworth, Jiten Bhagat, Philip Couch, Don Cruickshank, Mark Delderfield, Ian Dunlop, ... Carole Goble

Future Generation Computer Systems (2013-02)

DOI: [10.1016/j.future.2011.08.004](https://doi.org/10.1016/j.future.2011.08.004)

12. Using a suite of ontologies for preserving workflow-centric research objects

Khalid Belhajjame, Jun Zhao, Daniel Garijo, Matthew Gamble, Kristina Hettne, Raul Palma, Eleni Mina, Oscar Corcho, José Manuel Gómez-Pérez, Sean Bechhofer, ... Carole Goble

Web Semantics: Science, Services and Agents on the World Wide Web (2015-05)

DOI: [10.1016/j.websem.2015.01.003](https://doi.org/10.1016/j.websem.2015.01.003)

13. A lightweight approach to research object data packaging

Eoghan Ó Carragáin, Carole Goble, Peter Sefton, Stian Soiland-Reyes

Zenodo (2019)

DOI: [10.5281/zenodo.3250687](https://doi.org/10.5281/zenodo.3250687)

14. As a researcher...I'm a bit bloody fed up with Data Management

Cameron Neylon

Science In The Open (2017-07-16) <https://cameronneyleon.net/blog/as-a-researcher-im-a-bit-bloody-fed-up-with-data-management/>

15. Linked Data: Evolving the Web into a Global Data Space

Tom Heath, Christian Bizer

Synthesis Lectures on the Semantic Web: Theory and Technology (2011-02-09)

<http://www.morganclaypool.com/doi/abs/10.2200/{S00334ED1V01Y201102WBE001}>

DOI: [10.2200/s00334ed1v01y201102wbe001](https://doi.org/10.2200/s00334ed1v01y201102wbe001) · ISBN: [9781608454310](https://www.isbn-international.org/product/9781608454310)

16. Identifiers for the 21st century: How to design, provision, and reuse persistent identifiers to maximize utility and impact of life science data.

Julie A McMurtry, Nick Juty, Niklas Blomberg, Tony Burdett, Tom Conlin, Nathalie Conte, Mélanie Courtot, John Deck, Michel Dumontier, Donal K Fellows, ... Helen Parkinson

PLoS Biology (2017-06-29)

DOI: [10.1371/journal.pbio.2001414](https://doi.org/10.1371/journal.pbio.2001414) · PMID: [28662064](https://pubmed.ncbi.nlm.nih.gov/28662064/) · PMCID: [PMC5490878](https://pubmed.ncbi.nlm.nih.gov/PMC5490878/)

17. The BagIt File Packaging Format (V1.0)

J. Kunze, J. Littman, E. Madden, J. Scancella, C. Adams

RFC Editor (2018-10) <https://www.rfc-editor.org/info/rfc8493>

DOI: [10.17487/{rfc8493}](https://doi.org/10.17487/{rfc8493})

18. Oxford Common File Layout Specification

OCFL

OCFL (2020-07-07) <https://ocfl.io/1.0/spec/>

19. Linked data: the story so far

Christian Bizer, Tom Heath, Tim Berners-Lee

Semantic services, interoperability and web applications: emerging concepts (2011) <http://services.igi->

global.com/resolvedoi/resolve.aspx?doi=10.4018/978-1-60960-593-3.ch008

DOI: [10.4018/978-1-60960-593-3.ch008](https://doi.org/10.4018/978-1-60960-593-3.ch008) · ISBN: [9781609605933](https://www.isbn-international.org/product/9781609605933)

20. Internationalized resource identifiers (IRIs)

M. Duerst, M. Suignard

RFC Editor (2005-01) <https://www.rfc-editor.org/info/rfc3987>

DOI: [10.17487/rfc3987](https://doi.org/10.17487/rfc3987)

21. Dereferencing HTTP URIs

W3C Technical Architecture Group

W3C (2007-08-31) <https://www.w3.org/2001/tag/doc/{httpRange}-14/2007-08-31/{HttpRange}-14.html>

22. RO-Crate Metadata Specification 1.1.1

Peter Sefton, Eoghan Ó Carragáin, Stian Soiland-Reyes, Oscar Corcho, Daniel Garijo, Raul Palma, Frederik Coppens, Carole Goble, José María Fernández, Kyle Chard, ... Marc Portier

Zenodo (2021) <https://w3id.org/ro/crate/1.1>

DOI: [10.5281/zenodo.4541002](https://doi.org/10.5281/zenodo.4541002)

