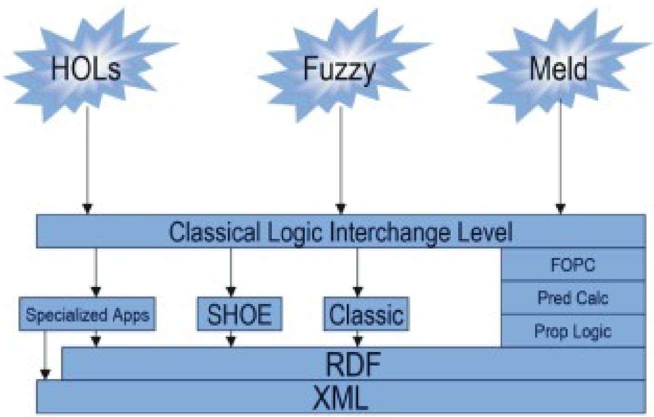
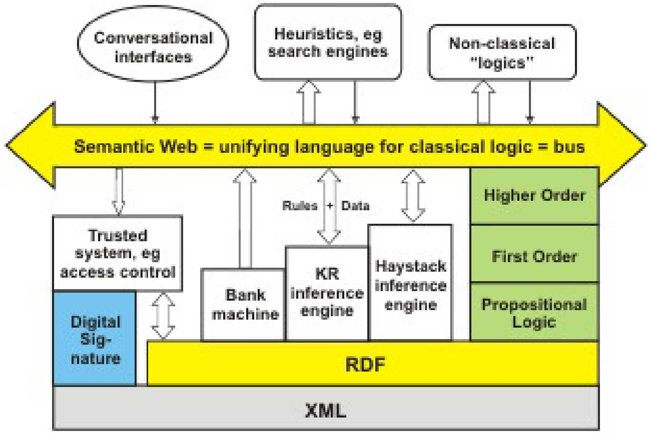
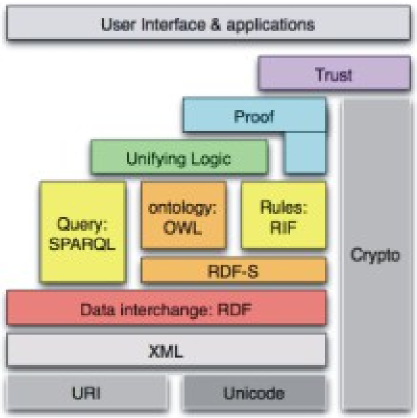
Language, Ontology, And the Semantic Web







John F. Sowa

3 June 2020

Language, Ontology, and the SW

Abstract: ***In 2000, Tim Berners-Lee proposed a vision for the Semantic Web that was more ambitious than the results delivered in 2005. Research in the past 15 years produced advanced technology in artificial intelligence, language processing, and reasoning methods, both formal and informal. But many systems are proprietary, incompatible with one another, and too complex for widespread adoption. Among the most important requirements, trusted systems were never adequately implemented. This talk surveys promising developments and suggests ways of adapting them to the Semantic Web.***

Contents:

1. Semantic Web From 2000 to 2005

In 2000, Tim Berners-Lee wrote an ambitious proposal.*

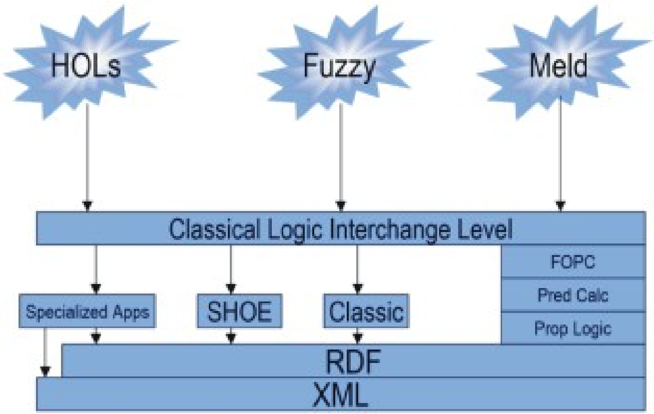
- The Semantic Web “as an interchange bus for on-line data.”
- RDF as a simple language for exchanging “raw data” among “heterogeneous systems.”
- SWeLL (Semantic Web Logic Language) “extends RDF by including negation and explicit quantification.”
- SWeLL should represent first-order and higher-order logic and pair “simple, predictable, reliable systems with complex, unpredictable, heuristic systems.”
- But the tools delivered in 2005 were more limited.

Goal: Implement Tim’s vision with a new generation of tools.

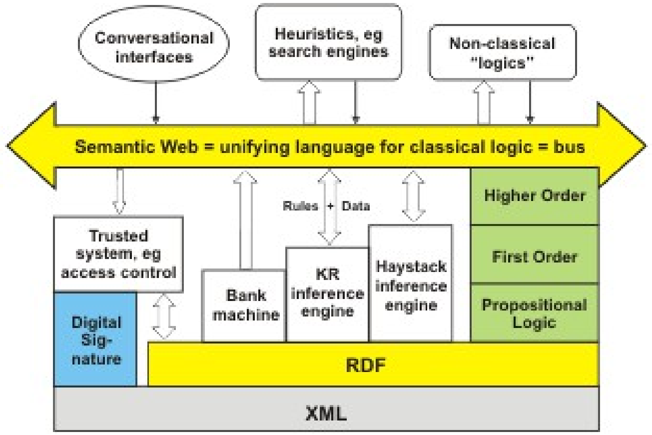
- More advanced methods for language, learning, and reasoning.
- Better methods for building trust and ensuring security.

* For the original documents, see http://jfsowa.com/ikl/

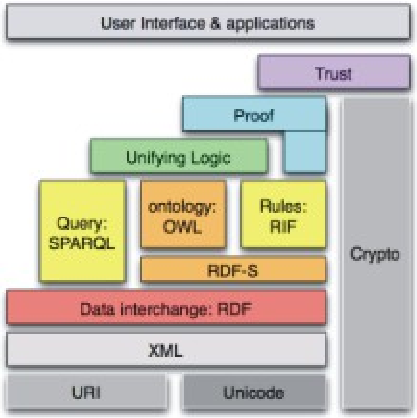
Semantic Web "Layer Cakes"



DAML Requirements (2000)



Winning Proposal (2000)



Final Report (2005)

The proposal was more ambitious than what was delivered.

- **New AI technology has been developed in the past 15 years.**
- **Advanced applications have gone far beyond the tools of 2005.**
- **The new technology should be more widely available.**
- **It should be easy to learn, easy to use, and upward compatible.**

In the diagram, the large yellow arrow is the SWeLL bus:

- **Semantic Web = unifying language for classical logic = bus.**

Two Examples of Advanced AI

Google's knowledge graphs (KGs) are represented in RDF.

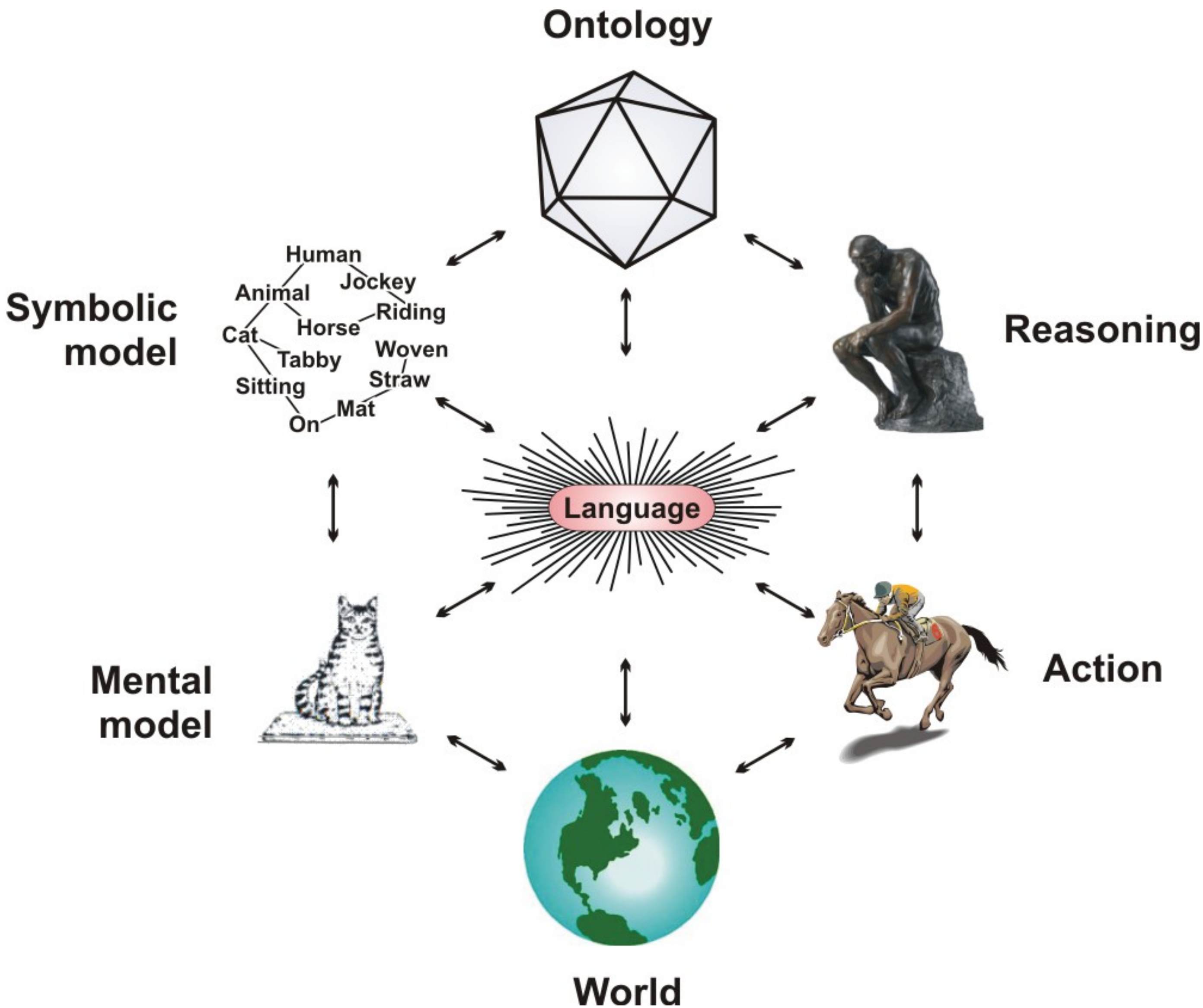
- DBpedia and other freely available resources provide the data.
- Google added AI methods for learning and reasoning with KGs and deriving new KGs from documents.

