LSID Deployment in the Catalogue of Life

Ewen R. Orme, Andrew C. Jones and Richard J. White
Cardiff University, School of Computer Science, Queen's Buildings, 5 The Parade, Cardiff CF24 3AA, UK
E.R.Orme@cs.cardiff.ac.uk
Andrew.C.Jones@cs.cardiff.ac.uk
R.J.White@cs.cardiff.ac.uk

Abstract. In this paper we describe a GBIF/TDWG-funded project in which LSIDs have been deployed in the Catalogue of Life’s Annual and Dynamic Checklist products as a means of identifying species and higher taxa in these large species catalogues. We look at the technical infrastructure requirements and topology for the LSID resolution process and characteristics of the RDF (Resource Description Framework) metadata returned by the resolver. Such characteristics include the use of concepts and relationships taken from the TDWG (Taxonomic Database Working Group) ontology and how a given taxon LSID relates to others including those issued by database providers and those above and below it in the taxonomic tree. Finally we evaluate the project and LSID usage in general. We also look to the future when the CoL LSID infrastructure will have to deal changing taxonomic information, annually in the case of the Annual Checklist and possibly much more frequently in the case of the Dynamic Checklist.

1 Introduction

The Catalogue of life\(^1\) (CoL) is seeking to build a catalogue of all known species. It uses a distributed architecture [1], which is important in order to provide suppliers of component databases with the autonomy and control they require. Users of scientific names are faced with the problem that disagreement amongst taxonomists will lead to different scientific names being used to refer to the same organism, and to variation in the range of organisms that a given name might refer to. In order to provide a complete “synonymic index” of all the world’s species, the Species 2000 programme was set up. It is creating a catalogue of known species, with their accepted names, ambiguous and unambiguous synonyms, misapplied names, vernacular names, and some other basic data, by dynamically linking available checklist databases for different higher taxa, aiming to complete the taxonomic hierarchy and hence all known species by 2012. In partnership with the North American ITIS organisation, it delivers the Catalogue of Life (CoL) in two main forms: the Dynamic Checklist is updated on the Web, ideally in real time as the component federated databases are updated.

\(^1\) http://www.catalogueoflife.org/
and the Annual Checklist is a snapshot of the CoL released on CD and on the Web every year.

There is therefore a need to be able to refer unambiguously to a particular species in a particular usage, e.g. in a classification. Life Science Identifiers (LSIDs) are useful as a means of referring unambiguously to objects in biological knowledge bases [2], and increasingly the biodiversity informatics community has been adopting LSIDs, adapting their usage to the specific kinds of knowledge that need to be handled in this field. The main purpose of the present paper is to describe how LSIDs have been introduced into the CoL in the GBIF/TDWG-funded SPICETIP (SPICE TDWG Infrastructure Project) project. We commence by introducing LSIDs; then we present details of how LSIDs have been incorporated in the Annual and Dynamic Checklist products of the CoL, following which we explain how these LSIDs can be used (the resolver, and the metadata available), and how LSIDs originating from a provider database are accommodated. At the end of the paper we evaluate the utility of LSIDs in our application domain, and consider how their use might evolve in future.

2 Introduction to LSIDs

The LSID resolution project defines LSIDs as “persistent, location-independent, resource identifiers for uniquely naming biologically significant resources”. An LSID identifies a piece of byte-persistent data and/or a metadata document description. Names of taxa, among other things, are suitable candidates to be assigned LSIDs. LSIDs are Universal Resource Names (URNs) and have the constituent parts shown in Figure 1. Provided with such an identifier, a user can make use of an on-line or offline client to contact the individual authority’s resolver. Page’s LSID tester [3] and the Firefox plugin are examples of such clients and are both available via the LSID resolution project site previously mentioned.

