User and Developer Interaction with Editable and Readable Ontologies

Aisha Blfgeh [1,2]* and Phillip Lord [1]

[1]School of Computing Science, Newcastle University, UK
[2]Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia

ABSTRACT

The process of building ontologies is a difficult task that involves collaboration between ontology developers and domain experts and requires an ongoing interaction between then. This collaboration is made more difficult, because they tend to use different tool sets, which can hamper this interaction. In this paper, we propose to decrease this distance between domain experts and ontology developers by creating more readable forms of ontologies, and further to enable editing in normal office environments.

Building on a programmatic ontology development environment, such as Tawny-OWL, we are now able to generate these readable/editable from the raw ontological source and its embedded comments. We have this translation to HTML for reading; this environment provides rich hyperlinking as well as active features such as hiding the source code in favour of comments. We are now working on translation to a Word document that also enables editing.

Taken together this should provide a significant new route for collaboration between the ontologist and domain specialist.

1 INTRODUCTION

Ontologies are wide-spread in the field of biology and biomedicine, as they facilitate the management of knowledge and the integration of information, as in the *Semantic* Web (Bermejo, 2007). Additionally, biological data are not only heterogeneous but also require complex domain knowledge to be dealt with (Stevens *et al.*, 2000). Therefore, ontologies are useful models for representing this complex knowledge that is potentially changing and are also widely used in biomedicine, examples being the GO (Gene Ontology) (Ashburner *et al.*, 2000), SNOMED (Systematized Nomenclature of Medicine) (IHTSDO, 2016).

However, building an ontology is a challenging task due to the use of languages with a sophisticated formalism (such as OWL), especially when combined with a complex domain such as biology or medicine. Normally ontologies are built as a collaboration between domain specialists who have the knowledge of the domain and ontology developers who know how to structure and represent the knowledge; they have to work together to construct a robust and accurate ontology. Often, community involvement during the process of building ontologies using meetings, focus groups and the like is very important (Mankovskii *et al.*, 2009), as in GO where biological community involvement is important for successful uptake (Bada *et al.*, 2004). In addition, Bult *et al.* (2011) state that the development of Protein Ontology requires wider range of involvement to include other users and developers of the associated ontologies (such as GO) to ensure consistent architecture of the ontology.

*To whom correspondence should be addressed: a.blfgeh1@newcastle.ac.uk or abelfaqeeh@kau.edu.sa

Biologists represent, manipulate and share their data in a wide-variety of tools such as Microsoft Excel spreadsheets and Word documents. Unfortunately, these environments are far removed from the formal structured representation of the ontology development environments with which the ontologists work to build ontologies. As a result of this difference in tools it is unclear how we can bridge the gap between the the two groups; this would be useful to facilitate the interaction between domain specialists and ontologists and help to make more convenient for both sides to read and/or manipulate the ontology.

Ontology development environments are designed to produce formal structured representation of any domain. Either using GUI software such as Protégé [1] or a textual programmatic environment such in Tawny-OWL (Lord, 2013). The next section describes these tools in more details.

2 BUILDING ONTOLOGIES

There are various tools for constructing and developing ontologies with a variety of user interfaces and environments. The most popular is Protégé which is an open-source tool that provides a user interface to develop and construct ontologies of any domain. It has been widely used for developing ontologies due to the variety of plug-ins and frameworks (Noy *et al.*, 2003). Protégé provides an easy interface for editing, visualisation and validation of ontologies as well as a useful tool for managing large ontologies (Horridge *et al.*, 2011).

Conversely, Tawny-OWL is a textual interface for developing ontologies in a fully programmatic manner (Warrender, 2015). This provides a convienient and readable syntax which can be edited directly using an IDE or text editor; in this style of ontology development, the ontologist ceases to mainpulate an OWL representation directly, and instead develops the ontology as programmatic source code. In contrast to developing ontologies in OWL, the ontologist can introduce new abstractions and syntax as they choose, whether for general use or specifically for a single ontology. An OWL version of the ontology can then be generated as required. It has been implemented in Clojure, which is a dialect of lisp and runs on the Java Virtual Machine (Lord, 2013). Like Protégé, it also wraps the OWL-API (Mankovskii *et al.*, 2009) which performs much of the actual work, including interaction with reasoners, serialisation and so forth.

