Methodology for Semantic Enhancement of Intelligence Data

Release Version: 3/12/2013

Prepared by:

Barry Smith, PhD (National Center for Ontological Research)

In collaboration with:

Tatiana Malyuta, PhD and William H. Mandrick, PhD (DataTactics, Inc.)

CUBRC Report

Table of Contents

1 Introduction to Semantic Enhancement

What follows is a contribution to the horizontal integration of warfighter intelligence data as defined in Chairman of the Joint Chiefs of Staff Instruction J2 CJCSI 3340.02AL:

> Horizontally integrating warfighter intelligence data improves the consumers' production, analysis and dissemination capabilities. HI requires access (including discovery, search, retrieval, and display) to intelligence data among the warfighters and other producers and consumers via standardized services and architectures. These consumers include, but are not limited to, the combatant commands, Services, Defense agencies, and the Intelligence Community. [22]

We describe the methodology for a process of Semantic Enhancement (henceforth: SE) that is designed to achieve these ends. SE is a general, flexible, incremental approach that is designed to be reused across multiple data collection endeavors. It is being implemented as part of the DCGS-A Cloud initiative, where it is being used to create a Shared Semantic Resource (SSR) for the Intelligence Community.

1.1 Outline of Terminology and Definitions

An **ontology**, in what follows, is to be understood as a graph-theoretical structure whose nodes are preferred terms (also called 'preferred labels'; often abbreviated to 'term' in what follows) connected together by the edges of the graph.

Preferred terms are singular nouns and noun phrases (for example 'vehicle', 'wheeled vehicle') representing types in reality.

Each node in the graph is associated in addition with

1. a definition
2. a unique alphanumeric ID
3. a namespace ID (an acronym derived from the name of the ontology)
4. a URI (internet address) built from a path including the namespace ID for the ontology and the alphanumeric ID (which appears as the terminal string of the URI)
5. a list of synonyms (nouns and noun phrases equivalent in meaning to the preferred term, including translations of this term into other languages).

The edges in the graph represent type-level **relations**. Relational expressions such as '*is_a'* and '*part_of*' are labels for such relations. [4]. The triple consisting of terms 'A' and 'B' connected by an edge labeled '*is_a*' can be understood as an assertion to the effect that all instances of A are instances of B, and thus that A is a subtype of B.

Semantic technology. Currently, the state of the art in implementing an ontology approach of this sort involves use of the W3C language specifications: XML (Schema), RDF(S), and the Web Ontology Language (OWL 2) [21], which in turn build on HTML and on the use of URIs to create unique labels for ontology terms. The methodology described in what follows incorporates references to this technology, and especially to OWL. However, the methodology is designed to be independent of specific implementation technologies, and we anticipate that future implementations will incorporate more expressive languages such as the Common Logic Interchange Format (CLIF) [28].

Semantic technology has evolved as a strategy for counteracting the problem of **data stovepipes** which arises as a result of the fact that data-models are created in heterogeneous, uncoordinated ways, leading to a failure of data integration and reuse. Ontologies are designed to ensure consistency and persistence in the usage of terms – thus also persistence of meaning – of a sort which is designed to counteract the consequences of frequent changes in the databases and software used to store and manage data. But ontologies provide not only for persistency of meaning of the sort which might be achieved by means of a simple glossary. Ontologies provide both natural language definitions and also formal definitions of the sort which can offer advantages to computer-based retrieval and analysis when software tools are applied to semantically enhanced data.

Semantic technology provides standard languages for ontology formulation, languages sanctioned by the W3C (http://www.w3.org/2001/sw/Specs), which have spawned in their turn powerful open-source software (http://www.w3.org/2001/sw/wiki/Tools). Unfortunately, the resultant popularity of semantic technology has itself led to a situation where ontologies themselves are 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 methodology here described is designed to counteract this tendency by providing a set of principles and rules and an associated governance regime that is designed to ensure that ontologies are developed in a consistent, non-redundant fashion.

The horizontal integration of data is a process whereby two or more data resources are combined in such a way that search and analysis processes can be applied to the combined data as if it were a single resource.

Sometimes such integration will be effected purely at the syntactic level. Two or more data resources are said to be s**yntactically integrated**, in this sense, when they share a common data format.

Syntactic integration does not go very far, however. For example, syntactically identical terms being used in two distinct resources R_1 and R_2 may be associated with different meanings, or two different terms from these resources may be associated with the same meaning.

The sort of integration we require in the intelligence, as in other, domains is **semantic integration**, which obtains where two or more data resources are integrated at the level of meaning. When integration of this sort is achieved, then search and analysis processes addressing the meanings of the terms used to describe the data in the integrated resources can be applied to their combined data as if they formed a single resource. Thus if resources R_1 and R_2 are semantically integrated, then any term and any relational expression that is used in both resources will have the same meaning.

Under these circumstances data about the same topic will always be described using the same terms. Thus, for any given topic T, if data about T is available in a given resource R, then this data will be returned in a single query for T even where the term 'T' is not used in R. R might be, for example, a resource in which the data models use idiosyncratic acronyms, or a resource containing data about the parts or subtypes of T but without using ‘T’ to describe them.

Such semantic integration can be achieved, in principle, following two strategies:

1. globally, by transforming the data resources in such a way that they conform to some single overarching data model
2. incrementally, by annotating the data resources with terms derived from ontologies built as an evolving set of modules.

Strategy 1 is designed to achieve full semantic integration through the imposition of a common set of terms for all data resources – which will in effect be replaced by one single data resource. The advantage of this strategy is that it aims, indeed, to achieve semantic integration across the entire set of data resources that have been globally transformed. The disadvantages, however, are many: the approach is

likely to be inflexible, ingestion of legacy data into the proposed global model will be expensive, and difficult to carry out consistently in areas such as military intelligence where highly heterogeneous bodies of data are involved, and will entail loss and or distortion of original data and semantics. Moreover, this strategy lacks agility of the sort needed to deal with Big Data in today's dynamic environments.

In what follows we describe a methodology for achieving semantic integration through strategy 2. This strategy has the advantages of flexibility and adaptability to evolving needs; its modular structure means that it can draw on the division of labor in ontology development and application. On the other hand it has the disadvantage that the semantic integration it achieves will always be partial; on the other hand we shall see that even partial coverage of very large data resources can bring significant benefits.

1.2 The Methodology of Semantic Enhancement

Semantic Enhancement (SE) is a strategy for supplementing source data artifacts with ontology terms in ways designed to counteract the variability in data-models. It provides semantic integration, and as a result, improved opportunities for computer-based retrieval, aggregation and analysis. We use the term 'SE ontologies' to refer to all ontologies created following the methodology described in this document; this includes both *asserted* ontologies (as described below) and the ontology resources which can be *inferred* therefrom. We use 'Shared Semantic Resource (SSR)' to refer to the totality of these SE ontologies.

SE is a tested approach to data integration that has been used successfully for over ten years especially in the bio-informatics field [16, 34]. As applied within the DSC DCGS-A Cloud it is designed to advance integration of warfighter intelligence data through application of semantic technology in a way that both counteracts the problem of data stovepipes in the future and incrementally rectifies the effects of legacy stovepiping in the past.

Use of the Web Ontology Language (OWL) and of the Protégé or Topbraid Ontology Editor

We recommend that ontologies be created as OWL files (more precisely: as files using the OWL 2 Web Ontology Language), with versions in other languages (such as RDF on the one hand, or CLIF on the other), to be created according to need. We shall concentrate on the methodology for creation of ontologies presupposing that those following this methodology are or will become familiar with either the Protégé editor [23] or with the TopBraid Composer [29].

Why this Methodology is Needed

Using OWL 2 standard provides one standardized implementation of the SE methodology, but it does not ensure the kind of constraint on ontology development that is needed for successful realization of the SE approach [24]. Our focus here is on the design, architecture and governance principles needed in order to provide the needed constraints through coordination of ontology development efforts across multiple domains and communities. Such coordination standardization is extremely important in the intelligence community. Our goal is thus to contribute to unity of effort in the use of intelligence data as defined in JP 1 as: "Coordination and cooperation toward common objectives, even if the participants are not necessarily part of the same command or organization." [25]

It is through creating a uniform cumulative set of annotations across ever-growing heterogeneous collections of source data that the approach unlocks the true potential of such data.

1.3 Principles of the SE Strategy

SE ontologies are ontologies which are compliant with the principles of the SE strategy.

1. The methodology must support an incremental process of ontology creation.
2. The ontologies must be constructed and maintained in such a way as to form a single non-redundant and consistent suite.
3. The suite of ontologies must be capable of evolving in an agile fashion in response to new sorts of data and new analytical and warfighter needs.
4. The suite of ontologies must be constructed and maintained by a collection of groups working in a distributed way and collaborating on the basis of rigorous sharing of information. Thus:
    a. New ontology initiatives are registered openly in a way that makes them visible to all collaborating groups.
    b. All participants are informed of the results achieved at every stage of ontology development.
5. There is a governance and change management process to ensure consistent ontology development and maintenance.

6. To ensure conformance to a single architecture all domain ontologies will be created by downward population from a common domain-neutral, upper level ontology with a common set of domain-neutral relations.

7. The ontologies must be capable of being applied to source data-models and terminology resources through a process of what we call 'annotation' or 'tagging' [5]. The goal is to ensure that data pertaining to a single topic X, even where they are captured in highly heterogeneous ways in separate source data-models, will all be associated with the single ontology term 'X'.

8. The SE ontologies must thereby remain external to the resources to which they are applied, so that the latter remain unchanged through the SE process and under the control of their original authors. Ontologies are applied to the annotation of data models in an arm's length process yielding a new, SE layer associated with the source data and building on the Shared Semantic Resource.

9. The ontologies themselves must remain free of entanglements with the data-models; this allows independent development of the ontologies in accordance with the SE methodology, and promotes reuse of the ontologies in annotating multiple successive data-models.

10. Common architecture: the ontologies must form an organized suite conforming to a single architecture based on a single upper level ontology, a common set of logically defined relations, and mid- and lower level plug-and-play ontology modules created by a process of downward population from parent ontologies at a higher level.

11. Leverage of existing resources: ontology development involves reuse of existing ontologies and authoritative data sources wherever possible.

12. Guiding role of SMEs: subject-matter experts, including analysts, must be involved in the construction and maintenance of all domain ontology content and in the selection of terms, which should be familiar to those who are using the ontologists which result.

13. Feedback loop: lessons learned in annotation and use by analysts (including detection of errors and gaps) are incorporated into the process of ontology update. This feedback loop between users of the ontologies and their developers is designed to ensure adequacy of the ontology resource to evolving warfighter and analyst needs.

14. Governance: conformity to the architecture and development principles is ensured through a common governance and change management process and through coordinated training of ontology developers and users.

15. Repeatable process: ontology development takes place on the basis of a repeatable process.

The essence of SE is to create a shared semantic resource that can serve semantic integration of multiple different source and legacy data-models and terminology resources. Such integration is already improving search results within the DCGS-A Cloud, where is being applied to a particular very large distributed resource containing highly heterogeneous intelligence data [19].

