WHITE PAPER

Best Practices of Ontology Development

October 25, 2016

Prepared by:

Ron Rudnicki (CUBRC, Inc.)
Barry Smith, PhD (University at Buffalo)
Tanya Malyuta, PhD (New York College of Technology)
COL William Mandrick, PhD (Institute for Military Support to Governance)

Table of Contents

Objective

Ontologies are among the building blocks of semantic web strategies for counteracting the problem of **data stovepipes**. A problem that results from the creation of data-models in heterogeneous, uncoordinated ways and that leads to a failure of data integration and reuse. Ontologies, when properly designed and developed, can ensure consistency and persistence in the usage of terms – thus also persistence of meaning – and thereby counteract these consequences of the development and frequent changes in the databases and software used to store and manage data.

The World Wide Web Consortium (W3C) has sanctioned languages for ontology formulation (http://www.w3.org/2001/sw/Specs), of which the most important is the Web Ontology Language (OWL). These have spawned in their turn powerful open-source software (http://www.w3.org/2001/sw/wiki/Tools) for developing and reasoning with ontologies and for querying data stores aligned to ontologies. Unfortunately, the resultant popularity of semantic technology has itself led to a situation where ontologies are now being created in heterogeneous, uncoordinated ways, thereby leading to a new problem of semantic stovepipes, and thus to a new failure of data integration and reuse. The best practices here described counteract this tendency by providing a set of principles and rules to lead to the development of ontologies in a consistent, non-redundant fashion.

Constraints

We begin by describing the set of tools and the modeling methodology for developing ontologies. We classify these as constraints since they are external to the design and development processes yet they limit those processes in significant ways.

Web Ontology Language (OWL)

We follow the W3C recommendation that ontologies be published for exchange as OWL files (more precisely: as files using the OWL 2 Web Ontology Language) in RDF/XML syntax (https://www.w3.org/TR/owl-conformance/). Requiring the exchange of ontologies in this syntax ensures that tools will be able to act on them without transformation (a result of RDF/XML being the syntax that any tool must support in order to be OWL compliant). Other OWL syntaxes are available for purposes other than exchange. (e.g. OWL/XML, Turtle, or Manchester). For example, Turtle and Manchester syntaxes foster readability.

Our following of the W3C recommendation regarding the use of OWL is not an endorsement of that language as being ideally suited to the task of publishing ontologies. Rather it is a recognition OWL is the current standard and the ecosystem of tools surrounding it makes it the current best choice. As technology evolves, other languages may replace OWL's role as the standard. For example, ISO has sanctioned an additional language, Common Logic (CL) (http://www.iso.org/iso/catalogue_detail.htm?csnumber=39175), and its use for ontology formulation is increasing (http://stl.mie.utoronto.ca/colore/ontologies.html). CL has greater expressive power than OWL, but lacks certain computational features and tooling which make it less attractive than OWL.

Basic Formal Ontology as Top-Level Ontology

The *level* of an ontology is determined by the level of generality of the types in reality that it represents. Object, for example, is a very general type; Living Thing and Person are less general types, and types such as Blue Eye Color are less general still.

As will be explained more fully in the section on design principles below, we recommend a tiered architecture of ontologies starting with a single top-level ontology that forms the base of all other ontologies. The purpose of the top-level ontology is to provide a high-level, domain-neutral representation of distinctions such as that between objects and events, and between objects and their attributes (qualities, roles, and so forth).

We recommend Basic Formal Ontology 2.0 (BFO) as the top-level ontology (Arp, Smith, & Spear, 2015). BFO is a small, highly abstract, upper level ontology designed for use *under the hood*. Its role is to provide a framework that can serve as a common starting point for representing types that are more specific in order to ensure consistent ontology development at lower levels in a way that maximizes the degree of interoperability between ontologies developed by different collaborating groups.