The authority domain part of the LSID shown in Figure 1 requires the existence of an SRV record on the Domain Name Server (DNS). This record must refer to the actual end point (the IP address of the server and the TCP/IP port number) for the resolution service. The implementation of such a service is independent of how a client accesses it and thus, as long as it makes provisions for the standard data and metadata requests, the language used to develop the resolver is left up to the authority. Typical actions of a client include looking up the SRV record, calling the endpoint and parsing the RDF response when it is returned. The namespace and object identifier parts of the LSID are used by the resolver to locate or build the corresponding data or metadata from local resources such as a database. The level of transparency of the object identifier is open to debate and while some authorities may use their local indexing sequence, others feel this

\[\text{http://wiki.gbif.org/guidwiki/}\]
\[\text{http://lsids.sourceforge.net/}\]
\[\text{http://linnaeus.zoology.gla.ac.uk/~rpage/lsid/tester/}\]
\[\text{http://lsids.sourceforge.net/resources/firefox-lsid-browser/}\]
may lead to unwarranted assumptions and predictions being made by external clients about the internal representation of the data or dynamic formation of the authority’s other LSIDs. In order to avoid such concerns, the object identifier used may be a longer and more obfuscated sequence. The end part of the LSID is the optional version field which many authorities omit altogether for the main reason that the persistent requirement of LSIDs means that change is signified by the assignment of a new identifier rather than versioning of an existing one.

Fig. 1. LSID Components

3 LSID Deployment in the Annual and Dynamic Checklists

Annually since 2001 the Catalogue of Life (CoL) has been producing a database accessible via the Web and also distributed on CD as an integrated and standardised taxonomic hierarchy and species name list consisting of records from a large number of participating databases. This is known as the Annual Checklist (AC). In addition there is the Dynamic Checklist (DC) facilitated by the SPICE CAS (Common Access System) software [1] which accesses the current state of provider databases exposed to the Web through the use of wrappers. This state is cached for efficiency. SPICE also incorporates a Web Service which provides SOAP (Simple Object Access Protocol) access to the DC which is used by the DC’s Web user interface.

The SPICETIP (SPICE TDWG Infrastructure Project) adds LSID support to both the Annual and Dynamic Checklists. As a result of this project, the 2008 AC contains LSIDs which will be used as examples in this paper. The CoL partners decided to use Universally Unique Identifiers (UUIDs) as the object identifiers and they also decided that the version field should be used. Using the version field allows the reuse of an object identifier between different AC years (e.g. :ac2008, :ac2009) and between the AC and the DC (e.g. :dc) in cases where they refer to the same taxon. This allows the possibility for software or humans simply to compare the object identification parts of two LSIDs to determine whether they refer to the same taxon, and to determine which CoL edition the LSID came from, without the need to call the resolution service. The implications for taxon matching are discussed in the final section of this paper.
In the AC, a new field was added to a table containing all taxa to hold the LSID. In the DC provisions were made to hold the LSIDs in a separate database from the cache. The MySQL `UUID()` function was used to generate the object identifier LSID part. In the AC, LSIDs were assigned by running the appropriate MySQL query to add them to accepted and provisionally accepted species and higher taxa prior to being released. In the DC, code was introduced into the SPICE CAS to assign LSIDs to new taxa as they enter the system. Additions were made to both the AC and DC interfaces to display LSIDs and the SPICE Web Service was modified in order to communicate them to the DC interface. The modified systems also allow data providers the option to supply their own LSIDs for their taxa and LSIDs for taxon names (accepted names and synonyms). These provider LSIDs are not made visible in the user interfaces: they are accessed via the CoL LSID resolver.

4 The Resolver and Metadata

The resolver for AC and DC LSIDs is written in Java using the LSID Java Toolkit available from the LSID resolution project site.\(^6\) The resolver has access to both the AC database and the SPICE cache database in order to form the response for a metadata request. (All the CoL information associated with an LSID is held in the metadata of the object referred to by the LSID, not its data.) It also has access to what we will refer to as the “LSID Repository”, which is a database that contains both a table to hold DC LSIDs and a table to hold relationships between LSIDs. The latter is consulted during every resolution process to enrich the current LSID with relationships to others. These may be relationships between the AC and the DC or between either the AC or DC and a provider assigned LSID. Modification of SPICE means that this table can be stocked with external LSIDs (if they are given by the provider) during the caching process.

