The nature.com ontologies portal

Tony Hammond and Michele Pasin*

Macmillan Science and Education, The Macmillan Campus,
4 Crinan Street, London, N1 9XW, UK
{tony.hammond,michele.pasin}@macmillan.com
http://se.macmillan.com

Abstract. This paper summarizes work done by Macmillan Science and Education to create a publicly accessible repository of the data models and datasets which underlie its semantic publishing architecture. In particular, we give a brief history of our work with linked data, we describe the `nature.com` ontologies portal and the data models and datasets we have published, we discuss mappings to external datasets and a mechanism for publishing to different knowledge-bases, and finally we provide some data handling best practices and conclude with a few hopes for future development.

Keywords: linked data, ontologies, science publishing, semantic publishing

1 Background

Macmillan Science and Education (MSE) is a publisher of high impact scientific and scholarly information and publishes journals, books, databases and other services across the sciences and humanities. Publications include the multidisciplinary journal *Nature*, the popular magazine *Scientific American*, domain specific titles and society owned journals under the Nature Publishing Group (NPG) and Palgrave Macmillan journal imprints. In total, the published corpus extends across more than 200 journals and over 1 million articles, and is growing by some hundreds of articles per day.[1]

In early 2012 we started experimenting with semantic technologies within the context of a 'digital first' internal process reengineering which was focussed on upgrading publishing operations to deal with 'born digital' assets and on making print a secondary deliverable. In order to manage better the various production workflows, we recognized the need to develop a generic data integration layer for the applications that power the specific products. In particular, we wanted

* Both authors contributed equally to this paper and are listed in alphabetical order.

[1] Springer Nature is a new merger in scientific, scholarly, professional and educational publishing and was created through the combination of Nature Publishing Group, Palgrave Macmillan, Macmillan Education and Springer Science+Business Media in May 2015. The joint archive amounts to over 6 million journal articles from around 3,000 journals, plus another 4 million book and reference work resources.

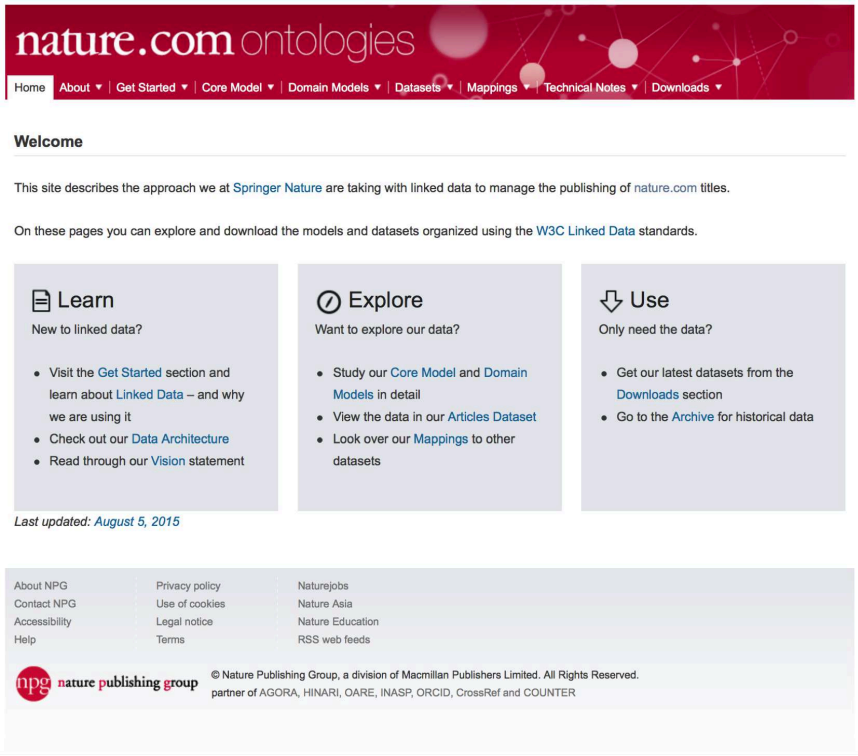

Fig. 1. Screenshot of the nature.com ontologies portal homepage.

interoperability both at the level of naming conventions – to facilitate communication within the enterprise when people talk about *articles* or *subjects* – and at a formal computational level, via shared metadata models which can be used for data validation and semantic integration.