One reason for the success of BFO is that it is a strict top-level ontology. As such, it is domain neutral and does not contain its own representations of physical, chemical, biological, psychological, social or other types of entities that would properly fall within mid- or lower-level domains. BFO is correspondingly very small, with a narrowly focused task: that of providing an upper ontology that supports the integration of multiple heterogeneous domain ontology plug and play modules at lower levels. Figure 1 depicts a set of modular BFO-based domain ontologies being created to describe all aspects of plant life (http://wiki.plantontology.org/index.php/The_cROP_(Common_Reference_Ontologies_for_Plants)_Initiative).

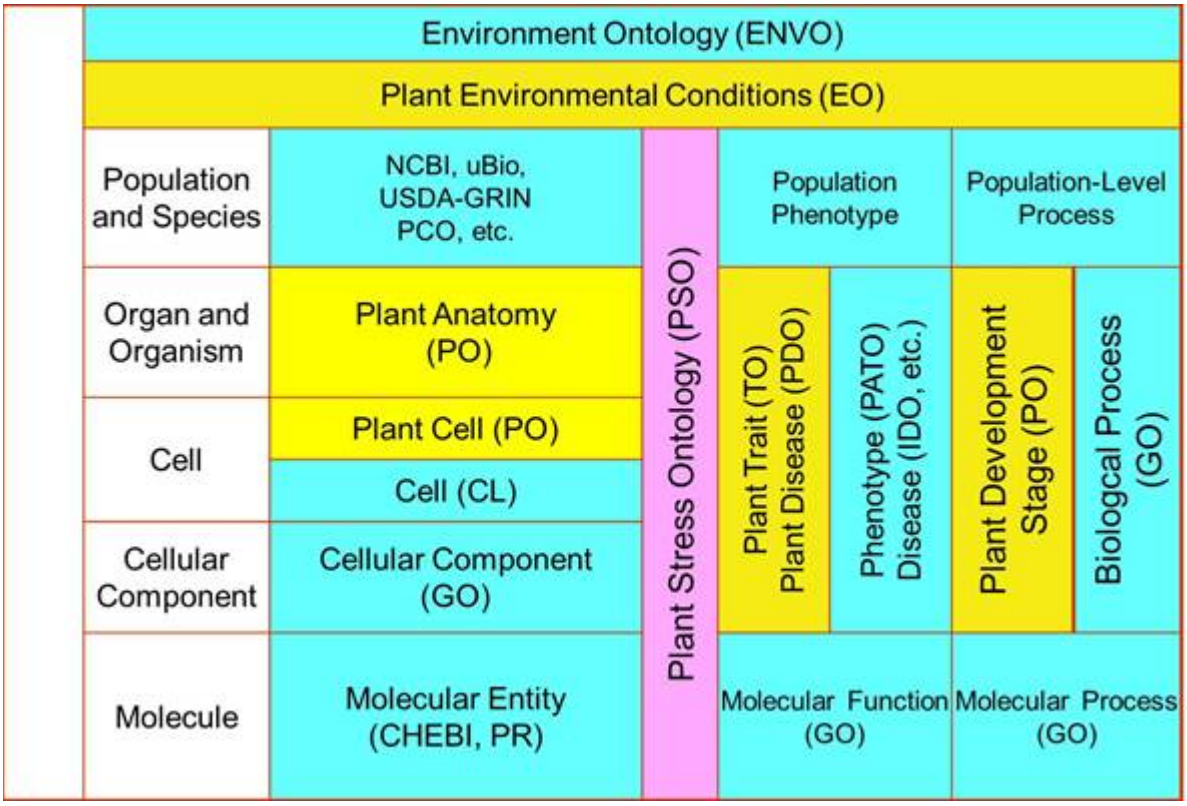

Figure 1: The CROP Common Reference Ontologies for Plants suite of ontology modules

Ontological Realism

To maximize both utility and stability, the modeling process of ontology development should rest on what in (Smith & Ceusters, 2010) is called 'ontological realism', which amounts to the idea that an ontology should be analogous not to a *data model*, but rather to a *reality model.* Under this constraint, ontologies are representations not of the *data* to be integrated, but rather of the entities to which those data refer. Some ontologies will indeed need to contain terms referring to data items – for example to types of images, or types of email or of other text documents. By following the practice of ontological realism these ontologies will treat these data items as entities in reality in their own right.

