

Knowledge Design Patterns

Combining logic, ontology, and computation

John F. Sowa

15 February 2016

Outline of This Tutorial

1. What are knowledge design patterns?
2. Foundations of ontology.
3. Syllogisms, categorial and hypothetical.
4. Patterns of logic.
5. Combining logic and ontology.
6. Patterns of patterns of patterns.
7. Simplifying the user interface.

Note: This outline and the section summaries have a green background. Detailed slides are in white. For references such as Tarski (1933), see the bibliography in <http://www.jfsowa.com/bib.htm>

Definition

***A design pattern* is a reusable building block for organizing a structure of any kind – physical or abstract.**

Architects, engineers, programmers, poets, musicians, and chefs use design patterns for assembling their creations.

Design patterns control every aspect of construction, from the smallest details to the global organization.

The choice of patterns depends on

- **Purpose, goal, or intended use of the construction,**
- **Available tools and resources for building it,**
- **Compatibility with other systems that interact with it.**

Mastery of any art depends on the stock of design patterns and the skill in combining them to achieve the intended goal.

1. Knowledge Design Patterns

Three kinds of patterns:

- **Syntax:** The grammar of a notation.
- **Semantics:** Content expressed in the notation.
- **Pragmatics:** Intended ways of using the content.

Three kinds of notations:

- The natural languages that people read, write, and speak.
- Precisely defined linear notations for logic and computation.
- Graphical diagrams and movies for showing patterns that have more dimensions than a linear string.

Natural languages can express all knowledge patterns.

The challenge is to simplify the ways of expressing the patterns and the training required to learn them.

Logic Patterns

First-order logic is a subset or superset of most knowledge representation languages.

FOL is also a subset of all natural languages.

English expresses FOL with the following subset:

- **Two quantifiers: *some* and *every*.**
- **Boolean operators: *and*, *or*, *not*, *if-then*, *if-and-only-if*.**
- **Relations represented by English words.**
- **Pronouns for cross references.**
- **English syntax for combining these operators.**

Predicate calculus uses special symbols instead of words.

Other logics use other patterns for various purposes.

Computer-Oriented Logics

Most computer logics avoid symbols not on the keyboard.

But they usually add patterns for other features:

- **SQL:** Links to tables that store the data.
- **RDF:** XML patterns for embedding in web pages.
- **CLIF:** A syntax that is easy to parse.
- **Conceptual graphs:** A graphic notation for logic.
- **Controlled English:** An English-like syntax that is easy to parse.
- **Prolog:** Horn-clause subset of FOL with procedural extensions.

Some logics add special-purpose ontology:

- **RDFS and OWL:** XML plus a metalevel ontology about ontology.
- **UML diagrams:** Graphic notations that add ontology for software design and specification.

World's Largest Formal Ontology

Cyc project founded by Doug Lenat in 1984:

- **Name comes from the stressed syllable of encyclopedia.**
- **Starting goal: Implement the background knowledge of a typical high-school graduate.**
- **Ultimate goal: Learn new knowledge by reading textbooks.**

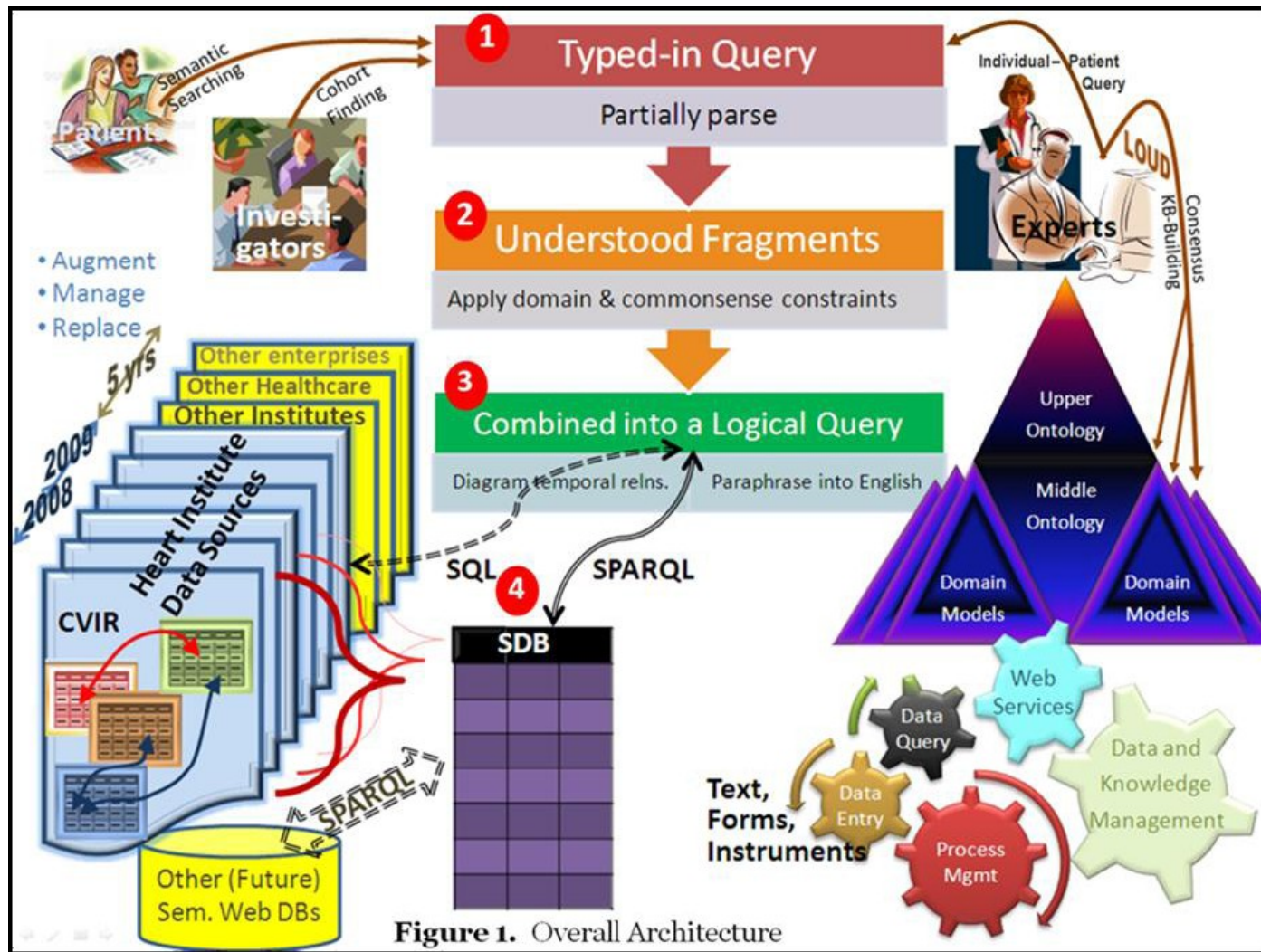
After the first 25 years,

- **100 million dollars and 1000 person-years of work,**
- **600,000 concepts,**
- **Defined by 5,000,000 axioms,**
- **Organized in 6,000 microtheories.**

Some good applications, but more work is needed:

- **Cyc cannot yet learn by reading a textbook.**
- **Cyc software is not well integrated with mainstream IT.**

Cyc at the Cleveland Clinic



Mismatched Design Patterns

Problems at the Cleveland Clinic:

- All the tools process the same semantics.
- But major differences in notations and methodologies.
- Steep learning curve for IT personnel who try to use Cyc.
- Complaint by Terry Longstreth at a DB symposium in 1980:
“Any one of those tools, by itself, is a tremendous aid to productivity. But any two of them together will kill you.”
- Thirty years later, that statement is just as true.

We need better tools, interfaces, and methodologies:

- Experts in any field spend years to become experts.
- They don't have time to learn complex tools and notations.
- The ideal amount of training time is ZERO.
- Subject-matter experts should do productive work on day 1.

Prospects for a Universal Ontology

Many projects, many useful theories, but no consensus.

- **4th century BC: Aristotle's categories and syllogisms.**
- **12th to 16th c AD: Scholastic logic, ontology, and semiotics.**
- **17th c: Universal language schemes by Descartes, Mersenne, Pascal, Leibniz, Newton, Wilkins. L'Académie française.**
- **18th c: More schemes. Satire of the Grand Academy of Lagado by Jonathan Swift. Kant's categories.**
- **19th c: Ontology by Hegel, Bolzano. Roget's Thesaurus. Boolean algebra. Modern science, philosophy of science, early computers.**
- **Late 19th and early 20th c: FOL. Set theory. Ontology by Peirce, Brentano, Meinong, Husserl, Leśniewski, Russell, Whitehead.**
- **1970s: Databases, knowledge bases, and terminologies.**
- **1980s: Cyc, WordNet, Japanese Electronic Dictionary Research.**
- **1990s: Many research projects. Shared Reusable Knowledge Base (SRKB), ISO Conceptual Schema, Semantic Web.**
- **21st c: Many useful terminologies, but no universal ontology.**

2. Foundations of Ontology

Knowledge patterns proposed by early philosophers:

- **Pythagoras:** Mathematical forms govern everything.
- **Heraclitus:** An eternal flux (*panta rhei*) governed by laws (*logos*).
- **Empedocles:** Four elements (fire, air, water, earth) and two principles of change, love (*eros*) and strife (*eris*).
- **Leucippus and Democritus:** Tiny atoms in constant motion.

Disciplined methods for analyzing the patterns:

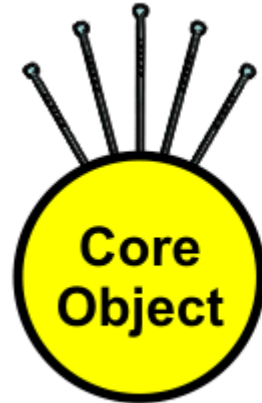
- **Socrates:** Conceptual analysis by systematic questioning.
- **Plato:** Theories presented in dialogs led by Socrates.
- **Aristotle:** Systematic treatises instead of free-flowing dialogs.

Aristotle's patterns set the standard for logic and ontology:

- **Stable objects as composites of form and matter.**
- **Ten categories for analyzing, describing, and classifying anything.**
- **Logic for specifying patterns and reasoning about them.**

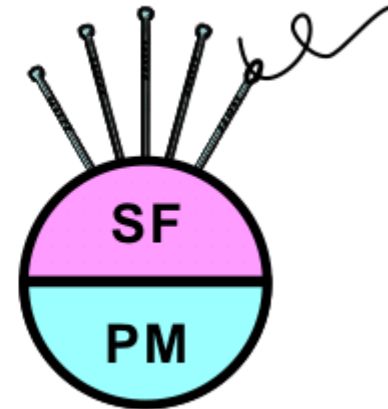
Differences Between Plato and Aristotle

Properties



Plato's Ontology

Accidents



Aristotle's Ontology

Paul Spade drew “pincushion diagrams” to illustrate ontologies.

For Plato, each object is represented as a bundle of properties (pins) contained in a receptacle (the pincushion).

The pins for optional properties can be removed or replaced.

But the pins for essential (necessary) properties have a barbed hook that cannot be removed without destroying the pincushion.

Aristotle's Ontology

The pincushions for Aristotle's ontology contain an unchangeable substantial form (SF) and changeable prime matter (PM).

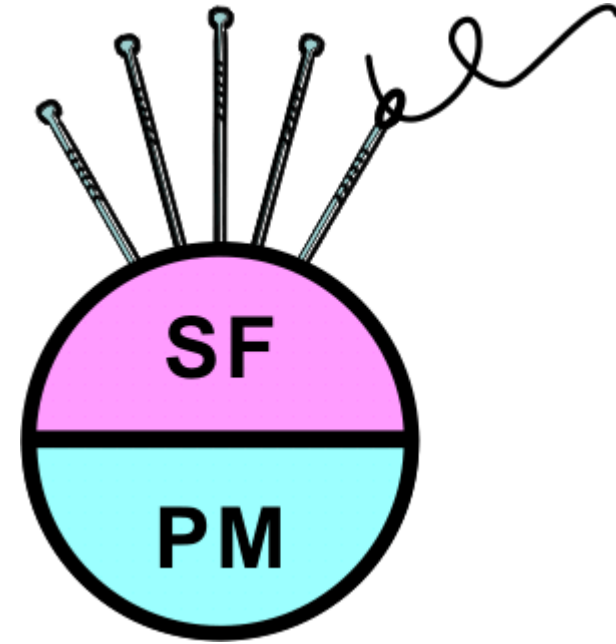
All pins represent *accidents* that can be removed or replaced.

For relations, the pins resemble needles with threads linked to the needles of other objects.

Although all accidents are changeable, some kinds of accidents are necessarily present.

For example, all physical objects must have weight, but the weight can change as the amount of matter changes.

For such accidents, the SF determines preassigned slots that must always be filled with some pin of the appropriate type.



Aristotle's Categories

Ten ways of describing anything that exists or can exist.

Each category has a corresponding question:

- **Substance – What is it?**
- **Relation – Toward what?**
- **Quantity – How much?**
- **Quality – What kind?**
- **Activity – Doing what?**
- **Passivity – Undergoing what?**
- **Condition – Having what?**
- **Position – How situated?**
- **Place – Where?**
- **Time – When?**

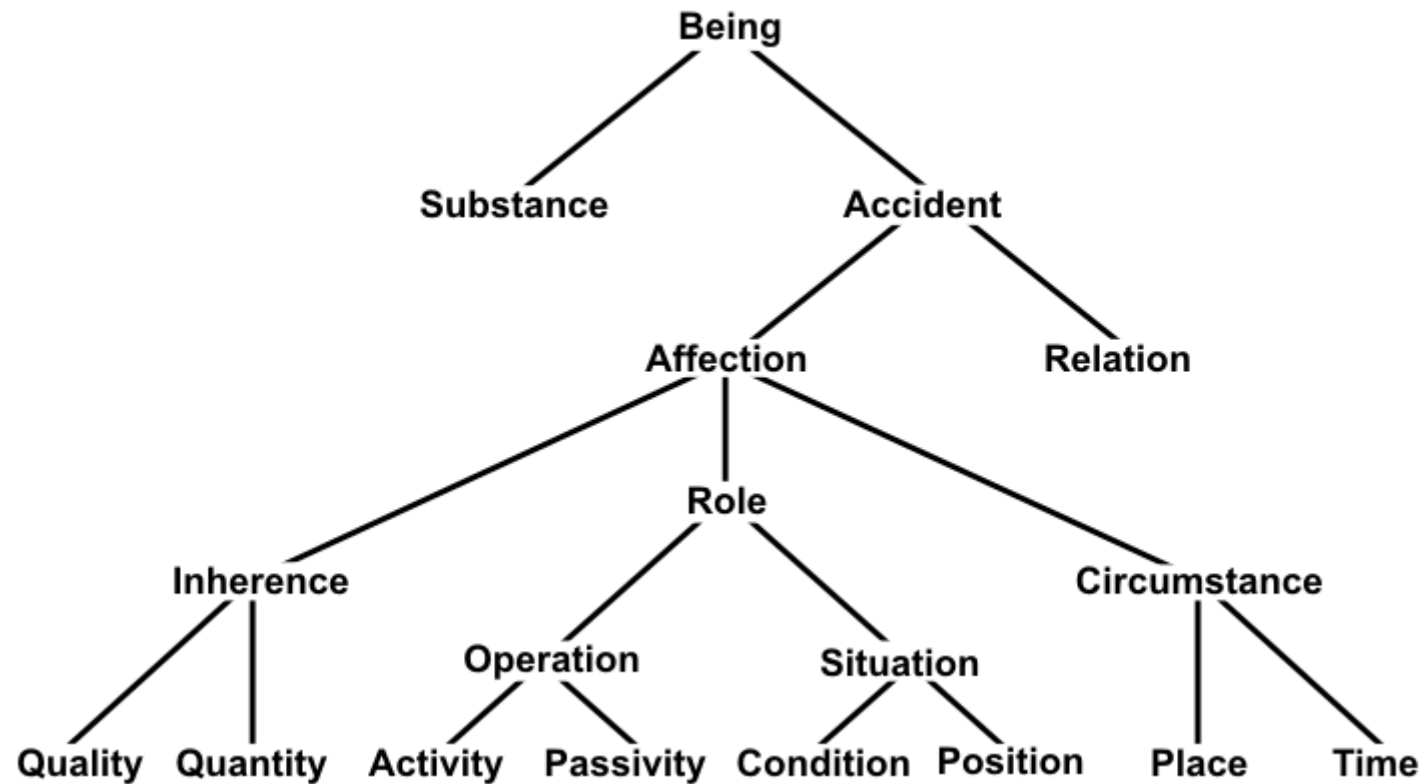
Substance is the unchanging form that determines what something is.

The other nine categories describe accidents that can change.

Describing George Washington

Category	Question	Answer
Substance	What is it?	A man
Relation	Toward what?	A general of the US army
Quality	What kind?	Handsome
Quantity	How much?	Tall
Activity	Doing what?	Talking
Passivity	Undergoing what?	Listening
Condition	Having what?	Victory in battle
Position	How situated?	Mounted on a horse
Place	Where?	Yorktown, Virginia
Time	When?	19 October 1781, 2 pm

Tree of Aristotle's Categories



Aristotle's categories, as arranged by Franz Brentano (1862).

The ten categories are the endpoints (leaves) of the tree.

The branch points are based on writings by Aristotle.

Aristotle's Pattern for Definitions

Define categories by *genus* and *differentiae*:

- Genus – a supertype of the category to be defined.
- Species – the subtype that is being defined.
- Differentia – a property or attribute that distinguishes the species to be defined from other species of the same genus.

This pattern is still the standard for a good definition.

Ideally, the differentiae should state necessary and sufficient conditions that uniquely distinguish the species.

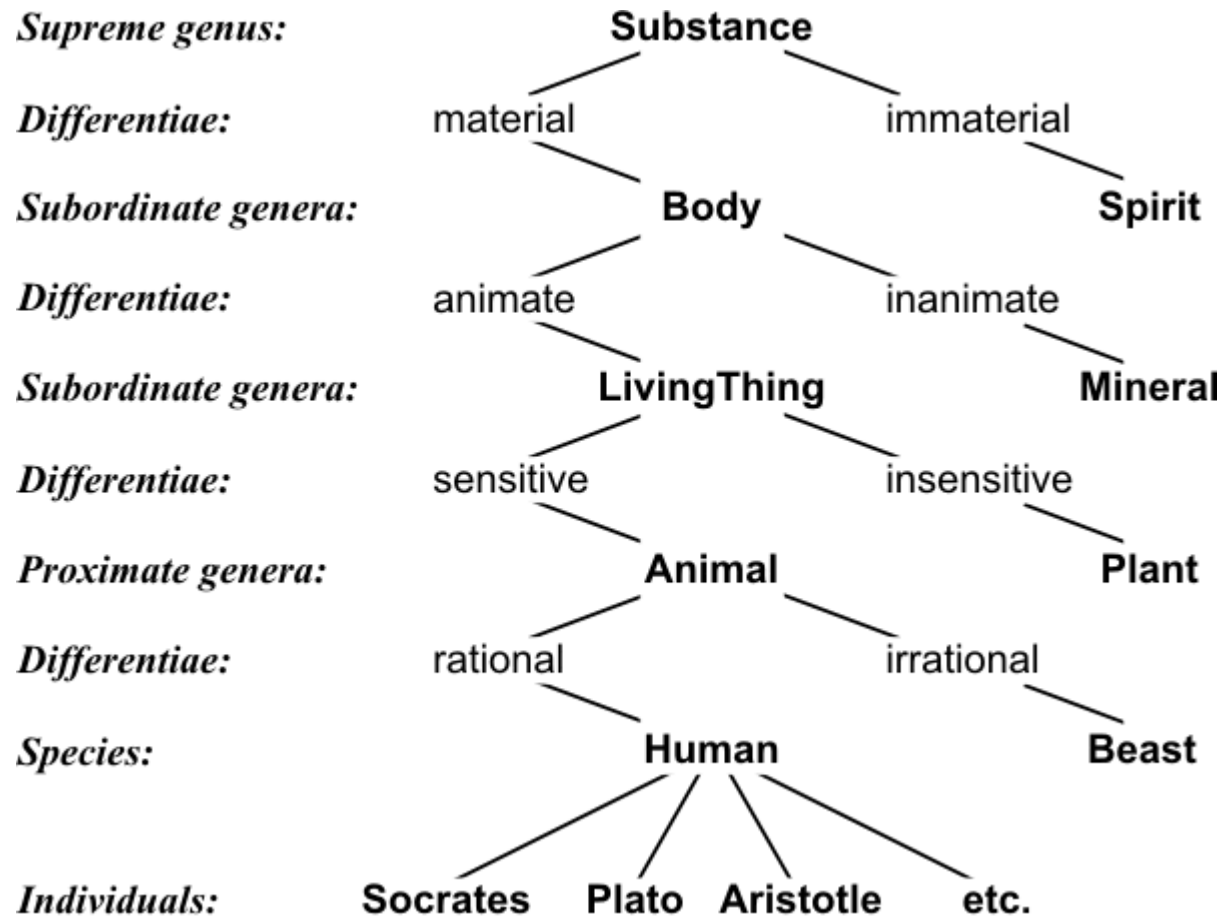
That goal is achieved in mathematics.

