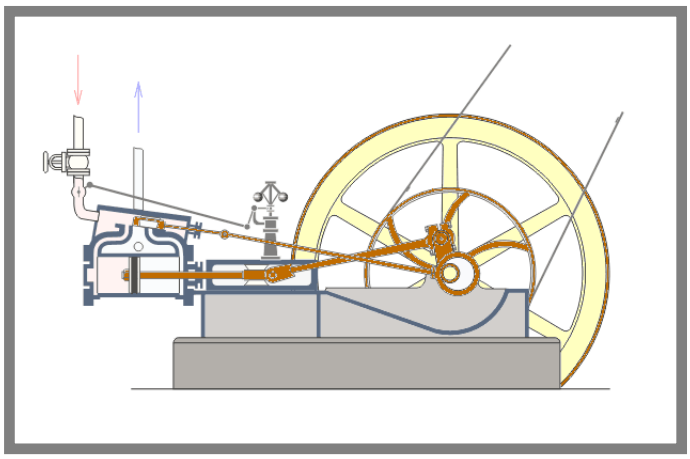
Creating a Platform for Knowledge-based Machine Intelligence

by Mike Bergman - Monday, September 21, 2015

http://www.mkbergman.com/1892/creating-a-platform-for-knowledge-based-machine-intelligence/



Practical and Reusable Designs to Make Knowledge Bases Computable

Wikipedia is a common denominator in question answering and commercial natural language applications that leverage artificial intelligence. Witness Siri, Watson, Cortana and Google Now, among others. DBpedia is a structured data representation of Wikipedia that makes much of this content machine readable. Wikidata is a multilingual repository of 18 million structured entities now feeding the Wikipedia ecosystem. The availability of these sources is remaking and accelerating the role of knowledge bases in powering the next generation of artificial intelligence applications. But much, much more is possible.

All of these noted knowledge bases lack a comprehensive and coherent knowledge structure. They are not computable, nor able to be reasoned over or inferenced. While they are valuable resources for structured data and content, the vast potential in these storehouses remains locked up. Yet the relevance of these sources to drive an artificial intelligence platform geared to data and content is profound.

And what makes this potential profound? Well, properly structured, knowledge bases can provide the features and generation of positive and negative training sets useful to machine learning. Coherent organization of the knowledge graph within the KB's domain enables various forms of reasoning and inference, further useful to making fine-grained recognizers, extractors and classifiers applicable to external knowledge. As I have pointed out before with regard to knowledge-based artificial intelligence (or KBAI) [1], these capabilities can work to extract still more accurate structure and knowledge from the knowledge base, creating a virtuous circle of still further improvements.

In all fairness, the Wikipedia ecosystem was not designed to be a computable one. But the free and open access to content in the Wikipedia ecosystem has sparked an explosion of academic and commercial interest in using this knowledge, often in DBpedia machine-readable form. Yet, despite this interest and more than 500 research papers in areas leveraging Wikipedia for AI and NLP purposes [2], the efforts

remain piecemeal and unconnected. Yes, there is valuable structure and content within the knowledge bases; yes, they are being exploited both for high-value bespoke applications and for small research projects; but, across the board, these sources are not being used or leveraged in anything approaching a systematic nature. Each distinct project requires anew its own preparations and staging.

And it is not only Wikipedia that is neglected as a general resource for AI and semantic technology applications. One is hard-pressed to identify any large-scale knowledge base, available in electronic form, that is being sufficiently and systematically exploited for AI or semantic technology purposes [3]. This gap is really rather perplexing. Why the huge disconnect between potential and reality? Could this gap somehow be related to also why the semantic technology community continues to bemoan the lack of "killer apps" in the space? Is there something possibly even more fundamental going on here?

I think there is.

We have spent eight years so far on the development and refinement of UMBEL [4]. It was intended initially to be a bridge between unruly Web content and reasoning capabilities in Cyc to enable information interoperability on the Web [5]; an objective it still retains. Naturally, Wikipedia was the first target for mapping to UMBEL [6]. Through our stumbling and bumbling and just serendipity, we have learned much about the specifics of Wikipedia [6], aspects of knowledge bases in general, and the interface of these resources to semantic technologies and artificial intelligence. The potential marriage between Cyc, UMBEL and Wikipedia has emerged as a first-class objective in its own right.

What we have learned is that it is not any single thing, but multiple things, that is preventing knowledge bases from living up to their potential as resources for artificial intelligence. As I trace some of the sources of our learning below, note that it is a combination of conceptual issues, architectural issues, and terminological issues that need to be overcome in order to see our way to a simpler and more responsive approach.

The Learning Process Began with UMBEL's SuperTypes

Shortly after the initial release of UMBEL we undertook an effort in 2009 to split it into a number (initially 33) of mostly disjoint "super types" [7]. This logical segmentation was done for practical reasons of efficiency and modularity. It forced us to consider what is a "concept" and what is an "entity", among other logical splits. It caused us to inspect the entire UMBEL knowledge space, and to organize and arrange and inspect the various parts and roles of the space.