Developing under the constraint of ontological realism positions the resulting ontologies as the best candidates for serving as common models of all the data sources within a complex information ecosystem. Realist ontologies serve as a proxy for some portion of the world, employing as far as possible consensus expressions (from natural language, including scientific vocabularies). Data sources are descriptions of the world but they add a layer of perspective in order to serve the application specific needs of their users. Realist ontologies enable enterprise-wide data integration by removing this layer of perspective from each of the several data sources involved and thereby arrive at a depiction of the domain that can serve as benchmark for their integration.

Adopting the method of ontological realism also contributes to improved governance. In order for an ontology to adhere to the realist approach, all of its assertions must be true of the world; if they are not

they must be corrected in order to be so. Using the world as a basis for determining the correctness of an ontology provides a clear methodology for adjudicating disputes and continually improving an ontology's content; this is an important benefit for long-term management.

Design Principles

The constraints described above, that the RDF/XML syntax of OWL should be used for the exchange of ontologies, that they use BFO as their top-level ontology, and that they be modeled under the principle of ontological realism are the starting points for building high quality ontologies. In this section, we offer guidance on how to design a network of ontologies that covers the content of a data enterprise and be extendible as needed to adapt to the inevitable changes and growth of that enterprise.

The Multi-Tiered Architectural Approach

Ontologies can enable semantic interoperability among heterogeneous data sources as long as they are constructed according to a rigorous ontological architecture. Such an architecture is based upon the internal graph structure of ontologies in which the nodes represent types in reality and the edges represent relations between these types (standardly called 'object properties' in the semantic technology literature). All ontologies include one or more central backbone hierarchies in which the relation of subtype (aka is_a) connect the nodes. Each backbone hierarchy has a single root node.

This internal structure allows the construction of ontologies to result in a multi-tiered network connected in the following ways:

- A single, small, domain-neutral top-level ontology
- Mid-level ontologies covering broad domains having root nodes that are either direct children of classes from the top-level ontology or of a term drawn from another mid-level ontology within the network
- Lower-level ontologies representing specialized domains having root nodes that are either direct children of classes from one of the mid-level ontologies or of a term drawn from another domain level ontology within the network.

On the choice of top-level ontology, three alternatives to BFO have acquired a certain following: the Domain Ontology for Linguistic and Cognitive Engineering (DOLCE), the Upper CyC Ontology, and Suggested Upper Merged Ontology (SUMO). Of these only BFO and DOLCE can claim to be true top-level ontologies. As described above we recommend BFO and to the reasons there cited we add that out of the alternatives it has the largest number of users (http://www.ifomis.org/bfo/users), the largest body of user documentation (http://www.ifomis.org/bfo), and the most active user community forum (http://groups.google.com/group/bfo-discuss?pli=1). Above all BFO is the most widely and the most actively re-used top-level ontology for purposes of creating suites of interoperable ontology modules.

The development of mid-level ontologies proceeds by downward extension from the top-level ontology. This means that mid-level ontologies consist of terms which are organized in strict subclass hierarchies starting from more and proceeding to less general; either in one such hierarchy, or in multiple separate hierarchies representing multiple axes of classification (such as object, attribute, process). Each such

hierarchy has exactly one root node, which represents the most inclusive type in the relevant domain along one or other axis.

Mid-level ontologies serve two purposes. First, they contain terminology that describes entities of interest to many, if not all, groups within an enterprise information system. Second, they provide a means by which the developers of lower-level ontologies can incorporate their work into the common semantic architecture – the root nodes for each lower-level ontology are existing terms in the appropriate mid-level ontology. In a way analogous to the role of the top-level ontology in providing a starting point for definitions of the terms in the mid-level ontologies, so the latter in turn provide a starting point for definitions at the lower levels, thereby helping to ensure consistency of lower-level ontology development. The development of lower-level ontologies helps drive mid-level ontology development, since terminological needs identified at the lower level will in some cases need to be met by adding content to the mid-level ontologies, which can serve as starting point for further downward population. In this way, we secure incremental conformity of the lower-level ontologies and mid-level ontologies to the common architecture.