Recursive Steps for Semantic Enhancement

What follows is a guide to the recommended methodology for building the shared semantic resource. The methodology proceeds through the following recursively applied steps:

1. Create an incrementally growing collective resource in the form of a suite of ontologies consisting of both newly created and legacy ontologies, the latter being modified as needed to ensure conformity to the SE architecture.
2. Set these ontologies to work in a process of annotation of data, initially through annotation of the column headers of data-models.
3. Use the resulting annotated data for computer-aided search and analysis.
4. Use feedback from 2 and 3 to identify gaps and errors in the ontologies as they exist at any given stage.
5. Return to step 1, improving and expanding the ontology resources as needed.

1.4 Benefits of the SE Approach

- The SE approach is incremental. Gains are visible already in the very first cycle of application of the methodology thereby justifying the investment of further effort. Additionally, the resource should be capable of being efficiently used in a way that does not compromise performance in search and analytics applied to the data.

- The SE approach will introduce into the intelligence domain a tested, state-of-the-art approach which is designed to improve data retrieval, integration and analysis. It combines:
    - Net-centricity
    - Re-use of existing ontological resources across different communities
    - Efficient and flexible application to large, heterogeneous and rapidly changing bodies of source data
    - Efficient integration with other resources developed on the basis of the same principles
- It will bring benefits in:
    - Search
    - Querying
    - Analytics
    - Reasoning
    - Data collection and production

2 The Ontological Approach to Data and Semantic Integration

2.1 Ontologies and Data Models

The SE strategy will consist initially in application of ontologies to the enhancement of data-models. Thus it is important to understand the difference between the two. (A document expounding these differences is available here.)

Ontologies specify meanings (the 'semantics' of terms) by providing definitions. The latter are in some ways analogous to the definitions provided in a dictionary, and ontologies do indeed in some contexts play the role of computational dictionaries (for example in serving educational purposes). Traditionally, dictionaries are focused on the usage of terms within a natural language; hence they need to define all the meanings associated with a given term by a given linguistic community. Ontologies, in contrast, are focused on providing a controlled vocabulary for talking about the types of entities and relations within a given domain. Hence they must provide one term, and one definition, for each salient type in each domain of interest. (Here entities may include not only physical things and physical events but also immaterial

entities, including the ideas, beliefs and plans in peoples' heads, laws, epidemics, military operations, property rights, national borders, credit default obligations, and so on.)

Data-model designers, too, use types, but they see types not as entities in reality but rather as abstractions embodying efficient ways of describing the data about reality that is needed by an application (efficient both for reasoning and for storage). Fig. 1 illustrates on the left an ontology fragment, and on the right a sample of different possible data-models drawing on the same repertoire of types.

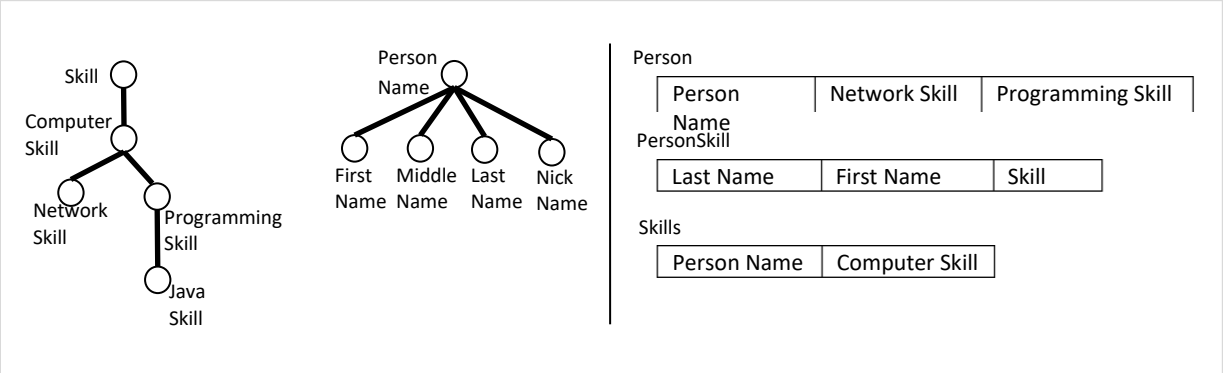

Figure 1: Example SE ontologies, with their constrained hierarchies (LEFT), and data-models, in which terms are combined in multiple ways in different data tables (RIGHT).

The main goal of data model construction is facility in the handling of specific data. The main goal of SE ontology construction is objectivity of representation – and thus shareability across many communities of interest.

The Figure shows how, when ontologies are properly constructed, then every term used to describe data appears only once in the ontology hierarchy. The ontology is synoptic – it represents an entire hierarchy of types at different levels of generality. Each term is associated in an intelligible way with its subsuming and subsumed terms (that is, with its ancestors and descendants) in the ontology hierarchy. Each data-model, in contrast, is flat and represents arbitrary combinations of types suitable for particular purposes. These data can often be re-combined for new purposes only through significant manual effort.

In both ontologies and data-models types are *general* or *repeatable* entities capable of being instantiated by indefinitely many particulars. In ontologies, however, the types and instances are on the side of reality; in data-models, the types are data abstractions on the side of a particular *representation* of reality.

Data-models are created to capture targeted selections of features as needed for a specific task. For the SE strategy to work, in contrast, it is required that there be exactly one authoritative ontology for each salient domain of reality. This is because to create a cumulating set of annotations the same ontologies must be reused over and over again in application to different data resources and in support of different analyst communities, whose data can thereby be pooled as it grows over time.

2.1.1 The Closed and Open World Assumptions

Each data-model is 'closed' in the sense that we can describe completely the world that it represents simply by examining the data representations that have been used in its specification. If, for example, there is no field for gender in a given data-model for person data, then there is no gender in the world defined by this data-model.

For reasons of efficiency of storage, database reasoning is still confined, in effect, to search based on this closed world assumption. If we do not find something in the database, then this means that this something does not exist in the limited world as defined by our database. If we do not find an assertion to the effect that F holds of a, in the database, then we must either: assume that F does not hold of a, or: add some new type of data to our database.

Ontologies, in contrast, are in the following sense 'open': they exploit a logical framework which is based on the idea that we can never describe entities in the real world completely. This means that from the absence in an ontology of a particular term 'A', we cannot infer that As do not exist. It means also that ontologies are constructed in a way which allows easy addition of new types and subtypes to the existing hierarchy. Use of OWL also allows flexible merging of ontologies for specific purposes, and flexible import of portions of existing ontology terms into other ontologies. Because ontologies are more flexible and easier to expand in these ways, the SE approach is also more stable than many common approaches based on data-model, which typically involve the building of further data models, to which existing data models then somehow have to be mapped. It is not necessary to build a new ontology every time your data changes. This, too, is a feature which is exploited when we use ontologies for data integration.

Dimension of Comparison	Traditional Data-Model	SE Ontologies

Focus	On types defined in a particular data representation	On types in reality
Closeness to reality	Variable, application-specific	Reality is always the prime focus
Conceptualization of the domain (see Fig. 1)	Plain and partial (always at the level of detail needed for a particular implementation)	Hierarchical, simultaneously describing the same domain at different levels of detail
Vocabulary	Application-specific, not intended for sharing	Application-independent, intended to support sharing and distributed reuse
Structures or organization of types	Groupings of types to accommodate data access patterns (e.g. relations in the RDB)	Taxonomies (type hierarchies) always used to describe/classify the domain
Combinability	Distinct data-models can rarely be combined; even where combination is possible this will typically require significant manual effort	If the SE methodology is followed when ontologies are constructed, then the results will be combinable automatically
Flexibility	Changes in data-models, for example to incorporate new types of data, normally require significant effort	Changes can normally be effected very easily.

2.2 Basis in Reality and Role of Doctrine

As already noted, ontologies have in the past often been created in *ad hoc* fashion to address specific local needs (for example reflecting specific ways in which data have been collected), with no attention to issues of consistency or interoperability. For SE, in contrast, the goal is to create a single, evolving suite of mutually consistent ontologies, and this requires a focused investment of effort involving multiple communities. To justify such investment the ontologies which result should enjoy a broad and lasting value, which means that they should be useable by multiple communities and their utility should be preserved – through a strategy of ontology versioning – even when they are amended through time.

Ontological Realism

To maximize both utility and stability, the process of ontology development should rest on what in [7] is called 'ontological realism', which rests on the idea that an ontology should be analogous not to a *data model*, but rather to a *reality model.* The central SE ontologies are accordingly representations not of the *data* to which the SE mapping process will be applied, but rather on the salient domain of reality to which such data refer. Some SE ontologies will indeed need to contain terms referring to data items – for example to emails or other text documents. But these ontologies will then treat these data items as

entities in reality in their own right, thus following the same principles as those used in building other SE ontologies.

The first step in all cases is to specify the salient types of entities within the domain. In some cases we can draw for specifications of such domains on subject-matter experts (SMEs), including intelligence analysts; in some cases we can find such specifications in existing authoritative publications.

The Role of Doctrine

One important source is doctrine, as expressed for example in Joint Publication 2-0 (Joint Intelligence), Joint Publication 2-01 (Joint and National Intelligence to Support Military Operations), and in the Army Field Manual 2-0 (Intelligence). [26] Indeed the SE strategy for ontology development follows as closely as possible the strategy that is applied in developing joint doctrine. In both cases the goal is to create a consistent, evolving suite of doctrinal publications in a way that requires a focused investment of effort involving multiple communities. In the case of doctrinal publications, this investment of effort is justified by the fact that the results have been shown to enjoy a broad and lasting value. Joint doctrine is developed in such a way that (1) it is useable by multiple communities and (2) its utility is preserved even when doctrinal publications are amended through time. We believe that a similar justification applied also in the case of Semantic Enhancement.

2.3 Entities and Domains

In sum, the ontologies in the Shared Semantic Resource will be constructed in such a way that they employ as far as possible the common terms used by analysts, and these terms should relate in almost all cases not to data artifacts but rather to the entities in reality which these data are about. The objective is to establish what we call domain ontologies consisting of terms used to describe the types of entities existing in individual domains of inquiry.

We use ‘entity’ to refer to anything that exists or has existed. Examples: President Obama, Julius Caesar, the Second World War, your body mass index, the US Marine Corps, Operation Desert Storm, a database, the US Constitution, the original documents on which the US Constitution is written, the documents on which the US Constitution is printed. Every entity is either a *particular* or a *type* (and never both) [7, 8, 9]. Particulars are entities which occupy specific regions of space or of spacetime. Types are repeatables; they

can be instantiated by (typically indefinitely) many particulars in different regions of space or of spacetime. Julius Caesar is an instance of the type *person*; each database is an instance of the type *database.* Instances and their associated types are divided into two broad categories of continuant and occurrent. The former are entities which continue to exist through time and have a history (for example: persons, countries, weapons). The latter are entities (also called events, processes, histories) which occur in time and are divided into successive phases (for example of beginning, middle and end).

