1.-Overview-of-Latimer-Core.md

1.1 Summary

Digitisation of objects held in natural history collections is accelerating worldwide, but object-level digitisation is a time- and resource-intensive process. By describing the objects in their care at a higher level, organisations are able to publicise and share the core characteristics of their collections more easily. Collection-level descriptions are an established concept in the natural science community, but no standards or frameworks exist in this space. Latimer Core has been developed to fill this gap: it is designed to be a flexible, sustainable framework of terms that can be assembled to create records that accurately represent a grouping of objects at any level of granularity, from an entire collection of an institution to a few objects in a single drawer.

1.2 What is Latimer Core?

Latimer Core (LtC), named after Marjorie Courtenay-Latimer (https://en.wikipedia.org/wiki/Marjorie_Courtenay-Latimer) and the Coelacanth, is a data standard for describing collections. It has been designed to support the representation and discovery of the **groups of items (https://github.com/tdwg/ltc/wiki/4.-Classes#the-objectgroup-class)** that those collections and their subcomponents encompass. The LtC classes and their properties (collectively called terms) aim to represent information that describes these groups of things in enough detail to inform deeper discovery of the resources they contain.

The LtC standard has significant overlap with existing data standards that represent concepts such as individual objects and occurrences (Darwin Core (https://dwc.tdwg.org/), ABCD (https://www.tdwg.org/standards/abcd/)) and organisations, people and activities (W3C ORG Ontology (https://www.w3.org/TR/vocab-org/), W3C PROV Ontology (https://www.w3.org/TR/prov-o/), Schema.org (https://schema.org/)). Where possible, the LtC standard has either borrowed terms directly from these other standards or is less formally aligned with them. As far as possible, the terms included in the LtC standard should not preclude their use across domains. The LtC standard also introduces more rules around data structure than are applied in some other TDWG standards (e.g., Darwin Core), in that properties may only be used within the context of their parent class, and some limitations are applied to the relationships that may exist between classes.

LtC is intended to be sufficiently flexible and scalable to apply to a wide range of use cases, from describing the overall collections holdings of an institution to the contents of a single drawer of material. Various approaches are used to enable this flexibility, including the use of generic classes to represent organisations, people, roles and identifiers, and allowing flexible relationships for constructing data models that meet different needs. The Latimer Core Scheme (https://github.com/tdwg/ltc/wiki/3.-Latimer-Core-Schemes) concept is introduced to enable adopters to specify rules and constraints in the use of the LtC standard for a specific implementation. For example, this may specify the subset of the LtC classes and properties that are relevant to the use case and which of those are mandatory or optional, and define a set of metrics (e.g., object counts or digitisation percentages) to be captured for a breakdown of the collections.

The central concept of the standard is the `ltc:ObjectGroup (https://github.com/tdwg/ltc/wiki/4.-Classes#the-objectgroup-class)` class, which represents ‘an intentionally grouped set of objects with one or more common characteristics’. Arranged around `ltc:ObjectGroup` are a set of classes that are commonly used to describe and classify the objects within `ltc:ObjectGroup`, classes covering aspects of the custodianship, management and tracking of the collections, a generic class (`ltc:MeasurementOrFact`) for storing qualitative or quantitative measures for the `ltc:ObjectGroup`, and a set of classes that are used to describe the structure and description of the dataset. A summary of the classes within the standard is shown in **Figure 1** below.

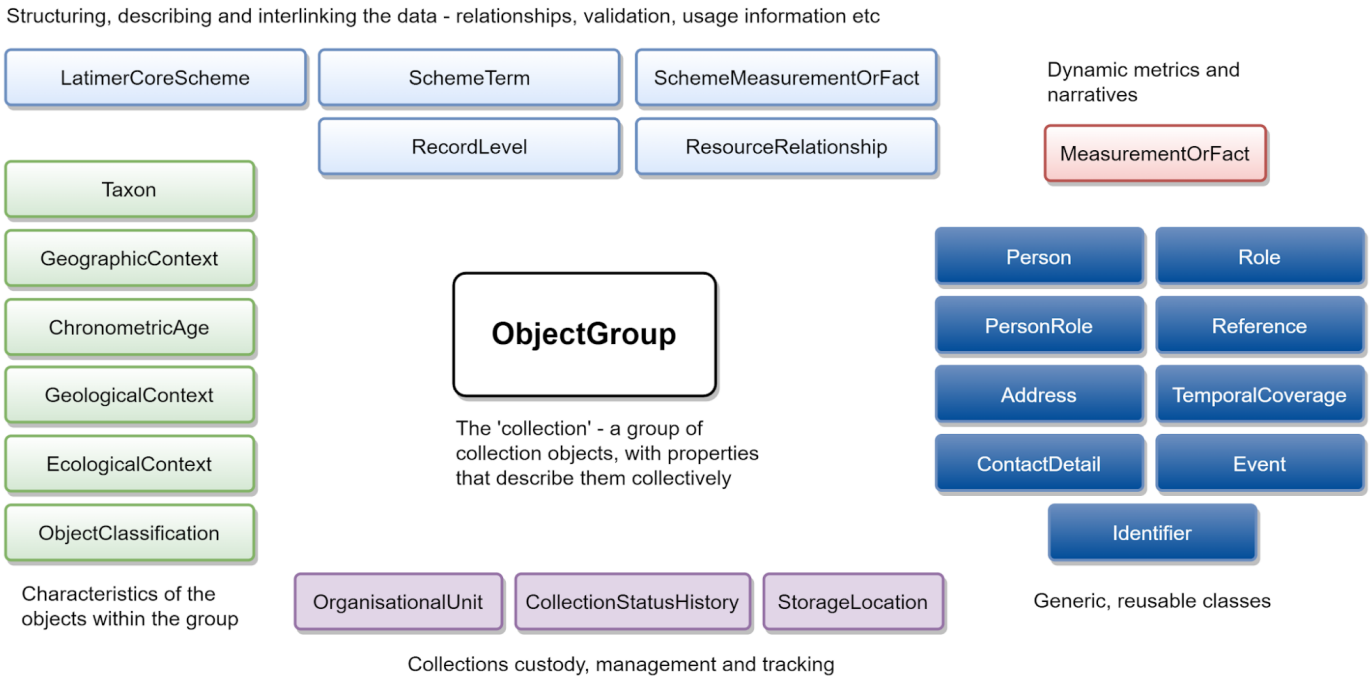


Figure 1: Overview and informal categorisation of Latimer Core standard classes for describing the object group's characteristics (green), collections custody (purple), generic reusable information types (dark blue), metrics (red), and data structure and links (light blue).

1.3 How is Latimer Core structured?

The LtC standard needs to be flexible in order to support the wide range of use cases identified for the structural, qualitative, and quantitative aspects of collections. There are some rules for which classes may be linked together and a very small number of mandatory elements to enforce a basic level of consistency of use across LtC implementations. Within these constraints, there is considerable scope for using the standard with different data modeling paradigms, such as relational, graph and dimensional approaches.

With this flexibility, there are challenges in ensuring that LtC datasets can achieve the required degree of interoperability. It is anticipated that, for different use cases, application profiles will be developed to support more targeted and simplified implementations within the wider LtC standard scope. For example, these may define:

- which classes and properties from the overall standard are included, which are mandatory, and which may be repeated
- the controlled vocabularies to be used for particular terms
- the relationship model to be used
- the metrics and descriptors to be included using the Measurement Or Fact class (see Metrics and Narratives section)

The Latimer Core Scheme class, in conjunction with the related Scheme Term and Scheme Measurement Or Fact classes, provide a rudimentary starting place for defining these application profiles. Depending on the serialisation and/or data platform used to manage LtC data for a given implementation, more specific and detailed approaches may also be used to achieve this, such as JSON Schema, XML Schema, RDF Shape Expressions or relational database schemas.

1.4 Where and how can Latimer Core be used?

Latimer Core is intended to be a generic data standard that allows collection descriptions to be represented in a range of formats to suit different use cases. These may include simple CSV files, exchange and linked data formats such as JSON(-LD), XML and RDF, and relational and non-relational database models. While the majority of reference examples are currently demonstrated with JSON, work will continue on building up a suite of examples and reference implementations using other formats.

Audiences

There are three main audiences for this documentation:

1. **Data aggregators** - users or groups who have a need to receive standardised collection descriptions.
2. **Data providers** - users or groups who have collections information that they would like to share.
3. **Data users** - users or groups who want to understand in more detail the information provided by LtC collection records and collection catalogues implementing the LtC standard.

Taking into account the variety of use-cases (documented in this wiki's Use Case section (https://github.com/tdwg/ltc/wiki/9.-Use-Cases), a github document (https://github.com/tdwg/cd/tree/master/reference/use_cases), and a google sheet (https://docs.google.com/spreadsheets/d/1SsfwogZ88TgouDJ7EoDqXJFol-eVs7aYdFx504qJNzc/htmlview?pru=AAABfyxGPeI*Y85ToB8bLmUyzDSk3_wuuA)) for the standard, it is intended that data-aggregators should define a minimal set of classes, properties and relationships between them that best suit their needs. The `ltc:LatimerCoreScheme` class is important in this context as a method for communicating to data providers the shape and types of information that they need to provide.

Who uses Latimer Core?

Whilst the initial use cases for the development of the LtC standard were contributed by natural history museums and biodiversity informatics communities, it is not intended to preclude extension to include other types of collection. The standard is intended to be broadly useful for:

- data providers and users who need to share and aggregate structured collections information
- data aggregators and collections registries who need to standardise collection- and institution-level data from multiple providers
- collections managers who need to inventory backlog and prioritise collections management and digitisation efforts
- museum staff who need to generate collection metrics and summaries
- communities who need to track and share collection history and provenance
- developers of collections management systems and other digital tools who need to support collection-level data and functionality within their platforms

1.5 Relationships with other standards

Many of the data concepts relevant to collection descriptions are similar to those reflected in specimen-level data standards - such as the type of object, geographic origins and taxonomy - or else are generic concepts like people, addresses and identifiers. Consequently, a concerted effort was made to borrow appropriate terms from existing standards rather than defining them anew, which brings the benefit of closer alignment with related standards.

To improve interoperability and reduce redundancy, some LtC properties reference and re-use properties from existing standards. Where the LtC version of a property's definition or permitted usage differs from the original, it will be narrower in scope. The provenance of any borrowed properties is referenced in the LtC normative documentation (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md). Any definitions which have been modified will note the LtC definition in the 'Usage' field and the original in the 'Definition' field, for example:

Label	Genus
Definition	The full scientific name of the genus in which the taxon is classified.
Usage	The full scientific name of the genus in which the collection's taxa are classified.
Existing property	dwc:genus
Existing class	dwc:Taxon

Table 1: Example of a property used in the Latimer Core standard that has been borrowed from the dwc:Taxon class

Any terms which aren't borrowed directly from elsewhere, but are conceptually aligned with terms found in other standards reference this in

the LtC term-level ‘notes’ field.

The full list and provenance of borrowed terms can be found in the LtC normative documentation (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md). A summary is provided in **Table 2**, including also sources that informed the development of LtC and are the origin of informally derived terms.

Standard name	Namespace	Documentation
Darwin Core (including extensions)	dwc	https://dwc.tdwg.org/terms/ (https://dwc.tdwg.org/terms/)
ABCD(EFG)	abcd	https://abcd.tdwg.org/ (https://abcd.tdwg.org/)
AIISO	aiiso	https://vocab.org/aiiso/ (https://vocab.org/aiiso/)
Dublin Core	dcterms	https://www.dublincore.org/specifications/dublin-core/dcmi-terms/ (https://www.dublincore.org/specifications/dublin-core/dcmi-terms/)
PROV (Provenance)	prov	https://www.w3.org/TR/prov-overview (https://www.w3.org/TR/prov-overview)
The Organization Ontology	org	http://www.w3.org/TR/vocab-org/ (http://www.w3.org/TR/vocab-org/)
FOAF	foaf	http://xmlns.com/foaf/spec/ (http://xmlns.com/foaf/spec/)
Schema.org	schema	https://schema.org/ (https://schema.org/)

Table 2: List of standards from which terms in LtC have been borrowed or derived.

Data Type	Standard
Date	ISO 8601 (https://www.iso.org/iso-8601-date-and-time-format.html)

Table 3: List of data type standards recommended for use with Latimer records.

1.6 Mapping using SKOS and SSSOM

Over the course of LtC development, efforts were made to identify and align with relevant standards and vocabularies, and adopt existing terms from them where possible. During the expert review phase, a more structured approach was proposed and implemented using the Simple Knowledge Organization System (SKOS) (https://www.w3.org/2004/02/skos/) mappingRelation vocabulary. This exercise helped to better describe the nature of the mappings between new LtC terms and related terms in other standards, and also to validate decisions around the borrowing of existing terms for inclusion in LtC.

Properties used:

- skos:**mappingRelation** (https://www.w3.org/TR/skos-reference/#mappingRelation (https://www.w3.org/TR/skos-reference/#mappingRelation)) --> reference
- skos:**exactMatch** (https://www.w3.org/TR/skos-reference/#exactMatch (https://www.w3.org/TR/skos-reference/#exactMatch)) --> Exact equivalence mapping link
- skos:**closeMatch** (https://www.w3.org/TR/skos-reference/#closeMatch (https://www.w3.org/TR/skos-reference/#closeMatch)) --> Close equivalence mapping link
- skos:**narrowMatch** (https://www.w3.org/TR/skos-reference/#narrowMatch (https://www.w3.org/TR/skos-reference/#narrowMatch)) --> Hierarchical mapping link between <A> and <B> (where <B> is narrower than <A>)
- skos:**broadMatch** (https://www.w3.org/TR/skos-reference/#broadMatch (https://www.w3.org/TR/skos-reference/#broadMatch)) --> Hierarchical mapping link between <A> and <B> (where <B> is broader than <A>)

- skos:**relatedMatch** (https://www.w3.org/TR/skos-reference/#relatedMatch (https://www.w3.org/TR/skos-reference/#relatedMatch)) --> Associative mapping link

Table 4 shows an the example of the mapping between the `ltc:collectionName` and the Dublin Core `dcterms:title` property.

term_localName	skos_mappingRelation	related_termName
http://rs.tdwg.org/ltc/terms/collectionName (http://rs.tdwg.org/ltc/terms/collectionName)	broadMatch	http://purl.org/dc/terms/title (http://purl.org/dc/terms/title)

Table 4: Example of SKOS mapping

A provisional set of the LtC SKOS mappings (https://github.com/tdwg/ltc/blob/main/source/terms/mapping/ltc_skos_mapping.csv) can be found in the Latimer Core GitHub repository.

A further exercise was also carried out that used elements of the Simple Standard for Sharing Ontological Mappings (SSSOM) (https://github.com/mapping-commons/sssom) to start to develop a more comprehensive set of metadata around these mappings, including further information about the justification for the mapping decisions and provenance around when the mappings were created and by whom. The SSSOM mappings (https://github.com/tdwg/ltc/blob/main/source/terms/mapping/ltc_sssom_mapping.csv) are also available in the LtC GitHub repository, and as with the SKOS mappings are expected to be refined and expanded over time.

1.7 Extending the standard

Latimer Core is intended to be a standard that can be expanded to include any genre of collection, physical or digital. As such an attempt has been made to ensure that the classes and properties contained within it do not preclude such extensions.

Flexibility

- The modular design of the standard allows a high degree of flexibility in terms of schema definition: composition and architecture can be designed around the requirements of the designers system/environment, but class and term definitions, data types and vocabularies are controlled in order to support and encourage schema and LtC re-use. (flexibility is in configuration, not customisation)
- Mandatory fields are rarely specified in the LtC standard. They are intended to be defined at the schema level, according to the requirements of the specific use-case or implementation.
- Generic classes are used where possible to provide the opportunity to add non-standard or unqualified metrics and identifiers to a LtC record, but class properties should hold enough information to enable definition of logical rules to support automated parsing, ingestion and data quality measures.

Extensibility

- Extensibility was a priority during design: requirements change, most of the collections that we are seeking to describe using the standard haven't been digitised before, so novel use cases are likely to emerge.
- Generic classes support extensibility: if new concrete classes are required, they can be added without affecting the standard definition or breaking functionality of existing implementations/schema (state/behaviour will not diverge significantly from the parent class/other implementations of the same parent class).
- Smaller, single-responsibility classes favoured: easier to understand + communicate to users, fewer dependencies, can be extended without impacting other classes.

2.-Glossary.md

We understand that many of the words used in the descriptions of data standards are interchangeable. For the purposes of documenting the use of the LtC standard in a consistent fashion, we have opted to use the following definitions. This by no means makes any presumptions about their use in any other documentation therefore it should NOT be assumed that anyone else uses the same definitions.

- **Attributes:** are the values that a property can take. Attributes can be prescribed/controlled for a particular property and can be thought of as lookup lists, or they may be uncontrolled and thought of as free-form strings. For example 'George', 'Rattus', 'under the tree by the rock'.
- **Class:** a logical grouping of properties, typically defining an entity or thing we are talking about. For example: a Person, a group of objects (ObjectGroup class), etc.
- **Controlled Vocabulary:** a prescribed set of values that a property can have. Controlled vocabularies improve search functionality, consistency, standardization and reuse, as well as form the basis for machine-readability and -actionability.
- **Property:** is a unitary concept (e.g., genus, givenName) within a class,_ _each representing a specific piece of data about the thing described by the class. In other words it is "*What we can say about the thing defined by the class"*.
- **Repeatability**: whether or not certain classes and/or properties can be repeated. For example, a Person can have several Identifiers of different identifierTypes. A collection (i.e. an ObjectGroup) can have several instances of the CollectionStatusHistory class to describe its different states through time.
- **Reusability**: several classes can be reused, that is, they can be simultaneously attached to different classes to expand the sets of properties of those classes. For example, the Identifier or TemporalCoverage classes can be used in conjunction with many other classes to describe in more detail and with enhanced functionality the identity and temporal scope of different aspects (represented by classes) of a single collection.
- **Scheme:** is the word we use for the defined structure of the record. The schema should describe shape, structure and rules of composition.
- **Term:** is a general word for either a class, a property, or an attribute.
- **Vocabulary:** is another general word for a set of classes, properties and/or attributes.

3.-Latimer-Core-Schemes.md

LtC does not impose a single structure on its implementations. Instead it provides a series of classes that provide users with a standardised methodology for documenting the model that they choose to use.

3.1 Introduction to Latimer Core Schemes

When describing large collections it is anticipated that the same collections can be described using different schemes for different purposes. For instance a museum collection may be described based on "famous named" collections or collectors (e.g., Darwin, Spruce) if an aggregator has the need to "find" lost specimens from previously formed collections. The same collection may be described in whole or part based on taxonomic or geographic properties for the purpose of environmental or taxonomic research or funding. For potential examples of existing use cases that could be defined as Latimer Core Schemes, see the Use Cases section later in this document.

The `ltc:LatimerCoreScheme (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_CollectionDescriptionScheme)` class, and the supporting `SchemeTerm (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_SchemeTerm)` and `SchemeMeasurementOrFact (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_SchemeMeasurementOrFact)` classes are intended to provide some parameters around the purpose and expectations of the descriptions and to indicate if objects within the descriptions are assigned attributes that will cause errors in metrics if not explicitly noted.