The Taxonomic Search Engine \(^4\) was likely the first application of LSIDs to taxon names stored in the databases it queries and while different in scope and coverage, we loosely follow many of the approaches laid out by this project for dealing with such things as classification and synonymy \(^5\) while opting to use ontology concepts as a basis for our metadata.

TDWG provides an ontology built on top of TCS (Taxon Concept Schema) elements to assist and standardise the RDF metadata returned by LSID resolvers.\(^7\) An ontology provides a set of entities that can exist in a certain domain (referred to as classes or concepts) and a set of relationships that can exist between them (referred to as object properties). Amongst the concepts provided by the TDWG ontology are `TaxonName` and `TaxonConcept`. As a provider of a complete taxonomy using the names contained in its databases, it was decided that the core element (that which has the `about=X` attribute where X is the current LSID) returned by the CoL’s resolver for each LSID should be of type `TaxonConcept`

\(^{6}\) [http://sourceforge.net/projects/lsid/](http://sourceforge.net/projects/lsid/)

\(^{7}\) [http://wiki.tdwg.org/twiki/bin/view/TAG/LsidVocs#The_TDWG_LSID_Vocabularies](http://wiki.tdwg.org/twiki/bin/view/TAG/LsidVocs#The_TDWG_LSID_Vocabularies)
i.e. the accepted concept that has a single accepted name. This could also be described as the “idea” in a person’s mind when using a certain name to describe a taxon. The ontology also provides a means to relate the core $\text{TaxonConcept}$ to others to better give it context by using the $\text{hasRelationship}$ object property and the $\text{Relationship}$ class. The $\text{HasVernacular}$ and $\text{HasSynonym}$ relationship types (or categories) are used to relate the current concept to its common name and synonym counterparts.

In a taxonomic hierarchy it can be beneficial to view a concept in the context of its parent (the next higher taxon of which it is a member) and children (the smaller taxa which it comprises). Usefully, the TDWG ontology also has relationship types to represent hierarchical structure and these are $\text{IsParentTaxonOf}$ and $\text{IsChildTaxonOf}$. Figure 2 shows a section of the metadata response for the LSID displayed in Figure 1 and shows examples of some of the previously described relationships. This LSID is for the species $\text{Abrus precatorius}$ and in Figure 2 we can see its parent LSID (for the genus $\text{Abrus}$) and one of its children LSIDs (for the subspecies $\text{Abrus precatorius subsp. africanus}$) as well as one of its common names (of which it has several). Accepted names, common names and synonyms can all have literature references attached to them using the $\text{publishedInCitation}$ although they have been removed, along with additional information, to save space in Figure 2. The ability to use only LSIDs to traverse up and down the hierarchy means that all taxa are reachable via the resolver from any single LSID’s metadata. Some clients such as the Firefox plugin can take particular advantage of this, as they display additional LSIDs within a document as links so a user can manually traverse up and down the tree with simple clicks. In addition, representing the hierarchy this way opens up further possibilities for specific programmatic access. Note that in Figure 2, the namespace declarations have been removed; also $C$ refers to elements in the $\text{TaxonConcept}$ namespace and $N$ refers to elements of the $\text{TaxonName}$ namespace. The same applies to Figure 3, referred to in the following section.

