Ontology is overrated: <http://shirky.com/writings/herecomeseverybody/ontology_overrated.html>

http://www.jfsowa.com/  
“capitalism incarnates hyperstitional dynamics at an unprecedented and unsurpassable level of intensity, turning mundane economic ‘speculation’ into an effective world-historical force”

"Hyperstition is a positive feedback circuit including culture as a component. It can be defined as the experimental (techno-)science of self-fulfilling prophecies. Superstitions are merely false beliefs, but hyperstitions – by their very existence as ideas – function causally to bring about their own reality. Capitalist economics is extremely sensitive to hyperstition, where confidence acts as an effective tonic, and inversely. The (fictional) idea of Cyberspace contributed to the influx of investment that rapidly converted it into a technosocial reality"

[http://merliquify.com/blog/articles/hyperstition-an-introduc...](http://merliquify.com/blog/articles/hyperstition-an-introduction/)

<https://www.semanticscholar.org/topic/ClearTalk/8230738> - **ClearTalk** is a controlled natural language—a kind of a formal language for expressing information that is designed to be both human-readable   
<https://en.wikipedia.org/wiki/Attempto_Controlled_English>  
  
  
<http://www.jfsowa.com/ontology/ontometa.htm>.

1. *Syntax*. "The first is called by Duns Scotus *grammatica speculativa*. We may term it *pure grammar*." Syntax is the study that relates signs to one another.
2. *Semantics*. "The second is logic proper," which "is the formal science of the conditions of the truth of representations." Semantics is the study that relates signs to things in the world and patterns of signs to corresponding patterns that occur among the things the signs refer to.
3. *Pragmatics*. "The third is... pure rhetoric. Its task is to ascertain the laws by which in every scientific intelligence one sign gives birth to another, and especially one thought brings forth another." Pragmatics is the study that relates signs to the agents who use them to refer to things in the world and to communicate their intentions about those things to other agents who may have similar or different intentions concerning the same or different things.

Unfortunately, most word processors deal only with a small subset of syntax. They have produced what St. Laurent (1999) calls *the WYSIWYG disaster*: "Plain text, dull though it may be, is much easier to manage than the output of the average word processor or desktop publishing program." In practice, the slogan "What you see is what you get" actually means WYSIAYG: "What you see is all you get." The text is so overburdened with formatting tags that there is no room for semantics or pragmatics. The so-called Rich Text Format (RTF) is semantically the most impoverished representation for text ever devised. Formatting is an aspect of signs that makes them look pretty, but it fails to address the more fundamental question of what they mean.

XML use tags in the text to represent semantics and put the formatting in more easily manageable style sheets. That separation is important, but the semantic tags themselves must have a clearly defined semantics. Most XML manuals, however, provide no guidelines for representing semantics. Following is an excerpt from one of the proposed standards for representing *resources* in XML:

“””A resource can be anything that has identity. Familiar examples include an electronic document, an image, a service (e.g., "today's weather report for Los Angeles"), and a collection of other resources. Not all resources are network "retrievable"; e.g., human beings, corporations, and bound books in a library can also be considered resources. (Berners-Lee, et al. 1998)”””

In that report, an electronic document is considered familiar, but human beings are unfamiliar "resources" mentioned only as an afterthought. Yet without the people, the document and its contents have no meaning.  
  
Many of the ontologies for web objects ignore physical objects, processes, people, and their intentions. A typical example is SHOE (Simple HTML Ontology Extensions), which has only four basic categories: String, Number, Date, and Truth (Heflin et al. 1999). Those four categories, which are needed to describe the syntax of web data, cannot by themselves describe the semantics. Strings contain characters that represent statements that describe the world; numbers count and measure things; dates are time units tied to the rotation of the earth; and truth is a metalanguage term about the correspondence between a statement and the world. Those categories can only be defined in terms of the world, the people in the world, and the languages people use to talk about the world. Without such definitions, the categories are meaningless tags that confer no meaning upon the data they are attached to.  
In discussing the Resource Description Framework (RDF), which is based on the XML facilities, Bray (1998) presented a broader view of the kinds of categories that web-based metadata should represent:

It seems unlikely that one PropertyType standing by itself is apt to be very useful. It is expected that these will come in packages; for example, a set of basic bibliographic PropertyTypes like Author, Title, Date, and so on. Then a more elaborate set from OCLC, and a competing one from the Library of Congress. These packages are called Vocabularies; it's easy to imagine PropertyType vocabularies describing books, videos, pizza joints, fine wines, mutual funds, and many other species of Web wildlife.