23. Schema.org: Evolution of Structured Data on the Web: Big data makes common schemas even more necessary

Ramanathan V Guha, Dan Brickley, Steve Macbeth

Queue (2015)

DOI: [10.1145/2857274.2857276](https://doi.org/10.1145/2857274.2857276)

24. JSON-LD 1.0

Manu Sporny, Dave Longley, Gregg Kellogg, Markus Lanthaler, Niklas Lindström

W3C (2014-01-16) <https://www.w3.org/TR/2014/REC-json-ld-20140116/>

25. Sharing interoperable workflow provenance: A review of best practices and their practical application in CWLProv

Farah Zaib Khan, Stian Soiland-Reyes, Richard O Sinnott, Andrew Lonie, Carole Goble, Michael R Crusoe
GigaScience (2019-11-01)

DOI: [10.1093/gigascience/giz095](https://doi.org/10.1093/gigascience/giz095) · PMID: [31675414](https://pubmed.ncbi.nlm.nih.gov/31675414/) · PMCID: [PMC6824458](https://pubmed.ncbi.nlm.nih.gov/PMC6824458/)

26. Portland Common Data Model

Stefano Cossu, Esmé Cowles, Karen Estlund, Christina Harlow, Tom Johnson, Mark Matienzo, Danny Lamb, Lynette Rayle, Rob Sanderson, Jon Stroop, Andrew Woods

GitHub duraspace/pcdm Wiki (2018-06-15) <https://github.com/duraspace/pcdm/wiki>

27. Bioschemas: From Potato Salad to Protein Annotation

Alasdair Gray, Carole Goble, Rafael Jimenez, Bioschemas Community

(2017-10-23) <https://iswc2017.semanticweb.org/paper-579/>

28. npm: ro-crate-html-js <https://www.npmjs.com/package/ro-crate-html-js>

29. Datacrate Submission To The Workshop On Research Objects

Peter Sefton

Zenodo (2018)

DOI: [10.5281/zenodo.1445817](https://doi.org/10.5281/zenodo.1445817)

30. ELIXIR: a distributed infrastructure for European biological data

Lindsey C Crosswell, Janet M Thornton

Trends in Biotechnology (2012-05) <http://dx.doi.org/10.1016/j.tibtech.2012.02.002>
DOI: [10.1016/j.tibtech.2012.02.002](https://doi.org/10.1016/j.tibtech.2012.02.002) · PMID: [22417641](https://pubmed.ncbi.nlm.nih.gov/22417641/)

31. **Digital crowdsourcing and public understandings of the past: citizen historians meet Criminal Characters**
Alana Piper
History Australia (2020-07-02) <https://www.tandfonline.com/doi/full/10.1080/14490854.2020.1796500>
DOI: [10.1080/14490854.2020.1796500](https://doi.org/10.1080/14490854.2020.1796500)
32. **RO-Crate Metadata Specification 1.0**
Peter Sefton, Eoghan Ó Carragáin, Stian Soiland-Reyes, Oscar Corcho, Daniel Garijo, Raul Palma, Frederik Coppens, Carole Goble, José María Fernández, Kyle Chard, ... Thomas Thelen
Zenodo (2019) <https://w3id.org/ro/crate/1.0>
DOI: [10.5281/zenodo.3541888](https://doi.org/10.5281/zenodo.3541888)
33. **RO-Crate Metadata Specification 1.1**
Peter Sefton, Eoghan Ó Carragáin, Stian Soiland-Reyes, Oscar Corcho, Daniel Garijo, Raul Palma, Frederik Coppens, Carole Goble, José María Fernández, Kyle Chard, ... Simone Leo
Zenodo (2020) <https://w3id.org/ro/crate/1.1>
DOI: [10.5281/zenodo.4031327](https://doi.org/10.5281/zenodo.4031327)
34. **Arkisto Platform: Describo**
Marco La Rosa, Peter Sefton
<https://arkisto-platform.github.io/describo/>
35. **Arkisto Platform: Describo Online**
Marco La Rosa
<https://arkisto-platform.github.io/describo-online/>
36. **npm: ro-crate-excel**
Mike Lynch, Peter Sefton
<https://www.npmjs.com/package/ro-crate-excel>
37. **GitHub - UTS-eResearch/ro-crate-js: Research Object Crate (RO-Crate) utilities**
GitHub
<https://github.com/{UTS}-{eResearch}/ro-crate-js>
38. **GitHub - ResearchObject/ro-crate-ruby: A Ruby gem for creating, manipulating and reading RO-Crates**
Finn Bacall, Martyn Whitwell
GitHub <https://github.com/{ResearchObject}/ro-crate-ruby>
39. **GitHub - ResearchObject/ro-crate-py: Python library for RO-Crate**
Simone Leo, Ignacio Eguinoa, Stian Soiland-Reyes, Bert Driesbeke, Laura Rodríguez-Navas, Alban Gaignard
<https://github.com/researchobject/ro-crate-py>
40. **WorkflowHub project Project pages for developing and running the WorkflowHub, a registry of scientific workflows** <https://about.workflowhub.eu/>
41. **LifeMonitor, a testing and monitoring service for scientific workflows**
CRS4
<https://about.lifemonitor.eu/>