We use 'domain' to refer to a portion of reality that is the focus of concern of a specific discipline, such as:

- Geospatial Feature
- Country
- Fingerprint
- Organization
- Military Operation

and so forth. Such domains can include further domains at lower levels, for example

- Military Operation
  - Army Operation
    - Army Reconnaissance Operation
      - Motorized Army Reconnaissance Operation
      - Long range surveillance operation

SE ontologies adopt this hierarchical architecture of domains and sub-domains to the degree that is needed to ensure that, wherever ontologies overlap, the parts they share in common will be consistent with each other because they draw on terms from the same higher-level ontologies. Single domain ontologies at the lowest level will thereby serve as plug-and-play non-overlapping ontology modules, which can be developed independently with the aid of corresponding teams of SMEs. Because these are non-overlapping (contain no terms in common), then can be used in annotations with the assurance that SE mappings created from multiple modules of this sort will never give rise to inconsistency. Only thus can these annotations cumulate in a way that gives rise to the sort of large semantically integrated space that is needed for efficient search and effective analysis.

3 Realization of the Strategy

3.1 An Incremental, Cumulative Process

For SE to succeed, we need to create an incremental, evolutionary process for ontology creation, where what is good survives, and what is bad fails. This means that we need to use stable, tested best practice ontology architecture components wherever possible. We also need an approach that can work with multiple different sorts of data, and with multiple data processing systems, both of which are changing over time.

To achieve this end, the SE approach requires an arms-length relationship with the data resources which it is used to enhance. An arms-length strategy of this sort has been tested in multiple areas – above all in biology and medicine – where knowledge is rapidly advancing and where data is constantly being subjected to new uses.

The ontologies developed to support the integration of data in such fields need to evolve to take account of new advances. For this, tested principles of ontology management need to be adopted, as outlined in [30]. In brief, ontology modules are released in new public versions at regular intervals in which changes are labeled in such a way that older annotations can be correctly interpreted and amended where necessary to remain in conformity with the new version of the ontology (and thus preserve the consistency of the semantically enhanced resources as a whole).

3.2 Reference and Application Ontologies

We distinguish ontologies of two sorts: **asserted** and **inferred**. An asserted ontology is an ontology that is created manually by its ontology developers to conform to certain principles, the most important of which concerns the explicit declaration of the *is_a* relations between its terms. An inferred ontology is what results when multiple asserted ontologies (or portions of asserted ontologies) are merged – which means: combined together into a single ontology – and an automatic reasoner [20] is used to infer new relations on the basis of the merged content [31, 32, 35].

Asserted ontologies serve as a basis of reference ontologies - the ontology modules on which the SSR is based. These are representations of single domains; they are designed to be shareable, and their

development is conducted by a designated technical team. The content of these reference ontologies is then re-used in building *application ontologies*, which arise by combining terms from reference ontologies with new local content through use of reasoners. Application ontologies are built to address specific local needs and to reflect a particular perception of the relevant domains The goal is that each domain will be described by just one reference ontology, which can be reused for multiple purposes. For each domain it is possible to have multiple application-specific ontologies. These application ontologies are not meant to be shared; but the data stores to which they will be applied will still be compatible and interoperable because the application ontologies will be based on a common collection of reference ontologies. In those cases where terms originally created within a specific application ontology are widely reused by multiple communities, such terms will be incorporated into the relevant reference ontology at the next level upwards within the reference ontology hierarchy.

In Protégé 4.1 merging is achieved (1) by loading the ontologies to be merged into the editor, (2) responding in the affirmative to the question "do you want to open the ontology in the current window?' which will appear when the second ontology is loaded, and (3) moving to the *Refactor → Merge Ontologies* menu and following the instructions.

3.2.1 Principle of Single Inheritance

Asserted ontologies that serve as a basis of reference ontologies are required to be created in accordance with the following:

> Principle of asserted single inheritance. Each asserted ontology module should be built as an asserted monohierarchy (a hierarchy in which each term has at most one parent).

Single inheritance means that everything which holds of a parent term holds also of all descendant terms at lower levels. This means that each asserted ontology is required to have either a single root node, or to be divisible into separate sub-ontologies each of which has a single root node. All non-root terms will have exactly one is_a parent and thus are connected by exactly one chain of *is_a* relations to the corresponding root. A fragment of a simple single-inheritance ontology is illustrated here:

vehicle =def: an object used for transporting people or goods

tractor =def: a vehicle that is used for towing

artillery tractor =def: a tractor that is used to tow artillery pieces

wheeled artillery tractor =def: an artillery tractor that runs on wheels

tracked artillery tractor =def: an artillery tractor that runs on caterpillar track

Given a set of such asserted modules, the inferred ontologies are created using the strategy set forth in [31] and using software tools such as ontofox. Because the ontology reasoners [20] will infer new *is_a* relations between the terms in the inferred ontology; the latter will typically be a polyhierarchy, in which some terms will have multiple *is_a* parents, as in:

Canadian artillery tractor *is_a* tractor

Canadian artillery tractor *is_a* artillery vehicle

Canadian artillery tractor *is_a* Canadian vehicle.

The goal is to create a basis of ontology monohierarchies – ontologies marked by single inheritance – which should as far as possible reflect the existing disciplinary division of knowledge among specialists in the relevant domains and subdomains. These ontology monohierarchies are designed to serve as a basis of the reference ontologies, which in turn serve as a global resource to be reused over and over again in all ontology development. Polyhierarchies should be created according to particular need, for example, support for information retrieval or for the representation of multi-disciplinary content or of the results of a particular body of intelligence. We will deal first with the asserted ontologies, to which the architecture and governance requirements described in what follows primarily relate. Analogous requirements on the side of the inferred ontologies are met automatically as a consequence of the way inferred ontologies are created.

3.3 The Architectural Approach

The goals of the SE strategy can be achieved only with a rigorous common ontological architecture. The architectural approach described in this communication supports development of a shared semantic resource with the desired properties without limiting its semantic expressiveness.

3.3.1 Three Levels of Asserted Ontologies

The asserted ontologies within the SSR are organized into three groups, as follows:

- A single, small, domain-neutral upper-level ontology (ULO)
- Mid-level asserted ontologies (MLOs) covering broad domains at levels of detail that are below that of the ULO but above the lowest level
- Lowest level ontologies (LLOs) representing homogeneous one-dimensional domains.

The role and organization of each level are described in the following sections.

All MLOs are created by downward population from the ULO. This means that they consist of terms which are organized in strict *is_a* hierarchies ('trees') of more and less general; either in one such tree, or in multiple separate *is_a* hierarchies. Each such hierarchy has exactly one root-node, which corresponds to the most general type in the relevant domain, and which is a direct child either of some ULO term, or of a term drawn from some already existing MLO at a more general level within the hierarchy. Each lower node in the hierarchy has exactly one parent.

3.3.2 Modularity and Ontology Reuse

The SSR is based on a set of reference ontology modules together with ontologies that can be derived therefrom through inference. The reference modules of the SSR exist at different levels of generality, but on each level the set of modules are non-overlapping in the sense that (1) they do not contain any terms in common, (2) they deal with non-overlapping domains of entities, typically domains defined as areas of interest (as for example: Intelligence, Operations, Logistics, at one level; Army Intelligence, Navy Intelligence, Airforce Intelligence, at the next lower level).

Modularity plays a central role in our strategy for building reference ontologies. Ontology developers have thus far – echoing the developers of databases – normally worked in uncoordinated fashion, with each separate group of developers creating a new ontology for each new purpose, with little attention to the benefits of reuse of existing resources. As justification for the acceptance of this state of affairs it is argued that even where ontologies are developed independently and are thus marked by mutual redundancies

and inconsistencies, the results can still be useful for purposes of cross-ontology data integration, because relevant ontologies can be supplemented by cross-ontology *mappings* or *alignments* [33].

Unfortunately experience tells us that such mappings are both expensive to create, since they require considerable investment of effort and expertise, and expensive to maintain, as the ontologies on both sides of the mapping will in all probability change independently. Experience shows that cross-ontology mappings are accordingly rarely maintained in tandem with the mapped ontologies themselves.

The result of this lack of coordination of ontology development efforts, and of the failure of ontology mapping, is that the very same problems of database integration which ontologies were designed to resolve are recreated at the level of the ontologies themselves. Our modular approach, however, avoids the need for such mapping effort by ensuring that ontology work collaboratively in the sense that they accept a common ontological architecture, common governance and a common set of rules. (Compare the discussion of 'unity of effort' as defined in JP 1-02 [26].)

3.4 Benefits of Normalized Ontology Modules

The grounding in hierarchically organized modules brings, first, the practical benefit of ensuring that the suite of asserted ontologies is easily surveyable. (This reflects a general strategy to promote surveyability, used for example in the file storage architecture in a computer, and in the hierarchical structure of military doctrine as depicted in Figure 2.) Surveyability is important because developers and users of ontologies need to know where the equivalents of a given term are to be found in the ontology hierarchy; they need to know also that, for a given term, equivalents are not to be found in the ontology hierarchy, and then they need to know also where the new term should be inserted.