This is a good statement of one issue, but it raises other issues: How are the packages related to one another? How is the Date property of the OCLC package related to the Vintage property of a wine package? Can packages inherit type definitions from other packages? If two packages are competing, is there any way to define conversion rules for translating or redefining the types of one in terms of another? A human reader may know that a wine vintage can be compared to an OCLC date, but without a formal definition, the computer cannot.

Ironically, the computer networks that make it easier to transmit data have made it more difficult to share data. In continuing his discussion, Bray raised further issues:

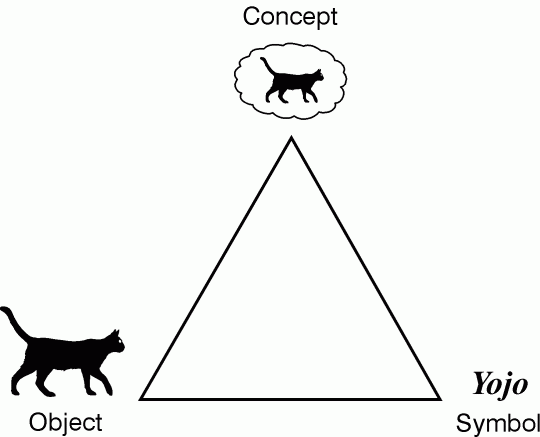
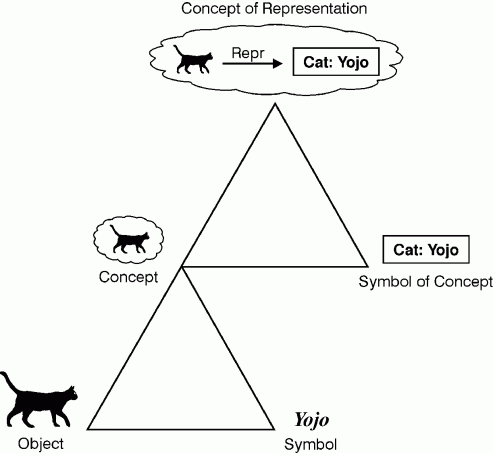
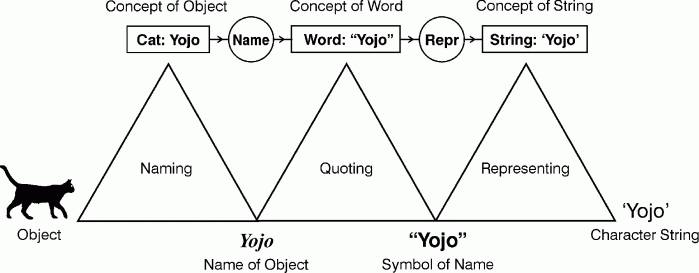
Nobody thinks that everyone will use the same vocabulary (nor should they), but with RDF we can have a marketplace in vocabularies. Anyone can invent them, advertise them, and sell them. The good (or best-marketed) ones will survive and prosper. Probably, most niches of information will come to be dominated by a small number of vocabularies, the way that library catalogues are today.

There are already thousands, if not millions of competing vocabularies. The tables and fields of every database and the lists of items in every product catalog for every business in the world constitute incompatible vocabularies. When product catalogs were distributed on paper, any engineer or contractor could read the catalogs from different vendors and compare the specifications. But minor variations in the terminology of computerized catalogs can make it impossible for a computer system to compare components from different vendors.

By standardizing the notations, XML and RDF take an important first step, but that step is insufficient for data sharing without some way of comparing, relating, and translating the vocabularies. Phipps (2000) warned that standardizing the vocabularies may create even more difficulties "by hiding complexities behind superficial agreements":To connect from the heart of my e-business to the heart of yours would be impossibly expensive in shared systems without XML, but even with it the system analysis needed to create the translation is a significant task. We should not assume that XML is a panacea or that the standardization of vocabularies will automatically bring interoperability. XML provides us with a medium to express our understanding of the meaning of data, but we still have to discern realities and differences of meanings when we exchange data.