We began to distinguish "attributes" and "relations" as separate from "concepts" and "entities". Within the clustering of "entities" we could also see that some things were distinct individuals or entity instances, while other terms represented "types" or classes of like entities. At that time, "named entity" was a more important term of art than is practiced today. In looking at this idea we noted [7]:

The intuition surrounding “named entity” and nameable “things” was that they were discrete and disjoint. A *rock* is not a *person* and is not a *chemical* or an *event*. ... some classes of things could also be treated as more-or-less distinct nameable “things”: *beetles* are not the same as *frogs* and are not the same as *rocks*. While some of these “things” might be a true individual with a discrete name, such as Kermit the Frog, or The Rock at Northwestern University, most instances of such things are unnamed. . . . The “nameability”

(or logical categorization) of things is perhaps best kept separate from other epistemological issues of distinguishing *sets*, *collections*, or *classes* from *individuals*, *members* or *instances*.

Because we were mapping Cyc and Wikipedia using UMBEL as the intermediary, we noticed that some things were characterized as a class in one system, while being an instance in the other [8]. In essence, we were learning the vocabulary of knowledge bases, and beginning to see that terminology was by no means consistent across systems or viewpoints.

This reminds me of my experience as an undergraduate, learning plant taxonomy. We had to learn literally hundreds of strange terms such as *glabrous* or *hirsute* or *pinnate*, all terms of art for how to describe leaves, their shapes, their hairiness, fruits and flowers and such. What happens, though, when one learns the terminology of a domain is that one's eyes are opened to see and distinguish more. What had previously been for me a field of view merely of various shades of green and shrubs and trees, emerged as distinct species of plants and individual variants that I could clearly discern and identify. As I learned nuanced distinctions I begin to be able to see with greater clarity. In a similar way, the naming and distinguishing of things in our UMBEL SuperTypes was opening up our eyes to finer and more nuanced distinctions in the knowledge base. All of this striving was in order to be able to map the millions and millions of things within Wikipedia to a different, coherent structure provided by Cyc and UMBEL.

ABox - TBox and Architectural Basics

One of the clearest distinctions that emerged was the split between the TBox and the ABox in the knowledge base, the difference between schema and instances. Through the years I have written many articles on this subject [9]. It is fundamental to understand the differences in representation and work between these two key portions of a knowledge base.

Instances are the specific individual things in the KB that are relevant to the domain. Instances can be many or few, as in the millions within Wikipedia, accounting for more than 90% of its total articles. Instances are characterized by various types of structured data, provided as key attribute-value pairs, and which may be explained through long or short text descriptions, may have multiple aliases and synonyms, may be related to other instances via type or kind or other relations, and may be described in multiple languages. This is the predominant form of content within most knowledge bases, perhaps best exemplified by Wikipedia.

The TBox, on the other hand, needs to be a coherent structural description of the domain, which expresses itself as a knowledge graph with meaningful and consistent connections across its concepts. Somewhat irrespective of the number of instances (the ABox) in the knowledge base, the TBox is relatively constant in size given a desired level of descriptive scope for the domain. (In other words, the logical model of the domain is mostly independent from the number of instances in the domain.)

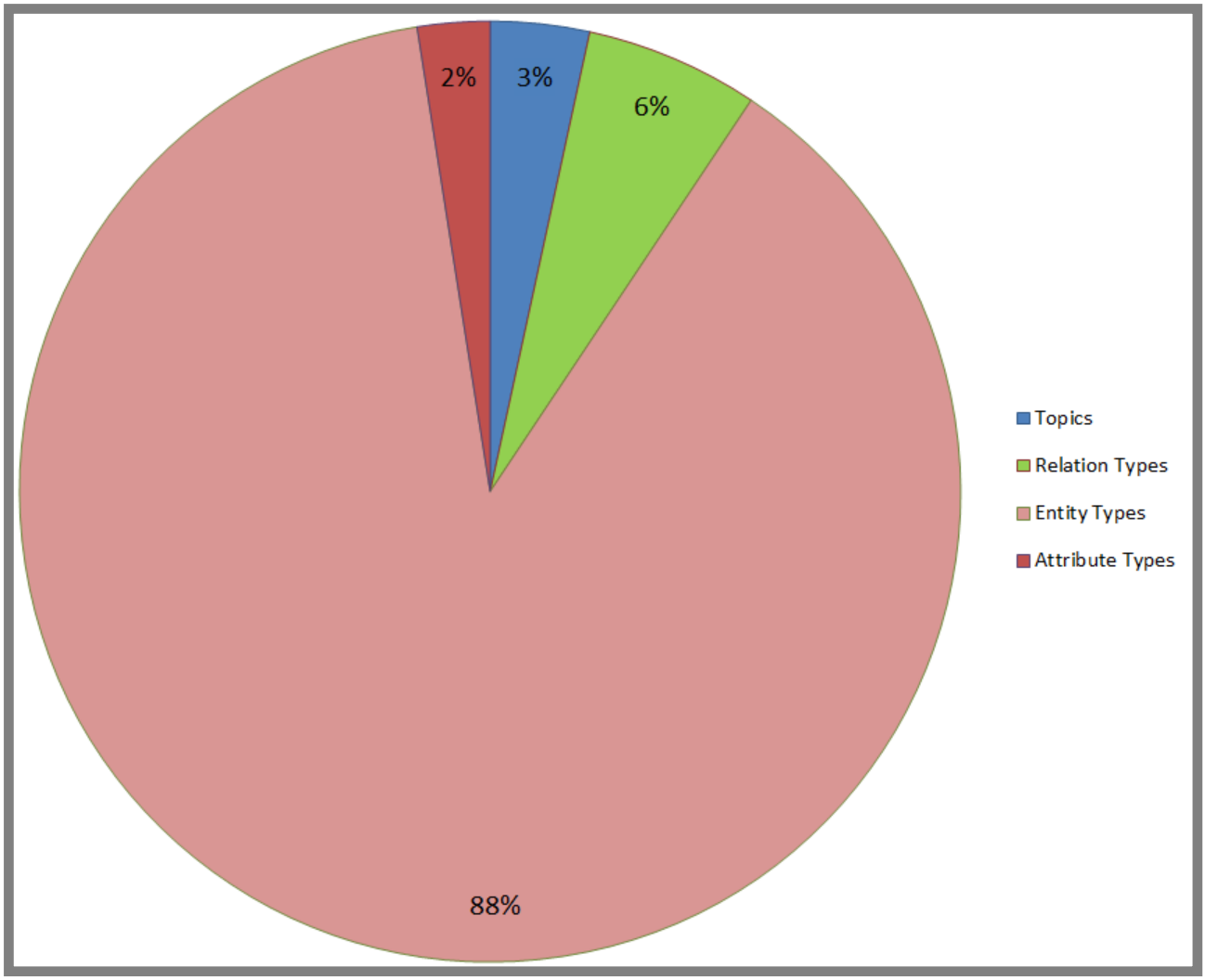
For a reference structure such as UMBEL, then, the size of its ontology (TBox) can be much smaller and defined with focus, while still being able to refer to and incorporate millions of instances, as is the case for Wikipedia (or virtually any large knowledge base). Two critical aspects for the TBox thus emerge. First, it must be a coherent and reasonable "brain" for capturing the desired dynamics and relationships of