But it is rarely possible for most artifacts and natural kinds.

As a discovery method in biology, Aristotle used an alternative:

- For each species, describe a type specimen in detail.
- Use looser criteria of similarity to determine the genus.

Tree of Porphyry



A tree of categories and differentiae was found in a commentary on Aristotle by the philosopher Porphyry (3rd century AD).

This tree is based on a version by Peter of Spain (1239).

Body and Soul

All the Greeks used the word *psychê*, which is translated as *anima* in Latin and *soul* in English.

Plato assumed a separable psyche, which uses the body as the instrument of perception and action.

But Aristotle assumed that every living thing has an embodied psyche, which serves as its substantial form:

- **Nutritive psyche for plants.**
- **Sensitive psyche for sessile animals like sponges and clams.**
- **Locomotive psyche for worms.**
- **Psyche with imagery for animals with eyes.**
- **Rational psyche for humans.**

The psyche controls all growth and motion from birth to death.

The psyches of the more advanced animals inherit all the functions of the more primitive psyches.

Experimental Science

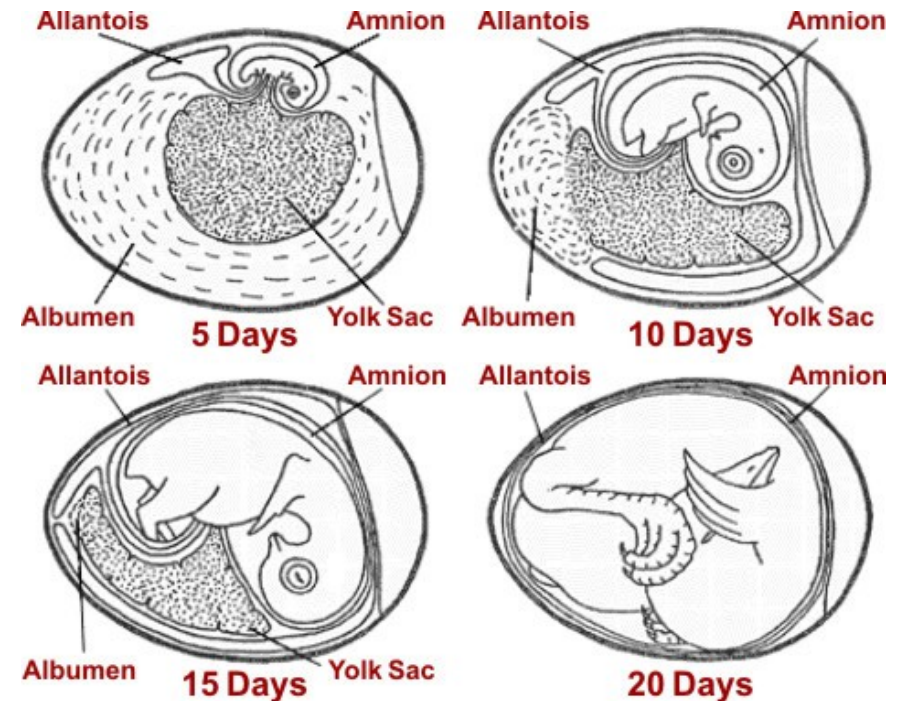
As a mathematician, Plato believed that mathematics is the the only reliable source of knowledge.

But Aristotle's father was a physician, who emphasized detailed observation and careful records.

To show the power of observation, Aristotle and his students carried out a classic experiment:

- Select about two dozen chicken eggs laid on the same day.
- Break open one egg each day, write a detailed description of the embryo, and analyze the stages of growth. *

As late as the 17th century, the physician William Harvey said that Aristotle's writings on embryology were still unsurpassed.



* See http://chickscope.beckman.uiuc.edu/resources/egg_to_chick/development.html

Modes of Explanation

Aristotle defined four modes or principles for explaining change.

Cicero translated Aristotle's word *aitia* to Latin as *causa*, but the English word *cause* has shifted in meaning.

Aristotle's four aitia:

- **Efficient:** Something that initiates a change or carries it through.
- **Material:** The matter or resources used in the change.
- **Formal:** The basic form, plan, or pattern of the change.
- **Final:** The goal, purpose, or function (in Greek, *telos*).

The English word *cause* is usually limited to the efficient aitia.

In biological explanations, Aristotle emphasized the function (final aitia or telos) of any organ or feature of a living thing.

This principle is still important for understanding how and why living things and their parts grow and operate.

3. Syllogisms

Aristotle coined the word *syllogism* for any valid pattern of inference. Today, they're called rules of inference.

Three kinds of syllogisms developed in antiquity:

- **Categorical syllogisms for reasoning about categories.**
- **Hypothetical syllogisms based on *if-then* patterns.**
- **Disjunctive syllogisms based on inclusive and exclusive *or*.**

All these patterns (and many more) are still important.

In fact, the overwhelming majority of OWL ontologies today don't use any logic beyond what Aristotle defined.

But Aristotle used a more readable notation based on a controlled natural language (CNL).

Patterns that Relate Categories

Four sentence patterns used in categorical syllogisms:

A	Universal affirmative:	Every S is P.
I	Particular affirmative:	Some S is P.
E	Universal negative:	No S is P.
O	Particular negative:	Some S is not P.

Boethius (6th century AD) introduced the letters A, I, E, and O as mnemonics for naming and remembering the combinations:

- A and I are the first two vowels in the Latin *affirmo* (I affirm).
- E and O are the first two vowels in *nego* (I deny).

The letters S and P represent *terms* in the subject and predicate.

Each term is a word that names a category or a phrase that states some *differentia* for defining a category.

Categorical Syllogisms

Barbara, the name of the first pattern, has three As, which indicate three type A sentences:

- A Every animal is a living thing.**
- A Every human is an animal.**
- A Therefore, every human is a living thing.**

The pattern named Darii has sentence types A, I, I:

- A Every beast is irrational.**
- I Some animal is a beast.**
- I Therefore, some animal is irrational.**

The pattern Barbara supports the inheritance of properties from a supertype to every individual of a subtype.

The pattern Darii supports the inheritance of properties from a type to particular individuals of a different type.

Negative Syllogisms

The first negative syllogism is named Celarent:

- E No body is a spirit.**
- A Every animal is a body.**
- E Therefore, no animal is a spirit.**

The negative syllogism Ferio applies to particular individuals:

- E No plant is rational.**
- I Some living thing is a plant.**
- O Therefore, some living thing is not rational.**

E and O sentences express constraints on the type hierarchy.

Negative syllogisms derive the implications of those constraints.

Syllogisms with Other Verbs

Sentences with the verb *has* and an arbitrary object:

- A Every quadruped has four legs.
- A Every elephant is a quadruped.
- A Therefore, every elephant has four legs.

But any verb or verb phrase can be used:

- A Every mammal breathes oxygen.
- I Some sea creature is a mammal.
- I Therefore, some sea creature breathes oxygen.

With a suitable paraphrase, all A, I, E, O sentences can be converted to a pattern with *is* as the main verb:

- Every quadruped is an animal with four legs.
- Every mammal is an oxygen-breathing animal.

Other Paraphrases

The pattern Cesare has the same predicate in both premises:

- E No oyster has lungs.
- A Every mammal has lungs.
- E Therefore, no mammal is an oyster.

With a paraphrase, Cesare can be converted to Celarent:

- E No animal with lungs is an oyster.
- A Every mammal is an animal with lungs.
- E Therefore, no mammal is an oyster.

Negating some verbs requires the auxiliary verb *does*:

- E No herbivore eats meat.
- I Some mammal is a herbivore.
- O Therefore, some mammal does not eat meat.

Patterns of Categorical Syllogisms

Each categorical sentence affirms or denies a type-subtype relationship between two categories.

The verb *is* links terms that can be either subjects or predicates.

Terms that use other verbs can only occur as predicates.

- A Every quadruped has four legs.
- I Some sea creature breathes oxygen.
- E No oyster has lungs.
- O Some mammal does not eat meat.

There are 256 possible patterns of 3-sentence syllogisms.

But only 19 of those patterns are logically valid.

The other 237 patterns are fallacious: they could derive a false conclusion from true premises.

The 19 Valid Patterns

Aristotle chose Barbara and Celarent as the two basic patterns.

The other 17 valid patterns are derived by conversion rules.

Darii is derived from Barbara, and Ferio from Celarent.

Barbara, Celarent, Darii, and Ferio belong to the *first figure*.

The other valid patterns belong to figures 2, 3, and 4:

2. Cesare, Camestres, Festino, Baroco.

3. Darapti, Felapton, Disamis, Datisi, Bocardo, Ferison.

4. Bamalip, Calames, Dimatis, Fesapo, Fresison.

Patterns whose name begins with B are derived from Barbara; with C, from Celarent; with D, from Darii; with F from Ferio.

The consonants after the vowels indicate the conversion rules.

Patterns of the Four Figures

$$\begin{array}{c} M - P \\ S - M \\ \hline \therefore S - P \end{array}$$

Figure 1

$$\begin{array}{c} P - M \\ S - M \\ \hline \therefore S - P \end{array}$$

Figure 2

$$\begin{array}{c} M - P \\ M - S \\ \hline \therefore S - P \end{array}$$

Figure 3

$$\begin{array}{c} P - M \\ M - S \\ \hline \therefore S - P \end{array}$$

Figure 4

The letters S, P, and M represent terms:

- S is the term that appears in the subject of the conclusion.
- P is the term in the predicate of the conclusion.
- M is the middle term. It appears in the premises, but not the conclusion.

For example, Felapton belongs to the third figure.

It would have E, A, O sentences in the following pattern:

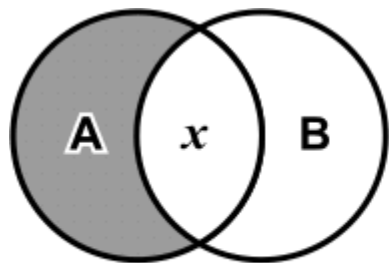
No M is P.

All M is S.

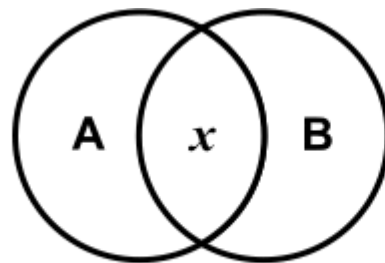
Therefore, some S is not P.

The name and the figure determine the pattern of sentences.

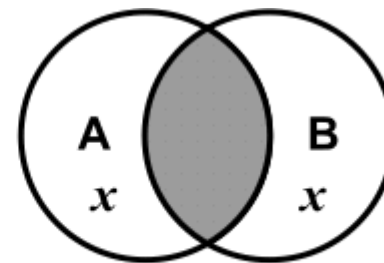
Venn Diagrams



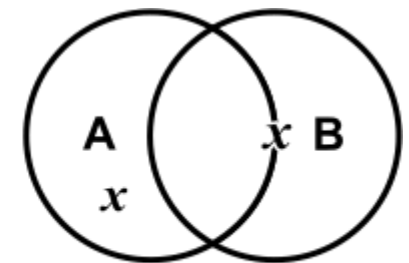
Every A is B



Some A is B



No A is B



Some A is not B

In the 19th century, John Venn used patterns of overlapping circles to determine the valid patterns of syllogisms.

Each circle contains every instance x for which some A is true.

Four kinds of areas:

- Shaded area: known to be empty – no instances exist.
- White area: contents unknown – some instance x might exist.
- White area marked with x : contains at least one instance x .
- Border marked with x : some x exists on one side or the other.

Note: Aristotle assumed that every category has at least one x .

Venn Diagrams for Barbara and Darii

For the pattern Barbara, the area of A outside B is empty, and the area of C outside A is empty.

For Darii, the area of A outside B is empty, but the area of C outside A is irrelevant.

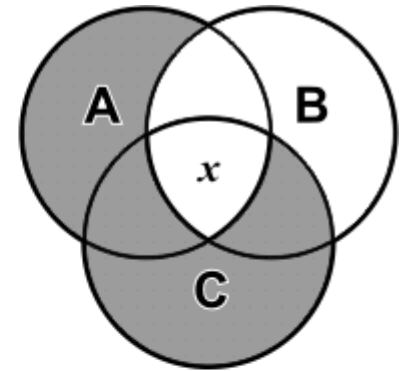
Therefore, any data that makes Barbara true will also make Darii true.

This observation shows that Aristotle's derivation of the pattern Darii from Barbara is valid: it preserves truth.

Exercise: Draw Venn diagrams to show how patterns of the second, third, and fourth figures can be derived from the first.

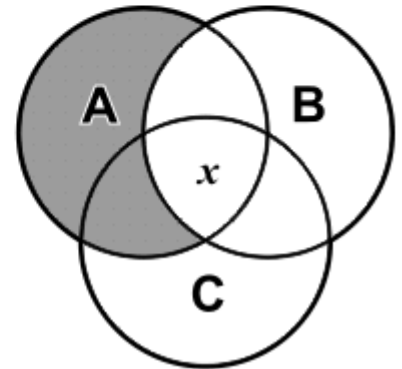
Every A is B.
Every C is A.

∴ Every C is B.



Every A is B.
Some C is A.

∴ Some C is B.



Model Theory

Alfred Tarski (1933) developed model theory for first-order logic, but Venn invented an equivalent model theory for syllogisms.

For First-order logic, a Tarski model consists of

- **A set D of individuals in the domain of discourse.**
- **For each N -adic relation, a set of N -tuples of individuals in D for which the relation is true.**
- **A method for evaluating the truth of any FOL statement in terms of the set D and the sets of N -tuples.**

For syllogisms, Venn's method is equivalent to Tarski's:

- **A set D of individuals in the domain of discourse.**
- **For each monadic relation, a circle that represents the set of individuals (1-tuples) in D for which the relation is true.**
- **A method for evaluating the truth of any A, I, E, O statement in terms of the set D and the sets of 1-tuples.**

Propositional Logic

Aristotle made some comments about propositional logic.

He even used variables to represent propositions:

- **“If when A is, B must be, then when B is not, A cannot be.”**
- **In Latin, this rule is called *modus tollens* (mode of taking away).**

The Stoic philosophers developed propositional logic further.

In his commentaries, Boethius summarized the Stoic logic.

- **When a formula mentions two propositions, Boethius used the Latin *hoc* (this) for the first and *illud* (that) for the second.**
- **But when a formula requires more than two, he used letters.**
- **Example: “Sicut cum A est, B est, C est.”**
If (when A is, B is), C is.
 $(A \supset B) \supset C$.

Hypothetical Syllogisms

A hypothetical syllogism uses *if-then* rules with arbitrary sentence patterns and two rules of inference.

The rule *modus ponens* (mode of putting) makes an assertion:

Given	If A, then B.
But	A.
Therefore	B.

The rule *modus tollens* (mode of taking away) denies the if-part:

Given	If A, then B.
But	not B.
Therefore	not A.

In modern rule-based systems, a deduction by repeated *modus ponens* is called *forward chaining*.

Repeated *modus tollens* is called *backward chaining*.

Hypothetical and Disjunctive Syllogisms

If p, then q.
But p.

∴ q.

Modus ponens

If p, then q.
But not q.

∴ not p.

Modus tollens

p or q.
But not p.

∴ q.

Inclusive or

p xor q.
But p.

∴ not q.

Exclusive or

p xor q.
But not p.

∴ q.

Exclusive or

The word *or* is ambiguous in natural languages.

The Stoic philosophers recognized that ambiguity.

They defined one pattern for inclusive or; two for exclusive or.

By convention, the Latin logicians used *vel* for inclusive or and the pattern *aut ... aut ...* for exclusive or.

But in ordinary usage, the Latin *aut ... aut ...* was just as ambiguous as the English *either ... or ...*

Disjunctive Syllogisms

A disjunctive syllogism is based on *either-or* patterns.

The basic disjunctive syllogism allows arbitrary sentences:

Given	Either A or B.
But	not A.
Therefore	B.

A disjunctive syllogism can also be applied to the predicate of a type I sentence, as in the following argument:

Given	Every philosopher is either sleeping or debating.
But	Socrates is a philosopher.
Therefore	Socrates is either sleeping or debating.
But	Socrates is not sleeping.
Therefore	Socrates is debating.

Note that a disjunctive syllogism cannot be applied to the first premise because it is a type A sentence.

Medieval Scholastics

The Scholastic logicians organized and systematized the contributions by Greek, Roman, and Arabic philosophers:

- **Aristotle's syllogisms, categories, and definitions.**
- **Propositional logic.**
- **Diagrams for showing relationships.**
- **Controlled Latin for the notation.**

They also made many innovations and extensions:

- **Ontology of suppositions for reasoning about fictional, hypothetical, and metalevel language and logic.**
- **Modal and temporal operators and rules of inference.**
- **Model-theoretic semantics for Latin by William of Ockham.**
- **Methodology of conceptual analysis.**

Unfortunately, Renaissance scholars ridiculed the “logic chopping.” In logic, the 17th century was less advanced than the 14th.

Observations

Aristotelian-Stoic-Scholastic logic was a brilliant achievement.

It is the foundation for *description logics* such as OWL.

The methods of conceptual analysis are essential for all work in ontology, semantics, and knowledge representation.

But traditional logic is not sufficiently expressive:

- **The controlled NL for syllogisms is a small subset of full NL.**
- **Euclid needed a larger subset for his textbook on mathematics.**
- **Computer science requires full first-order logic – or more.**

The patterns of syllogisms are useful, but the conversion rules are too complex.

Venn diagrams show that simple but general patterns can make a complex subject easier to learn, remember, and use.