More important than standardizing vocabularies is the development of methods for defining and translating vocabularies. To have a sound semantics and pragmatics, those methods must relate the terms in the vocabularies to the things they refer to and to the people who use them to communicate information about those things.

The purpose of this paper is to analyze the differences of meaning, to explore their implications for web-based metadata, and to show how the methods of logic and ontology can be used to define, relate, and translate signs from one vocabulary to another. Among the methods discussed in this paper are Peirce's systems of logic, ontology, and semiotics, which are presented in more detail in the book [*Knowledge Representation*](http://www.jfsowa.com/krbook/) by Sowa (2000).  
  
**2. Signs of Signs**

****  
 **  
  
The combination of concept and relation nodes is a** [***conceptual graph***](http://www.jfsowa.com/cg/index.htm) **(CG).**

**Different kinds of applications require different levels of detail in the metadata. For information retrieval (IR), a simple string search can often find a web page with the desired information. To find information about Yojo the cat, it could search for the strings "Yojo" and "cat"; to find information about Queequeg's ebony idol in the novel *Moby Dick*, it could search for the strings "Yojo" and "Queequeg". IR systems depend on a human reader to decide which strings to search for and to interpret the results that are retrieved. Systems that go beyond simple search must be able to distinguish the physical object Yojo from an icon that resembles the object, the name of the object, and the character string that represents the name. Following is an interchange between a human user who asked a question and a computer system that did not make those distinctions:**

**Q: What is the largest state in the US?**

**A: Wyoming.**

**To answer questions about sizes, the computer would use the greater-than operator to compare numbers. When it applied that operator to the character strings, it found the last state in alphabetical order, which does not happen to be the largest state in either area or population. A loosely defined system of metadata may be adequate for finding information, but inadequate for any further processing. As Phipps observed, superficial agreements about vocabulary may hide complexities that make interoperability impossible.**

| **Primitive** | **Informal Meaning** | **English Example** |
| --- | --- | --- |
| **Existence** | **Something exists.** | ***There is a dog.*** |
| **Coreference** | **Something is the same as something.** | ***The dog is my pet.*** |
| **Relation** | **Something is related to something.** | ***The dog has fleas.*** |
| **Conjunction** | **A and B.** | ***The dog is running, and the dog is barking.*** |
| **Negation** | **Not A.** | ***The dog is not sleeping.*** |

**The five primitives in** [**Table 1**](http://www.jfsowa.com/ontology/ontometa.htm#tab1) **are available in every natural language and in every version of first-order logic. They are called semantic primitives because they go beyond syntactic relations between signs to semantic relations between signs and the world. Any notation that is capable of expressing these five primitives in all possible combinations must include all of FOL as a subset.  
  
As an example, the WHERE clause of the SQL query language can express each of these primitives and combine them in all possible ways; therefore, first-order logic is a subset of SQL. Different languages may use different notations for representing the five primitives:**

* ***Existence*. In most natural languages, existence is implied by mentioning something. For emphasis, languages also provide an explicit*existential quantifier* such as the word *some*. In the algebraic notation for logic, existence may be expressed by an explicit symbol, such as ∃. In SQL, existence is stated implicitly by mentioning something or explicitly by using the keyword EXISTS.**
* ***Coreference*. To say that two different signs refer to the same thing, natural languages use a variety of methods, both explicit and implicit: pronouns, determiners, inflections, and forms of the verb *be*. Most linear notations for logic use variables and the equal sign, and graphic notations use connecting lines or ligatures. Like other linear notations, SQL uses variables and the equal sign.**
* ***Relation*. Content words in natural languages express some information about at least one entity, known as the *referent* of the word, but they may also relate or imply other entities as well. The verb *give*, for example, refers to an act of giving, but it also implies a giver, a gift, and a recipient. In SQL, relations are called *tables*.**
* ***Conjunction*. In both natural and artificial languages, conjunction may be expressed implicitly by making one statement after another or explicitly by a word like *and* or a symbol like ∧. SQL uses the keyword AND.**
* ***Negation*. All natural languages and most versions of logic provide words, inflections, or symbols to express negation. The biggest variations from one language to another are in the methods for distinguishing the context or scope of what is negated from what is not negated. SQL uses the keyword NOT with parentheses to show scope.**

