Controlled Natural Languages for Knowledge Representation

Rolf Schwitter

`Rolf.Schwitter@mq.edu.au`

Controlled Natural Languages for Knowledge Representation

Rolf Schwitter

Rolf.Schwitter@mq.edu.au

Controlled Natural Languages for Knowledge Representation

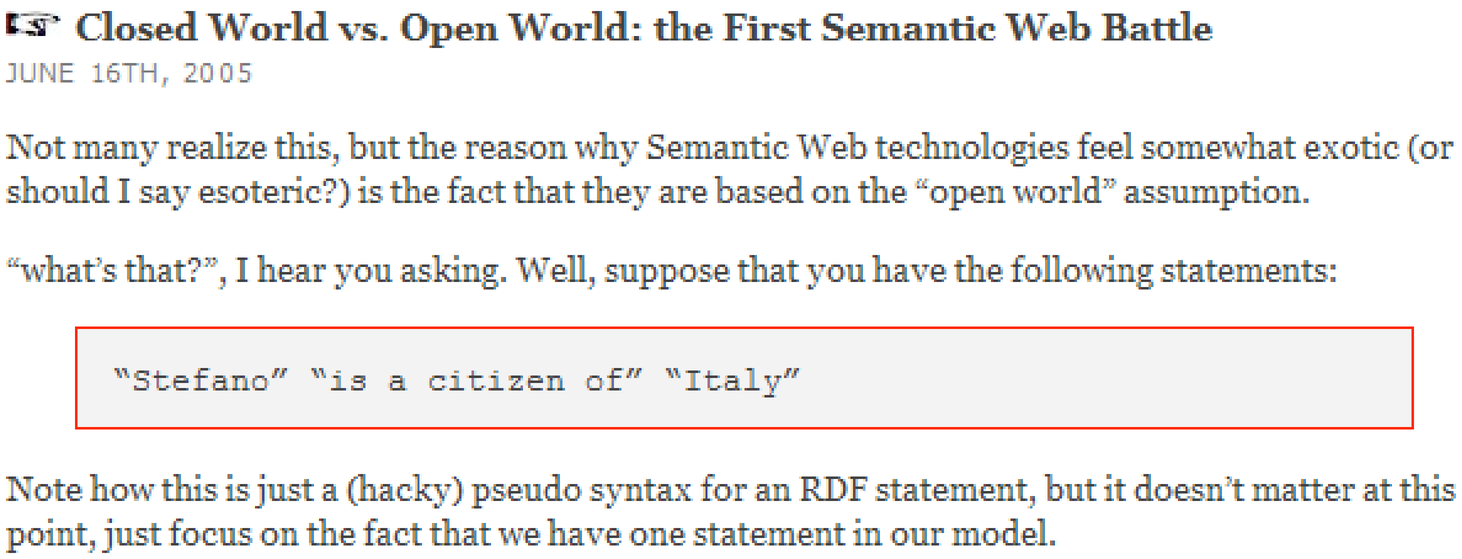
Rolf Schwitter

Rolf.Schwitter@mq.edu.au

Today’s Agenda

- Motivation for controlled natural languages
- General-purpose controlled natural languages
- Formalist versus natural approach to controlled language processing
- Theoretical considerations
- Controlled natural languages and the Semantic Web
- Writing support for controlled natural languages
- Applications of controlled natural languages

Stefano



☞ **Closed World vs. Open World: the First Semantic Web Battle**

JUNE 16TH, 2005

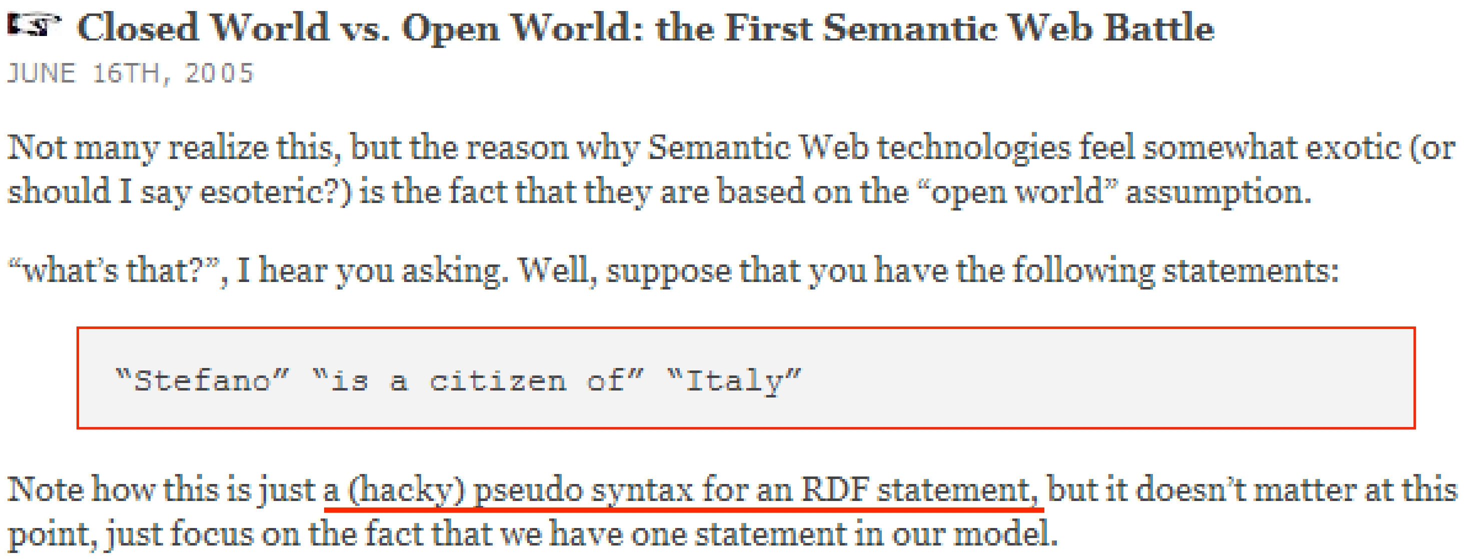
Not many realize this, but the reason why Semantic Web technologies feel somewhat exotic (or should I say esoteric?) is the fact that they are based on the "open world" assumption.