the domain. And, second, it must provide a robust, flexible, and expandable means for incorporating instance records. This latter "bridging" purpose is the topic of the next sub-section.

The TBox-ABox segregation, and how it should work logically and pragmatically, requires much study and focus. It is easy to read the words and even sometimes to write them, but it has taken us many years of practice and much thought and experimentation to derive workable information architectures for realizing and exploiting this segregation.

I have previously spelled out seven benefits from the TBox-ABox split [10], but there is another one that arises from working through the practical aspects of this segregation. Namely, an effective ABox-TBox split compels an understanding of the roles and architecture of the TBox. It is the realization of this benefit that is the central basis for the insights provided in this article.

We'll be spelling out more of these specifics in the sections below. These understandings help us define the composition and architecture of the TBox. In the case of the current development version of UMBEL [11], here are the broad portions of the TBox:



Distribution of Types in the UMBEL TBox

Structures (types) for organizing the entities in the domain constitute nearly 90% of the work of the TBox. This reflects the extreme importance of entity types to the "bridging" function of the TBox to the ABox.

Probing the Concept of 'Entities'

Most of the instances in the ABox are entities, but what is an "entity"? Unfortunately, that is not universally agreed. In our parlance, an "entity" and related terms are:

- The basic, real things in our domain of interest: **entities**
- The way we characterize and describe those individual things: **attributes**
- The way we describe connections between two or more of those things: **relations**, and
- Aggregations or collections or classes of similar entities, which also share some essence: **entity types**.

We no longer use the term **named entities**, though nouns with proper names are almost always entities. By definition, entities can not be topics or types and entities are not data types. Some of the earlier typologies by others, such as Sekine [12], also mix the ideas of attributes and entities; we do not. Lastly, by definition, entity types have the same attribute "slots" as all type members, even if no data is assigned in many or most cases. The glossary presents a complete compilation of terms and acronyms used.

The role for the label "entity" can also refer to what is known as the root node in some systems such as SUMO [13]. In the OWL language and RDF data model we use, the root node is known as "thing". Clearly, our use of the term "entity" is much different than SUMO and resides at a subsidiary place in the overall TBox hierarchy. In this case, and frankly for most semantic matches, equivalences should be judged with care, with context the crucial deciding factor.

Nonetheless, most practitioners do not use "entity" in a root manner. Some of the first uses were in the Message Understanding Conferences, especially MUC-6 and MUC-7 in 1995 and 1997, where competitions for finding "named entities" were begun, as well as the practice of in-line tagging [14]. However, even the original MUC conferences conflated proper names and quantities of interest under "named entities." For example, MUC-6 defined person, organization, and location names, all of which are indeed entity types, but also included dates, times, percentages, and monetary amounts, which we define as attribute types.

It did not take long for various groups and researchers to want more entity types, more distinctions. BBN categories, proposed in 2002, were used for question answering and consisted of 29 types and 64 subtypes [15]. Sekine put forward and refined over many years his Extended Entity Types, which grew to about 200 types [12]. But some of these accepted ‘*named entities*‘ are also written in lower case, with examples such as rocks (‘gneiss’) or common animals or plants (‘daisy’) or chemicals (‘ozone’) or minerals (‘mica’) or drugs (‘aspirin’) or foods (‘sushi’) or whatever. Some deference was given to the idea of Kripke's "rigid designators" as providing guidance for how to identify entities; rigid designators include proper names as well as certain natural kind of terms like biological species and substances. Because of these blurrings, the nomenclature of "named entities" began to fade away.

