Controlled Natural Languages For Semantic Systems

A Roadmap of Directions to Explore

John F. Sowa

1 September 2014

Directions to Explore

1. What are controlled natural languages (CNLs) ?
2. What are semantic systems?
3. Common Logic and its mapping to and from CNLs.
4. CNLs as a bridge between NLs and formal systems.
5. Methodologies and missed opportunities.
6. Full natural language, jargon, slang, and folksonomies.
7. How can semantic systems facilitate the interoperability and integration of all systems, independently of the language, tools, or methodology with which they were implemented?

A "Yellow Brick Road" with seven diversions to explore.

1. Controlled Natural Language

A subset of a natural language that has a well-defined mapping to and from a computable form.

First CNL: Aristotle's subset of Greek for expressing logic and the rules of *syllogisms* for reasoning about it.

CNLs support precise communication:

- **For stating requirements and specifications by humans to humans.**
- **For commands and assertions by humans to computers.**
- **For answers, explanations, and help from computers to humans.**

Advantages of controlled natural languages:

- **More readable than typical computer languages.**
- **Less training for people who write CNLs.**
- **No training for people who read CNLs.**

Some Examples of CNLs

1. Aristotle's notation for logic and syllogisms

2. Intellect keyword system

3. Transformational Question Answering System (TQA)

4. Executable English

5. Attempto Controlled English (ACE)

6. REL, ASK, and families of sublanguages

Systems #1, #3, #4, #5, and #6 meet the definition of a CNL, but #5 is the only one whose designers call a CNL.

The designers advertised #2 as "true natural language," but a more accurate description would be "keyword system."

Aristotle's Logic

Aristotle represented logic as a CNL for reasoning about ontology.

He used a subset of Greek to represent four sentence patterns:

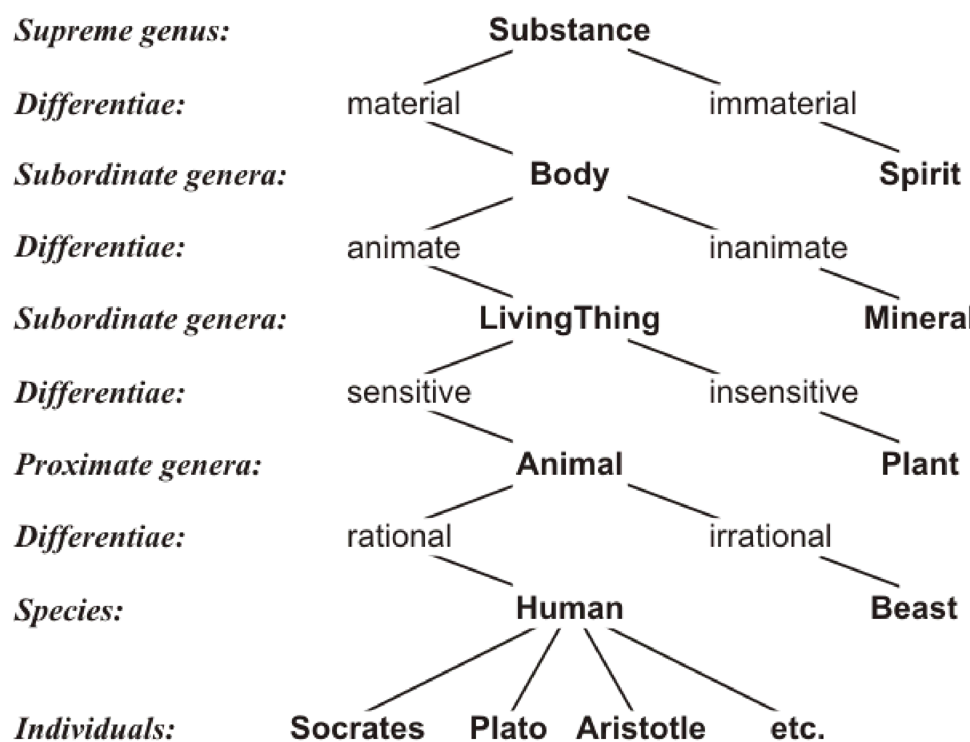
A	Universal affirmative:	Every X is Y.
I	Particular affirmative:	Some X is Y.
E	Universal negative:	No X is Y.
O	Particular negative:	Some X is not Y.

The letters A, I, E, and O were introduced in the middle ages as mnemonics for naming and remembering the combinations:

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

The letters X and Y represent words or phrases that specify categories of the ontology or differentiae that distinguish them.

Tree of Porphyry



A tree of Aristotle’s categories and differentiae was found in a manuscript of a commentary by the philosopher Porphyry.

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

Aristotle’s Syllogisms

The first and most famous syllogism is named Barbara:

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

The syllogism named Darii applies to particular individuals:

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

Syllogisms of the pattern Barbara support the inheritance of properties from a supertype (Animal) to a subtype (Human).

Syllogisms of the pattern Darii support the inheritance of properties to particular individuals.

Negative Syllogisms

The first negative syllogism is named Celarent:

E	No spirit is a body.
A	Every human is a body.
E	Therefore, no human 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.

Negative syllogisms express constraints on the type hierarchy.

Summary of the CNL for Syllogisms

One premise of a syllogism must be a type A or E sentence that specifies a type-subtype relationship among categories.

That premise uses the verb *is*: "(Every | No) X is a Y."

The other premise may use any verb or verb phrase.

These options are sufficient to express many description logics in the same sentence patterns as Aristotelian syllogisms

– usually with a great improvement in readability.

Aristotle's syllogisms were the dominant form of logic for over two millennia.

Their CNL notation is still useful for many purposes today.

Intellect

A database query system developed in the 1970s.

- **Translated English sentences to SQL queries.**
- **Advertised as "true natural language."**
- **For simple queries, it was much easier to use than SQL.**
- **But for complex queries, it was highly unreliable.**
- **The Intellect manuals did not explain how it worked.**

Examples in the following slides were tested on a sample database that was delivered with the Intellect system.

They show how the Intellect system actually worked.

Intellect Customization

Before Intellect can access a database, someone at the user's location must customize the dictionary.

Customization maps words or phrases to the values or the columns of the database tables.

For convenience, multiple terms (synonyms) can be mapped to the same column or value.

Those words may be any character strings – they can be taken from English or any other language.

Some special words, such as *not*, are recognized.

But many words are ignored: ***a, the, is, some, every, all, in, on, of...***

Warning: The words *some* and *every* are critical to logic.

Intellect Grammar

Advertising says that Intellect can handle incorrect grammar.

But that's because it ignores grammar.

Intellect gave the same response to the following 3 questions:

1. **Who are all of the exempt female managers?**
2. **Managers female exempt the of all are who?**
3. **Who the be of females all manager on exempts of of?**

Note that question #2 has all the words of #1 in reverse order.

Question #3 deletes the plural marker *s* on some words and adds it to other words.

Logical Weaknesses

Some questions for Intellect and the answers it returned:

Q: How many managers are exempt?
A: 36

Q: How many managers are nonexempt?
A: 5

Q: How many managers are not exempt?
A: 5

Q: How many exempt managers are not exempt?
A: Your request contains a logical inconsistency.

Q: How many exempt managers are nonexempt?
A: 41

Note: the first four answers are correct; the last answer is 36+5.

So-called Logical Form

Intellect heuristics for translating English to logic:

- **If two terms refer to different database columns, connect them with a logical *and*.**
- **If two terms refer to the same database column, connect them with a logical *or*.**
- **Assume that *not* negates the next database term.**
- **Treat all quantifier words as equivalent to *some*.**

For the question "How many exempt managers are nonexempt?" the words *exempt* and *nonexempt* refer to the same column.

Therefore, the question was treated as equivalent to "How many managers are exempt or nonexempt?"

More Logical Weaknesses

Q: Are all managers men?
A: Yes

Q: Are all managers women?
A: Yes

Q: How many men are also women?
A: 441

According to the Intellect rules for translating English to logic:

- **“all” is equivalent to “some”;**
- **“man” is defined as “male employee” ;**
- **“woman” is defined as “female employee”;**
- **“also” is ignored.**

Therefore, the last question above is equivalent to

Q: How many employees are male or female?

Evaluation of Intellect

Definitely not "true natural language."

Too loose to be considered a "controlled natural language."

The logical rules are hopelessly unreliable for complex queries.

But the rules are adequate for existential-conjunctive logic:

- **The only implicit quantifier is *some* (existential).**
- **The only implicit Boolean operator is *and* (conjunction).**
- **There are no negations or disjunctions, explicit or implicit.**

For this logic, Intellect acts like a keyword retrieval system.

With its customizing tools, a database administrator can easily define meaningful keywords as synonyms for the DB columns.

TQA System

Transformational Question Answering System at IBM Research

- **Based on a version of Chomsky's transformational grammar.**
- **Translated English questions to first-order logic.**
- **Translated the logic to SQL for a relational database.**
- **Verified its operations by translating the SQL to controlled English.**

Some observations:

- **TQA was called an English query system, but its formal grammar could be considered a version of controlled English.**
- **TQA had a rich grammar that allowed users to ask their questions in a wide range of commonly used expressions.**
- **But customizing TQA required more knowledge of grammar and logic than just defining synonyms for the columns of a DB table.**

Testing the TQA System

IBM decided to test TQA with a typical customer.

A problem faced by White Plains city officials:

> **During the 1974 gasoline shortage, the land use file was searched by hand on the "use" code to find the locations of all gas stations so that police could go there to direct traffic.**

At Yorktown Research, IBM installed a relational database for all the buildings and parcels of land in the city of White Plains.

Then they connected the TQA system at Yorktown to a terminal in the office of land use planning in the White Plains city hall.

The IBM researchers gave the city planners and officials a short tutorial and a demo on how to type queries and get answers.

Typical Questions Asked by Users

What is the total area of the parcels in ward 6 block 72?

What parcels in the R5 zone on Stevens St. have greater than 5000 sq. ft.?

How many parking spaces are there in ward 1 block 2?

What is the zone of the vacant parcels in subplanning area 410?

How many two family houses are there in the Oak Ridge Residents Assn. ?

Where are the apartment dwellings which have more than 50 units which are more than 6 stories high on Lake St. ?

What is the assessment of the parcels in the Battle Hill Assn. having more than 3 dwelling units?

Source: Operating statistics for the Transformational Question Answering System, by Fred J. Damerau, http://acl.ldc.upenn.edu/J/J81/J81-1003.pdf

Echo Generated by Q-Trans

Original English question, as typed by the user:

What parcels in the R5 zone on Stevens St. have greater than 5000 sq. ft.?

Translation of English to SQL by TQA:

```
SELECT UNIQUE A.JACCN, B.PARAREA
FROM ZONEF A, PARCFL B
WHERE A.JACCN = B.JACCN
AND B.STN = 'STEVENS ST'
AND B.PARAREA > 5000
AND A.ZONE = R5;
```

Translation of SQL to controlled English by Q-Trans:

Find the account numbers and parcel areas for lots that have the street name STEVENS ST, a parcel area of greater than 5000 sq. ft., and zoning code R5.

Evaluation of TQA

During the one-year trial, the users typed 788 questions.

65% of the questions were processed correctly.

The remaining 35% had various syntactic and semantic errors.

In most cases, the users rephrased the sentence to get a version that TQA could process.

In rare cases, they called the developers for help.

The users loved it. They were unhappy when the trial period ended, and IBM unplugged the terminal.

But IBM management decided that the cost of customizing and supporting TQA would be too expensive to make it profitable.

Executable English™

A system for writing and running question answering applications in a language called Open Vocabulary Executable English.

Example:

> ***some-person does research into some-subject***
> ***that-subject is a sub topic of some-topic***
> **--**
> ***that-person does research into that-topic***

The first two lines represent predicates in the if-part of some rule.

The line at the bottom represents the conclusion.

The prefix *some*- introduces a variable, and the prefix *that*- refers back to such a variable.

Mapping Executable English to Logic

Following is a translation of the previous rule to the CLIF dialect of Common Logic:

```
(forall (person-x subject-y topic-z)
   (if (and (does_research_into person-x subject-y)
            (is_a_sub_topic_of subject-y topic-z)
       (then (does_research_into person-x topic-z)) ))
```

This example illustrates the methods for mapping Executable English to more traditional notations for logic.

The result is so readable that the author or reader can treat it as English with some special formatting conventions.

The notation is as precise as TQA, but the parsing is simple, the customization is easy, and the implementation is straightforward.

Rules and Explanations

The Executable English system can explain its answers by displaying instances of the rules that were used:

estimated demand 523 in NJ is for 1000 gallons of product-y in October of 2005
for estimated demand 523 0.19 of the order will be product-x from Shell Canada One
1000 * 0.19 = 190
--
for demand 523 NJ for 1000 product-y we use 190 product-x from Shell Canada One

Note that mathematical expressions can be written on any line of a rule or explanation, with * for multiplication and = for equality.

The syntactic convention of writing predicate names with arbitrary English phrases or sentences makes comments unnecessary.

Mapping to Relational Databases

Tables of a relational database, either stored or computed, can be mapped to or from Executable English by writing a single line:

we have this-method transportation from refinery this-name to region this-region		
truck	Shell Canada One	NJ
rail	Shell Canada One	NJ

The prefix *this*- indicates a mapping to one column of a table.

With its inference engine and mapping to SQL, Executable English can support deductive databases.

Evaluation of Executable English

With its highly flexible and readable notation, Executable English demonstrates a novel method for designing user interfaces.

The lack of special grammars, dictionaries, and ontologies makes the rule formats easy to learn, read, and write.

That same lack, which is a benefit for a single developer, may make it difficult to coordinate and integrate the contributions of a large development team over an extended period of time.

But the system makes suggestions to help authors find or remember the exact phrasing of a previous sentence.