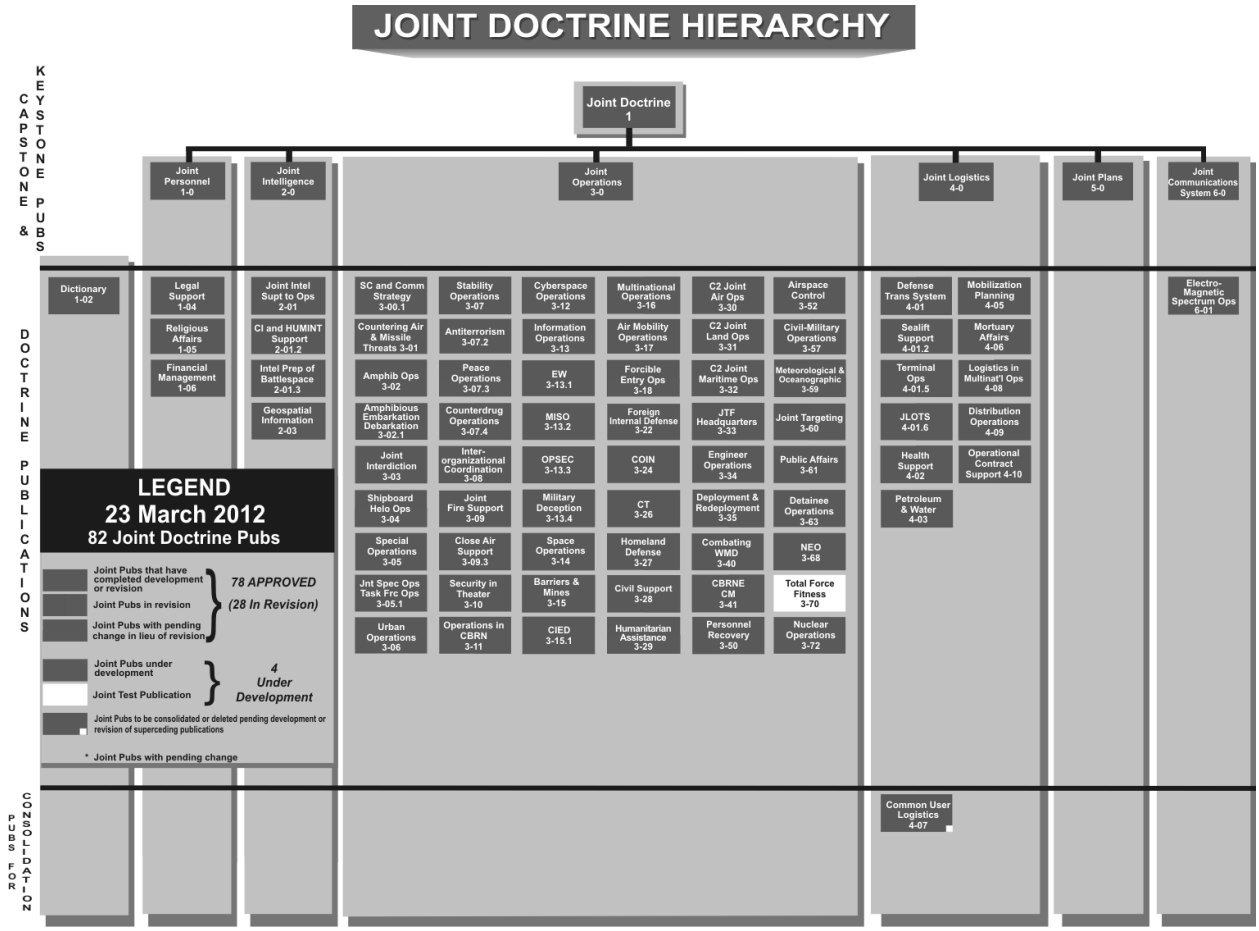

Figure 2: Joint Doctrine Hierarchy Chart (from http://www.dtic.mil/doctrine/)

Second, the modular approach brings benefits in allowing an effective division of labor among ontology developers and SMEs in developing and maintaining ontologies.

Third, the requirement that reference ontology modules should be based on monohierarchies brings computational (performance) benefits. Modules of this sort are referred to by Rector [18] as *normalized* ontologies, alluding to the way in which the process of normalization of a vector space, whereby a basis of normalized unit vectors is chosen, in such a way that every vector in the whole space may be identified as a linear combination of such unit vectors. Creating normalized monohierarchies to serve as a basis for the tangled polyhierarchies that need to be used in applications can be seen as a process of untangling. This, as Rector [18] shows, makes ontology-based integration of data easier to manage and scale because it is easier to master the problems associated with combinatorial explosions when single-inheritance modules serve to define what shall count as allowable sorts of combinations. It is also easier to maintain ontologies, for example, when a change must be made due to some advance in knowledge. This is because

the change in question can be made in just one place in the relevant asserted ontology, allowing consequent changes in the associated inferred polyhierarchies to be propagated automatically.

Building the SSR on the basis of asserted monohierarchies brings benefits also, as we shall see below, in providing a simple rule for formulating definitions of terms in ontologies of different sorts – a rule which can be applied only in a context where every term to be defined has exactly one parent.

SE ontology modules help further in preventing errors, because their plug-and-play character helps to encourage ontology reuse. Because those who are called upon to construct new ontologies are more easily able to draw upon ontology content that has been built following tested principles; they therefore do not need to construct ontology components anew; thus they can avoid creating new errors and inconsistencies; and thus they can avoid the creation of new avenues for silo formation. At the same time, where asserted modules for a given domain do not exist, the strategy provides a clear recipe for creating such modules in a way that will be of maximum benefit for the SE community in general.

Asserted ontologies are created for purposes of reuse of their content for multiple different purposes by multiple different groups. The Organization Ontology, for example, is created to represent the domain of organizations, both military and non-military, in a general, stable and consistent fashion. Thus the terms of the Organization Ontology pertain to features such as *membership*, *member role*, *organizational rules*, *chains of authority*, and so on.

An inferred ontology might be created to represent how these general types and relations are instantiated in specific groups of military and non-military organizations, for example because they are of special importance to analysts dealing with A. terrorist organizations participating in drug dealing, or B. financial organizations who may be participating in terrorist money laundering pipelines. The latter might draw on selected parts of multiple asserted ontologies as sources for the needed terms. Inferred ontology A. will add to the Organization Ontology terms drawn from the Terrorism and Drug Ontologies, for example. Inferred ontology B. will add terms drawn from a Financial Event Ontology.

Both asserted and inferred ontologies can be easily checked for consistency at every stage using ontology reasoners [20] such as are built into the Protégé and TopBraid tools for editing ontologies. These same reasoners can be used also to check the consistence of the resultant annotations; when inconsistencies

are identified, these can be flagged as being potentially of significance to the intelligence analyst, and corrected where possible.

Briefly, modularity based on monohierarchies promotes simplicity of focus in ontology development and consistency of the annotations which result through use of these ontologies. Specifically it allows:

- a clean division of labor among domain experts, who can therefore also manage most of the governance aspects pertaining to their own domain;
- automatic consistency of the results of the distributed ontology development efforts;
- additivity of annotations even where multiple independently developed ontologies are used;
- lessons learned in experience developing and using one module can be used by the developers and users of later modules;
- increased likelihood of reuse, since potential users will be aware that they are investing in the results of a coordinated approach of proven reliability;
- increased value of training in any given module, since the results of such training can easily be re-applied in relation to other modules;
- all of those involved can more easily inspect and criticize the results of others' work and will automatically be informed of salient developments;
- the resulting collaborative environment for ontology development serves as a platform for innovations which can be easily propagated throughout the whole system.

If ontologies are developed and used in this consistent fashion then the SE strategy will be marked by a number of network effects, whereby the value of the semantically enhanced data existing at any given stage increases as other data is semantically enhanced in the future, thereby increasing further the incentives to continue to use the SE strategy. But this effect is harnessed only if the ontologies enjoy broad use in annotations.

4 Upper-, Mid- and Lowest-Level Ontologies

The *level* of an ontology is determined by the level of generality of the types in reality which its nodes represent. Object, drawn from BFO, is a very general type; Living Thing and Person are terms at lower levels of generality. Terms such as Blonde Hair Color or Blue High Color are lowest level terms.

- The Upper Level Ontology (ULO) in the SE hierarchy must be maximally general – it must 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). Many of the terms in the ULO will be undefined. This is because the process of definition passes always from less familiar defined terms to more familiar terms used in the definitions. At the very top of the hierarchy we are dealing with expressions (such as 'is', 'identical', 'exists', 'process') which are so basic to our understanding that the process of definition cannot go further.
- The MLOs introduce less general and more detailed representation of types which apply in multiple domains – for example relating to different sorts of emergencies as in UCore SL [10], or different sorts of information artifacts as in the Information Artifact Ontology (IAO) [http://code.google.com/p/information-artifact-ontology/].
- An LLO is a maximally specific representation of the entities in a single-dimension domain such as the Skill Ontology or the HairColor Ontology. Such ontologies may be extended by authoritative sources such as ISO 3166 Country Codes, the NSAR Select Agents and Toxins List, and relevant single dimensional, list-like data resource conforming to DoD 8320.2: Authoritative Source.

4.1 Basic Formal Ontology as Upper-Level Ontology

As ULO we select Basic Formal Ontology 2.0 [2, 3, 7] (BFO). BFO is a small highly abstract upper level ontology, which is distinguished from the other ontologies addressed in this document, in that it is designed not for use by creators and by the analyst consumers of annotations, but rather for use *under the hood* (for example for governance purposes and to address the needs of the developers of the SE architecture). Its role is to provide a framework that can serve as a starting point for downward population in order to ensure consistent ontology development at lower levels. Since almost all SE ontology development is at the level of MLOs and LLOs, BFO itself will in most cases be invisible.

The application of a single formal ontology such as BFO across the entire suite of ontologies in SSR brings benefits of reuse, supporting consistent annotation of data across different domains. BFO also supports formal reasoning, and is associated with a set of common formal theories (for example of mereotopology [1] and of qualitative spatial reasoning [6]) which do not need to be redeveloped for each successive domain. The capacity of BFO to serve these purposes has already been thoroughly tested in multiple application areas.

For an overview of BFO see Figure 3 and http://ontology.buffalo.edu/BFO/Reference.

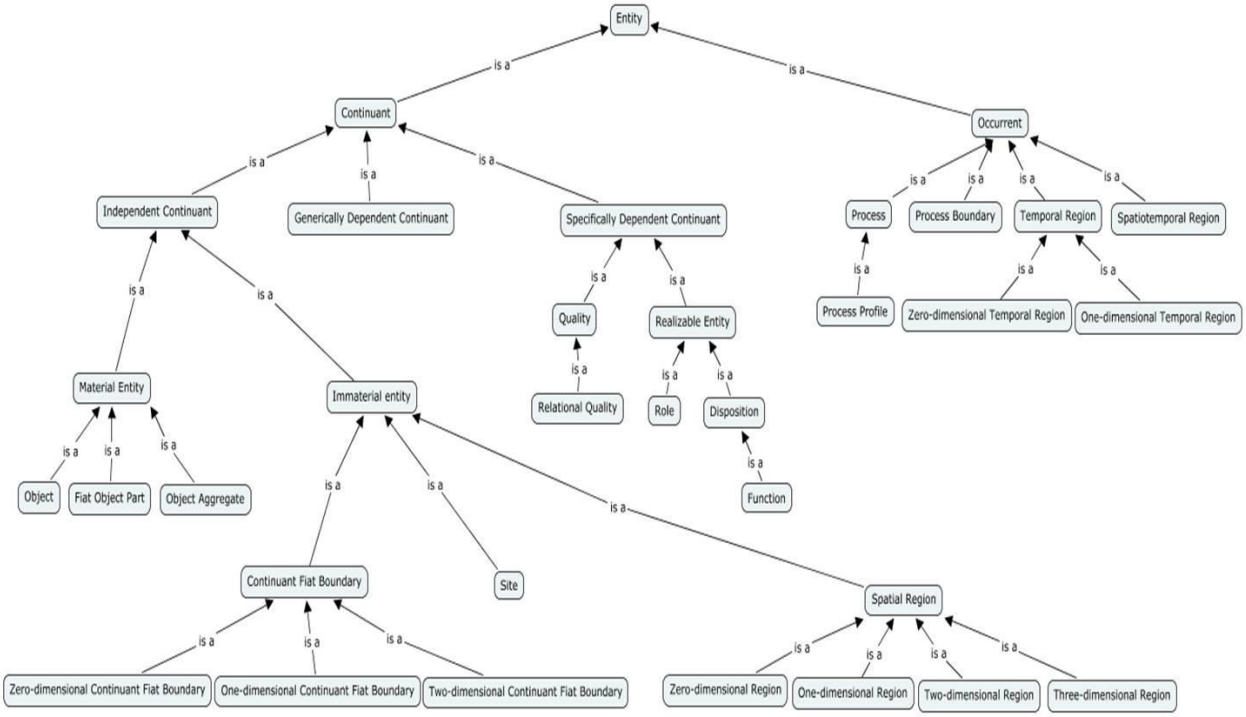

Figure 3: BFO 2.0 *is_a* Hierarchy

4.1.1 Why BFO?

There are four leading upper-level ontologies currently in use:

- Basic Formal Ontology (BFO)
- Domain Ontology for Logical and Cognitive Engineering (DOLCE)
- Suggested Upper Merged Ontology (SUMO)

- OpenCyc

We recommend use of BFO for a number of reasons, the most important of which is that, of all available upper level ontologies, 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).