"what's that?", I hear you asking. Well, suppose that you have the following statements:

```
"Stefano" "is a citizen of" "Italy"
```

Note how this is just a (hacky) pseudo syntax for an RDF statement, but it doesn't matter at this point, just focus on the fact that we have one statement in our model.

Stefano



☞ **Closed World vs. Open World: the First Semantic Web Battle**

JUNE 16TH, 2005

Not many realize this, but the reason why Semantic Web technologies feel somewhat exotic (or should I say esoteric?) is the fact that they are based on the "open world" assumption.

"what's that?", I hear you asking. Well, suppose that you have the following statements:

```
"Stefano" "is a citizen of" "Italy"
```

Note how this is just a (hacky) pseudo syntax for an RDF statement, but it doesn't matter at this point, just focus on the fact that we have one statement in our model.

Formal versus Natural Languages

- Formal languages
  - have a well-defined syntax and an unambiguous semantics
  - have limited expressivity
  - support automatic reasoning
  - are difficult to use and understand by domain specialists.
- Natural languages
  - are the most expressive knowledge representation languages
  - are easy for humans to use and to understand
  - can serve as their own metalanguages
  - are difficult to process by a machine.

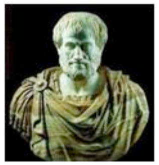
Controlled Natural Languages

- Controlled natural languages
  - are engineered subsets of natural languages
  - have a restricted grammar and a restricted vocabulary
  - reduce ambiguity and complexity of natural languages
  - look seemingly informal like natural languages
  - have the same characteristics as their formal target languages
  - can bridge the gap between natural and formal languages.

Two Categories of CNLs

- Two broad categories of controlled natural languages (CNLs):
  - human-oriented CNLs
  - machine-oriented CNLs.
- Human-oriented CNLs, e.g. for:
  - international auxiliary language for trade (Basic English)
  - maintenance documentation (ASD Simplified Technical English)
  - industrial warnings (Airbus Warning Language).
- Machine-oriented CNLs, e.g. for:
  - machine translation of technical documents (KANT)
  - knowledge acquisition and representation (ACE, PENG, CPL)
  - semantic web (ACE View, Rabbit, Lite Natural Language)
  - rule and policy languages (ACE rules).

CNLs for Knowledge Representation



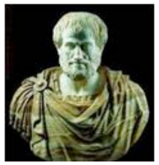
- Long tradition.
- Aristotle used a small subset of Greek for expressing logic and rules of syllogisms for reasoning.
- Example:

  All A are B.

  Some C are not B.

  Therefore: Some C are not A.
- First-order logic replaced Aristotle's syllogisms.
- First-order logic uses mathematical symbols in formulas.

CNLs for Knowledge Representation



- Long tradition.
- Aristotle used a small subset of Greek for expressing logic and rules of syllogisms for reasoning.
- Example: Baroco

  All A are B.

  Some C are not B.

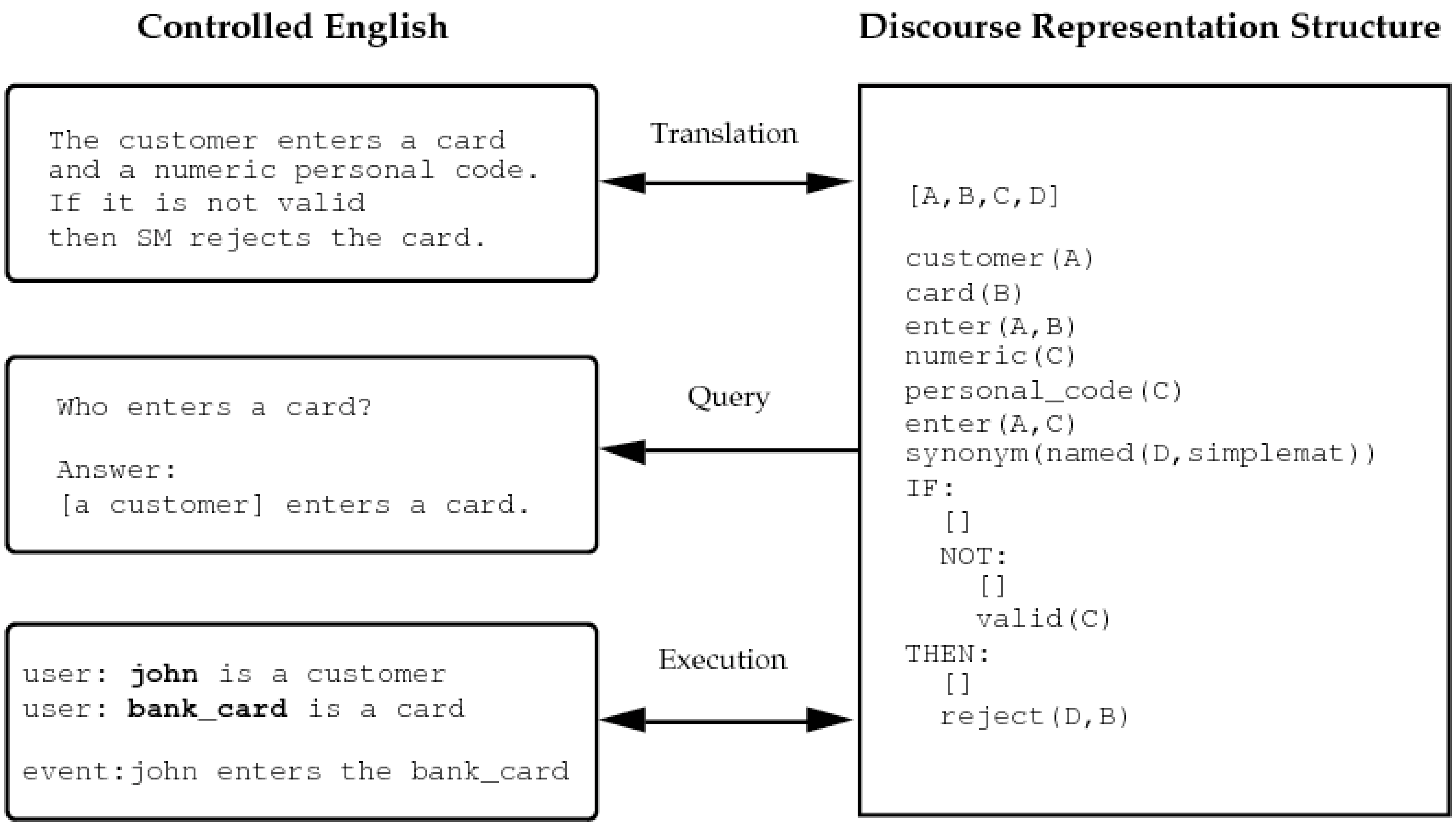
  Therefore: Some C are not A.
- First-order logic replaced Aristotle's syllogisms.
- First-order logic uses mathematical symbols in formulas.