For further information, see

http://www.reengineeringllc.com/

For more examples, see

http://www.reengineeringllc.com/Oil_Industry_Supply_Chain_by_Kowalski_and_Walker.pdf

http://www.reengineeringllc.com/EnergyIndependence1.pdf

http://www.reengineeringllc.com/EnergyIndependence1Video.htm

Attempto Controlled English (ACE)

ACE is a controlled natural language that has been developed and used at the University of Zurich for over 15 years.

Like both TQA and Executable English, ACE reliably maps English sentences to a subset of first-order logic.

Like TQA, but unlike Executable English, ACE has a large built-in grammar of English.

ACE has a dictionary of about 100,000 words of English, but it allows those words to be redefined as needed.

The complete ACE system with many associated tools is available for free download and use under the LGPL license.

See http://attempto.ifi.uzh.ch/site/

Verbalizing OWL in ACE

One of the applications of ACE is a verbalizer that translates OWL 2 ontologies to statements in ACE.

See http://attempto.ifi.uzh.ch/site/docs/owl_to_ace.html for a hundred-line OWL 2 ontology that is verbalized in ACE as

> **Everything that is eaten by a goat is a leaf.**
> **Everything that eats nothing but leaves is a goat.**
> **Every animal is something that is a cat or that is a goat.**
> **John is a man.**
> **Everything is eaten by at most 1 thing.**
> **Everything that is eaten by something is a food that is not an automobile.**
> **Everything that is an apple or that is a leaf is a food.**
> **Every human is something that is John or that is Mary.**
> **Every man is a person.**
> **Everything eats at most 1 thing.**
> **Every human is a person that own an automobile.**
> **If X eats something that eats Y then X eats Y.**
> **Everything that eats something is an animal.**
> **If X eats Y then Y hate X.**
> **If X hate Y then Y eats X.**

These ACE statements can then be translated back to OWL 2.

Evaluation of ACE

A good set of tools for developing and using CNL interfaces.

But after 15 years of development, most of the applications have been supported by research grants or student projects.

Self-sustaining development requires commercial applications.

ACE has many useful ideas that could be commercially successful.

But more R & D is needed on the development tools, the methodology, and the human factors for using CNLs.

Would a CNL be more successful if it were the centerpiece that integrated all the other applications of a major system?

REL and ASK Systems

Systems designed and implemented by Fred Thompson, Bożena Thompson, and their students at Caltech.

Rapidly Extensible Language System (REL) in the 1970s

A Simple Knowledgeable System (ASK) in the 1980s.

Based on a methodology for using controlled English:

- A single, grammar-driven processor for all “sublanguages.”
- A relational database for storing facts.
- Utilities for modifying the grammar and ontology and for mapping the resulting parse trees to and from external procedures.

The REL and ASK systems were used by students and faculty at Caltech to implement controlled English for various domains.

See http://authors.library.caltech.edu/26960/2/5165_TR_84.pdf

The New World of Computing

From a vision statement by Fred Thompson in 1992:

> ***"The next decade will see the telephone, personal computer, work station, and television set combined in a single, ubiquitous instrument – the telephone-computer."***

> **See http://www.jfsowa.com/misc/thompson.htm**

Thompson earned a PhD in logic under Alfred Tarski at Berkeley.

But before becoming a professor at Caltech, he had 14 years of experience in software development at GE and Rand.

Bożena Thompson did research for the Georgetown Automatic Translator (GAT), which was commercialized as the Systran MT system and is still widely used today.

That combination of theory and practice contributed to their designs for REL and ASK and their vision for the future.

Languages for the Telephone-Computer

To take advantage of the new hardware, Thompson advocated controlled NLs for both input and output, typed or spoken:

- **"The infinitely variable expressions of language give tangible form to our own immediate view of the world."**
- **"The mechanisms of language are precisely the tools we need and use to express the recursive structures we impose on our experience."**
- **"The rules of grammar, along with the corresponding semantic procedures, constitute the building blocks."**
- **"Sublanguage is a form of human communication that is domain specific, appropriate to that domain and, consequently, highly efficient."**

This vision, which is complementary to the Semantic Web and other semantic technology, should be explored further.

2. Semantic Systems

Computer systems that recognize, represent, and respond to the meaning of the data and the goals of the users.

Examples of semantic systems:

- **Deductive database systems,**
- **Natural language query and analysis systems,**
- **Expert systems and knowledge-based systems,**
- **The Semantic Web and its applications.**

Ultimate goal of semantic systems:

- **Integrate all software around the semantics of the data.**
- **Enable legacy systems to participate in the integration.**
- **Help people interact with computers in a more human way.**

Classical Artificial Intelligence

An approach to problem solving and question answering:

- **Some version of logic for knowledge representation,**
- **An inference engine for drawing conclusions,**
- **A database system for storing facts.**

This approach has been one of the mainstream paradigms of AI since the 1960s.

The largest and most sophisticated single system is Cyc, which has been in continuous development since 1984.

The Semantic Web is even larger, but it is a looser coalition of independently developed sites, and its knowledge representation is not as expressive as the CycL logic.

Rule-Based Expert Systems

A special case of classical AI that was popular in the 1980s and is still important for many kinds of applications.

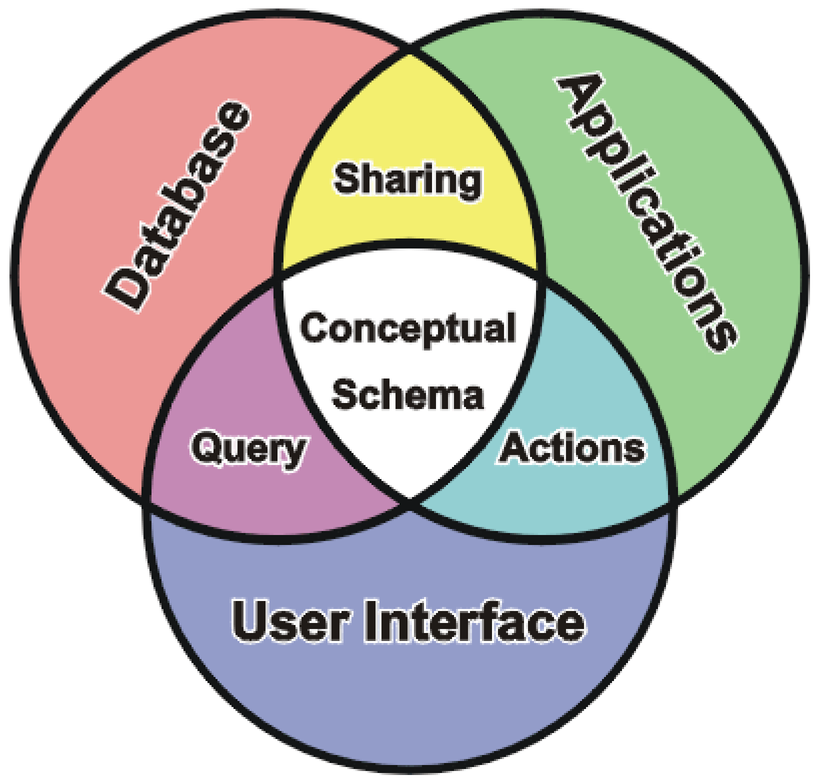
Unfortunately, expert systems acquired a bad reputation:

- **They were overhyped as a panacea for almost everything.**
- **The tools and methodologies available at that time were unable to support users who did not have sufficient training.**
- **They were not well integrated with the mainstream technologies for software design, development, and deployment.**

Some of the companies started in the 1980s still provide rule-based systems with good tools and methodologies.

But for commercial applications, the term 'business rule' has become more acceptable than 'expert system'.

Conceptual Schema



Three-schema architecture by ANSI SPARC in 1978:

- Conceptual schema defines the semantics of a database.
- Physical schema defines the storage and access methods.
- Application schema defines the APIs for programming languages.

Standardizing the Conceptual Schema

Many database experts agreed:

- Logic is neutral on the issue of data storage and access.
- Issues of storage and access are not relevant to semantics.
- Therefore, logic is the only logical choice for the conceptual schema.

But other DB experts disagreed:

- They claimed that programmers should specify the data formats.
- Some claimed that logic was hard for people to understand.
- Commercial DB vendors did not want to disrupt their implementations.

Over thirty years of R & D and ISO proposals:

- Better notations for logic than the SQL where-clause.
- Methodologies, tools, and technical reports, but no standards.

See The Orange Report ISO TR9007 (1982 – 1987): Grandparent of the Business Rules Approach and SBVR, History of the ISO TC97/SC5/WG3 Working Group, *Business Rules Journal,* by J. J. van Griethuysen, Vol. 10, No. 4 (April 2009),
http://www.BRCommunity.com/a2009/b474.html

Database and Knowledge Base Design

Good methodologies are essential:

- Based on natural language and logic, not programming details.
- Respect for legacy systems, not a total rejection of operational systems.
- Supported by tools that subject-matter experts can use and understand.

Some tools have been designed, implemented, and used:

- Natural-language Information Analysis Methodology (NIAM).
- Object-Role Modeling (ORM).
- DOGMA workbench for modeling meaning in context.

Generic ontologies must be specialized for each context:

- Different applications may use shared data in very different ways.
- Enormous amounts of context-dependent details for each application.
- Tools and methodologies must support customization and negotiation.

See Why a data model does not an ontology make, by Robert Meersman,
http://www.starlab.vub.ac.be/website/files/MeersmanBuffaloAug2007.pdf

Semantics of Business Vocabulary and Business Rules

SBVR is a system for representing business rules in logic:

- Semantics of Common Logic with extensions for modal terminology.
- Ontology for business vocabulary.
- Mapping to and from SBVR Controlled English.
- Methodology based on Object-Role Modeling (ORM).
- Evolved from NIAM (Natural language Information Analysis Method).
- Graphic tools derived from NIAM and compatible with UML.

Sample sentences in SBVR Controlled English:

1. It is obligatory that each rental car is owned by exactly one branch.
2. The age of the driver is at least the EU-Rent Minimum Driving Age.
3. Each customer (car rental responsibility) is a corporate renter or is an individual customer.

The capitalized words in sentence #2 or the parentheses in #3 identify the context in which the terms are defined.

Logic Programming (LP)

Logic can be used as a programming language:

- Specify a problem by axioms in some version of logic.
- Design a system that finds one or more models for those axioms.
- The result is the solution (or solutions) to the problem.
- It can also be the answer to some question.

Prolog is the one of the most widely used LP systems.

- When Ted Codd saw Prolog, he said "I wish I had invented that."

Example: The Experian credit bureau.

- They use rules written in Prolog to check everybody's credit score.
- But their rules are proprietary and confidential.
- They are so heavily dependent on Prolog that they bought Prologia, the company founded by Alain Colmerauer, who invented Prolog.

Integration, not Fragmentation

Meaning, as stated in ordinary language, is fundamental to semantic systems of every paradigm:

> **Semantic Web, Business Rules, Deductive Databases, Integrated Development Environments, Service-Oriented Architectures, Object-Oriented Systems, Semantic Clouds...**

Systems from many different paradigms should be able to interoperate, if they focus on the meaning of the data.

But current tools are based on different versions of logic.

Observation by an application developer:

> **"Any one of those tools, by itself, is a tremendous aid to productivity. But any two of them together will kill you."**

Knowledge Management

Karl-Erik Sveiby coined the term *Knowledge Management* for a human-oriented approach to organizational knowledge:

> **"The nurses in a Norwegian private hospital in Oslo wanted to solve a problem: how can we reduce the fear of patients going in to surgery? The idea came up: invite the old patients for coffee and cake together with the new patients and let them talk. The surgeons were against it, but the hospital decided to do a pilot test. It became a success! Both patient categories loved it, and both nurses and surgeons agreed afterwards: the patients' fear had been reduced."**

The hospital generalized that success story to a policy of promoting knowledge sharing among everybody – surgeons, nurses, patients, and support staff.

All semantic systems should focus on the human origin of the knowledge and its relationship to human needs and goals.

For the hospital example, see http://www.sveiby.com/articles/KMCaseHospital.pdf

For other articles by Sveiby, see http://www.sveiby.com/articles/

3. A Family of Logics

First-order logic is a subset or superset of most logic-based notations.