42. **GitHub - athenarc/schema: SCHeMa (Scheduler for scientific Containers on clusters of Heterogeneous Machines)**
Kostis Zagganas, Alvaro Gonzalez, Aggelos Kolaitis, Tewodros Deneke
(2021) <https://github.com/athenarc/schema>
43. **Use of RO-Crates in SCHeMa**
Thanasis Vergoulis
Zenodo (2021-04-08)
DOI: [10.5281/zenodo.4671709](https://doi.org/10.5281/zenodo.4671709)
44. **GitHub - workflowhub-eu/galaxy2cwl: Standalone version tool to get cwl descriptions (initially an abstract cwl interface) of galaxy workflows and Galaxy workflows executions**
<https://github.com/workflowhub-eu/galaxy2cwl>
45. **GitHub - CoEDL/modpdsc** <https://github.com/{CoEDL}/modpdsc/>
46. **Tools: Data Portal & Discovery** <https://arkisto-platform.github.io/tools/portal/>
47. **GitHub - CoEDL/ocfl-tools: Tools to process and manipulate an OCFL tree**
<https://github.com/{CoEDL}/ocfl-tools>
48. **eScienceLab: RO-Composer**
Finn Bacall, Stian Soiland-Reyes, Marina Soares e Silva
<https://esciencelab.org.uk/projects/ro-composer/>
49. **RO-Crate RDA maDMP Mapper**
Ghaith Arfaoui, Maroua Jaoua
Zenodo (2020)
DOI: [10.5281/zenodo.3922136](https://doi.org/10.5281/zenodo.3922136)
50. **Research Object Crates and Machine-actionable Data Management Plans**
Tomasz Miksa, Maroua Jaoua, Ghaith Arfaoui
PUBLISSO (2020)
DOI: [10.4126/fri01-006423291](https://doi.org/10.4126/fri01-006423291)
51.
Gabriel Brenner
Zenodo (2020)
DOI: [10.5281/zenodo.3903463](https://doi.org/10.5281/zenodo.3903463)
52. **KockataEPich/CheckMyCrate: A command line application for validating a RO-Crate object against a JSON profile.**
Kostadin Belchev
GitHub (2021-05) <https://github.com/KockataEPich/CheckMyCrate>
53. **No more business as usual: Agile and effective responses to emerging pathogen threats require open data and open analytics.**
Dannon Baker, Marius van den Beek, Daniel Blankenberg, Dave Bouvier, John Chilton, Nate Coraor, Frederik Coppens, Ignacio Eguinoa, Simon Gladman, Björn Grüning, ... Steven Weaver
PLoS Pathogens (2020-08-13)
DOI: [10.1371/journal.ppat.1008643](https://doi.org/10.1371/journal.ppat.1008643) · PMID: [32790776](https://pubmed.ncbi.nlm.nih.gov/32790776/) · PMCID: [PMC7425854](https://pubmed.ncbi.nlm.nih.gov/PMC7425854/)

54. The nf-core framework for community-curated bioinformatics pipelines.

Philip A Ewels, Alexander Peltzer, Sven Fillinger, Harshil Patel, Johannes Alneberg, Andreas Wilm, Maxime Ulysse Garcia, Paolo Di Tommaso, Sven Nahnsen
Nature Biotechnology (2020)
DOI: [10.1038/s41587-020-0439-x](https://doi.org/10.1038/s41587-020-0439-x) · PMID: [32055031](https://pubmed.ncbi.nlm.nih.gov/32055031/)

55. Nextflow enables reproducible computational workflows.

Paolo Di Tommaso, Maria Chatzou, Evan W Floden, Pablo Prieto Barja, Emilio Palumbo, Cedric Notredame
Nature Biotechnology (2017-04-11)
DOI: [10.1038/nbt.3820](https://doi.org/10.1038/nbt.3820) · PMID: [28398311](https://pubmed.ncbi.nlm.nih.gov/28398311/)