5 Relationships Between LSIDs

We have already discussed how a given CoL LSID relates to those of its children and parent. As mentioned previously, the resolver can communicate with three different databases: extended versions of the AC and DC databases, and the LSID Repository. The latter makes provisions for relating a CoL LSID to another CoL LSID or to an externally assigned one. For example, Index Fungorum\(^8\) is a contributor to the AC and also already provides its own LSIDs which usefully can be obtained without the need for additional information other than that already present in the AC. The core element returned by Index Fungorum’s resolver is the $\text{TaxonName}$ element from the TDWG ontology. Figure 3 shows an extract from a CoL LSID response for which Index Fungorum has already issued a name LSID. The TDWG ontology provides the $\text{IsCongruentTo}$ relationship type for describing such links. However, a $\text{Relationship}$ in the ontology is between two

\(^8\) http://www.indexfungorum.org
Fig. 2. Resolver Metadata Response Example 1

concepts of type TaxonConcept and here one subject of the relationship is a TaxonName. As can be seen from Figure 3, an anonymous TaxonConcept is built around the TaxonName element which links to the provider-assigned LSID and then this new TaxonConcept is used in the IsCongruentTo relationship.

The Common Data Model (CDM) is a schema to which provider wrappers need to conform in order for their data to be parsed correctly by the SPICE CAS. In order to accept provider-assigned LSIDs, provisional changes have been made to the CDM, which are optional on the part of the provider, to expose their own LSIDs through the use of two new attributes: CONCEPTLSID and NAMELSID. Figure 4 shows an extract from an experimental International Legume Database and Information Service (ILDIS) wrapper where these attributes have been filled with fictitious LSIDs; both attributes in the case of the accepted name but just NAMELSID in the case of the synonym. This is an example of what the CDM
document defines as a “type2” response: it contains all information about a given species (much of which has been removed from the figure, for clarity).

Fig. 3. Resolver Metadata Response Example 2

Fig. 4. Experimental LSID Enabled Wrapper Response (abbreviated)

6 Conclusion and Future Work

As time progresses, demand for the integrity and persistence of the CoL LSID infrastructure will increase along with the number of taxa it is responsible for. Not only will additional taxa be added to the databases but details on existing names may change. This will make it important to specify what changes constitute a revised taxon concept and therefore require the assignation of a new
LSID. Ultimately this will evolve beyond the simple name matching currently used in the LSID SPICE prototype, to begin employing rule-based matching on such things as references and synonyms, building on techniques such as those previously developed in the LITCHI project [1]. We have already discussed how the repository database is used very simply to store relationships between LSIDs. It is likely the future will see the importance of storing this information increase, and further development will be needed due to the need to store LSID provenance information and historical relationships.

The CoL is deploying LSIDs in line with the vision of GBIF and TDWG to promote the widespread sharing and interoperability of digital species data resources and LSIDs provide a means to this end. LSIDs are independent of any underlying protocol and so aim to have an extensive lifespan. In a field where users have varying degrees of IT experience, just being able to copy an identifier from a source directly into a browser window to obtain the RDF response would seem beneficial. Indeed, as it stands, users are forced to use some form of client, be it web-based or offline. However, the intention is for LSIDs to remain a persistent form of identification while the technology and resolution process may change around them and this is unlikely to be achieved if for example a URL-based standard was introduced instead. TDWG goes some way to addressing concerns of end-user accessibility by providing the ability to embed an LSID as a parameter in the URL for their service and thus a user can be displayed with a digest of the metadata response with links to the raw metadata and/or data from their browser. As well as encouraging users to make use of this service, TDWG also encourages authorities to put special relationships into their metadata responses to demonstrate congruence with the URL embedded version.

The uptake of LSIDs and associated standards is encouraging, and this in itself gives reason to hope that they will continue to be used in the future. The ability to provide persistent and recognisable identifiers for anything from a species name to a protein in such things as Web pages and academic publications means that the lasting value of these resources may be increased. The conventions for use of LSIDs will continue to evolve as further projects such as ours seek to apply LSIDs in various contexts. Indeed, we have seen that some issues regarding CoL LSIDs and the content of the metadata returned have not yet been fully resolved. Experiences from the 2008 CoL deployment will, it is hoped, assist us in determining how the evolution of the catalogue and the evolution of its LSIDs should be related in future.

Acknowledgements

We are grateful to the TDWG Infrastructure Project, developed by GBIF and TDWG, for providing the funding necessary to carry out the research reported in this paper.
References