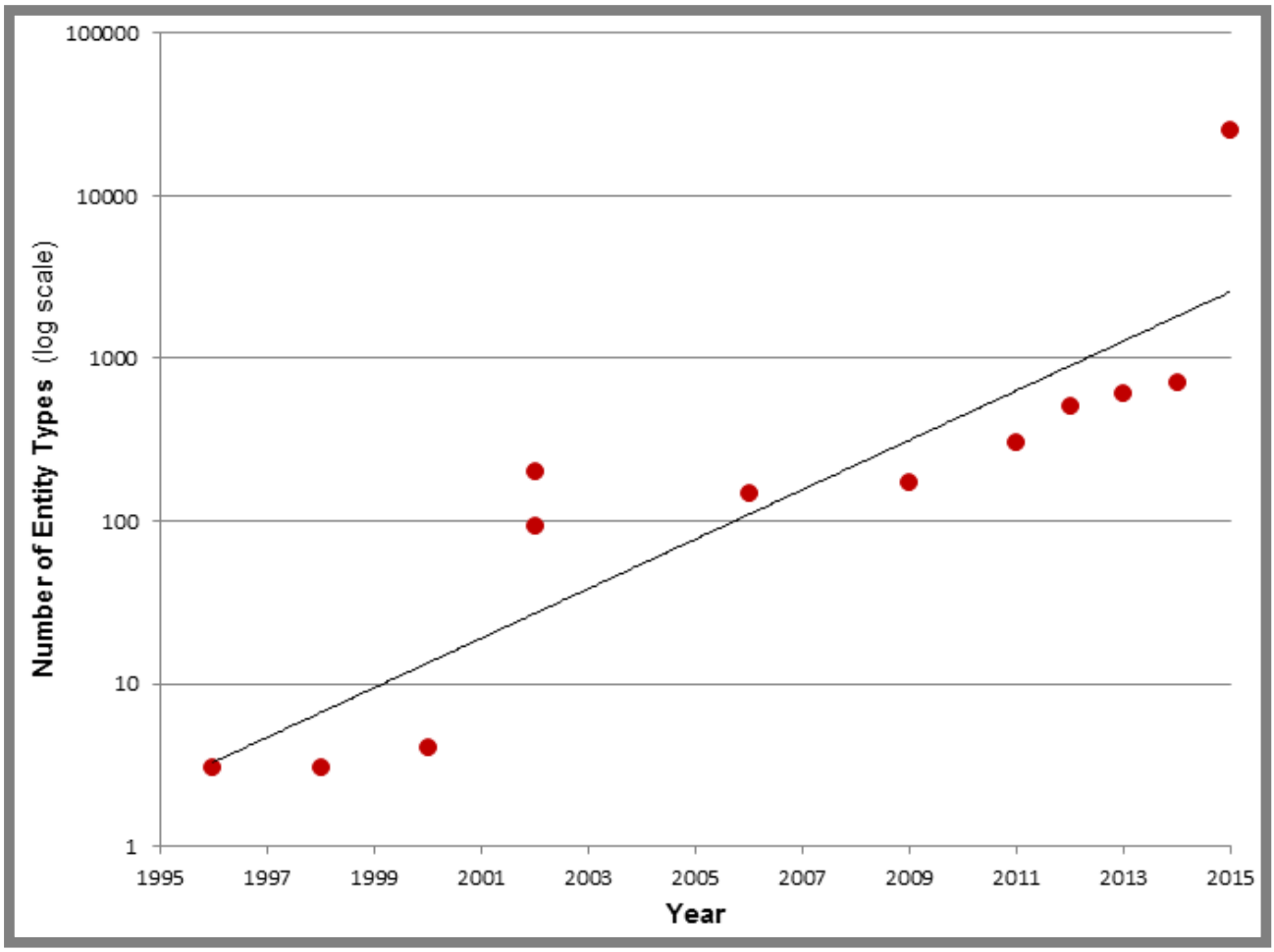
But it did not take but a few years where the demand was for "fine-grained entity" recognition, and scope

and numbers of types continued to creep up. Here are some additional data points to what has already been mentioned:

- DBpedia Ontology: 738 types [16]
- schema.org: 636 types [17]
- YAGO: 505 types; see also HYENA [18]
- Lee et al.: 147 types [19]
- FIGER: 112 types [20]
- Gillick: 86 types [21]
- OpenCalais: 42 types [22]
- GeoNames: 654 "feature codes" [23]
- Nadeau: ~100 types [24].

Lastly, the new version of UMBEL has 25,000 entity types, in keeping with this growth trend and for the "bridging" reasons discussed below.