Still Early Days

- Early days: executable specifications
- Library database specification

  If a borrower asks for a copy of a book and the copy is available and LibDB calculates the book amount of the borrower and the book amount is smaller than the book limit and a staff member checks out the copy to the borrower then the copy is checked out to the borrower.
- Translation of specification into Prolog via discourse representation structures.
- Execution of the specification:
  - query the engineer (definition of side effects for events)
  - query the user (unknown information about the situation).

Still Early Days



```
The customer enters a card
and a numeric personal code.
If it is not valid
then SM rejects the card.
```

```
Who enters a card?

Answer:
[a customer] enters a card.
```

```
user: john is a customer
user: bank_card is a card

event:john enters the bank_card
```

```
[A,B,C,D]

customer(A)
card(B)
enter(A,B)
numeric(C)
personal_code(C)
enter(A,C)
synonym(named(D,simplemat))
IF:
  []
  NOT:
    []
    valid(C)
THEN:
  []
  reject(D,B)
```

Fuchs & Schwitter 1996

General-Purpose CNLs

- General-purpose CNLs are designed for knowledge representation:
  - Attempto Controlled English (ACE)
  - Processable English (PENG)
  - Computer Processable English (CPL).
- CNLs are (automatically) translated into a formal target language.
- Interpretation of the machine is reflected via a paraphrase in CNL.
- Reasoning tools for: consistency checking and question answering.
- Important issue: writing support for CNLs.
- How can we make sure that an author sticks to a CNL?

General-Purpose CNLs

Attempto Controlled English

Every company that buys at least three machines gets a discount. Six Swiss companies each buy one machine. A German company buys four machines. Who gets a discount?

Processable English

The label of the box a says APPLES. The label of the box b says ORANGES. The label of the box c says BANANAS. APPLES stands for apples. ORANGES stands for oranges. BANANAS stands for bananas. All apples are fruits. All bananas are fruits. All oranges are fruits. Each box contains the apples or contains the bananas or contains the oranges. It is not the case that a box contains fruits and that the label of the box says something that stands for those fruits. It is not the case that a box X contains fruits and that a box Y contains those fruits. The box b contains the apples. What does the box a contain? What does the box c contain?

Computer-Processable Language

A man drives a car along a road for 1 hour.

The speed of the car is 30 km/h.

Computer-Processable Language Lite

A person drives a vehicle.

The path of the driving is a road.

The duration of the driving is 1 hour.

The speed of the driving is 30 km/h.

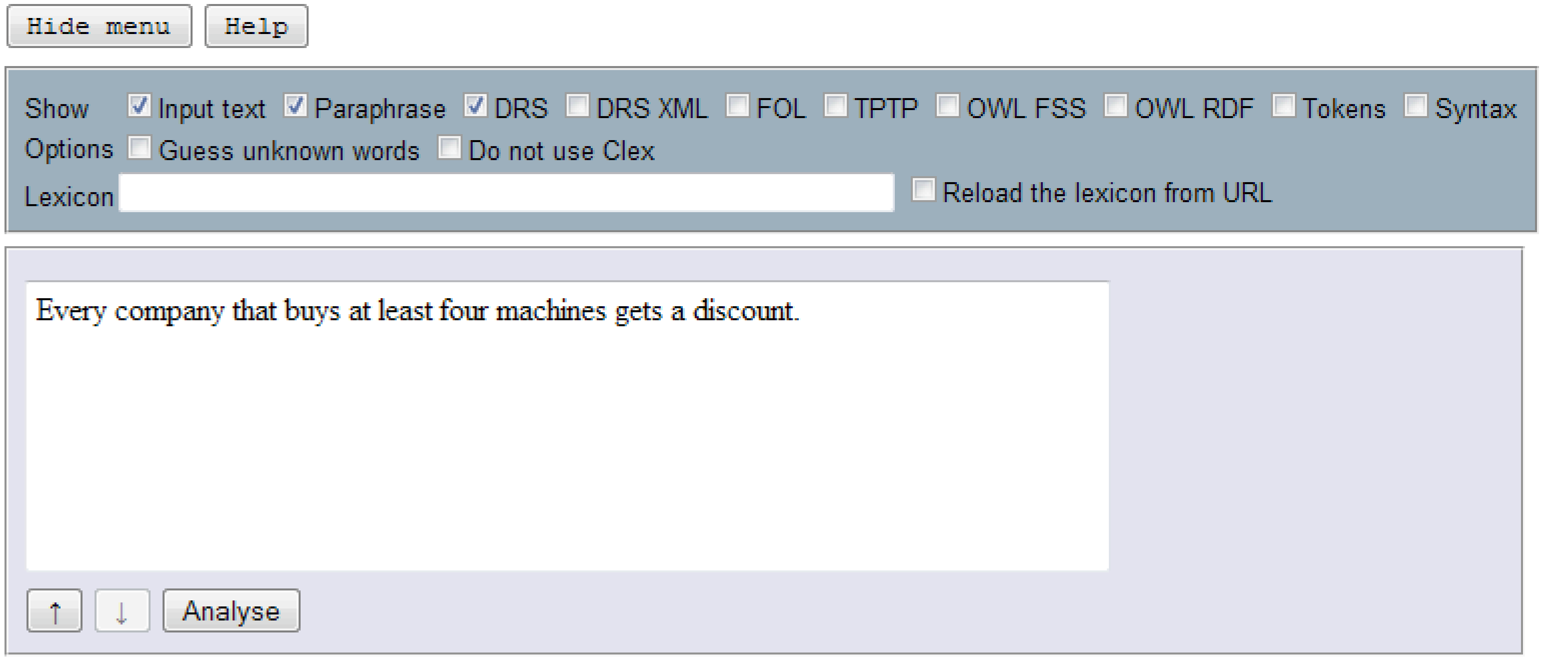
Attempto Controlled English

- ACE translates a subset of English into FOL via DRSs.
- ACE uses a small set of
  - construction rules for the syntax
  - interpretation rules for disambiguating constructs.
- ACE uses
  - predefined function words (and, or, if-then, …)
  - predefined fixed phrases (it is false that, there is, …)
  - content words (nouns, verbs, adjectives, adverbs).
- Various tools: parser, paraphraser, reasoner, AceWiki, …

ACE: Example

Every company that buys at least three machines gets a discount. Six Swiss companies each buy one machine. A German company buys four machines. Who gets a discount?