Using these three classes enables you to build a 'profile' for your LtC implementation, so that you may:

1. describe the purpose of your Latimer Core Scheme (using the `LatimerCoreScheme.basisOfScheme` property)
2. define whether the `ObjectGroups` within the scheme overlap (i.e., a single object might be represented in more than one `ObjectGroup`) or are distinct (using the `LatimerCoreScheme.isDistinctObjects` property`)
3. apply restrictions on which terms within the overall LtC standard can be included, and which are mandatory (using the `SchemeTerm`

class)

4. define the metrics that you want to be included in the scheme via the `MeasurementOrFact` class (using the `SchemeMeasurementOrFact` class)

Essentially, LtC is a fairly broad and flexible standard which can be applied in multiple ways. While this allows it to support a broad range of collection description use cases, it also presents a risk that if its use isn't constrained appropriately to fit the use case, data coherency and usability may be compromised. In particular, defining 1. common metrics and 2. controlled vocabularies for appropriate terms are vital steps for making sure that the data are consistent and interoperable. The Latimer Core Scheme concept and related LtC standard terms are intended to help to support this process.

3.2 Defining a Latimer Core Scheme - an example process

Below is an example of steps that you can take to begin defining a new Latimer Core Scheme in the LtC standard, using the `LatimerCoreScheme`, `SchemeTerm` and `SchemeMeasurementOrFact` classes and properties.

Step 1: Use `LatimerCoreScheme` for the basic definition of your scheme

```
{
    "@context": {
        "ltc": "http://rs.tdwg.org/ltc/terms/"
    },

    "@type": "ltc:LatimerCoreScheme",
    "schemeName": "NHM London departmental collections",
    "basisOfScheme": "Collections inventory",
    "isDistinctObjects": true
}
```

This provides a name for the scheme, allowing it to be distinguished from other schemes that might be in the same dataset, and what the scheme is intended to be for. It also dictates that no single object is expected to be represented in more than one `ObjectGroup` within the scheme, so it should be safe to aggregate metrics within the scheme without the risk of, for example, counting the same object multiple times.

Step 2: add `SchemeTerm` to define the terms that are allowed in the dataset

```
{
    "@context": {
        "ltc": "http://rs.tdwg.org/ltc/terms/"
    },

    "@type": "ltc:LatimerCoreScheme",
    "schemeName": "NHM London departmental collections",
    "basisOfScheme": "Collections inventory",
    "isDistinctObjects": true,

    "ltc:hasSchemeTerm": [
        {
            "@type": "ltc:SchemeTerm",
            "termName": "ObjectGroup.collectionName",
            "isMandatoryTerm": true,
            "isRepeatableTerm": false
        },{
            "@type": "ltc:SchemeTerm",
            "termName": "ObjectGroup.preservationMethod",
            "isMandatoryTerm": false,
            "isRepeatableTerm": true
        },{
            "@type": "ltc:SchemeTerm",
            "termName": "Taxon",
            "isMandatoryTerm": false,
            "isRepeatableTerm": true
        },{
            "@type": "ltc:SchemeTerm",
            "termName": "ObjectGroup.Identifier",
            "isMandatoryTerm": true,
            "isRepeatableTerm": false
        },{
            "@type": "ltc:SchemeTerm",
            "termName": "ObjectGroup.Identifier.identifierValue",
            "isMandatoryTerm": true,
            "isRepeatableTerm": false
        },{
            "@type": "ltc:SchemeTerm",
            "termName": "OrganisationalUnit",
            "isMandatoryTerm": true,
            "isRepeatableTerm": false
        },{
            "@type": "ltc:SchemeTerm",
            "termName": "OrganisationalUnit.organisationalUnitName",
            "isMandatoryTerm": true,
            "isRepeatableTerm": false
        },{
            "@type": "ltc:SchemeTerm",
            "termName": "OrganisationalUnit.organisationalUnitType",
            "isMandatoryTerm": true,
            "isRepeatableTerm": false
        },{
            "@type": "ltc:SchemeTerm",
            "termName": "StorageLocation",
            "isMandatoryTerm": true,
            "isRepeatableTerm": false
        },{
            "@type": "ltc:SchemeTerm",
            "termName": "StorageLocation.locationName",
            "isMandatoryTerm": true,
            "isRepeatableTerm": false
        },{
            "@type": "ltc:SchemeTerm",
            "termName": "StorageLocation.locationType",
```

```
            "isMandatoryTerm": true,
            "isRepeatableTerm": false
        }
    ]
}
```

In human-readable terms, this says:

*"We want the collection to be broken down by department and building, so each subcollection **must** have one, and **only one**, department (`OrganisationalUnit`) and building (`StorageLocation`) attached. We need to have the type and name for each of those things so that we know what they are. Each subcollection should also have a single short name (`ObjectGroup.collectionName`) and an identifier (`ObjectGroup.Identifier`) so that humans and machines can tell them apart. It may be useful, but not critical, to get an idea of the taxa represented and preservation methods used in each of those subcollections. However not everyone will have the time to add that data, so we'll make that option available but not force it at this stage."*

This has implications on the structure of the data, as the `isMandatoryTerm` = "true" and `isRepeatableTerm` = "false" values for `OrganisationalUnit` and `StorageLocation` dictate that there should be one `ObjectGroup` created for every combination of (in this example) department and building. More information on LtC standard modeling approaches can be found in the ObjectGroups and relationships (https://github.com/tdwg/ltc/wiki/7.-Relationships#objectgroups-and-relationships) section.

As shown in the example above, dot notation may be used in the `ltc:termName` values to be explicit about which a. which class a property belongs to ("<class name>.<property name>") and b. which class a generic class should be attached to ("<class name>"."<class name>").

For example, the `Identifier` class may be attached to multiple other classes within LtC. By using "ObjectGroup.Identifier" and "ObjectGroup.Identifier.identifierValue" when defining those terms as being mandatory in the `LatimerCoreScheme` above, we can be clear that these `SchemeTerms` are setting rules about identifiers that are linked to an `ObjectGroup` , not identifiers that are linked to (for example) a `Person` or a `Taxon` .

Step 3: add `SchemeMeasurementOrFact` to define the quantitative and qualitative measures that we want to include in the dataset

```
{
    "@context": {
        "ltc": "http://rs.tdwg.org/ltc/terms/"
    },

    "@type": "ltc:LatimerCoreScheme",
    "schemeName": "NHM London departmental collections",
    "basisOfScheme": "Collections inventory",
    "isDistinctObjects": true,

    "ltc:hasSchemeTerm": [...],

    "ltc:hasSchemeMeasurementOrFact": [
        {
            "@type": "ltc:SchemeMeasurementOrFact",
            "schemeMeasurementType": "Object count",
            "isMandatoryMetric": true,
            "isRepeatableMetric": false
        },{
            "@type": "ltc:SchemeMeasurementOrFact",
            "schemeMeasurementType": "Percentage barcoded",
            "isMandatoryMetric": true,
            "isRepeatableMetric": false
        },{
            "@type": "ltc:SchemeMeasurementOrFact",
            "schemeMeasurementType": "Historical narrative",
            "isMandatoryMetric": false,
            "isRepeatableMetric": true
        }
  ],
}
```

In human-readable terms, this says:

*"For every subcollection, we expect people to provide **one**, and **only one**, estimate or count of the number of objects in that subcollection, and the same for the percentage of those objects that have been barcoded. If people have the time, we'll also provide the facility to add a historical narrative to describe the subcollection, but make that optional."*

The implication of this is that we would expect to see two instances of the `MeasurementOrFact` class, one with `measurementType` of "Object count" and one of "Percentage barcoded", for every `ObjectGroup` linked to the `LatimerCoreScheme`, and can validate against that expectation.

The principle of defining a new Latimer Core Scheme using the LatimerCoreScheme classes as demonstrated in the example above is similar to (and simpler than) constructs such as JSON Schema and RDF Schema. JSON Schema (json-schema.org (https://json-schema.org/)) and RDF Schema (www.w3.org/TR/rdf-schema (https://www.w3.org/TR/rdf-schema/)) constructs include their own respective "built-in" methods for defining mandatory and repeatable terms in a given record structure, so LtC implementations in JSON or RDF could use either of those respective Schema formats. However, the LatimerCoreScheme class enables schema-definitions in different data serialisation formats that do not have "built-in" schema validation methods.

Examples

In the example below a LtC description record for the Insects and Invertebrate Zoology collections at the Field Museum is created and its three-term LatimerCoreScheme (https://drive.google.com/file/d/1-JAZODO9yPfRiuluWvBkKI45EKQ0xGbn/view?usp=sharing) is included.

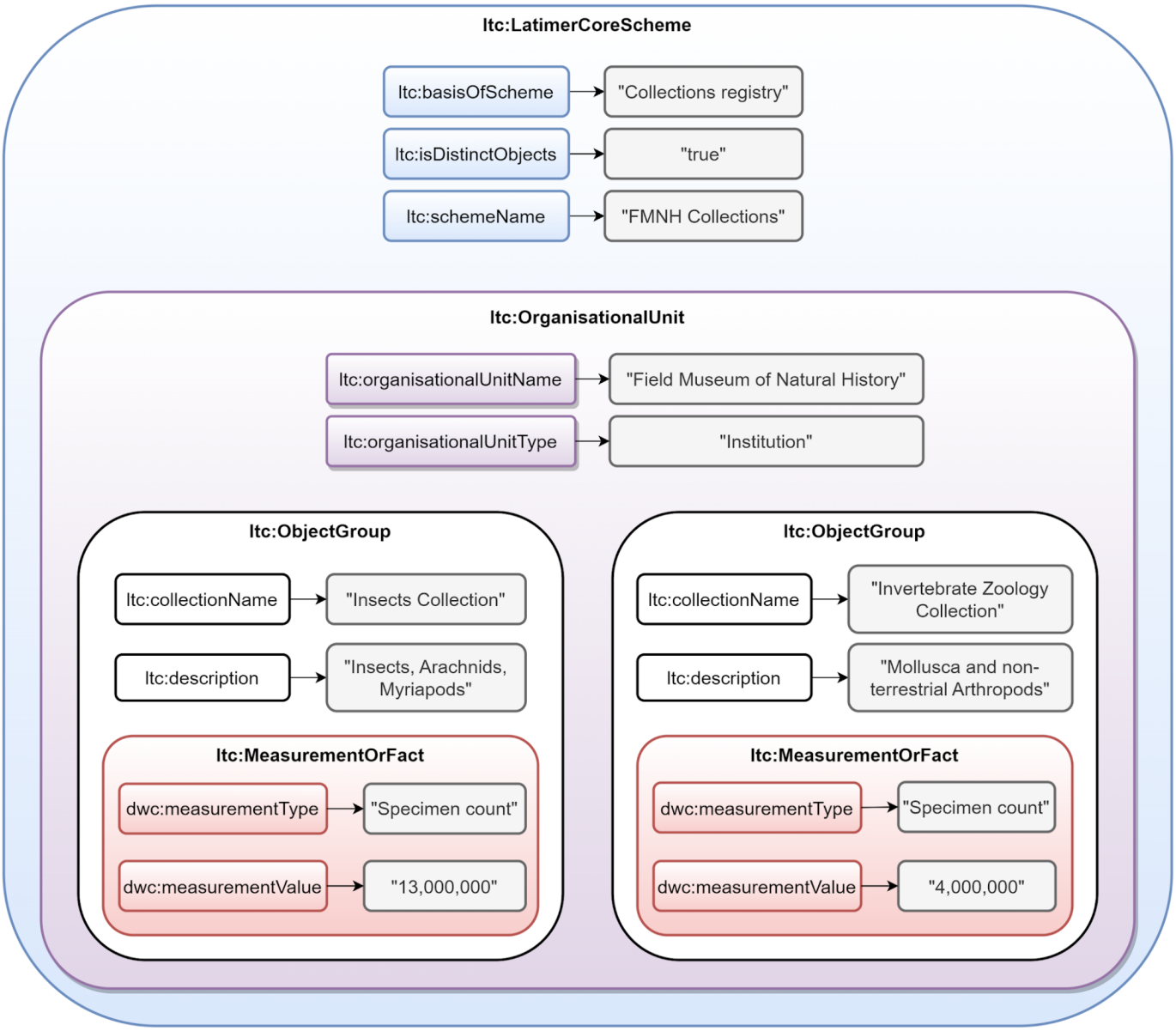


Figure 2: An example record structure that might be a useful scheme for an institution's contribution to a global registry of collections like GRSciColl (https://scientific-collections.gbif.org/).

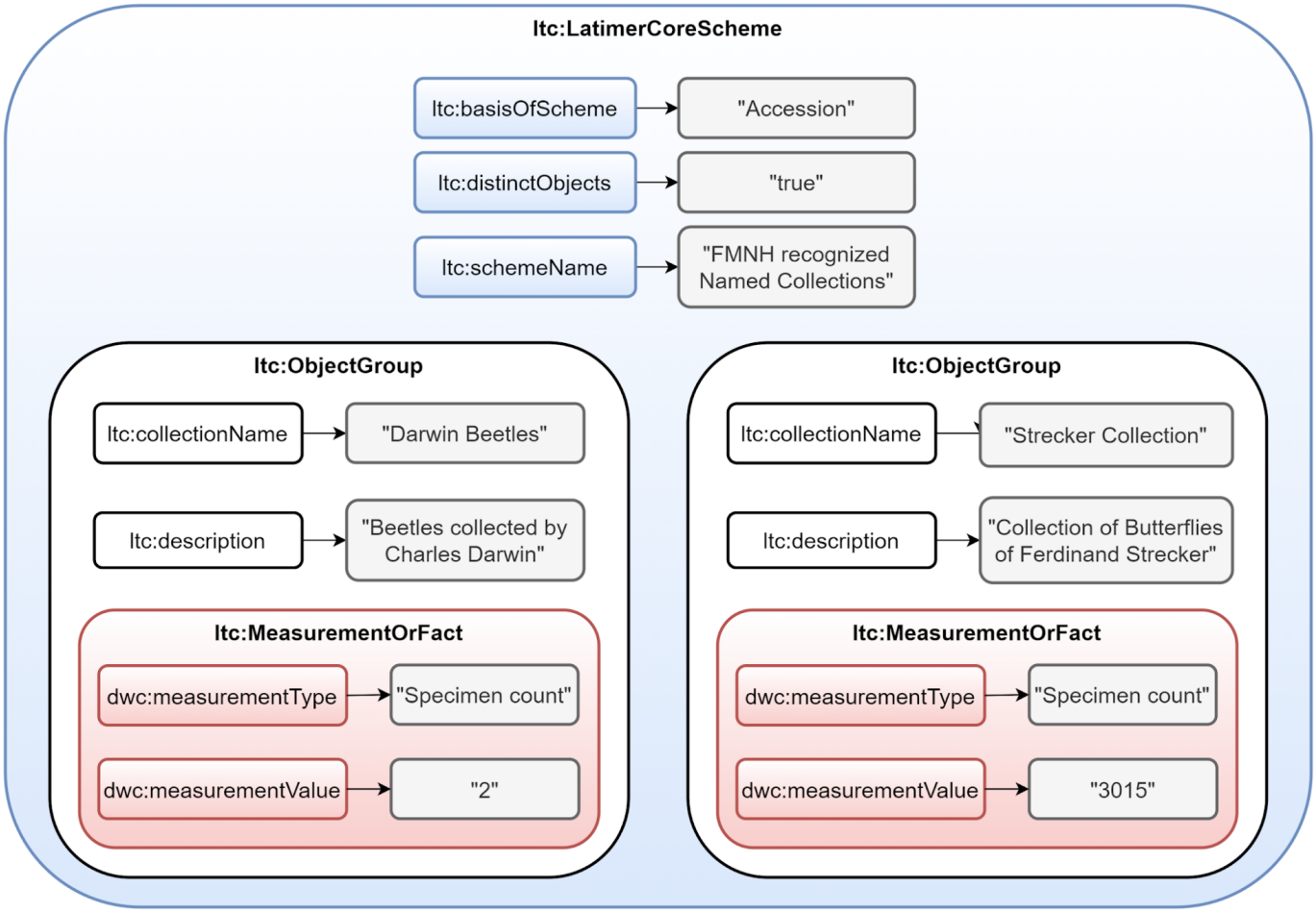


Figure 3: Another example record structure of a way to describe all of the “famous” collections within a larger collection. In this instance, the `OrganisationalUnit` class is omitted, as it’s implicit that this is a Field Museum of Natural History (FMNH) Latimer Core Scheme and so all collections in the dataset belong to the same institution.

In both of the above examples the `isDistinctObjects` term is ‘true’, because there is no overlap in objects between the two `ObjectGroups`, and so we can be sure that if the ‘Specimen count’ metric is being aggregated, nothing would be counted twice. However, if the two examples (“FMNH Collections” and “FMNH recognized Named Collections”) were to be combined as a single Latimer Core Scheme in the same dataset, the LatimerCoreScheme. `isDistinctObjects` term needs to be ‘false’ (**Figure 4**).

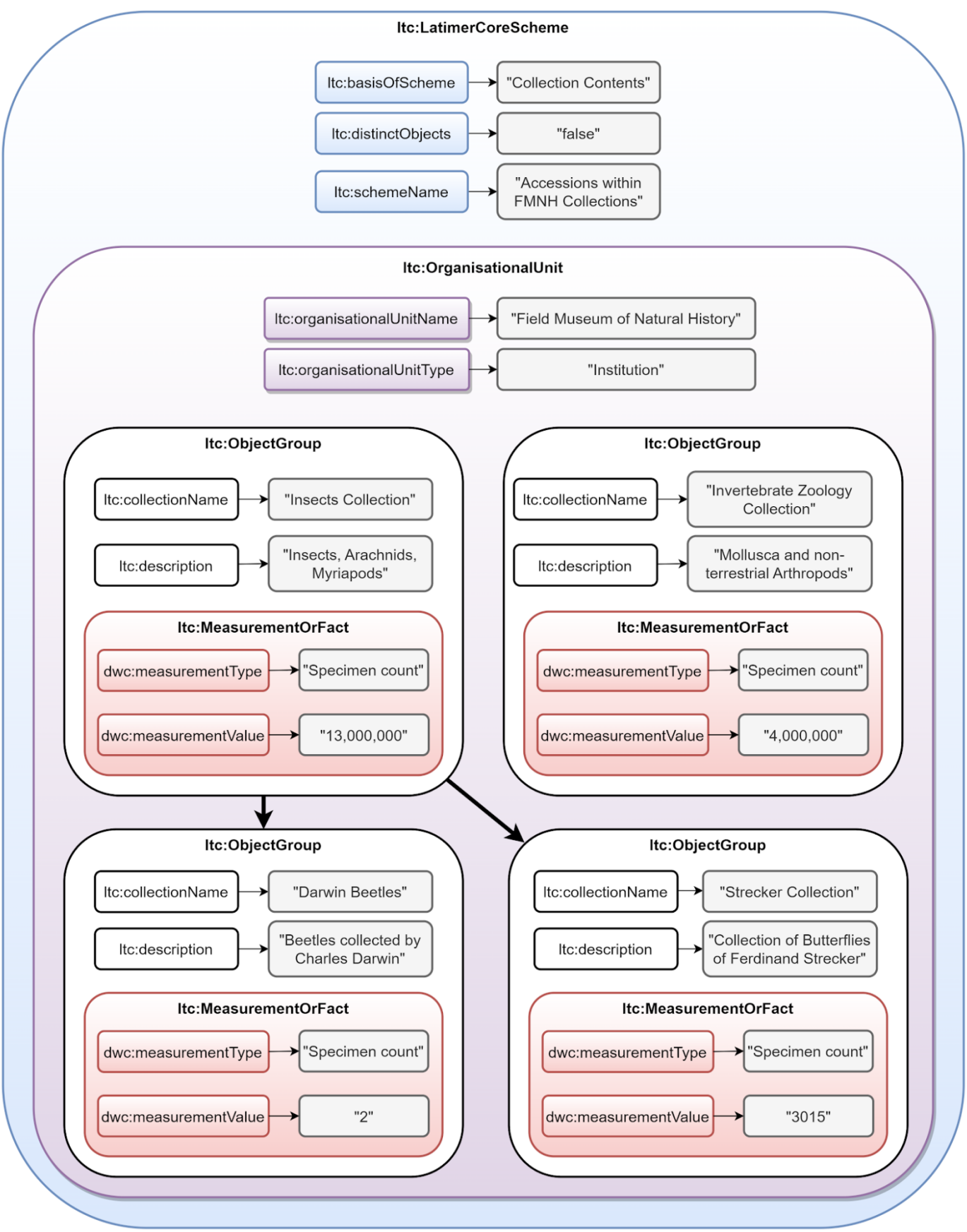


Figure 4: An example of a record-structure that combines ObjectGroups from the above examples, and has overlapping “Specimen count” measurements.