A lower-level ontology is a maximally specific representation of the entities in a single domain and thereby address the information needs of particular groups of users who require the bringing together of specific groups of terms in a single ontology. The scope of lower-level ontologies may be one or multi-dimensional. Examples of the former are those limited to the content contained in authoritative sources such as ISCO-08 Occupation Codes (http://www.ilo.org/public/english/bureau/stat/isco/isco08/), the NSAR Select Agents and Toxins List (http://www.selectagents.gov/SelectAgentsandToxinsList.html), and relevant single dimensional, list-like data resources conforming to DoD 8320.2: Authoritative Source. Examples of the latter are the Environment Ontology (https://bioportal.bioontology.org/ontologies/ENVO) or the Neurologic Disease Ontology (http://www.ncbi.nlm.nih.gov/pubmed/24314207).

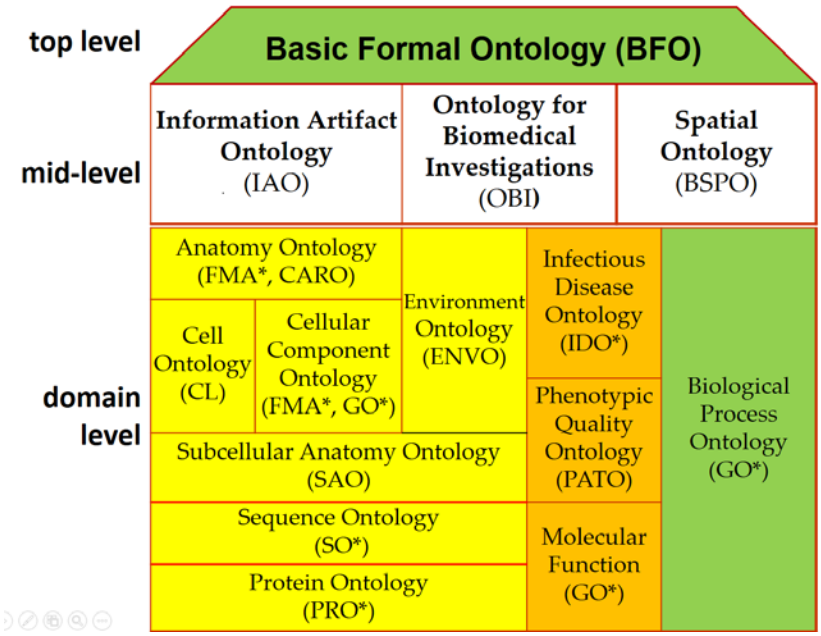

Figure 2: Top-, Middle and Lower-Level Ontologies in the OBO Foundry

Content Modularity

Consider the task of describing a large domain such as agriculture. Among the choices made during the completion of the task would be whether to create a single large mid-level ontology or to create many smaller ontologies that are then combined in order to provide coverage of the entire domain. Very large monolithic ontologies describing the entirety of a large and diverse domain carry significant drawbacks. When combined in a network finding information across multiple large ontologies is difficult, as they will have many elements in common. The notional agriculture ontology will share a great deal of content with other ontologies in the network that cover other segments of the economy such as raw materials, retail, manufacturing, supply chain, and so on. This duplication of content has the undesirable consequence that finding all of the information about a single topic will first require searching within each of the ontologies for whether they contain that topic and if so, how it is expressed. Large ontologies are also difficult to modify because of the high number of associations between elements that need to be altered as a result of a single addition or deletion. They are also difficult to govern because of the large number of stakeholders in the domain.

Because of these shortcomings, we advise developing smaller plug and play ontologies as the alternative to large ontologies. To facilitate the implementation of this advice we offer guidelines for circumscribing the horizontal scope of this content. The division of ontologies into top-, middle and lower-levels is a division along the axis of generality – the content of top-level ontologies is *more general* than that of the ontologies at lower levels. Circumscribing the horizontal scope of ontologies requires guidelines for the division of ontologies along the axis of content.