ACE Tools: ACE Parser



Show [✓ Input text] [✓ Paraphrase] [✓ DRS] DRS XML FOL TPTP OWL FSS OWL RDF Tokens Syntax

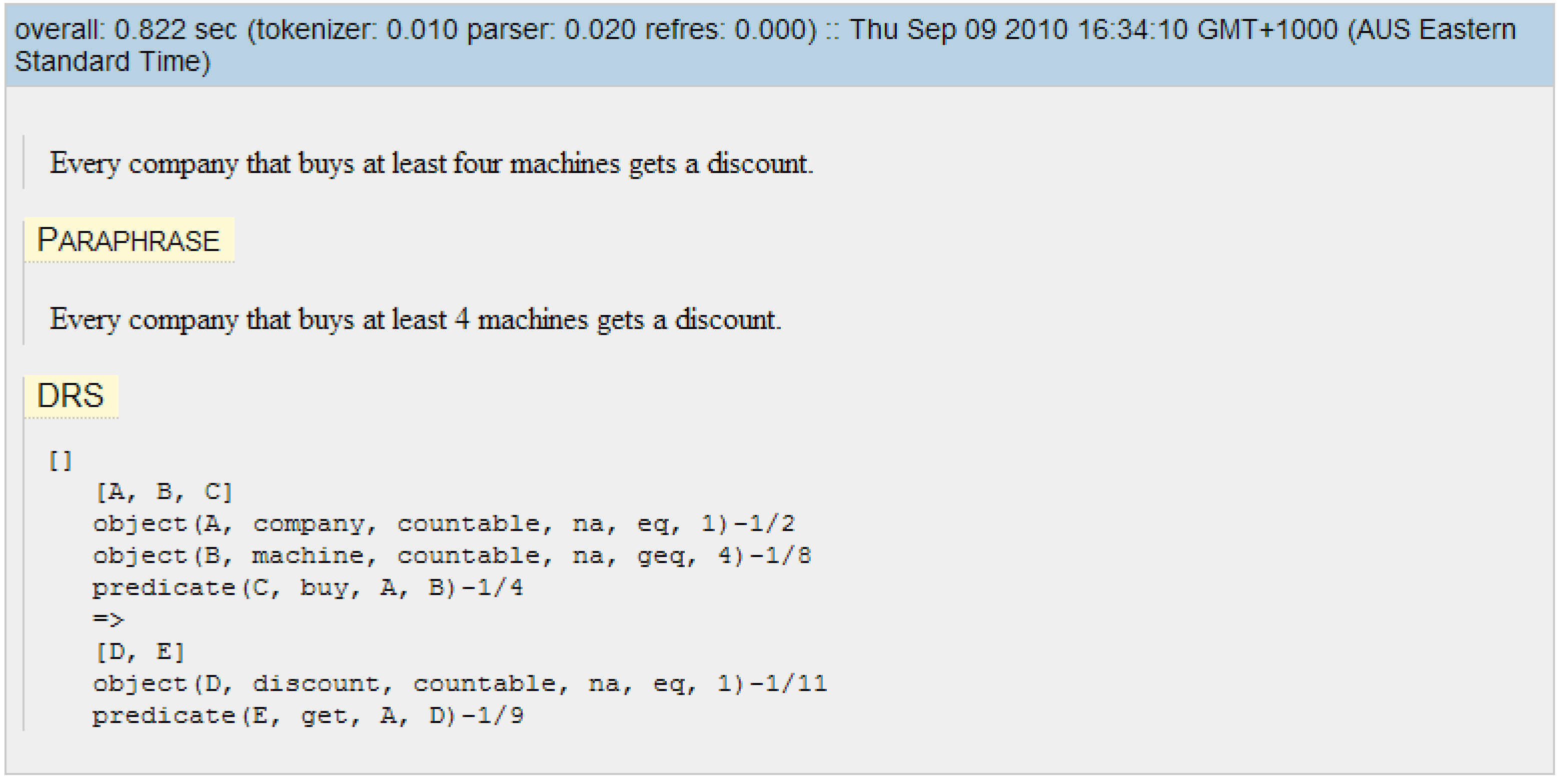
Options Guess unknown words Do not use Clex

Lexicon Reload the lexicon from URL

Every company that buys at least four machines gets a discount.

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

ACE Tools: ACE Parser Output



overall: 0.822 sec (tokenizer: 0.010 parser: 0.020 refres: 0.000) :: Thu Sep 09 2010 16:34:10 GMT+1000 (AUS Eastern Standard Time)

Every company that buys at least four machines gets a discount.