**Other logical operators can be defined in terms of these five primitives.** [**Table 2**](http://www.jfsowa.com/ontology/ontometa.htm#tab2) **shows three of the most common: the universal quantifier, implication, and disjunction. These operators do not qualify as semantic primitives because they are not as directly observable as the five in** [**Table 1**](http://www.jfsowa.com/ontology/ontometa.htm#tab1)**. Seeing Yojo, for example, is evidence that some cat exists, but there is no way to perceive every cat. Seeing two things together is evidence of a conjunction, and not seeing something is evidence of a negation. But there is no direct way of perceiving an implication or a disjunction: *post hoc*does not imply *propter hoc*, and seeing one alternative of a disjunction does not indicate what other alternatives are possible. Although the three operators of** [**Table 2**](http://www.jfsowa.com/ontology/ontometa.htm#tab2) **can be defined in terms of the five primitives, any assertion they make about the world can only be verified indirectly and usually with less certainty than the basic primitives.**

| **Operator** | **English Example** | **Translation to Primitives** |
| --- | --- | --- |
| **Universal** | ***Every dog is barking.*** | **not(*there is a dog* and not(*it is barking*))** |
| **Implication** | ***If there is a dog, then it is barking.*** | **not(*there is a dog* and not(*it is barking*))** |
| **Disjunction** | ***A dog is barking, or a cat is eating.*** | **not(not(*a dog is barking*) and not(*a cat is eating*))** |

**In summary, the algebraic notation for logic, which is popular with mathematicians, is only one of an open-ended number of semantically equivalent notations. The five semantic primitives of** [**Table 1**](http://www.jfsowa.com/ontology/ontometa.htm#tab1) **and the mechanisms for defining the other operators of first-order logic can be adapted to a wide variety of notations, including natural languages and the web-oriented notations of XML and RDF  
Logic can be and has been represented in a wide variety of graphic and linear notations of varying degrees of readability and suitability for different kinds of applications. EGs and CGs are graphic examples, and the Knowledge Interchange Format (KIF) is an equivalent linear form. Other linear versions can be written with the syntactic conventions of SQL, RDF, or even natural languages.**

1. **For better readability, it is possible to represent the logical operators in *controlled natural languages*, which use a subset of the syntax and vocabulary of natural languages. Although the task of translating unrestricted natural languages to any formal notation is still a research problem, it is much easier to translate conceptual graphs and other formal notations to a stylized version of natural language, such as the English translations of the CGs in this article.**
2. **Besides notation, logic has rules of definition and inference, which allow one representation to be translated to or from other synonymous representations. Figures 6 through 10 can be translated automatically to or from one another or the equivalent formulas in predicate calculus — provided that an appropriate ontology has been defined. With its formally defined semantics, logic provides the means for generating semantically equivalent translations to and from other languages with radically different syntax.**

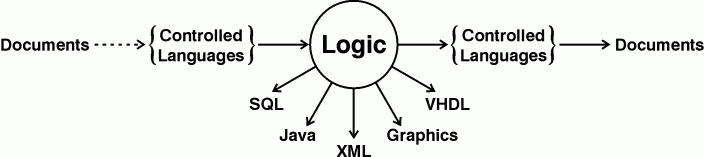
## **4. Combining Logic with Ontology**

Pure logic is ontologically neutral. It makes no presuppositions about what exists or may exist in any domain or any language for talking about the domain. To represent knowledge about a specific domain, it must be supplemented with an ontology that defines the categories of things in that domain and the terms that people use to talk about them. The ontology defines the words of a natural language, the predicates of predicate calculus, the concept and relation types of conceptual graphs, the classes of an object-oriented language, or the tables and fields of a relational database. To illustrate the issues of defining an ontology, consider the conceptual graph in Figure 12, which represents the sentence *Yojo is chasing a mouse*.

CG for 'Yojo is chasing a mouse.'