Recall from the Multi-Tiered Architectural Approach section above that mid-level ontologies are to be composed of subclass hierarchies having as a root either an element from the top-level ontology, or an

element drawn from another mid-level ontology. Rephrasing this leads to the first guideline for delimiting the scope of the content of a mid-level ontology: The domain of a mid-level ontology should be expressible either as a single class or as a statement composed of classes and an object property within that ontology. Moreover, that class or those classes should be at the root level of the ontology. The remainder of the content of a mid-level ontology are those terms and relations that adequately characterize entities of that subject. We leverage the content of the top-level ontology to build the guidelines of this task. The example of a mid-level ontology of artifacts will be used facilitate the discussion.

Following the first guideline in the creation of an ontology of artifacts, there will be a class named "Artifact" in the ontology. The task is to add content to the ontology so that it can describe artifacts without becoming too large and unwieldy. BFO provides a concise checklist of the kinds of things to which an artifact can be related. Using only leaf nodes of the BFO hierarchies of continuant and occurrent and ignoring sub-types of its classes of continuant fiat boundary, spatial region and temporal region leaves 17 kinds of things (bolded items in Table 1) to which an artifact (or anything else) can be related.

continuant
- independent continuant
  - material entity
    - **fiat object**
    - **object aggregate**
    - **Object**
  - immaterial entity
    - **spatial region**
    - **Site**
    - **continuant fiat boundary**
- specifically dependent continuant
  - realizable entity
    - **disposition**
      - **function**
    - **Role**
  - **quality**
    - **relational quality**
- **generically dependent continuant**

occurrent
- **temporal region**
- **spatiotemporal region**
- process
  - **process profile**
  - **history**
- **process boundary**

Table 1

A few examples may help illustrate the guideline in action. Generically dependent continuants (GDCs) are those entities that can be transferred from one entity to another while preserving their identity. Information remains the same while being transferred from one medium to another and so is a paradigmatic case of a GDC. So determining the scope of the Artifact Ontology includes answering the question: "Which, if any, types of information are related to artifacts?". Plausible answers would be artifact specifications such as blueprints, identifiers such as names and serial numbers and descriptions such as test reports and measurements. Continuing the method would lead to determining that artifacts relate to the BFO classes of Object and Object Aggregate via their being composed of and composed by systems, subsystems, components and parts. Artifacts have Qualities such as Dimension, Mass, Shape, and Color and Dispositions such as Tensile Strength and Functions like Flight and Surfactant. The Processes related to Artifacts are not only those in which an artifact is utilized but also those that design, produce, test and maintain them.

There must be a bound on this technique in order to make it effective. As currently stated it will result in an endless number of entities included in the scope of an ontology. One bound is to include only those entities that relate to the root type and not to other types as well. For example, in determining whether or not to include a given quality into the Artifact Ontology, answer the question of whether the quality is applicable to other types of entities as well. Qualities such as mass, shape, color, length, and texture are applicable to any material object and so do not belong in the Artifact Ontology. Continue by filtering out from those entities that are applicable to only the root node those that are not applicable to a large number of its subtypes. As an example, a quality such as caliber is not a member of the Artifact Ontology because it does not have a wide enough scope within the domain of artifacts. Instead, it would reside in a domain ontology of the artifacts to which it is broadly applicable.

Summarizing the forgoing discussion leads to the second guideline for scoping a mid-level ontology: Include all subtypes of the classes of the top-level ontology that relate to only the root class or classes of the mid-level ontology and which are applicable to a significant percentage of its instances.

Following this guidance, examples of mid-level ontology domains are: information entities, qualities, geospatial regions, temporal regions, agents, and events. In sum, the guideline is that the scope of a mid-level ontology should include all terms and relations needed to characterize – demarcate the scope of – the domain but not every subtype terms and relations. On just how many subtypes to include we offer the rule of thumb that no more than three levels of subtypes should be included for any one of the general terms or relations.

The principle of scope for lower-level ('domain') ontologies is essentially the same as that for mid-level ontologies. The root term of lower-level ontologies should be either a direct subclass of a term from a mid-level ontology or an element drawn from some already existing lower-level ontology. When developing lower-level ontologies under this guidance they become descriptions of the types of entities characteristic of the relevant domain. Examples of domains for such ontologies include: occupations, ethnicities, physiographic features, hydrographic features, criminal acts, governmental acts, artifact functions, watercraft and sensors.