4. Patterns of Logic

Several kinds of syntax:

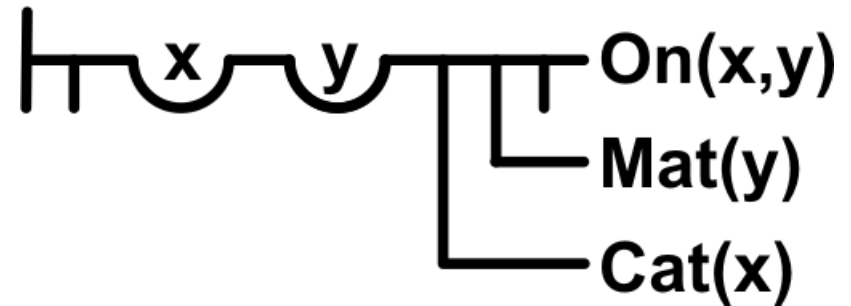
- **Algebraic:** Boolean algebra and predicate calculus.
- **General purpose diagrams:** Frege's trees, Peirce's graphs.
- **Special purpose diagrams:** Type hierarchies, UML diagrams.
- **Controlled natural languages.**
- **Miscellaneous:** Mixed notations for a variety of purposes.

Design choices:

- **Optimize readability.**
- **Optimize expressive power.**
- **Tailor the notation for the reasoning method.**
- **Tailor the notation for special kinds of applications.**
- **Mix some special-purpose ontology with the logic.**

Patterns for Saying “*A cat is on a mat.*”

Frege (1879) designed a tree notation whose structure was tailored to the patterns of his axioms and rules of inference.



But nobody else adopted his notation.

Peirce added *quantifiers* and *relations* to Boolean algebra.

Peano adopted Peirce's algebra, but with different symbols:

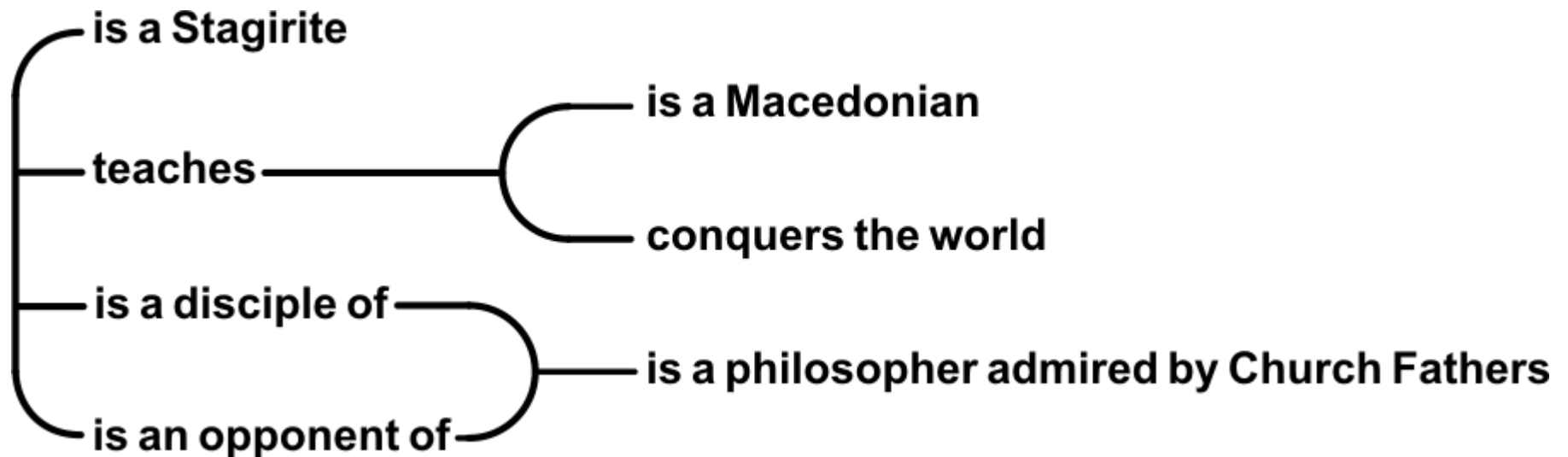
Peirce (1885): $\Sigma_x \Sigma_y \text{Cat}_x \cdot \text{Mat}_y \cdot \text{On}_{x,y}$

Peano (1895): $\exists x \exists y \text{Cat}(x) \wedge \text{Mat}(y) \wedge \text{On}(x,y)$

But in 1897, Peirce invented *existential graphs* as a simple, readable notation with extremely simple rules of inference:

Cat — On — Mat

Existential Graphs Without Negation



Simple patterns that can represent the content of many graphical notations, including RDF, Concept Maps, and Topic Maps.

The above example by Peirce has an exact translation to the algebra:

$$\begin{aligned} \exists x \exists y \exists z & (\text{isaStagirite}(x) \wedge \text{teaches}(x,y) \wedge \text{isaMacedonian}(y) \wedge \\ & \text{conquersTheWorld}(y) \wedge \text{isaDiscipleOf}(x,z) \wedge \text{isanOpponentOf}(x,z) \\ & \wedge \text{isaPhilosopherAdmiredByChurchFathers}(z)) \end{aligned}$$

EGs without negation can represent the content stored in a database, but a more expressive logic is needed for DB queries and constraints.

Existential Graphs for Full FOL

A graph notation for logic with a minimum of primitives:

Existence: —


Negation: 

Relations: Cat- -On- -Under- -With- -Mat

A cat is on a mat: Cat—On—Mat

Something is under a mat: —Under—Mat

Some cat is not on a mat: Cat——Mat

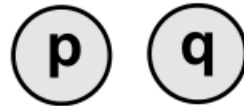
Some cat is on something that is not a mat: Cat—On—

Boolean Combinations

Areas nested inside an odd number of negations are shaded.

p q

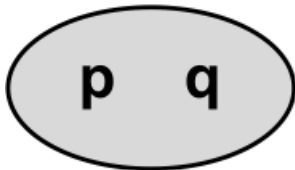
p and q



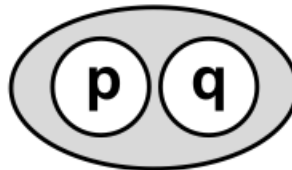
not p and not q



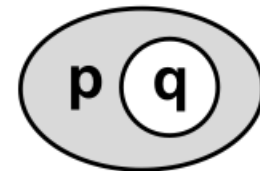
p and not q



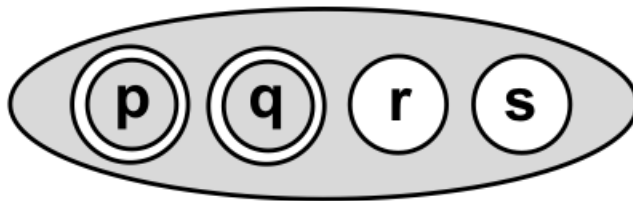
not (p and q)



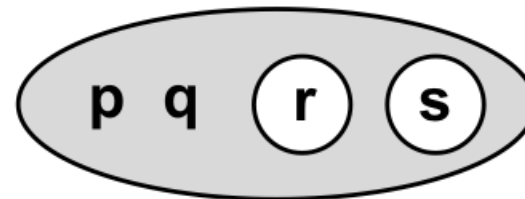
p or q



if p, then q



not p or not q or r or s



if p and q, then r or s

Representing Quantifiers

The scope of a quantifier is the outermost area that contains its line.

Cat—Black

Some cat is black.

Cat—Black

Some cat is not black.

Cat—Black

No cat is black.

An EG with two or more negations can be read in several ways:

It is false that some cat is not black.

If there is a cat, then it is black.

Every cat is black.

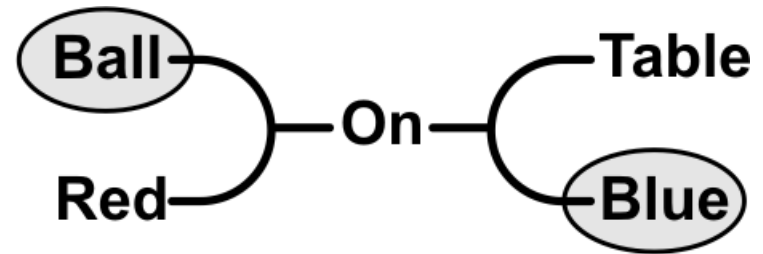
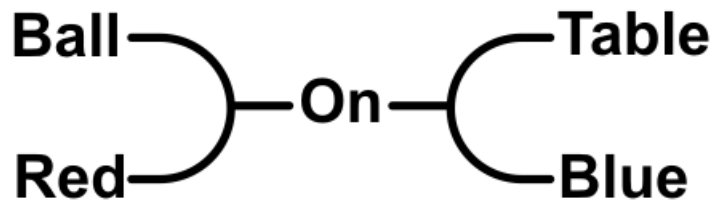
No cat is not black.

Cat—Black

These four sentences are synonymous (logically equivalent).

Translating EGs to and from English

Most existential graphs can be read in several equivalent ways.



Left graph:

A red ball is on a blue table.

Some ball that is red is on some table that is blue.

Right graph:

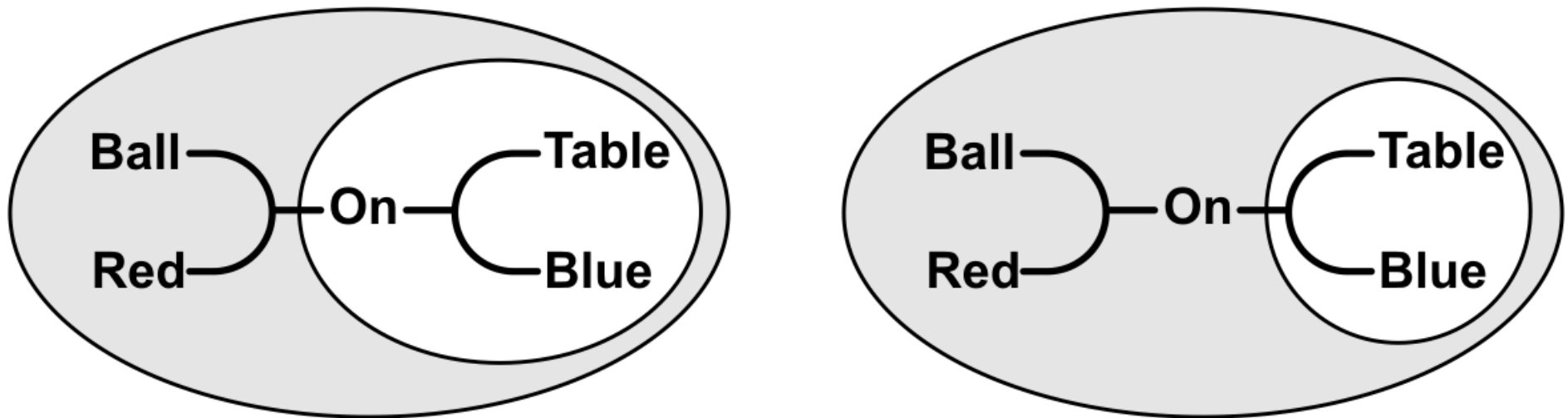
Something red that is not a ball is on a table that is not blue.

A red non-ball is on a non-blue table.

On some non-blue table, there is something red that is not a ball.

Scope of Quantifiers and Negations

Ovals define the scope for both quantifiers and negations.



Left graph:

If there is a red ball, then it is on a blue table.

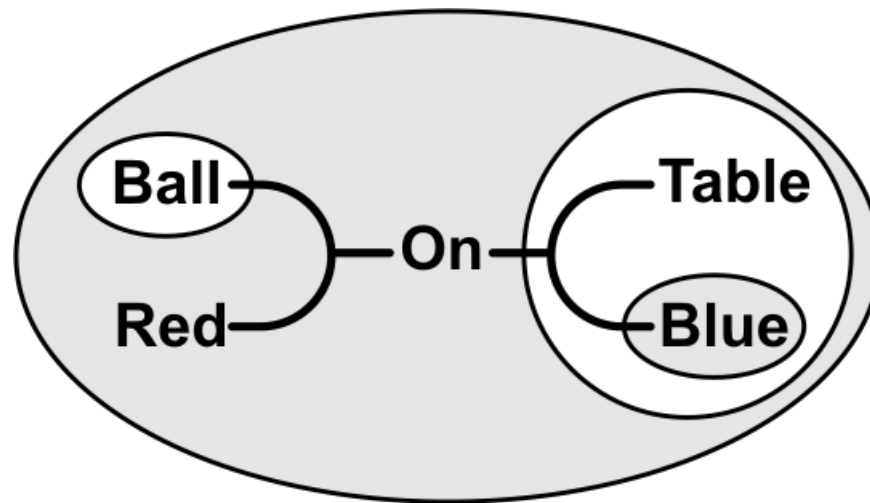
Every red ball is on some blue table.

Right graph:

If a red ball is on something x , then x is a blue table.

Multiple Nested Negations

Complex patterns of negations create more variations.

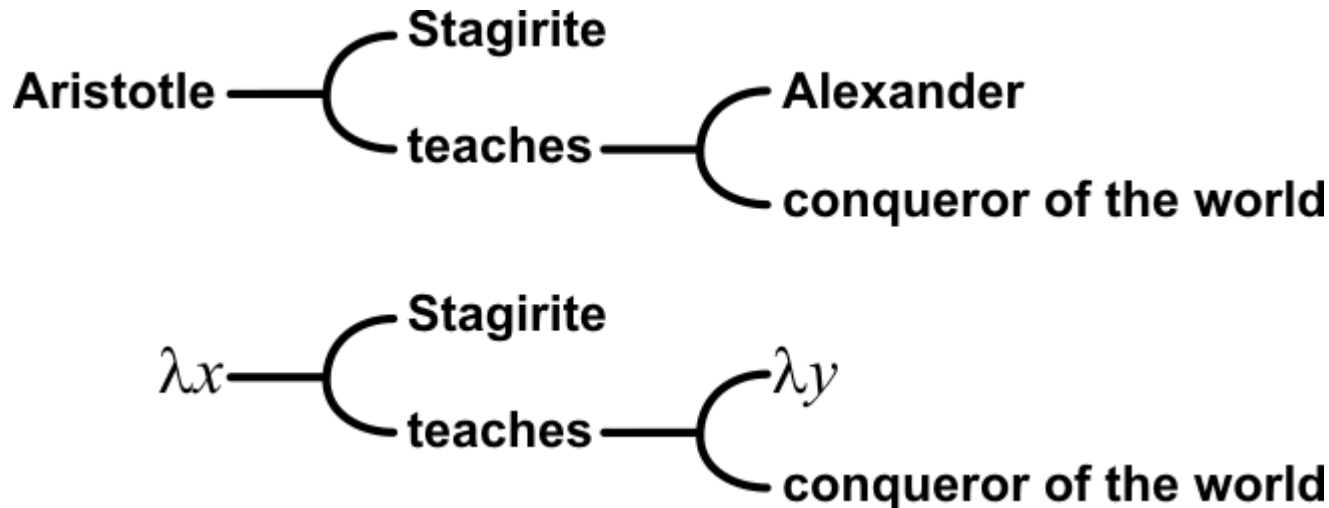


*If something red that is not a ball is on something y,
then y is a table that is not blue.*

*If a red thing x is on something y,
then either x is a ball, or y is a table that is not blue.*

*If a red thing x is on something that is not a non-blue table,
then x is ball.*

Lambda Abstraction



The top EG says *Aristotle is a Stagirite who teaches Alexander who conquers the world.*

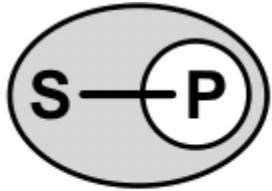
In the EG below it, the names Aristotle and Alexander are erased, and their places are marked with the Greek letter λ .

That EG represents a dyadic relation: *is a Stagirite who teaches* *who conquers the world.*

Peirce used an underscore to mark those empty places, but Alonzo Church marked them with λ .

Representing Syllogisms

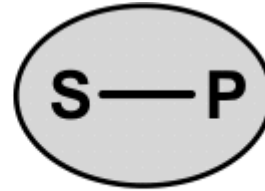
Existential graphs for the A, I, E, O sentences:



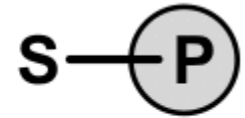
Every S is P.



Some S is P.



No S is P.



Some S is not P.

Each letter represents a monadic relation.

That relation could be a lambda abstraction from an EG in which one of the empty places is marked with the Greek letter λ .

For example, S could represent an EG for *red ball*, and P could represent an EG for *on a blue table*.

This subset of logic can represent the EL dialect of OWL, which supports terms that contain existential quantifiers.

Syntax

The syntax of any notation is the most obvious aspect.

Syntactic patterns are the easiest to process by computer.

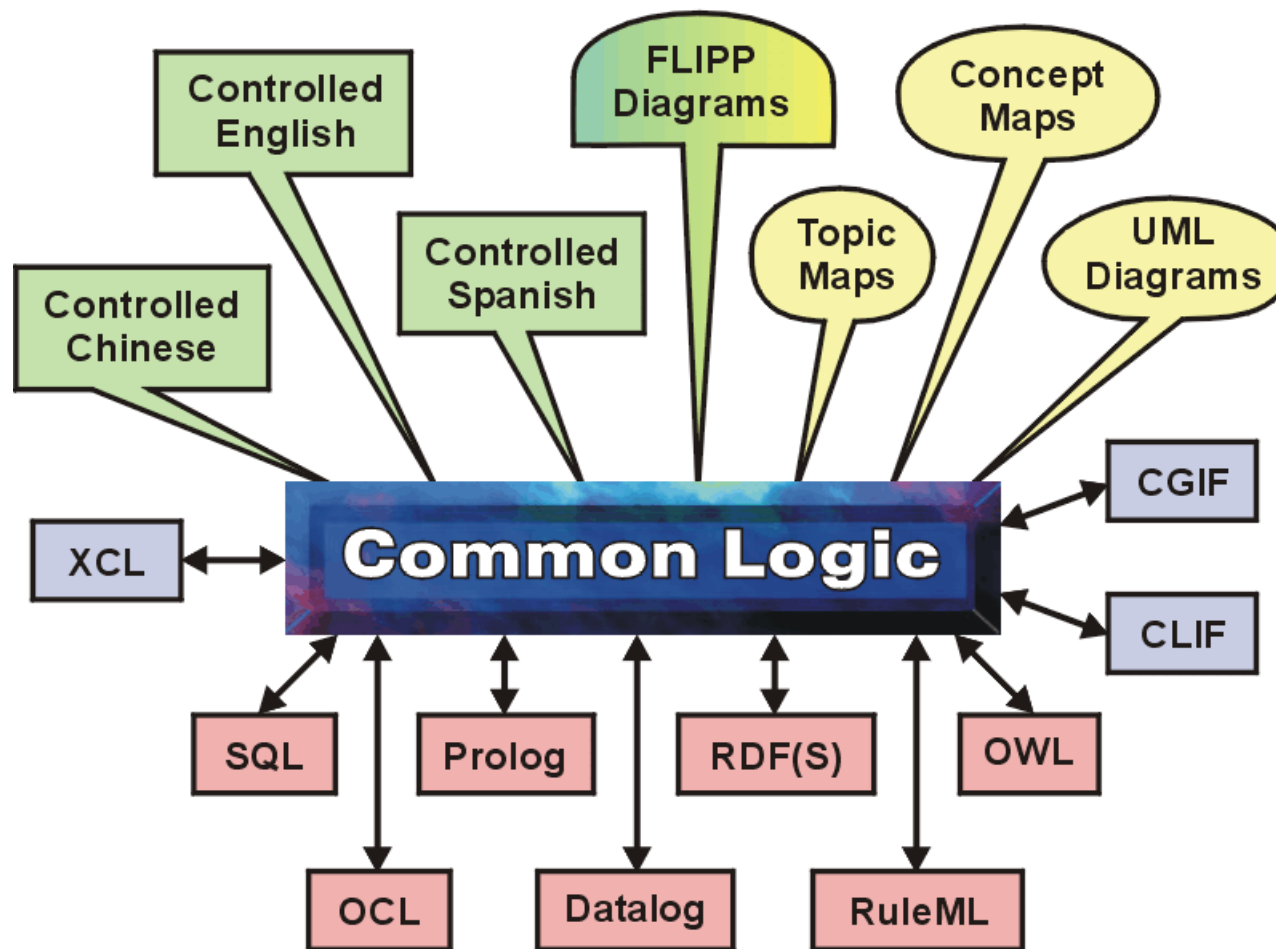
A good syntax is important for both human factors and computer efficiency.