56. Snakemake—a scalable bioinformatics workflow engine.

Johannes Köster, Sven Rahmann
Bioinformatics (2012-10-01)
DOI: [10.1093/bioinformatics/bts480](https://doi.org/10.1093/bioinformatics/bts480) · PMID: [22908215](https://pubmed.ncbi.nlm.nih.gov/22908215/)

57. EOSC-Life Methodology framework to enhance reproducibility within EOSC-Life

Florence Bietrix, José Maria Carazo, Salvador Capella-Gutierrez, Frederik Coppens, Maria Luisa Chiusano, Romain David, Jose Maria Fernandez, Maddalena Fratelli, Jean-Karim Heriche, Carole Goble, ... Jing Tang
Zenodo (2021-04-30)
DOI: [10.5281/zenodo.4705078](https://doi.org/10.5281/zenodo.4705078)

58. Methods Included: Standardizing Computational Reuse and Portability with the Common Workflow Language

Michael R. Crusoe, Sanne Abeln, Alexandru Iosup, Peter Amstutz, John Chilton, Nebojša Tijanić, Hervé Ménager, Stian Soiland-Reyes, Carole Goble
arXiv (2021) [http://arxiv.org/abs/2105.07028v1](https://arxiv.org/abs/2105.07028v1)

59. Implementing FAIR Digital Objects in the EOSC-Life Workflow Collaboratory

Carole Goble, Stian Soiland-Reyes, Finn Bacall, Stuart Owen, Alan Williams, Ignacio Eguinoa, Bert Driesbeke, Simone Leo, Luca Pireddu, Laura Rodríguez-Navas, ... Frederik Coppens
Zenodo (2021)
DOI: [10.5281/zenodo.4605654](https://doi.org/10.5281/zenodo.4605654)

60. Tracking Workflow Execution With TavernaPROV

Stian Soiland-Reyes, Pinar Alper, Carole Goble
(2016-06-06)
DOI: [10.5281/zenodo.51314](https://doi.org/10.5281/zenodo.51314)

61. Research Object Bundle 1.0

Stian Soiland-Reyes, Matthew Gamble, Robert Haines
(2014-11) <https://w3id.org/bundle/2014-11-05/>
DOI: [10.5281/zenodo.12586](https://doi.org/10.5281/zenodo.12586)

62. Protein Ligand Complex MD Setup tutorial using BioExcel Building Blocks (biobb) (jupyter notebook)

Douglas Lowe, Genís Bayarri
WorkflowHub (2021)
DOI: [10.48546/workflowhub.workflow.56.1](https://doi.org/10.48546/workflowhub.workflow.56.1)

63. Enabling precision medicine via standard communication of HTS provenance, analysis, and results.

Gil Alterovitz, Dennis Dean, Carole Goble, Michael R Crusoe, Stian Soiland-Reyes, Amanda Bell, Anais

Hayes, Anita Suresh, Anjan Purkayastha, Charles H King, ... Raja Mazumder

PLoS Biology (2018-12-31)

DOI: [10.1371/journal.pbio.3000099](https://doi.org/10.1371/journal.pbio.3000099) · PMID: [30596645](https://pubmed.ncbi.nlm.nih.gov/30596645/) · PMCID: [PMC6338479](https://pubmed.ncbi.nlm.nih.gov/PMC6338479/)

64. IEEE Standard for Bioinformatics Analyses Generated by High-Throughput Sequencing (HTS) to Facilitate Communication

DOI: [10.1109/ieeestd.2020.9094416](https://doi.org/10.1109/ieeestd.2020.9094416) · ISBN: [978-1-5044-6466-6](https://www.isbn-international.org/product/978-1-5044-6466-6)

65. Describing and packaging workflows using RO-Crate and BioCompute Objects

Stian Soiland-Reyes

Zenodo (2021-04)

DOI: [10.5281/zenodo.4633732](https://doi.org/10.5281/zenodo.4633732)

66. I'll take that to go: Big data bags and minimal identifiers for exchange of large, complex datasets

Kyle Chard, Mike D'Arcy, Ben Heavner, Ian Foster, Carl Kesselman, Ravi Madduri, Alexis Rodriguez, Stian Soiland-Reyes, Carole Goble, Kristi Clark, ... Arthur Toga