Development Principles

In this section, we offer guidance on performing some of the tasks that are part of creating an OWL version of an ontology.

Consistency

An ontology is consistent if and only if every one of the classes denoted by its terms can have instances. In its simplest form, an inconsistent ontology would define a class such that any instances would need to have a set of incompatible characteristics, X and not-X. Of course, inconsistent ontologies are not built intentionally. More often than not, they are the result of oversights caused by the complexity inherent in formalizing some domain. As the number of terms, relationship expressions, and axioms within an ontology grows, so too does the likelihood of introducing an inconsistency. This likelihood is increased further as the number of ontologies within an enterprise grows, as ontology developers are provided with the need, and the opportunity, to import terms from multiple sources.

To test an ontology for consistency there are a number of automated reasoners available that can trace through all of the relationships in the ontology and detect any inconsistencies. The majority of these reasoners are capable of performing consistency tests on ontologies written in the Web Ontology Language (OWL) DL dialect of OWL (OWL 2 DL).

Principle of Single Inheritance

The principle of single inheritance prescribes that each ontology class should be a subclass of one and only one ontology class. The resultant chain of single inheritance forms what is called the asserted taxonomy. Adherence to this principle ensures that everything which holds of a parent term holds also of (is inherited by) all descendant terms at lower levels. This means that each asserted taxonomy has a single root node and that every ontology will contain one or more asserted taxonomies as proper parts. All non-root terms will have exactly one *is_a* parent and thus be connected by exactly one chain of *is_a* relations to the corresponding root.

This principle does not prohibit the use of subclass or equivalent class axioms whereby a class is defined to be a subclass or equivalent class of an anonymous class. The OWL language provides the means to express facts about entities using relations to other entities. For example, a fact such as “Automobile has_part some Engine” uses the *has_part* relation to relate each automobile to at least one engine. The semantics of this fact is that Automobile has been made a subclass of the anonymous class “thing having at least one engine as part”. The class is anonymous because it is not part of the asserted taxonomy. The set of subclass and equivalent class axioms forms the *inferred taxonomy* of an ontology. In this inferred taxonomy classes previously anonymous become explicit and the principle of single inheritance is no longer in force. For further details see: (Smith & Ceusters, 2010)

Definitions

Every class in an ontology (other than the root) should be provided with an associated human-readable definition. Standardly, achieving this association is by way of adding annotation properties, which are a means for associating metadata with ontology elements. While use of the RDFS annotation property ‘comment’ is acceptable, it is preferable that a new annotation property be created expressly for this purpose.

The human-readable definitions of classes (terms) are required to be always of the two-part form:

an *S* = Def. a *G* which *D*s

where 'S' (for: species) is the term to be defined, 'G' (for: genus) is the immediate parent term of 'S' in the relevant ontology, and 'D' (for: differentia) provides the species-criterion; that is, it specifies what it is about certain G's which makes them S's. As ontologies evolve it is expected that the human readable definitions will be translated into subclass or equivalent class axioms as described above in the section on single inheritance.

Examples:

- Natural event =def. a Process that is not caused by any human act
- Geographic event = def. a Natural event that affects some Geographic feature
- Landslide =def. a Geographic event in which there is a rapid descent of soil or rock down a mountainside

As more and more specific terms are defined through the addition of ever more detailed differentiae, their definitions encapsulate the information regarding child-parent links connecting each type all the way back to the corresponding root. At the same time, the task of formulating definitions serves as a check on the correctness of the constituent hierarchies in these ontologies.

Use of the two-part definition structure helps to ensure that definitions are not circular (when the term defined appears in its own definition), and thus that they communicate information that is of value to the user – in conformity with the principle that a definition should use only terms which are easier to understand than the term defined.

Definitions are required also for (non-root) object properties, and these two should as far as possible be defined using the two part rule.