We can plot this out over time on log scale to see that the proposed entity types have been growing exponentially:



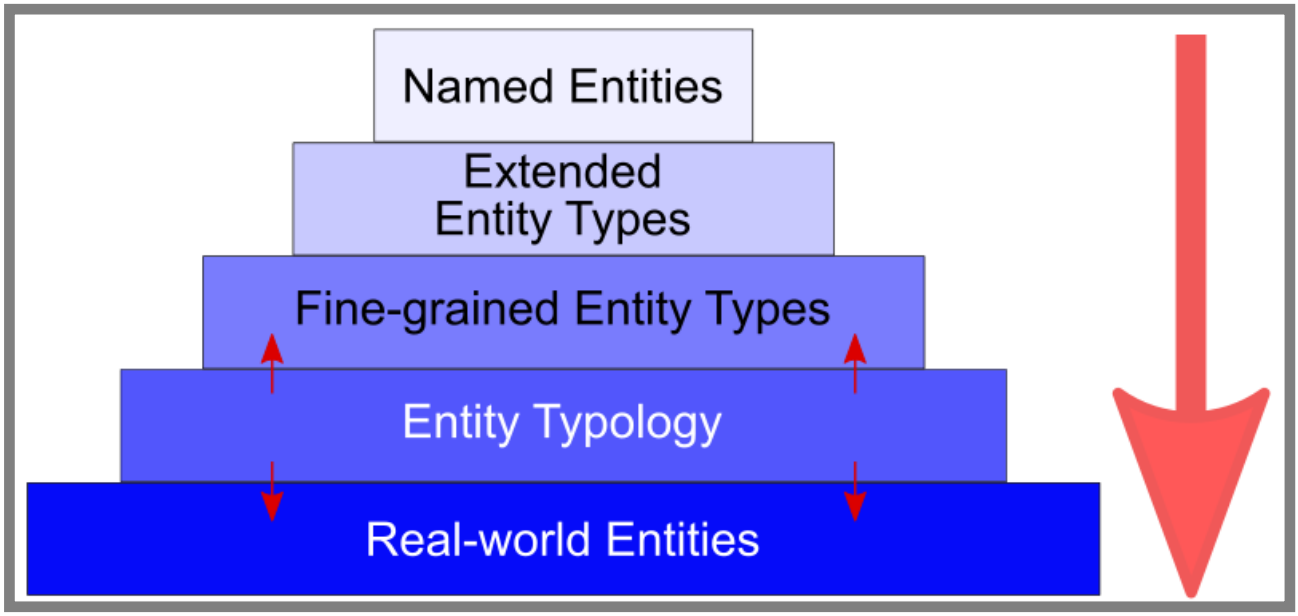
Growth in Recognition of Entity Types

This growth in entity types comes from wanting to describe and organize things with more precision. No longer do we want to talk broadly about people, but we want to talk about astronauts or explorers. We don't just simply want to talk about products, but categories of manufactured goods like cameras or sub-types like SLR cameras or further sub-types like digital SLR cameras or even specific models like the Canon EOS 7D Mark II (skipping over even more intermediate sub-types). With sufficient instances, it is possible to train recognizers for these different entity types.

What is appropriate for a given domain, enterprise or particular task may vary the depth and scope of what entity types should be considered, which we can also refer to as context. For example, the toucan has sometimes been used as a example of how to refer to or name a thing on the Web [25]. When we inspect what might be a definitive description of "toucan" on Wikipedia, we see that the term more broadly represents the family of *Ramphastidae*, which contains five genera and forty different species. The picture we display is but of one of those forty species, that of the keel-billed toucan (*Ramphastos sulfuratus*). Viewing the images of the full list of toucan species shows just how physically divergent these various "toucans" are from one another. Across all species, average sizes vary by more than a factor of three with great variation in bill sizes, coloration and range. Further, if I assert that the picture of the toucan is actually that of my pet keel-billed toucan, *Pretty Bird*, then we can also understand that this representation is for a specific individual bird, and not the physical keel-billed toucan species as a whole.

The point is not a lesson on toucans, but an affirmation that distinctions between what we think we may be describing occurs over multiple levels. Just as there is no self-evident criteria as to what constitutes an "entity type", there is also not a self-evident and fully defining set of criteria as to what the physical "toucan" bird should represent. The meaning of what we call a "toucan" bird is not embodied in its label or even its name, but in the accompanying referential information that place the given referent into a context that can be communicated and understood. A URI *points to* ("refers to") something that causes us to conjure up an understanding of that thing, be it a general description of a toucan, a picture of a toucan, an understanding of a species of toucan, or a specific toucan bird. Our understanding or interpretation results from the context and surrounding information accompanying the reference.

Both in terms of historical usage and trying to provide a framework for how to relate these various understandings of entities and types, we can thus see this kind of relationship:



Evolving Sophistication of Entity Types

What we see is an entities typology that provides the "bridging" interface between specific entity records and the UMBEL reasoning layer. This entities typology is built around UMBEL's existing SuperTypes. The typology is the result of the classification of things according to their shared attributes and essences. The idea is that the world is divided into real, discontinuous and immutable ‘kinds’. Expressed another way, in statistics, typology is a composite measure that involves the classification of observations in terms of their attributes on multiple variables. In the context of a global KB such as Wikipedia, about 25,000 entity types are sufficient to provide a home for the millions of individual articles in the system.