Recently, we have developed tolAPC ontology using a new document-centric approach by including an Excel spreadsheet directly in the development pipeline. The spreadsheet contains all knowledge for the ontology which has been created and maintained by a biologist. Meanwhile, we design the ontology patterns using Tawny-OWL, then generate the axioms by extracting data from the spreadsheet using Clojure. Thus, Tawny-OWL contains the

[1] `http://protege.stanford.edu/`

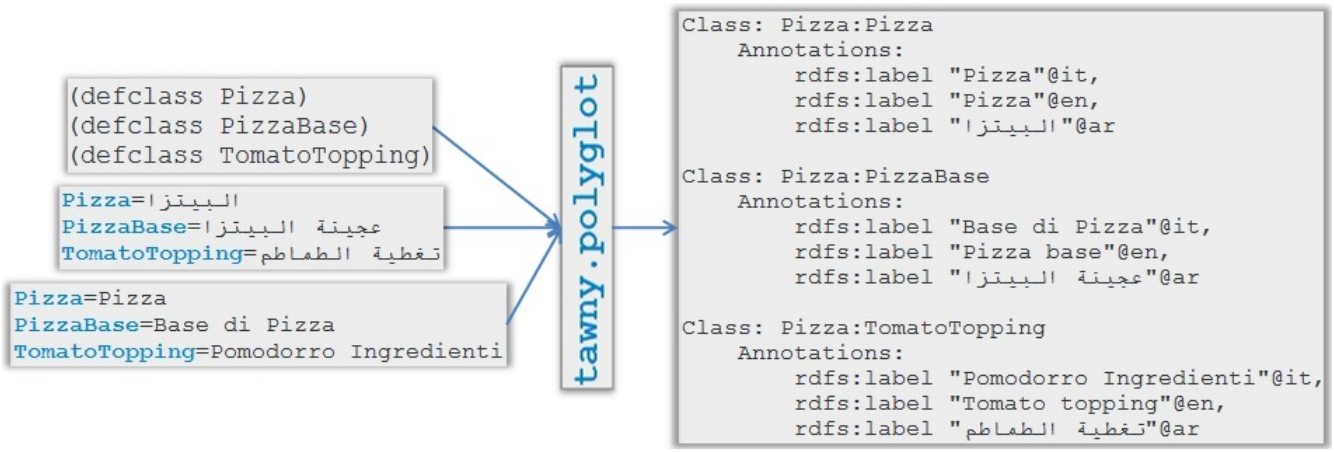


Fig. 1: Polyglot library in Tawny-OWL.

spreadsheet as a part of the source code; which can be freely updated and the ontology regenerated when needed. Hence, it remains as a part of the ontology development process (Blfgeh *et al.*, 2016).

In this approach, the Excel spreadsheet is totally developed by biologists; this has a significant advantage because it is a tool which they are familiar with and find convenient. However, we cannot ensure that the programmatic transformation of the values in the spreadsheet to the final ontology conforms with the domain specialists understanding, without the biologists reading and interacting with source code. Therefore, next we will discuss the probabilities of making this ontological source more readable by the specialists.

3 MULTILINGUAL ONTOLOGIES

The first and most obvious mechanism for increasing ontology readability is to enable users to read and write the ontology using their native language. Internationalisation technologies are widespread and enable support for multiple languages for applications with a graphical user interface.

We next consider how we can enable support for multiple languages in textual user interface such as Tawny-OWL, giving the ontologist the ability to use their own native language for all parts of the development process.

The first option is using polyglot library. This part of the system mimics a fairly standard technique for internationalisation of programmatic code; the ontology is developed with a set of programmatic labels which are then referenced in a language, or locale bundle with an appropriate translation. In the case of Tawny-OWL, this translation appears as `rdfs:label` annotations on the ontology entities (classes, properties etc). This overall process is shown in Figure 1, placing Italian and Arabic language translations onto the pizza ontology.

While this may enable internationalisation for users of the ontology, it does not change the English-centric editing environment. We would wish, instead, to internationalise the entire source code of the ontology. This will make the entire ontology more comprehensible and readable for all developers who communicate in Italian and/or Arabic. This is fully supported with a full conversion of the environment using the multilingual feature of Tawny-OWL as in Figure 2, which shows the English, Italian and Arabic version of the pizza ontology (Lord, 2012). The latter of these is a right-to-left alphabet, and we can use the IDE to change the direction that Tawny-OWL code is rendered in. This demonstrates the capability of Tawny-OWL to adapt with any language. The next language to be implemented will be French.

These multilingual environments are advantageous for being readable and comprehensible by users when using their own language. This still leaves us in a programming environment, which is an environment unlikely to be familiar or comfortable to the most domain users. Moreover the ontology lacks a narrative structure, which means that it cannot be read in a literate fashion. We consider how to enable this in the next section.