In April 2012 (and updated July 2012), we released `data.nature.com` – a linked data platform hosting a number of RDF datasets for downloading, together with a SPARQL endpoint for querying over the data. This data (some 270 million triples) comprised bibliographic metadata for all articles (and their references) published by NPG from 1845 through to the present day. The data model was essentially flat, typed with RDFS classes but with no class hierarchy. These datasets were made available under a Creative Commons Zero public domain dedication. The platform was intended for external use only and was essentially detached from the products that end-users would see on `nature.com`, but it allowed us to gain a better understanding of how to make use of these tools within our existing technology stack. It is important to remember that over the years we have made considerable investments in an XML-centered architecture [1], so finding a solution that could leverage the legacy infrastructure with these new technologies has always been a fundamental requirement for us.

Our focus from 2013 forwards has subsequently shifted towards enterprise applications of the linked data model. This model is now being used to manage our metadata and to deliver our content discovery. A new hybrid linked data platform combines the modeling flexibility and scalability of RDF with the efficiency and publishing orientation of XML stores [2]. At the same time, since user engagement levels on `data.nature.com` were low, we decided to terminate that service. The SPARQL query functionality was decommissioned in April 2014 with the datasets being maintained online for reference purposes.

In this paper we describe the `nature.com` ontologies portal (see Fig. 1) which is the new showcase for our open linked data work. The portal was released in April 2015 and it provides a comprehensive and up-to-date repository for the semantic schemas that drive our publishing platform and their corresponding datasets, both of which are available for inspection and download. In particular, it reflects the major restructuring of the content model that has been taking place during the past couple of years: starting from a rather limited set of classes and properties, we have evolved our models into a common network of interrelated and constantly evolving ontologies consisting of hundreds of entities.

This paper is organized as follows: in Sect. 2 we describe the ontologies portal and the ontologies that we are making available; in Sect. 3 we discuss mappings from our datasets to external datasets; in Sect. 4 we briefly review our data publishing mechanism; in Sect. 5 we outline some data handling principles and techniques; and finally Sect. 6 contains a summary of this work and a few hopes for future developments.

2 Ontologies Portal

The `nature.com` ontologies portal[2] provides access to the main semantic models which drive the `nature.com` publishing platform, together with corresponding datasets, e.g. *articles* and *contributors*. This data is being provided in RDF, both for inspection and download[3]. A summary of the models and datasets together with URIs and sizings can be found in Table 1.

The portal is organized around three main sections: a) the 'Core Model' section describes the core enterprise ontology we developed to achieve semantic integration across the various MSE divisions and data repositories; b) the 'Domain Models' section contains detailed information about more discipline-specific or application-specific models used by one or more of our systems; finally, c) the 'Instance Datasets' section provides access to the various datasets included in the archive.

Figure 2 shows an overview of our semantic architecture. The foundational layer is provided by the RDF family of languages which is used to encode our Core Ontology. This currently consists of around 50 classes and 140 properties, and in particular, by inheriting the formal semantics of the SKOS [3] model, it

[2] `http://www.nature.com/ontologies`

[3] Note that at the time of writing no SPARQL endpoint has been made available for this data.

Table 1. Models and datasets

Model	Description	Entities	Triples
Ontology			
Core	Formal model of key concepts	–	1,217
Taxonomies			
ArticleTypes	Genres for scholarly articles	64	977
Subjects	Subject areas for publications	2,636	48,500
Vocabularies			
Blogs	Master catalogue of blogs	32	270
Journals	Master catalogue of journals	236	3,831
PublishStates	Publish states for a publication	7	69
Relations	Relationships between publications	30	365
ReviewStates	Review states for a publication	5	53
SeverityLevels	Severity-levels used in build rules	8	93
SummaryTypes	Summary-types for summaries	4	45
Datasets			
Articles	Biblio data for scholarly articles	1,206,039	25,212,725
Contributors	Contributors to scholarly articles	2,698,052	11,052,560

defines the primitive concepts for a number of domain level categorizations. We call these our domain ontologies since they encode knowledge which is specific to single applications or domains within the enterprise. These are implemented at the instance level using SKOS taxonomy primitives.