One reason for the success of BFO is that it is a strict upper ontology, which means that it does not contain its own representations of physical, chemical, biological, psychological, social or other types of entities which would properly fall within mid- or lower-level domains. BFO is correspondingly a very small ontology, with a narrowly focused task: that of providing an upper ontology that could be used to support the integration of multiple heterogeneous domain ontologies. For examples of suites of modular ontologies built in these terms see [16, 34].

DOLCE [11] is closely related to BFO in its general aims and the two ontologies share a significant fraction of terms especially in their upper levels. But DOLCE has a much smaller number of users. It is based on a strategy, different from that of BFO, focusing on what it calls 'linguistic and cognitive engineering'. Those parts of DOLCE which provide content that is useful for the development of domain ontologies to serve the needs of intelligence analysts will be incorporated in the corresponding MLOs in the SE hierarchy.

SUMO, too, has proved to have value as an upper-level ontology for certain applications [13]. Unfortunately SUMO is for ULO purposes far too large, and it contains large portions of domain ontologies in areas such as biology and physics. This means that it conflicts with the requirement of modularity and thus cannot support the strategy of downward population that has proved so useful in other areas.

Cyc [12] has been applied in a number of projects over the years, but it is marred not least by its inclusion of very many terms and definitions which, because of Cyc's primary focus on formalizing what it calls 'common sense knowledge', deviate significantly from the terms and definitions employed by domain experts. The subtypes of Cyc's *partially tangible thing*, for example, include both *diisopropyl methylphosphonate* and *pay e-mail provider.* Another problem turns on the fact that Cyc does not strive for consistency among the various 'microtheories' which form its parts. Hence the very goal of creating a single consistent suite of interoperable ontologies which would capture the terminological content of domain knowledge of relevance to intelligence analysts is undermined by Cyc's own paraconsistent logical structure.

4.2 Mid-Level Ontologies

Cross-domain mid-level ontologies (MLOs) provide the ontological resources needed to ensure that where common structures are repeated in this way in multiple domains, a single set of terms and definitions will be available as common starting point for the needed more specific treatments in narrower domains thereby promoting consistency between different LLOs.

4.2.1 Examples of Mid-Level Ontologies

Person Ontology

The strategy for creating Person Reference Ontology is to select those existing LLOs and MLOs within the set of reference ontology modules whose terms are used to describe persons.

> Example LLOs: Person Name, Person Date, Hair Color, Gender, and so on.
>
> Example MLOs: Physical Characteristic (anatomy, markings, scars, …), Clothing, Skill, Ethnicity, Religion, and so on.

Some portions of these reference ontologies will form part of the Person Ontology, where they will be combined with additional terms, definitions and axioms for example asserting links between specific religions and specific ethnicities or styles of clothing. Reasoners are applied to these definitions to generate inferred types such as Male Adult, Female Infant, Husband, Father, and so on. Generating the Person Ontology by inference in this way brings a number of benefits, above all that the ontology can be re-created at regular intervals in order to incorporate content newly added to any of the constituent asserted modules within the Person subset.

Communication Ontology

Each communication event involves an agent, which initiates the communication, a message that is communicated, and a second agent (an individual or group) which receives the message. This common communication event structure is repeated in many different kinds of domains, for example in command and control messaging, emergency communication among first responders, email exchange, religious preaching, military training, and so forth.

Emergency Event Ontology

The following is a fragment of a draft Emergency Event Ontology, which is adapted from the Universal Core Semantic Layer [10], which is in turn adapted from Version 2 of the Universal Core (UCore), an information sharing initiative of the US Departments of Defense, Energy, Justice, and Homeland Security, the Intelligence Community, and a number of other national and international agencies. UCore supports the principles of the Department of Defense (DoD) and Intelligence Community (IC) Data Strategies by defining a small set of common data elements that are implemented in a lightweight information exchange schema that is shared across multiple agencies, primarily in the area of response to emergency events:

- Cyberspace Emergency Event
- Environmental Emergency Event
  - Biological Incident
  - Chemical Incident
  - Nuclear Incident
  - Oceanographic Emergency Event
- Evacuation Emergency Event
- Explosive Incident
- Financial Emergency Event
- Geopolitical Emergency Event
  - Immigration Emergency Event
  - Migration Emergency Event
  - Political Emergency Event
- Hazardous Emergency Event
  - Hazardous Spill
- Health-Related Emergency Event
  - Epidemic Emergency Event
  - Pandemic Emergency Event
  - Radiological Incident
- Infrastructure Emergency Event
  - Structural Collapse
- Natural Emergency Event
  - Avalanche
  - Earthquake
  - Flood
  - Landslide
  - Volcanic Eruption
  - Weather Emergency Event
    - Snow Ice Storm
    - Tornado
    - Tropical Hurricane
    - Tropical Storm
    - Tsunami

Wildland Fire
Military Emergency Event
National Special Security Event
Terrorist Act
Transportation Emergency Event

Some terms in this ontology may be generated by inference from definitions taken from other ontologies. For example, we can define

Evacuation Emergency Event =def. an Emergency Event that is associated with some Evacuation.

And similarly we can define 'Pandemic' and 'Endemic Emergency Event' by drawing on the definitions for 'Pandemic' and 'Endemic' from the Infectious Disease Ontology.

4.2.2 Other MLOs

Further areas provisionally identified for MLO development include information artifacts, where IAO will serve as starting point; and geospatial ontology, which will build on geospatial terminologies such as the USGS standards (http://nationalmap.gov/standards/index.html).

MLOs serve two purposes. First, they provide terminological resources that can be reused by multiple groups, both within and outside the SE community. Second, they provide a means by which the developers of LLOs can incorporate their work into the common semantic architecture – root nodes for LLOs must be drawn, as appropriate, from some salient MLO. Then, they help to ensure consistency of LLO development. The latter will in turn help to drive MLO development, since terminological needs identified at the lower level will in some cases need to be met by corresponding MLOs, which can serve as starting point for further downward population. In this way we thus secure incremental conformity of the LLOs and MLOs to the common SE architecture.

4.3 Lowest Level Ontologies

An LLO describes narrow homogeneous content. Examples are:

Person Name (with types such as: FirstName, LastName, Nickname, …)

Hair Color (with types such as Grey, Blonde, …)

Military Role (with types such as: Soldier [Role], Officer [Role], …)

Blood Type (with types: O, A-, …)

Eye Color (with types: Blue, Grey, …)

Gender (with types: Male, Female, …)

Age Group (with types: Infant, Teenager, Adult, …)

Person Date (with types: BirthDate, DeathDate, …)

Education Milestone (with types: HighSchoolGraduation, CollegeGraduation, …)

Education Date (with types: DateOfHighSchoolGraduation, …)

Criminal History (with types: FirstArrest, FirstProsecution, …)

LLOs are brought together to describe heterogeneous content in inferred ontologies along the lines described above. Such inferred ontologies are created to address the needs of particular groups of users who require specific groups of terms to be brought together in a single ontology. Our approach here will involve application of the ontobee import software already being used for this purpose for annotation of biological data [27]. We will create an ontobee console from LLOs such as those listed above, allowing the creation of inferred ontologies.

Figure 4 illustrates the multi-level organization of the SE suite of ontologies.

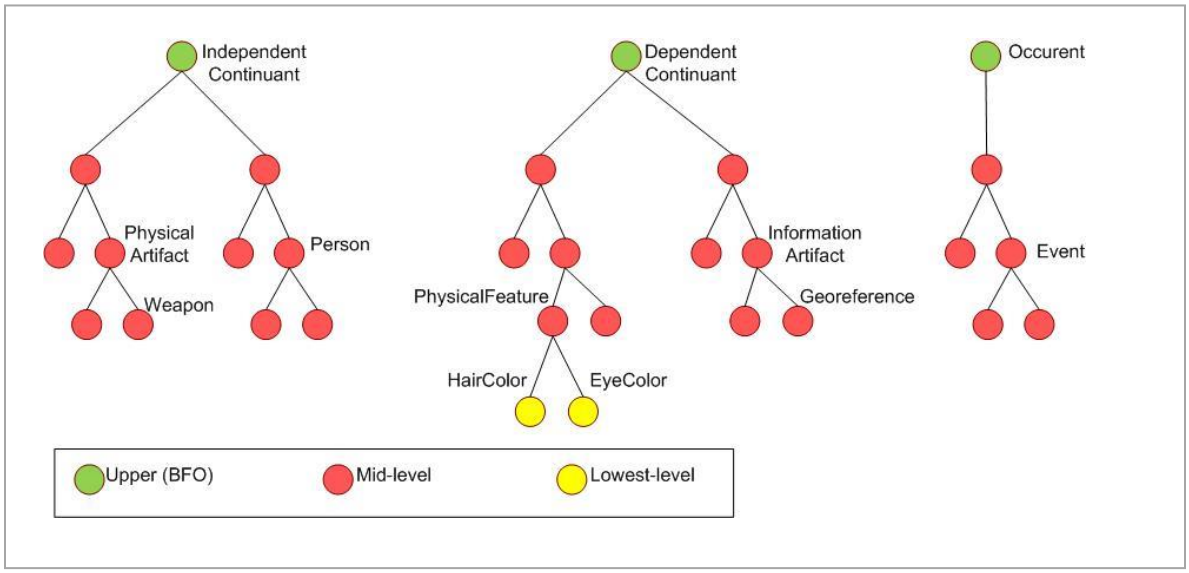

Figure 4: Organization of asserted ontologies.

4.4 Methodology for Definitions

Both in asserted and in inferred ontologies definitions of SE ontology 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 SE 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. Initially *D* will in many cases be formulated using natural-language expressions combined with logical operators. Over time, these natural-language expressions will be replaced by terms formally defined in other ontologies.

Examples:

- Natural event =def. a BFO: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 differentia, their definitions encapsulate the subsumption information relating the corresponding type within the SE ontology hierarchy 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.

Inferred Ontologies

To see why, in inferred ontologies, the monohierarchy principle can no longer be applied, consider the case where port authorities need to deal with the type *vehicle that has cargo role*. This type has both *vehicle* and *cargo* as its *is_a* parents. Similarly *explosion caused_by a terrorist organization* has as parents both *explosion* and *terrorist action.*

4.5 Combination of Top-Down, Bottom-Up and Horizontal Development Methodologies

LLOs are created following a bottom-up approach rooted in knowledge of the entities in the corresponding domain on the part of collaborating SMEs. This occurs, however, always under the top-down constraint created through the two-part definition approach, which implies always that a parent term must be identified in the ontology hierarchy before a new term can be defined. This gives rise to what we might call an ontology path dependence principle [7], whereby ontologies created earlier determine the shape of ontologies created later in ways which ensure preservation of the common SE architecture at each successive stage.