The `isDistinctObjects` term becomes ‘false’ because the “Darwin Beetles” and “Strecker Collection” `ObjectGroups` are actually contained

within the “Insect Collections” `ObjectGroup` , and so if we aggregate the ‘Specimen count’ metric across all of the `ObjectGroups` in the Latimer Core Scheme, the objects in those two smaller `ObjectGroups` would end up being counted twice. `isDistinctObjects` being set to false provides a warning that this is the case.

For this reason, combining the two schemes into one in this scenario is likely in this example to be detrimental, as it’s no longer easy to extract accurate aggregations of metrics from the dataset. It’s possible to maintain multiple `LatimerCoreSchemes` within the same LtC dataset, and if queries incorporate the `LatimerCoreScheme` into their logic (e.g., aggregate ‘Specimen count’ for all `ObjectGroups` in the “FMNH Collections” `LatimerCoreScheme`), then the issues with double-counting can still be avoided.

In addition, as shown in **Figure 5**, this makes it possible to add semantic relationships between `ObjectGroups` across the two schemes using the `ResourceRelationship` class for more detailed reporting - for example, to assert that the “Darwin Beetles” `ObjectGroup` in the “FMNH recognized Named Collections” `LatimerCoreScheme` ‘is part of’ the “Insect Collections” `ObjectGroup` in the “FMNH Collections” `LatimerCoreScheme` . More information on linking `ObjectGroups` can be found in the Linking ObjectGroups (https://github.com/tdwg/ltc/wiki/7.-Relationships#71-linking-objectgroups) section.

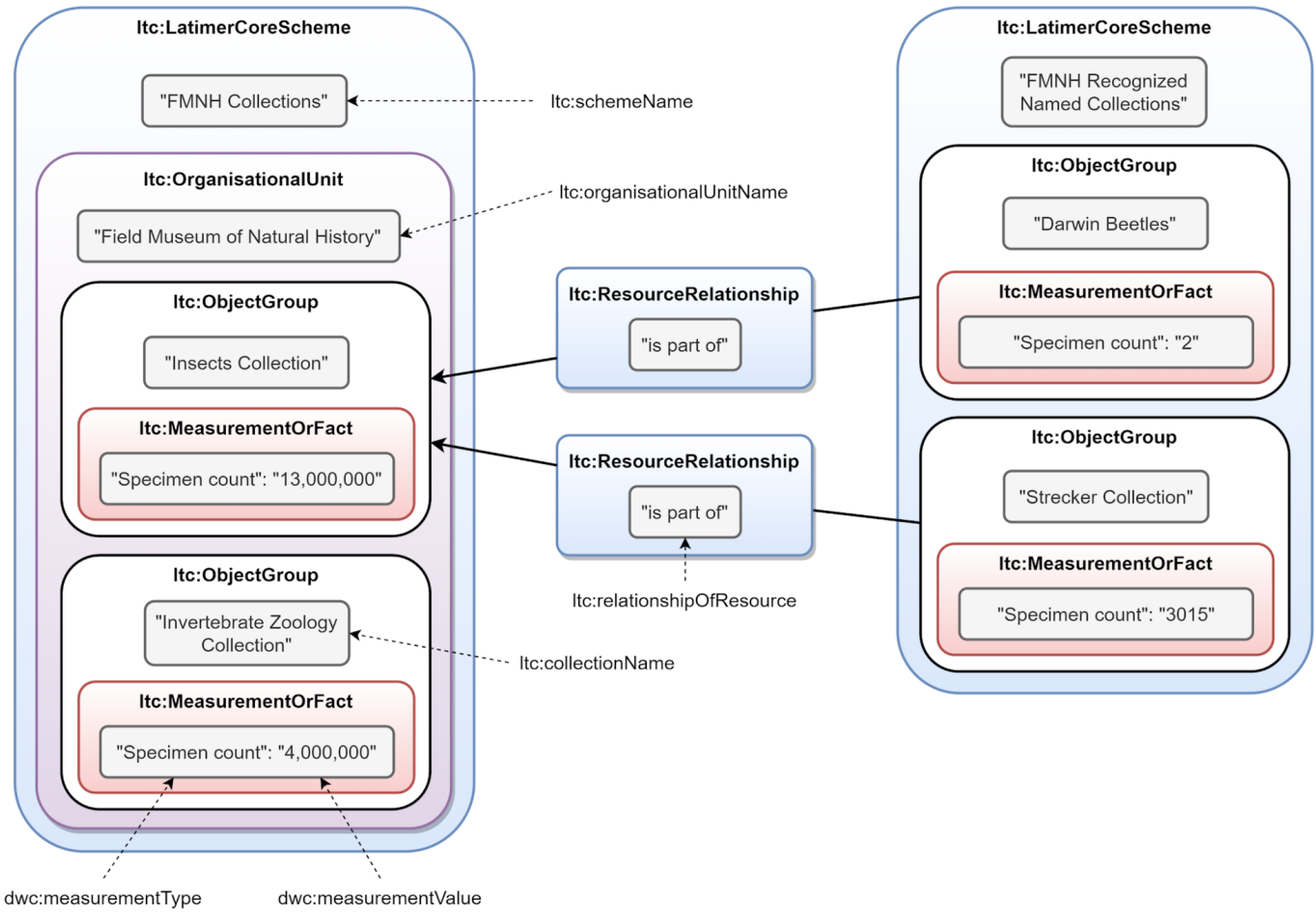


Figure 5: Example of maintaining two Latimer Core Schemes in parallel, with `ObjectGroup` to `ObjectGroup` relationships across schemes using the `ResourceRelationship` class.

4.-Classes.md

4.1 Summary of Latimer Core classes

For quick reference, **Table 5** below lists all of the classes included in the current version of the standard, plus some additional classes proposed but not yet incorporated.

Class Name	Description	Issue #	In LtC v1
Activity	An activity is something that occurs over a period of time and acts upon or with entities; it may include consuming, processing, transforming, modifying, relocating, using, or generating entities.	https://github.com/tdwg/cd/issues/310 (https://github.com/tdwg/cd/issues/310)	No
Address (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_Address)	A physical address for an organisational unit or person.	https://github.com/tdwg/cd/issues/208 (https://github.com/tdwg/cd/issues/208)	Yes
ChronometricAge (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_ChronometricAge)	The age of a specimen or related materials that is generated from a dating assay.	https://github.com/tdwg/cd/issues/268 (https://github.com/tdwg/cd/issues/268)	Yes
CollectionStatusHistory (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_CollectionStatusHistory)	A record of current and past statuses of the object group and the reason for status changes.	https://github.com/tdwg/cd/issues/120 (https://github.com/tdwg/cd/issues/120)	Yes
ContactDetail (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_ContactDetail)	Details of a method by which an entity such as a Person or OrganisationalUnit may be contacted.	https://github.com/tdwg/cd/issues/329 (https://github.com/tdwg/cd/issues/329)	Yes
EcologicalContext	The ecological and/or biogeographic classification of the region from which objects associated with the ObjectGroup were collected, or where an Event took place.		Yes
Event (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_Event)	An action that occurs at some location during some time.	https://github.com/tdwg/cd/issues/362 (https://github.com/tdwg/cd/issues/362)	Yes
GeographicContext (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_GeographicOrigin)	The geographic location from which objects associated with the ObjectGroup were collected, or where an Event took place.	https://github.com/tdwg/cd/issues/131 (https://github.com/tdwg/cd/issues/131)	Yes
GeologicalContext (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#dwc_GeologicalContext)	Geological information, such as stratigraphy, that qualifies a region or place.	https://github.com/tdwg/cd/issues/178 (https://github.com/tdwg/cd/issues/178)	Yes

Identifier (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_Identifier)	A numeric, textual value, or reference such as an IRI, that can be used to uniquely identify the object to which it is attached.	https://github.com/tdwg/cd/issues/126 (https://github.com/tdwg/cd/issues/126)	Yes
LatimerCoreScheme (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_LatimerCoreScheme)	A grouping of multiple ObjectGroups for a particular use case, purpose or implementation.	https://github.com/tdwg/cd/issues/109 (https://github.com/tdwg/cd/issues/109)	Yes
MeasurementOrFact (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#dwc_MeasurementOrFact)	A measurement of or fact about a class within the standard, or a relationship between the ObjectGroup and an associated class.	https://github.com/tdwg/cd/issues/289 (https://github.com/tdwg/cd/issues/289)	Yes
ObjectClassification (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_ObjectClassification)	An informal classification of the type of objects within the ObjectGroup, using a hierarchical structure.	https://github.com/tdwg/cd/issues/214 (https://github.com/tdwg/cd/issues/214)	Yes
ObjectGroup (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_ObjectGroup)	An intentionally grouped set of objects with one or more common characteristics.	https://github.com/tdwg/cd/issues/55 (https://github.com/tdwg/cd/issues/55)	Yes
OrganisationalUnit (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_OrganisationalUnit)	A unit within an organisational hierarchy which may be at, above or below the institutional level.	https://github.com/tdwg/cd/issues/197 (https://github.com/tdwg/cd/issues/197)	Yes
Person (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_Person)	A person (alive or dead).	https://github.com/tdwg/cd/issues/219 (https://github.com/tdwg/cd/issues/219)	Yes
PersonActivity	A link between a Person and an Activity to enable the role that the Person played in the Activity to be specified.	https://github.com/tdwg/cd/issues/314 (https://github.com/tdwg/cd/issues/314)	No
PersonRole (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_PersonRole)	A qualified association between a Person or OrganisationalUnit and an entity such as an ObjectGroup or MeasurementOrFact that enables the relationship to be contextualised with a specific role and time period.	https://github.com/tdwg/cd/issues/316 (https://github.com/tdwg/cd/issues/316)	Yes

RecordLevel (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_RecordLevel)	The machine-actionable information profile for the collection description digital object.	https://github.com/tdwg/cd/issues/43 (https://github.com/tdwg/cd/issues/43)	Yes
Reference (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#dwc_Reference)	A reference to external resources and information related to the class.	https://github.com/tdwg/cd/issues/231 (https://github.com/tdwg/cd/issues/231)	Yes
ResourceRelationship (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#dwc_ResourceRelationship)	A relationship between an instance of a class in the collection description standard to another instance of the same class, or an instance of a different class in the standard.	https://github.com/tdwg/cd/issues/279 (https://github.com/tdwg/cd/issues/279)	Yes
Role	The function of a Person with respect to an activity or entity.		Yes
SchemeMeasurementOrFact (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_SchemeMeasurementOrFact)	A type of measurement or fact used by the LatimerCoreScheme, and the rules relating to its application.	https://github.com/tdwg/cd/issues/117 (https://github.com/tdwg/cd/issues/117)	Yes
SchemeTerm (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_SchemeTerm)	A Latimer Core term used by the LatimerCoreScheme and the rules relating to its application.	https://github.com/tdwg/cd/issues/113 (https://github.com/tdwg/cd/issues/113)	Yes
StorageLocation (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_StorageLocation)	A physical location (such as a building, room, cabinet or drawer) within the holding institution where objects associated with the collection description are stored or exhibited.	https://github.com/tdwg/cd/issues/164 (https://github.com/tdwg/cd/issues/164)	Yes
Taxon (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#dwc_Taxon)	A group of organisms (sensu http://purl.obolibrary.org/obo/OBI_0100026 (http://purl.obolibrary.org/obo/OBI_0100026)) considered by taxonomists to form a homogeneous unit.	https://github.com/tdwg/cd/issues/269 (https://github.com/tdwg/cd/issues/269)	Yes
TemporalCoverage (https://github.com/tdwg/rs.tdwg.org/blob/latimer/process/page_build_scripts/index.md#ltc_TemporalCoverage)	The time period during which the related event, activity or status was occurring.	https://github.com/tdwg/cd/issues/333 (https://github.com/tdwg/cd/issues/333)	Yes

Table 5: A summary of the classes in the LtC standard, with links to the normative definitions and Github issues used to build the definitions.

4.2 Core elements of the standard

As described in the Summary of LtC classes (https://github.com/tdwg/ltc/wiki/4.-Classes#41-summary-of-latimer-core-classes) section, the LtC standard (version 1) is made up of 23 classes, each with two or more properties. The central concept of the standard is the `ObjectGroup` class, which represents ‘an intentionally grouped set of objects with one or more common characteristics’. Arranged around the

`ObjectGroup` are a set of classes, such as `GeographicContext`, `Taxon` and `GeologicalContext`, which are commonly used to describe and classify the objects within the `ObjectGroup`. There are a set of classes (`OrganisationalUnit`, `CollectionStatusHistory` and `StorageLocation`) intended to reflect aspects of the custodianship, management and tracking of the collections, and a generic class (`MeasurementOrFact`) for storing metrics, narratives and other qualitative or quantitative measures within the standard. A further set of generic, reusable classes are included that enable common concepts (such as people, identifiers and references) to be attached to multiple classes within the standard. Finally, there's a set of classes that are used to describe the structure of and metadata about the LtC records and contents themselves.

These are illustrated in **Figure 6** below, and the concepts are described in more detail later in this section.

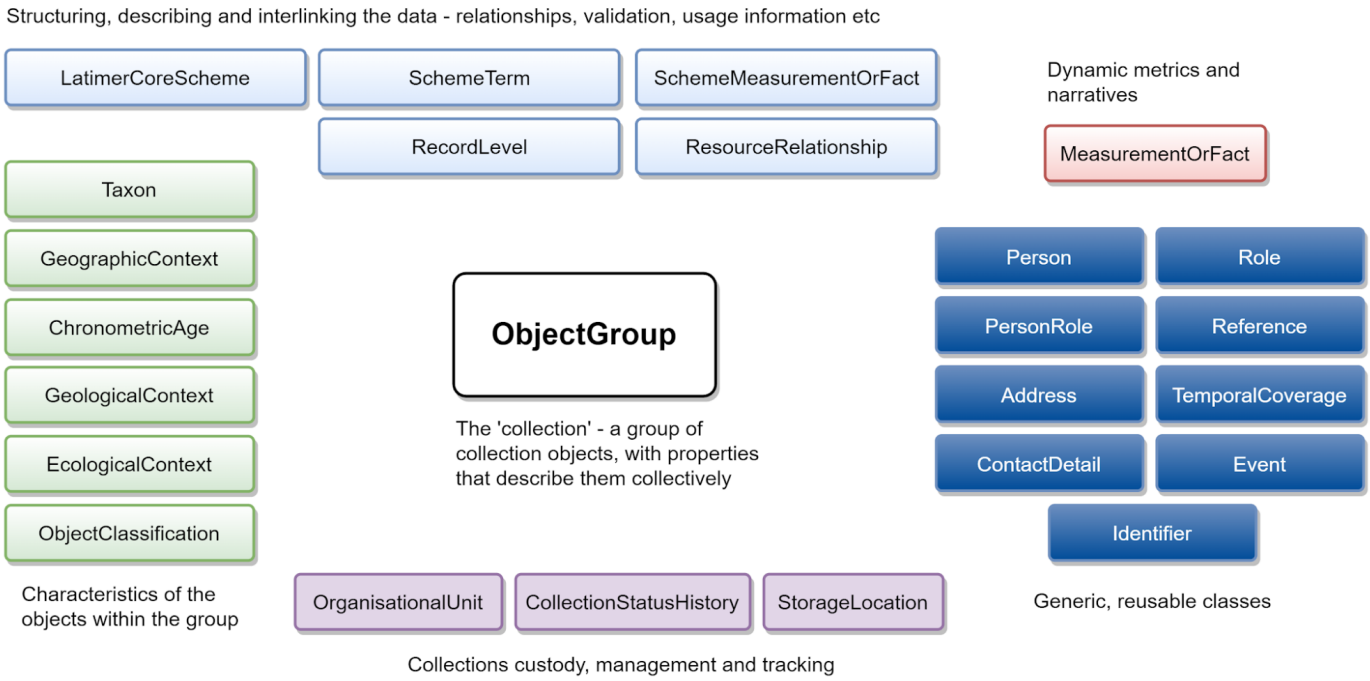


Figure 6: An overview and informal categorisation of LtC classes for describing the ObjectGroup's characteristics (green), collections custody (purple), generic reusable information types (dark blue), metrics (red), and data structure and links (light blue).

The ObjectGroup class

The `ObjectGroup` is the core class and concept of the LtC standard. It represents any set of physical or digital objects that we want to describe together as a group, as opposed to representing and describing each individual separately. This will generally be for one or both of the following reasons:

1. There aren't yet, or may never be, individual digital records for all of the separate objects, but we still want to capture, use and share data about them at a higher level, or
2. There is information that applies to the group as a whole (e.g. a narrative or history, a registration number or a contact person), so we want to store this information at the group level rather than duplicating it across multiple object-level records.

In cases where collections are completely digitized much of the ObjectGroup data could be generated by aggregating individual records, but, there are still likely to be use cases for managing some information at the group level which can't be obtained from the specimen records.

What can an ObjectGroup represent?

The Latimer Core definition of an `ObjectGroup` is

"An intentionally grouped set of objects with one or more common characteristics."

This is a purposefully broad definition that allows the standard to encompass a wide range of use cases for describing collections or their

parts for different reasons. Although the most common and familiar use of collection descriptions has been to describe the collection of a particular institution, conceptually an `ObjectGroup` can represent any set and number of objects that need to be grouped for any purpose. This can scale from a few objects in a drawer to the sum total of all the collections of global natural science institutions: both of these can be represented by an `ObjectGroup`, as can any level of grouping in between.

Some more practical examples of what may be represented by an `ObjectGroup` include:

- The collections of a single institution
- The herbarium collection of a single institution
- A specific collection of a single institution, e.g. a herbarium collection, wet invertebrates collection, or Mesozoic mammal collection.
- A named collection provided by a specific donor, or collected by a significant collector
- The unincorporated objects of a particular department within an institution
- The objects in a single drawer, cabinet or room for inventory purposes
- A set of objects brought into an institution together, e.g. from a field trip, several field trips, or a donation from another institution.
- A database of images taken from collection specimens
- A dataset of butterfly observations taken as a monitoring time series
- An ex situ tree plantation rescuing remaining genotypes of European ash
- A virtual collection representing all penguin individuals and populations in zoos and aquaria worldwide
- Collections of sightings of animals as part of environmental impact/mitigation projects (e.g., Toads on Roads (https://www.froglife.org/what-we-do/toads-on-roads/))

Each of these represent a number of objects that are grouped for a certain purpose, with one or more common characteristics (for example, belonging to the same institution, being from the same stratigraphic time period, or being collected by the same person). Those characteristics are described by the `ObjectGroup's` properties and the other associated classes in LtC, as summarised in **Figure 7**.