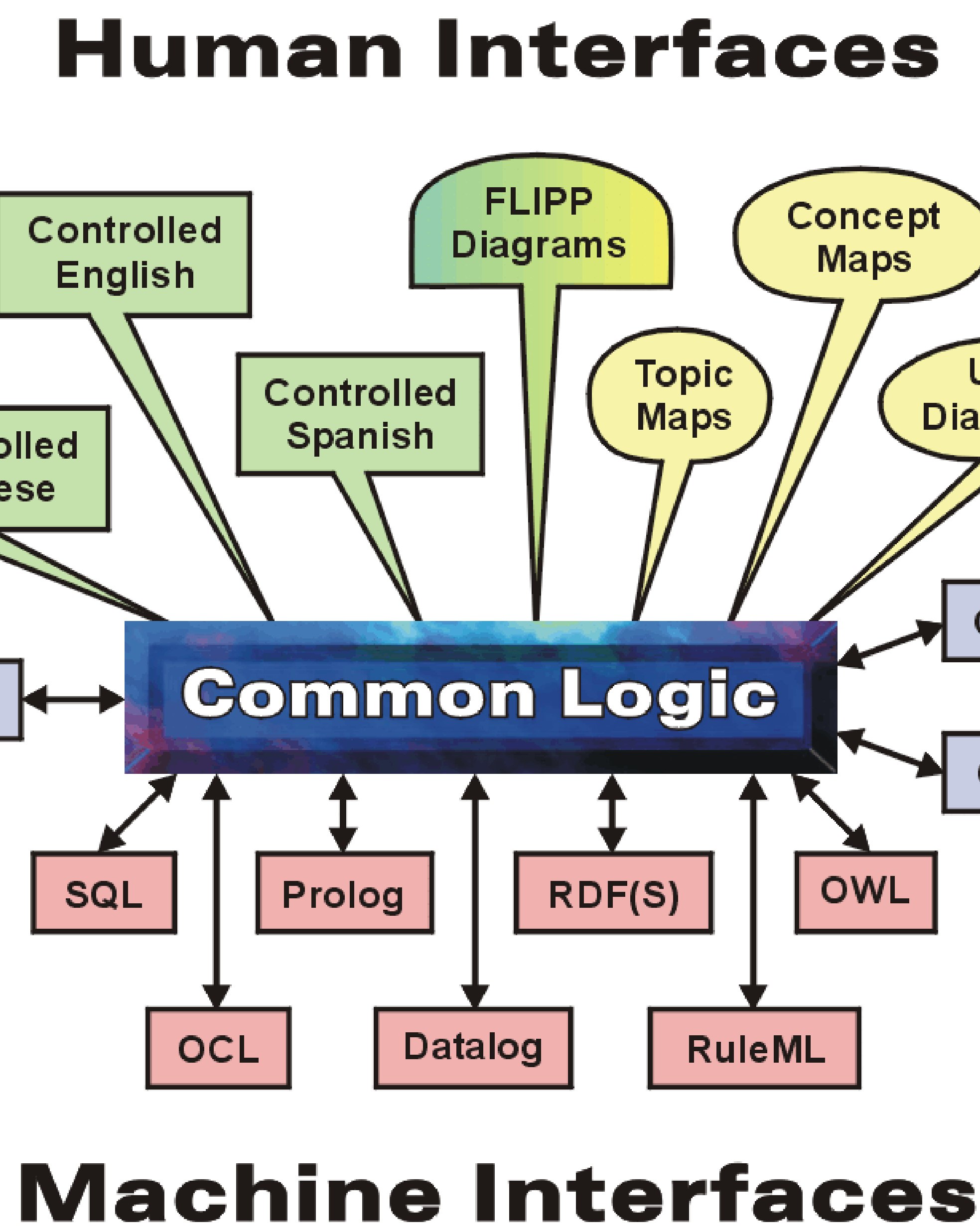
But people are constantly inventing new notations, and they don't want to abandon their favorite notation in favor anybody else's.

The ISO standard 24707 for Common Logic defines a very general semantic foundation for an open-ended family of dialects.

Three dialects are specified in ISO 24707:

- CLIF — Common Logic Interchange Format
- CGIF — Conceptual Graph Interchange Format
- XCL — XML-based notation for Common Logic

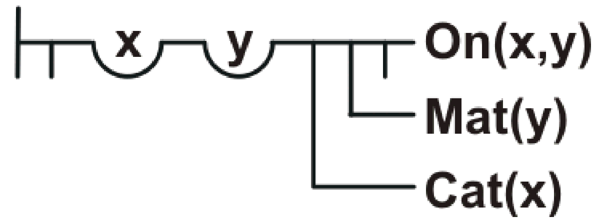
But any notation that uses the common semantics can join the family.



Human Interfaces
Controlled English
FLIPP Diagrams
Concept Maps
Controlled Spanish
Topic Maps
UML Diagrams
Controlled Chinese
CGIF
XCL
Common Logic
CLIF
SQL
Prolog
RDF(S)
OWL
OCL
Datalog
RuleML
Machine Interfaces

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

Gottlob Frege (1879):



Charles Sanders Peirce (1885):

$$\Sigma_x \Sigma_y \text{Cat}_x \bullet \text{Mat}_y \bullet \text{On}_{x,y}$$

Giuseppe Peano (1895):

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

Frege and Peirce developed their notations independently.

Peano adopted Peirce's notation, but changed the symbols.

But all three notations have identical semantics.

Some Modern Notations

SQL query:

```
SELECT FIRST.ID, SECOND.ID
FROM   OBJECTS FIRST, OBJECTS SECOND, SUPPORTS
WHERE  FIRST.TYPE = "Cat"
AND    SECOND.TYPE = "Mat"
AND    SUPPORTS.SUPPORTER = SECOND.ID
AND    SUPPORTS.SUPPORTEE = FIRST.ID
```

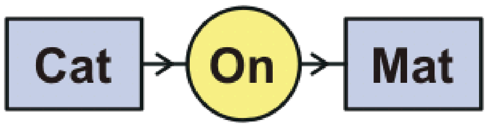
Common Logic Interchange Format (ISO 24707):

```
(exists ((x Cat) (y Mat)) (On x y))
```

Conceptual Graph Interchange Format (ISO 24707):

```
[Cat *x] [Mat *y] (On ?x ?y)
```

Conceptual Graph Display Form:



Controlled English:

```
A cat is on a mat.
```

An Example of OWL

An example of OWL by Pat Hayes, slide 22, of http://is.gd/1ehQK

```
<owl:Class rdf:id="#ChildOfUSCitizenPost1955">
   <owl:intersectionOf rdf:parseType="Collection">
       <owl:Restriction>
          <owl:onProperty rdf:resource="#parentOf"/>
          <owl:allValuesFrom>
             <owl:Restriction>
               <owl:onProperty rdf:resource="#isCitizenOf"/>
               <owl:hasValue rdf:resource="#USA"/>
            </owl:Restriction>
         </owl:allValuesFrom>
      </owl:Restrrition>
      <owl:Restriction>
         <owl:onProperty rdf:resource="#dateOfBirth"/>
         <owl:allvaluesFrom rdf:resource="#YearsSince1955"/>
      </owl:Restriction>
   </owl:intersectionOf>
</owl:Class>
```

Translation to OWL-CL

A language semantically identical to OWL, but translated to Common Logic (with some supporting axioms written in CL).

Previous example translated to OWL-CL in the CLIF dialect:

```
(= ChildOfUSCitizenPost1955
    (And (AllVals parentOf (ValueIs isCitizenOf USA))
          (AllVals dateOfBirth YearsSince1955) )
```

This is a valid CLIF statement, which uses special terms 'And', 'AllVals', and 'ValueIs', which are defined by axioms in CLIF.

Any tool that generates, uses, or reasons with OWL could be adapted to generate, use, or reason with this notation.

More statements could be written in CLIF or any other dialect of CL to relate this statement to other CL statements.

Translation to Controlled English

The previous example of OWL or its translation to OWL-CL could also be written in Common Logic Controlled English:

Define "x is a ChildOfUSCitizenPost1955"
as "every parent of x is a citizen of USA,
and the date of birth of x is after 1955".

The noun 'ChildOfUSCitizenPost1955' is permissible, but a more natural statement would use simpler words:

If the date of birth of a person x is after 1955,
and every parent of x is a citizen of USA,
then x is a citizen of USA.

Common Logic Controlled English

A dialect of Common Logic that looks like English.

CLCE uses a subset of English syntax and vocabulary.

But the CLCE grammar avoids constructions that may cause ambiguities.

CLCE replaces pronouns with temporary names called *variables*.

Examples:

> For every company C,
> exactly one manager in C is the CEO of C;
> every employee of C except the CEO reports to the CEO;
> the CEO of C does not report to any employee of C.
>
> If an integer N is 5, then (N^3 = 125).

The scope of variables, such as C or N, extends to the period at the end.

Note: CLCE is not an ISO standard, but it uses the CL semantics.

CLCE Semantics

CLCE can express the full semantics of Common Logic.

A recursive definition of "reports" in terms of "directly reports":

> **Every employee who directly reports to a manager reports to that manager.**
>
> **If an employee of a company C directly reports to a manager M1 in C, and the manager M1 reports to a manager M2 in C, then the employee reports to the manager M2.**

Definitions link CLCE words and phrases to other versions of logic:

> **Define "x directly reports to y" as (DirectlyReports x y).**
>
> **Define "x directly reports to y" as SQL(select emp, mgr from employees).**

Cautionary Note

Anybody who can read English can read a CLCE statement.

But writing clear, precise, readable English is not easy.

And writing clear, precise, readable CLCE requires

(a) the ability to write clear, precise, readable English, and

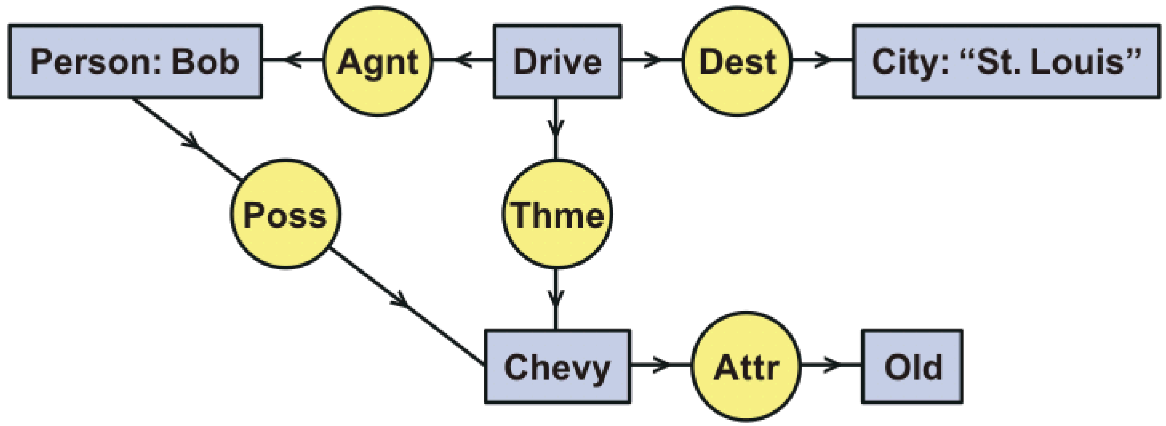
(b) some understanding of logical principles.

Of these two requirements, (a) is hard to find, even among people who have taken a course in (b).

But with good tools, CLCE or other controlled languages can be a useful aid for domain experts who need to translate their ideas to a computable form.

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

Conceptual graph display form:



Conceptual Graph Interchange Format (CGIF):

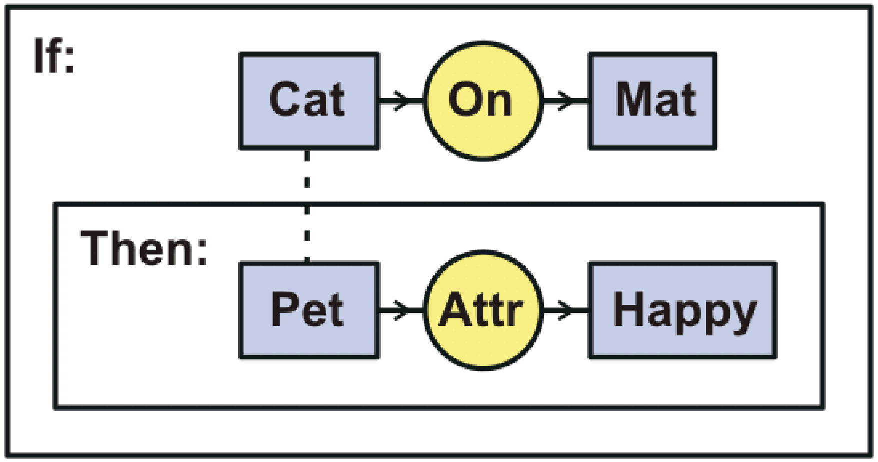
```
[Drive *x] [Person Bob] [City "St. Louis"] [Chevy *y] [Old *z]
(Agnt ?x Bob) (Dest ?x "St. Louis") (Thme ?x ?y) (Poss Bob ?y)
(Attr ?y ?z)
```

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))
```

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

A Logically Equivalent Variation

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:

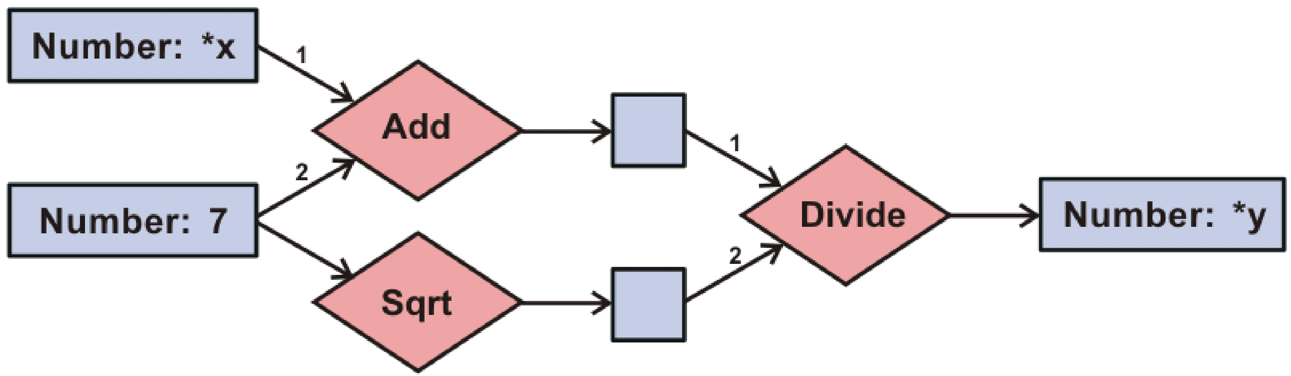
```
(forall ((x Cat) (y Mat))
   (if (On x y)
       (and (Pet x) (exists ((z Happy)) (Attr x z)))))
```

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

For common variations such as these, the proof of equivalence can be done in polynomial or even linear time.

CLCE: For a number x, a number y is ((x+7) / 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

Although Common Logic has a first-order semantics, it does permit 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)))
```

Defining New Words in CLCE

Although CLCE supports the full semantics of Common Logic, the word “relation” is not a reserved word.

But CLCE allows new words to be defined by their mapping to CLIF, CGIF, or other languages, such as SQL or OWL.