For the Jeopardy challenge, IBM Watson also used DBpedia.*

Watson added a wide range of AI technology: English parsers, question classification, question decomposition, automatic source acquisition and evaluation, entity and relation detection, logical form generation, statistics, machine learning, knowledge representation, and several methods of reasoning.

Goals for the future: Automated and semi-automated tools to make such systems easier to design, cheaper to build, and more reliable.

* See https://www.aaai.org/Magazine/Watson/watson.php



RDF can represent symbolic models that are directly related to ontology, language, and mental models. Tools for logic and language can relate them to reasoning, action, and the world.

Implementing the Hexagon

The corners of the hexagon represent aspects of knowledge.

1. The world is everything we encounter in space and time.
2. Mental models represent everything we experience or imagine.
3. Symbolic models consist of words related by words to other words.
4. Ontology is a catalog of words and the kinds of things they refer to.
5. Reasoning includes all our ways of thinking about anything.
6. Action is what our thinking leads us to do in and on the world.

Natural languages represent conscious knowledge.

- They can represent and relate all six corners of the hexagon.
- Every artificial language, notation, or diagram is a simplified or stylized version of something that could be said in a natural language.
- But the nervous system contains an enormous amount of unconscious knowledge that supports the basic operations of the human body.

Challenge: Implement AI tools to support all the above.

2. Interoperability

DOL is a standard for integration and interoperation among distributed ontologies, models, and specifications (OMS). *

- **UML and the Semantic Web logics are supported by DOL.**
- **DOL tools can relate anything specified by those logics or an open-ended variety of others.**
- **That includes the notations for representing legacy software and the latest technologies of the 21st century.**

DOL is formally defined by logic and mathematics.

- **Logic is essential for guaranteeing precision.**
- **DOL can integrate heterogeneous OMS by relating the logics that specify them. Common Logic (CL) is one of the most general.**
- **But people may continue to use any notations they prefer.**

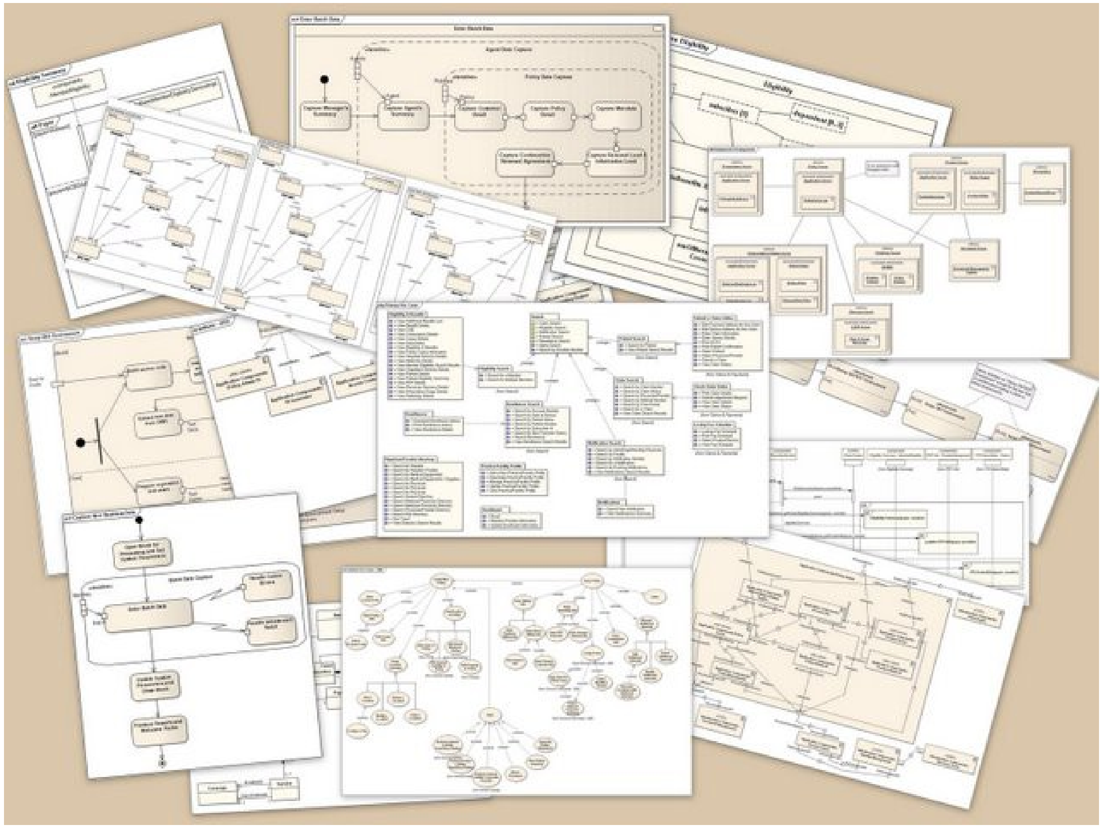
*** OMG Standard for DOL: Distributed Ontology, Modeling, and Specification Language: https://www.omg.org/spec/DOL/1.0**

Unified Modeling Language (UML)

A family of diagrams for representing database and computer system designs.

Originally specified as informal notations without a precise definition in logic.

The Object Management Group (OMG) standardized formal UML by definitions stated in Common Logic.*



By mapping UML diagrams and SW logics to CL, DOL can facilitate data sharing among applications in any field.