2.1 Core Ontology

The Core Ontology[4] is a formal model providing definitions for the key concepts used by MSE for content publishing. It includes both entities normally associated with the publishing domain (e.g. *articles*, *journals*), as well as more abstract concepts (e.g. *agents*, *events*) that group together other, more specialized, domain entities.

It should be noted that what we have published is a subset of the actual ontology used in our content production systems. Some terms have not been shared because they are inherently internal, while others are still undergoing testing and are liable to change. In general, the Core Ontology represents a measured balance between supporting legacy practices (some stretching back over many years) and enabling new requirements (which may only be revealed incrementally). It has been developed and grown within a cross-functional software delivery team. Some of the modelling clearly reflects immediate pragmatic

[4] The Core Ontology has the ontology URI `http://ns.nature.com/terms/` and uses this namespace for defining its terms. The preferred prefix is '`npg`'. This namespace is taken here to be the default namespace.

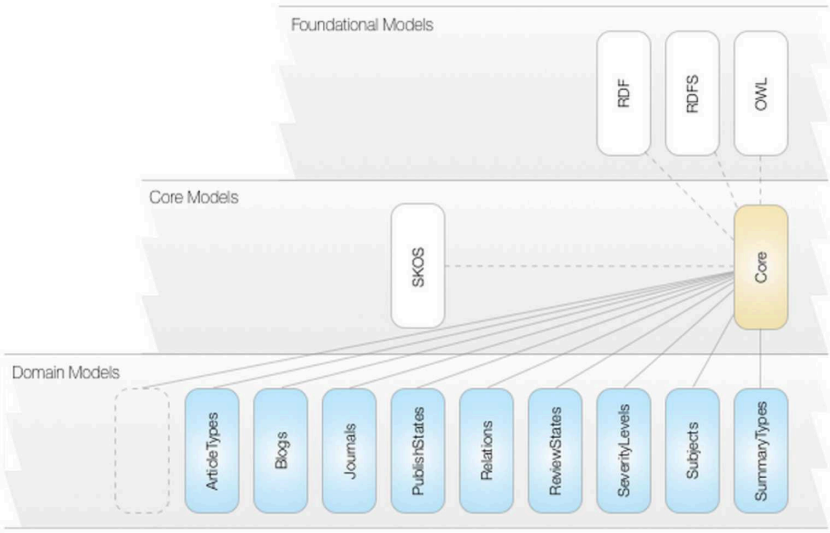

Fig. 2. Overview of our semantic architecture.

concerns and the 'operational semantics' originating from our specific system architecture, others are more clearly representative of our publishing industry bias. In both cases, we decided to publish this ontology as an enterprise artefact 'as is' so to document more realistically how we are using it to drive forward our content publishing and discovery processes.

The Core Ontology is conformant to the OWL 2 [4] specification, with a Description Logic [5] expressivity of $\mathcal{ALCHI}(\mathcal{D})$ [5]. The class hierarchy (see Fig. 3) is organized around four main branches: *agents*, *assets*, *events* and *concepts* (comprised of *publications* and *types*).

Agents The *agents* branch defines the business actors connected with the things that we publish. At this time we do not have any *persons* in our model but we do have some limited support for *organizations*, i.e. *publishers*. (Note that we do have *contributors* for the articles we publish but these have only local identity within a given article and are treated as publication components.) This is an area which is still under development.

Assets The *assets* branch defines the content storage entities for the things that we publish. We are using a centralized content storage facility – the Content Hub

[5] The notation here denotes an attributive language ($\mathcal{AL}$) with complex concept negation ($\mathcal{C}$), role hierarchy ($\mathcal{H}$), inverse properties ($\mathcal{I}$) and datatypes (($\mathcal{D}$)). See `https://en.wikipedia.org/wiki/Description_logic#/Nomenclature` for details.