In addition to these top-down and bottom-up ("vertical") constraints, the SE strategy imposes also what we might call a 'horizontal' constraint of orthogonality in order to ensure mutual consistency and compatibility of MLOs and LLOs at each level of the asserted hierarchy.

Imagine for example, that we are concerned with a user interested in targeting ship-borne vehicular cargo. We would then have multiple asserted hierarchies as follows:

- Role -> Cargo Role -> Shipborne Cargo Role
- Role -> Target Role -> Building Target Role, Vehicle Target Role
- Physical Object -> Vehicle

together with definitions such as

- Shipborne Vehicle=def. Vehicle that has Cargo Role
- Shipborne Vehicle Target =def. Shipborne Vehicle that has Vehicle Target Role
- Land Vehicle Target =def. Land Vehicle that has Vehicle Target Role

all of which can combined into a single inferred ontology for example by using the ontobee tool.

5 Methodology for Ontology Developers

We provide a set of rules governing implementation of different aspects of the SE methodology. Conformance to these rules helps to ensure intelligibility of the ontologies to human users (who will more readily understand what terms mean), they make possible quality assurance of the SE hierarchy in each successive stage and ensure effective use of computational reasoning applied to annotated data.

5.1 The General Strategy

At the core of each ontology is a hierarchically organized collection of terms representing types on the side of reality. These types have instances, and it is these instances which are described in intelligence reports that yield the source data that is to be annotated using ontology terms.

- Instances – for example, this person, that building, this meeting, that explosion – are the sorts of things that can be observed – you can see people and buildings and meetings, hear explosions, weigh a vehicle.
- Types – for example, weapon, country, terrorist organization – are *general* or *repeatable* in the sense that they exist in many instances, while the instances themselves *exist only once*.

Examples from the Domain of Information Artifacts

Type: Terrorist Organization
Instance: Izz ad-Din al-Qassam Brigades

Type: Twitter Account Of Terrorist Organization
Instance: Twitter account of al-Qassam Brigades (http://twitter.com/AlqassamBrigade)

Type: Tweet Posted To Twitter Account of Terrorist Organization
Instance: Tweet: #Gaza@AlqassamBrigade International pressure mounts over Gaza blockade http://t.co/gbcTQWeu (12:29 AM, June 16, 2012).

5.2 Rules for Creating Ontologies

1. Consistency principle	Each new ontology should be consistent with already existing ontologies as tested by use of reasoners; to preserve consistency changes will in some cases need to be made to existing ontologies.
2. Ontology community conformance principle	Terms used in MLOs, LLOs and inferred ontologies should conform as far as possible to the common usage of SMEs. Where usage differs among different groups of SMEs, additional terms will be included in the ontology as synonyms.
3. Domain coverage principle	SMEs in each community will determine what content each MLO and LLO will need to cover.
4. Principle of low hanging fruit	Ontology development is incremental, and starts always with the simplest terms and definitions and with assertions of the simplest relations between them. Even what might seem to human beings to be very trivial ontology content is needed by computers to exploit this content.
5. Division of labor principle	Ontology development is modular. Each domain ontology should be developed by a team including SMEs in the relevant domain. SMEs should be involved also in reviewing the ontology.
6. Community feedback principle	Each domain ontology should evolve on the basis of feedback derived from those who are using the ontology for purposes of annotation. This principle ensures that the process of using the ontology contributes to its on-going improvement. Software trackers should be implemented to enable the easy transmission of such feedback from users to ontology developers.
7. Disjointness principle	Inspect each set of terms within your ontology on a single level (which means: terms having a common parent). Assert disjointness axioms where necessary (axioms asserting that the types represented given pairs of terms have no instances in common).
8. *Is_a* completeness	Fill in missing terms to ensure a complete hierarchy (which means: a hierarchy which is such that every term is connected to a root node through a chain of *is_a* relations.

5.3 Rules for Ontology Term Formation

Rule	Explanation
1. Terms name types	Each term A in an asserted ontology is a name of the type A.
2. Avoid acronyms	Use acronyms 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.
3. Avoid ellipses	Avoid abbreviations created through ellipsis, even when it is clear in context what they mean. (Compare, in the hospital ward, use of 'breast' for 'breast tumor patient'.)
4. Rule of singular nouns	All terms within an ontology should be nouns and noun-phrases that are *singular in number*. (Each term 'A' in an ontology should be viewed as shorthand for a term of the form 'the type A'.)
5. Rule of traceability to the root.	All terms in an asserted ontology should be traceable via *is_a* relations to the relevant ontology root node.
6. Avoid 'other'	Do not create terms such as 'other terrorist', 'other weapon', …
7. In reference ontologies, do not use disjunctive or conjunctive or 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.
8. 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', …
9. Do not confuse information artifacts with the reality they denote	Distinguish explicitly between a real entity and information about it. Avoid confusing for example: 'education' and 'education record' 'employment' and 'employment record'. Keep words and things separate. Do not confuse an entity with its name or ID.

10. Do not confuse reality with our knowledge about reality	Avoid terms like 'unknown terrorist', 'known terrorist', 'unclassified participant', 'participant not otherwise specified'.
11. The principle of univocity	Each term and each relational label must have exactly one meaning in all contexts. In particular, complex terms should be constructed in such a way that their constituent parts preserve their standard meanings.
12. Avoid mass nouns	Avoid mass terms ('flesh', 'fuel', 'information' ...) and 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 (thus allowing association with count expressions), as in: 'portion of meat', 'amount of fuel', 'information artifact'. (Again: this rule follows from the principle according to which each term 'A' in an asserted ontology is shorthand for a term of the form 'the type A'.)
13. 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 [15].
14. The principle of compositional term construction	If an ontology O uses terms of the form '*a* † *b*' (where '†' stands in for some term-binding operator such as 'of' or 'with') then the corresponding *a* and *b* terms should be incrementally included, either in O, or in some ontology within the SE hierarchy) [15].
15. The principle of coherence in the use of generic term-building operators	If an ontology uses in a systematic way terms of the form '*a* † *b*' (where '†', again, stands in for 'with', 'without', 'of', etc.), then it should specify clearly the syntax of '†', provide a statement of what expressions of the form '*a* † *b*' mean in terms of the meanings of '*a*' and '*b*' and expressions from the Relation Ontology, and use each such form in the same way throughout.

5.4 Rules for Definitions

1. Distinguish primitives and defined expressions	The upper ontology will include some terms and some relations which are declared as primitive, which means: they cannot be defined (on pain of infinite regress). Examples of primitive terms are: 'entity', 'process'. Examples of primitive relational expressions: 'identical_to', 'instance_of'.
2. The principle of intelligible definitions	Use definitions which are both (1) humanly intelligible (to avoid error in human use and maintenance) and (2) formally specifiable in languages such as OWL or CLIF. [14]

3. Avoid subjectivity	When formulating definitions avoid the use of phrases like 'which may ...', 'that indicates …', '… characterize …', 'an aspect of …', which invite subjective interpretation.
4. Definitions should be non-redundant	Do not include clauses in definitions which contribute nothing to the application of the definition as a specification of necessary and sufficient conditions.

Appendix I: Approach to Unique Identifiers in the SE Ontology Suite

The goal of what follows is to provide a framework for stable and homogeneous identifiers for Shared Semantic Resource (SSR) representations, and to be able to respond with providing a bounded amount of useful information for each URI denoting a term in a SSR ontology.

Background

We shall create a SSR which will contain both asserted and inferred ontologies. Each ontology in the SSR shall have a name (for example: Intelligence Ontology); an associated idspace designator (for example: SSR-IO), a **permanent ontology owl-file URI**, pointing to the most recent version of the ontology itself (for example: http://purl.ssr.org/io.owl) and a **permanent ontology URI** pointing to a page containing the most recent information about the ontology (for example: http://purl.ssr.org/io), including a link to the owl file.

Each version of the ontology shall have a version name (for example Intelligence Ontology Version 2012/06/27), and a **version URI** pointing to this version (for example http://purl.ssr.org/io/YYYY-MM-DD/io.owl). Each time a new version is released, the permanent URI will resolve to this new version. Thus the current version will at any given time have two URIs.

Each ontology term shall have a unique identifier that is a URI, including the alphanumeric ID for the term as final string. Each ontology term should point to its own webpage (for example http://purl.ssr.org /iao/iao_1000454.owl). (Topbraid can generate these automatically)

- The URIs should resolve to useful information about a term.
- The URIs should be designed so that they can be maintained over time to keep pointing to useful information.

  See for an example from OBO:

http://www.ontobee.org/browser/rdf.php?o=GO&iri=http://purl.obolibrary.org/obo/GO_0005623

SE Ontology Page URI ::= "http://purl.ssr.org /"IDSPACE

SE Ontology URI ::= "http://purl.ssr.org /"IDSPACE.owl

SE Version URI ::= "http://purl.ssr.org/YYYY-MM-DD/"IDSPACE.owl

SE Ontology Term URI (for whatever is the most recent version) ::= "http://purl.ssr.org /"IDSPACE"_" LOCALID.owl

SE Ontology Term URI (permanent URI pointing to a given term in a specific version) ::= "http://purl.ssr.org /"YYYY-MM-DD/"IDSPACE"_"LOCALID.owl

SE Ontology Relation URI (for relations not included in BFO 2.0): "http://purl.ssr.org/ro/"LOCALID.owl where 'ro' is the name of the ontology containing our special purpose relations (some of these relations may be MIREOTed from http://obofoundry.org/ro).

Good Practice Criteria for Usage of URIs

(borrowed from the Shared Name Initiative (http://sharedname.org/) as base line)

1. It must be clearly stated what the intended referent of each URI is supposed to be, i.e. that the URI denotes some particular record from some particular database.

2. Information about the URI and its referent, including such a statement, must be made available, and in order to leverage existing protocol stacks, it must be obtainable via HTTP. (We will call such information "URI documentation".)

3. URI documentation must be provided in RDF.

4. Provision of URI documentation must be an ongoing concern. The ability to provide it may have to outlive the original ontology developer's group or creator.

5. The provider of the URI documentation must be responsive to community needs, such as the need to have mistakes fixed in a timely manner.

6. URI documentation must be open so that it can be replicated and reused.

Governance Role in Allocating IDSPACEs

IDSPACEs within SSR are unique for the DSC project and must be registered with the governance board. Although IDSPACEs are case-sensitive, there will never be more than one IDSPACEs that are the same

when compared in a case-insensitive manner. Therefore, although "IO" and "io", "Io" and "iO" are different IDSPACEs, the IDSPACE "io", "Io" and "iO" will not be used as "IO" has already been allocated.

A registry of allocated IDSPACEs will be maintained. Requests for an IDSPACE should be made by sending mail to XXX. A request should include information about the ontology, including: specification of scope, name, affiliation, title and contact details of maintainer.

Current Ontology Document

The most current version of an ontology will be at the following URL, where "IDSPACE" is replaced with the IDSPACE of the given ontology in lower case.