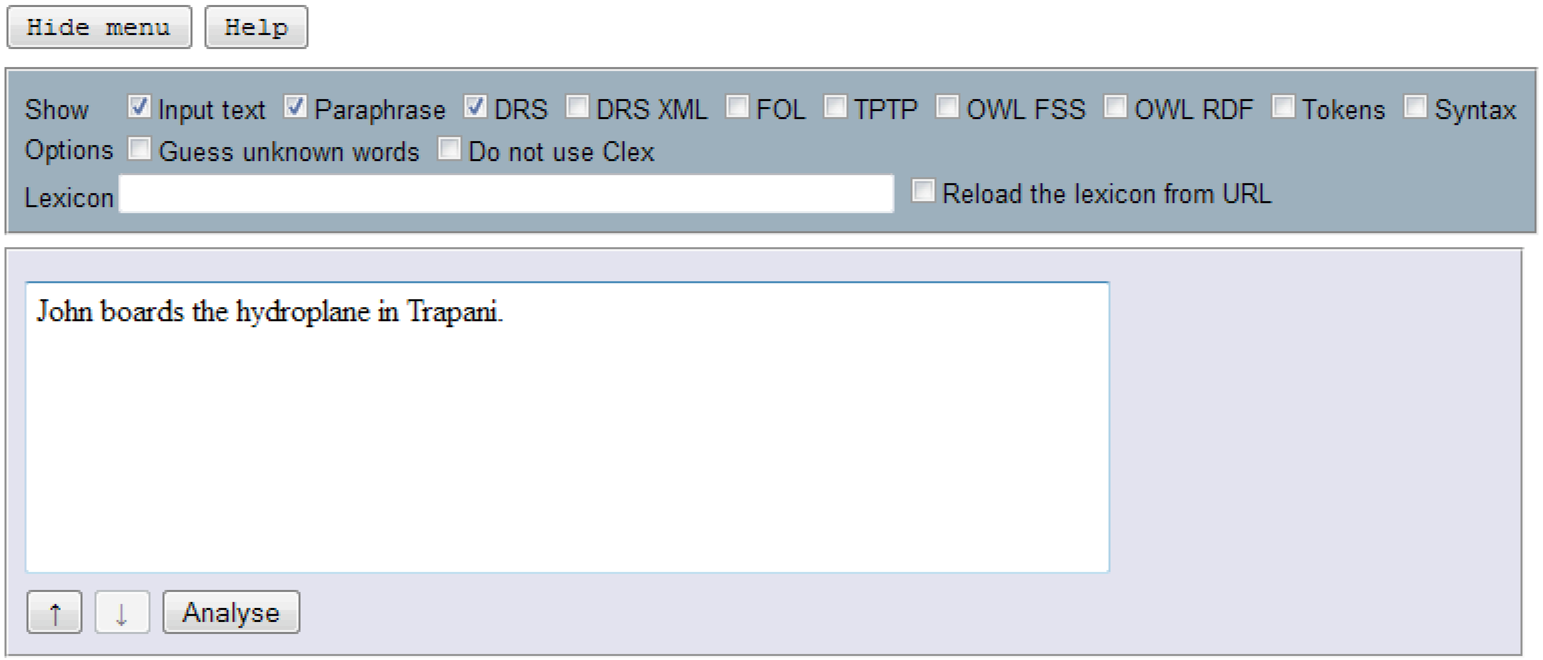
PARAPHRASE

Every company that buys at least 4 machines gets a discount.

DRS

```
[]
   [A, B, C]
   object(A, company, countable, na, eq, 1)-1/2
   object(B, machine, countable, na, geq, 4)-1/8
   predicate(C, buy, A, B)-1/4
   =>
   [D, E]
   object(D, discount, countable, na, eq, 1)-1/11
   predicate(E, get, A, D)-1/9
```

ACE Tools: ACE Parser



Show [✓ Input text] [✓ Paraphrase] [✓ DRS] ☐ DRS XML ☐ FOL ☐ TPTP ☐ OWL FSS ☐ OWL RDF ☐ Tokens ☐ Syntax

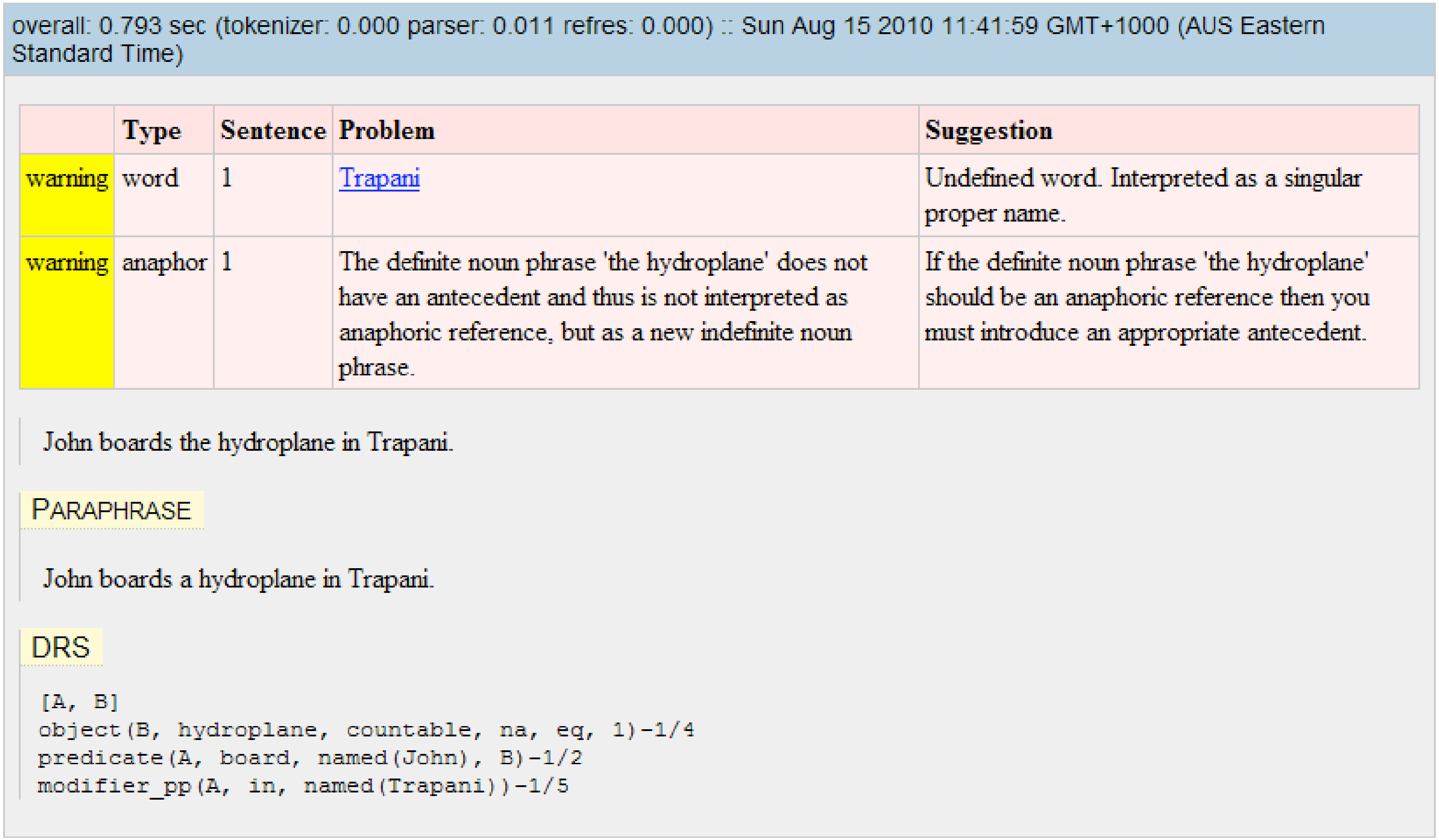
Options ☐ Guess unknown words ☐ Do not use Clex

Lexicon ☐ Reload the lexicon from URL

John boards the hydroplane in Trapani.