4 LITERATE ONTOLOGIES

The term literate programming was invented by (Knuth, 1992) where the program is treated as a piece of literature rather than a program. The main idea in this paradigm is to insert text along with code and the program will also be its own documentation. The intentionality here is that the program should become easier to understand and, conversely, that the documentation is less likely to become out-of-date, as it is maintained in the same place.

As Tawny-OWL is a fully programmatic environment, we can add comments freely, along with any additional mark-up that we wish. This enables us to produce different representation of the ontology.

We have previously discussed two examples of literate ontologies: the first is the Amino Acid Ontology, taken from a previous ICBO2015 tutorial about Tawny-OWL [2], while the second is a version of the Karyotype ontology (Lord and Warrender, 2015). In both cases, they have been produced using the Tawny-OWL source code, with markup in the comments being interpreted using a markup processing tool. Figure 3 shows a snippet from the literate ontology Amino Acids as a webpage. The result appears as a normal web page, with syntax highlighting for the source code.

[2] `http://homepages.cs.ncl.ac.uk/phillip.lord/take-wing/take_wing.html`

```
(defontology pizzaontology
  :iri "http://www.ncl.ac.uk/pizza"
  :prefix "piz:"
  :comment "An example ontology modelled on the
            Pizza tutorial ontology from Manchester
            University, written using the tawny-owl
            library"
  :versioninfo "Unreleased Version"
  :seealso "Manchester Version"
  )

(defaproperty myOpinion
  :subproperty owl-comment-property
  :label "My Opinion"
  :comment "Do I think this is a good pizza to eat?"
  )

(defclass Pizza
  :label "Pizza")
```

(a) English Pizza Ontology

```
(defontologia pizzaontologia
  :iri "http://www.ncl.ac.uk/pizza"
  :prefisso "pizza"
  :commento "Un esempio ontologico modellato
             su un tutorial di ontologia sulla
             Pizza dall'Università di Manchester,
             scritta utilizzando la libreria
             Tawny-OWL"
  :versione "Versione non pubblicata"
  :vedianche "versione di Manchester")

(defoproprietà miaOpinione
  :etichetta "la mia opinione"
  :commento "Penso che sia una buona pizza
             da mangiare?")

(defclasse Pizza
  :etichetta "pizza")
```

(b) Italian Pizza Ontology

```
(عرف-الأنتولوجيا البيتزا-العربية
:آي-آر-آي "http://:جامعة.نيوكاسل.بر/البيتزا"
:بادئة "بيتزا"
:تعليق "مثال لاستخدام المصطلحات العربية
        لتعريف أنتولوجيا البيتزا العربية"
:الإصدار "الإصدار الأول"
:انظر-أيضا "إصدار مانشستر")

(عرف-خاصية-التدوين رأيي
:علامة "الرأي العام"
:تعليق "هل يعتقد المتذوق أن هذا النوع من
البيتزا جيد ويستحق أن يؤكل")

(عرف-الصنف البيتزا
:علامة "البيتزا")
```

(c) Arabic Pizza Ontology

Fig. 2: Multilingual Pizza ontology

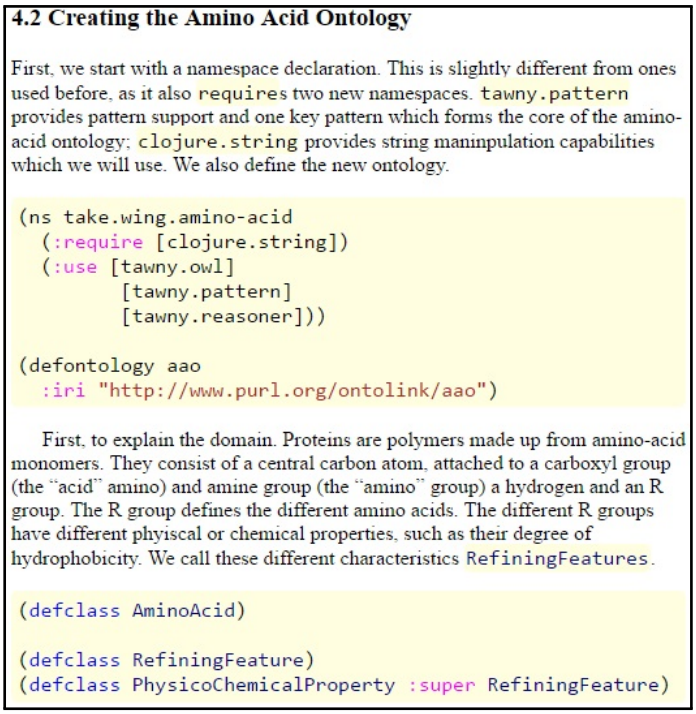
Literate ontologies can be represented in different forms; using the various techniques for converting the markup text into different formalisms; webpages for example. Representing the ontology as an HTML webpage gives us the ability to navigate and browse the documentation either in order (section by section) or with a navigation facility (jumping between sections). It is also possible to hide or expose the "source" sections, leaving the reader to see just the documentation as appropriate. From the developer perspective, while the reader may still not be able to see the axiomatization in this way, the comments that they have checked are embedded directly next to the code which is an interpretation of them.