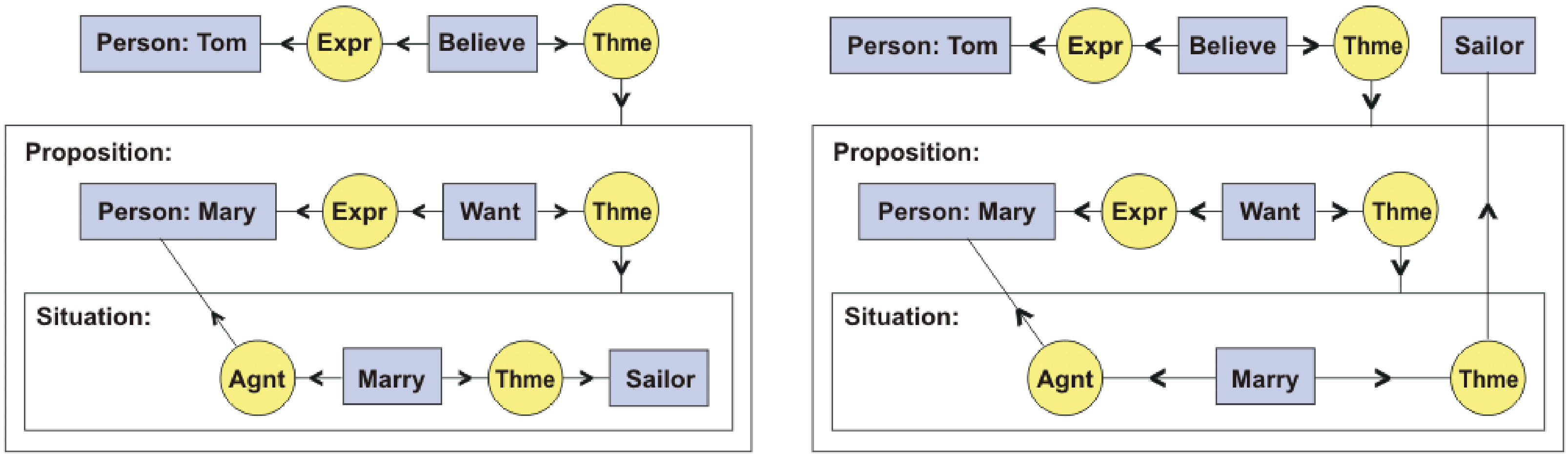
Define "familial relation r" as (and (Familial r) (Relation r)).

Define "relation r between x and y" as (and (Relation r) (r x y)).

With these definitions, the following CLCE sentence can be translated to the CLIF and CGIF sentences in the previous slide:

> **There is a familial relation between Bob and Sue.**

Extensions for Metalevel Reasoning



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

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

Sentences about propositions and situations involve metalevel language about language.

Metalanguage is also necessary to support Cyc, SBVR, and other systems that reason about propositions and situations.

The IKL Extension to Common Logic

A proposed extension to CL, called IKL, introduces propositions as formal entities that are expressed by sentences in CL or IKL.

Following is the CGIF notation for the CG on the left side 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)])])
```

To represent the CG on the right of the previous slide, move the concept node [Sailor: *x4] in front of the concept [Person: Tom].

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
```

A Knowledge Language for Interoperability

Common Logic is a superset of the logics used in many semantic systems, but some systems use even more expressive logics.

To design IKL, several groups, including Cyc, collaborated to find a minimal extension to CL that could support all the requirements.

The combination of IKL metalevels with the CL quantification over relations can support a very wide range of semantic features.

For the IKL extension to CL semantics, see
http://www.ihmc.us/users/phayes/IKL/SPEC/SPEC.html

For discussion of the example, *Tom believes that Mary wants to marry a sailor,* and its translation to CGIF and CLIF, see
http://www.jfsowa.com/pubs/cg4cs.pdf

For the use of metalanguage to represent modal logic, see
http://www.jfsowa.com/pubs/laws.htm

4. CNLs as a Bridge

Natural languages evolved to express and support human ways of thinking.

Computer languages enable IT professionals to think about the data and operations inside the computer system.

Forcing subject matter experts (SMEs) to think about their own subject in computer terms is counterproductive:

- Some of them become bad IT professionals.
- Some become good IT professionals, but compromise and distort their intuitions about their own subject.
- Very few become equally good at both.

CNLs can form a bridge between the NL of the subject and the computational requirements for precision.

Translating English to Controlled English

Requires translators with some training and good tools:

- A basic knowledge of the subject and its terminology.
- Some training in logic and the ability to write clear English.
- The tools can provide dictionaries for the subject terminology.
- They can help the translator stay within the controlled dialect.

Not necessary for the translator from English to controlled English to be an expert in the subject matter.

But it’s essential for a subject matter expert to read and verify the translation to controlled English.

With typical computer languages, the translator is rarely, if ever, a SME and the verification step is impossible.

An Example of Medical English

Find the percentage of patients with AMI2 receiving persistent beta blockers (for 135 of 180 days following discharge).

Numerator: Of patients in denominator, those prescribed a beta blocker following date of discharge with supply for at least 135 of next 180 days

Denominator: Age >= 35 years. All AMI cases except those transferred to another facility during the hospitalization.

Exclude patients with a history of Asthma, COPD3, Hypotension, Bradycardia (heart block > 1st degree or sinus bradycardia) or prescription of inhaled corticosteroids.

CLCE Definitions

Define "the age of x" as (– (Year CurrentDate) (Year (DoB x))).

Define "x is at least y" as (ge x y).

Define "x is transferred" as (Transferred x).

Define "AMI case" as "patient who has AMI".

Define "x is prescribed y on z for w days" as (Prescribed x y z w).

Define "x is a y" as (Type x y).

Define "x is discharged on y" as (DateOfDischarge x y).

Define "x is after y" as (gt x y).

Define "x has a history of y" as (HistoryOf x y).

Define "Bradycardia x" as
(or (and (HeartBlock x) (gt (Degree x) 1)) (SinusBradycardia x)).

Define "x is excluded" as (Excluded x).

Translation from English to CLCE

The denominator D is the set of every AMI case x where the age of x is at least 35, and x is not transferred;

The numerator N is the set of every patient x in the denominator D where x is prescribed a drug y on date z for w days, and y is a beta blocker, and x is discharged on z2, and z is after z2, and w is at least 135;

If a patient x has a history of asthma, or x has a history of COPD3, or x has a history of hypotension, or x has a history of bradycardia, or (x is prescribed a drug y, and y is inhaled, and y is a corticosteroid), then x is excluded;

The ratio R is (the count of every patient in the numerator N who is not excluded) divided by (the count of every patient in the denominator D who is not excluded).

Translation from CLCE to CLIF

```
(Set D)  (Set N)  (Set N1)  (Set D1)  (Number R)

(forall ((x Patient))
        (if (and (Has x AMI) (not (Transferred x)
                 (ge (– (Year CurrentDate) (Year (DoB x))) 35) ))))
            (In x D) )))

(forall ((x Patient))
        (if (and (In x D) (exists ((y Drug) (z Date) (w Number))
                                   (and (Prescribed x y z w) (Type y BetaBlocker)
                                        (DateOfDischarge x z2) (gt z z2) (ge w 135) )))
            (In x N) ))

(forall ((x Patient))
        (if (or (HistoryOf asthma x) (HistoryOf COPD3 x)
                (HistoryOf hypotension x) (HistoryOf Bradycardia x)
                (exists ((y Drug) (z Date) (w Number))
                        (and (Prescribed x y z w) (Inhaled y) (Type y Corticosteroid)) )))
            (Excluded x) ))

(and (forall ((x Patient))
             (if (and (In x N) (not (Excluded x))) (In x N1) ))
     (forall ((x Patient))
             (if (and (In x D) (not (Excluded x))) (In x D1) )))
     (= R (/ (Count N1) (Count D1)) )
```

Translation from CLCE to CGIF

```
[Set D]  [Set N]  [Set N1]  [Set D1]  [Number R]

[If  [Patient *x]  (has ?x AMI)  ~[ (transferred ?x) ]
     (year CurrentDate | *y1)  (DoB x | *y2)  (– ?y1 ?y2 | *a)  (ge ?a 35)
     [Then  (In ?x D) ] ]

[If  [Patient *x]  (in ?x D)  [Drug *y]  [Date *z]  [Number *w]  (Prescribed ?x ?y ?z ?w)
     [BetaBlocker ?y]  (DateOfDischarge ?x [Date *z2])  (gt ?z ?z2)  (ge ?w 135)
     [Then  (in ?x N) ] ]

[If  [Patient *x]
     [Either  [Or  (HistoryOf asthma ?x) ]  [Or  (HistoryOf COPD3 ?x) ]
              [Or  (HistoryOf hypotension ?x) ]  [Or  (HistoryOf Bradycardia ?x) ]
              [Or  [Drug *y]  [Date *z]  [Number *w]  (Prescribed ?x ?y ?z ?w)
                   (Inhaled ?y)  (Corticosteroid ?y) ] ]
     [Then  (excluded x) ] ]

[If  [Patient *x]  (In ?x N)  ~[ (Excluded ?x) ]
     [Then  (in ?x N1) ] ]
[If  [Patient *x]  (In ?x D)  ~[ (Excluded ?x) ]
     [Then  (in ?x D1) ] ]
[ (Count N1 | *x)  (Count D1 | *y)  (/ ?x ?y | R) ]
```

Choice of Representation

Two critical design points:

1. Translator interface: Mapping English to controlled English.

2. IT interface: Choosing the machine-oriented representations.

Options at the first interface are determined by subject matter experts and their colleagues.

Options at the second interface are determined by the IT professionals and the hardware-software developers.

Ease of use and accurate translation requires professionals at both levels to collaborate in the design choices.

Computational Complexity

CLCE imposes no restriction on the expressive power of the logic.

But most practical applications (including this medical example) can be translated to an efficiently computable subset of logic.

Many tools and techniques can automatically

1. Check the subset of logic required for the CLCE statements,
2. Determine the computability for various kinds of problems,
3. Convert the statements to a suitable form for the application.

Computational complexity depends on the problem, not on the language in which it is expressed.

Subject matter experts should not discard or distort their familiar forms of expression because of concerns about computability.

See “Fads and Fallacies about Logic,” http://www.jfsowa.com/pubs/fflogic.pdf

5. Methodologies and Missed Opportunities

Good technology can lead to major breakthroughs.

But even the best technology requires good methodologies and tools to enable people to use it effectively.

This section shows some methodologies that have proved to be effective, but many more examples could be given.

Just as important as the successful examples are the missed opportunities that might have led to even greater success.

All the missed opportunities are still available.

New projects can take advantage of them.

Solving Problems Stated in English

Part of Project Halo, whose goal is to build a Digital Aristotle.

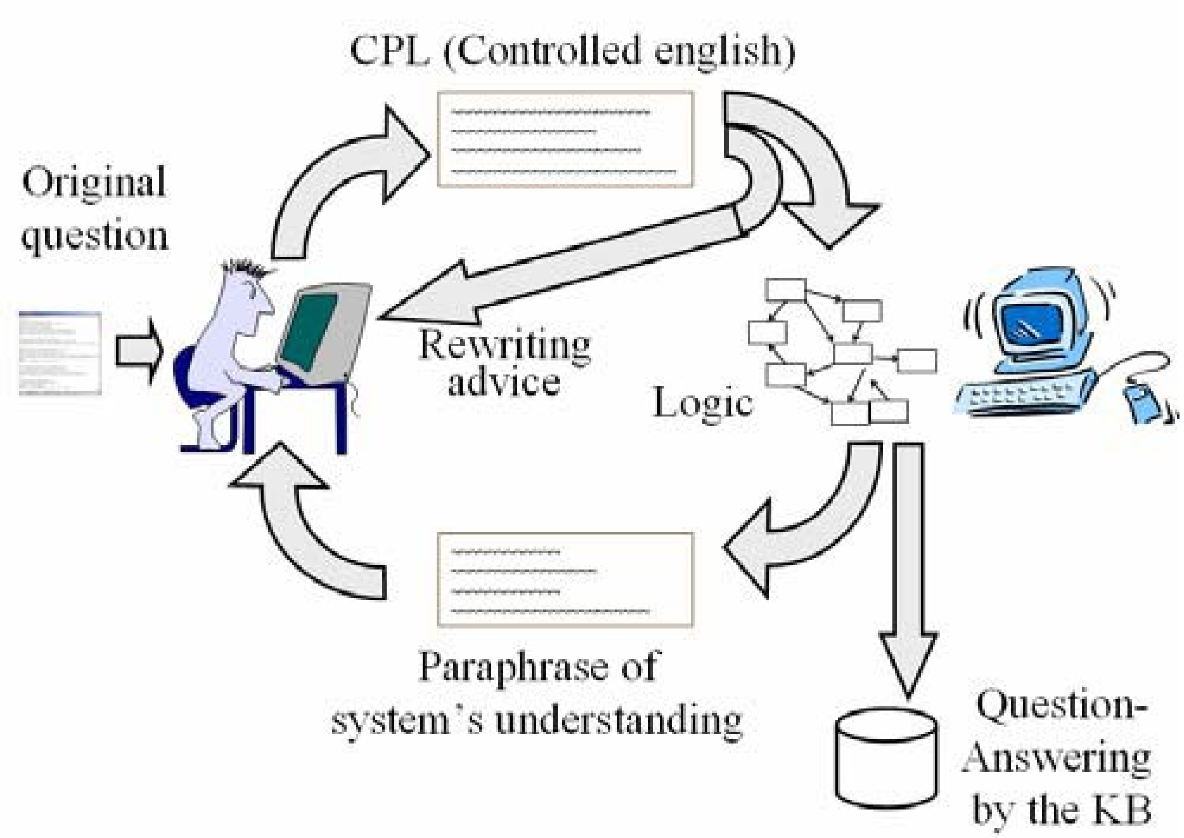
A question from the Advanced Placement Exam in physics:

> **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?**

Helping a Translator Map NL to CNL



From a paper by members of Project Halo: P. Clark, S-Y Chaw, K. Barker, V. Chaudhri, P. Harrison, J. Fan, B. John, B. Porter, A. Spaulding, J. Thompson, P. Z. Yeh, Capturing and Answering Questions Posed to a Knowledge-Based System, http://www.ai.sri.com/pubs/files/1547.pdf

Mean Tries to Completion (MTC)

Users often fail to stay within the limits of a controlled NL.