2016 IEEE International Conference on Big Data (Big Data) (2016-12-05)

<https://static.aminer.org/pdf/fa/bigdata2016/BigD418.pdf>

DOI: [10.1109/bigdata.2016.7840618](https://doi.org/10.1109/bigdata.2016.7840618) · ISBN: [978-1-4673-9005-7](https://www.isbn-international.org/product/978-1-4673-9005-7)

67. Ten principles for machine-actionable data management plans.

Tomasz Miksa, Stephanie Simms, Daniel Mietchen, Sarah Jones

PLoS Computational Biology (2019-03-28)

DOI: [10.1371/journal.pcbi.1006750](https://doi.org/10.1371/journal.pcbi.1006750) · PMID: [30921316](https://pubmed.ncbi.nlm.nih.gov/30921316/) · PMCID: [PMC6438441](https://pubmed.ncbi.nlm.nih.gov/PMC6438441/)

68. Machine-Actionable Data Management Plans: A Knowledge Retrieval Approach to Automate the Assessment of Funders' Requirements

João Cardoso, Diogo Proença, José Borbinha

Advances in Information Retrieval (2020)

DOI: [10.1007/978-3-030-45442-5_15](https://doi.org/10.1007/978-3-030-45442-5_15) · ISBN: [978-3-030-45442-5](https://www.isbn-international.org/product/978-3-030-45442-5)

69. RDA DMP Common Standard for Machine-actionable Data Management Plans

Paul Walk, Tomasz Miksa, Peter Neish

Research Data Alliance (2019)

DOI: [10.15497/rda00039](https://doi.org/10.15497/rda00039)

70. Towards semantic representation of machine-actionable Data Management Plans

João Cardoso, Leyla Jael Garcia Castro, Fajar Ekaputra, Marie-Christine Jacquemot-Perbal, Tomasz Miksa, José Borbinha

PUBLISSO (2020) <https://repository.publisso.de/resource/fri:6423289>

DOI: [10.4126/fri01-006423289](https://doi.org/10.4126/fri01-006423289)

71. Data Catalog Vocabulary (DCAT) - Version 2

Riccardo Albertoni, David Browning, Simon Cox, Alejandra Gonzalez Beltran, Andrea Perego, Peter Winstanley, Dataset Exchange Working Group

W3C (2020-02-04) <https://www.w3.org/TR/2020/REC-vocab-dcat-2-20200204/>

72. A case for data commons: toward data science as a service

Robert L Grossman, Allison Heath, Mark Murphy, Maria Patterson, Walt Wells

Computing in science & engineering (2016)

DOI: [10.1109/mcse.2016.92](https://doi.org/10.1109/mcse.2016.92)

73. The Australian Research Data Commons

Michelle Barker, Ross Wilkinson, Andrew Treloar

Data Science Journal (2019-09-05) <http://datascience.codata.org/articles/10.5334/dsj-2019-044/>

DOI: [10.5334/dsj-2019-044](https://doi.org/10.5334/dsj-2019-044)

74. The NCI Genomic Data Commons as an engine for precision medicine.

Mark A Jensen, Vincent Ferretti, Robert L Grossman, Louis M Staudt

Blood (2017-07-27)

DOI: [10.1182/blood-2017-03-735654](https://doi.org/10.1182/blood-2017-03-735654) · PMID: [28600341](https://pubmed.ncbi.nlm.nih.gov/28600341/) · PMCID: [PMC5533202](https://pubmed.ncbi.nlm.nih.gov/PMC5533202/)

75. Harvard Data Commons

Mercè Crosas

Septentrio Conference Series (2020-03-20)

DOI: [10.7557/5.5422](https://doi.org/10.7557/5.5422)

76. The Dataverse Network: An Open-Source Application for Sharing, Discovering and Preserving Data

Mercè Crosas

D-Lib Magazine (2011-01)

DOI: [10.1045/january2011-crosas](https://doi.org/10.1045/january2011-crosas)

77. Efficient and Secure Transfer, Synchronization, and Sharing of Big Data

Kyle Chard, Steven Tuecke, Ian Foster

IEEE Cloud Computing (2014)

DOI: [10.1109/mcc.2014.52](https://doi.org/10.1109/mcc.2014.52)

78. Dataset reuse: toward translating principles to practice

Laura Koesten, Pavlos Vougiouklis, Elena Simperl, Paul Groth

Patterns (New York, N.Y.) (2020-11-13)

DOI: [10.1016/j.patter.2020.100136](https://doi.org/10.1016/j.patter.2020.100136) · PMID: [33294873](https://pubmed.ncbi.nlm.nih.gov/33294873/) · PMCID: [PMC7691392](https://pubmed.ncbi.nlm.nih.gov/PMC7691392/)