It is interesting to enable specialists to read and navigate through the ontology and its documentation. However, with HTML there are no editing facilities to modify and update the ontology. Therefore, rather than using HTML, we have also investigated the possibility to turn the whole ontology into a Word document, an environment which can also be modified, changed or updated. Now biologists and domain specialists are placed in an environment in which they can freely provide feedback on an existing ontology simply by interacting with Word documents.

5 DISCUSSION

In this paper, we have described our approach to the translation of ontologies into a form that domain users can interact with more naturally.

We have shown that it is possible to translate a textual environment like Tawny-OWL into another human language, or indeed a different script, including right-to-left text. To our knowledge, this is the first ontology editing environment with such textual and syntactic flexibility. Despite the fact that the multilingual ontologies approach is less relevant for scientific ontologies, it is already applied by some means in some cases of terminologies. For example, the use of `some` and `only` in Tawny-OWL rather than using *universal* and *existential* notations which implies the agreement for using alternative language for ontology development.



4.2 Creating the Amino Acid Ontology

First, we start with a namespace declaration. This is slightly different from ones used before, as it also `requires` two new namespaces. `tawny.pattern` provides pattern support and one key pattern which forms the core of the amino-acid ontology; `clojure.string` provides string maninpulation capabilities which we will use. We also define the new ontology.

```
(ns take.wing.amino-acid
  (:require [clojure.string])
  (:use [tawny.owl]
        [tawny.pattern]
        [tawny.reasoner]))

(defontology aao
  :iri "http://www.purl.org/ontolink/aao")
```

First, to explain the domain. Proteins are polymers made up from amino-acid monomers. They consist of a central carbon atom, attached to a carboxyl group (the "acid" amino) and amine group (the "amino" group) a hydrogen and an R group. The R group defines the different amino acids. The different R groups have different phyiscal or chemical properties, such as their degree of hydrophobicity. We call these different characteristics `RefiningFeatures`.

```
(defclass AminoAcid)

(defclass RefiningFeature)
(defclass PhysicoChemicalProperty :super RefiningFeature)
```

Fig. 3: Literate Amino Acid ontology in HTML representation.

Further than this, however, we also translate the ontological source code into alternative visualisations such as HTML and Word documents which map directly back to the source, but which can differ from it: for instance, by enabling hyperlinks, adding section links and hide source code in favour of commentary. Especially with a Word document, this should enable a novel mechanism for interacting with an ontology: users can see and edit comments, with change tracking switched on, and use this as mechanism for feeding back to the ontology developer.

Using this approach, of course, only enables us to visualise ontologies developed using Tawny-OWL. While a migration path is provided (Warrender, 2015), a whole-sale switch to Tawny-OWL is not effort-free. We note, however, that many ontologies are developed partly in Protégé and partly using OWL generated from other sources; a secondarly migratory path would be to use Tawny-OWL for these sections.

We still need to evaluate this kind of interaction rigorously. For this, we are proposing a focus group test which will be include specialists participants to read the document of the ontology and provide their opinion about it and whether they prefer to update any terminologies according to their expertise.

We are not proposing that Word documents will be directly used by domain specialists for editing ontologies. We expect that an ontologist will be involved with incorporating changes suggested back to the domain user; in this sense, we are using a Word document as an *intermediate representation* (Rector *et al.*, 2001). Our hope is that the reviewing features of Word should, however, enable us to provide a rich environment to support the ontologist in this process. Taken together, these should provide us with a significantly enhance process for the knowledge capture, ontology development and refinement from the process that we currently have.

ACKNOWLEDGEMENTS

Thanks to Newcastle university for supporting this research. Also, thanks to King Abdulaziz University, Jeddah, Saudi Arabia for funding and supporting the study.

REFERENCES

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., and Sherlock, G. (2000). Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nat Genet*, **25**.