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

Clark et al. found that 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 the biology performance was so much better is that many of the questions could be approximated by simple queries of the form “What is an X?” or “What is the Y of X?”

Given the nature of the problems, these rates were not bad.

Business Rules in Retail

Tesco.com, a large Internet retailer, needed a flexible system that would allow employees to update business rules dynamically.

One vendor designed a system that would require Tesco employees to call an expert in RDF and OWL for every update.

Gerard Ellis, who had over ten years of R & D experience with conceptual graphs, designed and implemented a new system:

- **The internal knowledge representation was conceptual graphs.**
- **The interface for Tesco employees was controlled English.**
- **Tesco employees could extend or modify the rule base by typing the conditions and conclusions in controlled English.**
- **The system used the methodology of ripple-down rules to update the knowledge base, check for errors, and preserve consistency.**

See Qusai Sarraf and Gerard Ellis, "Business Rules in Retail: The Tesco.com Story," ***Business Rules Journal*****, Vol. 7, No. 6 (Jun. 2006), http://www.BRCommunity.com/a2006/n014.html**

Typical Business Rules

Rules in controlled English, automatically generated from statements in controlled English typed by Tesco employees:

- **If a television product description contains "28-inch screen", add a screen_size attribute_inches with a value of 28.**
- **a) If a recipe ingredient contains butter, suggest "Gold Butter" as an ingredient to add to the basket. b) If the customer prefers organic dairy products, suggest "Organic Butter" as an ingredient to add to the basket.**
- **If a customer buys 2 boxes of biscuits, the customer gets one free.**
- **If the basket value is over £100, delivery is free.**
- **If the customer is a family with children, suggest "Buy one family sized pizza and get one free".**

These rules were generated from a decision tree, as described on the next slide.

Ripple-Down Rules (RDR)

A methodology for subject matter experts to build and maintain a large, consistent rule base with little or no training:

- Internally, the rules are organized as a decision tree.
- Each link of the tree is labeled with one condition.
- Each leaf (end point) is labeled with a conclusion (or a conjunction of two or more conclusions).
- Any update that would create an inconsistency is blocked.
- If the update is consistent, the tree is automatically reorganized.
- For maximum performance, the decision tree can be compiled to a nest of if-then-else statements in a programming language.

For this application, the rules were represented in conceptual graphs, but they could be represented in any notation for logic.

See B. R. Gaines and P. Compton, Induction of Ripple-Down Rules Applied to Modeling Large Databases, http://pages.cpsc.ucalgary.ca/~gaines/reports/ML/JIIS95/index.html

Combination of CNL, RDR, and CGs

The three technologies have complementary strengths:

- **Controlled English: Readable by anyone who can read English and easier to write than most computer notations.**
- **Ripple-down rules: Consistent knowledge bases with thousands of rules can be developed by subject matter experts with no training in programming or knowledge engineering.**
- **Conceptual graphs: A dialect of Common Logic, which can serve as an intermediate notation between CNLs and other formalisms.**

Tesco applications that use this combination:

- **Manage product information for the electrical and wine departments.**
- **Provide product information to business affiliates.**
- **Create dynamic rule-based linkages between recipes, ingredients, and products.**

Applying RDR in Medicine

A medical application was the first to use the RDR methodology.

Compton et al.* described a larger pathology application, called Labwizard, which was in routine commercial use:

- **During a 29-month period, 16,000 rules were added.**
- **A total of over 6 million cases were interpreted by RDR rules.**
- **The total time for several different pathologists to add rules was 353 person hours – an average of 77 seconds per rule.**
- **The pathologists who used Labwizard and added new rules to it were very satisfied with its performance and functionality.**

For more detail, see

*** P. Compton, L. Peters, G. Edwards, T. G. Lavers, Experience with Ripple-Down Rules, http://www.cse.unsw.edu.au/%7Ecompton/publications/2005_SGAI.pdf**

Formal Concept Analysis (FCA)

A theory with supporting algorithms and methodology:

- **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 of computing the minimal lattice from a specification of concepts and attributes.**
- **Methodology. Techniques for describing concept types by attributes and using lattices for organizing ontologies and inference methods.**

Applications:

- **Ontology development and alignment; classification methods; machine learning; defining concepts used in other logics.**

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

A tutorial on FCA:
http://www.fbmn.fh-darmstadt.de/home/wolff/Publikationen/A_First_Course_in_Formal_Concept_Analysis.pdf

Combination of FCA and RDR

FCA and RDR have complementary strengths:

- FCA has the expressive power of Aristotle's syllogisms, and its algorithms automatically derive a complete and consistent lattice.
- Some description logics are more expressive than FCA, but harder to learn and harder to check for completeness and consistency.
- Ripple-down rules can use the concepts defined by FCA, and they can express logic that cannot be written in FCA alone.

Benefits of combining FCA and RDR:

- Reuse concepts defined by FCA methods in RDR rules.
- Use the semi-automated development methodologies of both.
- Support further combinations with CNLs, CGs, and other logics.

See D. Richards and P. Compton, Combining Formal Concept Analysis and Ripple Down Rules to Support the Reuse of Knowledge, http://en.scientificcommons.org/42976244

Cyc Project

Started in 1984 by Doug Lenat.

Name comes from the stressed syllable of 'encyclopedia'.

Goal: Implement the commonsense knowledge of an average human being.

After the first 20 years:

- 70 million dollars and 700 person-years of work,
- 600,000 concepts,
- Defined by 2,000,000 axioms,
- Organized in 6,000 microtheories,
- But not enough applications to support continued research.

In 2004, the Cyc project was scaled back, and more emphasis was placed on developing applications.

Lack of Methodology

For many years, Lenat refused to "dilute" the Cyc research by working on applications.

He claimed that a large knowledge base of common sense is a prerequisite for intelligence.

Major weakness: No clear definition of common sense.

Every field requires common sense: farming, aviation, truck driving, skiing, cooking, computer programming...

In each field, common sense depends heavily on context-dependent details of the subject matter.

The Cyc project developed a very large ontology, but there was no methodology for using it in practical applications.

Using Cyc as a Development Environment

Cyc is a good platform for defining ontologies, but many applications only require a small subset of the axioms.

Suitable tools can extract axioms from Cyc, translate them to other notations, and integrate them with other software.

In fact, some Cyc users developed tools for extracting axioms and tailoring them to other platforms.

See the "knowledge bus" in

Peterson, Brian J., William A. Andersen, & Joshua Engel (1998) Knowledge bus: generating application-focused databases from large ontologies, http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-10/

Missed Opportunity

In 1998, Lenat could have hired the trio that had designed the knowledge bus.

They could have developed Cyc as a tool for creating and exporting specially tailored knowledge bases.

Knowledge engineers and programmers could use Cyc to generate semantic applications to run on any platform:

- Deductive databases,
- Semantic interfaces to legacy systems,
- Rule-based systems,
- Applications for the Semantic Web.

Instead, Lenat did not approve of people extracting axioms from Cyc and using them on other platforms.

New Opportunities

The Cyc ontology is the world's largest body of knowledge represented in logic and suitable for detailed deduction.

Since 2004, Cycorp has focused on applications, including many that translate the ontology to other versions of logic.

Full CycL notation can be translated to IKL and controlled English. Subsets can be translated to RDF and OWL.

The OpenCyc Foundation has also made the Cyc ontology and some applications available under the Apache license.

For white papers and research publications about Cyc, see http://cyc.com

To browse the ontology or to download OpenCyc, see http://opencyc.org/

New Methods of Indexing

Relational databases use indexes to reduce search.

Novel methods have also been developed for indexing databases of graphs, especially chemical graphs.

Similar techniques could be applied to arbitrary graphs from any source.

But languages that expose the physical formats are premature optimizations that can block innovation.

See "Mining Patents Using Molecular Similarity Search," by James Rhodes, Stephen Boyer, Jeffrey Kreulen, Ying Chen, & Patricia Ordonez
http://psb.stanford.edu/psb-online/proceedings/psb07/rhodes.pdf

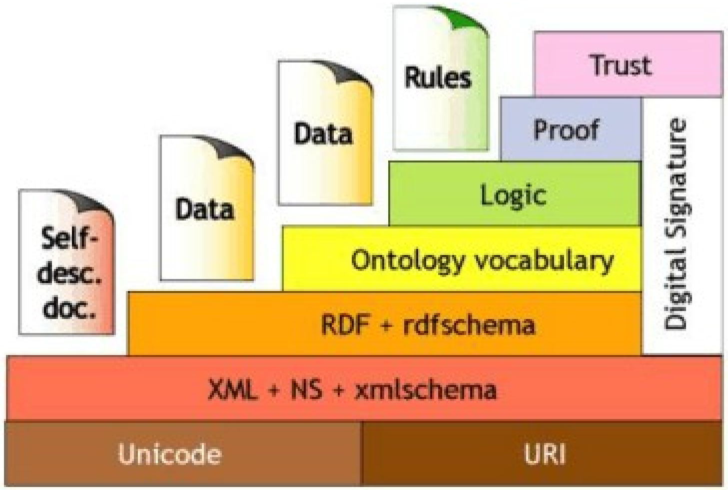
Hadoop DB

A combination of Google's MapReduce algorithm, available in the Hadoop software, with an API from PostgreSQL.

- **A relational view of large amounts of widely distributed data.**
- **High performance and scalability on parallel hardware.**
- **Freely available as open-source software.**
- **An illustration that the logical view of data is independent of the physical format, location, and representation.**
- **An example of the kinds of innovation that are blocked by linking the logical view to a particular physical format or representation.**

See "HadoopDB: An Architectural Hybrid of MapReduce and DBMS Technologies for Analytical Workloads," Azza Abouzeid, Kamil Bajda-Pawlikowski, Daniel J. Abadi, Avi Silberschatz, Alex Rasin. *Proceedings of VLDB*, 2009. http://db.cs.yale.edu/hadoopdb/hadoopdb.pdf

Semantic Web



The original “layer cake” diagram embodied many good ideas.

But building semantics on top of syntax was not one of them.

The semantics of first-order logic has not changed in 130 years.

But there are endless arguments (good and bad) over syntax.

Missed Opportunity

When the Semantic Web appeared, most commercial web sites, large and small, were built around relational databases.

In fact, the acronym LAMP characterized small and medium sites: Linux, Apache, MySQL, and Perl, Python, or PHP.

The Semantic Web had a great opportunity to provide an upward compatible semantics for both relational and object-oriented DB:

- Type hierarchies defined by description logics.
- Query and constraint languages based on typed FOL.
- Deductive database tools based on a typed version of Datalog.
- Arbitrary n-tuples to support relational tables.
- A common logic-based semantics for all semantic systems.

Such an approach is still possible as an upward compatible extension of the current Semantic Web technology.

Integrating CNLs with Graphics

A picture is sometimes worth a thousand words.

But sometimes, a single word can clarify a thousand pictures.

Different people have different preferences for thinking in terms of words or pictures, but everybody can benefit from both.

Many kinds of diagrams and methodologies for using diagrams have been developed for computer systems.

An integrated semantic system should

- **Support the ways that people talk about their subject.**
- **Support the diagrams they find useful to supplement that talk.**
- **Integrate the diagrams and the words as complementary forms for representation, communication, and thinking.**

A Migration Path to the Future

Any declarative notation, graphic or linear, can be mapped to some version of logic.

Common Logic is upward compatible with nearly all the systems mentioned so far. Other systems, such as Cyc and SBVR, can be supported by extensions to Common Logic, such as IKL.

Good development methodologies supported by controlled natural languages can be used effectively by domain experts.

Recommendation:

- **Integrate all systems, including legacy systems, with a common semantic foundation.**
- **Use methodologies that let domain experts update and extend the knowledge base without depending on IT specialists.**

6. Unrestricted Natural Languages

Semantic systems recognize, represent, and respond to the meaning and purpose of the data they process.

But any particular meaning can be expressed in many different languages, notations, or diagrams.

Natural languages are the normal means for people to express their meanings, but they are not the most computable.

Full natural language understanding involves unsolved research problems, but there are many useful ways of processing NLs short of total understanding.

This section surveys the issues and prospects for expressing and using more of the semantics of natural languages.

The Flexibility of Natural Languages

The languages we speak today are based on the syntax and vocabulary of our stone-age ancestors.

In fact, some stone-age tribes and their languages have adapted to modern culture and technology within a single generation.

Natural languages have the ability to grow, evolve, and accommodate every aspect of every human life.

What makes natural languages so powerful and expressive, yet computable by a brain that weighs less than 2 kilograms?

Can we ever design or simulate a system as flexible?

What kinds of knowledge representation are necessary?

What reasoning methods can processes those representations?

Mapping Language to Formal Logic

Impossible except under tightly controlled conditions:

- Unambiguous grammar.
- Consistent use of each word in exactly one word sense (or a small number of predefined word senses that can be distinguished by context).
- A formal ontology that defines each word sense.

These conditions hold for the specially designed, controlled natural languages discussed in Section 1.

They almost never hold for the kind of language used by people in talking, writing, or twittering to other people.

Limits of Logic

Alfred North Whitehead, *Modes of Thought* :

“Both in science and in logic, you have only to develop your argument sufficiently, and sooner or later you are bound to arrive at a contradiction, either internally within the argument, or externally in its reference to fact.”

“The topic of every science is an abstraction from the full concrete happenings of nature. But every abstraction neglects the influx of the factors omitted into the factors retained.”

“The premises are conceived in the simplicity of their individual isolation. But there can be no logical test for the possibility that deductive procedure, leading to the elaboration of compositions, may introduce into relevance considerations from which the primitive notions of the topic have been abstracted.”

Formal and Informal Methods

The formal methods of mathematics and logic have been spectacularly successful in applying the knowledge generated by science.

But every application of the general laws of science to particular problems requires domain-dependent approximations.

There will never be a universal, exception-free ontology until every question in every branch of science (natural and social) is answered.