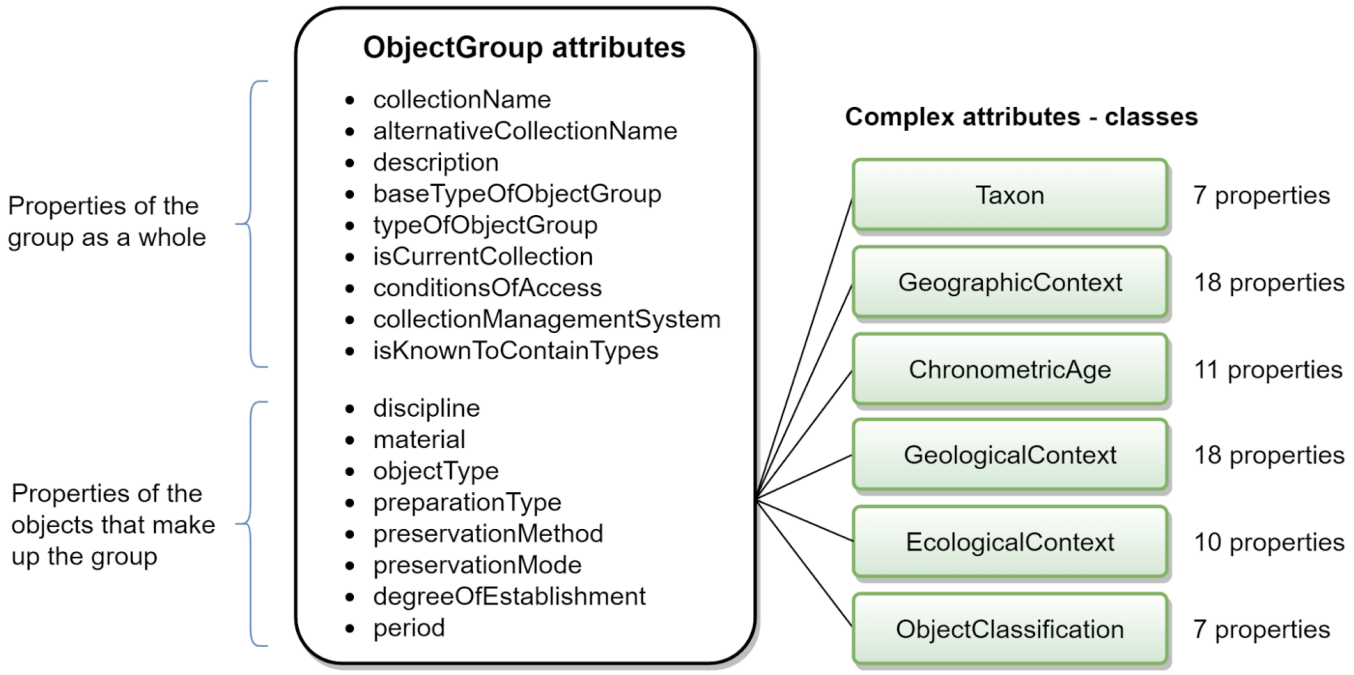


Figure 7: A summary of the `ObjectGroup's` properties and other associated classes that describe the characteristics of the objects within

the group.

The OrganisationalUnit Class

Institutions and organisational units

Collections are held by caregivers. Some collections start out as privately held assemblages, associated generally with a single person. Over the longer term, most surviving collections become preserved and managed by institutions, including for example public or private institutions, organisations, and corporations. Institutions are the administrative entities that are stewards or owners of collections, provide administrative services, maintain and preserve collections, and often employ or dedicate responsible staff to actively manage collections.

For the purpose of integration into a globally applicable collections registry scheme, private collections can be considered to be held by a person of a corresponding role (e.g. “steward”) and be part of a theoretical organisation (e.g. “independent operation”, “private residence”). For a more general discussion of organisations, persons and groups as agents see the FOAF Vocabulary Specification (http://xmlns.com/foaf/spec/) (http://xmlns.com/foaf/spec/)).

An important element to stress is the distinction between the organisational unit that holds a collection, and the collection itself. An institution is represented by the class `OrganisationalUnit` and its properties. A collection within that institution is represented by an `ObjectGroup`. The two can be linked together (**Figure 8**) to represent the custodianship of the `OrganisationalUnit` for the `ObjectGroup`. This means that the properties describing the institution are restricted to the `OrganisationalUnit` and associated classes, while any data describing the collection and the objects within it are represented either by `ObjectGroup` properties, or instances of other LtC classes linked to the `ObjectGroup`.

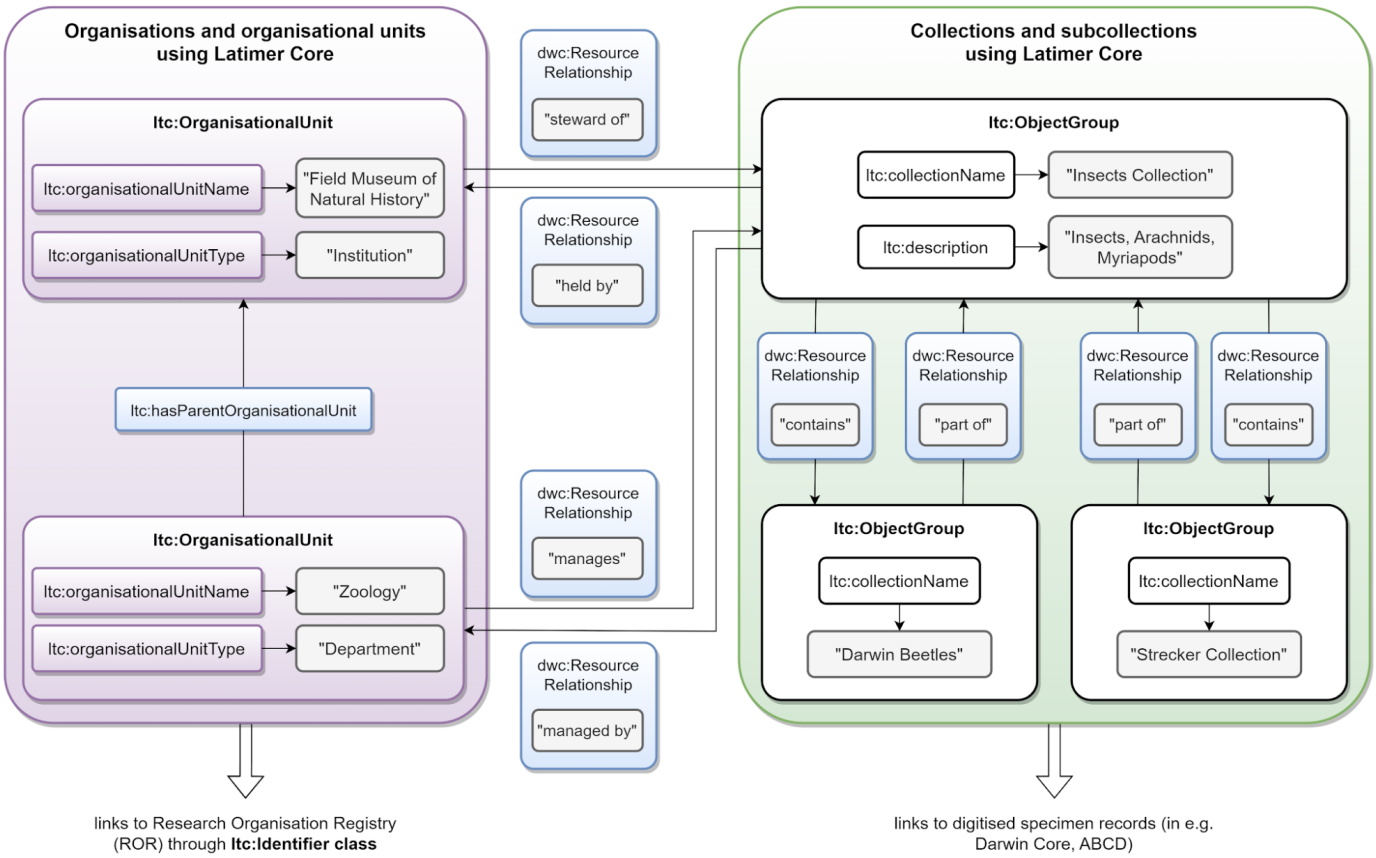


Figure 8: An example of representing and linking organisational units and collections using the LtC standard.

In the LtC standard, the institution concept is inspired by the more generic framework provided by the W3C Organization Ontology (ORG, https://www.w3.org/TR/vocab-org/) (https://www.w3.org/TR/vocab-org/)). Institutions can be represented by the `OrganisationalUnit` class, which is more broadly defined than the corresponding term in the ORG standard. In the LtC standard, its definition combines characteristics of both the `org:Organization` and `org:OrganizationalUnit` classes, providing more flexibility. Plans for the future include further

development of the intersection between ORG and LtC, to improve alignment and thus enable seamless extension and integration of both standards by each other.

The LtC class `OrganisationalUnit` provides a simplified approach to representing institutional information and institutional structure, for example, for a local collection management catalogue maintained by a collection-holding institution or a local research group managing their research collections and groups of specimens on loan from a variety of scientific collections. In these cases, it can be sufficient to provide the `ltc:OrganisationalUnits` directly responsible for the collections or groups of objects. It might not be necessary to represent complete, sometimes byzantine institutional hierarchies.

For larger-scale, e.g. national to regional and global, registry projects, as for example GRSciColl (https://www.gbif.org/grscicoll (https://www.gbif.org/grscicoll)) and GGBN (https://www.ggbn.org/ggbn_portal/) (https://www.ggbn.org/ggbn_portal/)), as well as longer-term LtC-based implementations that are expected to grow and develop, we recommend to take advantage of the full functionality and power of the Organization Ontology standard for representing institutions, their legal status, structure, employees, addresses, etc. (**Figure 9**). The seamless linking to and extension of LtC by the Organization Ontology W3C standard, and the other way around, is made possible by the Darwin Core `dwc:ResourceRelationship` class (see below). Thereby, the integration of collection and institution registries, e.g. the GBIF Registry of Scientific Collections (https://www.gbif.org/grscicoll (https://www.gbif.org/grscicoll)) and the Research Organization Registry (https://ror.org/ (https://ror.org/)) is enabled (see e.g. Politze 2021, https://easychair.org/publications/open/r8Dv) (https://easychair.org/publications/open/r8Dv)). In general, this option provides implementers of LtC to choose whether to use the LtC `OrganisationalUnit` and associated generic classes to represent organisations and related structures, or to instead link in ORG for more complex structures. It should also prove relatively straightforward to start with the former, and transition to the latter at a later point in time.

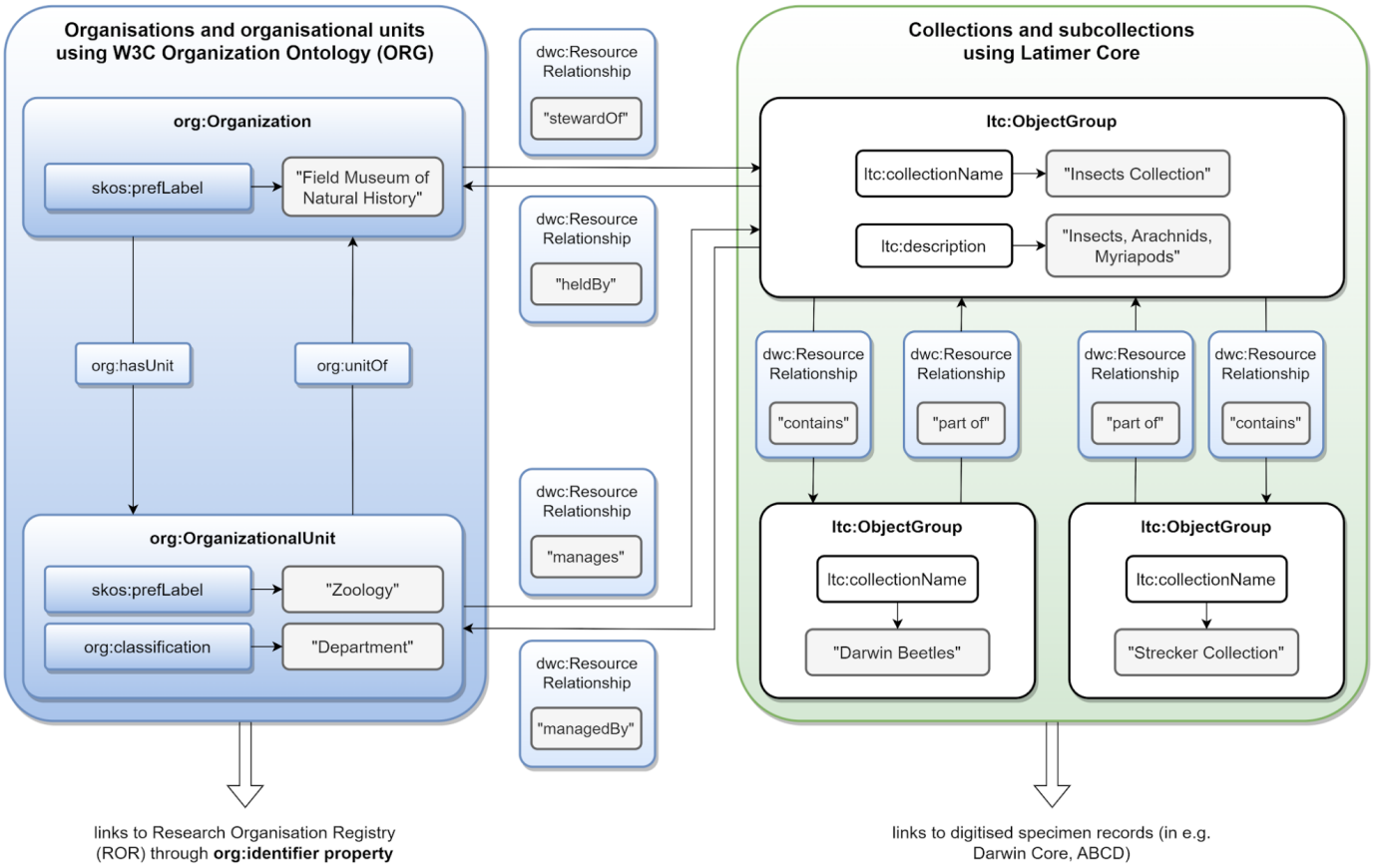


Figure 9: With reference to the example in **Figure 8**, an example of using ORG with LtC to represent organisations and organisational units, instead of the LtC `OrganisationalUnit` class.

The `ltc:OrganisationalUnit` class can represent, in addition to institutions as (legal) entities, subdivisions at different levels within an institution (e.g. departments, divisions, sections, faculties etc), as well as organisations at a higher level of the collection-holding institution (for example, a museum may be a subunit within the organisational structure of a university). Terms to be used in predefined vocabularies identifying the hierarchical position of an organisational unit within an organisational structure can be found in the “Academic Institution

Internal Structure Ontology" (AIISO; https://vocab.org/aiiso/) (https://vocab.org/aiiso/)).

Within LtC, `OrganisationalUnits` can be linked together into a hierarchy or structure using the `ltc:hasParentOrganisationalUnit` class. Similarly, the links between the institutional level (implemented within the LtC or following the ORG standard) and the collection level (LtC standard) are built using the `dwc:ResourceRelationship` class. The `dwc:relationshipOfResource` property of the class should be used to describe the nature of the relationship. It is recommended best practice (see https://dwc.tdwg.org/list/#dwc_ResourceRelationship (https://dwc.tdwg.org/list/#dwc_ResourceRelationship)) to use a controlled vocabulary to describe those relationships. Relationship types listed as examples by the Darwin Core webpage are "same as", "part of", "contains". Alternatively, AIISO terms can be used, e.g. part_of (https://vocab.org/aiiso/#part_of), responsibilityOf (https://vocab.org/aiiso/#responsibilityOf), responsibleFor (https://vocab.org/aiiso/#responsibleFor).

Within the `ObjectGroup` -centric LtC model, associated `OrganisationalUnits` are essentially treated as properties of the `ObjectGroup`, rather than as a higher level in a rigid hierarchy. The notation for the property `organisationalUnitName` of the class `OrganisationalUnit` is correctly `ltc:ObjectGroup.OrganisationalUnit.organisationalUnitName` or in a short form `ltc:OrganisationalUnit.organisationalUnitName`, since in LtC the `ObjectGroup` class is per default the core element.

In a general graph context (**Figure 8** and **Figure 9**), both classes, `ObjectGroup` and `OrganisationalUnit`, can be considered to be at the same level and can be directly associated (they are within each other's range, see the section on Class ranges and relationships (https://github.com/tdwg/ltc/wiki/6.-Structural-Connections#61-class-ranges-and-relationships)). This link can be looked at and approached from both directions, the point of view taken thereby varies between concrete use cases and serialisations. The collection-centric LtC perspective is to approach the association from the side of the ObjectGroup, therefore, in LtC `OrganisationalUnit` is a class-level property of `ObjectGroup`, accordingly in **Table 1** of the Class ranges and relationships (https://github.com/tdwg/ltc/wiki/6.-Structural-Connections#61-class-ranges-and-relationships) section, only `OrganisationalUnit` can be attached to `ObjectGroup` and not the other way around. However, in a context outside of LtC, e.g. with a focus on describing organisations (cp. the W3C org ontology (https://www.w3.org/TR/vocab-org/)), `ObjectGroup` will be a class-level property of `Organization` or `OrganizationalUnit`.

4.3 Generic Classes

Some classes can be used in multiple contexts within the LtC standard. These classes, listed in Table 6, are referred to here as 'generic' classes.

Generic classes represent a type of entity or concept that can be a property of, or associated with, several LtC classes. An `OrganisationalUnit`, `Taxon`, and `Person` all might require an `Identifier` of some sort, for instance. Rather than explicitly repeating the same set of fields within each class, each class includes a `hasIdentifier` term which may contain one or more instances of the generic `Identifier` class.