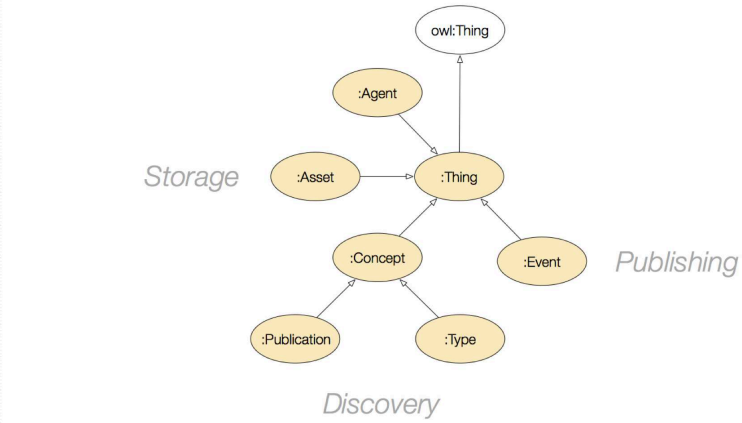

Fig. 3. Main taxonomy tree of the Core Ontology.

– to manage our content. The Content Hub implements a graph model over the content storage layer. This decoupling allows for a distributed set of physical content stores. This is achieved by maintaining a description of the content storage entity – the *asset* – which is an integral part of the graph model. For example, the *asset* records the repository and repository ID, along with other key physical characteristics.

Events The *events* branch defines the notable occurrences relating to the things that we publish. There are currently two main branches: *aggregation-events* and *publication-events*. The first deals with publication structure events, while the second deals with publication lifecycle events. We are actively making use of *publication-events* to drive forward our metadata-based publishing workflows.

Publications The *publications* branch defines the (types of) things that we publish – see Fig. 4. This is broken down into four branches: *components*, *datasets*, *documents* and *serials*. (We envisage adding in additional branches for books and other publication types as we ingest these into our systems.)

Types The *types* branch defines the domain models used to categorize the things that we publish. Each `:Type` subclass corresponds to a domain model which is implemented at the instance level using the SKOS vocabulary.

2.2 Domain Ontologies

Our domain models are implemented as SKOS concept schemes. A concept scheme comprises a set of terms representing key features of a target domain

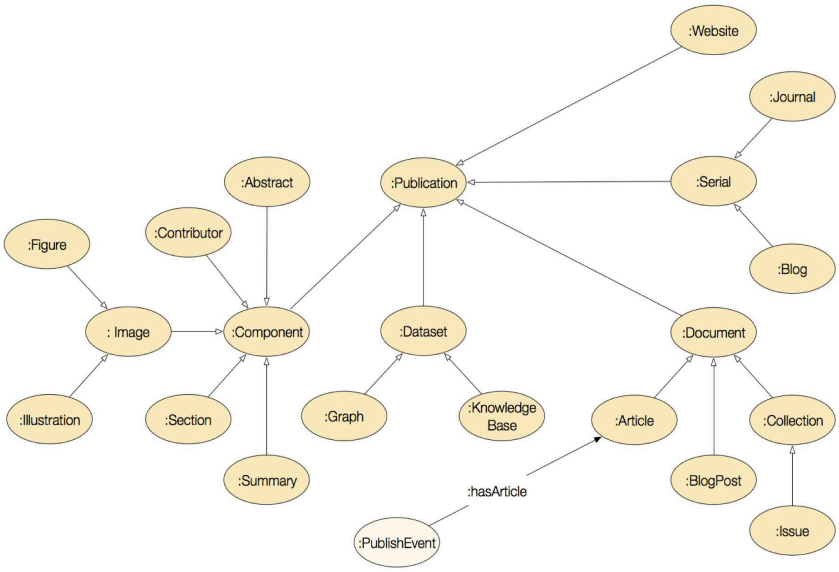

Fig. 4. The Publications branch of the Core Ontology.

and is organized either as a simple catalogue (i.e. a controlled vocabulary) or as a hierarchy (i.e. a taxonomy). Terms within a concept scheme are dually classed using one of our Core Ontology classes and the `skos:Concept` class. The terms are related via a `skos:inScheme` property to a SKOS concept scheme using the `skos:ConceptScheme` class. The SKOS concept scheme itself is additionally categorized as an OWL ontology. Additional SKOS properties are used for labelling (e.g. `skos:prefLabel`) and for relating (e.g. `skos:broader`) the terms.

There are nine domain models currently available as listed in Table 1.

2.3 Instance Datasets

The datasets referred to here cover the documents-based metadata and are not organized into any obvious model structure. Instead these datasets are being published as simple instance data and packaged by class type. The two main datasets we are publishing[6] are the following:

Articles A full inventory of all *articles* published by MSE from 1845 until 2015. In total, there are around 25 million records (for 1.2 million *articles*) containing detailed metadata such as title, authors, etc.