Every answer raises many more questions that are even harder to answer.

Even a single application may require multiple, inconsistent approximations for different aspects of the same project.

Formal and informal methods are complementary.

The human brain processes both kinds of methods together.

Can any artificial system be as flexible?

How?

Ludwig Wittgenstein's Development

Early philosophy: ***Tractatus Logico-Philosophicus*** **(1921).**

- **Single system of logic and ontology with a single proof theory.**
- **Foundation for classical artificial intelligence.**

Transitional period: ***Philosophical Remarks*** **(1929-1930).**

- **Multiple sentence systems (Satzsysteme),**
- **Each with multiple proof theories (Beweissysteme).**
- **Foundation for logics that go beyond classical FOL.**

Later philosophy: ***Philosophical Investigations*** **(1952).**

- **Multiple language games integrated with every aspect of life.**
- **Foundation for unrestricted natural language.**

Defaults, Exceptions, Unknowns, Vagueness, Fuzziness, and Uncertainty

Early Wittgenstein, as influenced by Frege and Russell:

- “Everything that can be said can be said clearly.”
- “Whereof one cannot speak, thereof one must be silent.”

Transitional Wittgenstein:

- Distinguish the language (Satzsystem) from the proof theory.
- The same language can be used with different proof theories.

For nonclassical reasoning:

- Modified proof theory for defaults, exceptions, and unknowns.
- Metalanguage for reasoning about the vagueness, fuzziness, and uncertainty in the way language relates to the world.

Limits of Definability

Immanuel Kant:

"Since the synthesis of empirical concepts is not arbitrary but based on experience and as such can never be complete... only arbitrarily made concepts can be defined synthetically.... This is the case with mathematicians."

Ludwig Wittgenstein's family resemblance:

Empirical concepts cannot be defined by a fixed set of necessary and sufficient conditions. Instead, they can only be taught by giving a series of examples and saying "These things any anything that resemble them are instances of the concept."

Friedrich Waismann's open texture:

For any proposed definition of empirical concepts, new instances will arise that "obviously" belong to the category but are excluded by the definition.

Note: Folksonomies consist entirely of "undefinable" concepts.

Language Games

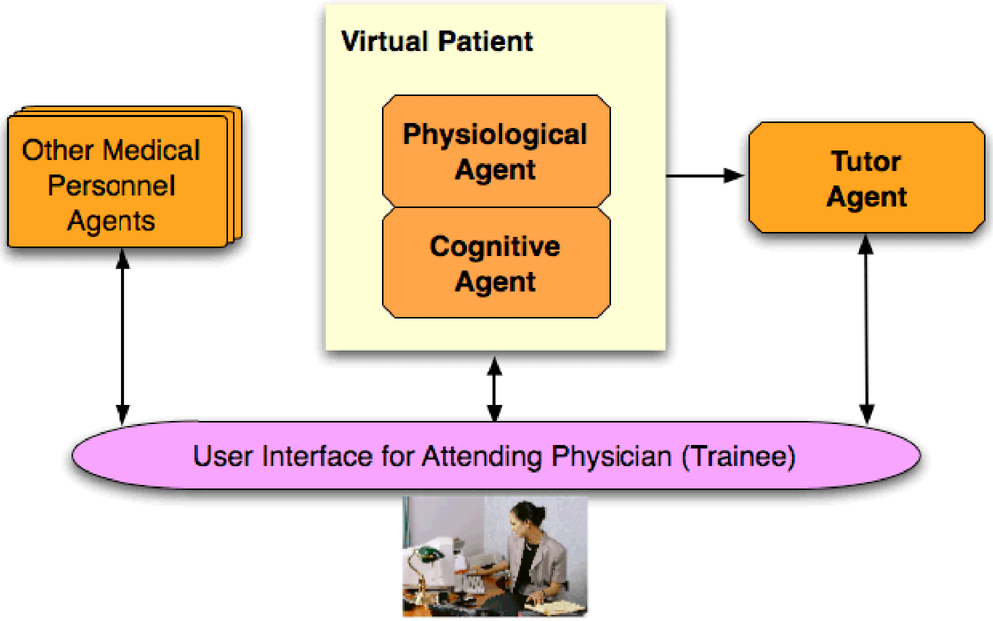
In his later philosophy, Wittgenstein realized that logical assertions and proofs are just one way of using language.

He used the term *language game* (Sprachspiel) for the open ended number of ways people use language.

Examples:

> **“Giving orders, and obeying them; describing the appearance of an object, or giving its measurements; constructing an object from a description (a drawing); reporting an event; speculating about an event; forming and testing a hypothesis; presenting the results of an experiment in tables and diagrams; making up a story, and reading it; play acting; singing catches; guessing riddles; making a joke, telling it; solving a problem in practical arithmetic; translating from one language into another; asking, thanking, cursing, greeting, praying.”**

Maryland Virtual Patient



The MVP system simulates a patient who carries on a dialog with a medical student who tries to diagnose the patient's disease.

The student asks questions in unrestricted English, and MVP generates responses in a version of controlled English.

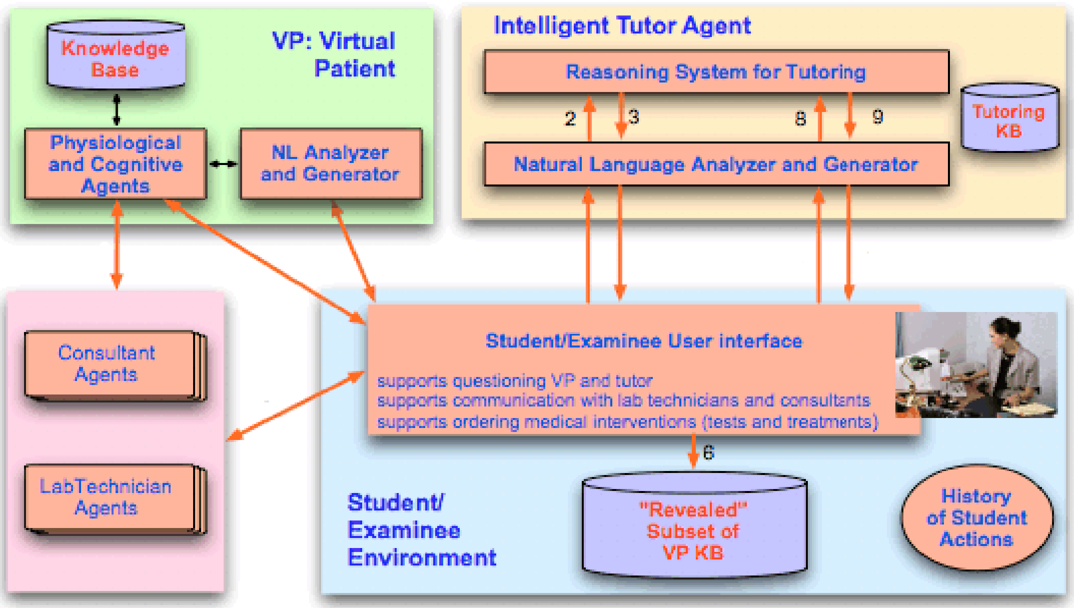
Source: Maryland virtual patient: a knowledge-based, language-enabled simulation and training system, by M. McShane, S. Nirenburg, B. Jarrell, S. Beale, & G. Fantry, http://bams.cm-uj.krakow.pl/bams3_pdf/bams%209.pdf

A Dialog with MVP

A medical student diagnoses an MVP "patient" named Mr. Wu:

Student:	So you have difficulty swallowing?
Mr. Wu:	Yes.
Student:	Do you have difficulty swallowing solids?
Mr. Wu:	Yes.
Student:	Liquids?
Mr. Wu:	No.
Student:	Do you have chest pain?
Mr. Wu:	Yes, but it's mild.
Student:	Any heartburn?
Mr. Wu:	No.
Student:	Do you ever regurgitate your food?
Mr. Wu:	No.
Student:	How often do you have difficulty swallowing?
Mr. Wu:	Less than once a week.
Student:	It is too early to take any action. Please come back in 9 months.
Mr. Wu:	OK.

Multiple Simulated Agents



Students talk with a patient, tutor, consultant, or lab technician.

All dialogs use the same ontology and knowledge base.

But each type of dialog is based on a different language game.

Source: Adaptivity in a multi-agent clinical simulation system, by S. Nirenburg, M. McShane, S. Beale, & B. Jarrell, http://www.cis.hut.fi/AKRR08/papers/nirenburg.pdf

MVP Syntax, Semantics, and Pragmatics

Two kinds of syntax:

- User inputs: A large English grammar that imposes very few restrictions on what the users can say.
- MVP responses: Controlled English, tailored to the subject matter.

Two kinds of semantics:

- Lexical semantics: Patterns of concept types and the expected relations among them. No detailed definitions or constraints.
- Subject matter: Detailed ontology, definitions, rules, constraints, and background knowledge about each disease and therapy.

Pragmatics tailored to each type of dialog:

- Different goals, speech acts, and language games.

A balanced combination of state-of-the-art technologies.

Learning by Reading

The ultimate goal of automated knowledge acquisition:

- Let the computer learn by reading a book.

But what would the computer learn?

- A formal ontology of the subject matter in the book?
- Sufficient knowledge about the subject to answer questions?
- An improved ability to read other documents?

How would that knowledge be represented?

- A collection of bits and pieces of information?
- Some updates and extensions to its previous knowledge?
- Inferences derived from the new knowledge?

VivoMind Language Processor (VLP)

A system that learns by reading:

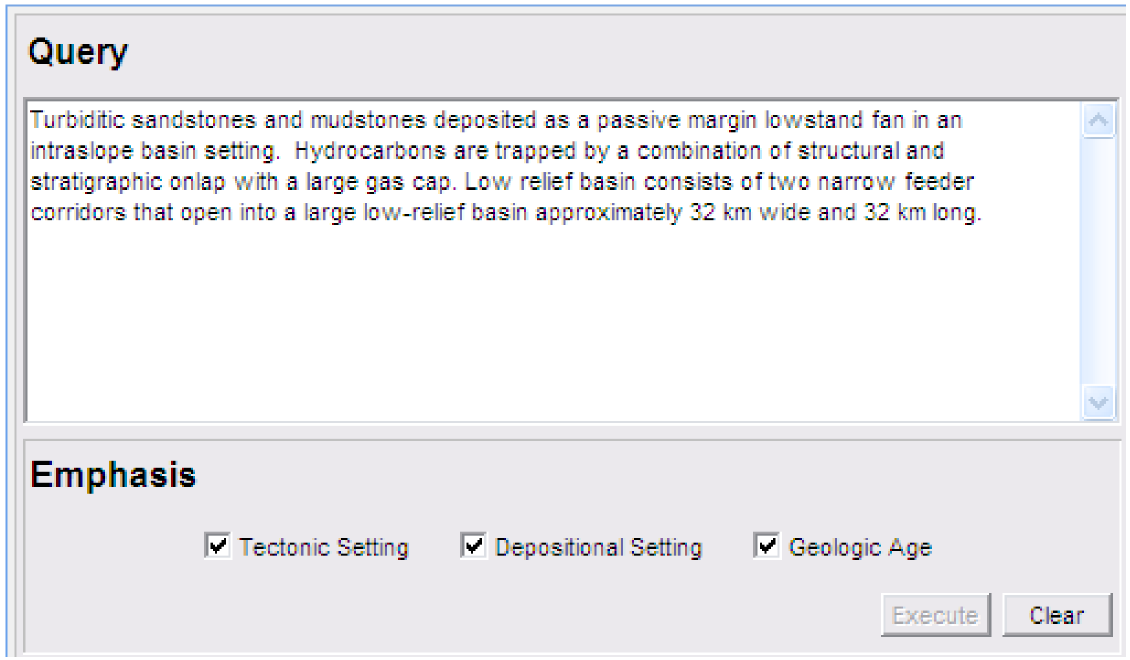
- 79 documents about the geology of oil and gas fields.
- English, as written for human readers (no semantic tagging).
- Additional data from relational DBs and other structured sources.
- Basic VivoMind ontology plus a domain-dependent ontology written in controlled English by geologists at the University of Utah.
- Very few detailed axioms in the ontology.

After reading, VLP answers questions by geologists:

- Input: Description of a geological site in unrestricted English,
- Query: Find, compare, and rank all similar sites in the documents.