Current OWL: http://purl.ssr.org/IDSPACE.owl

Ontology Subsets and Variants

Aside from the standard version, an ontology may provide differently packaged elements of the ontology, or ontology versions with additional contents. For example:

- Subsets of the ontology tailored to particular purposes or user communities, sometimes called subsets, slims, views, or modules.
- A version with or without inferred relations as part of it
- A version that includes more or less metadata, such as change tracking
- Pre-release work or experimental extensions

The current version of such additional artifacts should be accessible at URIs with the prefix

http://purl.ssr.org/<idspace>/<name>.owl

Where <idspace> is replaced by the IDSPACE in lower case, and <name> is the name for the artifact.

For example, IAO distributes its ontology metadata set as a distinct document, with <name> "ontology-metadata".

Versions of Ontologies

Versions are named by a date in the following format: YYYY-MM-DD. For a given version of an ontology, the ontology should be accessible at the following URL, where <idspace> is replaced by the IDSPACE in

lower case. For example, for the version of OBI released 2009-11-06, the OWL document is accessible at http://purl.obolibrary.org/obo/obi/2009-11-06/obi.owl. Ontology variants are versioned in the same manner.

URIs for a given version have the version date between <idspace> and <name>. For example:

http://purl.ssr.org/iao/ontology-metadata.owl,

for the version of date 2009-11-02, has the URI:

http://purl.ssr.org/iao/2009-11-02/ontology-metadata.owl

Eliminating Terms from an Ontology

IDs are permanent and unique; once publicly released, they can never be used again. If the term represented by an ID needs to be eliminated for some reason, the term and ID are marked with an *is_obsolete* or 'soft deletion' tag, along with meta-data explaining why the term was made obsolete and suggestions for alternatives. This prevents the ID from being reused for other terms; it provides a warning to potential users of the term in future annotations; and it also helps to preserve the value of older (legacy) annotations, in which the now obsoleted term was used.

Tracker

Each ontology has a tracker for submitting and monitoring term and other requests accessible at http://purl.ssr.org/IDSPACE/tracker. For example the OBI tracker is accessible at:

http://purl.obolibrary.org/obo/obi/tracker

References

http://en.wikipedia.org/wiki/Uniform_Resource_Identifier, a **Uniform Resource Identifier** (**URI**) consists of a string of characters used to identify or name a resource on the Internet. Such identification enables interaction with representations of the resource over a network (typically the World Wide Web) using specific protocols.

http://purl.org/, A PURL is a **P**ersistent **U**niform **R**esource **L**ocator. Functionally, a PURL is a URL. However, instead of pointing directly to the location of an Internet resource, a PURL points to an intermediate

resolution service. The PURL resolution service associates the PURL with the actual URL and returns that URL to the client. The client can then complete the URL transaction in the normal fashion. In Web parlance, this is a standard HTTP *redirect*.

http://en.wikipedia.org/wiki/Domain_name_system, the **Domain Name System** (DNS) associates various sorts of information with so-called domain names; most importantly, it serves as the "phone book" for the Internet by translating human-readable computer hostnames, e.g. *www.example.com*, into the IP addresses, e.g. *208.77.188.166*, that networking equipment needs to deliver information.

Acknowledgments: The framework described in the above is modeled on that provided for the OBO Foundry at http://obofoundry.org/id-policy.shtml) which was drafted and implemented as part of the development of the OBI project. Authors: Alan Ruttenberg, Melanie Courtot and Chris Mungall. Thanks to Jonathan Rees, Bill Bug, Colin Batchelor, David Osumi-Sutherland, Duncan Hull, Peter Robinson, Michel Dumontier, the OBO coordinators and the OBI Consortium for their help.

Appendix II: Methodology for Annotators

The prime purpose of SE ontologies is to provide natural language terms which are used to enhance source data through tagging or annotations. If the same terms can be used over and over again for annotating diverse data, these data become more easily searchable and easier to integrate and analyze. Annotation is an arms-length approach, which leads to the creation of a separate Semantic Layer. The native data and data-semantics themselves remain unaffected.

Annotations can be applied to databases, but they can be applied also to other information articles, including ontologies themselves. We can also annotate semi-structured data, for example in XML files and in parsed text. Here, however, we focus exclusively on the annotation of data models – that is on structured data – through association of ontology terms with table column names.

Not all the content of a data resource needs to be annotated. For example, we may have data that is not meant to be shared, or, data that is needed for maintaining the store (usually various surrogate keys and identifiers).

There are two ways in which data can be enhanced by assigning ontology terms to the source data table headings:

1. through manual assignment by human annotators on the basis of inspection of data models
2. through computational assignment

Again we focus here on 1., leaving to a later time the discussion of machine-aided annotation.

Role of Definitions

Ontology terms need to be associated with human understandable and logical definitions. The former are needed to ensure that human annotators correctly and consistently understand the meanings of the terms they use in annotations. (They serve also to ensure that there is agreement on the meaning of the term before it is included in the ontology.) Human understandable definitions provide also the basis for the logical definitions that are needed to allow the application of computational tools, both in generating annotations computationally through use of NLP tools applied to free text sources, and also in processing the enhanced data for purposes of retrieval and analysis.

Benefits of Modularity

Modularity brings benefits also to the annotator, by providing easy surveyability of the SE resources to be used in annotations and by ensuring a clean division of labor amongst domain experts. Modularity also promotes consistency of the results of distributed annotation efforts. If A is annotating using ontology O_1 and B is annotating using ontology O_2, then, because O_1 and O_2 do not overlap (either in their terms or in their domain coverage), the annotations made by A and by B are in almost every case trivially compatible. Thus modularity brings *additivity of annotations:* even when multiple independently developed ontologies are used, because the ontologies do not overlap, use of multiple ontologies will not bring any additional inconsistencies into the annotation file.

Rules for Annotators

Ontology terms are general expressions referring to *types* in reality: weapon, meeting, facility … Such types are instantiated by *particulars* in reality – this weapon #BU-433, this meeting #M4-11-2012, this facility 1256 C, and so on. Particulars are what is observed, reported, planned.

For using ontology terms in annotations, three sorts of cases must be distinguished, corresponding to three sorts of references of data model headings:

a. to types in reality;

b. to collections of instances drawn on the basis of data-processing convenience;

An annotation is in either case an assertion of the form:

Data model header A can be classified using ontology term O:nnnn derived from ontology O.

In other words the annotation asserts that the type or collection to which the entities referred to under A belong can be classified using O:nnnn; all relevant instances are instances of the type O:nnnn.

Note that we discuss here exclusively annotation at the level of types. Instance-data, too, may be enhanced through annotation (for example data to the effect that *Ahmed's brother* is also *Suleema's sister*. Here, however, we leave consideration of this matter to one side.

Evidence, Quality, Provenance

To annotate an ontology term T to a term D of a native data model is to assert that Ds are best classified as Ts. Such an assertion presupposes that there is evidence that the referent of D can be best classified using this ontology term T.

To enable quality control of annotations the *source* of the evidence for each annotation should thus be associated with the annotation, together with an annotation pertaining to the *quality* of the source.

Examples of possible terms for an evidence/quality ontology are:

Direct report

Human analyst inference

Computational inference

(Compare the Gene Ontology Evidence Codes at http://www.geneontology.org/GO.evidence.shtml.)

Such terms will form part of a larger collection of terms relating to metadata, containing among evidence codes, terms relating provenance, and so forth.

Formatting of the Annotation Resource

The precise format of the annotation file to be created for each data source is a question of implementation. However, we envisage each annotation in the file as having at least the following five components:

A = some data model heading

B = some ontology term (of the form O:nnnnn, where 'O' is the name of the ontology and nnnn is the term ID)

C = some relation between A and B (almost always the relation will be *is_a*) D = a reference to the source from which the data item is taken

D = reference to the source of (evidence for) the annotation

E = a term from a controlled provenance vocabulary indicating how the annotation is supported

Feedback Loop from Annotators to Ontology Developers

Maintenance of ontologies over time depends on establishing a virtuous feedback cycle from the annotators who use them [5]. The process of annotation may bring new results into conflict with the knowledge captured in existing ontologies. Thus each ontology must be associated with a tracker, linked from the software tool used by annotators, allowing those using its terms for annotations to provide feedback to the ontology developers (a) concerning errors, e.g. in the definitions, (b) concerning gaps (areas where the ontology does not meet your annotation needs).

In this way a feedback loop is created between annotators and ontology developers. If you are annotating a source, and find:

- that the ontologies you are using do not contain a term you need,
- or that they define a specific term incorrectly
- or that a new ontology resource is needed

use the tracker to send feedback to the ontology developers

The process of annotation may bring new results into conflict with the knowledge captured in existing ontologies. Identifying these conflicts leads to improvements in the ontologies, which thereby embody accumulated knowledge from many annotators and intelligence analysts that is consistently improving over time.

Simple Rules for Annotators

1. Identify the relevant data item to be annotated. (This may be just a fragment of a larger item, for example just one word in a sentence).

2. Identify the appropriate ontology for the annotation.

3. Identify the appropriate term in the ontology, following the general rule: always annotate to the term that is at the lowest level in the hierarchy but still correct for this instance.

4. Even trivial annotations (for example annotating ‘person’ to the data item ‘Ahmed’s relative’) can be computationally useful).

5. A single data item may be annotated by multiple annotators using terms from multiple distinct ontologies.

6. Evaluate annotations for consistency (both for a single annotator, and between multiple annotators).

7. Associate source, quality and provenance data to the annotation.

8. Harvest knowledge emerging from the annotation effort to refine and extend the ontology resources available for use in future annotation cycles.

Appendix III: Proposed SE Governance Process

The suite of SE ontologies must evolve in a coordinated way, something that is difficult to achieve in diverse and dynamic environment in which ontologies are developed and used by distributed teams.

The governance rules and processes that are needed to ensure such coordination are described below. They are to be realized in accordance with authoritative DoD instructions, specifically:

- CJCSI 5705.01D retrieved 15 January 2012: http://www.dtic.mil/doctrine/new_pubs/cjcsi5705_01d.pdf
- DoDI 5025.12 retrieved 15 January 2012: http://www.dtic.mil/whs/directives/corres/pdf/502512p.pdf

The stages of the ontology life cycle at which governance policies are needed include:

1. Ontology design and creation
2. Ontology registration and dissemination
3. Ontology maintenance
    - Change management
    - Ontology versioning
2. Application by users of ontologies
    - Use of ontologies in creating annotations
    - Ensuring feedback from users to developers

The strategy we propose requires two sorts of governance units and associated policies:

1. A Coordinating Board, whose role is:
    a. to ensure conformance of the ontologies to the SE architecture
    b. to minimize redundancy of effort by maximizing reuse of SE ontologies and of authoritative external sources
    c. to ensure broad spectrum dissemination of on-going ontology efforts through ontology registration and information sharing policies
2. An Editorial Board for each reference ontology module which will
    a. include one or more relevant SMEs

b. nominate a representative to serve as liaison with the Coordinating Board

The governance process draws directly on the hierarchical architecture of the asserted ontologies. Wherever possible this structure will be used to ensure that the relevant policies are satisfied automatically by means of software. (Since the inferred ontologies are created by very simple software tools, quality assurance for these ontologies can itself be realized computationally, however, human review is recommended.) Each reference ontology is based on an asserted hierarchy created via downward population from the ontology above it in the hierarchy. Where reference ontologies need to overlap in their domains – for example where a Crime Ontology needs to use terms from a Property Ontology – then the terms in question will have a single official home in one ontology (with the corresponding Namespace and Local ID), and be imported into the other ontology for example using the ontofox tool while preserving these identifiers. Thus it is important that when a new term is introduced in an area where two ontologies overlap, agreement is reached among the two sets of developers as to who owns (and thus has responsibility for logically defining) this term. In this way, when changes are made in one ontology, for example in logical definitions, then these changes are automatically transmitted to all the ontologies which import this term or use it either in a definition or as a constituent in some more compound expression.