Preservation of Meaning of Higher-Level Ontology Terms

Re-using terms from other ontologies is a fundamental principle of the best practices espoused here, since it contributes to the interoperability of ontology modules and thus also of the information systems which these modules support. However, if not performed carefully the reuse of terms can bring the risk of altering the meaning of the re-used term. The most common way in which this type of problem occurs is when an ontology reuses a term from a higher-level ontology but adds to its content through the addition of an axiom. An example is reusing the term Military Organization but adding an axiom that asserts that every such organization has exactly one leader. To be conformant the creator of the lower level ontology should either request that the curators of the upper-level ontology add the axiom, or introduce into the lower-level ontology a subtype of Military Organization (e.g. Single Leader Military Organization) to which the axiom could then be added without altering the meaning of the original.

A Step-by-Step Guide to Ontology Development

Preliminaries

1. Form a team. Make sure your team includes both SMES familiar with both subject-matter (the entities in your domain) and data (the information resources that the ontology will be used to integrate and analyze and to make discoverable).
2. Make sure that your team includes persons with ontology-building experience. (Facility with ontology software does not imply the ability to create ontologies.)
3. Identify the primary tasks your ontology will be designed to realize.
4. Identity the domain of your ontology – the types of objects and attributes and processes which the ontology will need to represent.
5. Identity the primary bodies of data your ontology will be used to annotate.
6. Be aware that your goal is to maximize the ability of your ontology to address these primary tasks *but without detriment to* its ability to address secondary uses not yet identified. (Experience shows that secondary uses are often significantly more important than primary uses, and that secondary uses are almost always what guarantees the enduring value of an ontology.) The main strategy to ensure future-proofing against problems in addressing secondary uses is to ensure that the terms and definitions in your ontology are of broad understandability and validity (rather than being understandable and valid only by your immediate collaborators and only when used in relation to your currently available data).

Gather and Select Terminology

7. Identify and evaluate existing ontologies with overlapping domains. Reuse as far as possible the ontology content which satisfies the principles enunciated in this guide.
8. Create a draft list of terms covering your domain (for example, the 100 terms most commonly used in authoritative data sources or doctrinal publications).
9. Follow the principle according to which each term 'A' on this list is shorthand for a term of the form 'the type A'.) This will mean that the list will contain only common nouns (such as ‘weapon’ or ‘building’).
10. Strive to ensure maximal consensus with SME and doctrinal usage.
11. Identify areas of disciplinary overlap where terminological usage is not consistent, and provide synonyms where SME sub-communities use alternative terms.
12. Avoid mass terms (such as 'flesh', 'fuel', 'information' ...); always use in their place an appropriate equivalent count term (which means, a term which can be preceded by 'a' and has a plural form as in: 'portion of meat', 'amount of fuel', 'information artifact'.

Avoid Standard Ontological Errors

13. Do not confuse information artifacts with the reality they denote - Distinguish explicitly between each real entity and information about it. Avoid confusing for example: ‘education’ and ‘education record’ or ‘employment’ and ‘employment record’.
14. Keep words and things separate. – Do not confuse an entity with its name or identifier.

15. Do not confuse reality with our knowledge about reality- Avoid terms like ‘unknown terrorist’, ‘known terrorist’, ‘unclassified participant’, ‘participant not otherwise specified’. Describe what exists (what types of things exist) in reality, not what is known about what exists in reality.

Format and Refine Your Terminology

16. All terms in your ontology should be nouns and noun-phrases that are singular in number. This follows, again, from the rule according to which each term ‘A’ in the ontology should be viewed as shorthand for a term of the form ‘the type A’.
17. Use acronyms sparingly and, if at all, only where they are part of common discourse (HQ, IED, WMD). Acronyms tailored to the ontology or to local habits of data recording should be recorded in the ontology as synonyms.
18. Avoid abbreviations and ellipses even when it is clear in context what they mean. To maximize secondary uses of your ontology terms should be meaningful independently of context.
19. Ensure univocity of terms and relational expressions – which means ensure that each term and each relational expression has the same meaning on each occasion of use within the framework of the ontology.
20. Do not use negative terms. There is no such type as ‘non-terrorist’, since complements of types are not themselves types. Terms such as ‘non-mammal’, ‘non-membrane’, ‘either a vehicle or a person’ do not designate types in reality. Avoid terms involving time and space modifiers. Do not include in the term words that imply time or space restrictions, e.g. ‘current’, ‘previous’, ‘local’, …
21. Do not use disjunctive or conjunctive terms. There is no type in reality *mammal or violin*; there is no type in reality *apple and sphere.* Rather, there are the types *mammal*, *violin*, *apple* and *sphere.*
22. Avoid use of modifying expressions (such as ‘cancelled’, or ‘absent’, or ‘fake’) which cancel the meaning of the modified term.
23. The principle of terminological coherence - For any expression 'E' in an ontology, 'E' means E. (The rule of univocity follows immediately from this principle.) This principle is normally, and for good reason, regarded as a trivial constraint on all sensible use of language; but for examples of violation see (Smith, 2006).