**Figure 12. CG for *Yojo is chasing a mouse.***

**The contexts of conceptual graphs are based on Peirce's logic of existential graphs and his theory of indexicals. Yet the CG contexts happen to be isomorphic to the similarly nested *discourse representation structures* (DRS), which Hans Kamp (1981a,b) developed for representing and resolving indexicals in natural languages. When Kamp published his first version of DRS, he was not aware of Peirce's graphs. When Sowa (1984) published his book on conceptual graphs, he was not aware of Kamp's work. Yet the independently developed theories converged on semantically equivalent representations; therefore, Sowa and Way (1986) were able to apply Kamp's techniques to conceptual graphs. Such convergence is common in science; Peirce and Frege, for example, started from very different assumptions and converged on equivalent semantics for FOL, which 120 years later is still the most widely used version of logic. Independently developed, but convergent theories that stand the test of time are a more reliable basis for standards than the consensus of a committee.  
5. Extracting Logic from Language  
For database queries and constraints, natural language statements with the full expressive power of FOL can be translated to SQL. Although many NL query systems have been developed, none of them have yet become commercially successful. The major stumbling block is the amount of effort required to define the vocabulary terms and map them to appropriate fields of the database. But if KE tools are used to design the database, the vocabulary needed for the query system can be generated as a by-product of the design process. As an example, the RÉCIT system (Rassinoux 1994; Rassinoux et al. 1998) uses KE tools to extract knowledge from medical documents written in English, French, or German and translates the results to a language-independent representation in conceptual graphs. The knowledge extraction process defines the appropriate vocabulary, specifies the database design, and adds new information to the database. The vocabulary generated by the KE process is sufficient for end users to ask questions and get answers in any of the three languages.  
  
Design and specification languages have multiple metalevels. As an example, the Unified Modeling Language has four levels: the metametalanguage defines the syntax and semantics of the UML notations; the metalanguage defines the general-purpose UML types; a systems analyst defines application types as instances of the UML types; finally, the working data of an application program consists of instances of the application types. To provide a unified view of all these levels, Olivier Gerbé and his colleagues at the DMR Consulting Group implemented design tools that use conceptual graphs as the representation language at every level (Gerbé et al. 1995, 1996, 1997, 1998, 2000). For his PhD dissertation, Gerbé developed an ontology for using CGs as the metametalanguage for defining CGs themselves. He also applied it to other notations, including UML and the Common KADS system for designing expert systems. Using that theory, Gerbé and his colleagues developed the Method Repository System for defining, editing, and displaying the methods used by the DMR consultants. Internally, the knowledge base is stored in conceptual graphs, but externally, the graphs can be translated to web pages in either English or French. About 200 business processes have been modeled in a total of 80,000 CGs. Since DMR is a Canadian company, the language-independent nature of CGs is important because it allows the specifications to be stored in the neutral CG form. Then any manager, systems analyst, or programmer can read them in his or her native language.  
  
Translating an informal diagram to a formal notation of any kind is as difficult as translating unrestricted NL to executable programs. But it is much easier to translate a formal representation in any version of logic to controlled natural languages, to various kinds of graphics, and to executable specifications. Walling Cyre and his students have developed KE tools for mapping both the text and the diagrams from patent applications and similar documents to conceptual graphs (Cyre et al. 1994, 1997, 1999). Then they implemented a scripting language for translating the CGs to circuit diagrams, block diagrams, and other graphic depictions. Their tools can also translate CGs to VHDL, a hardware design language used to specify *very high speed integrated circuits* (VHSIC).**No single system discussed in this paper incorporates all the features desired in a KE system, but the critical research has been done, and the remaining work requires more development effort than pure research. Figure 18 shows the flow of information from documents to logic and then to documents or to various computational representations. The dotted arrow from documents to controlled languages requires human assistance. The solid arrows represent fully automated translations that have been implemented in one or more systems.



**Figure 18. Flow of information from documents to computer representations**

For the KE tools, the unifying representation language is logic, which may be implemented in different subsets and notations for different tools. All the subsets, however, use the same vocabulary of natural-language terms, which map to the same ontology of concepts and relations. From the user's point of view, a KE system communicates in a subset of natural language, and the differences between tools appear to be task-related differences rather than differences in language.