Appendix IV: A Repeatable Process for Creation of Mid-Level SE Ontologies

We outline here an expanded iterative process which complements the material presented in section 0 above:

1. Identify the entities represented by the terms in salient databases, and create a table of entities organized by level of generality (thus for example the term the *land vehicle* stands in a *less general than* (henceforth '*is_a*') relation to the term *vehicle*).

2. Work to create an initial set of ~50 most commonly used terms corresponding to types in the salient portions of reality (thus eliminating those terms which are artifact of a given data resource, following the rule: do not confuse information artifacts with the reality they denote.

3. Check for pre-existing ontology resources that can be reused to represent these types, drawing on UCore SL, Command and Control Ontology and related work, field manuals, ontologies developed in other areas of the IC

4. Fill in gaps with new terms.

5. Arrange these terms into an informal *is_a* hierarchy according to the principle:

   A *is_a* B =def. A and B are types & every instance of A is an instance of B

6. Fill in missing terms to ensure a complete hierarchy (which means: a hierarchy conforming to the traceability to the root principle; every term is connected to a root node through a chain of *is_a* relations).

7. Review the resultant heirarchy with domain specialists and repository data managers and identify gaps and errors; add new terms and correct as necessary; add necessary disjointness axioms.

8. Import Basic Formal Ontology (BFO) and salient MLOs, and identify the most general types to which the entities discovered in prior steps belong.

9. Revise the classification of the corresponding entities to allow downward population from the result of step 8, which means classifying all entities according to the tripartite division of *dependent* and *independent continuants* on the one hand, and *occurrents* on the other.

10. Fill in gaps where necessary to ensure that each hierarchy is *is_a* complete (consists of continuous branches from every term in the hierarchy to the topmost node).

11. Check whether terms in the resultant hierarchy belong to existing ontologies and if so whether they are provided there with adequate definitions. If yes, consider link to the corresponding external ontologies, rather than recreating redundant content.

12. Add further relations taken over from BFO 2.0, for example:

    - **Part_of**(*x*, *y*), for: individual *x* is part of individual *y*
    - **Inheres**(*x*, *y*), for: individual *x* inheres in individual *y*
    - **Precedes**(*x*, *y*), for: individual process *x* precedes individual process *y*
    - **Has_Participant**(*x*, *y*), for: individual thing *y* participates in individual occurrent *x*
    - **Has_Agent**(*x*, *y*), for: individual thing *y* is agent of individual occurrent *x*
    - **Realizes**(*x*, *y*), for: individual process *x* realizes individual function *y*

13. Create a Protégé-OWL ontology file and create a beta version of the ontology in OWL format incorporating all terms and relations identified in the above

14. Review this beta version according to

    - User acceptability
    - Accurate representation of the domain as validated by SMEs
    - Support for computational tasks
    - Compatibility with existing SE ontologies
    - Positive value to intelligence analysts
    - Consistency with definitions created according to the two-part methodology for definition creation.

15. Create an alpha version of the ontology and release it for use in annotation.

References

1. Barry Smith, "Mereotopology: A Theory of Parts and Boundaries", *Data and Knowledge Engineering*, 20 (1996), 287–303.

2. Pierre Grenon and Barry Smith, "SNAP and SPAN: Towards Dynamic Spatial Ontology", *Spatial Cognition and Computation*, 4: 1 (March 2004), 69–103.

3. Barry Smith and Pierre Grenon, "The Cornucopia of Formal-Ontological Relations", *Dialectica,* 58: 3 (2004), 279–296.

4. Barry Smith, Werner Ceusters, Bert Klagges, Jacob Köhler, Anand Kumar, Jane Lomax, Chris Mungall, Fabian Neuhaus, Alan Rector and Cornelius Rosse, "Relations in Biomedical Ontologies", *Genome Biology* (2005), 6 (5), R46.

5. David P. Hill, Barry Smith, Monica S. McAndrews-Hill, Judith A. Blake, "Gene Ontology Annotations: What they mean and where they come from", *BMC Bioinformatics*, 2008; 9(Suppl 5): S2.

6. Thomas Bittner, Maureen Donnelly and Barry Smith, "A Spatio-Temporal Ontology for Geographic Information Integration", *International Journal for Geographical Information Science,* 23 (6), 2009, 765–798.

7. Barry Smith and Werner Ceusters, "Ontological Realism as a Methodology for Coordinated Evolution of Scientific Ontologies", *Applied Ontology*, 5 (2010), 139–188.

8. Fabian Neuhaus, Pierre Grenon and Barry Smith, "A Formal Theory of Substances, Qualities, and Types", Achille Varzi and Laure Vieu (eds.), *Formal Ontology and Information Systems. Proceedings of the Third International Conference (FOIS 2004)*, Amsterdam: IOS Press, 2004, 49–58.

9. Barry Smith, "The Logic of Biological Classification and the Foundations of Biomedical Ontology", in Petr Hájek, Luis Valdés-Villanueva and Dag Westerståhl (ed.), *Logic, Methodology and Philosophy of Science. Proceedings of the 12th International Conference*, London: King's College Publications, 2005, 505–520.

10. Barry Smith, Lowell Vizenor and James Schoening, "Universal Core Semantic Layer", *Ontology for the Intelligence Community*, Proceedings of the Third OIC Conference, George Mason University, Fairfax, VA, October 2009, CEUR Workshop Proceedings, vol. 555.

11. Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., and Schneider L. (2002). "Sweetening ontologies with DOLCE", *Knowledge Engineering and Knowledge Management: Ontologies and the Semantic Web*, Berlin: Springer (LNCS Vol. 2473), 223–233.

12. Lenat, D. (1995). "CYC: a large-scale investment in knowledge infrastructure", *Communications of the ACM Archive*, *38* (11), 33–38.

13. Niles, I. and Pease, A. (2001). "Towards a standard upper ontology". In: C. Welty and B. Smith, editors, *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems* (FOIS), ACM Press, 2–9.

14. Jacob Köhler, Katherine Munn, Alexander Rüegg, Andre Skusa, Barry Smith, "Quality Control for Terms and Definitions in Ontologies and Taxonomies", *BMC Bioinformatics*, 2006, 7, 212.

15. Barry Smith, "Against Idiosyncrasy in Ontology Development", in B. Bennett and C. Fellbaum (Eds.), *Formal Ontology in Information Systems* (FOIS 2006), Amsterdam: IOS Press, 2006, 15–26.

16. Barry Smith, et al., "The OBO Foundry: Coordinated Evolution of Ontologies to Support Biomedical Data Integration", *Nature Biotechnology*, 25 (11), November 2007, 1251–1255.

17. Michael Ashburner, *et al*. "Creating the Gene Ontology Resource: Design and Implementation", *Genome Research*, 2001, 11, 1425–1433.

18. Rector, A. L. "Modularisation of Domain Ontologies Implemented in Description Logics and Related Formalisms including OWL". *Proceedings of the 2nd International Conference on Knowledge Capture*, ACM, 2003, 121–128

19. David Salmen, Tatiana Malyuta, Alan Hansen, Shaun Cronen, Barry Smith, "Integration of Intelligence Data through Semantic Enhancement", *Proceedings of the Conference on Semantic Technology in Intelligence, Defense and Security* (STIDS), George Mason University, Fairfax, VA, November 16-17, 2011, CEUR, Vol. 808, 6–13.

20. OWL 2 Reasoners, http://www.w3.org/2007/OWL/wiki/Implementations.

21. OWL 2 Web Ontology Language Document Overview, W3C Recommendation 27 October 2009, http://www.w3.org/TR/owl-overview/

22. Chairman of the Joint Chiefs of Staff Instruction. J2 CJCSI 3340.02A, http://www.dtic.mil/cjcs_directives/cdata/unlimit/3340_02.pdf

23. Protégé-OWL User Documentation, http://protege.stanford.edu/doc/users.html.

24. Barry Smith, Tatiana Malyuta, David Salmen, William Mandrick, Kesny Parent, Shouvik Bardhan, Jamie Johnson, Ontology for the Intelligence Analyst, *Crosstalk: The Journal of Defense Software Engineering*, November/December 2012, 18-25.

25. Joint Publication 1, Doctrine for the Armed Forces of the United States, Chairman of the Joint Chiefs of Staff. Washington, DC. 20 March 2009.

26. Joint Electronic Library: The Joint Publications, http://www.dtic.mil/doctrine/new_pubs/jointpub.htm

27. Xiang Z, Mungall C, Ruttenberg A, He Y. "Ontobee: A Linked Data Server and Browser for Ontology Terms". *International Conference on Biomedical Ontologies* (ICBO), University at Buffalo, NY, July 26-30, 2011, 279–281.

28. *Common Logic – A Framework for a Family of Logic-Based Languages*, ed. Harry Delugach. ISO/IEC JTC 1/SC 32N1377, International Standards Organization Final Committee Draft, 2005-12-13.

29. Topbraid Composer, http://www.topquadrant.com/products/TB_Composer.html.

30. Principles of Good Practice for Managing RDF Vocabularies and OWL Ontologies, http://www.w3.org/2006/07/SWD/Vocab/principles.

31. Marianne Shaw, Landon T Detwiler, James F Brinkley, Dan Suciu, and Jose L V Mejino, "Generating Application Ontologies from Reference Ontologies", *Proceedings, American Medical Informatics Association Fall Symposium*, 2008, 672-676.

32. James Malone and Helen Parkinson, Reference and Application Ontologies, http://ontogenesis.knowledgeblog.org/295.

33. Pavel Shvaiko and Jeróme Euzenat, "Ontology Matching: State of the Art and Future Challenges", *IEEE Transactions on Knowledge and Data Engineering*, 10(10), in press.

34. Fahim T. Imam, Stephen D. Larson, Anita Bandrowski, Jeffery S. Grethe, Amarnath Gupta, and Maryann E. Martone, "Development and use of Ontologies Inside the Neuroscience Information Framework: A Practical Approach", *Frontiers in Genetics*, 2012; 3: 111.

35. Barry Smith, Tatiana Malyuta, William S. Mandrick, Chia Fu, Kesny Parent, Milan Patel, "Horizontal Integration of Warfighter Intelligence Data. A Shared Semantic Resource for the Intelligence Community", *Proceedings of the Conference on Semantic Technology in Intelligence, Defense and Security* (STIDS), George Mason University, Fairfax, VA, October 23-25, 2012, CEUR 996, 112-119.