Create a Hierarchy

24. Arrange your terms to form a backbone *is_a* hierarchy, whereby each node at a level lower than the root level is connected by a single *is_a* link to its parent node. Each parent node is thus more general than its child nodes. This means that your ontology should be built in order to guarantee single inheritance.
25. Assert an *is_a* relation – A *is_a* B – only if it means sense to assert that every instance of the type *A* is an instance of the type *B.*
26. The hierarchy should be *is_a* complete in the sense that each term in the ontology is connected via a unique chain of *is_a* links to the relevant root node. Add additional terms to fill gaps in this chain. All terms in the taxonomy will then be traceable via *is_a* relations to the relevant ontology root node.

27. The hierarchy should be *is_a* complete also in the sense that each sub-term in the ontology – which means each constituent of composite terms – should be similarly connected in this way to a relevant root node (which may be a node either of this ontology or if some identified supplementary ontology). (Composite terms are for example of the forms: 'A *without* B', 'A *with* B'.)
28. Avoid 'other' - Do not create terms such as 'other terrorist', 'other weapon', …
29. Both developers and users of an ontology should respect the open-world assumption, which means that the ontology implies no assertion to the effect that it is complete (and no assertion to the effect that any list of child terms under parent 'A' is a complete list of subtypes of A).

Principles for Definitions

30. Provide all non-root terms with definitions
31. Follow the Aristotelian definition strategy, whereby each definition is of the form 'A =def. a B which Cs', where B is the parent term of A in the ontology, and C is the *differentia* which marks out those Bs which are As.
32. Use essential features in defining terms, which means that the differentia C should represent something essential to the As.
33. Avoid circularity in defining terms.
34. Ensure the intelligibility of definitions, by using terms in your definition that are simpler than the term you are defining.
35. Ensure that your definitions are un-packable, which means that if an assertion involving an ontology term 'A' makes sense, then it will still make sense when you replace 'A' with your definition of 'A'.

Bibliography

Arp, R. A., Smith, B., & Spear, A. A. (2015). *Building Ontologies with Basic Formal Ontology.* Cambridge, MA: MIT Press.

Grenon, P., & Smith, B. (2004). The Conrncopia of Formal-Ontological Relations. *Dialectica, 58:3*, 279-296.

Grenon, P., & Smith, B. (March 2004). SNAP and SPAN: Towards Dynamic Spatial Ontology. *Spatial Cognition and Computation, 4: 1*, 69-103.

Imam, F. T., Larson, S. D., Bandrowski, A., Grethe, J. S., Gupta, A., & Martone, M. E. (2012; 3:111). Development and use of Ontologies Inside the Neuroscience Information Framework: A Practical Approach. *Frontiers in Genetics*.

Smith, B. (2006). Against Idiosyncrasy in Ontology Development. In B. Bennett, & C. Fellbaum, *Formal Ontology in Information Systems* (pp. 15-26). Amsterdam: IOS Press.

Smith, B. (November 2007,). The OBO Foundry: Coordinated Evolution of Ontologies to Support Biomedical Data Integration. *Nature Biotechnology 25 (11)*, 1251-1255.

Smith, B., & Ceusters, W. (2010). Ontological Realism as a Methodology for Coordinated Evolution of Scientific Ontologies. *Applied Ontology 5*, 139-188.