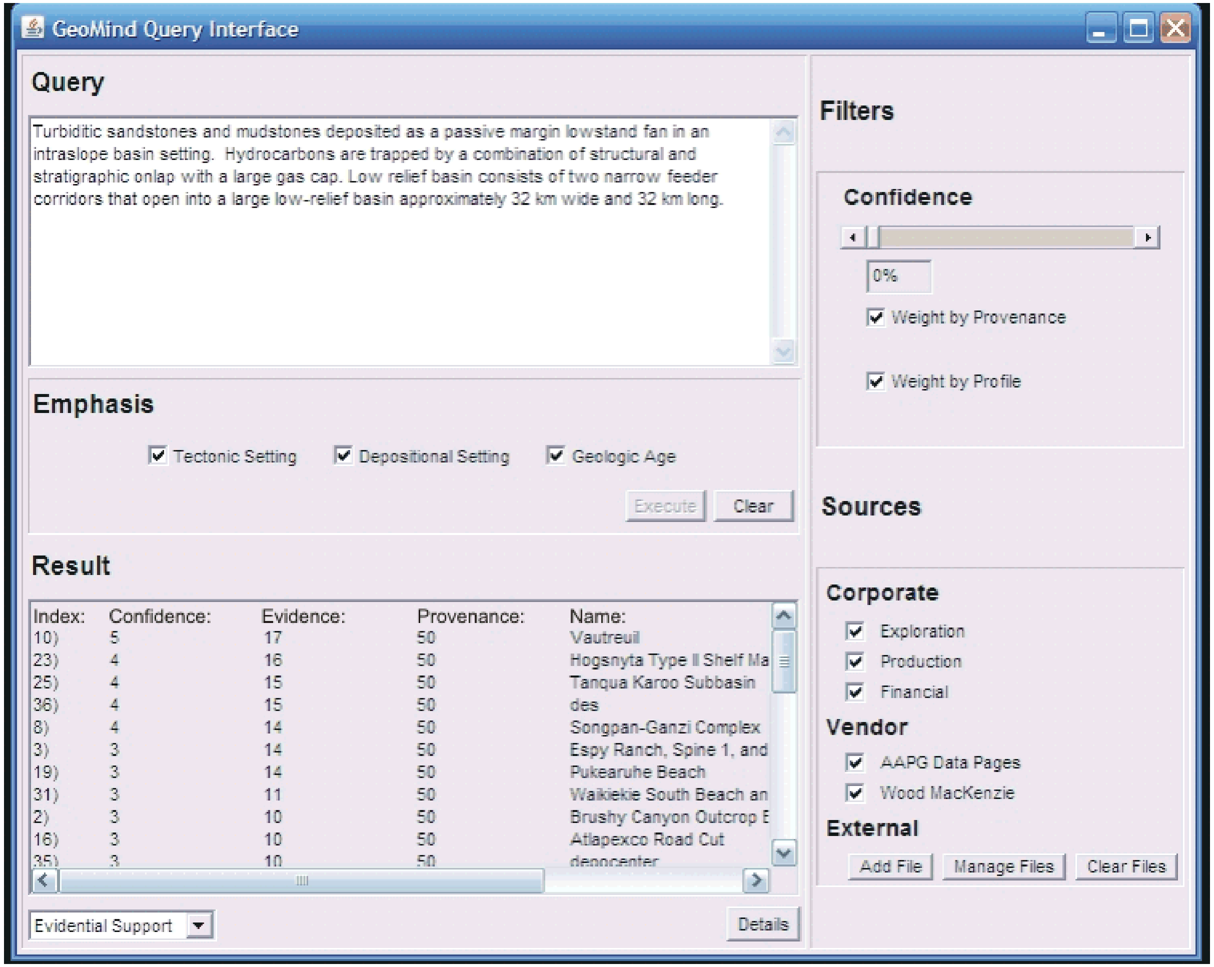
See Two paradigms are better than one, and multiple paradigms are even better, by A. K. Majumdar and J. F. Sowa, http://www.jfsowa.com/pubs/paradigm.pdf

A Query Written by a Geologist

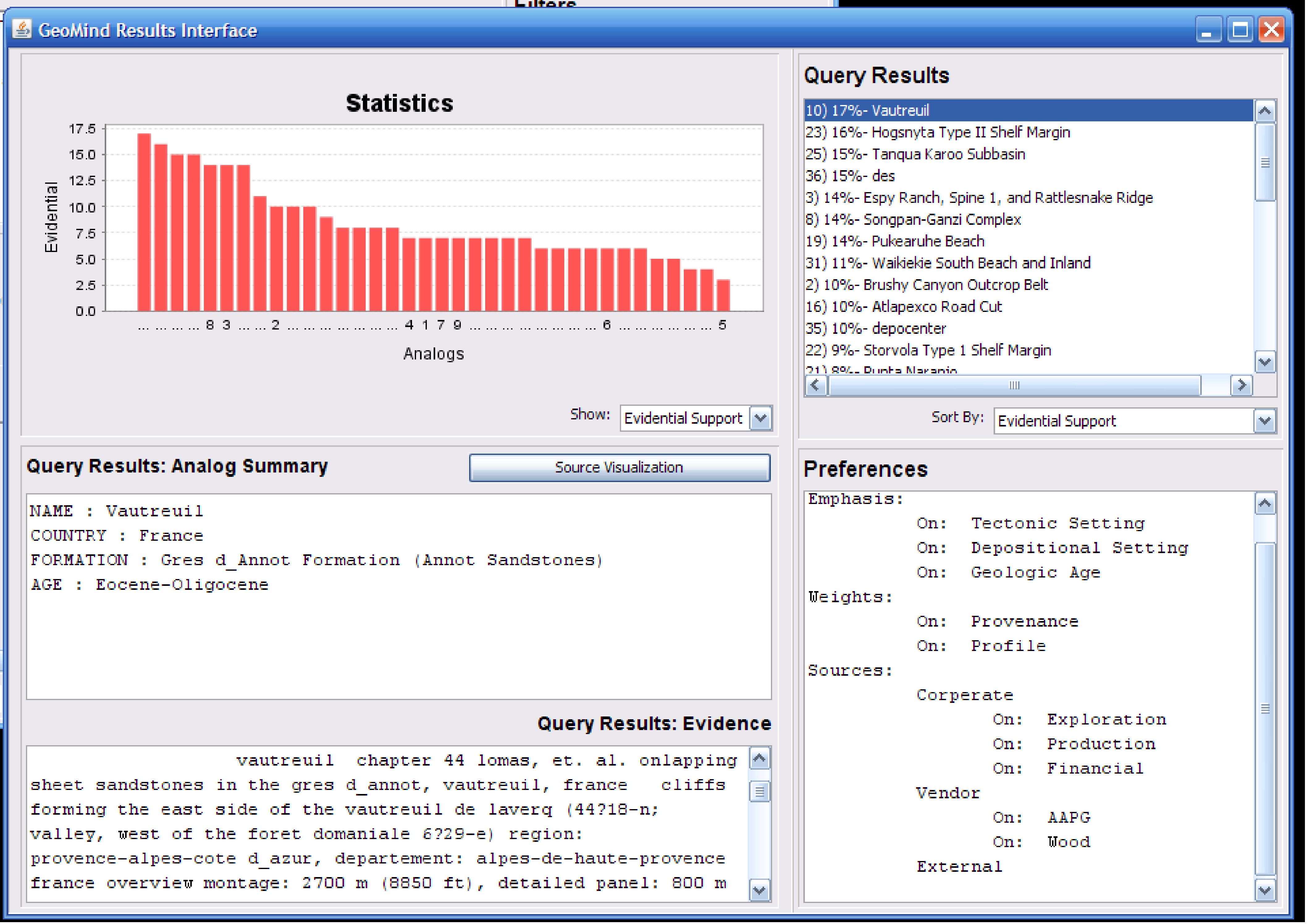


Turbiditic sandstones and mudstones deposited as a passive margin lowstand fan in an intraslope basin setting. Hydrocarbons are trapped by a combination of structural and stratigraphic onlap with a large gas cap. Low relief basin consists of two narrow feeder corridors that open into a large low-relief basin approximately 32 km wide and 32 km long.

Similar Sites Found in the Documents



Sites are ranked by evidence (Dempster-Shafer) and confidence factors.



After clicking the “Details” button on the previous window

Using Multiple Knowledge Sources

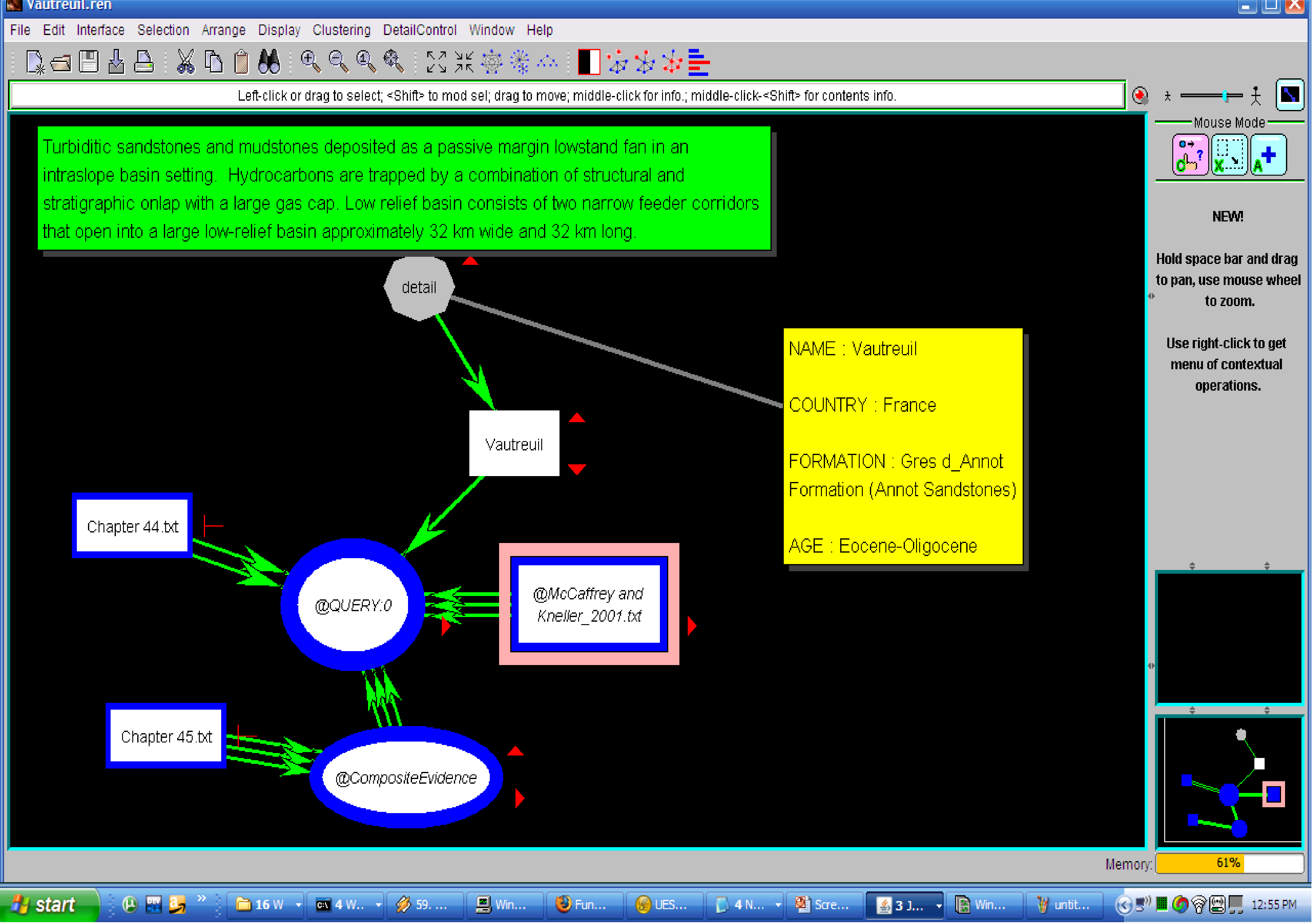
The geologists who wrote an ontology in controlled English included very little domain knowledge.

But that simple ontology was sufficient for the system to acquire more knowledge from a textbook of geology.

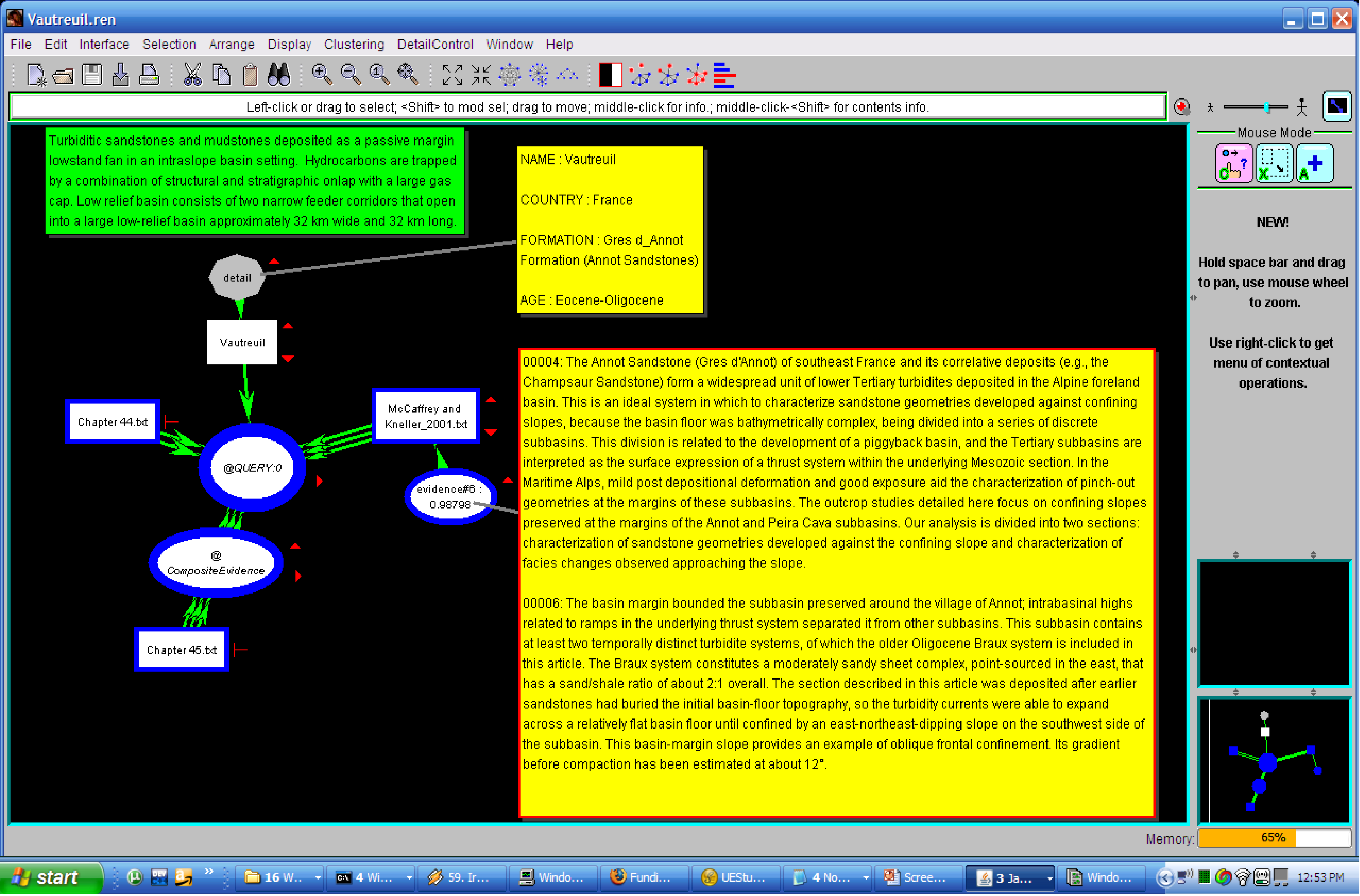
The following screen shot shows that the system accessed parts of four documents to respond to this query:

- A description of an oil field in the Vautreil region of France,
- Chapters 44 and 45 from a textbook on geology,
- A research paper by McCaffrey and Kneller (2001).