But no syntax can be ideal for all purposes.

Compiler technology for translating from one syntax to another became mature in the 1960s.

Therefore, semantics can and should be specified in a way that is independent of any particular syntax.

Human Interfaces



Machine Interfaces

Common Logic

A highly expressive version of logic that is designed as a superset of many widely used notations.

Standardized by ISO/IEC 24707 as a formal semantics with an abstract syntax that can be mapped to many different notations.

Three concrete dialects specified in the standard:

- **CLIF: Common Logic Interchange Format.**
- **CGIF: Conceptual Graph Interchange Format.**
- **XCL: An XML-based notation.**

Any notation that has a formal mapping to the CL semantics may be called a CL dialect.

Translating EGs to Common Logic

Existential graphs can be translated to or from many different notations for logic.

The simplest mapping is to a subset of CGIF, which is designed to support graph notations:

- Each feature of an EG maps to exactly one feature of CGIF.**
- Like EGs, CGIF has no implicit ordering of nodes.**
- Like EGs, the conjunction 'and' is implicit.**

But as a linear notation, CGIF uses labels to represent the cross links of a graph.

Those labels have a direct mapping to variables in other notations, such as CLIF.

Conceptual Graph Interchange Format

A standard dialect of Common Logic:

Existence: — $[*x]$

Negation:  $\sim[\]$

Relations: (Cat ?x) (On ?x ?y) (Under ?x ?y) (Mat ?y)

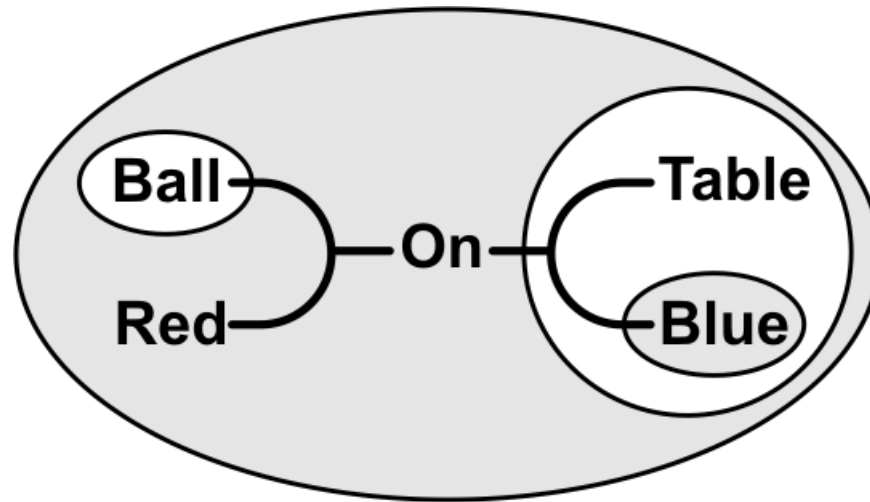
A cat is on a mat: $[*x] [*y] (\text{Cat } ?x) (\text{On } ?x ?y) (\text{Mat } ?y)$

Something is under a mat: $[*x] [*y] (\text{Under } ?x ?y) (\text{Mat } ?y)$

Some cat is not on a mat: $[*x] (\text{Cat } ?x) \sim[[*y] (\text{On } ?x ?y) (\text{Mat } ?y)]$

Some cat is on something that is not a mat:
 $[*x] [*y] (\text{Cat } ?x) (\text{On } ?x ?y) \sim[(\text{Mat } ?y)]$

Mapping an EG to CGIF



If something red that is not a ball is on something y, then y is a table that is not blue.

$\sim[[*x] [*y] (\text{Red } ?x) \sim[(\text{Ball } ?x)] (\text{On } ?x ?y) \sim[(\text{Table } ?y) \sim[(\text{Blue } ?y)]]]$

Note the one-to-one mapping of EG features to CGIF features:

- Four ovals map to four negations, represented as $\sim[]$.
- Two *ligatures* of connected lines map to $[*x]$ and $[*y]$.
- Five EG relation names map to five CGIF relation names.
- Six connections of lines to relations map to six occurrences of $?x$ or $?y$.

Common Logic Interchange Format

Another standard dialect of Common Logic:

Existence: — (exists (x))

Negation:  (not)

Relations: (Cat x) (On x y) (Under x y) (Mat y)

A cat is on a mat: (exists (x y) (and (Cat x) (On x y) (Mat y)))

Something is under a mat: (exists (x y) (and (Under x y) (Mat y)))

Some cat is not on a mat:

(exists (x) (and (Cat x) (not (exists (y) (and (On x y) (Mat y))))))

Some cat is on something that is not a mat:

(exists (x y) (and (Cat x) (On x y) (not (Mat y))))

Predicate Calculus

The widely used Peirce-Peano algebraic notation:

Existence: — $(\exists x)$

Negation:  $\sim()$

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

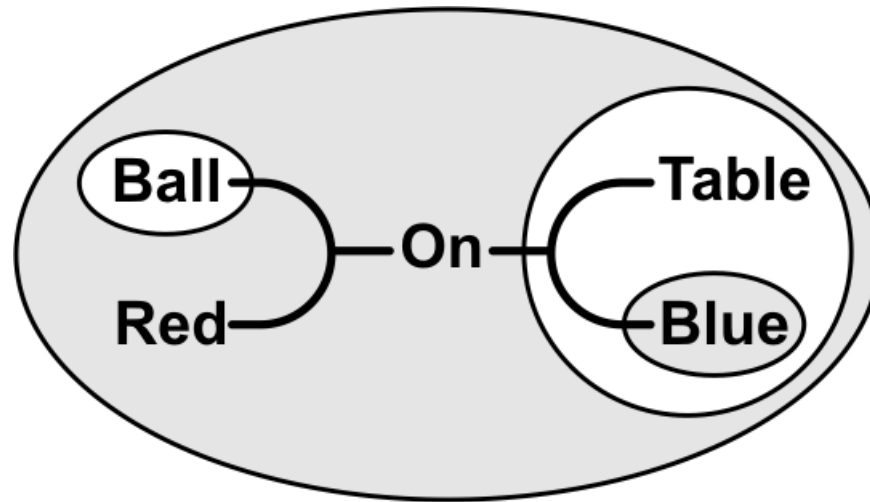
A cat is on a mat: $(\exists x)(\exists y)(\text{Cat}(x) \wedge \text{On}(x,y) \wedge \text{Mat}(y))$

Something is under a mat: $(\exists x)(\exists y)(\text{Under}(x,y) \wedge \text{Mat}(y))$

Some cat is not on a mat: $(\exists x)(\text{Cat}(x) \wedge \sim(\exists y)(\text{On}(x,y) \wedge \text{Mat}(y)))$

Some cat is on something that is not a mat:
 $(\exists x)(\exists y)(\text{Cat}(x) \wedge \text{On}(x,y) \wedge \sim \text{Mat}(y))$

Mapping to CLIF and Predicate Calculus



CGIF:

$\sim[[*x] [*y] (\text{Red } ?x) \sim[(\text{Ball } ?x)] (\text{On } ?x ?y) \sim[(\text{Table } ?y) \sim[(\text{Blue } ?y)]]]$

CLIF:

$(\text{exists } (x \ y) (\text{and } (\text{Red } x) (\text{not } (\text{Ball } x)) (\text{On } x \ y) (\text{not } (\text{and } (\text{Table } y) (\text{not } (\text{Blue } y)))))$

Predicate calculus:

$\sim(\exists x)(\exists y)(\text{Red}(x) \wedge \sim\text{Ball}(x) \wedge \text{On}(x,y) \wedge \sim(\text{Table}(y) \wedge \sim\text{Blue}(y)))$

Additional Operators

Existential graphs represent full first-order logic with just 3 operators.

Those operators are also sufficient for FOL in other notations.

But more symbols are convenient for commonly used patterns.

In predicate calculus, the universal quantifier ($\forall x$), which may be read *for every x* or *for all x*, can be defined by an equivalence:

$(\forall x)P(x)$ is defined as $\sim(\exists x)\sim P(x)$

In this definition, P is either a predicate name or an expression that could be used for a lambda abstraction.

CLIF uses the symbol 'forall' for the universal quantifier, and CGIF uses the symbol '@every'.

Predicate calculus, CLIF, and CGIF also introduce symbols to represent *or*, *if*, and *if and only if*.

These symbols are defined by Boolean combinations.

Type Constraints

In an untyped version of logic: any quantifier, such as $(\exists x)$ or $(\forall x)$, can range over anything in the universe.

In a typed logic, monadic relations restrict the domain of quantifiers: $(\exists x:\text{Cat})$ limits the variable x to things called cats.

Two policies for treating type mismatches:

- 1. Strong typing: A type mismatch causes a syntax error.**
- 2. Weak typing: A type mismatch causes an expression to be false, but it does not create a syntax error.**

Existential graphs and the Common Logic base are untyped.

The extended versions of CLIF and CGIF support weak typing by monadic relations that restrict the domain of quantifiers.

Any typed statement in Common Logic can be converted to a logically equivalent untyped statement.

Typed and Untyped Statements

Type constraints in predicate calculus, CLIF, and CGIF can be expressed by monadic relations that restrict the quantifier.

Predicate calculus:

Untyped: $(\exists x)(\exists y)(\text{Cat}(x) \wedge \text{Mat}(y) \wedge \text{On}(x,y))$

Typed: $(\exists x:\text{Cat})(\exists y:\text{Mat})\text{On}(x,y)$

Common Logic Interchange Format (CLIF):

Untyped: $(\text{exists } (x \ y) (\text{and } (\text{Cat } x) (\text{Mat } y) (\text{On } x \ y)))$

Typed: $(\text{exists } ((x \ \text{Cat}) (y \ \text{Mat})) (\text{On } x \ y))$

Conceptual Graph Interchange Format (CGIF):

Untyped: $[*x] [*y] (\text{Cat } ?x) (\text{Mat } ?y) (\text{On } ?x \ ?y)$

Typed: $[\text{Cat } *x] [\text{Mat } *x] (\text{On } ?x \ ?y)$

All six of these statements are logically equivalent: they are true if and only if a cat is on a mat.

Common Logic Controlled English

CLCE is a formally defined language that uses English syntax.

Every CLCE sentence can be read as if it were English.

But it can be translated automatically to and from CGIF or CLIF.

In fact, every English sentence in these slides that is translated to or from an existential graph uses only the CLCE subset.

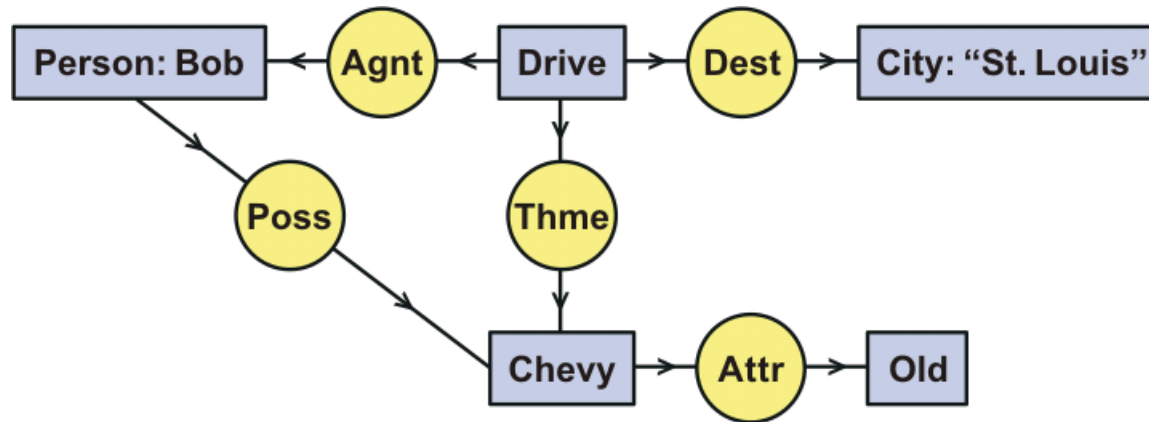
To avoid ambiguous pronouns, CLCE permits temporary names, which look like variables in predicate calculus:

*If a red thing x is on something y ,
then either x is a ball or y is a table that is not blue.*

In CLCE, the letters x , y , and z by themselves or followed by one or more digits are treated as unambiguous pronouns.

CLCE: *Bob drives his old Chevy to St. Louis.*

Conceptual graph display form:



Conceptual Graph Interchange Format (CGIF):

[Drive *x] [Person Bob] [City "St. Louis"] [Chevy *y] [Old *z]
(Agnt ?x Bob) (Dest ?x "St. Louis")
(Thme ?x ?y) (Poss Bob ?y) (Attr ?y ?z)

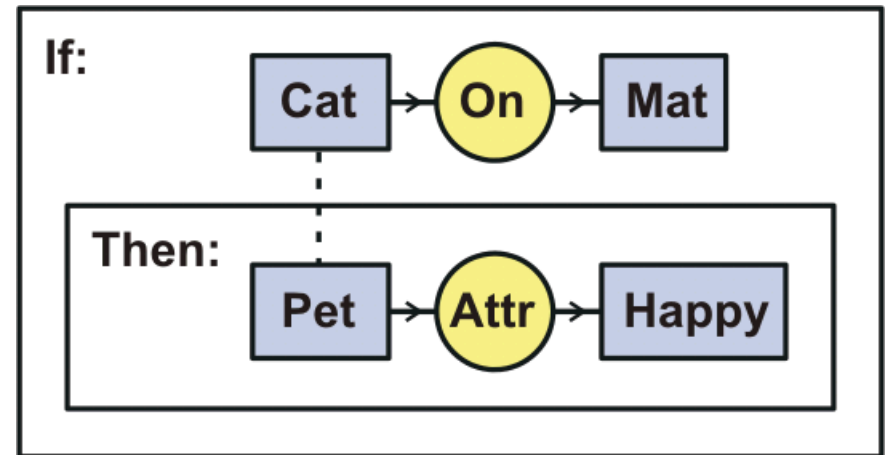
Common Logic Interchange Format (CLIF):

(exists ((x Drive) (y Chevy) (z Old))
 (and (Person Bob) (City "St. Louis") (Agnt x Bob)
 (Dest x "St. Louis") (Thme x y) (Poss Bob y) (Attr y z)))

If-Then Statements

CLCE: *If a cat is on a mat, then the cat is a happy pet.*

Conceptual graph display form:



CGIF:

[If: [Cat: *x] [Mat: *y] (On ?x ?y)
[Then: [Pet: ?x] [Happy: *z] (Attr ?x ?z)]]

CLIF:

(not (exists ((x Cat) (y Mat)) (and (On x y)
(not (exists z) (and (Pet x) (Happy z) (Attr x z))))))

Note: In CGs, nested if-then boxes are identical to Peirce's ovals.

A Logically Equivalent Pattern

CLCE: *For every cat x and every mat y,
if x is on y, then x is a happy pet.*

CGIF:

[Cat: @every *x] [Mat: @every *y]

[If: (On ?x ?y)

[Then: [Pet: ?x] [Happy: *z] (Attr ?x ?z)]]

CLIF:

(forall ((x Cat) (y Mat))

(if (On x y)

(and (Pet x) (exists ((z Happy)) (Attr x z))))))

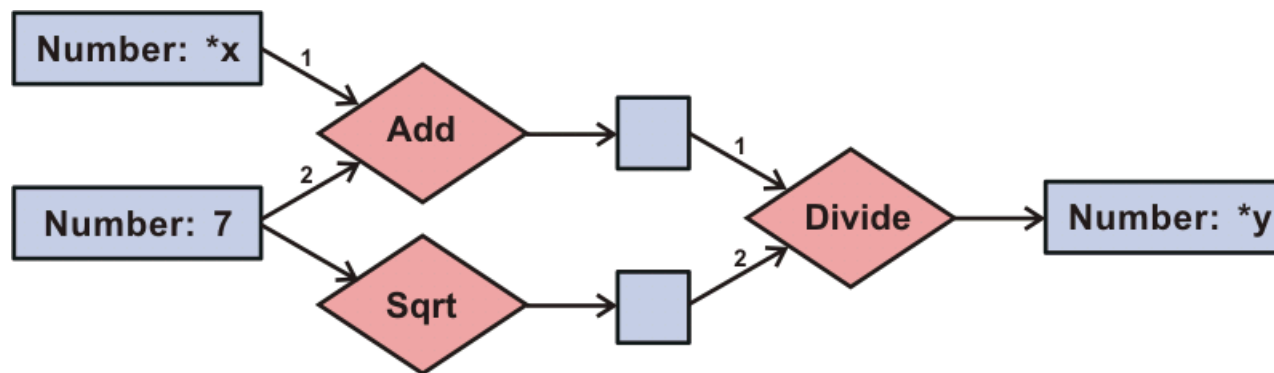
Most dialects of logic and natural languages permit many different ways of expressing logically equivalent statements.

For common patterns such as these, a proof of equivalence takes linear time. More complex examples may take polynomial time.

Representing Functions

CLCE: *For a number x , a number y is $((x+7) / \text{sqrt}(7))$.*

Conceptual graph display form:



CGIF:

[Number: *x] [Number: *y]
(Add ?x 7 | *u) (Sqrt 7 | *v) (Divide ?u ?v | ?y)

CLIF:

(exists ((x Number) (y Number))
 (= y (Divide (Add x 7) (Sqrt 7))))

Quantifying Over Relations

Common Logic has a first-order model theory, but it allows quantified variables to refer to functions and relations.

English: “Bob and Sue are related.”

CLCE: *There is a familial relation between Bob and Sue.*

CGIF: [Relation: *r] (Familial ?r) (?r Bob Sue)

CLIF: (exists ((r Relation)) (and (Familial r) (r Bob Sue)))

As another example, x is equal to y if and only if every relation that is true of one is also true of the other:

```
(forall (x y)
  (iff (= x y)
    (forall (R) (iff (R x) (R y)))) ))
```

5. Combining Logic and Ontology

Aristotle's syllogisms use logic to analyze the ontology.

Professionals in every field develop notations that mix logic with an ontology specialized for their subject matter.

Areas of active research:

- **Parts, shapes, spaces, systems, and life.**
- **Time, events, situations, agents, processes, and causality.**
- **Patterns of signs: Syntax, semantics, and pragmatics.**
- **Intentionality: Attention and purpose in every aspect of life.**

Ways of combining logic and ontology:

- **Simplest method: Use some notation for logic to represent the ontology with some choice of functions and relations.**
- **More complex method: Invent new syntax to represent the most important functions and relations in the ontology.**

Patterns of Ontology

Everything that anybody knows, does, sees, or thinks:

- **Parts:** Physical, abstract, discrete, continuous.
- **Shapes:** Natural, artificial, regular, irregular, static, dynamic.
- **Spaces:** 3-D space, 4-D space-time, any theory of physics.
- **Systems:** Dynamic patterns of interoperating parts.
- **Life:** Every kind of organism, behavior, society, ecology.