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

ACE Tools: ACE Parser Output



overall: 0.793 sec (tokenizer: 0.000 parser: 0.011 refres: 0.000) :: Sun Aug 15 2010 11:41:59 GMT+1000 (AUS Eastern Standard Time)

	Type	Sentence	Problem	Suggestion
warning	word	1	Trapani	Undefined word. Interpreted as a singular proper name.
warning	anaphor	1	The definite noun phrase 'the hydroplane' does not have an antecedent and thus is not interpreted as anaphoric reference, but as a new indefinite noun phrase.	If the definite noun phrase 'the hydroplane' should be an anaphoric reference then you must introduce an appropriate antecedent.

John boards the hydroplane in Trapani.

PARAPHRASE

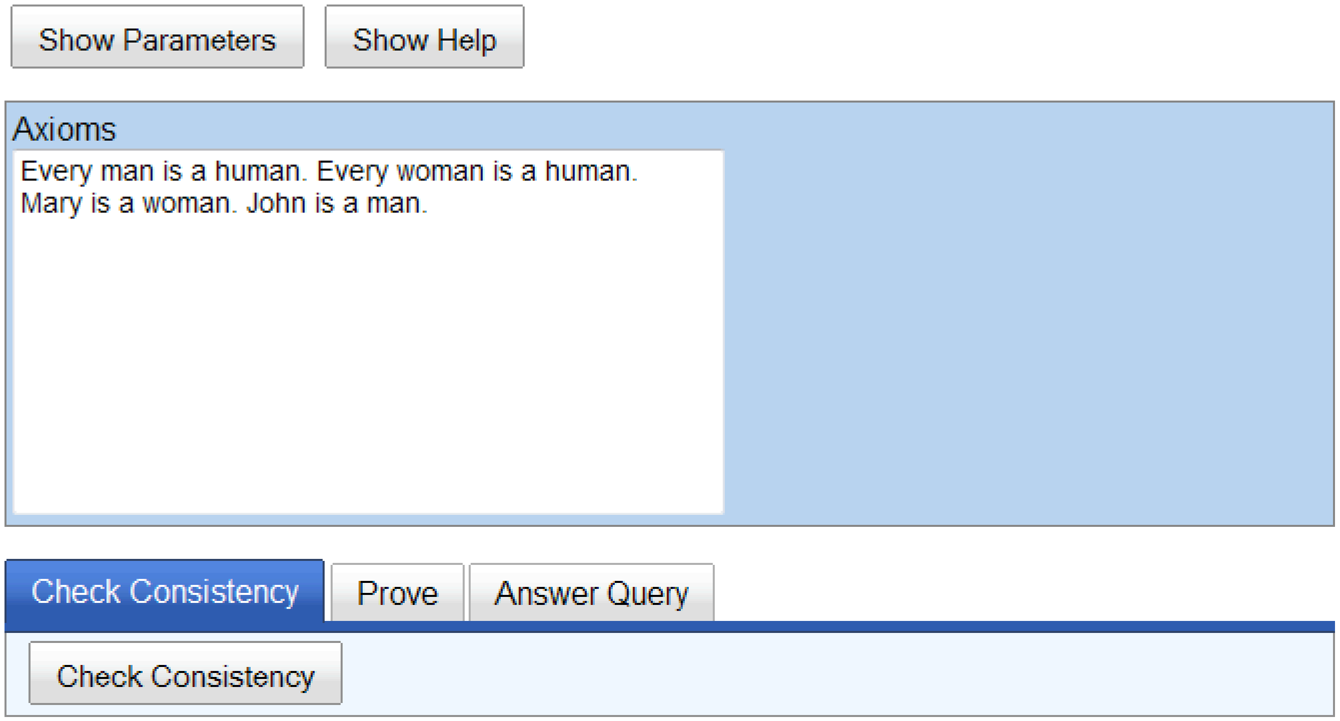
John boards a hydroplane in Trapani.

DRS

```
[A, B]
object(B, hydroplane, countable, na, eq, 1)-1/4
predicate(A, board, named(John), B)-1/2
modifier_pp(A, in, named(Trapani))-1/5
```

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

ACE Tools: Racer



Every man is a human. Every woman is a human.
Mary is a woman. John is a man.