*** See https://www.omg.org/spec/FUML/1.4**

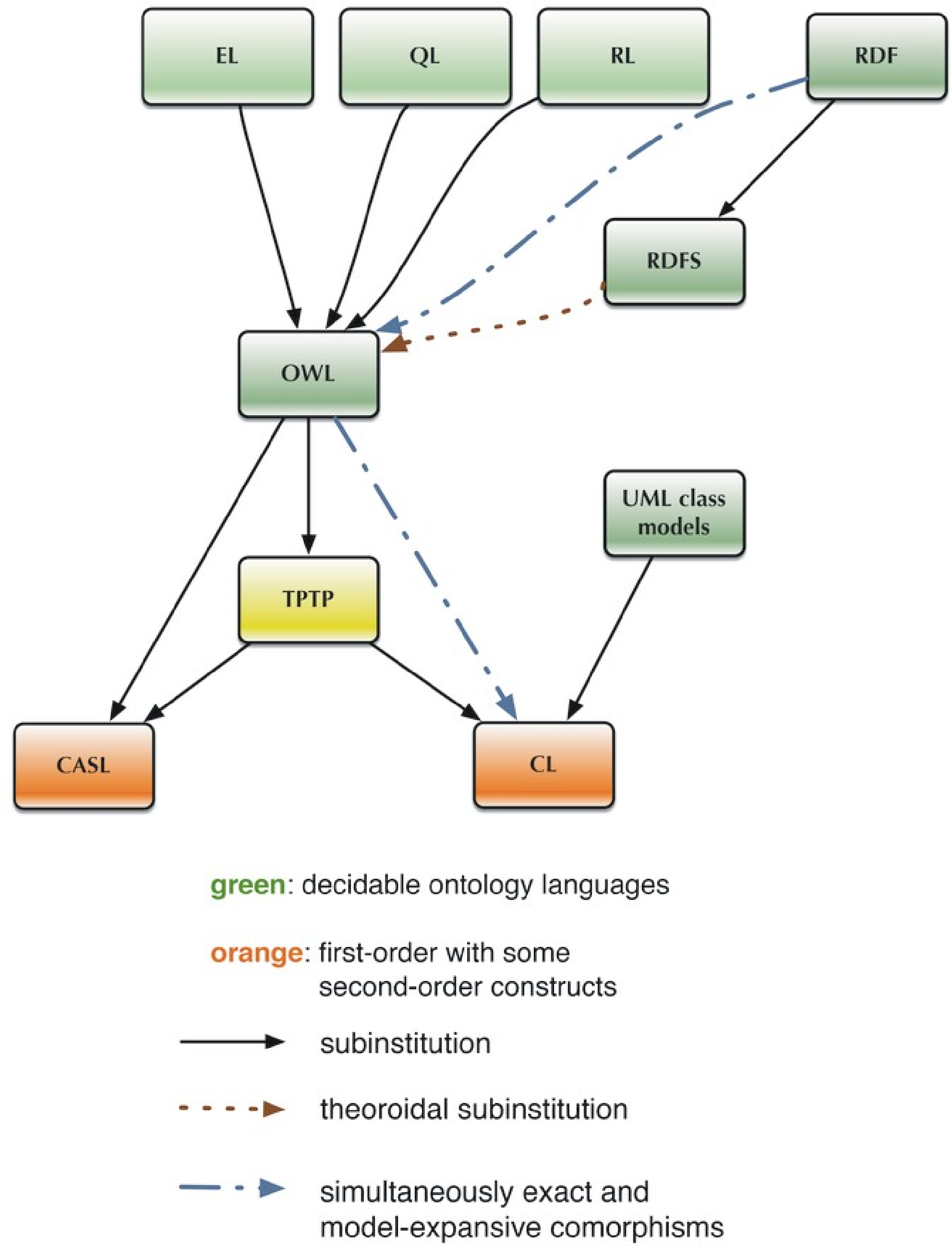
Mapping UML and the Semantic Web to CL

The diagram shows the most widely used logics supported by DOL.

Arrows show the mappings from less expressive logics to more expressive logics. Common Logic is at the lower right.

TPTP notation (for Thousands of Problems for Theorem Provers) is a version of many-sorted logic, of which classical first-order logic is a single-sorted subset.

HeTS (the Heterogeneous Tool Set) uses CASL as the interchange logic for this diagram.   But other tools may use other logics.



Supporting Interoperability

Usage scenarios for DOL. *

- Interoperability between OWL and FOL ontologies
- Module extraction from large ontologies
- Interoperability between closed-world data and open-world metadata
- Verification of rules translating Dublin Core into PROV
- Maintaining different versions of an ontology in languages with different expressivity
- Metadata within OMS repositories
- Modularity of specifications
- Refinement of specifications
- Consistency among UML models of different types
- Refinements between UML models of different types, and their reuse
- Coherent semantics for multi-language models

* See Section 7 of the DOL standard (pp. 33 to 48).

3. Common Logic is SWeLL

A proposal for the Semantic Web Logic Language (SWeLL) evolved into the ISO/IEC standard for Common Logic (CL).*

The CLIP dialect combines the best features of two dialects, CLIF and CGIF, which are specified in the CL standard.

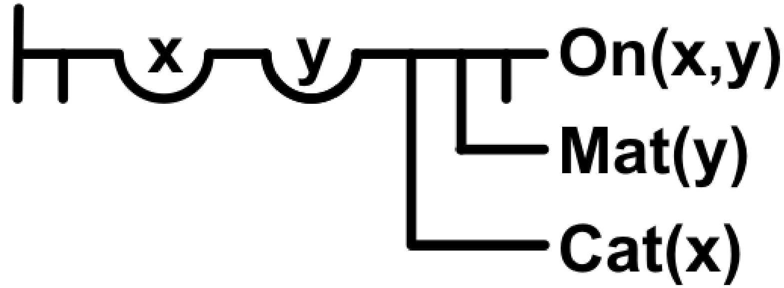
Design goals for CLIP:

- **Immediately readable by anyone who knows predicate calculus.**
- **As readable as Turtle for the RDF and OWL subsets.**
- **As readable as any notation for if-then rules.**
- **Serve as a linearization for a wide range of graph logics, including CGs, EGs, KGs, RDF, OWL, and UML diagrams.**
- **Query option: Select (*list of names*) where (*any CLIP sentence*).**
- **Support mappings of logics ↔ natural languages (NLs).**

*** The LBASE proposal: http://www.w3.org/TR/2003/NOTE-lbase-20031010**

How to say "A cat is on a mat."

Gottlob Frege (1879):

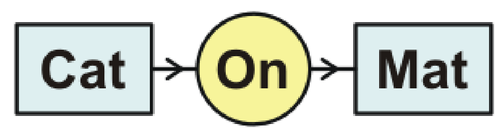


Charles Sanders Peirce (1885): $\Sigma_x\ \Sigma_y\ \mathrm{Cat}_x \bullet \mathrm{On}_{x,y} \bullet \mathrm{Mat}_y$

Giuseppe Peano (1895): $\exists x\ \exists y\ \mathrm{Cat}(x) \land \mathrm{On}(x, y) \land \mathrm{Mat}(y)$

Existential graph by Peirce (1897): **Cat —— On —— Mat**

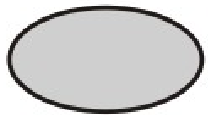
Conceptual graph (1976):



CLIP dialect of Common Logic: **(∃ x y) (Cat x) (On x y) (Mat y).**

Existential Graphs (EGs)

Existence: —

Negation: 

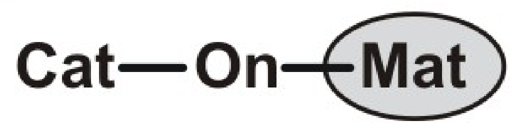
Relations: **Cat- Mat- Pet- Happy- -On- -Under-**

***A cat is on a mat*:** **Cat—On—Mat**

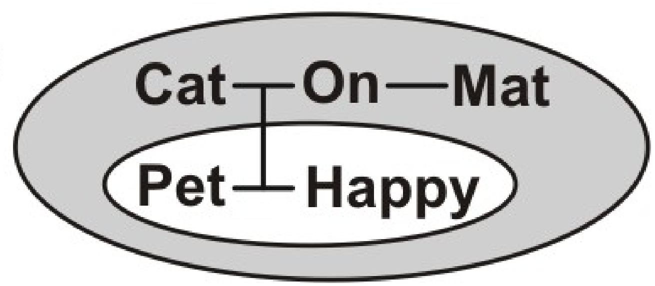
***Something is under a mat*:** **—Under—Mat**

***Some cat is not on a mat*:**

***Some cat is on something that is not a mat*:**



***If a cat is on a mat, then it is a happy pet*:**



The Core CLIP Notation

Existence: (∃ x) or (Exists x)

Negation: ~[ ] but ~[~[ ]] may be written [If [Then]]

Relations: (Cat x), (Mat x), (Pet x), (Happy x), (On x y), (Under x y)

A cat is on a mat: (∃ x y) (Cat x) (On x y) (Mat y).

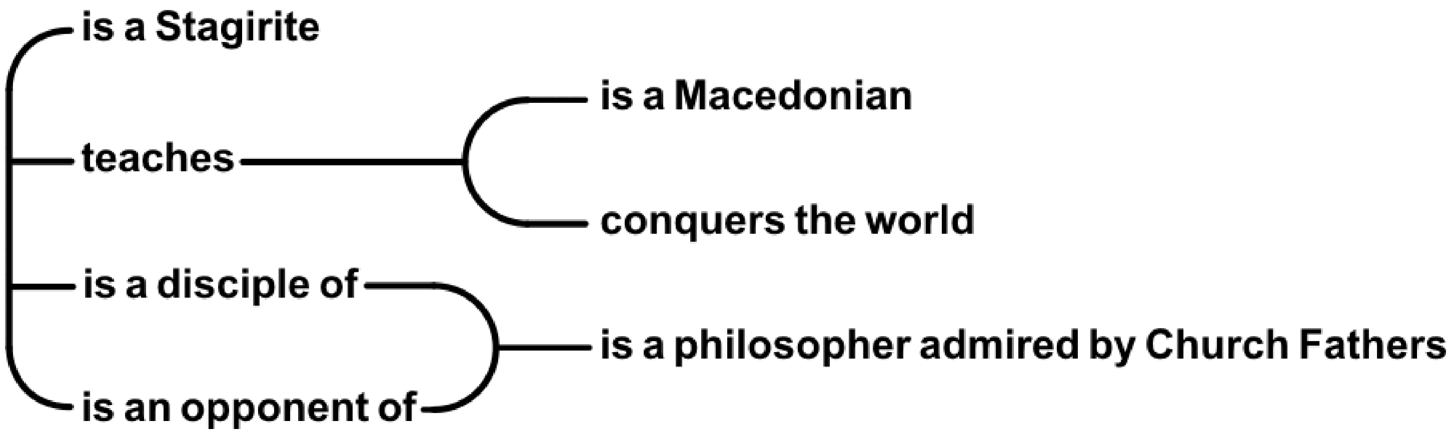
Something is under a mat: (∃ x y) (Under x y) (Mat y).