Notations that mix ontology with some version of logic:

- **Temporal logic** includes an ontology of time.
- **Higher-order logic** assumes a hierarchy of infinite sets.
- **Procedural languages** represent sequences of events.
- **Every type of UML diagram** uses a different combination.

Temporal Patterns

Ways of analyzing patterns that occur in space and time:

- **Time:** Discrete, continuous, branching, 4-D space-time.
- **Events:** Meaningful occurrences in space and time.
- **Situations:** Chunks of space-time that contain events.
- **Processes:** Causally connected chains of events.

The word *meaningful* raises a philosophical puzzle.

- **What is a meaningful event or situation?**
- **Scientists observe patterns in nature, and they form theories about the laws that generate the patterns.**
- **But the laws say nothing about meaning or value.**
- **Why should one pattern be more meaningful than another?**

Intentionality

Meaning in the universe arose with the first living things:

- **Philosopher Franz Brentano:** Intentionality is “*the directedness of thought toward some object, real or imagined.*”
- **Biologist Lynn Margulis:** “*The growth, reproduction, and communication of these moving, alliance-forming bacteria become isomorphic with our thought, with our happiness, our sensitivities and stimulations.*”
- **A bacterium swimming upstream in a glucose gradient marks the beginning of goal-directed intentionality.**

Without life, there is no meaning in the universe.

Meaning, intention, purpose, and value originate with life.

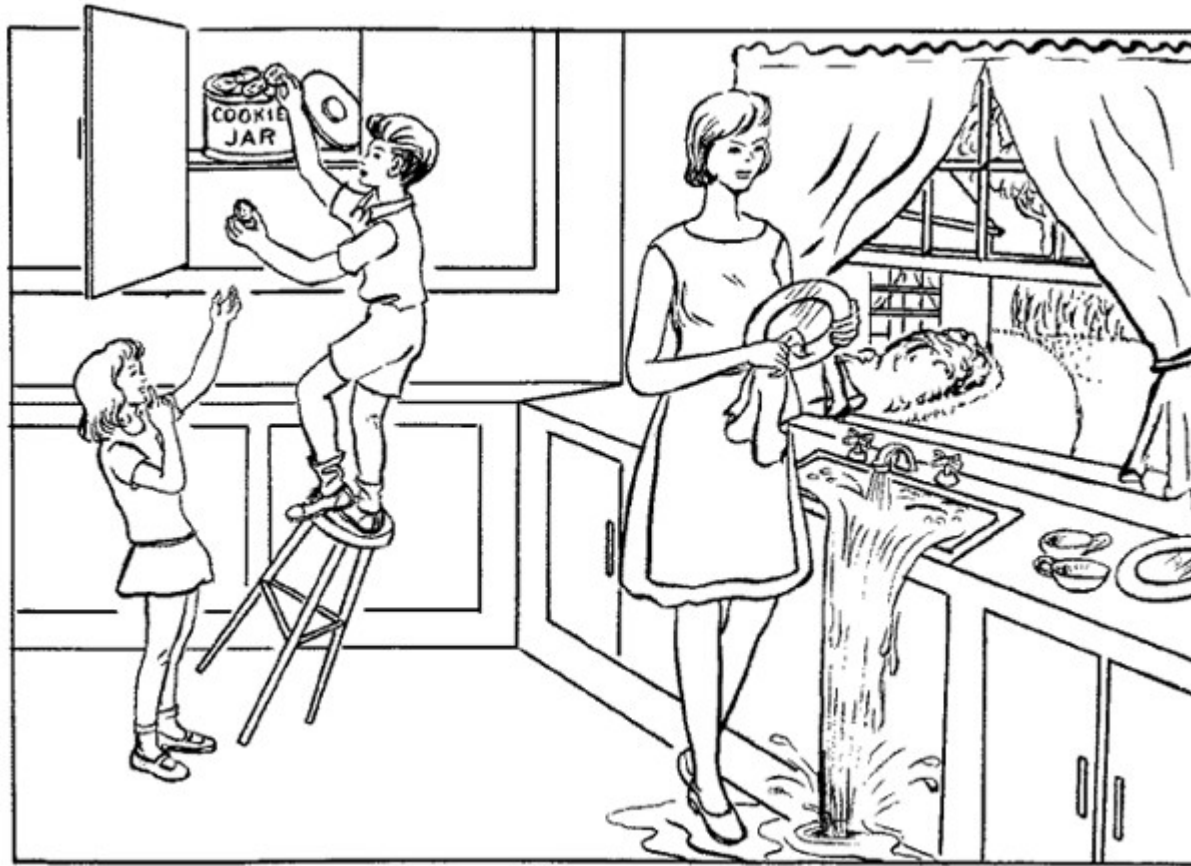
Aristotle the biologist used the term *telos* or *final aitia*.

What is a Situation?

Definition: A situation is a region of space-time that bounds the range of perception, action, interaction, and communication of one or more agents.

- The boundary of a situation is determined by the range of perception, action, and communication by the agents in it.
- A situation without agents is possible, but meaningless.
- Microscopes, telescopes, and TV use enhanced methods of perception and action to change the boundary of a situation.
- Psychologists and sociologists study human situations.
- Logicians and philosophers formulate theoretical models of agents interacting and communicating in situations.
- Computer scientists develop methods for simulating and reasoning about the models.

Example of a Situation



This is a test picture used to diagnose patients with aphasia. A patient's description of the situation can show the effects of lesions caused by wound, stroke, tumor, or infection.

Patterns of Situations

Philosophers, linguists, logicians, and computer scientists emphasize different aspects of situations.

But there are important commonalities:

- **A situation is an actual, hypothetical, or fictional region.**
- **Somebody decided that the region is significant.**
- **Some version of logic and ontology can be used to describe it.**
- **Linguistic theories relate sentences to situations and speakers.**

Questions:

- **How do we decide what situations are important?**
- **How can we describe them effectively?**
- **How can we reason about them?**

Meaningful Aspects of the Situation

Space-time region shown in the diagram:

- The kitchen of a private home.

Agents:

- Girl, boy, woman.

Goals of the agents:

- Girl, boy: get cookies.
- Woman: wash dishes; maintain discipline.

Actions:

- Wiping, spilling, reaching, holding, grasping, tipping, falling.

Question:

- How can we represent this situation in logic?

Representing Situations

Common Logic in any dialect – CLIF, CGIF, or CLCE – can represent a situation and the things and events in it.

But an extension to Common Logic is necessary to express theories about propositions and situations.

The critical extension is the ability to make metalevel statements that relate propositions and situations.

That extension is necessary to relate statements in logic to the propositions they express.

It introduces an ontology about propositions, situations, and the language for relating them.

IKRIS Project

DoD-sponsored project: Design an Interoperable Knowledge Language (IKL) as an extension to Common Logic.

Goals:

- **Enable interoperability among advanced reasoning systems.**
- **Test that capability on highly expressive notations for logic.**

Show that semantics is preserved in round-trip mapping tests:

- **Cycorp: Cyc Language \rightarrow IKL \rightarrow CycL**
- **RPI / Booz-Allen: Multi-Sorted Logic \rightarrow IKL \rightarrow MSL**
- **Stanford / IBM / Battelle: KIF \rightarrow IKL \rightarrow KIF**
- **KIF \rightarrow IKL \rightarrow CycL \rightarrow IKL \rightarrow MSL \rightarrow IKL \rightarrow KIF**

Conclusion: “IKRIS protocols and translation technologies function as planned for the sample problems addressed.”

The IKL Extension to Common Logic

Common Logic is a superset of most logics used in semantic systems, but some require even more expressive logics.

Only one new operator is needed: a metalanguage enclosure, which uses the keyword 'that' to mark the enclosed statement.

- **The enclosed statement denotes a proposition.**
- **That proposition could be a conjunction of many statements.**
- **It can be given a name, and other propositions can refer to it.**
- **In effect, IKL can be used as a metalanguage for talking about and relating packages of IKL statements nested to any depth.**

CL with the IKL extensions can represent a wide range of logics for modality, defaults, probability, uncertainty, and fuzziness.

Using CLCE to Express IKL

The operator 'that' of IKL can be used in CLCE:

Tom believes that Mary knows that $(2 + 2 = 4)$.

In CLIF notation for IKL:

(Believes Tom (that (Knows Mary (that (= (+ 2 2) 4))))))

In CGIF notation for IKL:

(Believes Tom [Proposition (Knows Mary [Proposition (+ 2 2 | 4)])])

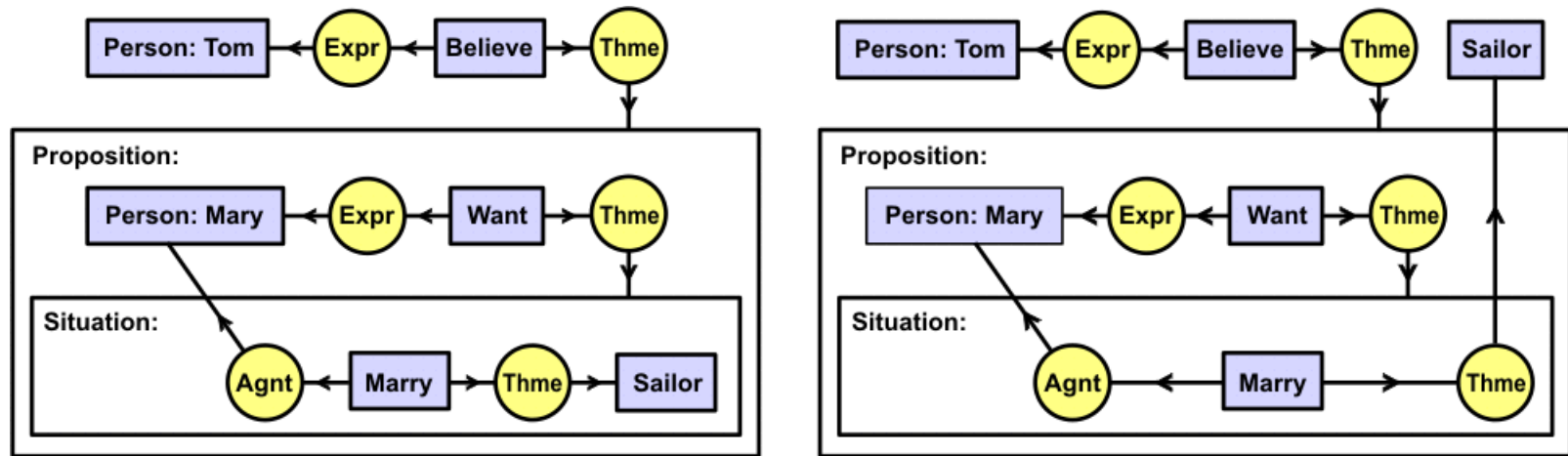
The operator 'that' is a powerful metalevel extension.

It enables IKL to specify languages, define their semantics, and specify transformations from one language to another.

Writing complex statements in CLCE requires training in logic.

But anybody who can read English can read CLCE.

Propositions and Situations



The two CGs above show two different interpretations of the English sentence *Tom believes that Mary wants to marry a sailor*:

- *There exists a sailor, and Tom believes a proposition that Mary wants a situation in which she marries the sailor.*
- *Tom believes a proposition that Mary wants a situation in which there exists a sailor whom she marries.*

IKL semantics permits the quantifier for “*a sailor*” to include the enclosed statements within its scope.

Representing IKL in CLIF and CGIF

Following is the CGIF representation for the CG on the left of the previous slide:

```
[Person: Tom] [Believe: *x1] (Expr ?x1 Tom) (Thme ?x1 [Proposition:  
  [Person: Mary] [Want: *x2] (Expr ?x2 Mary) (Thme ?x2 [Situation:  
    [Marry: *x3] [Sailor: *x4] (Agnt ?x3 Mary) (Thme ?x3 ?x4)]))])
```

In CLIF notation, the operator 'that' applied to a CL or IKL sentence denotes the proposition stated by the sentence:

```
(exists ((x1 Believe)) (and (Person Tom) (Expr x1 Tom) (Thme x1 (that  
  (exists ((x2 Want) (s Situation)) (and (Person Mary) (Expr x2 Mary)  
    (Thme x2 s) (Dscr s (that  
      (exists ((x3 Marry) (x4 Sailor)) (and (Agnt x3 Mary) (Thme x3 x4)  
        )))))))))))
```

To represent the CG on the right of the previous slide, move the concept node [Sailor: *x4] in front of the concept [Person: Tom] for CGIF notation. For CLIF, move (x4 Sailor) in front of (x1 Believe).

Situation Semantics

In situation theory, the unit of information is called an *infor* σ , which is entailed by some *situation* s : $s \models \sigma$

The meaning of a language expression φ is a triad $\varphi(d, c, a)$ that relates a *discourse situation* d , a *speaker connection function* c , and a *described situation* e . It is usually written $d, c \parallel \varphi \parallel e$.

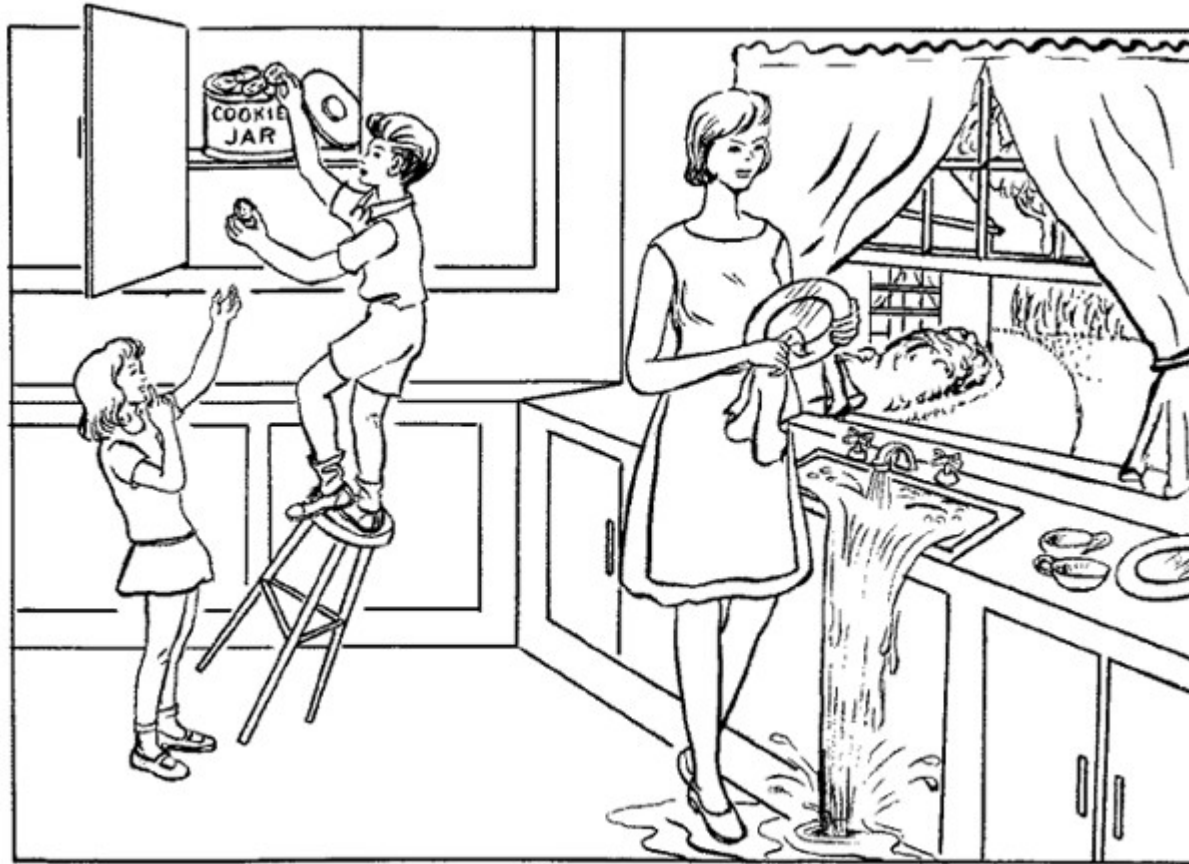
Those features can be expressed in Common Logic:

- A single relation with all its arguments corresponds to an infor.
- A compound infor is a CL expression that combines the relations.
- The theory, notation, and terminology by Devlin (1991) can be directly mapped to any dialect of Common Logic.

The IKL extension to CL provides a metalanguage for talking about situations, agents, and the intentions of the agents.

Controlled NLs can serve as a readable notation for CL and IKL.

Describing a Situation in CLCE



{Situation: A woman, a girl, and a boy are in a kitchen of a house. The woman wipes a plate with a cloth. Water spills on the floor of the kitchen. The girl reaches for a cookie. The boy holds a cookie in his left hand. The boy grasps a cookie with his right hand. The boy stands on a stool. The stool tips over. The boy falls down.}

6. Patterns of Patterns of Patterns

An ontology is a pattern of signs for describing and classifying whatever exists or can exist in some domain of interest.

A logic is a system of signs for relating and reasoning about the patterns of signs in an ontology.

Semantics evaluates truth by relating the patterns of signs of logic and ontology to the patterns of signs in the world.

Semiotics, the study of signs, is the foundation for language:

- Every living cell responds to signs and communicates by generating signs.**
- Larger organisms are colonies of cells that communicate by signs.**
- Neurons are cells that facilitate communication in an organism.**
- A social system is a community of organisms of some species.**
- Every language, natural or artificial, is a system of signs that facilitates communication in one or more social systems.**

Is Logic the Foundation for Language?

Richard Montague (1970) treated NLs as a version of logic:

“I reject the contention that an important theoretical difference exists between formal and natural languages.”

Hans Kamp (2001) followed Montague with qualifications:

“The basic concepts of linguistics — and especially those of semantics — have to be thought through anew... Many more distinctions have to be drawn than are dreamt of in current semantic theory.”

Barbara Partee (2005) added further qualifications:

“The present formalizations of model-theoretic semantics are undoubtedly still rather primitive compared to what is needed to capture many important semantic properties of natural languages... There are other approaches to semantics that are concerned with other aspects of natural language, perhaps even cognitively deeper in some sense, but which we presently lack the tools to adequately formalize.”

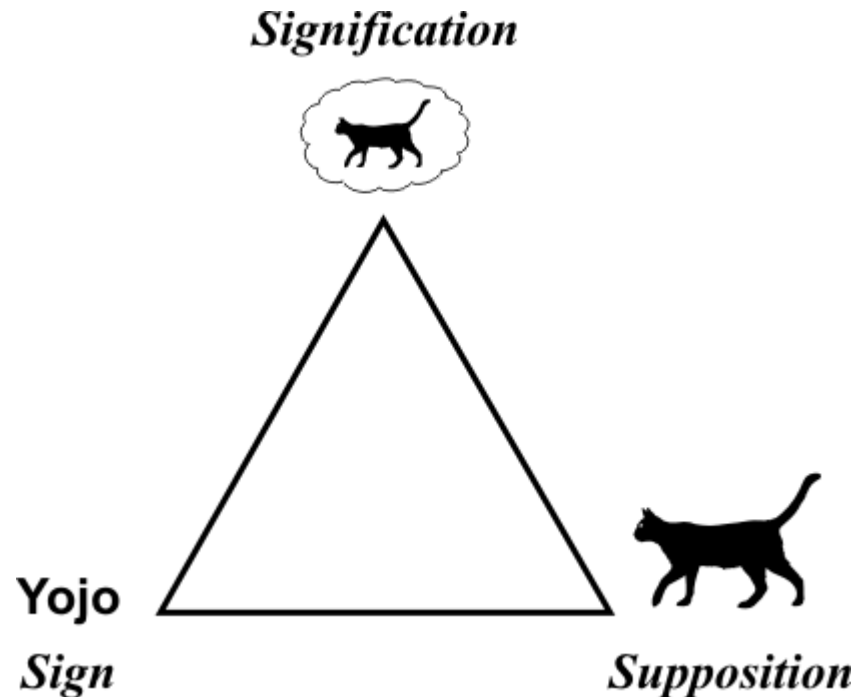
Aristotle's Foundation

The most influential and most widely debated paragraph about signs and language ever written:

First we must determine what are noun (*onoma*) and verb (*rhêma*); and after that, what are negation (*apophasis*), assertion (*kataphasis*), proposition (*apophansis*), and sentence (*logos*). Those in speech (*phonê*) are symbols (*symbola*) of affections (*pathêmata*) in the psyche, and those written (*graphomena*) are symbols of those in speech. As letters (*grammata*), so are speech sounds not the same for everyone. But they are signs (*sêmeia*) primarily of the affections in the psyche, which are the same for everyone, and so are the objects (*pragmata*) of which they are likenesses (*homoîomata*). On these matters we speak in the treatise on the psyche, for it is a different subject. *

* Aristotle, *On Interpretation*, 16a1.