But the system also used other documents to evaluate the less highly ranked sites.



Top-level source visualization and the highest-ranked result



Show which paragraphs of a document were used for this query

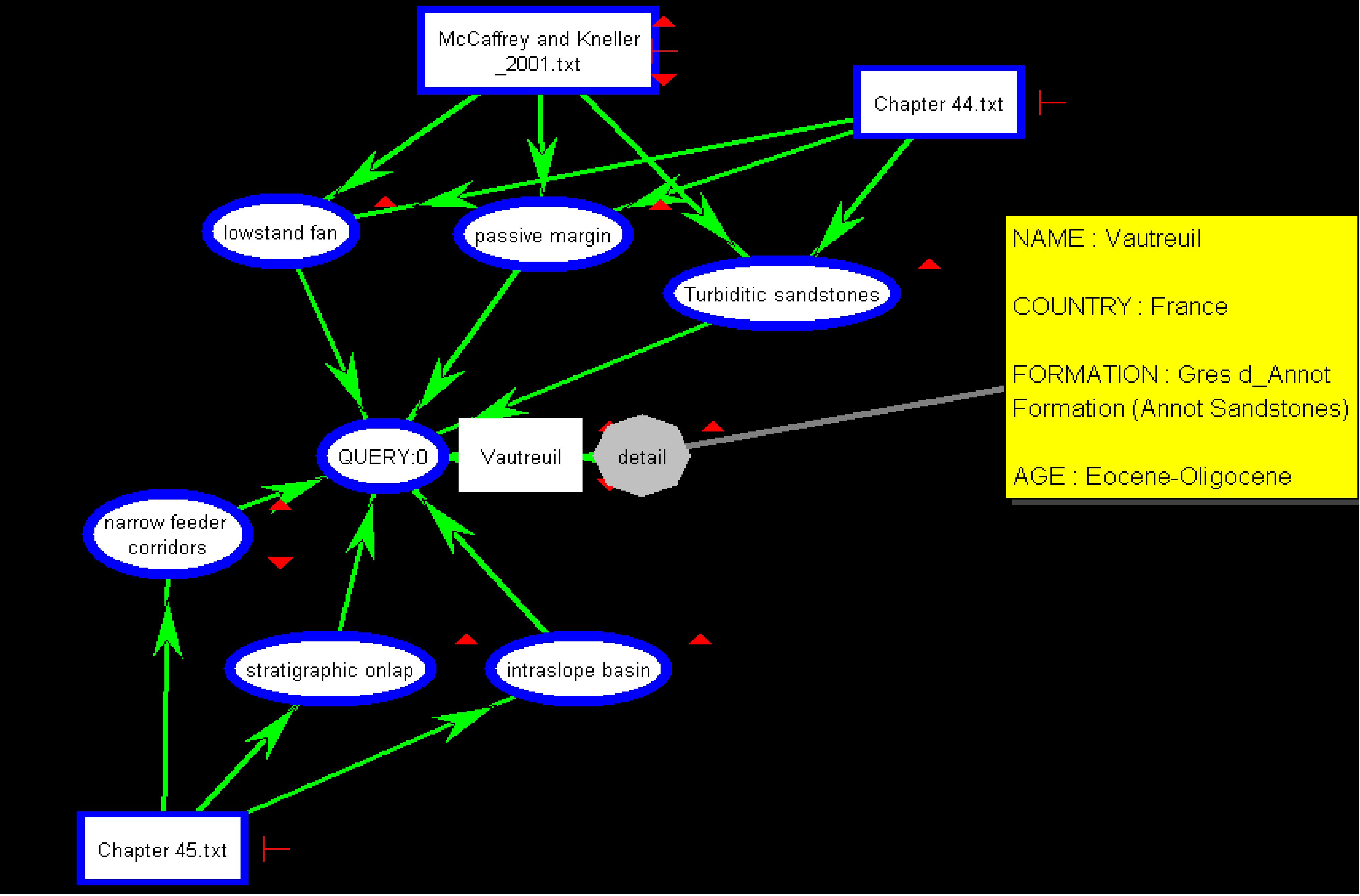
Showing Details of the Analysis

Before answering questions, the system translates all sentences of the source documents to conceptual graphs and indexes them by their Cognitive Signatures™.

Conceptual graphs derived from the query may trigger searches for CGs that provide background knowledge.

The next screen shot expands the previous screen to show which phrases of the query required additional knowledge.

For each document that was accessed, the geologist can ask for the specific sentences or paragraphs that were analyzed to generate the answer.



Drill down into the query and its relationships to the source documents

Emergent Knowledge

When reading the 79 documents,

- VLP translates the sentences and paragraphs to CGs.
- But it does not do any further analysis of the documents.

When a geologist asks a question,

- The VivoMind system may find related phrases in many sources.
- To relate those phrases, it may need to do further searches.
- The result is a conceptual graph that relates the question to multiple passages in multiple sources.
- Some of those sources might contribute information that does not have any words that came from the original question.
- That new CG can be used to answer further questions.

By a "Socratic" dialog, the geologist can lead the system to explore novel paths and discover unexpected patterns.

Cautionary Notes by Logicians and Poets

Alfred North Whitehead:

"Human knowledge is a process of approximation. In the focus of experience, there is comparative clarity. But the discrimination of this clarity leads into the penumbral background. There are always questions left over. The problem is to discriminate exactly what we know vaguely."

Charles Sanders Peirce:

"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."

Robert Frost:

"I've often said that every poem solves something for me in life. I go so far as to say that every poem is a momentary stay against the confusion of the world.... We rise out of disorder into order. And the poems I make are little bits of order."

Alfred North Whitehead:

"We must be systematic, but we should keep our systems open."

Some Observations

History of natural language processing:

- 1949: Warren Weaver wrote a highly influential memo to suggest that computers could be used to translate natural languages.
- 1950s: Early research on machine translation (MT).
- 1963: Research terminated on the GAT translator, which was converted to a commercial system for MT called SYSTRAN.
- 1970s: Logic-based natural language query systems, such as TQA.
- 1980s: New logic-based semantic systems: Montague grammar, Discourse Representation Theory, Situation Semantics, conceptual graphs...
- 1990s: Statistical processing of language becomes popular.
- 2000s: SYSTRAN is still one of the most widely used MT systems.

Pure logic-based NLP systems are useful for controlled NLs, but they have been fragile and inflexible for unrestricted NLs.

But systems that combine formal and informal methods, such as MVP and VLP, have been more robust and practical.

7. Integrating All Semantic Systems

Semantic systems recognize, represent, and respond to the meaning of the data and the goals of the users.

But any particular meaning can be expressed by equivalent statements in different languages, notations, and diagrams.

The proliferation of incompatible semantic systems is a scandal that is strangling the growth of the entire field.

The oldest legacy systems can interoperate more easily than two modern systems based on different logics or ontologies.

Semantics should facilitate the interoperability and integration of all systems, legacy or modern, independently of the language, ontology, or methodology with which they were implemented.

Two Views of Legacy Systems

By the philosopher 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."

By the computer scientist Harlan Mills:

"OS/360 is like a cow." It's not the most beautiful or efficient, and many people think they can design a better one. But if you put hay and water in one end, you get fertilizer from the other end and milk from the middle. You can use it effectively if you recognize its limitations and remember which end is which.

Both views are true:

- Every system eventually becomes a legacy.
- But legacy systems can remain useful for a long time.
- They must be able to interoperate with semantic systems.

Classification

Distinct, but related classifications:

- Terminology. A list of terms (words and phrases) of some branch of science, art, engineering, business, medicine, or law.
- Taxonomy. A classification of types of entities – objects, properties, events, and relations – of some field.
- Ontology: A formal theory of the principles underlying some taxonomy.

These three classifications differ in precision and stability:

- Terminology. The most stable, but the least precise and theoretical.
- Taxonomy. A pretheoretical guide and prerequisite for a theory.
- Ontology. Most precise and detailed, but the most likely to change with new discoveries and inventions.

Terminologies are the basis for human communication.

They are also the key to interoperability with legacy systems.

Attempts to Force Compatibility

Legislate a universal, upper-level ontology:

- Design a top-down framework based on philosophical principles.
- Define the categories and axioms for all expected applications.
- Edict the use of that ontology for those applications.

Problems with legislating a universal ontology:

- Mismatch between elegant principles and messy low level details.
- Incompatibility with legacy systems that have no explicit ontology.
- Incompatibility with systems based on different ontologies.
- Incompatibility with new requirements for major revisions.
- Attempts to enforce a detailed upper level have been replaced with collections of inconsistent low-level modules or "microtheories."

Top-down principles are useful as optional guidelines, but they can be counterproductive as rigid requirements.

Limitations of Top-Down Legislation

Primary obstacle: too much detail in the specifications.

- **Each additional axiom increases the chance of inconsistency.**

Peirce's observations:

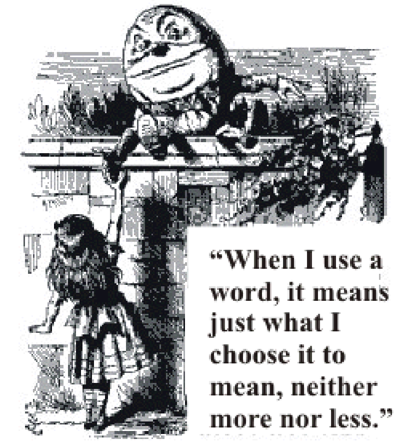
- **To be certain, "one has only to be sufficiently vague."**
- **Not so difficult "to be pretty precise and fairly certain at once about a very narrow subject."**

Recommendations:

- **Keep the upper-level ontology vague: delete unnecessary axioms.**
- **Focus on the specific tasks on which systems interoperate.**
- **Formalize only the minimum task-dependent input/output details.**

These guidelines are equally valid for old legacy systems and for the latest and greatest semantic systems.

Multiple Word Senses



The Humpty-Dumpty theory of word senses contains a large grain of truth.

Example:

- **The English word *give* is extremely ambiguous.**
- **Large unabridged dictionaries list dozens of word senses.**
- **No two dictionaries distinguish exactly the same word senses.**
- **Even professional lexicographers can't agree on the word senses.**
- **No computer program can reliably annotate the word senses.**
- **No robust NLP system should assume that any *a priori* annotation of a word sense is reliable.**

Conclusion:

- **It is pointless to specify word senses to a greater precision than the speaker would intend, expect, or understand.**

Technology for Controlled NLs

Two views about the nature of CNLs:

- **Bridge:** A controlled natural language is an intermediate step between a natural language and a formal notation for logic.
- **Formal:** A controlled natural language is a formal notation for logic.

Two ways of enforcing control:

- **Syntactic:** Strict constraints on the permissible grammar.
- **Semantic:** Strict constraints on the semantics, but an allowance for any grammatical pattern that has a unique semantic interpretation.

Advantages and disadvantages:

- **Syntactic:** More predictable, but harder for people to write.
- **Semantic:** More natural and easier to write, but requires an “echo” that shows exactly how each sentence is interpreted.

See Naturalness vs. Predictability: A Key Debate in Controlled Languages, by P. Clark, P. Harrison, W. R. Murray, & J. Thompson, http://www.cs.utexas.edu/users/pclark/papers/cnl09.pdf

Interfaces to Semantic Systems

All computer systems, including legacy systems, are becoming semantic systems that directly or indirectly access the WWW.

Different people require different interfaces:

- **Casual users – anybody who opens an unfamiliar application.**
- **Subject matter experts who are updating a knowledge base.**
- **IT professionals who must address the internal representations.**

The terminology of a subject is the key to interoperability:

- **SMEs are the people who know the subject matter.**
- **Their terminology is the basis for all communications about the subject among people and computer systems.**
- **Their primary interface must be a CNL that is automatically translated to and from any internal knowledge representations.**
- **The technology for doing those translations was a research topic thirty years ago, but it has been deployed in many practical systems today.**

Conclusions

The logicians Peirce, Whitehead, and Wittgenstein

- Had the highest regard for the precision of logic and mathematics,
- Observed that the starting assumptions for logic and ontology are based on prelogical insights expressed in ordinary language.

Semantic systems are based on the semantics of communication

- Among people in ordinary language,
- Between people and computers in ordinary language,
- Among computers in categories expressible in ordinary language.

Technology available today can

- Translate controlled NLs to and from computable notations,
- Find and extract significant patterns from unrestricted NLs.

Methodologies for developing and using semantic systems should take advantage of technology for NLs, controlled and unrestricted.

Related Readings

Fads and Fallacies About Logic, by J. F. Sowa,
http://www.jfsowa.com/pubs/fflogic.pdf

From existential graphs to conceptual graphs, by J. F. Sowa,
http://www.jfsowa.com/pubs/eg2cg.pdf

Two paradigms are better than one, but multiple paradigms are even better, by A. K. Majumdar & J. F. Sowa, http://www.jfsowa.com/pubs/paradigm.pdf

Pursuing the goal of language understanding, by A. K. Majumdar, J. F. Sowa, & J. Stewart, http://www.jfsowa.com/pubs/pursuing.pdf

Papers from a workshop on controlled natural languages,
http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-448/

Web site for controlled natural languages,
https://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