Some cat is not on a mat: (∃ x) (Cat x) ~[(∃ y) (On x y) (Mat y)].

Some cat is on something that is not a mat:
(∃ x y) (Cat x) (On x y) ~[(Mat y)].

If a cat is on a mat, then it is a happy pet:
[If (∃ x y) (Cat x) (On x y) (Mat y)
[Then (Pet x) (Happy x)]].

One of Peirce's Examples



Peirce's translation to English: ***"There is a Stagirite who teaches a Macedonian conqueror of the world and who is at once a disciple and an opponent of a philosopher admired by Fathers of the Church."***

A translation to CLIP:

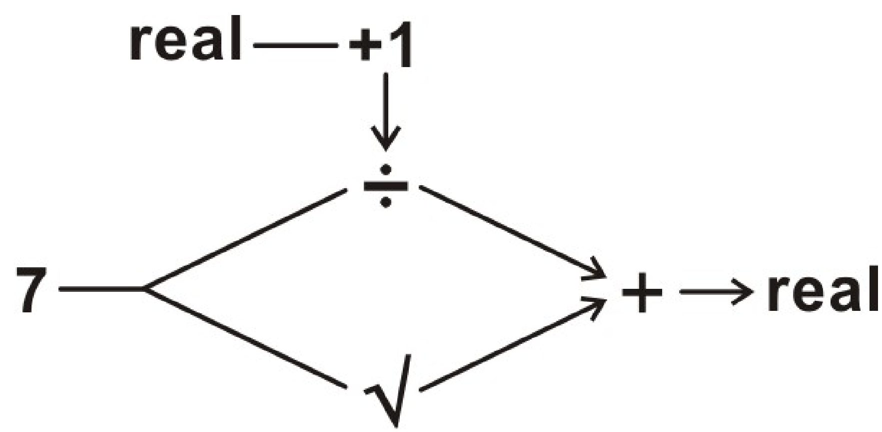
**(∃ x y z) ("is a Stagirite" x) (teaches x y) ("is a Macedonian" y)
("conquers the world" y) ("is a disciple of" x z) ("is an opponent of" x z)
("is a philosopher admired by church fathers" z).**

Without negation, CLIP can represent the content of a relational database or the graph databases of the Semantic Web.

Representing Functions

An example in mathematical notation: $y = 7 \div (x + 1) + \surd\, 7.$

In EGs, a function may be represented as a relation with an arrow for its last line of identity. The four functions may be named +1, ÷, √, +.



A direct mapping of the EG to CLIP:

(∃ (x y u v w)'real) (+1 x → u) (÷ 7 u → v) (√ 7 → w) (+ v w → y).

Another option eliminates the need for the names u, v, w:

(∃ (x y)'real) (= y (+ (÷ 7 (+1 x)) (√ 7)))).

Relating and Integrating Everything

CLIP can relate legacy systems to the latest AI tools.

- **Freely mixing and matching any notations supported by DOL.**
- **Anyone may continue to use their favorite notations indefinitely.**

Semantic Web annotations may be replaced by CLIP:

- **Any URI, enclosed in quotes, is a valid CLIP name.**
- **An annotation that uses the full expressive power of CLIP is written <clip> (*one or more CLIP sentences*) </clip>**
- **Any annotation written in a Semantic Web logic x may be rewritten <clip logic=x> (*one or more CLIP sentences*) </clip>**
- **For tools that do not support CLIP, a preprocesor may translate CLIP annotations to the corresponding SW logic.**

For integrating legacy systems with AI technology,

- **Any software that is described or specified in any UML or SW notation can take advantage of tools that process CLIP.**

4. Relating Logic to Natural Languages

For computers, informal mappings must be formalized.

- **Informal mappings to natural languages (NLs) are OK for humans.**
- **But anything a computer does is formal.**

Discourse Representation Theory specifies a subset of NLs.*

- **DRT is widely used for natural language processing (NLP).**
- **Discourse representation structures (DRSs) support full FOL.**
- **The DRS logic has a precise mapping to EG and to CLIP.**

Semi-automated translation of NLs to and from CLIP:

- **Computer translation of NL → CLIP is error prone.**
- **Computer translation of CLIP → NL is precise, but verbose.**
- **Human translation is as reliable as the human.**
- **Simpler and more reliable: Human-aided computer translation.**

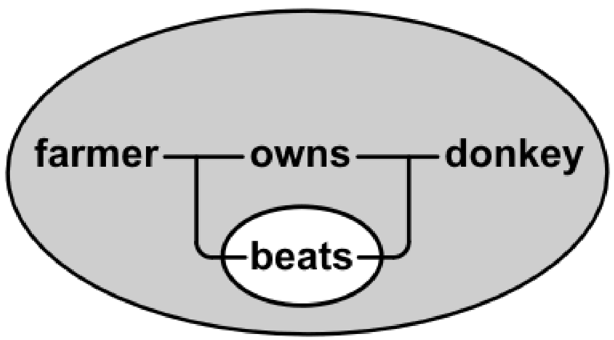
*** Hans Kamp & Uwe Reyle (1993) *From Discourse to Logic*, Dordrecht: Kluwer.**

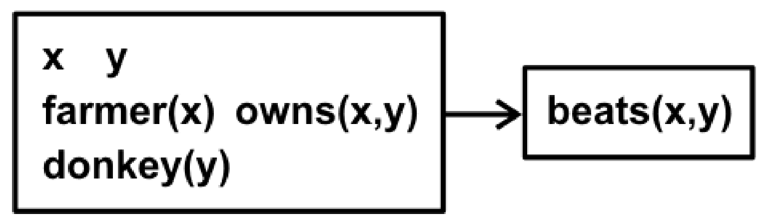
Mapping EG and DRS to CLIP

Peirce and Kamp independently chose equivalent structures.

- **Peirce chose ovals for EG with lines to show references.**
- **Kamp chose boxes for DRS with variables to show references.**
- **But the boxes and ovals represent the same logic in the same way.**

Example: ***If a farmer owns a donkey, then he beats it.***





The EG and DRS may be translated to and from exactly the same CLIP:
[If (∃ x y) (farmer x) (donkey y) (owns x y) [Then (beats x y)]].

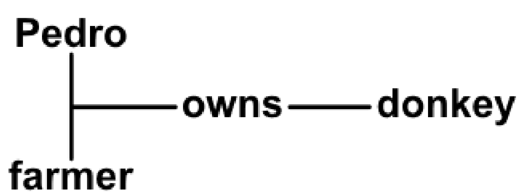
Combining EG Graphs or DRS Boxes

Two English sentences, *Pedro is a farmer. He owns a donkey,* **are represented by EG graphs (left) and DRS boxes (right):**

x
Pedro(x) farmer(x)

y z
owns(y,z) donkey(z)

Combine them by connecting EG lines or merging DRS boxes:



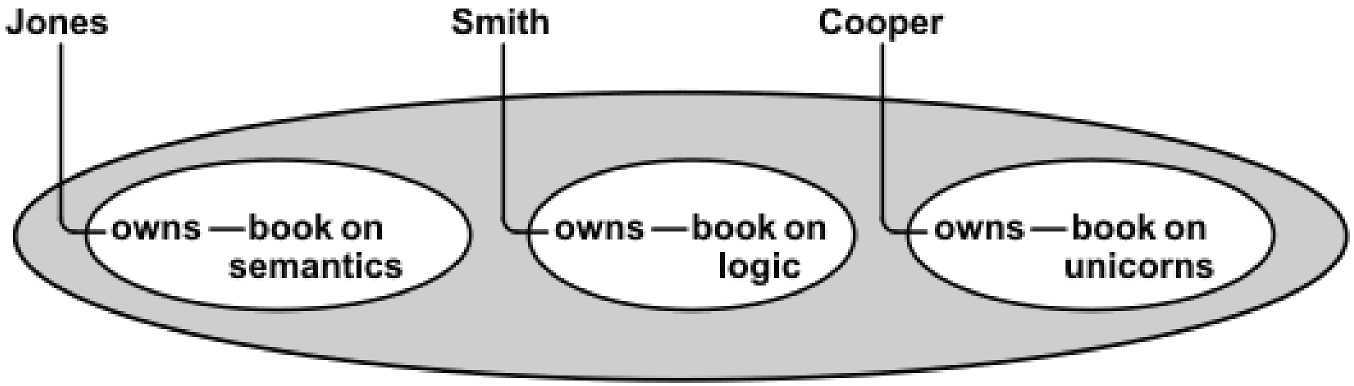
x y z x=y
Pedro(x) farmer(x)
owns(y,z) donkey(z)

Equivalent operations on EG and DRS produce the same CLIP:
(∃ x y z) (Pedro x) (farmer x) (= x y) (owns y z) (donkey z).