Generic Class	Range
Address	OrganisationalUnit, Person, PersonRole, StorageLocation
ContactDetail	OrganisationalUnit, Person, PersonRole
Identifier	Address, ChronometricAge, CollectionStatusHistory, ContactDetail, EcologicalContext, Event, GeographicContext, GeologicalContext, LatimerCoreScheme, MeasurementOrFact, ObjectClassification, ObjectGroup, OrganisationalUnit, Person, PersonRole, RecordLevel, Reference, ResourceRelationship, Role, SchemeMeasurementOrFact, SchemeTerm, StorageLocation, Taxon
MeasurementorFact	ChronometricAge, CollectionStatusHistory, EcologicalContext, Event, GeographicContext, GeologicalContext, ObjectClassification, ObjectGroup, OrganisationalUnit, Person, PersonRole, StorageLocation, Taxon, TemporalCoverage
Person	PersonRole
PersonRole	Event, ObjectGroup, OrganisationalUnit, RecordLevel
Reference	ChronometricAge, CollectionStatusHistory, EcologicalContext, Event, GeographicContext, GeologicalContext, Identifier, LatimerCoreScheme, MeasurementOrFact, ObjectClassification, ObjectGroup, OrganisationalUnit, Person, PersonRole, RecordLevel, ResourceRelationship, SchemeMeasurementOrFact, SchemeTerm, StorageLocation, Taxon, TemporalCoverage

ResourceRelationship	ObjectGroup, RecordLevel
Role	PersonRole
TemporalCoverage	CollectionStatusHistory, Event, PersonRole

Table 6: List of Latimer Core generic classes and ranges

Figures 10, 11, and 12 illustrate how instances of the generic `Identifier` class can be used to represent `Person`, institutional (`OrganisationalUnit`) and `Taxon` identifiers.

```
{
    "@context":
    {
        "ltc": "http://rs.tdwg.org/ltc/"
    },
    "@type": "ltc:OrganisationalUnit",
    "ltc:hasIdentifier":
    [
        {
            "@type": "ltc:Identifier",
            "ltc:identifierType": "Acronym",
            "ltc:identifierValue": "EMNH"
        },
        {
            "@type": "ltc:Identifier",
            "ltc:identifierType": "Acronym",
            "ltc:identifierValue": "EM(NH)"
        },
        {
            "@type": "ltc:Identifier",
            "ltc:identifierType": "URI",
            "ltc:identifierSource": "https://ror.org/",
            "ltc:identifierValue": "099zvsn29"
        }
    ],
    "ltc:organisationalUnitName": "Erehwon Museum of Natural History",
    "ltc:organisationalUnitType": "Institution",
    "ltc:organisationalUnitAlternativeName": "Museum of Natural History, Erehwon"
}
```

Figure 10: Using the `Identifier` class to represent an institutional identifier. (JSON file in LtC repo (https://github.com/tdwg/ltc/blob/main/docs/examples/json/identifiers_institution_example.json))

```
{
    "@context":
    {
        "schema": "https://schema.org/",
        "abcd": "http://rs.tdwg.org/abcd/",
        "ltc": "http://rs.tdwg.org/ltc/"
    },
    "@type": "ltc:Person",
    "ltc:hasIdentifier":
    [
        {
            "@type": "ltc:Identifier",
            "ltc:identifierType": "URI",
            "ltc:identifierSource": "https://www.worldcat.org/identities/",
            "ltc:identifierValue": "lccn-n85816396"
        },
        {
            "@type": "ltc:Identifier",
            "ltc:identifierType": "URI",
            "ltc:identifierSource": "https://www.wikidata.org/wiki/",
            "ltc:identifierValue": "Q371568"
        },
        {
            "@type": "ltc:Identifier",
            "ltc:identifierType": "URI",
            "ltc:identifierSource": "https://viaf.org/viaf/",
            "ltc:identifierValue": "38383989"
        }
    ],
    "schema:additionalName": "Eileen Doris",
    "schema:familyName": "Courtenay-Latimer",
    "schema:givenName": "Marjorie",
    "abcd:fullName": "Marjorie Eileen Doris Courtenay-Latimer"
}
```


Figure 11: Using the `Identifier` class to represent a person identifier. (JSON file in Latimer Core repo (https://github.com/tdwg/ltc/tree/main/docs/examples/json/identifiers_person_example.json))

```
{
    "@context":
    {
        "dwc": "http://rs.tdwg.org/dwc/",
        "ltc": "http://rs.tdwg.org/ltc/"
    },
    "@type": "ltc:Taxon",
    "ltc:hasIdentifier":
    [
        {
            "@type": "ltc:Identifier",
            "ltc:identifierType": "URI",
            "ltc:identifierSource": "https://www.checklistbank.org/dataset/9812/taxon/",
            "ltc:identifierValue": "84SNJ"
        },
        {
            "@type": "ltc:Identifier",
            "ltc:identifierType": "URI",
            "ltc:identifierSource": "https://marinespecies.org/authority/metadata.php?lsid=",
            "ltc:identifierValue": "urn:lsid:marinespecies.org:taxname:116116"
        },
        {
            "@type": "ltc:Identifier",
            "ltc:identifierType": "URI",
            "ltc:identifierSource": "https://www.wikidata.org/wiki/",
            "ltc:identifierValue": "Q2391885"
        }
    ],
    "dwc:order": "Priapulimorphida",
    "dwc:family": "Priapulidae",
    "dwc:genus": "Priapulus"
}
```


Figure 12: Using the `Identifier` class to represent a taxon identifier. (JSON file in the Latimer Core repo (https://github.com/tdwg/ltc/tree/main/docs/examples/json/identifiers_taxon_example.json))

The overall purpose of any given instance of a generic class can be inferred from its context in the record: a `ContactDetail` that is associated with or nested within an `OrganisationalUnit` is an organisational contact point, but a `ContactDetail` associated or nested within a `Person` should be interpreted as that `Person's` preferred method of contact (illustrated in **Figure 13**).

```
{
  "@context": {
    "ltc": "http://rs.tdwg.org/ltc/",
    "schema": "https://schema.org/",
    "abcd": "http://rs.tdwg.org/abcd/"
  },
  "@type": "ltc:OrganisationalUnit",
  "ltc:hasContactDetail": [
    {
      "@type": "ltc:ContactDetail",
      "contactDetailCategory": "email",
      "contactDetailType": "General enquiries",
      "contactDetailValue": "enquiries@emnh.org"
    }
  ],
  "ltc:hasPersonRole": [
    {
      "@type": "PersonRole",
      "role": "Registrar",
      "ltc:hasPerson": [
        {
          "@type": "ltc:Person",
          "schema:additionalName": "Virginia",
          "schema:familyName": "Mountweazel",
          "schema:givenName": "Lillian",
          "abcd:fullName": "Lillian Virginia Mountweazel",
          "ltc:hasContactDetail": [
            {
              "@type": "ltc:ContactDetail",
              "contactDetailCategory": "email",
              "contactDetailType": "Collection enquiries",
              "contactDetailValue": "v.mountweazel@emnh.org"
            }
          ]
        }
      ]
    }
  ],
  "ltc:organisationalUnitAlternativeName": "Museum of Natural History, Erehwon",
  "ltc:organisationalUnitName": "Erehwon Museum of Natural History",
  "ltc:organisationalUnitType": "Institution"
}
```


Figure 13: Using the `ContactDetail` class to represent an institutional email address and a named individual's email address. (JSON file in the Latimer Core repo (https://github.com/tdwg/ltc/tree/main/docs/examples/json/contact_detail_example.json))

There are several benefits to taking this approach, using the `Identifier` class as an example:

1. Flexibility: a generic `Identifier` class can be used to represent any identifier relevant to classes within its range, so the standard needn't attempt to anticipate all potential current or future use cases.
2. Extensibility: The concept can have additional properties that describe it further: e.g. `Identifier` is further described by `identifierSource`, `identifierType` etc.
3. It reduces the number of properties in the standard by reusing generic classes.
4. It supports the association of multiple identifiers (as shown in Figures 4, 5 and 6) with an entity: see the Repeatability (https://github.com/tdwg/ltc/wiki/6.-Structural-Connections#62-repeatability) section.

4.4 People and roles

To handle people and roles, LtC uses a simplified version of the RDA/TDWG Attribution metadata recommendations (https://

datascience.codata.org/articles/10.5334/dsj-2019-054/) (https://datascience.codata.org/articles/10.5334/dsj-2019-054/)), which is largely based on the PROV ontology for presenting provenance (https://www.w3.org/TR/prov-o/) (https://www.w3.org/TR/prov-o/)). Using this approach avoids both the hardcoding of all potential roles people might play in relation to the different classes in the standard, and the proliferation of similar properties that would be required to support them.

To this end, there are three classes within the standard that are used to represent people and roles: `Person`, `PersonRole` and `Role`. The `Person` class is used to describe the relevant person, and the `PersonRole` to attach that person to another class and, with an associated `Role` class, to define the role that they played in the context of that class (and, optionally, the time frame during which they fulfilled that role). `PersonRole` is a generic class (see previous section) which can be attached to a range of other classes within the standard, as demonstrated in **Figures 14, 15 and 16** below. The separation of `Role` into a separate class is largely to help to support the use of shared role vocabularies such as the Contributor Roles Taxonomy (CRediT) (https://credit.niso.org/) with machine-readable persistent identifiers, and to potentially attribute more than one of these roles to a single `PersonRole` association between a `Person` and an `OrganisationalUnit`, `Event`, `ObjectGroup` etc.

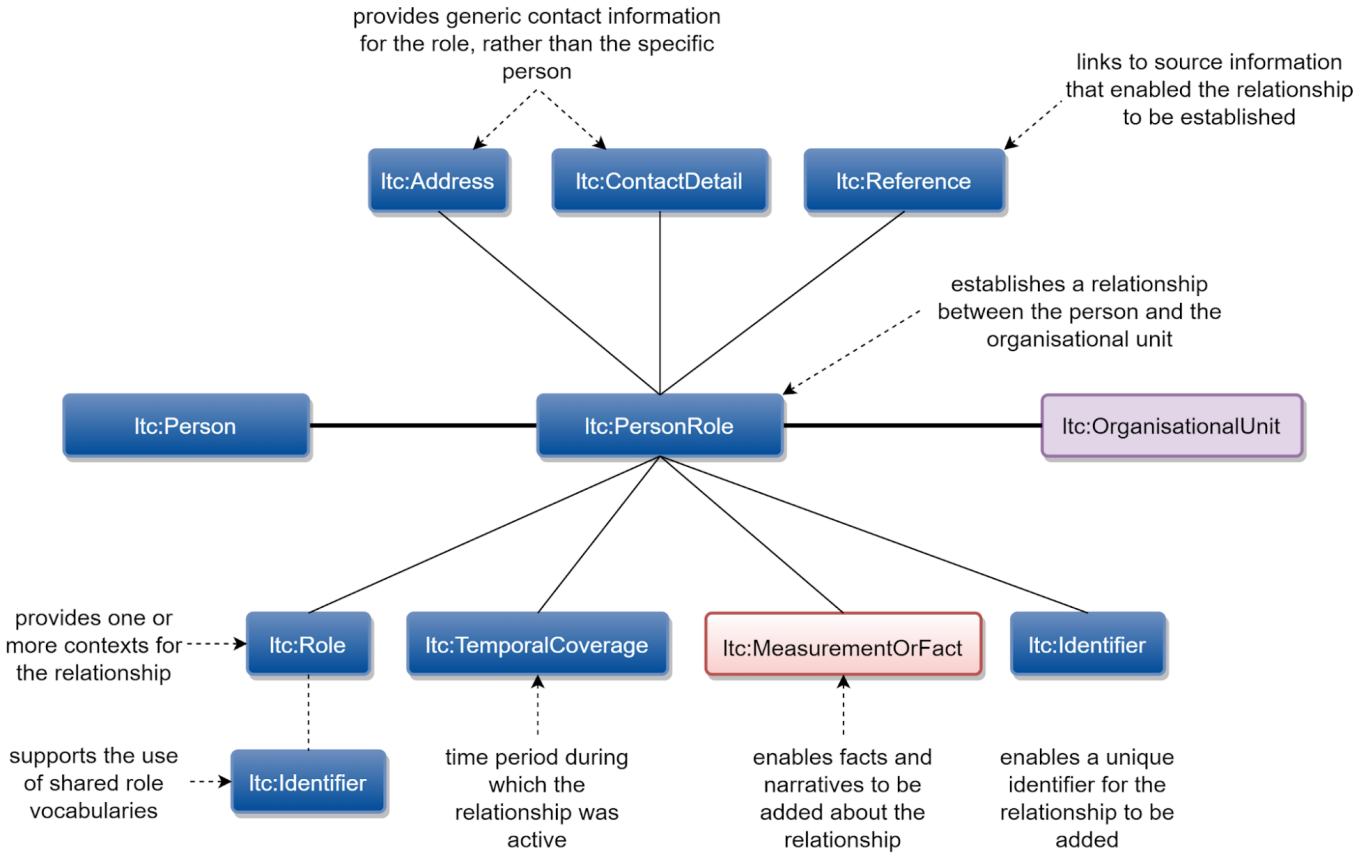


Figure 14: Summary of the classes that can be used to define the relationships between people and (in this example) organisational units.

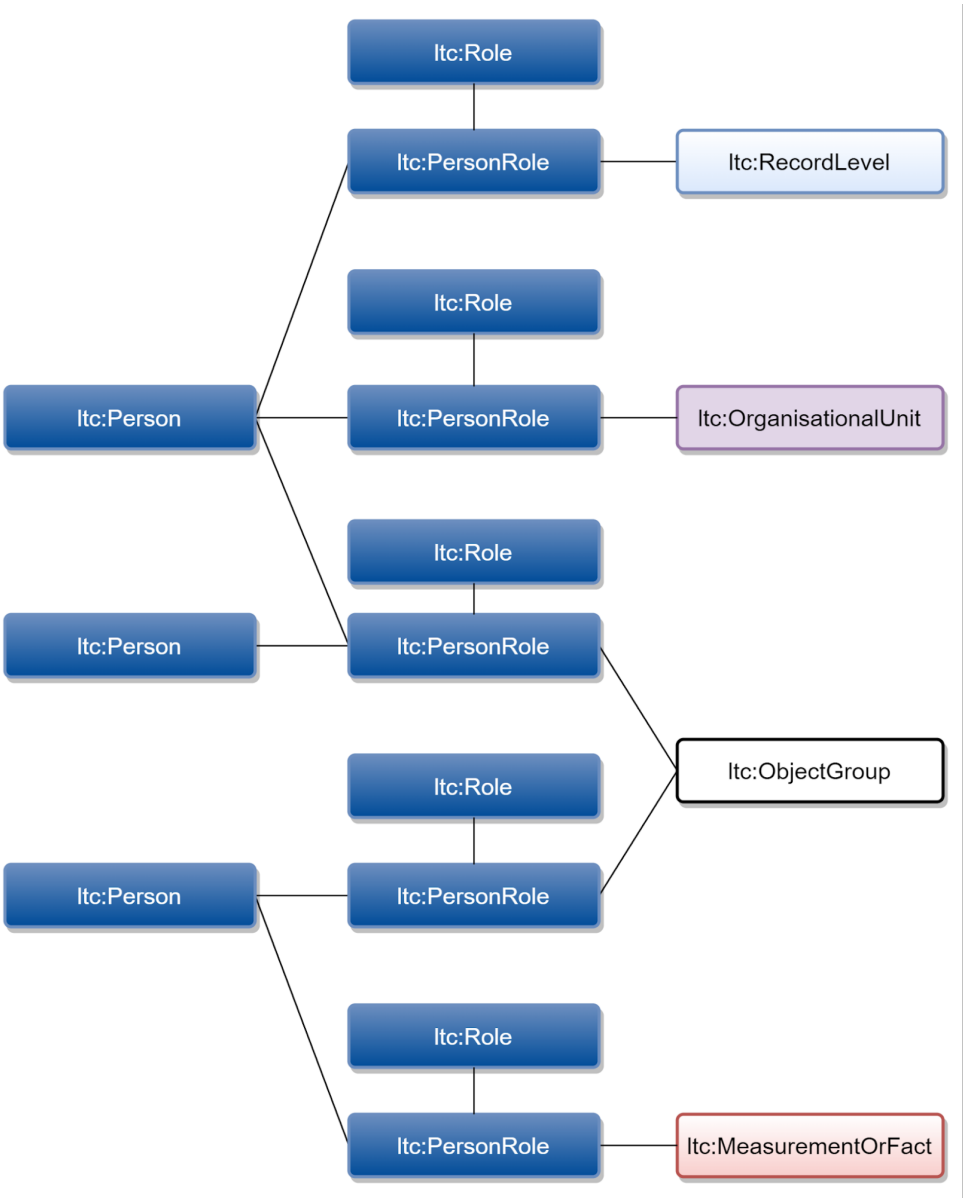


Figure 15: Linking people to other classes using the `PersonRole` and `Role` classes.

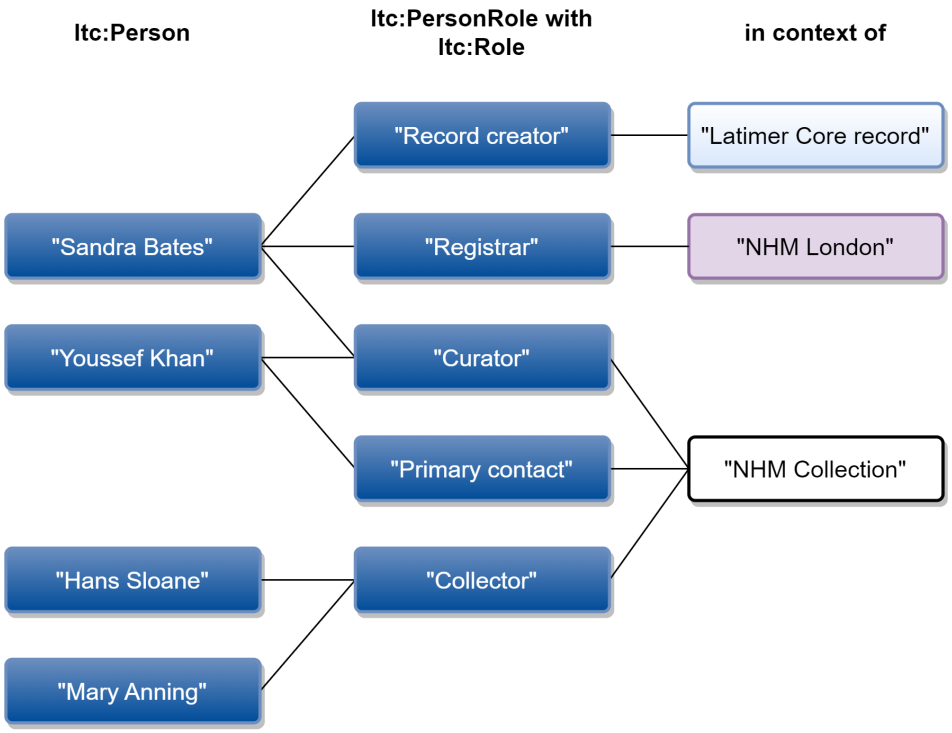


Figure 16: An example of using the `Person` and `PersonRole` classes to link people to collections, organisations and digital records in the relevant contexts.

The LtC standard implementation differs from the RDA/TDWG recommendations in a couple of key respects: - Rather than use the PROV `Agent` entity, which has subclasses of `Person`, `Organization` and `SoftwareAgent`, LtC currently only uses the 'Person' subtype within this model, and OrganisationalUnit is represented by a separate class. - The PROV model connects the `Agent` to an `Entity` using an `Activity` class, with the role represented by a qualified association between the `Agent` and the `Activity`. In the current version of LtC, the activity concept has been removed from the model to simplify the association. These modifications are largely to reduce the level of abstraction in the initial version of LtC, and make the classes more familiar and model simpler for users of the standard. There may be potential to use the full `Agent` scope and introduce the `Activity` concept in future iterations of LtC to more fully align with PROV and possible developments within other TDWG standards.

5.-Metrics-and-narratives.md

Two components frequently included in collection descriptions are quantitative metrics (e.g. the precise or estimated number of objects or taxa) and richer narratives about the collection and its history. To represent these, LtC has adopted the `MeasurementOrFact` class used in Darwin Core and ABCD, and added an `ltc:measurementFactText` property. This property is for holding the richer text narratives in collection descriptions and, if required, categorising them using the `measurementType` property. This might be used, for example, to define and store a historical narrative and brain-dump of notes from a retiring curator about a collection separately. It also separates the text descriptions from quantitative measures stored in the `measurementValue` property, which is intended to make validation and programmatic aggregation and computation on the `measurementValue` property more straightforward.

```
"ObjectGroup": {

    "collectionName": "Collection of the Natural History Museum, London",

    "hasMeasurementOrFact": [
    {
        "measurementType": "Object count",
        "measurementValue": "72190070"
    },
    {
        "measurementType": "Collection history",
        "measurementMethod": "Text narrative",
        "measurementRemarks": "Text derived from Wikipedia",
        "measurementFactText": "The foundation of the collection was that of the Ulster doctor Sir Hans Sloane (1660–175
    }
    ]
}
```

Figure 17: JSON example of a quantitative metric and textual narrative using the `MeasurementOrFact` class.

Similarly to the other generic classes described earlier, the `MeasurementOrFact` class can be associated with many of the other classes in the standard to add defined, dynamic properties, and is also repeatable.

It's also possible to use the `MeasurementOrFact` class to qualify or quantify relationships between other classes in the standard, in conjunction with the `ResourceRelationship` class. This is covered in more detail in the later section on modelling approaches (https://github.com/tdwg/ltc/wiki/7.-Relationships#72-modelling-approaches).