79. Electronic documents give reproducible research a new meaning

Jon F. Claerbout, Martin Karrenbach

SEG Technical Program Expanded Abstracts 1992 (1992-01)

DOI: [10.1190/1.1822162](https://doi.org/10.1190/1.1822162)

80. Interoperability for the Discovery, Use, and Re-Use of Units of Scholarly Communication

Herbert Van de Sompel, Carl Lagoze

CTWatch Quarterly (2007-08) <http://icl.utk.edu/ctwatch/quarterly/articles/2007/08/interoperability-for-the-discovery-use-and-re-use-of-units-of-scholarly-communication/>

81. myExperiment: a repository and social network for the sharing of bioinformatics workflows.

Carole A Goble, Jiten Bhagat, Sergejs Aleksejevs, Don Cruickshank, Danus Michaelides, David Newman, Mark Borkum, Sean Bechhofer, Marco Roos, Peter Li, David De Roure

Nucleic Acids Research (2010-07-01)

DOI: [10.1093/nar/gkq429](https://doi.org/10.1093/nar/gkq429) · PMID: [20501605](https://pubmed.ncbi.nlm.nih.gov/20501605/) · PMCID: [PMC2896080](https://pubmed.ncbi.nlm.nih.gov/PMC2896080/)

82. myExperiment: An ontology for e-Research

David Newman, Sean Bechhofer, David De Roure

Proceedings of the Workshop on Semantic Web Applications in Scientific Discourse (SWASD 2009) (2009-10)

<http://ceur-ws.org/Vol-523/Newman.pdf>

83. **myExperiment Ontology Modules** (2009)
<http://web.archive.org/web/20091115080336/http://3a%2f%2frdf.myexperiment.org/ontologies>
84. **Web Annotation as a First-Class Object**
Paolo Ciccarese, Stian Soiland-Reyes, Tim Clark
IEEE Internet Computing (2013-11) <https://doi.org/gmgtkn>
DOI: [10.1109/mic.2013.123](https://doi.org/10.1109/mic.2013.123)
85. **PROV-O: The PROV Ontology**
Timothy Lebo, Satya Sahoo, Deborah McGuinness, Khalid Belhajjame, James Cheney, David Corsar, Daniel Garijo, Stian Soiland-Reyes, Stephan Zednik, Jun Zhao
(2013-04-30) <http://www.w3.org/TR/2013/REC-prov-o-20130430/>
86. **What Is Ontology Reuse?**
Megan Katsumi, Michael Grüninger
Formal Ontology in Information Systems (2016) <https://doi.org/10.3233/978-1-61499-660-6-9>
DOI: [10.3233/978-1-61499-660-6-9](https://doi.org/10.3233/978-1-61499-660-6-9) · ISBN: [978-1-61499-660-6](https://doi.org/10.3233/978-1-61499-660-6)
87. **Enabling FAIR research in Earth Science through research objects**
Andres Garcia-Silva, Jose Manuel Gomez-Perez, Raul Palma, Marcin Krystek, Simone Mantovani, Federica Foglini, Valentina Grande, Francesco De Leo, Stefano Salvi, Elisa Trasatti, ... Ilkay Altintas
Future Generation Computer Systems (2019-09)
DOI: [10.1016/j.future.2019.03.046](https://doi.org/10.1016/j.future.2019.03.046)
88. **Toward Enabling Reproducibility for Data-Intensive Research Using the Whole Tale Platform**
Chard Kyle, Gaffney Niall, Hategan Mihael, Kowalik Kacper, Ludäscher Bertram, McPhillips Timothy, Nabrzyski Jarek, Stodden Victoria, Taylor Ian, Thelen Thomas, et al.
Advances in Parallel Computing (2020)
DOI: [10.3233/apc200107](https://doi.org/10.3233/apc200107)
89. **FAIR digital objects for science: from data pieces to actionable knowledge units**
Koenraad De Smedt, Dimitris Koureas, Peter Wittenburg
Publications (2020-04-11)
DOI: [10.3390/publications8020021](https://doi.org/10.3390/publications8020021)
90. **Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities**
Sarah Cohen-Boulakia, Khalid Belhajjame, Olivier Collin, Jérôme Chopard, Christine Froidevaux, Alban Gaignard, Konrad Hinsén, Pierre Larmande, Yvan Le Bras, Frédéric Lemoine, ... Christophe Blanchet
Future Generation Computer Systems (2017-10)
DOI: [10.1016/j.future.2017.01.012](https://doi.org/10.1016/j.future.2017.01.012)
91. **Provenance trails in the Wings/Pegasus system**
Jihie Kim, Ewa Deelman, Yolanda Gil, Gaurang Mehta, Varun Ratnakar
Concurrency and Computation: Practice and Experience (2008-04-10) <http://doi.wiley.com/10.1002/cpe.1228>
DOI: [10.1002/cpe.1228](https://doi.org/10.1002/cpe.1228)
92. **Practical computational reproducibility in the life sciences.**
Björn Grüning, John Chilton, Johannes Köster, Ryan Dale, Nicola Soranzo, Marius van den Beek, Jeremy Goecks, Rolf Backofen, Anton Nekrutenko, James Taylor
Cell Systems (2018-06-27)
DOI: [10.1016/j.cels.2018.03.014](https://doi.org/10.1016/j.cels.2018.03.014) · PMID: [29953862](https://pubmed.ncbi.nlm.nih.gov/29953862/) · PMCID: [PMC6263957](https://pubmed.ncbi.nlm.nih.gov/PMC6263957/)