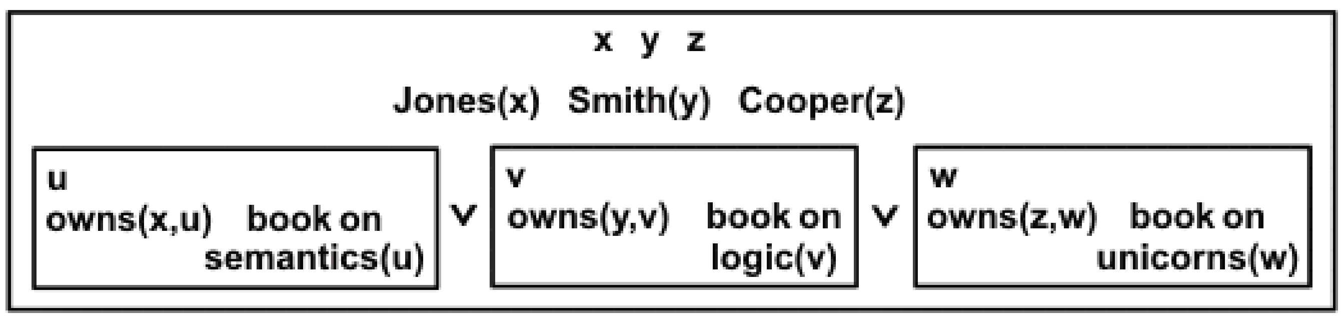
Disjunction in EG, DRS, and CLIP

Kamp and Reyle (1993): ***"Either Jones owns a book on semantics, or Smith owns a book on logic, or Cooper owns a book on unicorns."***

EG:



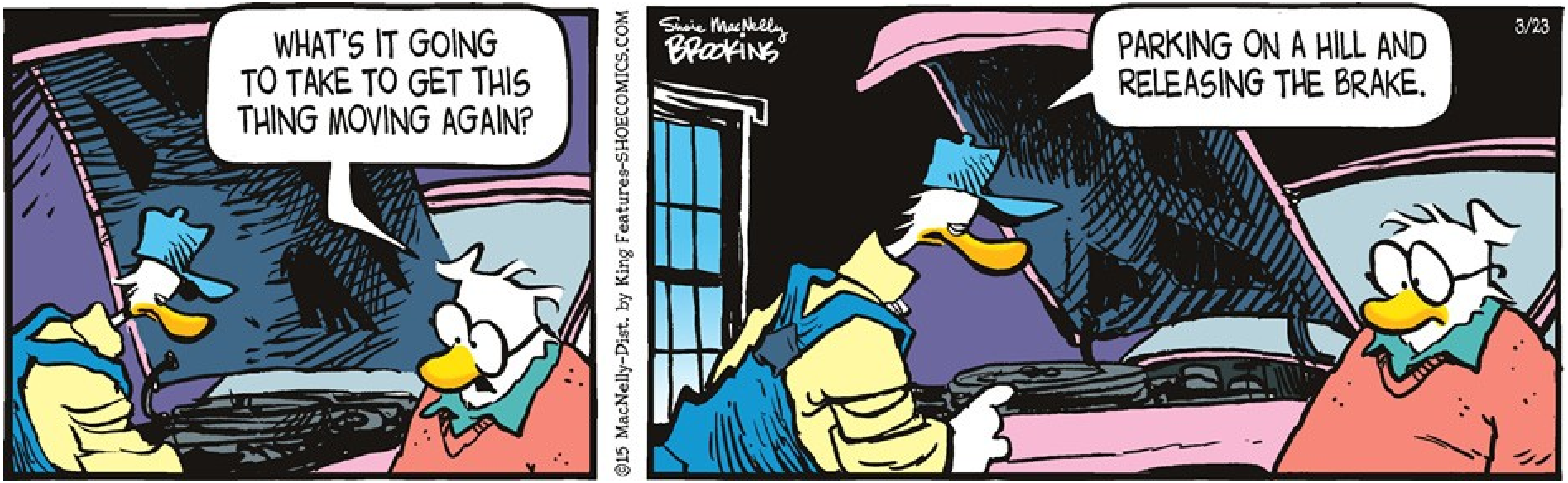
DRS:



CLIP:

```
(∃ x y z)   (Jones x)   (Smith y)   (Cooper z)
[Or    [ (∃ u)   (owns x u)  ("book on semantics" u)  ]
       [ (∃ v)   (owns y v)  ("book on logic" v)  ]
       [ (∃ w)   (owns z w)  ("book on unicorns" w)  ] ].
```

Context and Purpose



Syntax is easy: Parse the question and the answer.

Semantics depends on context and background knowledge:

- **Interpret the meaning of *thing*, *take*, and *move* In this situation.**
- **Apply the laws of physics to understand what would happen.**

Pragmatics depends on the intentions of the participants.

- **No computer system today could understand that cartoon.**
- **Computers should ask people about purpose or intentions.**

*** Source of cartoon: search for 'moving' at http://www.shoecomics.com/**

5. Metalanguage and Metadata

Metalanguage is language about language, natural or artificial.

- To define semantics, Tarski (1933) used logic as a metalanguage for defining the truth value of any statement in logic.
- Annotations in SW logics state metadata about documents.
- But metadata about the metadata can also be useful.

The IKL extension to Common Logic supports metalanguage. *

- IKL enables CLIP to comment on anything expressed in CLIP.
- It can represent metadata about the sources and reliability of data.
- It can support reasoning about metaphor, modality, and the issues of vague, fuzzy, missing, erroneous, or fraudulent information.

Any Unicode strings may be used for CLIP names. Metadata may even be expressed by emojis.

* For the IKL documents, see http://jfsowa.com/ikl/ .

Metalanguage About Situations

The drawing on the right may be interpreted in three ways.

1. Actual: ***Pierre is thinking of Marie, who is thinking of him.***

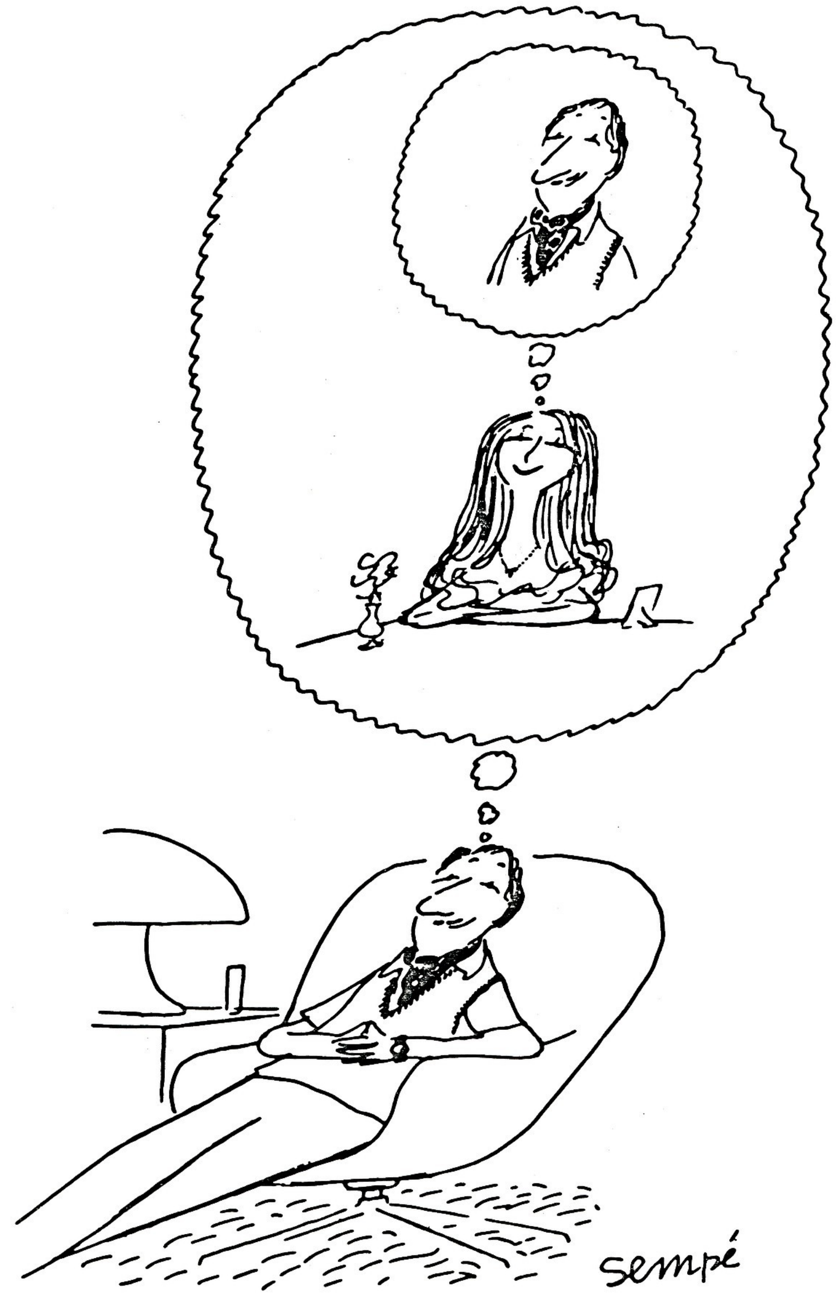
2. Modal: ***Pierre is thinking of Marie, who may be thinking of him.***