5.1 Metric examples

There are no prescribed metrics included in the standard. It is hoped that over time data requestors/aggregators will define the metrics that they require using the standard and its `MeasurementOrFact` class. Data providers can then use these as recipes to follow to generate the information needed to facilitate easier and more automated sharing and comparison. The intent being that over time libraries of standard metrics can be published and shared. For example, BioSchemas provides a library of usage profiles: https://bioschemas.org/profiles (https://bioschemas.org/profiles/).

Metric Description	measurementType
object count	The total number of specimens and/or items. If the metric is attached to an object group then the count is of the collection being described. If the metric is attached to the Institution then it is the overall count for that institution.
digitisation level percentage	The percentage of the whole collection being described, that is "digitised". Explicitly state what is meant by digitised - imaged and/or a database record (metadata) exists etc. Could be used in combination with the object count metric.
digitisation level count	An actual number of digitised records in the collection being described [define digitised]
imaged level percentage	The percentage of the collection described in the record that has an image. Could be used in combination with the object count metric.
imaged level count	An actual number of digitised records with images
georeferenced level percentage	The percentage of the collection described in the record that has verified Lat Lon coordinates. Could be used in combination with the "object count" metric.
georeferenced level count	The actual number of records that have verified Lat Lon coordinates

taxonomic rank	Count of the taxa at the rank indicated in measurementFactText. Do not abbreviate the rank.
storage volume	The cubic volume of the record (Institutional or Collection). Could be used in combination with the object count metric.
storage footprint	The area of the record (Institutional or Collection). Could be used in combination with the object count metric.

Table 7: Example Latimer Core simple metrics in tabular format (CSV file in the Latimer Core repo (https://github.com/tdwg/ltc/tree/main/source/examples/flat/metric_object_count_example.csv))

Metric Description	measurementType
https://github.com/tdwg/mids/blob/working-draft/current-draft/MIDS-definition-v0.15-29Jul2021.md#44-information-elements-expected-at-mids-level-0 (https://github.com/tdwg/mids/blob/working-draft/current-draft/MIDS-definition-v0.15-29Jul2021.md#44-information-elements-expected-at-mids-level-0)	MIDS-0 object count
https://github.com/tdwg/mids/blob/working-draft/current-draft/MIDS-definition-v0.15-29Jul2021.md#43-information-elements-expected-at-mids-level-1 (https://github.com/tdwg/mids/blob/working-draft/current-draft/MIDS-definition-v0.15-29Jul2021.md#43-information-elements-expected-at-mids-level-1)	MIDS-1 object count
https://github.com/tdwg/mids/blob/working-draft/current-draft/MIDS-definition-v0.15-29Jul2021.md#43-information-elements-expected-at-mids-level-2 (https://github.com/tdwg/mids/blob/working-draft/current-draft/MIDS-definition-v0.15-29Jul2021.md#43-information-elements-expected-at-mids-level-2)	MIDS-2 object count
https://github.com/tdwg/mids/blob/working-draft/current-draft/MIDS-definition-v0.15-29Jul2021.md#43-information-elements-expected-at-mids-level-3 (https://github.com/tdwg/mids/blob/working-draft/current-draft/MIDS-definition-v0.15-29Jul2021.md#43-information-elements-expected-at-mids-level-3)	MIDS-3 object count

Table 8: Example MIDS metrics in tabular format (CSV file in the Latimer Core repo (https://github.com/tdwg/ltc/tree/main/source/examples/flat/metric_digitisation_level_percentage_example.csv))

6.-Structural-Connections.md

6.1 Class Ranges and Relationships

The range of a LtC class defines the classes to which it can validly be attached. This is intended to provide guidelines for reasonable interconnections between classes that are useful when describing a given aspect of a collection. For example, the `TemporalCoverage` class can be validly attached to the `CollectionStatusHistory`, `Event`, or `PersonRole` classes in order to document the time spans for those classes.

Class Name	Range (i.e., other classes to which the class in the first column can be attached)
Address	OrganisationalUnit, Person, PersonRole, StorageLocation
ChronometricAge	ObjectGroup
CollectionStatusHistory	ObjectGroup
ContactDetail	OrganisationalUnit, Person, PersonRole
EcologicalContext	Event, ObjectGroup
Event	Event, ObjectGroup
GeographicContext	Event, ObjectGroup
GeologicalContext	ObjectGroup

Identifier	Address, ChronometricAge, CollectionStatusHistory, ContactDetail, EcologicalContext, Event, GeographicContext, GeologicalContext, LatimerCoreScheme, MeasurementOrFact, ObjectClassification, ObjectGroup, OrganisationalUnit, Person, PersonRole, RecordLevel, Reference, ResourceRelationship, Role, SchemeMeasurementOrFact, SchemeTerm, StorageLocation, Taxon
LatimerCoreScheme	
MeasurementOrFact	ChronometricAge, CollectionStatusHistory, EcologicalContext, Event, GeographicContext, GeologicalContext, ObjectClassification, ObjectGroup, OrganisationalUnit, Person, PersonRole, StorageLocation, Taxon, TemporalCoverage
ObjectClassification	ObjectClassification, ObjectGroup
ObjectGroup	LatimerCoreScheme, RecordLevel
OrganisationalUnit	ObjectGroup, OrganisationalUnit
Person	PersonRole
PersonRole	Event, ObjectGroup, OrganisationalUnit, RecordLevel
RecordLevel	
Reference	ChronometricAge, CollectionStatusHistory, EcologicalContext, Event, GeographicContext, GeologicalContext, Identifier, LatimerCoreScheme, MeasurementOrFact, ObjectClassification, ObjectGroup, OrganisationalUnit, Person, PersonRole, RecordLevel, ResourceRelationship, SchemeMeasurementOrFact, SchemeTerm, StorageLocation, Taxon, TemporalCoverage
ResourceRelationship	ObjectGroup, RecordLevel
Role	PersonRole
SchemeMeasurementOrFact	LatimerCoreScheme
SchemeTerm	LatimerCoreScheme
StorageLocation	ObjectGroup, StorageLocation
Taxon	ObjectGroup
TemporalCoverage	CollectionStatusHistory, Event, PersonRole

Table 9: Ranges for each Latimer Core class

Conversely, the class-level properties define which other classes can be attached to a given LtC class:

Class Name	Class-level Properties
Address	Identifier
ChronometricAge	Identifier, MeasurementOrFact, Reference
CollectionStatusHistory	Identifier, MeasurementOrFact, Reference, TemporalCoverage
ContactDetail	Identifier
EcologicalContext	Identifier, MeasurementOrFact, Reference
Event	EcologicalContext, GeographicContext, Identifier, MeasurementOrFact, Event, PersonRole, Reference, TemporalCoverage
GeographicContext	Identifier, MeasurementOrFact, Reference

GeologicalContext	Identifier, MeasurementOrFact, Reference
Identifier	Reference
LatimerCoreScheme	Identifier, ObjectGroup, Reference, SchemeMeasurementOrFact, SchemeTerm
MeasurementOrFact	Identifier, Reference
ObjectClassification	Identifier, MeasurementOrFact, ObjectClassification, Reference
ObjectGroup	ChronometricAge, CollectionStatusHistory, EcologicalContext, Event, GeographicContext, GeologicalContext, Identifier, MeasurementOrFact, ObjectClassification, OrganisationalUnit, PersonRole, Reference, ResourceRelationship, StorageLocation, Taxon
OrganisationalUnit	Address, ContactDetail, Identifier, MeasurementOrFact, OrganisationalUnit, PersonRole, Reference
Person	Address, ContactDetail, Identifier, MeasurementOrFact, Reference
PersonRole	Address, ContactDetail, Identifier, MeasurementOrFact, Person, Reference, Role, TemporalCoverage
RecordLevel	Identifier, ObjectGroup, PersonRole, Reference, ResourceRelationship
Reference	Identifier
ResourceRelationship	Identifier, Reference
Role	Identifier
SchemeMeasurementOrFact	Identifier, Reference
SchemeTerm	Identifier, Reference
StorageLocation	Address, Identifier, MeasurementOrFact, StorageLocation, Reference
Taxon	Identifier, MeasurementOrFact, Reference
TemporalCoverage	MeasurementOrFact, Reference

Table 10: Class-level properties for each Latimer Core class

6.2 Repeatability

The design decisions for the structure of LtC give rise to the general principle that, with one exception, only classes within the standard can be repeated, not individual properties within classes. For example, if a work phone contact point has two phone numbers, then the ltc:ContactDetail class is repeated for this contact point, resulting in two full sets of phone contact details. Repeating only the ltc:ContactDetail.contactDetailValue property, or changing its stored information from a single string to an array containing several strings, is not allowed. Such an approach would run counter to the spirit of the standard's design, and break its overall structure. In consequence, the contact point, e.g. a ltc:ObjectGroup or ltc:PersonRole that has the two phone contact options will have a "hasContactDetail" term, which represents and collects links to, i.e. stands in for the ltc:ContactDetail class, with a value that is an array storing the two sets of ltc:ContactDetail information as sub-arrays.

One important exception however, exists in the form of the standard's core ltc:ObjectGroup class that rests at the center of the standard's data model. In this class, all properties have as values arrays with zero to many entries. Thus, for an ltc:ObjectGroup that digitally represents a fish collection with objects preserved in alcohol or formaldehyde, ltc:ObjectGroup.preservationMethod validly can be represented as an array of [alcohol, formaldehyde].

Over time different standards have handled repeatability by pipe-delimiting multiple values in a single field, versus nested or related tables or classes. LtC aims to take a pragmatic approach, given the variety of possible relationships among classes.

Repeatable Properties

How repeatability is handled in an LtC implementation depends largely on which data serialisation format is used. Taking the `period`

property of the `ObjectGroup` class as an example, the appropriate methods might be:

- **an array**, if using JSON or RDF ** `"ltc:period": ["Neolithic Period", "McCarthy Era"]`
- **a normalised table**, if using a relational database

• auto_id	• ltc:period
• 1	• Neolithic Period
• 2	• McCarthy Era

- **a repeated set of numbered columns for the same property**, if using a spreadsheet or csv file (e.g. period.1, period.2)

• ltc:period.1	• ltc:period.2
• Neolithic Period	• McCarthy Era

- **multiple pipe-delimited values**, if using a flat text format

• ltc:period
• Neolithic Period | McCarthy Era

Repeatable Classes

How can classes be repeated?

As it stands, repeatable classes are repeatable in any context, i.e., regardless of the class to which they are attached. Some classes may seem to warrant different repeatability allowances in different situations. We leave the choice to the user.

Taking the `Identifier` class as an example, the appropriate methods for representing multiple identifiers for an `ObjectGroup` might be:

- **an array of objects**, if using JSON or RDF

```
"Identifier": [
{
        "ltc:identifierSource": "GrSciColl",
        "ltc:identifierValue": "urn:uuid:ea4f0640-ef20-40aa-b359-166f07c7492a",
        "ltc:identifierType": "UUID"
    },
    {
        "ltc:identifierSource": "GrBio",
        "ltc:identifierValue": "http://grbio.org/cool/zdsi-36ka",
        "ltc:identifierType": "LSID"
    }
]
```

- **a normalised table**, if using a relational database

auto_id	ltc:identifierSource	ltc:identifierValue	ltc:identifierType
1	GrSciColl	urn:uuid:ea4f0640-ef20-40aa-b359-166f07c7492a	UUID
2	GrBio	http://grbio.org/cool/zdsi-36ka (http://grbio.org/cool/zdsi-36ka)	LSID

- **a repeated set of numbered columns for the same property**, if using a spreadsheet or csv file

ltc:identifierSource.1	ltc:identifierValue.1	ltc:identifierType.1	ltc:identifierSource.2	ltc:identifierValue.2	ltc:identifierType.2

GrSciColl	urn:uuid:ea4f0640-ef20-40aa-b359-166f07c7492a	UUID	GrBio	http://grbio.org/cool/zdsi-36ka (http://grbio.org/cool/zdsi-36ka)	LSID

- **multiple pipe-delimited values**, if using a flat text format
    - In LtC, it is not recommended to repeat properties within repeatable classes. For example, if an `identifierSource` corresponded to two ltc:identifierValue values, repeat each identifier's source, value, and type in parallel:

ltc:identifierSource	ltc:identifierValue	ltc:identifierType
GrSciColl | GrBio	urn:uuid:ea4f0640-ef20-40aa-b359-166f07c7492a | http://grbio.org/cool/zdsi-36ka (http://grbio.org/cool/zdsi-36ka)	UUID | LSID

* Do not nest comma-delimited values within pipe-delimited values:

ltc:identifierSource	ltc:identifierValue	ltc:identifierType
Ouch | Yikes	urn:uuid:ea4f0640-ef20-40aa-b359-166f07c7492a | http://grbio.org/cool/zdsi-36ka (http://grbio.org/cool/zdsi-36ka), http://grbio.org/cool/zdsi-36kb (http://grbio.org/cool/zdsi-36kb)	UUID | LSID, LSID

7.-Relationships.md

7.1 Linking ObjectGroups

There are several methods by which associations between different `ObjectGroups`, representing collections and subcollections, may be constructed.

Direct linkage using the ResourceRelationship class

The `ResourceRelationship` class can be used to make direct links between `ObjectGroups`, using the `relationshipOfResource` property to specify the nature of the relationship. For example, a hierarchy of parent-child relationships can be constructed by using `ResourceRelationships` with a `relationshipOfResource` of 'part of', or more semantic relationships can be reflected, such as between an `ObjectGroup` of physical collection objects, and an `ObjectGroup` representing digital multimedia artefacts derived from those objects.

With the application of the `relationshipEstablishedDate` property, this method of linkage may also be used to represent changes to physical collections over time (for example, collections that are split up or merged together), or track changes to how the data are organised in LtC (for example, a year after the initial creation of an institution's collection descriptions dataset, it's decided to split the `ObjectGroup` representing the whole zoological collection into two `ObjectGroups` representing the vertebrates and invertebrates separately, in order to store more specific descriptions and metrics against them. Creating this kind of provenance relationship enables some continuity of reporting across the two versions of the dataset, even though the shape of the data has changed.

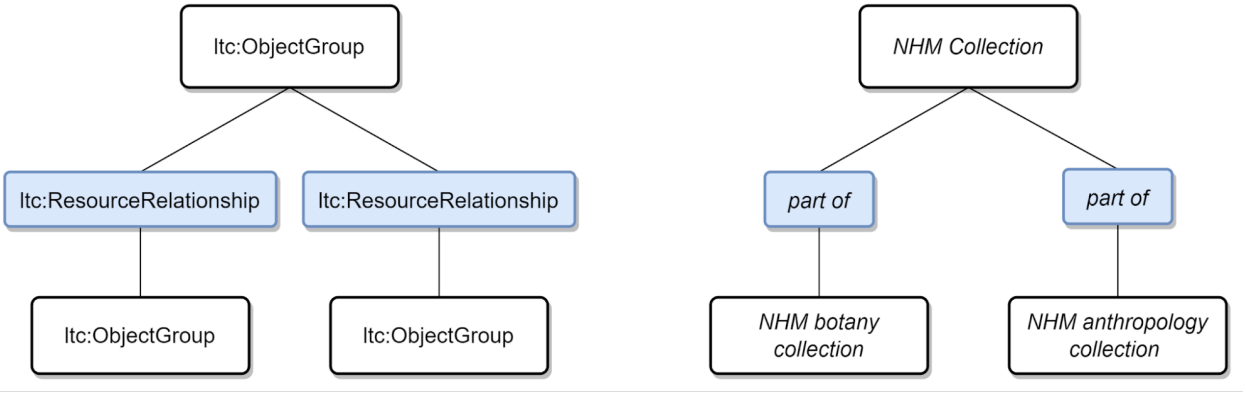


Figure 18: An example of using the `ResourceRelationship` class to represent `ObjectGroups` that are part of a larger `ObjectGroup` in a collections hierarchy.

By a similar method, hierarchical structures may also be constructed in other relevant classes to represent breakdowns of collections in different contexts. For example, `OrganisationalUnit` instances may be chained together with `part of` relationships to represent the divisions and subdivisions within an institution. `Taxon` instances may be linked to form a taxonomic hierarchy, or `ObjectClassification` instances linked to create a less formal hierarchy of object types. Using these linked classes and attaching them to `ObjectGroups` enables collections to be arranged in contextual hierarchies instead of (or in addition to) creating explicit relationships between the `ObjectGroups` themselves. **Figure 19** below shows a simple example of creating parallel hierarchies in `ObjectGroups` and `OrganisationalUnits` in a LtC dataset.

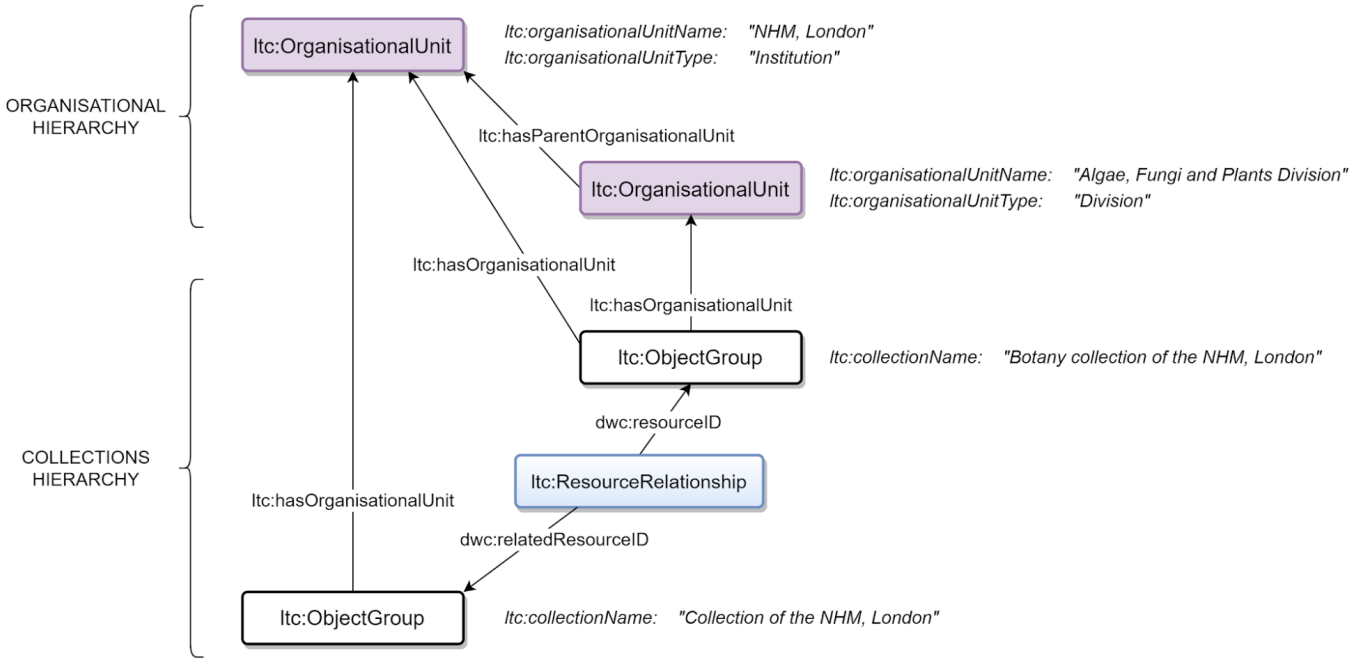


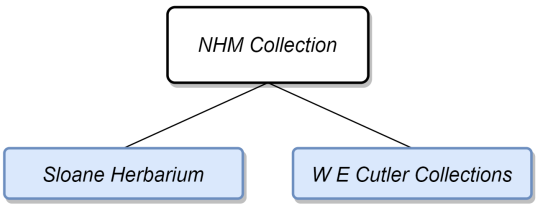
Figure 19: Example of parallel hierarchies in the `OrganisationalUnit` and `ObjectGroup` classes, with some additional semantic relationships between instances of the two classes.