Each SuperType related to entities has its own typology, and is generally expressed as a taxonomy of 3-4 levels, though there are some cases where the depth is much greater (9-10 levels) [26]. There are two flexible aspects to this design. First, because the types are "natural" and nested [27], coarse entity schema can readily find a correspondence. Second, if external records have need for more specificity and depth, that can be accommodated through a mapped bridging hierarchy as well. In other words, the typology can expand and contract like a squeezebox to map a range of specificities.

The internal process to create these typologies also has the beneficial effect of testing placements in the knowledge graph and identifying gaps in the structure as informed by fragments and orphans. The typologies should optimally be fully connected in order to completely fulfill their bridging function.

Extending the Mindset to Attributes and Relations

As with our defined terminology above [28], we can apply this same mindset to the characterizations (**attributes**) of entities and the **relations** between entities and TBox concepts or topics. Numerically, these two categories are much less prevalent than entity types. But, the construction and use of the typologies are roughly the same.

Since we are using RDF and OWL as our standard model and languages, one might ask why we are not

relying on the distinction of object and datatype properties for these splits. Relations, it is true, by definition need to be object properties, since both subject and object need to be identified things. But attributes, in some cases such as rating systems or specific designators, may also refer to controlled vocabularies, which can (and, under best practice, should) be represented as object properties. So, while most attributes are built around datatype properties, not all are. Relations and attributes are a better cleaving, since we can use relations as patterns for fact extractions and the organization of attributes give us a cross-cutting means to understand the characteristics of things independent of entity type. These all become valuable potential features for machine learning, in addition to the standard text structure.

Though, today, UMBEL is considerably more sophisticated in its entities typologies, we already have a start on an attributes typology by virtue of the prior work on the Attributes Ontology [29], which will be revised to conform to this newer typology model. We also have a start on a relations typology based on existing OWL and RDF predicates used in UMBEL, plus many candidates from the Activities SuperType. As with the entities typology, relation types and attribute types may also have hierarchy, enabling reasoning and the squeezebox design. As with the entities typology, the objective is to have a fully connected hierarchy, of perhaps no more than 3-4 levels depth, with no fragments and no orphans.

A Different Role for Annotations

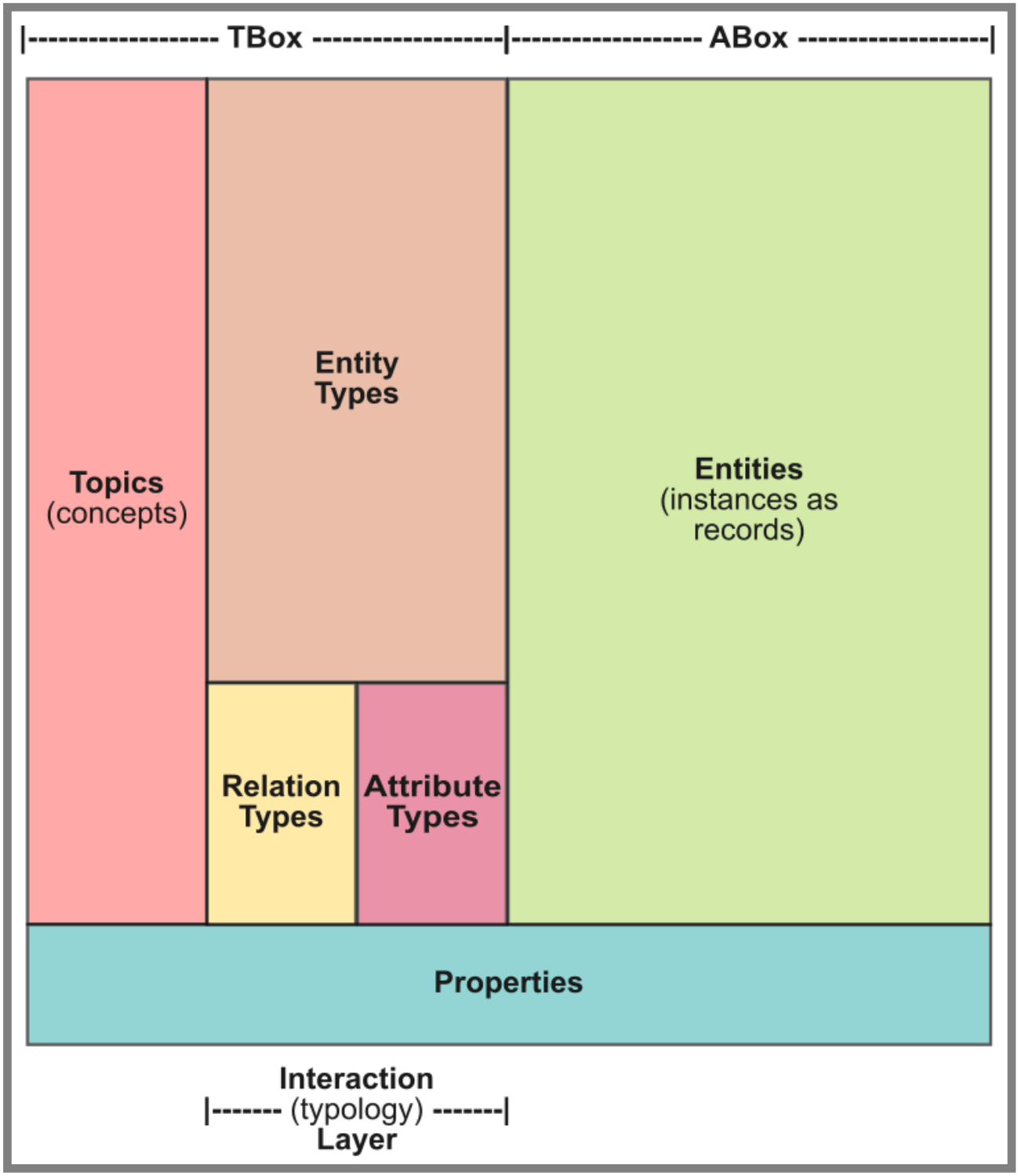
Annotations about how we label things and how we assign metadata to things resides at a different layer than what has been discussed to this point. Annotations can not be reasoned over, but they can and do play pivotal roles. Annotations are an important means for tagging, matching and slicing-and-dicing the information space. Metadata can perform those roles, but also may be used to analyze provenance and reasoning, if the annotations are mapped to object or datatype properties.