Scholastic Meaning Triangle



Ogden and Richards (1923) drew meaning triangles, but Aristotle and the Scholastics developed the theory.

The Scholastics used the Latin *signum* for the sign, *significatio* for the affection in the psyche, and *suppositio* for the object.

For the signification, this diagram follows Aristotle by showing a cloud that contains a likeness (*homoîôma*) of the object.

Different Labels for the Triangle

A Scholastic alternative:

- *Signum, conceptus, objectus* (sign, concept, object).

Charles Sanders Peirce:

- Sign, interpretant, object.

Gottlob Frege:

- *Zeichen, Sinn, Bedeutung* (sign, sense, reference).

Edmund Husserl:

- *Zeichen, Bedeutung, Gegenstand* (sign, meaning, object).

Ferdinand de Saussure:

- Signifier, signified. (He ignored the vertex on the lower right.)

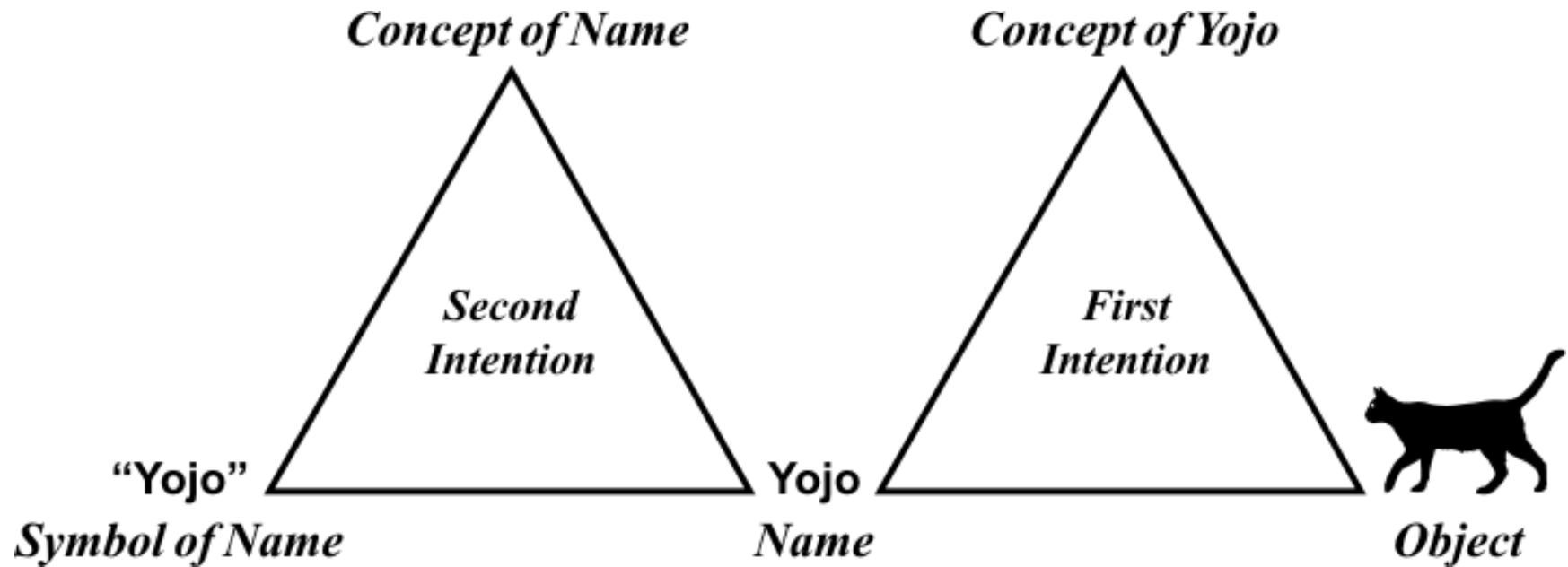
Alfred Tarski:

- Sign, object. (He ignored the top of the triangle.)

The dyads by Saussure and Tarski omit important distinctions:

- Saussure did not distinguish the meaning from the object.
- Tarski did not distinguish different meanings with the same object.

Metalinguage



Aristotle: “Written words are symbols of those in speech.”

The Scholastics generalized that principle:

- **First intentions:** Signs that refer to things in the world.
- **Second intentions:** Signs that refer to other signs.

In the 1930s, Alfred Tarski introduced the terms *object language* for first intentions and *metalinguage* for second intentions.

Universal Language Schemes

With the printing press and the new nation states, a flood of books in modern languages was rapidly displacing Latin.

In the 17th century, scientists, philosophers, merchants, bankers, and diplomats felt the need for a new universal language.

Francis Bacon claimed that “real characters” similar to Chinese characters could be used for mutually unintelligible languages.

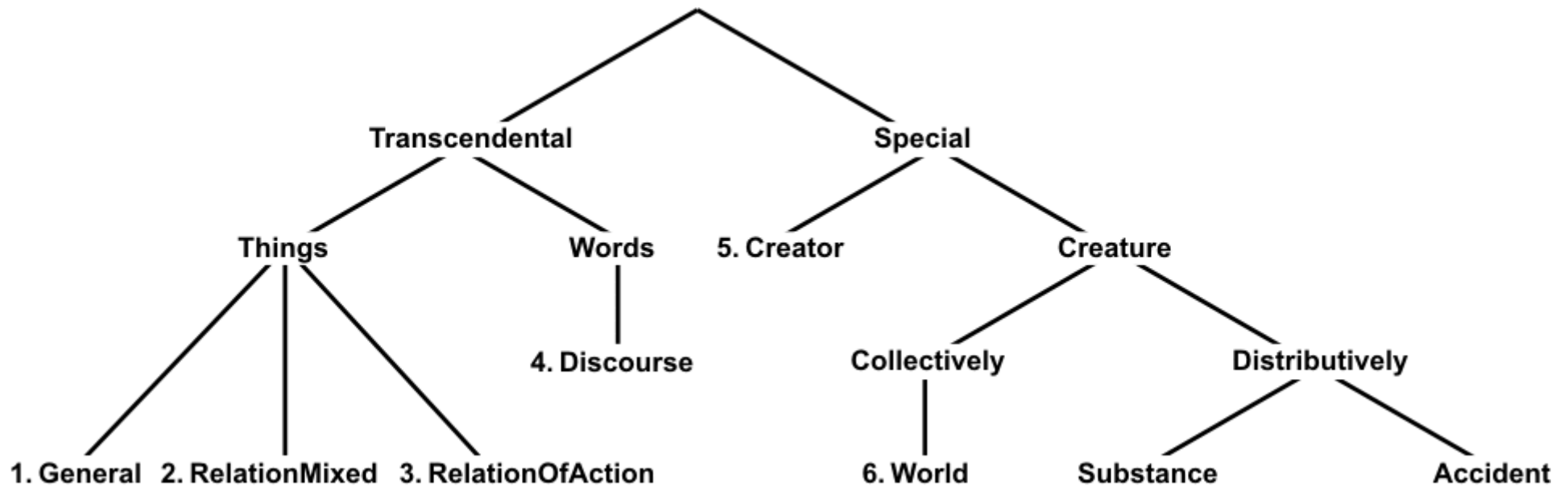
Descartes, Mersenne, Pascal, Newton, and Leibniz proposed mathematical principles as the basis for a universal language.

The largest and most impressive system was the *Real Character and Philosophical Language* by John Wilkins.

Wilkins was secretary of the British Royal Society. Several other members collaborated on the project.

For further discussion, see Knowlson (1975), Eco (1995), and Okrent (2009).

Wilkins' Upper-Level Ontology



In a 600-page book, Wilkins (1668) devoted 270 pages to tables that define 40 genera subdivided in 2,030 species.

The categories labeled 1 through 6 are the first of his 40 genera. The other 34 genera are subtypes of Substance or Accident.

Inheritance: Each species is defined by the conjunction of all the differentiae along the path from one of the 40 genera.

Summary of Wilkins' System

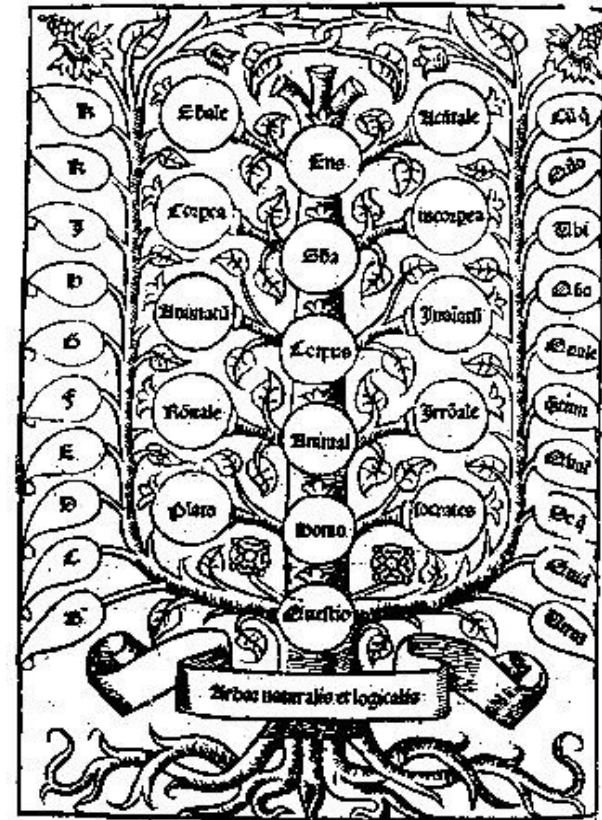
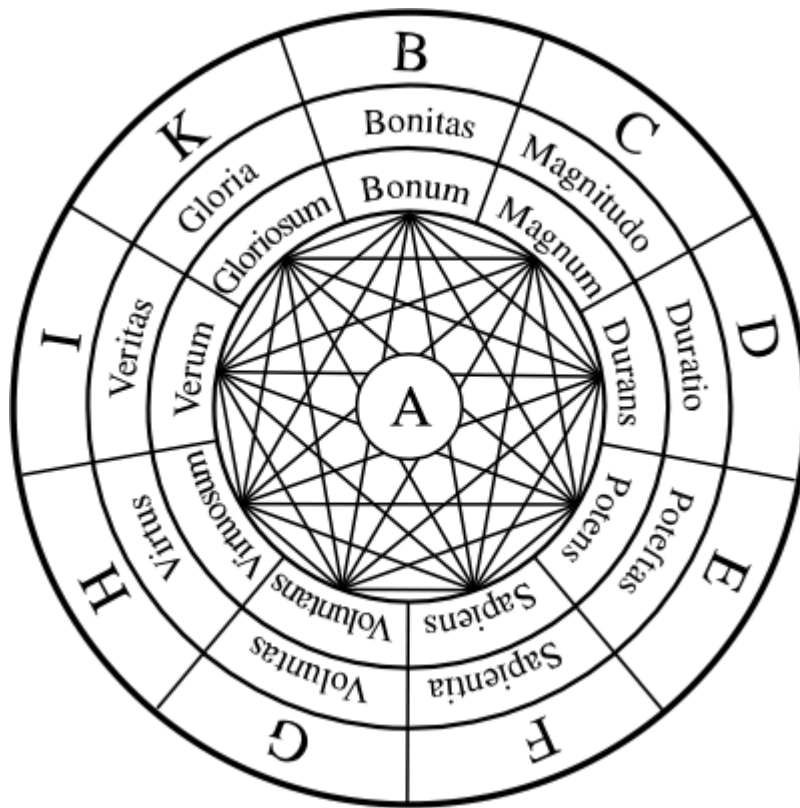
An impressive combination of upper-level ontology, metalevel ontology, mid-level ontology, thesaurus, and notation:

- A failure as a replacement for Latin, but an inspiration for Leibniz, Kant, Roget, and many others.**
- The division of Transcendental vs. Special corresponds roughly to the distinction between signs and their referents.**
- The division of Collectively vs. Distributively is an important distinction that many ontologies ignore.**
- But 2,030 categories at the endpoints of the tree are inadequate for a general-purpose language.**

Other members of the Royal Society added about 15,000 English words as approximate synonyms of those 2,030 categories.

Unfortunately, the system contained many ad hoc features that were ridiculed by Jonathan Swift and Jorge Luis Borges.

Circles and Diagrams by Ramon Lull



Lull was a Catalan poet, philosopher, and missionary.

He developed a system of rotating circles for combining attributes and relating them to Aristotle's categories.

It was an early device for mechanical reasoning.

Gottfried Wilhelm Leibniz

Leibniz was a mathematician, philosopher, and diplomat who was impressed by Wilkins' system:

*“The art of ranking things in genera and species is of no small importance and very much assists our judgment as well as our memory.... And those who have laid out all sorts of notions under certain headings or categories have done something very useful.” **

Leibniz was also inspired by Lull's combinatorial method.

For his *Universal Characteristic*, he assigned a prime number to each differentia and products of primes to each category.

Category A is more general than category B iff the integer for A divides the integer for B.

The result is a system of multiple inheritance called a *lattice*.

Encoding the Tree of Porphyry

Assign 1 to Substance, the supreme genus.

Assign prime numbers to the differentiae:

**material 2; immaterial 3; animate 5; inanimate 7; sensitive 11;
insensitive 13; rational 17; irrational 19.**

**The number for each category is the product of the
number for its genus times the number for its differentia:**

Body is $1 \times 2 = 2$. Spirit is $1 \times 3 = 3$.

LivingThing is $2 \times 5 = 10$. Mineral is $2 \times 7 = 14$.

Animal is $10 \times 7 = 70$. Plant is $10 \times 13 = 130$.

Human is $70 \times 17 = 1190$. Beast is $70 \times 19 = 1330$.

**Category A is more general than category B iff the number
for A divides the number for B.**

Calculating Machine by Leibniz



Leibniz invented the first calculating machine that could do multiplication and division.

He used it for mechanical reasoning about patterns of relations in the numeric encoding of his categories.

His method of using prime numbers inspired Kurt Gödel, who used prime numbers to encode patterns of logic.

Lattice-Based Methods

Many classification schemes are organized as trees, which limit inheritance to just one parent for any node beneath the top.

To support multiple inheritance, S. R. Ranganathan developed a system of faceted classification for library catalogs:

- Each facet represents a monadic relation.**
- Each category is defined by a conjunction of facets.**

Formal Concept Analysis (FCA) generates a minimal lattice for any concepts or categories defined by such a conjunction:

- Input to the FCA tools is a list of concepts and definitions.**
- Those definitions could be the list of facets for each concept.**
- The output is a minimal lattice that shows all inheritance paths.**
- FCA is often used to check OWL ontologies for consistency.**

**For FCA tools and techniques, see <http://www.upriss.org.uk/fca/fca.html>
For faceted classification, see <http://www.iskoug.org/kokonov2007.htm>**

Immanuel Kant

Quantity	Quality	Relation	Modality
Unity	Reality	Inherence	Possibility
Plurality	Negation	Causality	Existence
Totality	Limitation	Community	Necessity

Kant defined 12 categories, organized in four groups of three. He also claimed that his categories could replace the top level in an ontology such as Aristotle's or Wilkins':

*“If one has the original and primitive concepts, it is easy to add the derivative and subsidiary, and thus give a complete picture of the family tree of the pure understanding.... It can easily be carried out with the aid of the ontological manuals.” **

But nobody ever carried out that “easy” task.

* Kant (1787) *Critique of Pure Reason*, (A:82, B:108).

Charles Sanders Peirce

Peirce was a pioneer in modern logic, and he was familiar with logic, ontology, and semiotics from Aristotle to the 19th century.

He also studied Kant and analyzed the patterns of triads in Kant's table of categories.

He discovered metalevel patterns underlying various triads:

- Firstness: Quality may be expressed by a monadic predicate.**
- Secondness: Reaction may be expressed by a dyadic relation.**
- Thirdness: Mediation requires a triadic relation to bring a First and a Second into a dyadic relationship.**

The basic triad: Anything observable may be a Mark (1), which may be interpreted as a Token (2) of some Type (3).

The most commonly cited triad: Icon, Index, Symbol.

He developed the triads into a rich combinatorial system.

Peirce's Triple Trichotomy

1. Quality

2. Indexicality

3. Mediation

1. Material	Mark <i>A quality which is a sign.</i>	Token <i>An actual existent thing or event which is a sign.</i>	Type <i>A principle, habit, or law which is a sign.</i>
2. Relational	Icon <i>Refers by virtue of some similarity to object.</i>	Index <i>Refers by virtue of being affected by object.</i>	Symbol <i>Refers by virtue of some law or association.</i>
3. Formal	Predicate <i>A sign of qualitative possibility.</i>	Assertion <i>A sign of actual existence.</i>	Argument <i>A sign of law.</i>

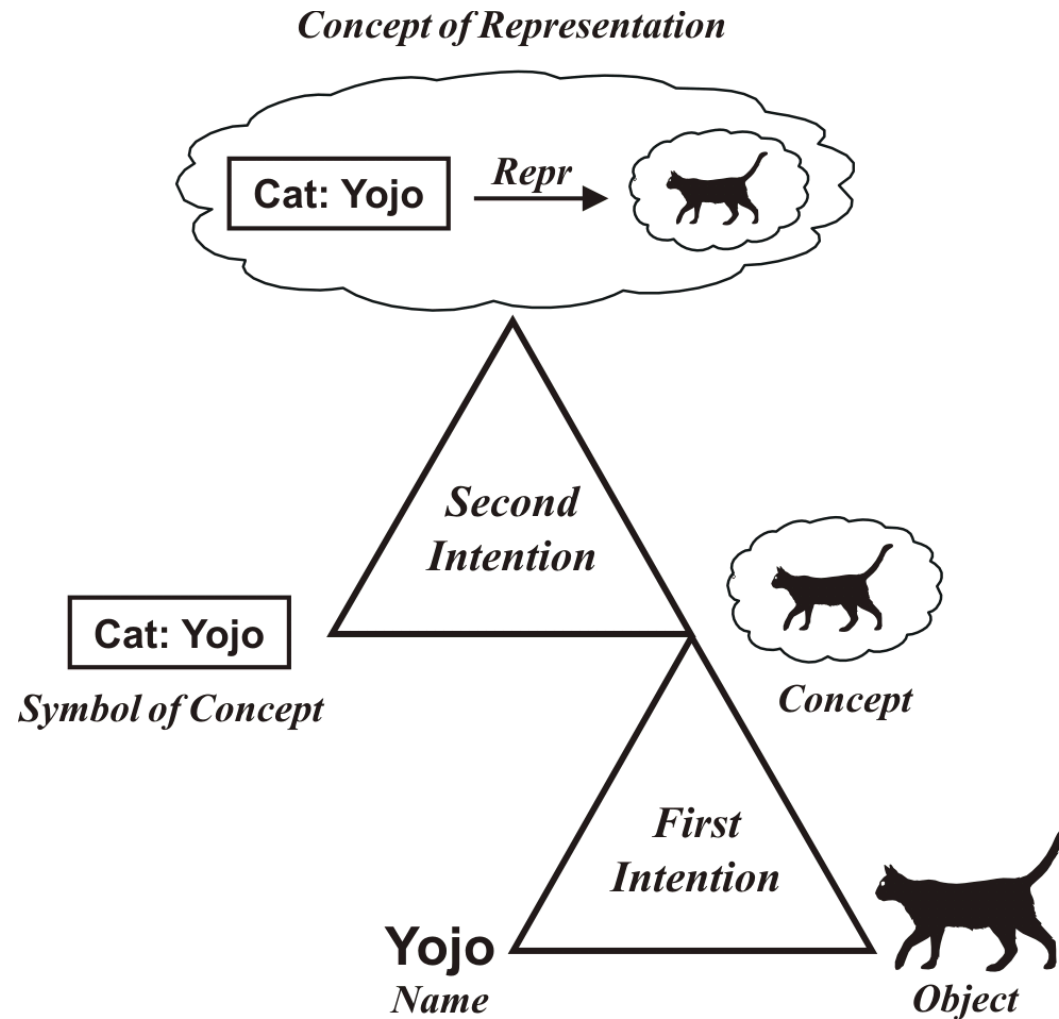
All animals can recognize signs of the material triad.