3. Intentional: ***Pierre hopes that Marie is thinking of him.***

In the second clause of #1, the verb *is* implies that Pierre's thought is true.

In #2, the verb *may* implies that his thought is a possible proposition.

In #3, the object of the verb *hopes* is a situation Pierre intends in some way.



Propositions and Situations

Three ways of relating a proposition p to a situation s.

- **Actual: p is true or false about s.**
- **Modal: p is related to s in some manner or mode m.**
- **Intentional: Some agent x relates p to s for some reason r.**

English and CLIP for the sentences about Pierre.

English: ***Pierre is thinking of Marie, who is thinking of him.***
CLIP: (thinkingOf Pierre Marie) (thinkingOf Marie Pierre).

English: ***Pierre is thinking of Marie, who may be thinking of him*.**
CLIP: (thinkingOf Pierre Marie) (possible [(thinkingOf Marie Pierre)]).

English: ***Pierre hopes that Marie is thinking of him*.**
CLIP: (hopesFor Pierre [Situation (thinkingOf Marie Pierre)]).

IKL does not support modal logic, but metalanguage can be used to define modality in terms of laws and facts.*

*** See http://jfsowa.com/pubs/worlds.pdf**

6. Automated and Semi-automated Tools

The tools should support a dialogue.

- Explanation requires more interaction than question-answering.
- Anyone from a beginner to an expert should be able to carry on an open-ended conversation about any subject they choose.
- Follow-up questions may drill down to any depth required.
- Computers should accept any language or notation people prefer, and they should read documents without requiring prior annotations.
- If a computer can't understand some text, it should ask people for help. People should answer in their own language.
- Computers may annotate texts, but human assistance is important.

The dialogue may be as precise or vague as the subject matter.

- Human languages can describe a continuous, dynamically changing world at any level of precision that may be required.
- Reasoning methods must be as flexible as human language.

Cognitive Memory (CM)

CM is an associative memory for large volumes of graphs. *

- **Approximate pattern matching for analogies and metaphors.**
- **Associative storage and retrieval of graphs in log(N) time.**
- **Precise pattern matching (unification) for logic and mathematics.**

Analogies can support informal, case-based reasoning:

- **CM can store large volumes of previous knowledge and experience.**
- **Any new case can be matched to similar cases in long-term memory.**
- **Close matches are ranked by a measure of semantic distance.**

Formal reasoning is based on a disciplined use of analogy:

- **Induction: Generalize multiple cases to create rules or axioms.**
- **Deduction: Match (unify) part of a new case with some rule or axiom.**
- **Abduction: Form a hypothesis based on aspects of similar cases.**

*** Survey of CM and applications: http://jfsowa.com/talks/cogmem.pdf**

CLIP for Help and Explanations

CM has been used to support NL dialog and explanations.

- **CM uses CLIP as the basic representation.**
- **But CM can also use any notation that can be mapped to CLIP.**
- **That includes any Semantic Web notations or UML diagrams.**
- **Tools for CLIP and IKL can support natural language and logic. ***

Recommendation for a new generation of development tools:

- **Integrate all systems, including legacy systems, with logic-based methodologies.**
- **Enable subject-matter experts to review, update, and extend their knowledge bases with little or no assistance from IT specialists.**
- **Provide tools that support collaboration, review, and testing by people with different levels and kinds of expertise.**
- **An open-ended variety of software, including NNs, may be used.**

*** Slides about natural logic: http://www.jfsowa.com/talks/natlog.pdf**

Formal Concept Analysis (FCA)

A theory and tools for semi-automated ontology design:

- Theory. Define a minimal lattice that shows all inheritance paths among a set of concept types, each defined by a list of attributes.
- Algorithms. Efficient ways for computing a minimal lattice from a list of terms and defining features.

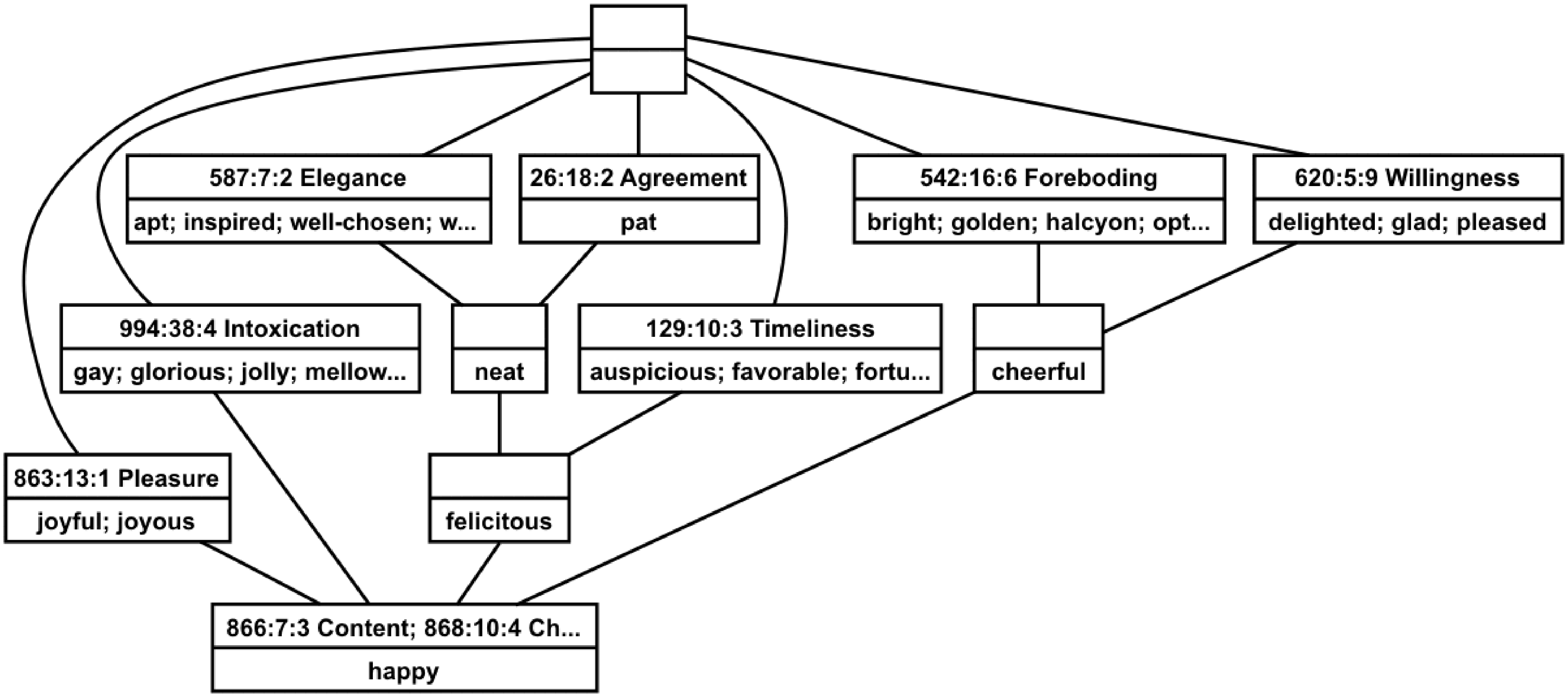
Applications:

- Ontology development and alignment; classification methods; machine learning; defining concepts used in other logics.
- FCA tools are often used to check whether ontologies specified in OWL and other notations are consistent.
- They can also be used to detect inconsistencies among two or more independently developed ontologies.