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

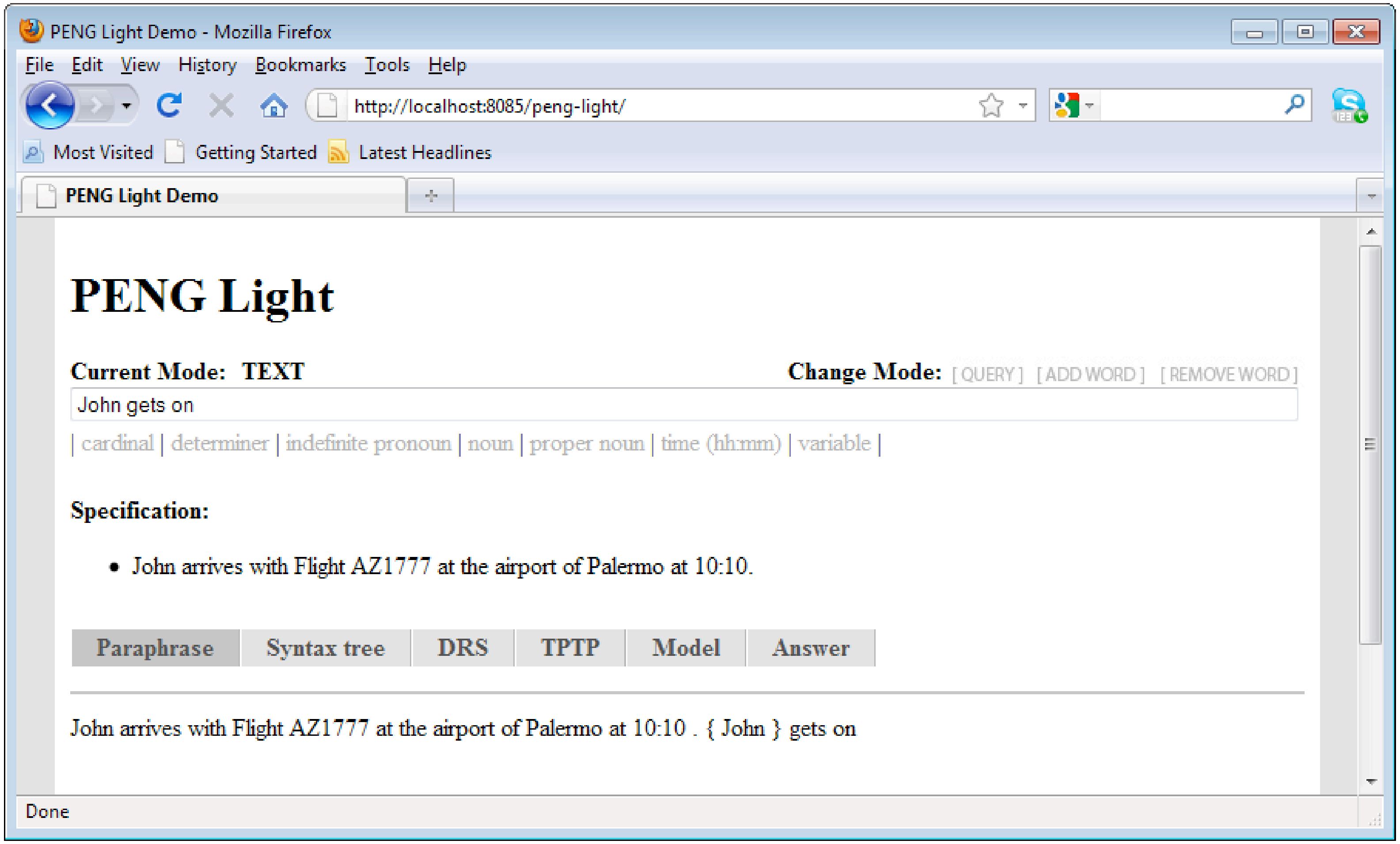
PENG

PENG
Processable ENGlish

- Uses a unification-based phrase structure grammar & chart parser.
- Parsing is done incrementally during the writing process.
- During parsing the following activities occur in parallel:
  - anaphoric expressions are resolved
  - a discourse representation structure is generated
  - a paraphrase is produced
  - look-ahead information is generated.
- PENG uses a predictive text editor that guides the writing process.
- FOL representation is processed by a model builder:
  - consistency checking, question answering.

PENG: Interface

PENG
Processable ENGlish


PENG Light Demo - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://localhost:8085/peng-light/
Most Visited Getting Started Latest Headlines
PENG Light Demo
Done

PENG Light

Current Mode: **TEXT**

Change Mode: [QUERY] [ADD WORD] [REMOVE WORD]

John gets on

| cardinal | determiner | indefinite pronoun | noun | proper noun | time (hh:mm) | variable |

Specification:

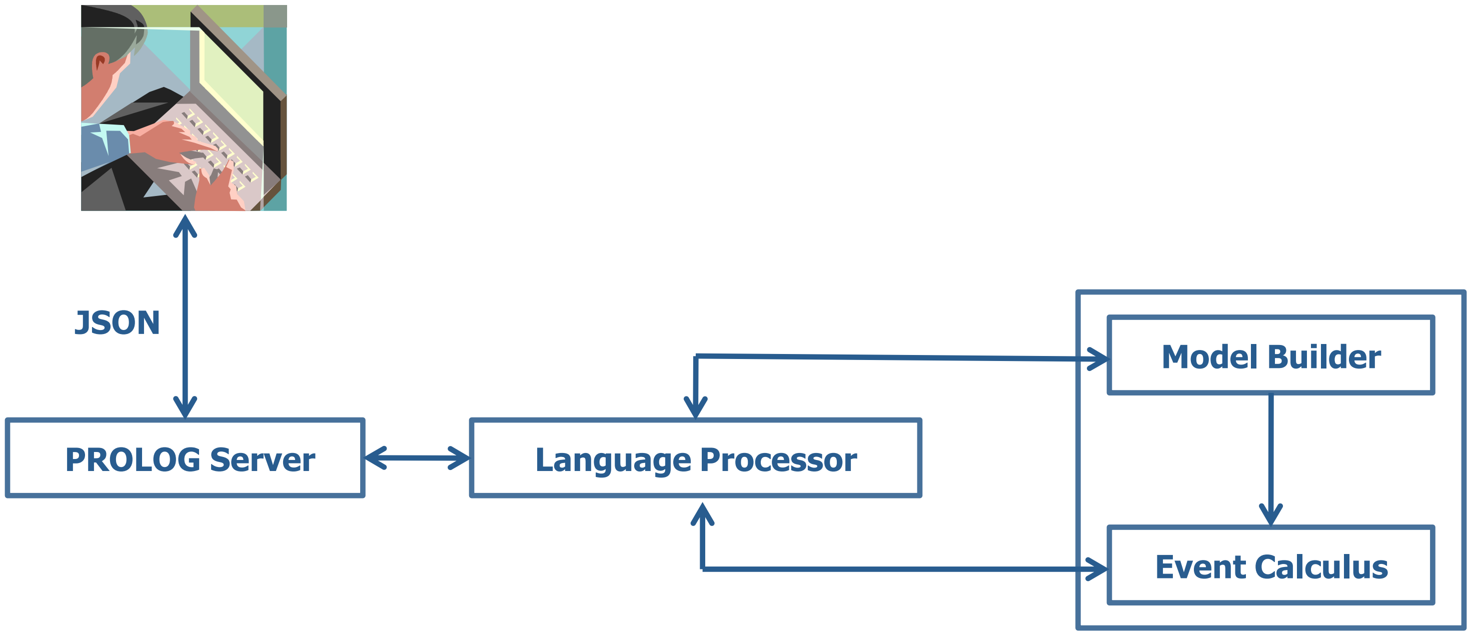
- John arrives with Flight AZ1777 at the airport of Palermo at 10:10.