Simple representation of subcollections

A frequent use case of collection descriptions is to have the ability to store a list of notable, historic and important collections that may be part of a larger institutional collection. This scenario can be modelled in LtC using the approach above, by creating an `ObjectGroup` for the parent collection and an `ObjectGroup` for each of the child collections, and linking the children to the parent through the `ResourceRelationship` class.

While this approach provides the ability to attach rich LtC descriptions (including metrics and narratives) to each of these ‘named’ collections, there is also a degree of effort involved in assembling and maintaining this more complex dataset which may not be commensurate with the

time and resources available. A more lightweight approach may also be taken in LtC, where use of the `ResourceRelationship's relationshipResourceName` and `relationshipOfResource` properties can be used to generate a simple list of subcollections and attach them to the parent `ObjectGroup` (see **Figure 20** below).




```
"ResourceRelationship": [
    {
        "relatedResourceName": "Sloane Herbarium",
        "relationshipOfResource": "contains"
    },{
        "relatedResourceName": "W E Cutler Collections",
        "relationshipOfResource": "contains"
    }
]
```

Figure 20: Diagram and JSON example of using the `ResourceRelationship` class for simple representation of subcollections.

This approach also provides a methodology for starting with a fairly simple LtC dataset, and building up the detail over time as and when opportunity and resourcing permits. For example, a dataset might begin with a single ObjectGroup representing the whole institutional collection, and a set of `ResourceRelationship` records listing the divisional collections within that collection. At later points, those subcollections could then be broken out into their own `ObjectGroups` in order to describe and quantify them in more detail.

Using common entities and controlled vocabularies

`ObjectGroups` can also be indirectly related by the use of common entities and controlled vocabularies to allow associations to be made between them (although this method is more general data good practice, rather than anything specific to the LtC standard). For example, if a controlled vocabulary is used for the `discipline` property of the `ObjectGroup` class, then queries such as 'find all the `ObjectGroups` representing 'Botany' objects' and 'provide the total number of 'Botany' objects held by the institution' are easy to execute without the need to create explicit relationships between the multiple `ObjectGroups` involved.

Using a LatimerCoreScheme

The `LatimerCoreScheme` class provides a construct for grouping together `ObjectGroups` as part of the same LtC implementation, for a particular use case, and applying rules about how the data may be constrained, validated and interpreted by software agents. This approach is described in more detail in the Latimer Core Schemes section.

7.2 Modelling approaches

ObjectGroups and relationships

In many ways, the terms that are used to characterise the collection objects within an `ObjectGroup` are similar or identical to those that are used to describe an individual collection object. However, there is a fundamental difference in the relationship with those properties between the two examples. For an `ObjectGroup`, representing multiple objects, there is always the potential for there to be more than one value for any of those terms.

For example, a single object will only have one preservation method (e.g. 'dried and pinned'), whereas an `ObjectGroup` can represent objects with a variety of preservation methods. The former represents a one-to-many relationship between the term and the object (an object may not have more than one preservation method, but one preservation method may relate to many objects). The latter has a many-to-many relationship between the term and the `ObjectGroup` (an `ObjectGroup` may reflect more than one preservation method, and one preservation method may relate to many ObjectGroups).

Why do relationships matter?

The main impact is on the metrics (represented by the `MeasurementOrFact` class) that can be attached to an `ObjectGroup`, and how they can be used. If an `ObjectGroup` has more than one value for the same term, then it's not possible to tell how metrics attached to the `ObjectGroup` are distributed across those values.

For example, if an `ObjectGroup` has a single preservation method of 'dried and pinned', and an 'object quantity' (represented by a `MeasurementOrFact` record) of 10,000, we know that there are 10,000 dried and pinned objects. If however, that `ObjectGroup` has two preservation methods, 'dried and pinned' and 'alcohol', we know that there are 10,000 objects that are either dried and pinned OR preserved in alcohol, but we cannot calculate how many there are of each.

The only way to get an accurate assessment of the overall object quantity AND the object quantity for each preservation method is to split the `ObjectGroup` into two ObjectGroups: one containing just the 'dried and pinned' objects, and one for the 'alcohol' objects. This means that the preservation method maintains a one-to-many relationship with the ObjectGroups, rather than a many-to-many relationship. This is the key to being able to accurately aggregate and report metrics against the preservation method property, as well as the ObjectGroups.

'Dimensions' and 'associations'

We've established that: *Terms can either have a one-to-many or a many-to-many relationship with the* `ObjectGroup` One-to-many relationships between the `ObjectGroup` and a term are required to be able to report accurate metrics against that property

Within the LtC model concept, a term where a one-to-many relationship with the `ObjectGroup` has been enforced is referred to as a dimension. These are the terms that are effectively used to determine how a collection needs to be broken down into multiple ObjectGroups, in order to satisfy the requirements for numeric reporting. It's important to note that the term or terms to be designated as dimensions will vary between implementations, depending on the use case and requirements, and so are not prescribed as part of the LtC model. They can, however, be defined for an implementation using the `LatimerCoreScheme` structure, as described earlier in the document.

Terms that can have a many-to-many relationship with the `ObjectGroup` are referred to, for the purposes of this guidance documentation, as associations. They provide information about what is in the `ObjectGroup` , but cannot generally be quantified using the metrics attached to the `ObjectGroup` (but see the metric options discussed in section X).

When should dimensions and associations be used?

There are both benefits and limitations to applying terms as either dimensions or associations.

Associations

Associations essentially allow you to use the properties like `ObjectGroup` tags, attaching a number of values for the same property to a single `ObjectGroup` (**Figure 21**).

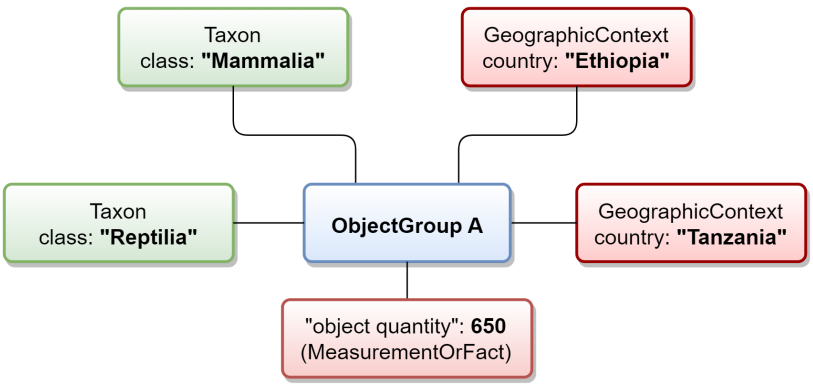


Figure 21: An `ObjectGroup` with two terms (`Taxon` and `GeographicContext`) attached as associations.

Associations enable you to: *reflect the scope of your collections using a range of properties and their associated values* keep the data structure relatively simple and maintainable, with a small number of ObjectGroups * link between ObjectGroups using common properties and vocabularies

However they will only give you basic figures for your collections, restricted to the absolute numbers in the ObjectGroups, and not reflecting the associated properties.

Dimensions

Using dimensions creates a consistent structure to the breakdown of the collection according to one or more common properties. Metrics can be used to accurately describe, analyse and visualise the collections in numeric terms.

As a dimension can only have one value for any given `ObjectGroup`, this effectively means the collections being described must be split into as many ObjectGroups as there are values for that dimension. If more than one property is designated as a dimension, then there must be an `ObjectGroup` for every valid combination of values in those dimensions.

Figure 22 shows an example where two terms - `Taxon` and `GeographicContext` - have been designated as dimensions. Each dimension has two possible values, so the collection must be split into four ObjectGroups, one for each combination of those two dimensions.

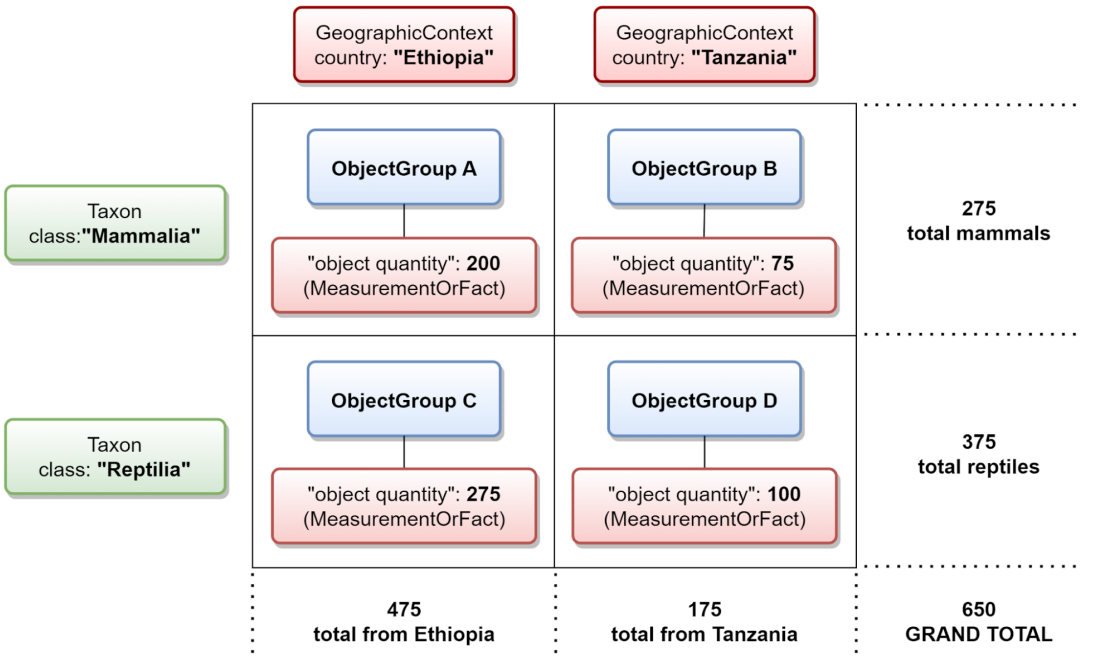


Figure 22: Breaking down a collection using two dimensions.

Metrics (represented by the `MeasurementOrFact` class) are attached to each `ObjectGroup`, in this example 'object quantity'. This means that it's possible to calculate the number of objects within a dimension (e.g. 175 objects from Tanzania or 375 reptiles), across the two dimensions (e.g. 200 Ethiopian mammals, 100 Tanzanian reptiles), and for the collection as a whole by aggregating numbers from all ObjectGroups.

In the real world however a collection is likely to have many more taxa, and many more geographic origins, and so the number of ObjectGroups in the grid would probably be considerably larger. Also, if a third dimension is introduced then the number of ObjectGroups is also multiplied by the number of values in that new dimension. So there are some practical limitations on the number of dimensions that can be used (and the number of values in each dimension), related to the manageability of the data but also primarily the amount of time and effort required to estimate collection metrics at such a detailed and granular level.

Within these constraints, using dimensions is most effective in scenarios for showing collections demographics or inventories, where: *a structured breakdown of the collection is needed using a small number of properties* there is a need for dynamic, quantitative reporting across those properties * there is sufficient resource available to estimate or calculate metrics for a larger number of ObjectGroups, and maintain a more complex dataset

Combining dimensions and associations

In practice, it is likely that most use cases of the LtC model would be best suited by a combination of terms as dimensions, and terms as associations. For example, an institution might describe its collection by applying an organisational hierarchy as a dimension, so that there is one `ObjectGroup` for each `OrganisationalUnit` defined by the institution, with associated metrics. To each of those ObjectGroups, there might then be a number of other terms attached as associations to further describe and reflect the scope (e.g. taxonomic or geographic) of the objects within the group.

Model options for metrics

There are two main options for how metrics can be used within the LtC data model, each with strengths and weaknesses.

Option 1: Metrics attached to the `ObjectGroup`

The first option (**Figure 23**) is to attach the metric directly to the `ObjectGroup`. This means that you can get an accurate figure for the total number of objects within the `ObjectGroup`, but not for the number of objects according to each of the terms (unless the terms are designated as dimensions, and the `ObjectGroup` split into multiple ObjectGroups, as described in the previous section). This option tends to be better suited to the dimensional model, for providing precise statistics.

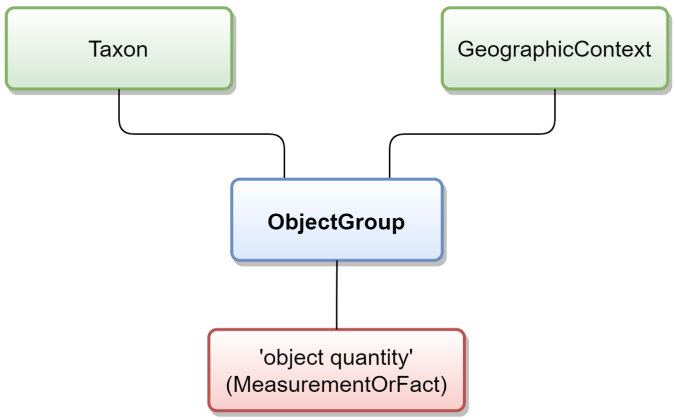


Figure 23: Attaching metrics directly to the `ObjectGroup` class.

Option 2: Metrics attached to the relationship between the `ObjectGroup` and each property

The second option (**Figure 24**) is to attach metrics to the relationships between the `ObjectGroup` and the terms (designated as associations). For example, an ObjectGroup has 100 objects from Tanzania, 150 objects from Ethiopia, 75 reptiles and 125 mammals. This means that you have accurate figures relating to each property, but do not know how many objects there are overall - there is no denominator. In practice, this can be achieved by embedding instances of the `MeasurementOrFact` in a `ResourceRelationship` used to create the relationship between the `ObjectGroup` and the term.

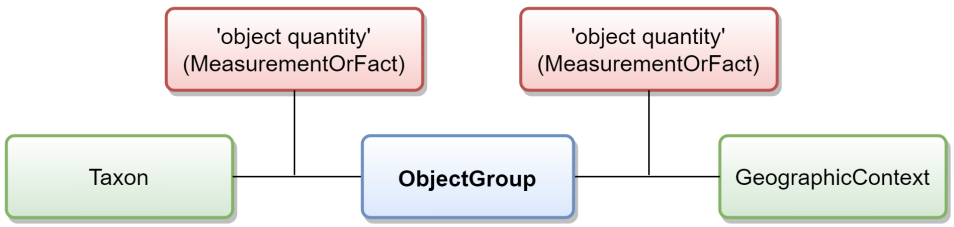


Figure 24: Attaching metrics to links between classes, in order to quantify or qualify the relationship.

This option is better for a less structured, more graph-like approach to modelling the collections. It does avoid the need to break down into a greater number of more granular ObjectGroups, as per the dimensional model used in Option 1, but has limitations in providing accurate quantitative data.

As with the two options of using properties as associations or dimensions, there is potential to combine these two approaches to suit the use case.

Key points

- Most terms related to an `ObjectGroup` can be used as either an association or a dimension.
- Associations require less effort and are good for descriptive purposes, but bad for quantitative reporting.

- Dimensions are good for quantitative reporting, but require more effort, and have limitations in how many can be applied.
- Metrics can potentially be attached either directly to the `ObjectGroup`, or to the relationships between the `ObjectGroup` and its terms.

8.-Reference-examples-and-implementations.md

8.1 Wikibase as a modelling tool

Usability of a standard is not easily tested during the development phase. In order to test usability, one needs to make a conceptual model using the terms inside the standard and fit the data against it. This is potentially a very technical and time consuming task, for which it is not easy to integrate the wider community into it. To serve this purpose, we've set up a Wikibase instance that contains all terms from the LtC standard: https://latimer-core.wikibase.cloud (https://latimer-core.wikibase.cloud).

Wikibase is the underlying software package that drives Wikidata. Wikidata has proven to be an ideal tool to allow non-technical users to add information into a database. The approach also allows to model different ways of representing the information on collections. Having an easily accessible graphical user interface allows fast data entry for testing (see **Figure 25**).

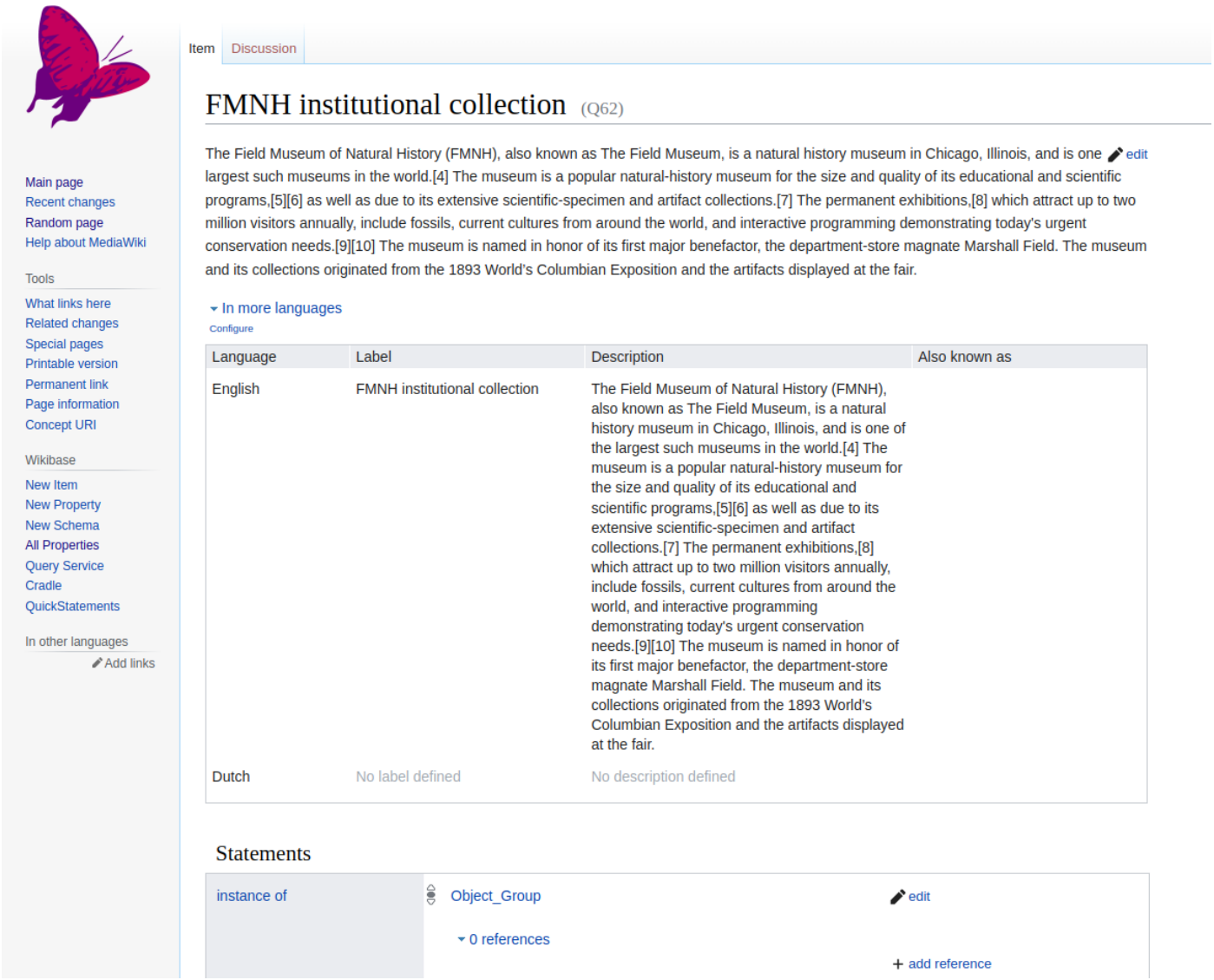


Figure 25: Screenshot of a record (https://latimer-core.wikibase.cloud/entity/Q62) in the Latimer Core Wikibase instance

Adding records inside the Wikibase is a way of testing the different use-cases and models of the standard. It also provides a way to prepare the standard for linked open data approaches. During the life cycle of the standard, the tool can still be used to point out weaknesses of the

current version of it and provide useful insight in potential improvements.