The FCA Homepage: http://www.upriss.org.uk/fca/fca.html

For deriving lattices from lexical resources: http://www.upriss.org.uk/papers/jucs04.pdf

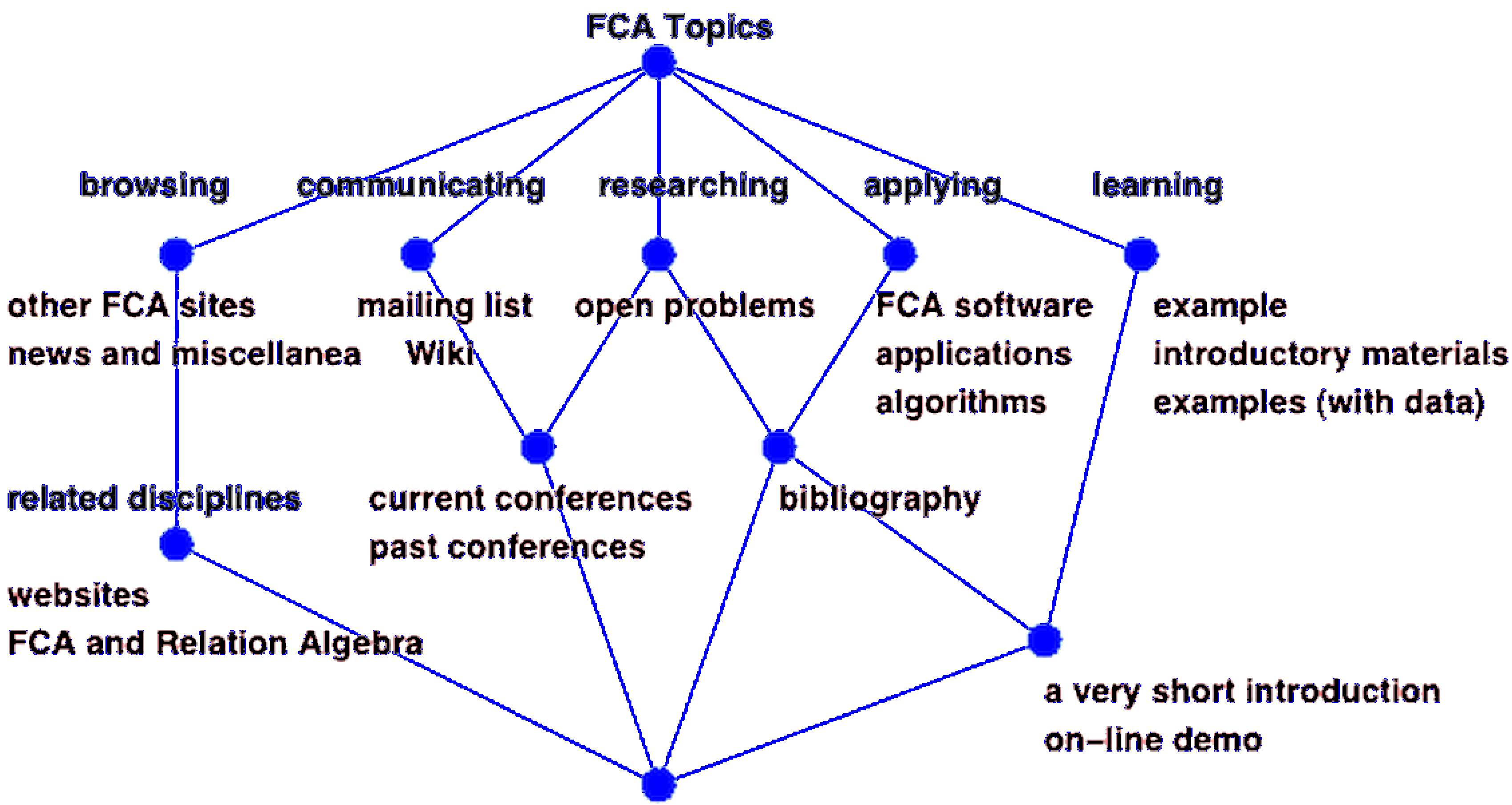
Generating Lattices Automatically



FCA tools used the data in Roget's Thesaurus to generate this lattice for the word 'happy' and its hypernyms (supertypes).

To generate this or similar lattices, enter 'happy' or any other word at the web site http://www.ketlab.org.uk/roget.html

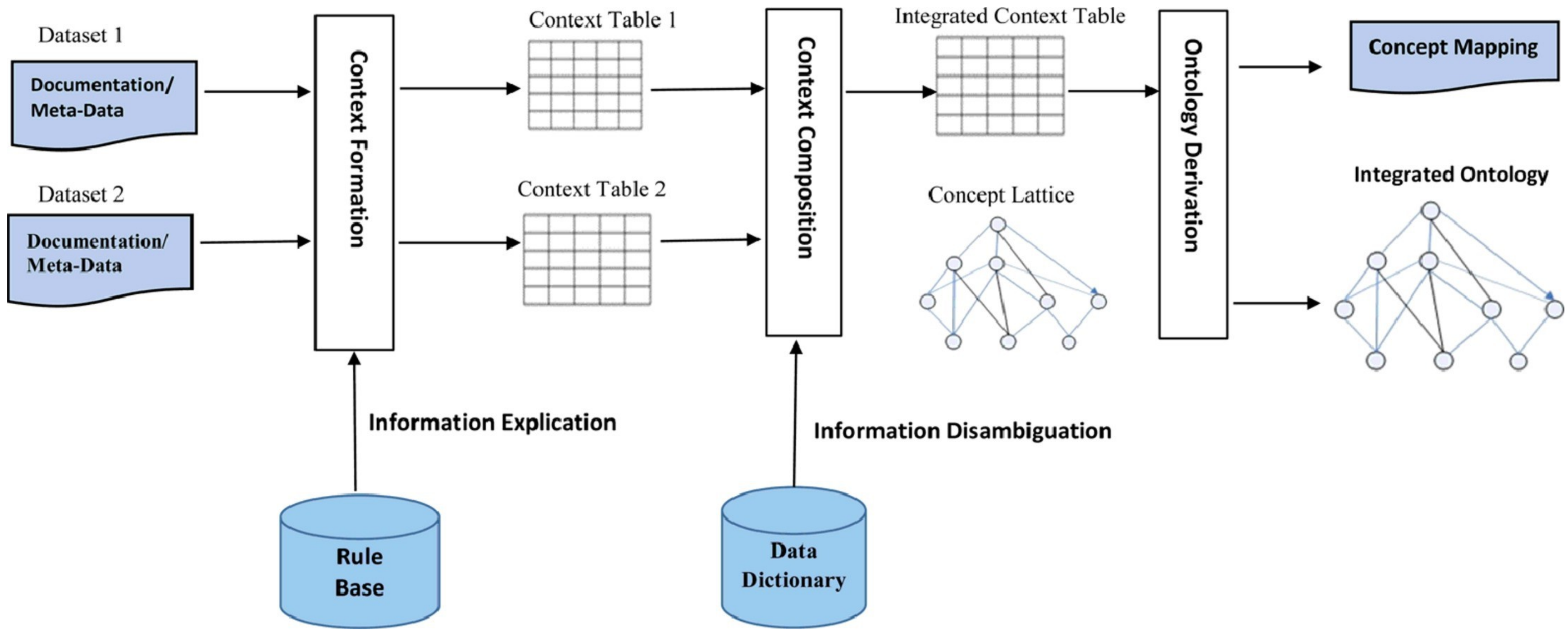
Classifying Resources by Purpose



FCA tools may use a variety of criteria for classification.

- For ontology, the usual criterion is type/subtype.
- But a person who asks a question has some purpose in mind.
- The lattice above classifies resources by purpose, not type.