Labels are the means to broaden the correspondence of real-world reference to match the true referents or objects in the knowledge base. This enables the referents to remain abstract; that is, not tied to any given label or string. In best practice we recommend annotations reflect all of the various ways a given object may be identified (synonyms, acronyms, slang, jargon, all by language type). These considerations improve the means for tagging, matching, and slicing-and-dicing, even if the annotations are not able to be reasoned over.

As a mental note for the simple design that follows, imagine a transparent overlay, not shown, upon which best-practice annotations reside.

A Simple Design Brings it All Together

The insights provided here have taken much time to discover; they have arisen from a practical drive to make knowledge bases computable and useful to artificial intelligence applications. Here is how we now see the basics of a knowledge base, properly configured to be computable, and open to integration with external records:



Boiling KBs Down to Basics

At the broadest perspective, we can organize our knowledge-base platform into a "brain" or organizer/reasoner, and the instances or specific things or entities within our domain of interest. We can decompose a KB to become computable by providing various type groupings for our desired instance mappings, and an upper reasoning layer. An interface layer of "types", organized into three broad

groupings, provides the interface, or "bridging" layer, between the TBox and ABox. We thus have an architectural design segregating:

- Topics and upper level -- the general organization and "brains" of the domain
- Entity types -- categorizations of the actual things in the space
- Relation types -- the ways that different things are related to, or act upon, one another
- Attribute types -- a structured organization of the ways that individual entities can be described
- Instances -- the individual entities of the domain, and
- Properties -- the source grist for annotations, relation types and attribute types.

Becoming familiar with this terminology helps to show how the conventional understanding of these terms and structure have led to overlooking key insights and focusing (sometimes) on the wrong matters. That is in part why so much of the simple logic of this design has escaped the attention of most practitioners. For us, personally, at Structured Dynamics, it had eluded us for years, and we were actively seeking it.

Irrespective of terminology, the recognition of the role of types and their bridging function to actual instance data (records) is central to the computability of the structure. It also enables integration with any form of data record or record stores. The ability to understand relation types leads to improved relation extraction, a key means to mine entities and connections from general content and to extend assertions in the knowledge base. Entity types provide a flexible means for any entity to connect with the computable structure. And, the attribute types provide an orthogonal and inferential means to slice the information space by how objects get characterized.

Because of this architecture, the reference sources guiding its construction, its typologies, its ability to generate features and training sets, and its computability, we believe this overall design is suited to provide an array of AI and enterprise services:

Machine Intelligence Apps and Services

- Entity recognizers
- Relation extractors
- Event extractors
- Phrase identification
- Classifiers
- Q & A systems
- Cognitive computing
- Semantic publishing
- Knowledge base mappings
- Sub-graph extraction
- Ontology development
- Ontology mappers
- Entity dictionaries
- Entity linkers
- Data conversion and mapping
- Master data management
- KB improvements
- Attribute “slot filling”
- Disambiguators
- Duplicates removal
- Inference and reasoning
- Sentiment analysis
- Semantic relatedness
- Recommendation systems
- Bespoke analysis
- Bespoke platforms

By cutting through the clutter -- conceptually and terminologically -- it has been possible to derive a practical and repeatable design to make KBs computable. Being able to generate features and positive and negative training sets, almost at will, is proving to be an effective approach to machine learning at mass-produced prices.

[1] See M. K. Bergman, 2014. "Knowledge-based Artificial Intelligence," from *AI3:::Adaptive Information* blog, November 17, 2014. For additional writings in the series, see http://www.mkbergman.com/category/kbai/.

[2] See Fabian M. Suchanek and Gerhard Weikum, 2014. "Knowledge Bases in the Age of Big Data Analytics," *Proceedings of the VLDB Endowment* 7, no. 13 (2014) and M.K. Bergman, "SWEETpedia," listing of Wikipedia research articles, on *AI3:::Adaptive Information* blog, January 25, 2010; the listing as of its last update included 246 articles. Also, see Wikipedia's own "Wikipedia in Academic Studies."

[3] A possible exception to this observation is the biomedical community through its Open Biological and Biomedical Ontologies (OBO) initiative.