In a later stage, it is planned to implement Shape Expressions (https://www.wikidata.org/wiki/Wikidata:WikiProject_Schemas) in the Wikibase instance of the standard. These are expressions of the way the standard can be implemented and how a record should look like. This would allow us to provide tools to people to validate their records against a certain scheme.

9.-Use-Cases.md

Use Case Analysis

Each group member (and other interested parties) was asked to break down each use case (above) using the “epic stories format” that comes from the Agile Software Development methodology. This consists of identifying the Actor and the Actor’s role, and what they would like to be able to do, if they had the necessary collections metadata available.

- Please use this form (https://docs.google.com/spreadsheets/d/1SsfwogZ88TgouDJ7EoDqXJFol-eVs7aYdFx504qJNzc/edit?usp=sharing) to add your analysis of the above use cases. Those developing the data model will use these data to test the model scenarios to help insure we will be able to do (share) the kinds of tasks we want to do with this information.

Index Herbariorum

The IH search page is consistently the most-visited webpage in the NYBG Virtual Herbarium (72,998 visits in the past 12 months, or 199/day). Study of web use statistics and direct feedback from users reveals a wide user base, the main groups of which are summarized below.

Herbarium commerce

Annually, herbaria send and receive an estimated 10 million specimens as loans, gifts, exchange, and other transfers. The information contained in IH allows herbaria not only to address their shipping boxes correctly, but also to assess the reliability of herbaria to which they are considering lending specimens. Herbaria use the information included in IH about staff expertise to find individuals who can provide identification or evaluation of specimens of a particular group or region. Herbaria use IH to find partners for the exchange of specimens that augment their own holdings.

Biodiversity surveys or evolutionary reconstructions of a taxonomic group

Researchers use IH to find previously collected specimens that are pertinent to their studies, to determine what taxa and geographic areas are well represented and which are not (and therefore require additional study) and to develop collaborations to facilitate new field studies. In order to obtain permits for field surveys in a foreign country, it is usually essential to develop a partnership with an herbarium in that country, where one set of voucher specimens must be deposited. Scientists conducting research in foreign countries use IH to determine the best partner in the relevant country and to make contact with those individuals.

Scientific journals

Journals use IH codes in the citation of specimens examined for a given study, including the designation of type specimens in the description of new species. Journals such as Systematic Botany, Mycologia, Brittonia and The Bryologist will not accept for publication an article describing a new species that does not cite a standard IH acronym as the place of deposition for the type specimen of a new species.

Collaborative projects

Collaborative projects often use IH as a tool for identifying the range of participants and resources for such projects. For example, IH played a key role in identifying potential participants in the Global Plants Initiative (5) a project to digitize all type specimens in the world’s herbaria and serve these through a common web portal. The National Ecological Observation Network (6) has used IH to find taxonomic expertise and collection repositories near NEON sites to help with the biodiversity inventories of these sites. The International Association of Plant Taxonomists uses IH data to determine the number of institutional votes each herbarium should have for the nomenclature session at the International Botanical Congress meetings every six years.

U.S. Government agencies use IH for several purposes

Collecting permits for National Parks and other protected federal lands require that specimens collected on these sites be deposited in IH listed herbaria. The U.S. Fish & Wildlife Service uses IH as a resource for determining whether or not an institution should be granted a permit to house endangered species (a CITES permit) and uses IH code as part of the CITES permit identifier. Recently U.S. Fish and Wildlife used the email contact list for IH to send a questionnaire to all herbaria asking them to evaluate their experiences applying for and managing CITES permits. Additionally, IH is used by Homeland Security to find specialists for the identification of unknown specimens confiscated at U.S. customs. Natural Resource Managers use IH to find experts to identify species on federal lands. The National Park Service uses IH to keep track of specimens collected in National Parks for their annual inventories and collection assessments.

Smithsonian Field Book Project

The Field Book Project is a collaborative grant-funded initiative based at Smithsonian to catalog, digitize, and provide open access to scientists' field notes held in the collections of the Smithsonian Institution Archives and other Smithsonian departments.

The Field Book Project cataloging approach involved creating collection level records, item level records, and authority records for people, organisations, and expeditions. Collection level records were created using a subset of elements from NCD v.0.7. Item level records were created using MODS (Metadata Object Description Schema, a somewhat simplified version of the widely adopted library schema MARC), and EAC-CPF (Encoded Archival Context for Corporate Bodies, Persons, and Families) for the authority files.

This use case will focus on the collection level records which were created using the subset of NCD v 0.7 data elements. As of 23 February 2018, 658 collection records have been created by the Field Book Project. The majority of these records have been published to the Smithsonian Collections Search Center and can be viewed here: http://collections.si.edu/search/results.htm?view=&dsort=&date.slider=&q=unit_code%3AFBR+collection+name (http://collections.si.edu/search/results.htm?view=&dsort=&date.slider=&q=unit_code%3AFBR+collection+name)

Collections as defined for field book description

From Nakasone, S. and Sheffield, C. Descriptive Metadata for Field Books D-Lib Magazine, Volume 19, Number 11/2 (November/December 2013) http://www.dlib.org/dlib/november13/nakasone/11nakasone.html#4 (http://www.dlib.org/dlib/november13/nakasone/11nakasone.html#4):

> A 'collection' is defined as any group of field books with a unifying relationship (4). Field book collections can be assembled in many ways; Smithsonian collections, however, are usually grouped by collector or expedition. For example, a collection grouped by the collector Alexander Wetmore would consist of field books created or owned by Wetmore. Alternatively, a collection grouped by the expedition United States Exploring Expedition might consist of field books created by various individuals that participated in that expedition. Less frequently, collections are assembled by the organisations as a creator. However they are grouped, collections are determined based on the way the field books were physically organized, with respect to the provenance and order in which they were received and maintained, prior to our involvement, in accordance with archival practice (5).

> (4) Ibid, 203.

> (5) We retain the organization of collections already accessioned by SIA. We use descriptive information in finding aids, when available, as a spring board for our own records and to maintain consistency. For collections maintained by museum departments, generally each collector is given her own collection because that is how the museum organizes it. If the museum chose another way to organize a collection, for example, kept all the field books inherited by another museum together as one group, we would organize the collection in the same way.

NCD 0.7 as used by Smithsonian Field Book Project

These collection records are currently used to aggregate item records from a single field book collection together on the Smithsonian Collections Search Center (SI-CSC): http://collections.si.edu/search/results.htm?view=&dsort=&date.slider=&q=unit_code%3AFBR+collection+name (http://collections.si.edu/search/results.htm?view=&dsort=&date.slider=&q=unit_code%3AFBR+collection+name)

Each of these collection and item records also link to the Field Book Project-produced authority profiles for persons, organisations, and/or expeditions involved in their creation.

For more information on how the Field Book Project used NCD v0.7, the Field Book Project Cataloging Manual is available here: https://

drive.google.com/file/d/0BzbZlJVfq9rPUHQ3NDl6Y2Zha0U/view (https://drive.google.com/file/d/0BzbZlJVfq9rPUHQ3NDl6Y2Zha0U/view) Section 1 focuses on Collection Level Description and includes a summary of elements used followed by detailed information specific to how the Field Book Project uses those elements, including definitions, recommended data values, and examples.

GBIF Data Management

GBIF has a multi-faceted interest in collection descriptions metadata.

Publishing and sharing collection description metadata

Some GBIF nodes (notably Atlas of Living Australia) collect and manage collection metadata as a national-scale catalogue. This is frequently a scale at which such information can efficiently be gathered and curated, and GBIF is interested in promoting and reinforcing this approach across its nodes, while at the same time avoiding unproductive overlaps with Index Herbariorum and other community initiatives.

Increasing linked-open-data aspects of GBIF aggregated data

A primary goal for GBIF in aggregating and indexing specimen data (and other occurrence records) is to increase normalisation and to deduplicate diffuse text references to known concepts and entities, including e.g. countries, taxon concepts, field collection protocols, etc. A key class of entities for which this should be possible is the set of natural history collections. A well-managed catalogue/vocabulary of the world's NHCs would offer many opportunities to normalise collection codes/identifiers in specimen records and establish linked-open-data pathways within accessible biodiversity data.

Collection description metadata as anchor for subsequent digitisation

Publishing a collection description can be seen as the first step in bringing a collection online and is consequently a useful point to start engagement with collections. The description record can then serve as the anchor point for linking further digitisation products, including refining taxonomic scope through checklists, publishing Darwin Core records, and augmenting with rich digital media and other artifacts.

Promoting data mobilisation by GBIF national nodes

A comprehensive baseline catalogue of the world's NHCs, including information on taxonomic/geographic scope and coverage and of current digitisation status, would form a key tool to support annual planning by nodes for priority actions to support or advance data mobilisation, and a baseline for measuring progress.

Supporting researcher discovery and access for undigitised materials

GBIF is interested in understanding which collections hold currently indigitised materials, so these collecitons can be highlighted as part of some searches. This will assist taxonomists and others in locating material and perhaps stimulating digitisation.

Developing fundable digitisation proposals

It will be much easier to plan and seek funding for digitisation projects for a single institution or a network of related collections if there is a clear public view of what they hold and what they have to offer as part of an integrated biodiversity knowledge resource. This is an opportunity for collections to highlight what they have that is unique.

Field Museum of Natural History

Undigitized Collections Statistics

https://collections-dashboard.fieldmuseum.org (https://collections-dashboard.fieldmuseum.org) The Collections Dashboard project is an attempt to automate the process of estimating undigitized backlog (both uncatalogued and catalogued) across institutions for the purpose of revealing hidden collections and estimating digitization effort.

This use case will focus on the collection-level record structure that is required for calculating estimates of backlog, and comparing with digitized catalog record structure. It also focuses on the need for standardized language and field structure for taxonomy and geography in collection-level records.

Documenting Institutional Collections Through Time

The documentation of institutional collections and their changes through time has the goal of facilitating accurate attribution and collections-tracking.

This use-case will focus on the representation of an institutional collection through time, involving additions & subtractions, name-changes, and nested relationships between collection elements.

Use Cases from fishfindR (http://www.fishfindr.net)

A website to explore data from institutional fish collections integrated in iDigBio, a data aggregator. Using iDigBio's specimen data API, these data can be retrieved without involving collections staff, and the data can be used in near-real time to give the most accurate snapshot of current collection holdings as they are published. This prototype website provides a framework in which all digitized collections can be analyzed and compared using data from iDigBio's API's.

Motivating collections to publish well curated data

Some institutional collections are not publishing their data for integration by a data aggregator, despite the approachability and ease of publishing data. For the most part, collections that are not published are small regional collections not associated with natural history museums, although some large institutions also are not publishing digitized data. By illuminating the relationship between data publishing and data use, collections staff see how data are used more frequently from collections who publish to aggregators and it incentivizes the push to publish their data via data aggregators which in turn increases value and use as well as expands our information on biodiversity.

Profiling collections

Each fish collection has their own method for naming and describing themselves. Traditionally, ichthyology collections use the American Society of Ichthyologists and Herpetologists Standard Symbolic Codes for Institutional Collections to identify themselves, but individual institutional decisions can disrupt adherence to community naming conventions. Instead, many collections publish data to aggregators that obfuscates their institutional or collection affiliation and fishfindR attempts to bring them to a set of standards (ASIH codes) for use in the data reports. In addition, it is a goal of this project to motivate the fish community to work towards a set of standards for describing collections data. In addition, the community may decide that some of the data synthesized from these types of analyses might be useful in describing collections.

Aiding collections in communicating value

fishfindR provides tools for collections staff to compare across and within collections. In doing so collections staff can find unique characteristics of their collection. These superlatives and unique attributes can be used to help secure funding or as a lobbying tool within their institution.

Collections community empowerment

Using the fishfindR website, it is now possible to explore and to communicate about every U.S. fish collection that is digitized and aggregated within iDigBio. The data from these analyses can be used to answer many questions related to biodiversity, conservation, behavior, and life history, and can in turn be given back to collections staff to aid in soliciting support. The goal of the fishfindR project is to give data back to collections and facilitate exploration of the data. With their data visible in an easy to analyze and approachable way collections staff will know how their data looks to end users and be empowered to make the necessary changes to better facilitate the needs of their stakeholders.

Use Cases from Global Registry of Biodiversity Repositories GRBio.org (http://www.grbio.org)

This list from document dated 1-April-2012 and shared at the TDWG 2016 meeting of the NCD Interest Group.

```
I’m in charge of a collection (or all collections) at a biorepository and I want to **create and/or update information**

I’m looking for basic information about: particular museum/herbarium/other biorepository institution; particular collecti

I would like to receive information: * As a webpage I can browse * As a webservice response

I’m doing a survey/study of biorepositories and I would like to get a data dump on all institutions and their collections

I found a reference to a specimen of interest * In a publication * In GBIF (through a taxonomic search, on a distributior

I am writing a scientific article and I want to include references to particular specimens. How should I cite these speci

I’m submitting data records to GenBank, GBIF or another database and I want to make sure they are traceable back to their

I’m looking for a specimen in a collection that no longer exists. I want to find out if it has been absorbed by another b
```

Use Cases from ICEDIG

ICEDIG is one of several design studies helping to develop the DiSSCo vision. One ICEDIG group took up the task to define the requirements for a collections digitization dashboard (CDD) for DiSSCo. As part of this task, the CDD group gathered use cases in a specific format. These use cases are available \

- from the ICEDIG Design of a Collection Dashboard (https://drive.google.com/file/d/1RLNwHuZn0xLZuLWTrJaKiQn8IjQXwbCE/view) (Deliverable D2.3), or \
- from the documents folder (https://github.com/tdwg/cd/blob/master/documents/final/Deliverable%20D2.3%20ICEDIG%20-%20Design%20of%20a%20Collection%20Digitisation%20Dashboard%20v1.0.docx) in this repository.

Home.md

Latimer Core Guidance Documentation

About the Wiki

Current Version: 1.0

This wiki is intended to support the normative definitions (https://tdwg.github.io/ltc/terms/index.html) by providing further information on the terms and concepts represented in the Latimer Core (LtC) standard, and guidance on how the standard may be used in practice. The wiki content does not form part of the LtC standard itself (it is non-normative) and may be modified, improved and extended over time.

The classes and properties in the first version of the LtC standard were primarily intended to allow for the description of natural history collections, as such they reflect the categories of specimens, objects and items most usually associated with them. This should however, not in any way preclude future developments by other expert groups to include different types of collections.

Versions

Standard

Version History

Label	Version	Review Purpose	Contributors	Status
1.0	2024-28-02	Ratification	Matt Woodburn, Jutta Buschbom, Sarah Vincent, Kate Webbink, Maarten Trekels, Janeen Jones, Sharon Grant, Ben Norton	Complete
0.4	2023-11-12	Public Review	Matt Woodburn, Jutta Buschbom, Sarah Vincent, Kate Webbink, Maarten Trekels, Janeen Jones, Sharon Grant, Ben Norton	Complete
0.3	2023-19-10	Expert Review	Matt Woodburn, Jutta Buschbom, Sarah Vincent, Kate Webbink, Maarten Trekels, Janeen Jones, Sharon Grant, Ben Norton, Ian Engelbrecht, Robert Sanderson	Complete
0.2	2022-06-10	Working Group Proposal	Matt Woodburn, Jutta Buschbom, Sarah Vincent, Kate Webbink, Maarten Trekels, Janeen Jones, Sharon Grant	Archived (https://github.com/tdwg/ltc/blob/main/Ltc_wiki_v1.pdf)
0.1	2022-02-10	Original Draft	Matt Woodburn, Jutta Buschbom, Sarah Vincent, Kate Webbink, Maarten Trekels, Janeen Jones, Sharon Grant, David Bloom, Gabi Droege	Complete

Archived Meeting Notes

Date Archived	Title	file
12 April 2024	Pre-ratification Monthly CD Interest Group Weekly Notes	pdf (https://github.com/tdwg/ltc/files/14964280/TDWG_CD_Task_GroupMonthlyMeetingNotes.pdf)
4 April 2024	Pre-ratification Task Group Weekly Notes	pdf (https://github.com/tdwg/ltc/files/14948515/CD_LtC_Notes2021_24_ARCHIVED.pdf)
18th November 2021	CD Interest Group - Working session @ TDWG2021 Notes	pdf (https://github.com/tdwg/ltc/files/14948673/NOTES_.ITG08_.CollectionDescriptionsInterestGroup_working_session_notes.pdf)

Release-Notes.md

Expert Review 2022

Changes and Updates

- Created label “UpdateReview1stPhase” (JuB)
- Changed Identifier.identifier to Identifier.identifierValue: updated in issues/129 (https://github.com/tdwg/cd/issues/129)) and Collection Description Class Index spreadsheet (https://docs.google.com/spreadsheets/d/1w8DMgUwl7tf-9AXQOpT6IRQeMuUbxUZlJwQinrtUvAs/edit#gid=478678659) - tabs “1st draft properties worksheet”, “CD classes and properties”, “Final R&R review” “Institution registry mapping”, “GBIF Collection registry mapping”, “GBIF Tables” and “Sheet25”. Updated title of issue #129 (https://github.com/tdwg/cd/issues/129) to “Property:identifierValue”. (JuB)
- 20220811 JuB: Updated Identifier.identifier to Identifier.identifierValue on Wiki page 2.2 “Latimer Core schemes”, code block Step 2.
- 20220822 JuB: Wiki section 2.4 Table 1: removed “SpecimenIdentifierSystem” from range of “Person”, added “ChronometricAge”,

“GeologicalContext”, “GeographicContext”

- 20220822 JuB: Wiki section 2.4 fig. 5, 6 & 7 - updated Identifier.identifier to Identifier.identifierValue
- 20220822 JuB: Wiki section 2.6 - updated Identifier.identifier to Identifier.identifierValue throughout examples
- 20220822 JuB: Wiki section 3 - added extension to text before Table 2 as an explanation why this table differs from the table in the normative part (“, including also sources that informed the development of LtC and are the origin of informally derived terms”)

Future components/discussions

In addition to the version of Latimer Core explained in the wiki, the Latimer Core Task Group discussed other classes and terms to provide users with ways to describe other aspects of collections in more detail when relevant or possible. These other classes were excluded from the initially proposed version of the standard because they were either not found to be critically necessary for the known needs and use cases, or needed further investigation and definition. If such needs or use cases arise in the future, though, relevant notes are highlighted here.

Activity and PersonActivity

- Activity and PersonActivity

• Class Name	• Range (i.e., other classes to which this class can be attached)
• Activity	• Event, ObjectGroup
• PersonActivity	• Activity, Event, MeasurementOrFact, ObjectGroup, OrganisationalUnit, RecordLevel

• Class Name	• Class-level Properties
• Activity	• Identifier, PersonActivity
• PersonActivity	• Address, ContactDetail, Identifier, Person, Reference, TemporalCoverage

Connection between Latimer Core (ltc) and Audiovisual Core (ac)

- See github issue 369 (https://github.com/tdwg/cd/issues/369)