[6] In the earlier releases from 2012 we also produced a *citations* dataset comprising approximately 218 million records (for 9.3 million *citations*). This is currently available on the site as historical data and has not been upgraded to use our new data models and naming conventions.

Contributors A dataset comprising more than 11 million records (for 2.7 million *contributors*) containing the *contributors* to (i.e. authors of) the publications listed in the *articles* dataset described above. Note than this data is structured (i.e. fields such as given name and family name are broken out) and ordered (i.e. authorship order) but it has not been disambiguated (hence more than one *contributor* instance may refer to the same person, e.g. because of alternate spellings, etc.).

3 Data Mappings and Linksets

In order to make our dataset more interoperable and usable by the linked data community, we have begun to link our datasets to external datasets. We discuss these in turn.

Core Ontology For the core model we are mapping our classes and properties using `owl:equivalentClass` and `owl:equivalentProperty`, respectively, to over a dozen external schemas such as BIBO, FABIO, FOAF, etc. These mappings were created in two phases: first, a list of mapping candidates were extracted automatically into spreadsheets using the open-source OntoSPy [6] tool; second, we manually combed through these lists and selected those mapping pairs that made most sense. As regards the choice of ontologies to map to, we tried to select widely known models which are relevant to both the scholarly and the publishing communities. In future iterations we hope to refine these choices based on feedback from users and interested parties.

Domain Ontologies We are now mapping many of our domain models to DBpedia [7] and to Wikidata [8] – and for documentary purposes we are providing links to Wikipedia. We are also mapping our subjects to Bio2RDF [9] and to the MeSH RDF dataset from NLM (currently in beta) [10]. For our domain models we are making use of SKOS properties for linking, e.g. `skos:broadMatch`, `skos:closeMatch`, `skos:exactMatch`, and `skos:relatedMatch`. As above, we bootstrapped these mappings using automatic methods, then worked with domain experts in order to correct or refine the semantics of the links being created. See Fig. 5 for examples of our current mapping coverages.

Instance Datasets We are also linking the *articles* dataset to other RDF datasets. In particular, we have discovered over 51,000 links in Wikipedia articles (and hence DBpedia resource URIs) which reference our dataset. We extracted these references using the Wikipedia API and encoded them using the `cito:isCitedBy` property.

4 Data Publishing

Our models are maintained natively in RDF as Turtle files within GitHub repositories. The models are currently curated by hand with some lightweight valida-

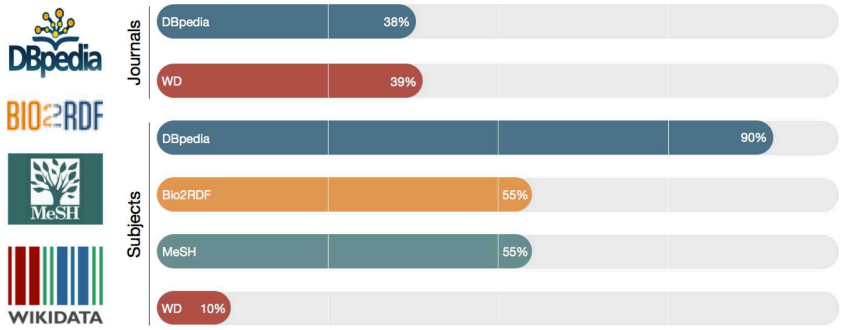

Fig. 5. Mapping coverages for domain models to common external datasets.

tions of data elements applied during the build process. By contrast, our datasets are sourced from master XML documents using Scala code for extraction of elements. In both processes we build in-memory RDF models using Apache Jena which ensures that the RDF models are valid.

Our ontologies (core and domain models) and datasets include data elements which can be shared with customers as well as additional data elements which are specific to our data production workflows and other data elements which are still undergoing testing. In order to select those data elements for sharing we have devised a generic mechanism based on knowledge-bases and contracts.

First a note about our data organization. All our RDF data is typed and is managed within multiple named graphs corresponding to RDF type. For example, all instances of `:Article` are managed within an *articles* graph, all instances of `:Abstract` are managed within an *abstracts* graph, etc. This data partitioning scheme allows us to annotate and generally admin the various data subsets and is also used in controlling data exports.