Using FCA to Merge Ontologies

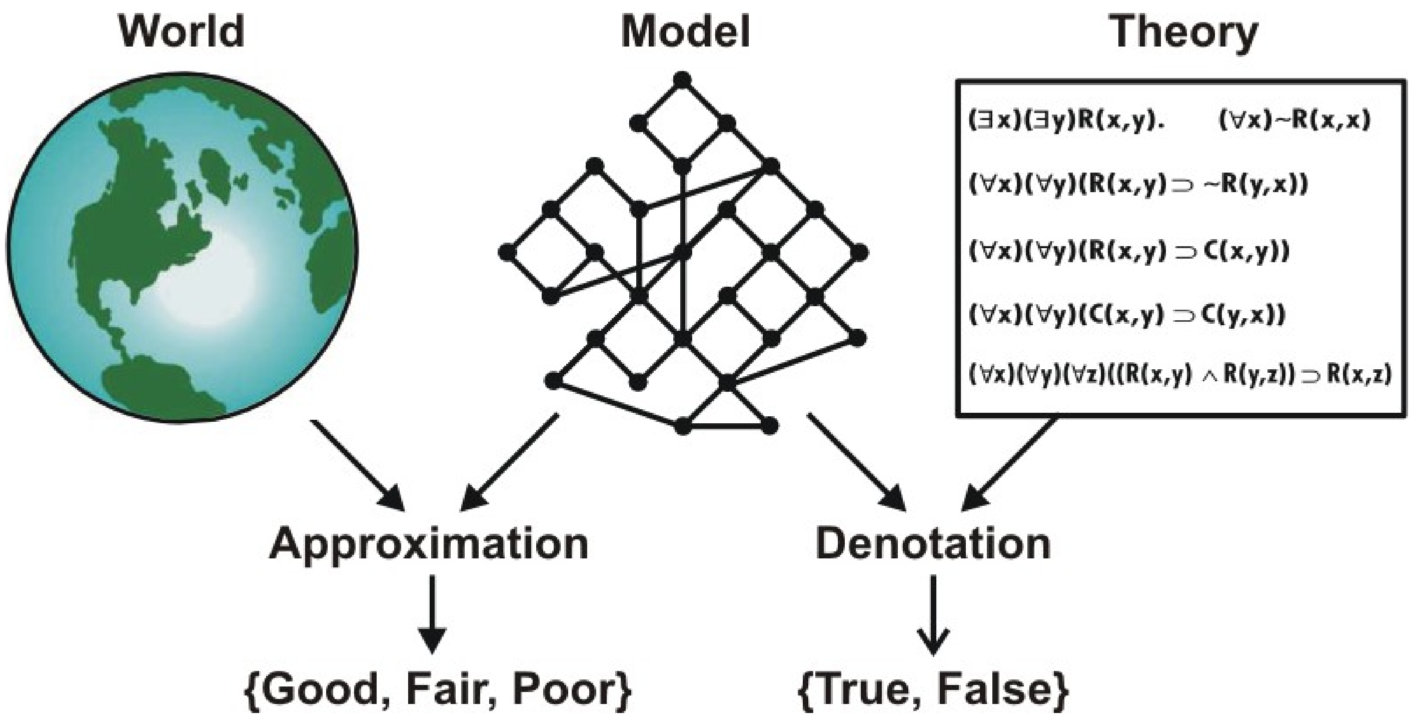


Semi-automated method for integrating diverse ontologies.*

- **Independently developed systems are usually incompatible.**
- **FCA tools can detect similarities and conflicts in definitions.**
- **With some human assistance, the tools can derive a merged ontology that can support data sharing among the systems.**

*** Gaihua Fu, FCA based ontology development for data integration, https://www.sciencedirect.com/science/article/pii/S030645731630019X**

Relating Models to the World



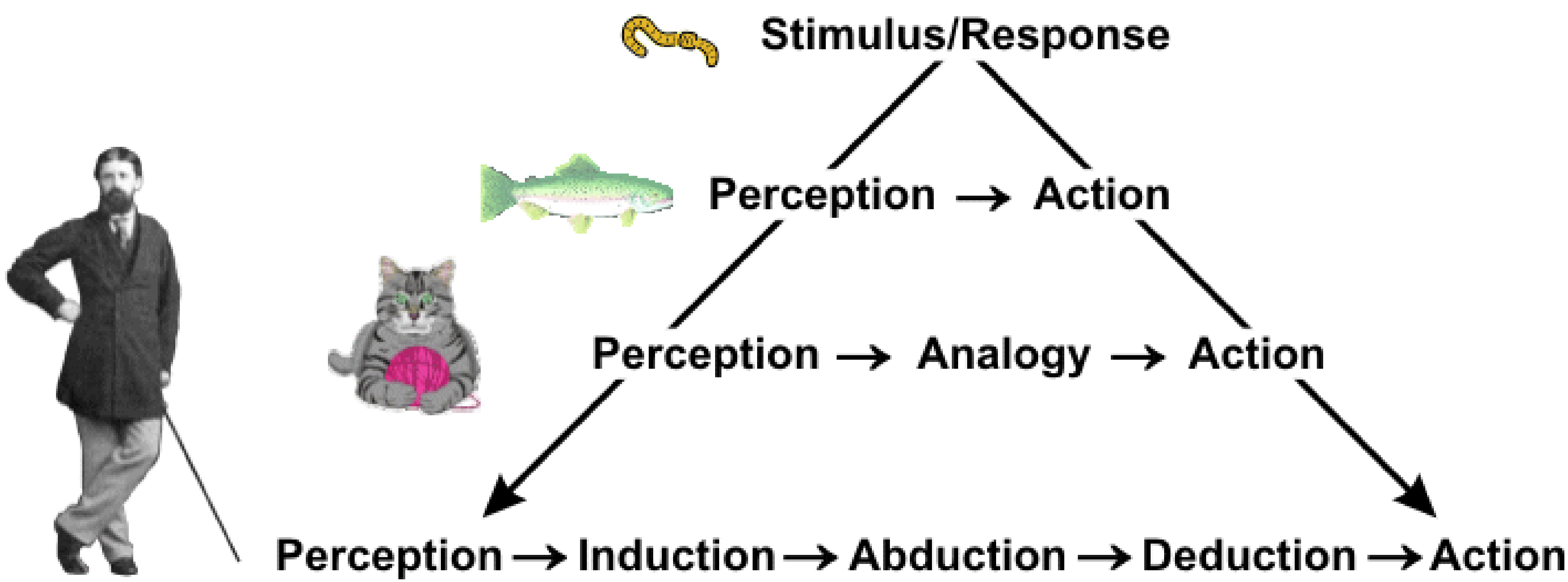
Engineers: "All models are wrong, but some are useful."

- **Discrete symbolic models can be clear, sharp, and precise.**
- **But the world is continuous, disordered, and fuzzy.**

Natural languages are flexible. They can adapt to anything.

- **They can be as vague or precise as the situation requires.**
- **SW tools should be flexible: Detailed levels must be precise, but the top-level ontology must accommodate anything.**

Thinking Beyond the First Second



Perception and classification take one second or less.

- **Neural nets are valuable for learning and recognizing patterns.**
- **By themselves, NNs support S-R (knee-jerk) reactions.**
- **Reasoning by analogy can support a cat-level intelligence.**

Analysis, planning, discovery, and innovation take more time.

- **They require cycles of induction, abduction, deduction...**

Knowledge of Good and Evil

Observation by Immanuel Kant:

> ***Socrates said he was the midwife to his listeners, i.e., he made them reflect better concerning that which they already knew, and become better conscious of it. If we always knew what we know, namely, in the use of certain words and concepts that are so subtle in application, we would be astonished at the treasures contained in our knowledge...***
>
> ***Platonic or Socratic questions drag out of the other person's cognitions what lay within them, in that one brings the other to consciousness of what he actually thought.***

From his ***Vienna Logic***

Some research projects can support such a dialogue.

They could enable us to "drag out" the treasures or the treachery hidden in the Giant Global Graph

Related Readings

ISO/IEC standard 24707 for Common Logic, http://standards.iso.org/ittf/PubliclyAvailableStandards/c039175_ISO_IEC_24707_2007(E).zip

Majumdar, Arun K., John F. Sowa, & John Stewart (2008) Pursuing the goal of language understanding, http://www.jfsowa.com/pubs/pursuing.pdf

Majumdar, Arun K., & John F. Sowa (2009) Two paradigms are better than one and multiple paradigms are even better, http://www.jfsowa.com/pubs/paradigm.pdf

Majumdar, Arun K., & John F. Sowa (2018) Relating language, logic, and imagery, http://jfsowa.com/pubs/relating.pdf

Petarinen, Ahti-Veikko (2011) Moving pictures of thought II, *Semiotica* 186:1-4, 315–331, http://www.helsinki.fi/~pietarin/publications/Semiotica-Diagrams-Pietarinen.pdf

Putnam, Hilary (1982) Peirce the logician, http://jfsowa.com/peirce/putnam.htm

Sowa, John F. (2006) Peirce's contributions to the 21st Century, http://www.jfsowa.com/pubs/csp21st.pdf

Sowa, John F. (2006) Worlds, models, and descriptions, http://www.jfsowa.com/pubs/worlds.pdf

Sowa, John F. (2010) Role of logic and ontology in language and reasoning, http://www.jfsowa.com/pubs/rolelog.pdf

Sowa, John F. (2011) Peirce's tutorial on existential graphs, http://www.jfsowa.com/pubs/egtut.pdf

Sowa, John F. (2013) What is the source of fuzziness? http://jfsowa.com/pubs/fuzzy.pdf

Sowa, John F. (2013) From existential graphs to conceptual graphs, http://www.jfsowa.com/pubs/eg2cg.pdf

Sowa, John F. (2018) Reasoning with diagrams and images, http://www.collegepublications.co.uk/downloads/ifcolog00025.pdf