Bada, M., Stevens, R., Goble, C., Gil, Y., Ashburner, M., Blake, J., Cherry, M., Harris, M., and Lewis, S. (2004). A Short Study on the Success of Gene Ontology. *Web Semantics: Science, Services and Agents on the World Wide Web*, **1**(2), 235–240.

Bermejo, J. (2007). A Simplified Guide to Create an Ontology. *ASLab.org*. `http://tierra.aslab.upm.es/documents/controlled/ASLAB-R-2007-004.pdf`.

Blfgeh, A., Warrender, J. D., Hilkens, C. M. U., and Lord, P. (2016). A document-centric approach for developing the tolapc ontology. In F. Loebe, M. Boeker, H. Herre, L. Jansen, and D. Schober, editors, *Proceedings of the 7th Workshop on Ontologies and Data in Life Sciences, ODLS 2016, organized by the GI Workgroup Ontologies in Biomedicine and Life Sciences (OBML), Halle (Saale), Germany, September 29-30, 2016.*, volume 1692 of *CEUR Workshop Proceedings*, pages 1–6. CEUR-WS.org. `http://ceur-ws.org/Vol-1692/paperB.pdf`.

Bult, C. J., Drabkin, H. J., Evsikov, A., Natale, D., Arighi, C., Roberts, N., Ruttenberg, A., D'Eustachio, P., Smith, B., Blake, J. A., and Wu, C. (2011). The representation of protein complexes in the protein ontology (pro). *BMC Bioinformatics*, **12**(1), 371.

Horridge, M., Knublauch, H., Rector, A., Stevens, R., Wroe, C., Jupp, S., Moulton, G., Drummond, N., and Brandt, S. (2011). A Practical

Guide To Building OWL Ontologies Using Protégé 4 and CO-ODE Tools Edition 1.3. http://dio.freelabs.net/downloads/ProtegeOWLTutorialP4{_}v1{_}3.pdf.

IHTSDO (2016). International health terminology standards development organisation.

Knuth, D. (1992). *Literate Programming*. Center for the Study of Language and Information Publication Lecture Notes. Cambridge University Press.

Lord, P. (2012). Tawny-owl pizza. https://github.com/phillord/tawny-pizza.

Lord, P. (2013). The Semantic Web takes Wing: Programming Ontologies with Tawny-OWL. *CoRR*, **abs/1303.0**. http://arxiv.org/abs/1303.0213.

Lord, P. and Warrender, J. D. (2015). A highly literate approach to ontology building. **abs/1512.04250**.

Mankovskii, S., Gogolla, M., Urban, S. D., Dietrich, S. W., Urban, S. D., Dietrich, S. W., Yang, M.-H., Dobbie, G., Ling, T. W., Halpin, T., Kemme, B., Schweikardt, N., Abelló, A., Romero, O., Jimenez-Peris, R., Stevens, R., Lord, P., Gruber, T., Leenheer, P. D., Gal, A., Bechhofer, S., Paton, N. W., Li, C., Buchmann, A., Hardavellas, N., Pandis, I., Liu, B., Shapiro, M., Bellatreche, L., Gray, P. M. D., Aalst, W. M. P., Palmer, N., Palmer, N., Risch, T., Galuba, W., Girdzijauskas, S., and Bechhofer, S. (2009). OWL: Web Ontology Language. In *Encyclopedia of Database Systems*, pages 2008–2009. Springer US, Boston, MA. http://www.springerlink.com/index/10.1007/978-0-387-39940-9{_}1073.

Noy, N. F., Crubézy, M., Fergerson, R. W., Knublauch, H., Tu, S. W., Vendetti, J., and Musen, M. A. (2003). Protégé-2000: An Open-Source Ontology-Development and Knowledge-Acquisition Environment. *AMIA Annu Symp Proc*, **953**, 953. http://protege.stanford.edu.

Rector, A. L., Wroe, C., Rogers, J., and Roberts, A. (2001). Untangling taxonomies and relationships: Personal and practical problems in loosely coupled development of large ontologies. In *Proceedings of the 1st International Conference on Knowledge Capture*, K-CAP '01, pages 139–146, New York, NY, USA. ACM.

Stevens, R., Goble, C. a., and Bechhofer, S. (2000). Ontology-based knowledge representation for bioinformatics. *Briefings in bioinformatics*, **1**(4), 398–414.

Warrender, J. D. (2015). *The Consistent Representation of Scientific Knowledge: Investigations into the Ontology of Karyotypes and Mitochondria*. Ph.D. thesis, Newcastle University. https://theses.ncl.ac.uk/dspace/bitstream/10443/2910/1/Warrender,J2015.pdf.