Developing an Ontology from the Application Up

James Malone[1], Tomasz Adamusiak[1], Ele Holloway[1], Misha Kapushesky[1], Helen Parkinson[1]

[1] European Bioinformatics Institute, Cambridge,
CB10 1SD, UK
{malone, tomasz, ele, ostolop, parkinson}@ebi.ac.uk

Abstract. The biomedical ontology community is producing ontologies which represent biological knowledge and with a bias towards a realist perspective due to the use of the upper level ontology BFO [1]. While these ontologies are useful representations of the knowledge space they are not always applicable in the context of typical biomedical use cases. In this paper we describe our use of reference ontologies in the development of the Experimental Factor Ontology (EFO) and its application to public gene expression data. Finally, we present guidelines for rapid development and deployment of an application ontology.

Keywords: Application ontology, gene expression ontology, EFO, GXA

1 Introduction

There are a growing number of potential reference ontologies developed by the community, many of which are made available under the umbrella of the OBO Foundry [2]. These ontologies are engineered with the intention of providing canonical models of a delineated part of biology, such that the models are interoperable, non-overlapping and consistent. While this is a laudable aim, there are outstanding obstacles to using these reference ontologies in an application such as the Gene Expression Atlas (GXA) (www.ebi.ac.uk/gxa). Many such efforts, presented as potential reference ontologies (within and without the OBO Foundry), contain overlapping content, for example there are multiple ontologies that cover aspects of anatomy, disease and species information. Furthermore, these ontologies are not fully interoperable in the true computational ontological sense. Presently, the ontologies do not all use a single upper ontology and a single set of aligned relations. This makes importing these resources and reasoning in an OWL framework an unfeasible prospect. This is of particular importance when there is a need to build classes which are 'cross-products', i.e. classes which have a composition of a combination of other classes, e.g. 'bone cancer' from 'bone' and 'cancer' classes. Clearly, this would not be possible if the imported ontologies produced inconsistencies with the model.

We describe here our approach to building the Experimental Factor Ontology (EFO), an application ontology built in OWL, which we have developed for use with the GXA and the ArrayExpress Archive. We outline our use cases, outline the tools we have developed and illustrate the use of EFO within the GXA.

Proceedings of OWL: Experiences and Directions 2009 (OWLED 2009),
Rinke Hoekstra and Peter F. Patel-Schneider, editors. http://www.webont.org/owled/2009

2 Methods

The GXA is a newly developed resource based at the European Bioinformatics Institute. It provides a summary view of gene expression across various experimental conditions (experimental factors) using curated data from the ArrayExpress database [3]. One of the key aims of the GXA was to enable more powerful, semantically meaningful querying across the experimental data. In order to answer queries such as "*which genes are over-expressed in cancer samples from mammalian brain*" we require an ontological framework that is able to provide information from multiple source ontologies since the data covers such a wide range of domains. These include experimental and biological processes, cell lines, cell types, disease, anatomical information, organism and strain information, chemical roles, and array designs and require cross products between ontologies, concept mapping and an ability to maintain the application ontology once generated.

As an application ontology, EFO is designed and tested to use cases, including:

1. Data annotation – the annotation of transcriptomics data in ArrayExpress
2. Query support – e.g. query for all cell line data that is derived from epithelial tissue and are associated with cancer
3. Data visualization – presenting an ontology tree to the user to show which classes have associated data
4. Data integration – both across experiments in ArrayExpress and externally
5. Data summarization – the ability to analyse and compare samples given common conditions of interest

From these use cases we generate a set of competency questions, such as "which organism parts are part of brain" and build classes and relations into the ontology until we can resolve them all. The methodology is summarised as follows:

1. Extract data annotations from the GXA and target most frequently occurring.
2. Use the query use cases obtained from analysis of query logs to build an appropriate hierarchy
3. Identify reference ontologies relevant to an EFO category based on annotation use cases and perform mapping between annotations and reference ontologies using the Metaphone and Double Metaphone phonetic matching algorithm [4]. This produces a list of candidate ontology class matches.
4. Manually validate candidate matches, curate and include into EFO hierarchy
5. Refine structure to provide an intuitive hierarchy with user friendly labels
6. Add restrictions e.g. associate cell lines with cell types and tissues of origin

For step 3 above, we chose not to discriminate between ontologies that overlap in their content since it is a non-trivial exercise to decide which represents the definitive source. Additionally we add mappings to multiple ontologies, as different communities use different ontologies and we do not wish to exclude users by insisting on use of a single 'authoritative ontology'. Instead, we base the decision simply on whether the ontology appears to offer content that provides coverage for our use cases. Clearly, there is an overhead, both to mapping to multiple ontologies and also in keeping this process up to date as those external ontologies change over time. To facilitate the development of the ontology, we have developed several tools to expedite the process[1].