93. **A new genomic blueprint of the human gut microbiota.**
Alexandre Almeida, Alex L Mitchell, Miguel Boland, Samuel C Forster, Gregory B Gloor, Aleksandra Tarkowska, Trevor D Lawley, Robert D Finn
Nature (2019-02-11)
DOI: [10.1038/s41586-019-0965-1](https://doi.org/10.1038/s41586-019-0965-1) · PMID: [30745586](https://pubmed.ncbi.nlm.nih.gov/30745586/) · PMCID: [PMC6784870](https://europepmc.org/article/PMC/PMC6784870)
94. **FTP index of /pub/databases/metagenomics/umgs_analyses/**
EMBL-EBI Microbiome Informatics Team
ftp.ebi.ac.uk (2019-09-12) http://ftp.ebi.ac.uk/pub/databases/metagenomics/umgs_analyses/
95. **GitHub - Finn-Lab/MGS-gut: Analysing Metagenomic Species (MGS)**
EMBL-EBI Microbiome Informatics Team
GitHub <https://github.com/Finn-Lab/{MGS}-gut>
96. **I am looking for which bioinformatics journals encourage authors to submit their code/pipeline/workflow supporting data analysis**
Stian Soiland-Reyes
Twitter (2020-04-16) <https://twitter.com/soilandreyes/status/1250721245622079488>
97. **Re-run, Repeat, Reproduce, Reuse, Replicate: Transforming Code into Scientific Contributions.**
Fabien CY Benureau, Nicolas P Rougier
Frontiers in Neuroinformatics (2017)
DOI: [10.3389/fninf.2017.00069](https://doi.org/10.3389/fninf.2017.00069) · PMID: [29354046](https://pubmed.ncbi.nlm.nih.gov/29354046/) · PMCID: [PMC5758530](https://europepmc.org/article/PMC/PMC5758530)
98. **What is Reproducibility? The R* Brouhaha**
Carole Goble
(2016-09-09) <http://repscience2016.research-infrastructures.eu/img/{CaroleGoble}-{ReproScience2016v2}.pdf>
99. **Robust Cross-Platform Workflows: How Technical and Scientific Communities Collaborate to Develop, Test and Share Best Practices for Data Analysis**
Steffen Möller, Stuart W. Prescott, Lars Wirzenius, Petter Reinholdtsen, Brad Chapman, Pjotr Prins, Stian Soiland-Reyes, Fabian Klötzl, Andrea Bagnacani, Matúš Kalaš, ... Michael R. Crusoe
Data Science and Engineering (2017-11-16)
DOI: [10.1007/s41019-017-0050-4](https://doi.org/10.1007/s41019-017-0050-4)
100. **Bioconda: sustainable and comprehensive software distribution for the life sciences.**
Björn Grüning, Ryan Dale, Andreas Sjödín, Brad A Chapman, Jillian Rowe, Christopher H Tomkins-Tinch, Renan Valieris, Johannes Köster, Bioconda Team
Nature Methods (2018)
DOI: [10.1038/s41592-018-0046-7](https://doi.org/10.1038/s41592-018-0046-7) · PMID: [29967506](https://pubmed.ncbi.nlm.nih.gov/29967506/)
101. **BioContainers: an open-source and community-driven framework for software standardization.**
Felipe da Veiga Leprevost, Björn A Grüning, Saulo Alves Aflitos, Hannes L Röst, Julian Uszkoreit, Harald Barsnes, Marc Vaudel, Pablo Moreno, Laurent Gatto, Jonas Weber, ... Yasset Perez-Riverol
Bioinformatics (2017-08-15)
DOI: [10.1093/bioinformatics/btx192](https://doi.org/10.1093/bioinformatics/btx192) · PMID: [28379341](https://pubmed.ncbi.nlm.nih.gov/28379341/) · PMCID: [PMC5870671](https://europepmc.org/article/PMC/PMC5870671)
102. **Community-driven computational biology with Debian Linux.**
Steffen Möller, Hajo Nils Krabbenhöft, Andreas Tille, David Paleino, Alan Williams, Katy Wolstencroft, Carole Goble, Richard Holland, Dominique Belhachemi, Charles Plessy
BMC Bioinformatics (2010-12-21)
DOI: [10.1186/1471-2105-11-s12-s5](https://doi.org/10.1186/1471-2105-11-s12-s5) · PMID: [21210984](https://pubmed.ncbi.nlm.nih.gov/21210984/) · PMCID: [PMC3040531](https://europepmc.org/article/PMC/PMC3040531)