Many mammals and birds can recognize signs of the relational triad.

Signs of the formal triad require some kind of language. Apes, dolphins, and parrots may be able to learn and use some formal signs.

For further discussion, see <http://www.jfsowa.com/pubs/signs.pdf>

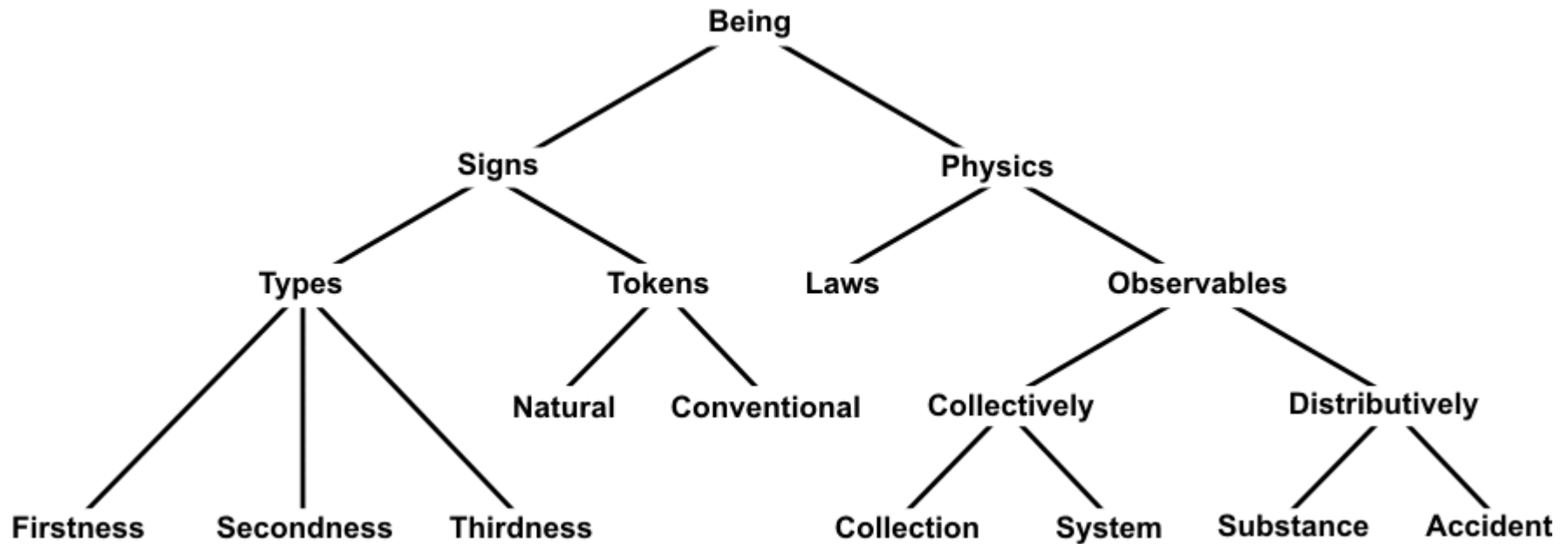
Reasoning About Representations



Peirce allowed meaning triangles to be linked at any nodes:

- A symbol of a concept may refer to a concept.
- A concept of representation relates the symbol to the concept.

Updated Version of Wilkins' Ontology



This diagram preserves the pattern, but it relabels the nodes:

- The top division distinguishes signs from their physical referents.
- The creator is replaced by the laws of physics. Theists can think of the laws as the *logos*, which John the Evangelist said is God.
- Sign types are defined by laws, and sign tokens refer to observables.
- The types may be organized in triads, as defined by C. S. Peirce.

Can Any Ontology be Complete?

The updated version of Wilkins' ontology is more complete than the great majority of published ontologies:

- Signs would include all languages, natural or artificial, and any kind of data or metadata on the WWW.
- Laws would include theories about any natural, artificial, planned, hypothetical, or fictional phenomena in the universe.
- Collections and Systems include all structures and organizations.

But science, technology, and the world are constantly changing.

- A general framework can remain useful for centuries.
- But nobody can anticipate the innovations in the next 20 years.

Observation by Alfred North Whitehead:

“Systems, scientific and philosophic, come and go. Each method of limited understanding is at length exhausted. In its prime, each system is a triumphant success: in its decay it is an obstructive nuisance.”

Thesaurus vs. Ontology

Peter Roget was a secretary of the Royal Society who developed a thesaurus of words instead of an ontology of things.

- **A much simpler system of classification than Wilkins'.**
- **A top level with just six categories: Abstract relations, Space, Matter, Intellect, Volition, Affections.**
- **A bushy hierarchy with just three layers beneath the top level.**
- **No definitions, differentiae, inheritance, or logic.**

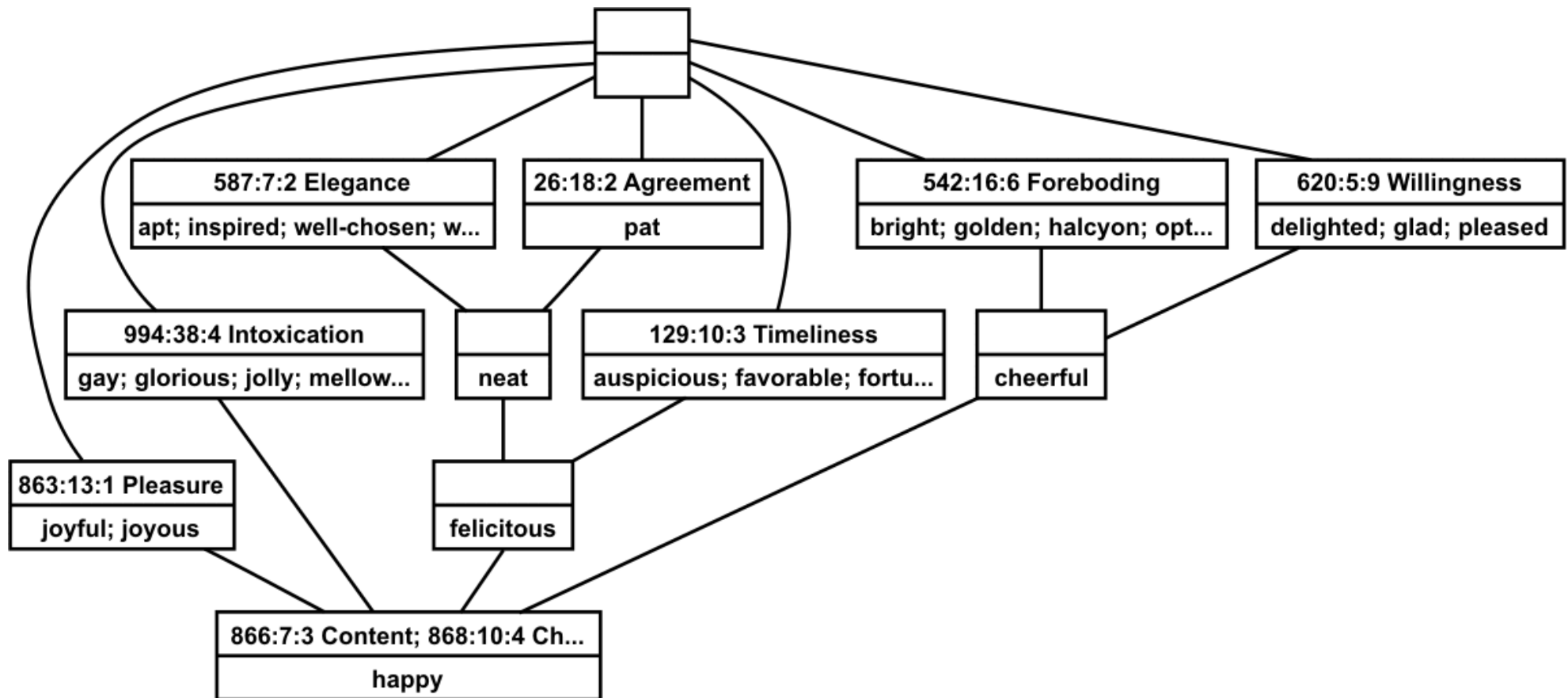
Roget's first edition (1852) was an instant success:

- **By 1869, he had produced 28 editions; his son continued the work.**
- **Computer versions are used in natural language processing (NLP).**

The modern WordNet is closer to a thesaurus than an ontology:

- **WordNet has a simple top level.**
- **It has no formal definitions, differentiae, inheritance, or logic.**
- **But it is a widely used resource for NLP in several languages.**

Concept Neighborhood for *happy*



A lattice of word senses derived from Roget's Thesaurus by FCA.

To see a similar lattice for any word, go to the FCA web site:

For Roget's Thesaurus, <http://www.ketlab.org.uk/roget.html>

For WordNet, <http://www.ketlab.org.uk/wordnet.html>

For concept neighborhoods, <http://www.upriss.org.uk/papers/icfca10.pdf>

Lexical Collocations

Patterns of words that are typically combined for some purpose:

1. Creation or activation patterns: Verb Noun.

Make an impression, compose music, fly a kite, spin a top, launch a missile.

2. Eradication or nullification patterns: Verb Noun.

Reject an appeal, lift a blockade, raze a house, repeal a law, revoke a license.

3. Modifiers that intensify a noun: Adjective Noun.

Reckless abandon, pitched battle, crushing defeat, sweeping generalization.

4. Characteristic verbs with a given subject: Noun Verb.

Alarms go off, bees swarm, blizzards rage, blood circulates, bombs explode.

5. Units of things or units of stuff: Noun of Noun.

Herd of buffalo, bouquet of flowers, word of advice, act of violence.

6. Modifiers that intensify an adjective: Adverb Adjective.

Deeply absorbed, strictly accurate, intimately acquainted, keenly aware.

7. Modifiers that intensify a verb: Verb Adverb | Adverb Verb.

Affect deeply, anchor firmly, appreciate sincerely, argue heatedly.

Examples from M. Benson, E. Benson, & R. Ilson (1986) *The BBI Combinatory Dictionary of English*.

Relating Language to Logic

Peirce wrote a succinct, but accurate summary of the issues:

“It is easy to speak with precision upon a general theme. Only, one must commonly surrender all ambition to be certain. It is equally easy to be certain. One has only to be sufficiently vague. It is not so difficult to be pretty precise and fairly certain at once about a very narrow subject.” (CP 4.237)

Implications:

- **A precise formal ontology of everything can be stated in logic, but it’s almost certainly false in many important respects.**
- **A looser classification, such as WordNet or Roget’s *Thesaurus*, can be more flexible for representing patterns of words.**
- **A specification in logic can be “pretty precise and fairly certain” only for a very narrow subject.**

Logic is an abstraction from language that emphasizes patterns of reasoning, but the patterns of words are also important.

Organizing a Large Ontology

No single ontology can ever be complete, consistent, and useful.

An underspecified framework for everything with an open-ended collection of microtheories is more useful and practical:

- **An upper-level ontology, such as Aristotle's, Wilkins', or Kant's, can show the broad patterns of how everything fits together.**
- **But different problems for different purposes require different representations and algorithms for processing the details.**
- **For interoperability, upper level definitions must be underspecified with the barest minimum of axioms and differentiae.**
- **For precise reasoning and problem solving, the details must be pushed down to the highly specialized, low-level microtheories.**

To be humanly intelligible, ontology must be related to language.

Lexical resources that show the patterns of words are valuable, but they should never be confused with ontologies.

7. Simplifying the User Interface

Tools like iPhone and Facebook help people do complex tasks without forcing them to take a training course.

Those tools take advantage of their background knowledge.

The patterns of knowledge in any field are available in the documents, web sites, and databases of that field.

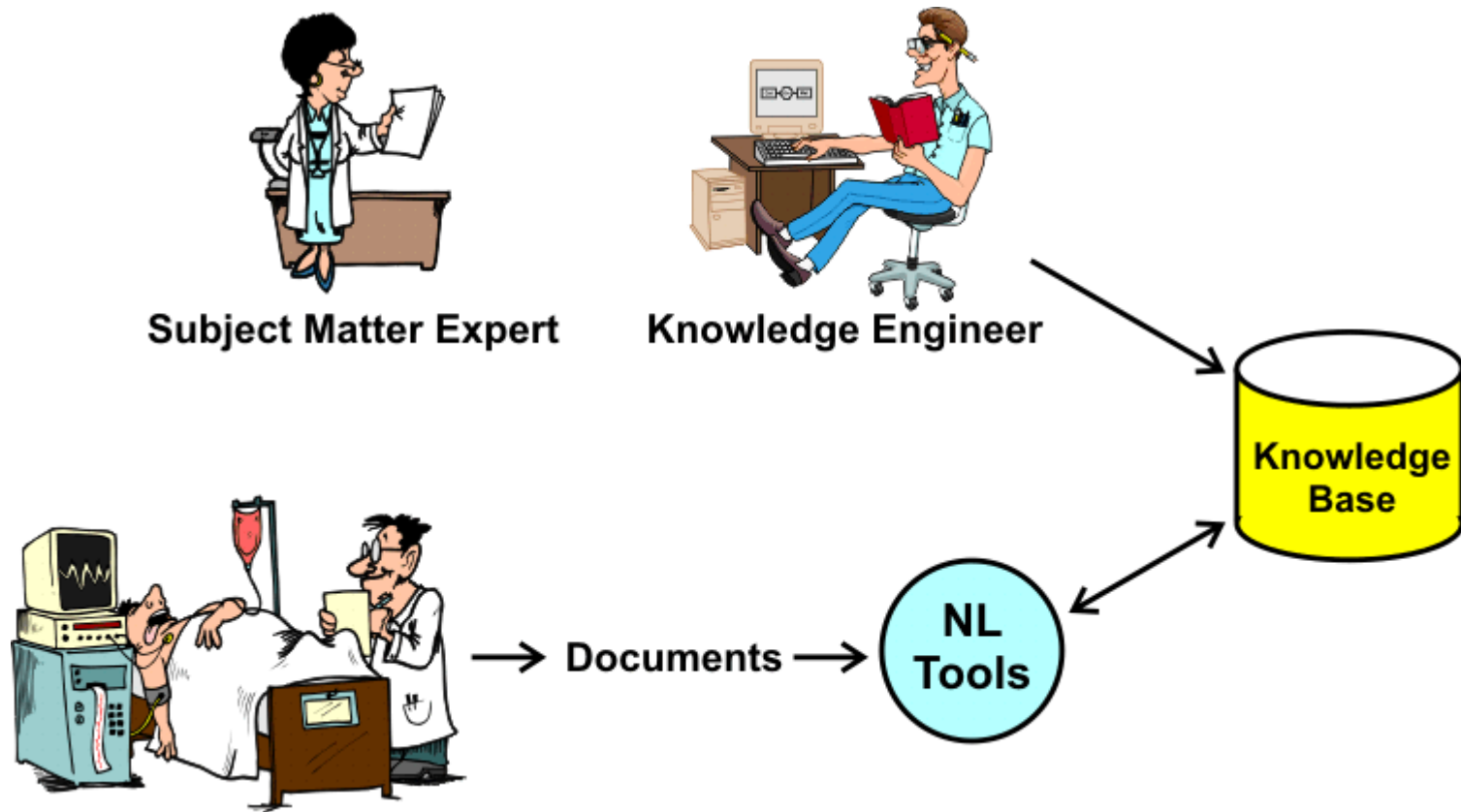
Knowledge acquisition tools should derive the basic patterns from the same sources as people.

Subject-matter experts should use their preferred languages and diagrams to read and correct what the computer derives.

Zero training principle:

- SMEs should not take a training course from the computer.**
- Instead, the computer system should learn from the SMEs and from the same sources as the SMEs.**

Old Fashioned Knowledge Acquisition



Knowledge engineers acquire knowledge from a SME and translate it to some knowledge representation language.

But a KE is also a highly trained professional.

Hiring a SME to train a KE doubles the cost.