2.1 EFO Ontology Tools

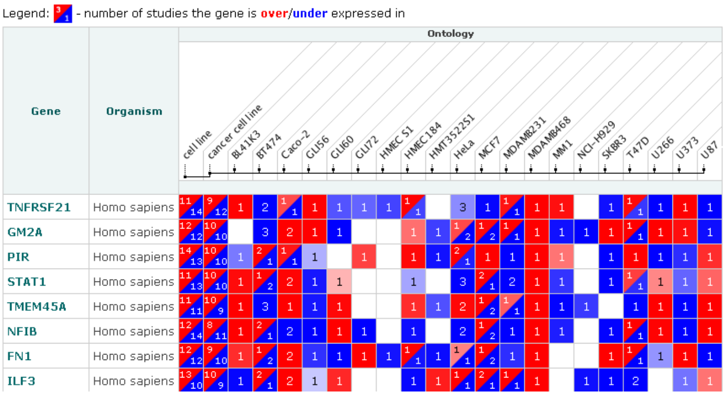
Mappings to external resources are maintained as an ID from external resources into a definition_citation annotation property. In this way, we are able to map equivalent classes from EFO in to multiple other ontologies, e.g. neoplasm in EFO maps to neoplasm in NCI Thesaurus hence the definition_citation property has the value NCI thesaurus:C3262 to encapsulate this. This way it is easy to map data annotated with EFO to data annotated with the many other bioontology efforts. The process of adding these mappings has been automated and wraps around BioPortal REST services to automatically follow all definition_citation properties from EFO into relevant ontologies, pulling in synonyms and definitions. These are then time and resource stamped in EFO using inline accessedResource and accessDate tags for auditing purposes and to warn of changes in external resources.

Since ontologies develop rapidly, there is an overhead to developing an ontology that references multiple such external ontologies. To counter this, we have also developed a tool that enables the frequent checking of external ontologies to determine if newer versions include changes which might affect EFO. The tool works in three steps. Firstly, it accesses the latest version of an ontology by loading from a static location and attempts to validate the ontology. Secondly, it compares to the ontology that EFO presently maps to. A ‘change’ is considered to be (i) a new class (ii) a removed class, (iii) a new axiom, (iv) a removed axiom. Presently the tool does not detect annotation property changes between versions, however this is scheduled for the next tool release. Finally, the tool flags any changes which affect EFO, i.e. any changes in an external ontology class which is mapped to in EFO.

3 Discussion

EFO has become an integral component of the GXA. Figure 1 illustrates a query in the GXA which uses EFO classes. Here, the query is for a defined class (in OWL) ‘cancer cell line’. This class has the necessary condition of cell line and a necessary and sufficient condition of ‘`bearer_of some cancer`’, i.e. any cell lines which bear the disease cancer. The relations we create in OWL also enable us to offer a range of general level queries which was also not previously possible. For instance, a query for ‘digestive system components’ or all ‘bacterial diseases’ would not fetch all classes which are related via ‘`part_of`’ relation and an ‘`is_a`’ relation, respectively. The use of complex OWL axioms allows us to provide the desired rich queries, whilst the application hides the specific details from the user.

[1] All tools are available from our public SVN site http://sourceforge.net/projects/efo/