103. **The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update.**

Enis Afgan, Dannon Baker, Bérénice Batut, Marius van den Beek, Dave Bouvier, Martin Cech, John Chilton, Dave Clements, Nate Coraor, Björn A Grüning, ... Daniel Blankenberg

Nucleic Acids Research (2018-07-02)

DOI: [10.1093/nar/gky379](https://doi.org/10.1093/nar/gky379) · PMID: [29790989](https://pubmed.ncbi.nlm.nih.gov/29790989/) · PMCID: [PMC6030816](https://pubmed.ncbi.nlm.nih.gov/PMC6030816/)

104. **Beyond authorship: attribution, contribution, collaboration, and credit**

Amy Brand, Liz Allen, Micah Altman, Marjorie Hlava, Jo Scott

Learned Publishing (2015-04-01) <https://doi.org/gc6v3m>

DOI: [10.1087/20150211](https://doi.org/10.1087/20150211)

1. IRIs[[20](#)] are a generalisation of URIs (which include well-known http/https URLs), permitting international Unicode characters without % -encoding, commonly used on the browser address bar and in HTML5.[↵](#)
2. The avid reader may spot that the RO-Crate Metadata file use the extension `.json` instead of `.jsonld`, this is to emphasize the developer expectations as a JSON format, while the file's JSON-LD nature is secondary.[↵](#)
3. Recommended properties for shown types in Listing 1 also include `description`, `datePublished`, `encodingFormat`, `encodingFormat`, `affiliation`, `contactPoint`, `publisher`, `funder`, `citation`, `identifier`, `keywords`, `subjectOf`; these properties and corresponding contextual entities are excluded here for brevity.[↵](#)
4. CWLProv and TavernaProv predate RO-Crate, but use RO-Bundle[[61](#)], a similar Research Object packaging method with JSON-LD metadata.[↵](#)
5. IEEE 2791-2020 do permit user extensions in the *extension domain* by referencing additional JSON Schemas.[↵](#)
6. The Endings Project is a five-year project funded by the Social Sciences and Humanities Research Council (SSHRC) that is creating tools, principles, policies and recommendations for digital scholarship practitioners to create accessible, stable, long-lasting resources in the humanities.[↵](#)
7. For simplicity, blank nodes are not included in this formalisation, as RO-Crate [recommends the use of IRI identifiers](#)[↵](#)
8. The full list of types, relations and attribute properties from the RO-Crate specification are not included. Examples shown include `datePublished`, `CreativeWork` and `name`.[↵](#)
9. This simplification does not cover the extensive list of literal datatypes built-in to RDF 1.1, only strings and decimal real numbers. Likewise, language of literals are not included.[↵](#)
10. **Limitations:** Contextual entities not related from the RO-Crate (e.g. using inverse relations to a data entity) would not be covered by the single direction *Mentions*(*R*, *s*) production rule; see [issue 122](#). The `datePublished(e, d)` rule do not include syntax checks for the ISO 8601 datetime format. Compared with RO-Crate examples, this generated JSON-LD does not use a `@context` as the IRIs are produced unshortened, a post-step could do JSON-LD Flattening with a versioned RO-Crate context. The `@type` expansion is included for clarity, even though this is also implied by the `type(e, t)` expansion to `Relation(e, xsd:type)`.[↵](#)