A Better Division of Labor

Knowledge system:

- **Acquire knowledge from structured data, training examples, unstructured natural language, and occasional questions.**
- **Communicate with people in any notation they prefer.**

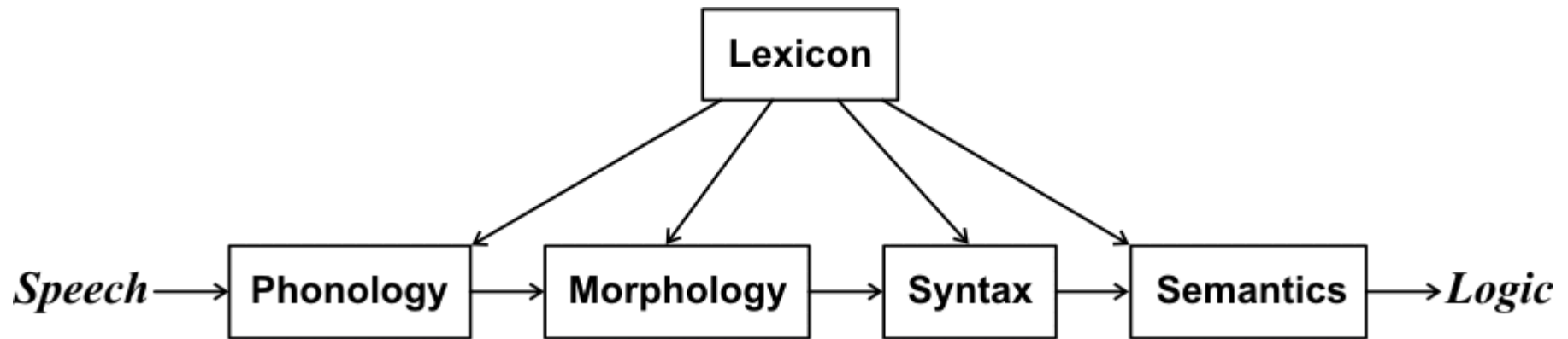
Subject matter experts:

- **Answer questions and correct errors by the system.**
- **Communicate in their preferred notations and diagrams.**
- **Contact a KE only to report problems or to request new features.**

Knowledge engineers:

- **Work with application programmers on the interfaces between the programs and the knowledge system.**
- **Respond to requests by the SMEs.**

Translating Language to Logic



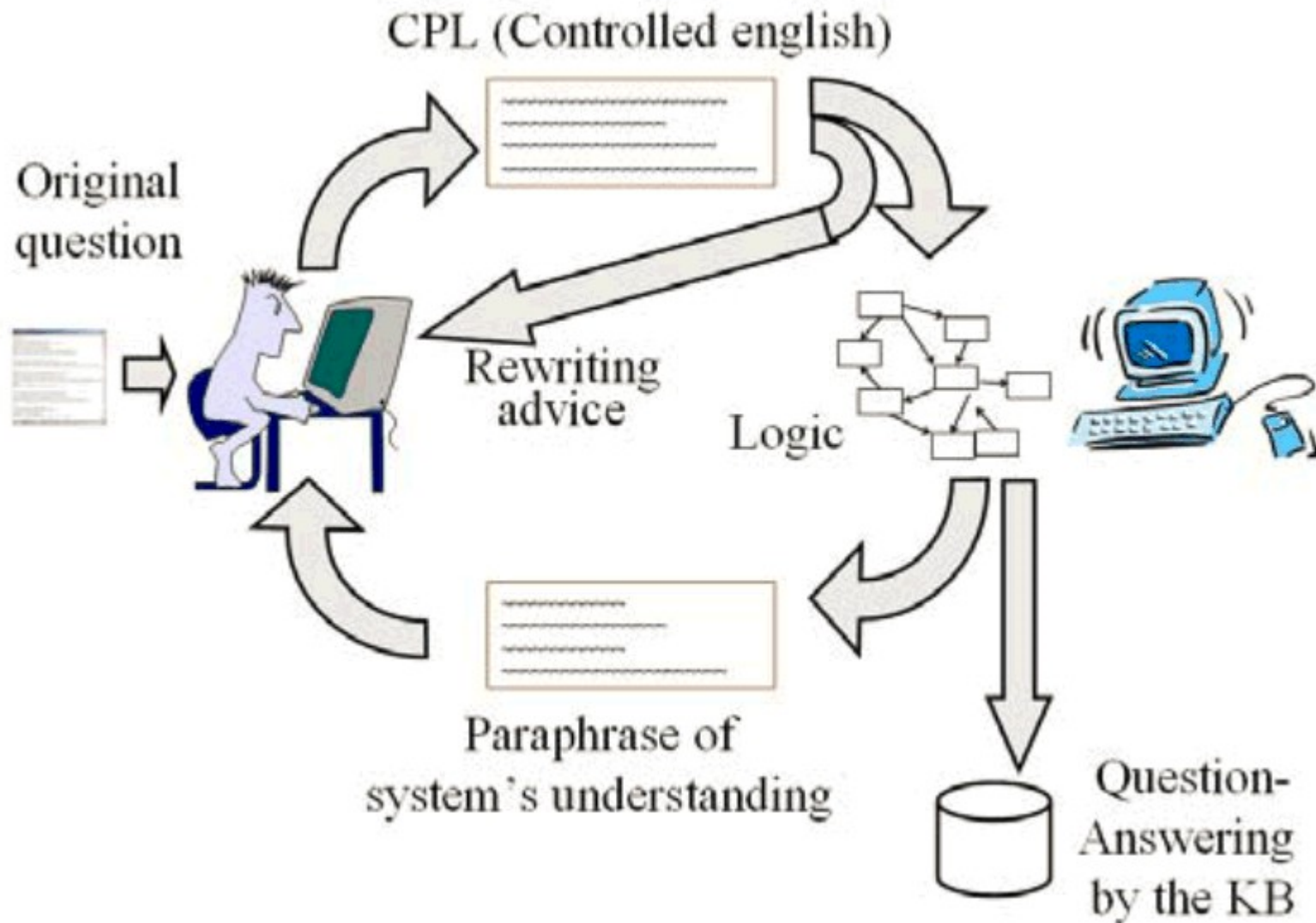
This diagram represents Montague's view that natural languages can be treated as formal languages.

Each stage requires a complete specification of every detail in logic or some equally formal notation.

That level of precision is possible for a controlled NL, but not for the kind of language that people normally use.

As Alan Perlis said, “*You can't translate an informal language to a formal language by any formal algorithm.*”

Human Translation from NL to CNL



Source: P. Clark, et al., Capturing and Answering Questions Posed to a Knowledge-Based System, <http://www.ai.sri.com/pubs/files/1547.pdf>

Translating English to CPL

A question from the Advanced Placement Exam in physics:

A cyclist must stop her bike in 10 m. She is traveling at a velocity of 17 m/s. The combined mass of the cyclist and bicycle is 80 kg. What is the force required to stop the bike in this distance?

Restated in a version of controlled English (CPL):

An object moves.

The mass of the object is 80 kg.

The initial velocity of the object is 17 m/s.

The final velocity of the object is 0 m/s.

The distance of the move is 10 m.

What is the force on the object?

The method of translation is hard to learn, and the people are the only ones who do any learning.

Mean Tries to Completion (MTC)

People often stray outside the limits of a controlled NL.

The MTC rate measures the mean number of tries to rephrase a sentence before the author succeeds or gives up.

For questions in physics taken from the Advanced Placement Exams, the MTC was 6.3.

For AP chemistry questions, the MTC was 6.6.

But for AP biology questions, the MTC was 1.5.

The reason why biology performance was better is that most of the questions could be stated in simple patterns, such as *What is an X?* or *What is the Y of X?*

Instead of training people to write CNLs, it's better to train the computers to read the NLs that people already know.

Controlled Natural Language

Aristotle invented the first controlled NL for his syllogisms.

CNLs are easy to read, but they require training to write.

SMEs don't use CNLs when they talk or write to one another.

The restrictions on syntax and semantics required for a CNL are much easier for a computer to enforce.

Recommendation:

- **Let people use any notation they prefer, including full NLs.**
- **Develop learning methods that enable computers to interpret more NL patterns as they acquire more knowledge.**
- **But the computers always generate CNLs in response.**

Over time, communication becomes more efficient as people and computers learn each other's patterns.

Translation by Pattern Matching

Many machine-translation systems learn corresponding patterns in two NLs from a bilingual corpus:

- For any language pair, find texts written in both languages.
- For each pair of texts, identify corresponding sentences.
- Analyze multiple sentence pairs to derive patterns of phrases.

NLs have patterns at many different levels:

- N-grams: unanalyzed strings of N words (or other units).
- Named entities: *Bob* = *Robert Smith* = *Mr. R. W. Smith*.
- Parse trees: syntactic patterns of phrases in each sentence.
- Logic: quantifiers, Boolean operators, and anaphoric references.
- Ontology: categories of the referents of each word or phrase.
- Context: patterns in the text that surrounds each sentence.

A translation can be improved by using multiple kinds of patterns to verify, extend, and correct one another.

Relating NLs to Structured Data

A task that seems very different from relating two NLs:

- All natural languages can be represented as linear strings, but structured data can have many different formats.
- NL texts consist of sentences, but database tables, RDF triples, and logical formulas don't have a 1-to-1 mapping to sentences.

But at a semantic level, the tasks are similar:

- Semantic theories for NLs use logic to represent the meaning.
- All versions of structured data can be translated to logic.
- Therefore, the task consists of relating two versions of logic.
- Unfortunately, the hardest part of the task is to derive a correct translation from NLs to logic.

Suggestion: Use patterns derived from structured data to verify and correct the mapping from NL to logic.

Training Example

Mass	Initial velocity	Final velocity	Distance	Force
80 kg	17 m/s	0 m/s	10 m	?

This table is a structured form that relates English words and phrases to the values needed to solve the problem.

An NLP system can translate the English labels at the top of the table to a version of logic, such as conceptual graphs (CGs).

To relate this table to the original English text, it must map graphs derived from the table to graphs derived from the text.

This is a graph mapping problem. The labels and values in the table provide starting points for the mapping.

For a computer to learn from such examples, it must have a method for finding, relating, and remembering similar graphs.

Tools for Processing CGs

1. Structure mapping.

- Algorithms based on 30 years of R & D in artificial intelligence.

2. Ontology extraction.

- Translate a group of documents to conceptual graphs.
- Identify and classify the significant terms (words and phrases).
- Use #1 to determine structural patterns that relate the terms.
- Iterate these steps to improve the ontology and the translation.

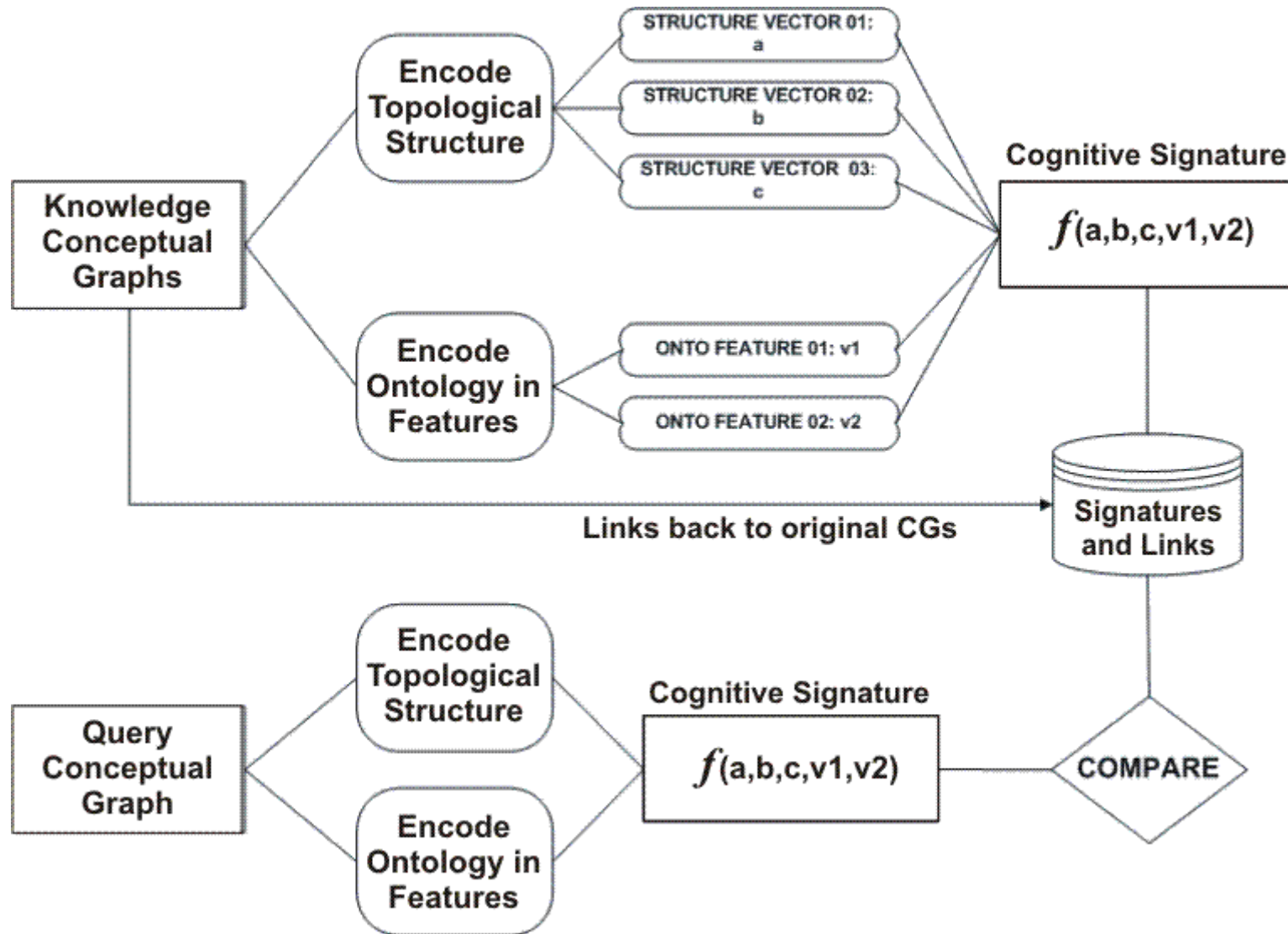
3. Cognitive Memory™.

- Use ontology from #2 to encode CGs in numeric signatures.
- Use Locality Sensitive Hashing (LSH) to find related graphs.

4. VivoMind Analogy Engine™.

- Use methods of #1, #2, and #3 to find mappings among CGs.
- With the speed of #3, VAE can find related CGs in log time.

Cognitive Memory™



Basis for the VivoMind Analogy Engine (VAE)

Mapping the Example to the Text

Mass	Initial velocity	Final velocity	Distance	Force
80 kg	17 m/s	0 m/s	10 m	?

Mass → “*The combined mass of the cyclist and bicycle is 80 kg.*”

Initial velocity → “*She [a cyclist] is traveling at a velocity of 17 m/s.*”

Final velocity → “*A cyclist must stop her bike in 10 m.*”

Distance → “*A cyclist must stop her bike in 10 m.*”

Force → “*What is the force required to stop the bike in this distance?*”

The mapping uses multiple knowledge sources:

- English grammar, basic vocabulary, general ontology.
- Words and phrases that occur in the examples and the text.
- Numeric values that occur in the examples and the text.

Learning and Generalizing

A computer system can begin the mapping with one example.

But more examples and documents would enable it to be more accurate and need less help and correction from a SME.

For this problem, the phrase *combined mass* can be treated as a single unit, but other problems may require further analysis.

The system might guess that the following sentence states the initial velocity: *She [a cyclist] is traveling at a velocity of 17 m/s.*

It would need to know or learn that *stop* implies a final velocity of 0 in the sentence *A cyclist must stop her bike in 10 m.*

It would need to know or learn that *10 m* in that sentence is a measure of distance.

The system can solve some problems with a minimal amount of knowledge and solve more problems as it learns more.

A Commercial Application

A legacy re-engineering task for a large corporation.

Analyze and relate all software and documentation.

Some programs in daily use are up to 40 years old:

- **1.5 million lines of COBOL programs.**
- **100 megabytes of English documentation — reports, manuals, e-mails, Lotus Notes, HTML, and program comments.**

Goal:

- **Analyze the COBOL programs.**
- **Analyze the English documentation.**
- **Compare the two to generate:**
 - English glossary of all terms with index to the software,**
 - Structure diagrams of the programs, files, and data,**
 - List of discrepancies between the programs and documentation.**

Excerpt from the Documentation

The input file that is used to create this piece of the Billing Interface for the General Ledger is an extract from the 61 byte file that is created by the COBOL program BILLCRUA in the Billing History production run. This file is used instead of the history file for time efficiency. This file contains the billing transaction codes (types of records) that are to be interfaced to General Ledger for the given month.

For this process the following transaction codes are used: 32 — loss on unbilled, 72 — gain on uncollected, and 85 — loss on uncollected. Any of these records that are actually taxes are bypassed. Only client types 01 — Mar, 05 — Internal Non/Billable, 06 — Internal Billable, and 08 — BAS are selected. This is determined by a GETBDATA call to the client file.

Note that this text contains specialized jargon, nonstandard syntax, and names of computer programs and files.

An Important Simplification

An extremely difficult and still unsolved problem:

- **Translate English specifications to executable programs.**

Much easier task:

- **Translate the COBOL programs to conceptual graphs.**
- **Those CGs provide the ontology and background knowledge.**
- **The CGs derived from English may contain errors.**
- **VAE matches the CGs from English to CGs from COBOL.**
- **The COBOL CGs show the expected patterns in the text.**
- **They can also detect errors and insert missing information.**

The patterns derived from COBOL provide a formal semantics for interpreting the patterns in English.

Interpreting Novel Patterns

Many texts contain unusual lexical patterns.

They may be elliptical abbreviations of standard syntax.

Or they may be *ad hoc* grammar invented for a special purpose:

- 32 — *loss on unbilled*
- 72 — *gain on uncollected*
- 85 — *loss on uncollected*

The parser generated a CG with a default relation (Link):

[Number: 32]→(Link)→[Punctuation: “—”]→(Link)→[Loss]→(On)→[Unbilled]

The value 32 was stored as a constant in a COBOL program.

The phrase “loss on unbilled” was written as a comment.

The patterns derived from the COBOL data and comments match the patterns derived from the English documents.

Results

Job finished in 8 weeks by Arun Majumdar and André LeClerc.

- **Four weeks for customization:**
Design, ontology, and additional programming for I/O formats.
- **Three weeks to analyze the documents and run VAE:**
VAE handled matches with strong evidence (close semantic distance).
Weak matches were confirmed or corrected by Majumdar and LeClerc.
- **One week to produce a CD-ROM with the desired results:**
Glossary, data dictionary, data flow diagrams, process architecture, system context diagrams.

Estimate by a major consulting firm: 40 people two for years to analyze the documentation and generate the cross references.

With the VivoMind Analogy Engine, it took 15 person weeks.

Relating Formal and Informal CGs

The legacy re-engineering task used two kinds of processing.

Precise, detailed analysis:

- **Analyze the COBOL programs and translate them to CGs.**
- **Detect discrepancies between different programs.**
- **Detect discrepancies between programs and documentation.**

Less detailed analysis for indexing and cross references:

- **Create an index of English terms and names of programs.**
- **Map English documents to the files and programs they mention.**

Conceptual graphs derived from COBOL are precise.

But CGs derived from informal English are just as informal.

All precise reasoning is performed on CGs derived from COBOL or on CGs from English that are *corrected* by CGs from COBOL.

Patterns of Patterns of Patterns

Knowledge in the brain consists of patterns of patterns.

All languages, natural and artificial, express patterns.

Structured representations in tables, diagrams, and networks are based on computable patterns.

Recommendations:

- **For teaching people, emphasize the patterns of the subject.**
- **For designing artifacts – hardware, software, or houses – start by analyzing the patterns and revising them as needed.**
- **For human-computer interfaces, build on the patterns that people use in talking, thinking, and working with the subject.**
- **For interoperability among computer systems, focus on the fundamental patterns, not the quirks of specific notations.**

Related Readings

Future directions in semantic systems, by J. F. Sowa,

<http://www.jfsowa.com/pubs/futures.pdf>

Fads and fallacies about logic, by J. F. Sowa,

<http://www.jfsowa.com/pubs/fflogic.pdf>

The role of logic and language in ontology, by J. F. Sowa,

<http://www.jfsowa.com/pubs/rolelog.pdf>

Conceptual graphs, by J. F. Sowa,

http://www.jfsowa.com/cg/cg_hbook.pdf

Slides for a tutorial on the goal of language understanding, by J. F. Sowa,

<http://www.jfsowa.com/talks/goal.pdf>

Publications about tools and techniques that use ontology design patterns,

http://dx.doi.org/10.1007/11574620_21

<http://www.semantic-web-journal.net/sites/default/files/swj65.pdf>

<http://www.cs.ubc.ca/~poole/papers/PooleSmythSharma2009.pdf>

Web site for controlled natural languages,

<http://sites.google.com/site/controllednaturallanguage/>

ISO/IEC standard 24707 for Common Logic,

[http://standards.iso.org/ittf/PubliclyAvailableStandards/c039175_ISO_IEC_24707_2007\(E\).zip](http://standards.iso.org/ittf/PubliclyAvailableStandards/c039175_ISO_IEC_24707_2007(E).zip)

For other references, see the bibliography at <http://www.jfsowa.com/bib.htm>