[4] See M.K. Bergman, 2007. "Announcing UMBEL: A Lightweight Subject Structure for the Web," *AI3:::Adaptive Information* blog, July 12, 2007. Also see http://umbel.org.

[5] See M.K. Bergman, 2008. "Basing UMBEL's Backbone on OpenCyc," *AI3:::Adaptive Information* blog, April 2, 2008.

[6] See M.K. Bergman, 2015. "Shaping Wikipedia into a Computable Knowledge Base," *AI3:::Adaptive Information* blog, March 31, 2015.

[7] M.K. Bergman, 2009. 'SuperTypes' and Logical Segmentation of Instances, *AI3:::Adaptive Information* blog, September 2, 2009.

[8] This possible use of an item as both a class and an instance through "punning" is a desirable feature of OWL 2, which is the language basis for UMBEL. You can learn more on this subject in M.K. Bergman, 2010. "Metamodeling in Domain Ontologies," *AI3:::Adaptive Information* blog, September 20, 2010.

[9] For a listing of these, see the Google query https://www.google.com/search?q=tbox+abox+site%3Amkbergman.com. One of the 40 articles with the most relevant commentary to this article is M.K. Bergman, 2014. "Big Structure and Data Interoperability," *AI3:::Adaptive Information* blog, August 14, 2014.

[10] M.K. Bergman, 2009. " Making Linked Data Reasonable using Description Logics, Part 1," *AI3:::Adaptive Information* blog, February 11, 2009.

[11] The current development version of UMBEL is v 1.30. It is due for release before the end of 2015.

[12] See the Sekine Extended Entity Types; the listing also includes attributes info at bottom of source page.

[13] For SUMO, see http://virtual.cvut.cz/kifb/en/toc/229.html.

[14] N. Chinchor, 1997. "Overview of MUC-7," *MUC-7 Proceedings*, 1997.

[15] Ada Brunstein, 2002. "Annotation Guidelines for Answer Types". *LDC Catalog*, Linguistic Data Consortium. Aug 3, 2002.

[16] Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann, 2009. "DBpedia-A Crystallization Point for the Web of Data." *Web Semantics: science, services and agents on the world wide web* 7, no. 3 (2009): 154-165; 170 classes in this paper. That has grown to more than

700; see http://mappings.dbpedia.org/server/ontology/classes/ and http://wiki.dbpedia.org/services-resources/datasets/dataset-2015-04/dataset-2015-04-statistics.

[17] The listing is under some dynamic growth. This is the official count as of September 8, 2015, from http://schema.org/docs/full.html. Current updates are available from Github.

[18] Joanna Biega, Erdal Kuzey, and Fabian M. Suchanek, 2013. "Inside YAGO2: A Transparent Information Extraction Architecture," in *Proceedings of the 22nd international conference on World Wide Web*, pp. 325-328. International World Wide Web Conferences Steering Committee, 2013. Also see Mohamed Amir Yosef, Sandro Bauer, Johannes Hoffart, Marc Spaniol, Gerhard Weikum, 2012. "HYENA: Hierarchical Type Classification for Entity Names," in *Proceedings of the 24th International Conference on Computational Linguistics, Coling 2012*, Mumbai, India, 2012.

[19] Changki Lee, Yi-Gyu Hwang, Hyo-Jung Oh, Soojong Lim, Jeong Heo, Chung-Hee Lee, Hyeon-Jin Kim, Ji-Hyun Wang, and Myung-Gil Jang, 2006. "Fine-grained Named Entity Recognition using Conditional Random Fields for Question Answering," in *Information Retrieval Technology*, pp. 581-587. Springer Berlin Heidelberg, 2006.

[20] Xiao Ling and Daniel S. Weld, 2012. "Fine-Grained Entity Recognition," in AAAI. 2012.

[21] Dan Gillick, Nevena Lazic, Kuzman Ganchev, Jesse Kirchner, and David Huynh, 2104. "Context-Dependent Fine-Grained Entity Type Tagging." *arXiv preprint arXiv*:1412.1820 (2014).

[22] See http://new.opencalais.com/wp-content/uploads/2015/08/ThomsonReutersOpenCalaisAPIUserGuide_17_8a.pdf, p 89.

[23] See https://en.wikipedia.org/wiki/GeoNames.

[24] David Nadeau, 2007. "Semi-supervised Named Entity Recognition: Learning to Recognize 100 Entity Types with Little Supervision." PhD Thesis, *School of Information Technology and Engineering, University of Ottawa*, 2007.

[25] M.K. Bergman, 2012. " Give Me a Sign: What Do Things Mean on the Semantic Web?," *AI3:::Adaptive Information* blog, January 24, 2012.

[26] A good example of description and use of typologies is in the archaelogy description on Wikipedia.

[27] M.K. Bergman, 2015. "'Natural' Classes in the Knowledge Web", *AI3:::Adaptive Information* blog, July 13, 2015.

[28] Also see my Glossary for definitions of specific terminology used in this article.

[29] M.K. Bergman, 2015. "Conceptual and Practical Distinctions in the Attributes Ontology", *AI3:::Adaptive Information* blog, March 3, 2015.