In order to serve different 'views' over the data for different customers we define a `:KnowledgeBase` class, and maintain instances within a *knowledge-bases* domain model. One of these instances is the knowledge-base for the public data we are sharing through the ontologies portal.

We also define a `:hasContract` property for the `:KnowledgeBase` class which aggregates a set of data bindings (using a `:hasBinding` property) that separately list the allowed RDF predicates (using an `:allowsPredicate` property) for various OWL terms: ontology, class, property, instance. See Fig. 6 for an example of the contract used to generate the *article-types* domain model.

```
knowledge-bases:public
  ...
  npg:hasContract [
    rdfs:comment "Contract for ArticleTypes Ontology" ;
    npg:graph npgg:article-types ;
    npg:hasBinding [
      npg:onOntology article-types: ;
      npg:allowsPredicate
        dc:creator , dc:date , dc:publisher , dc:rights , dcterms:license ,
        npg:webpage , owl:imports , owl:versionInfo , rdf:type , rdfs:comment ,
        skos:definition , skos:prefLabel , skos:note ,
        vann:preferredNamespacePrefix , vann:preferredNamespaceUri
      ;
    ] , [
      npg:onInstanceOf npg:ArticleType ;
      npg:allowsPredicate
        npg:hasRoot , npg:isPrimaryArticleType ,
        npg:id , npg:isLeaf , npg:isRoot , npg:treeDepth ,
        rdf:type , rdfs:isDefinedBy , rdfs:seeAlso ,
        skos:broadMatch , skos:broader , skos:closeMatch ,
        skos:definition , skos:exactMatch , skos:inScheme , skos:narrower ,
        skos:prefLabel , skos:relatedMatch , skos:topConceptOf
      ;
    ] ;
  ] ;
  ...
```

Fig. 6. Example of a contract for the public knowledge-base.

Our publishing workflow provides us with the ability to run rules (SPARQL queries[7]) against our model data at build time so as to enrich the datasets and to validate the data. For the knowledge-bases we generate SPARQL queries from the contracts and store those as properties within the respective knowledge-base. Effectively we build up a query map for each knowledge-base which provides for each RDF type a query with the predicate filtering for the various OWL terms required: *core model* – ontology, class, property; *domain model* – ontology, instance; *instance dataset* – instance.

5 Best Practices

Working within a production enviroment we have developed certain priciples and techniques for managing the data more efficiently.

[7] Some of the queries make use of SPIN constructs [11], so we loosely refer to these as 'SPIN rules'. Note also that we are eagerly anticipating the development of SHACL [12] as a generic RDF data enrichment and validation language.

5.1 Principles

In order to create a data and information architecture which is easy to understand, scalable and consistent throughout all of its aspects, we identified some core principles:

Incremental formalization We started out with a relatively flat model and tested it against our use cases and system architecture adding additional structure as more precise requirements were made available. So for example although we make use of SPIN rules and some basic inference in the data enrichment phase, we have not yet really taken advantage of the various inference mechanisms that can be built on top of OWL.

Enterprise integration We have primarily focused on building a shared enterprise model, e.g. by getting the core classes and properties right and thus achieving some simple yet fundamental level of data integration.

Model coherence Although we do make some use of public vocabularies such as BIBO and FOAF, in general we decided to follow a minimal commitment to external vocabularies as that would let us retain more control over our model and also create a much more coherent ontology. This is mainly because currently our main driver is to support enterprise applications. In order to facilitate web-scale data integration we have whenever possible added mappings to other commonly used vocabularies.

5.2 Techniques

On a technical level we can also mention the following techniques:

Naming architecture As mentioned above we work within a general named graphs environment whereby every triple is assigned to a named graph corresponding to the triple subject's RDF type. So, all instances of class `:Journal`, say, will be assigned to the named graph `journals:` for *journals*.

We adhere to a well-defined naming policy for our RDF term names. Class and property names are generally taken from the `npg:` namespace, although for legacy reasons (and especially for properties) we are still making use of some external names. Instance names use a simple binomial form based on named graph and local ID.