Legend: 🟥3 🟦1 - number of studies the gene is over/under expressed in

Gene	Organism	Ontology																					
		cell line	cancer cell line	BL41K3	BT474	Caco-2	GLI56	GLI60	GLI72	HMEC S1	HMEC184	HMT3522S1	HeLa	MCF7	MDAMB231	MDAMB468	MM1	NCI-H929	SKBR3	T47D	U266	U373	U87
TNFRSF21	Homo sapiens	🟥11 🟦14	🟥9 🟦12	🟥1	🟦2	🟥1 🟦1	🟥1	🟦1	🟦1	🟦1	🟥1 🟦1		🟦3	🟦1	🟥1 🟦1	🟥1	🟥1		🟦1	🟥1 🟦1	🟦1	🟥1	🟦1
GM2A	Homo sapiens	🟥12 🟦12	🟥10 🟦10		🟦3	🟥2	🟥1	🟦1			🟥1	🟦1	🟥1 🟦2	🟥1 🟦2	🟥1 🟦1	🟥1	🟦1	🟦1	🟥1	🟥1 🟦1	🟥1	🟥1	🟦1
PIR	Homo sapiens	🟥14 🟦13	🟥10 🟦10	🟦1	🟥2 🟦1	🟥1 🟦1	🟦1		🟥1		🟥1	🟦1	🟥2 🟦1	🟥1	🟦1	🟥1	🟥1		🟦1	🟥1	🟦1	🟦1	🟥1
STAT1	Homo sapiens	🟥11 🟦13	🟥10 🟦10	🟥1	🟥1 🟦2	🟥2	🟦1	🟥1			🟦1		🟦3	🟥2 🟦1	🟦2	🟥1	🟦1		🟦1	🟥1 🟦1	🟥1	🟦1	🟥1
TMEM45A	Homo sapiens	🟥11 🟦11	🟥10 🟦9	🟥1	🟦3	🟥1	🟦1	🟥1			🟥1	🟦1	🟥2	🟥1 🟦2	🟥1 🟦1	🟥1	🟦1	🟦1		🟦1	🟥1	🟦1	🟥1
NFIB	Homo sapiens	🟥12 🟦14	🟥8 🟦11	🟥1	🟥2 🟦1	🟦2	🟦1	🟥1	🟦1		🟦1		🟦2	🟥1 🟦2	🟦1	🟥1	🟦1		🟦1	🟥1 🟦1	🟥1	🟦1	🟥1
FN1	Homo sapiens	🟥12 🟦12	🟥9 🟦10	🟥1	🟥1 🟦2	🟥2	🟦1	🟦1	🟥1	🟦1	🟥1 🟦1	🟦1	🟥1 🟦1	🟥1 🟦2	🟦1	🟥1			🟥1	🟥1 🟦1	🟦1	🟥1	🟦1
ILF3	Homo sapiens	🟥13 🟦10	🟥10 🟦9	🟦1	🟥2 🟦1	🟥2	🟦1	🟥1			🟦1	🟥1	🟥1 🟦1	🟥2 🟦1	🟥1 🟦1	🟥1		🟦1	🟦1	🟦2		🟦1	🟥1

Fig. 1. Query in the GXA using EFO defined class ‘cancer cell line’.

During this work, we have identified a series of use cases based on a very large public dataset, assessed the quality and nature of the annotations and produced an application ontology which can be used for text mining, by annotators, for queries and for data mining. We offer the following 10 guidelines for application ontology developers:

1. Re-use whenever possible, do not re-invent
2. Use case development is key, these should be documented and testable
3. Upper level ontologies are useful but not essential, they can help organize the ontology, but there is a trade off with complexity
4. Upper level ontology classes should not be visible in a UI for biological users
5. OBO and OWL formats is desirable by users in the biomedical domain
6. Cross products are required, as these are being generated slowly by the community generating these for defined use cases is needed
7. Use of a normalization methodology helps to control the inferences made
8. Deployment of an ontology is an excellent QC tool, curators and users quickly spot errors in the ontology in the context of data that ontology developers miss
9. Be a good citizen, where errors or omissions are present in other ontologies, report these and request new classes
10. Adhere to good community practice, e.g. OBO foundry principles if possible

References

1. Grenon, P., Smith, B., Goldberg, L.: Biodynamic ontology: applying BFO in the biomedical domain. Studies in health technology and informatics, 102, pp. 20-38 (2004)
2. Smith, B., Ashburner, M., et al.: The OBO Foundry: coordinated evolution of ontologies to support biomedical data integration. Nature Biotechnology 25(11), pp. 1251-1255 (2007)
3. Parkinson, H., Kapushesky, M., Kolesnikov, et al.: ArrayExpress update—from an archive of functional genomics experiments to the atlas of gene expression, Nucleic Acids Res., 37(Database issue):D868-72 (2009)
4. Phillips, L.: The Double Metaphone Search Algorithm. C/C++ Users Journal. (2000)