RDF serializations We work with various serializations of RDF. Our domain models are managed in Turtle and are sourced from GitHub. These models are assembled and enriched in memory using Apache Jena models and then loaded as RDF/XML into a MarkLogic XML database for querying. Our documents data is extracted from the *article* XML and assembled and enriched in memory

using Apache Jena models and then injected back as RDF/XML sections into the *article* XML for querying. Our data exports are in Turtle for models, and N-Quads for datasets.

RDF annotations Our historic datasets included RDF annotations [13] to describe the RDF data we were publishing – i.e. provenance, license, and metrics. For this we made extensive use of the VoID [14] vocabulary. As we are beginning to resume our data publishing activities we will need to revisit and enhance our annotations. Specifically we have been looking at the HCLS Community Profile for dataset descriptions [15].

6 Conclusions

In this paper we have given a brief summary of the `nature.com` ontologies portal. We believe that this resource has the potential to become a major component of the LOD cloud, both because of the quality and richness of its data and also because of the ever-increasing number of links to other well-known ontologies and datasets. As a scholarly publications dataset it may even serve to function as something of a local hub for generic linked science datasets since it describes a part of the network of publications that discuss and interpret the research datasets.

Furthermore, we are making our data available under very permissive licenses: CC0 for the datasets and domain models, and CC-BY for the core model. We are keen to work with the linked data community and, more broadly, the scientific community, in order to increase the reusability and interestingness of our datasets.

Acknowledgments. We wish to thank Andrew Needham, Ontology Manager at MSE, for his careful review of this paper as well as his unstinting work in maintaining the key domain ontologies and contributing to the ontologies portal.

References

1. Donohoe, P., Sherman, J., Mistry, A.: The Long Road to JATS. In: Journal Article Tag Suite Conference (JATS-Con) Proceedings 2015 [Internet]. Bethesda (MD): National Center for Biotechnology Information (US); 2015. `http://www.ncbi.nlm.nih.gov/books/NBK279831/`
2. Hammond, T., Pasin, M.: Linked data experience at Macmillan: Building discovery services for scientific and scholarly content on top of a semantic data model. In: The 13th International Semantic Web Conference, 2014. `http://ceur-ws.org/Vol-1383/paper4.pdf`
3. Miles, A., Bechofer, S. (Eds.): SKOS Simple Knowledge Organization System Reference. W3C Recommendation, 18 August 2009. `http://www.w3.org/TR/skos-reference/`

4. W3C OWL Working Group, Eds.: OWL 2 Web Ontology Language Document Overview (Second Edition). W3C Recommendation, 11 December 2012. http://www.w3.org/TR/owl2-overview/
5. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.: The Description Logic Handbook: Theory, Implementation, Applications. Cambridge University Press, Cambridge, UK, 2003. ISBN 0-521-78176-0
6. OntoSPy – RDFLIb-based Python toolkit for inspecting ontologies on the Semantic Web. https://github.com/lambdamusic/OntoSPy
7. DBpedia. http://wiki.dbpedia.org/
8. Wikidata. https://www.wikidata.org/
9. Bio2RDF. http://bio2rdf.org/
10. National Library of Medicine: Medical Subject Headings (MeSH) RDF Linked Data (beta). http://id.nlm.nih.gov/mesh/
11. Knublauch, H., Hendler, J.A., Idehen, K.: SPIN – Overview and Motivation. W3C Member Submission, 22 February 2011. http://www.w3.org/Submission/spin-overview/
12. Knublauch, H. (Ed.): Shapes Constraint Language (SHACL). W3C Editor's Draft 03, July 2015. https://w3c.github.io/data-shapes/shacl/
13. Nuno Lopes, N., Zimmermann, A., Hogan, A., Lukacsy, G., Polleres, A., Straccia, U., Decker, S.: RDF Needs Annotations. http://www.w3.org/2009/12/rdf-ws/papers/ws09
14. Alexander, K., Cyganiak, R., Hausenblas, M., Zhao, J.: Describing Linked Datasets with the VoID Vocabulary. W3C Interest Group Note, 03 March 2011. http://www.w3.org/TR/void/
15. Gray, A.J.G., Baran, J., Marshall, M.S., Dumontier, M. (Eds.): Dataset Descriptions: HCLS Community Profile. W3C Interest Group Note, 14 May 2015. http://www.w3.org/TR/hcls-dataset/