Paraphrase **Syntax tree** **DRS** **TPTP** **Model** **Answer**

John arrives with Flight AZ1777 at the airport of Palermo at 10:10 . { John } gets on

PENG: Architecture

PENG
Processable ENGlish



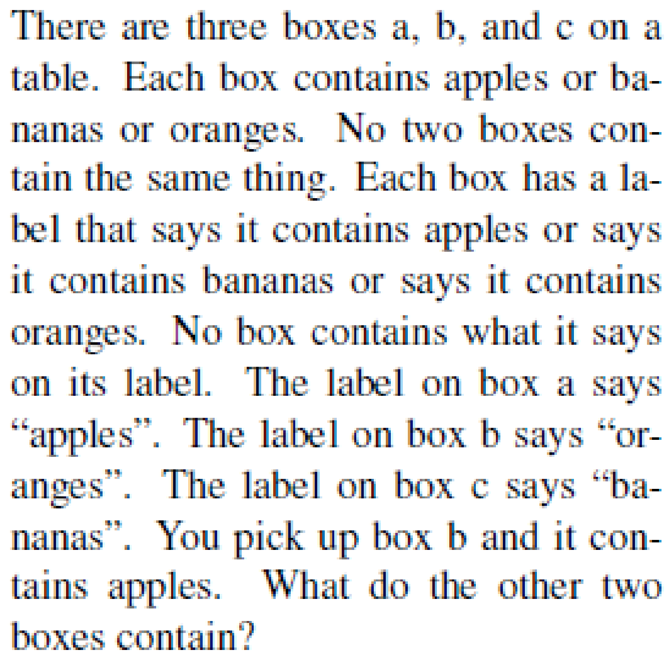
PENG Light

The Movie

PENG: Reconstructing a Problem

PENG
Processable ENGlish

- How to solve this NL problem by a machine?



There are three boxes a, b, and c on a table. Each box contains apples or bananas or oranges. No two boxes contain the same thing. Each box has a label that says it contains apples or says it contains bananas or says it contains oranges. No box contains what it says on its label. The label on box a says "apples". The label on box b says "oranges". The label on box c says "bananas". You pick up box b and it contains apples. What do the other two boxes contain?

Reconstruction: FOF or CNF

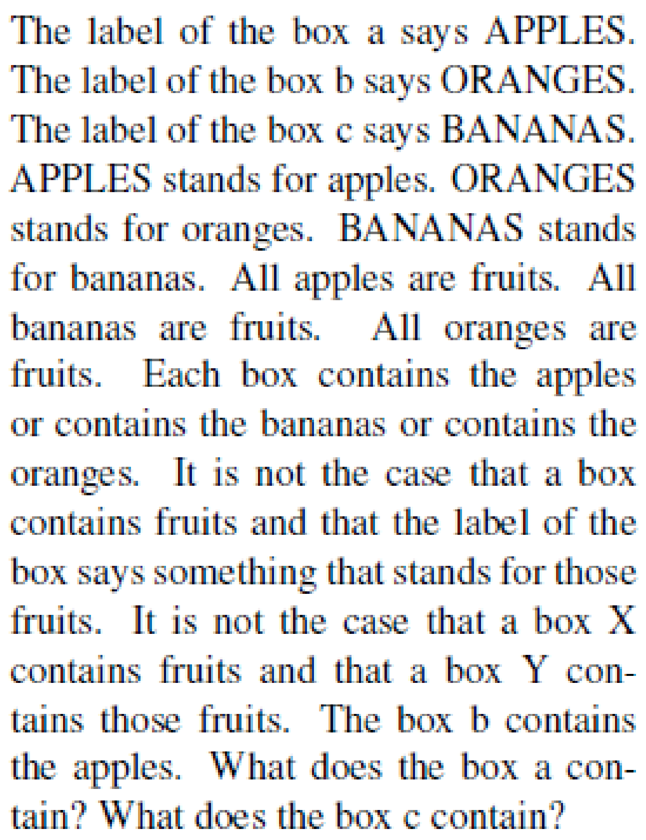
- Reconstruction in a formal notation:

```
cnf(label_is_wrong,axiom,
    ( ~ label(X,Y)
    | ~ contains(X,Y) )).

cnf(each_thing_is_in_a_box,axiom,
    ( contains(boxa,X)
    | contains(boxb,X)
    | contains(boxc,X) )).

cnf(each_box_contains_something,axiom,
    ( contains(X,apples)
    | contains(X,bananas)
    | contains(X,oranges) )).

cnf(contains_is_well_defined1,axiom,
    ( ~ contains(X,Y)
    | ~ contains(X,Z)
    | equal_fruits(Y,Z) )).


 ...
```

PENG: Reconstruction

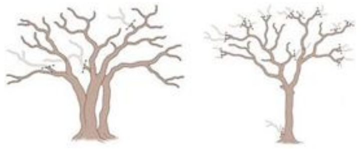
PENG
Processable ENGlish

- Reconstruction in PENG:



The label of the box a says APPLES. The label of the box b says ORANGES. The label of the box c says BANANAS. APPLES stands for apples. ORANGES stands for oranges. BANANAS stands for bananas. All apples are fruits. All bananas are fruits. All oranges are fruits. Each box contains the apples or contains the bananas or contains the oranges. It is not the case that a box contains fruits and that the label of the box says something that stands for those fruits. It is not the case that a box X contains fruits and that a box Y contains those fruits. The box b contains the apples. What does the box a contain? What does the box c contain?

Two Controlled Language Variants



- Two divergent schools of thought to the design of CNLs:
  - formalist approach
  - naturalist approach.
- Formalist approach sees CNL as
  - an English-like formal/programming language
  - a well-defined and predictable language
  - a human-readable formal language.
- Naturalist approach sees CNL as
  - a simpler form of full natural language
  - a language with reduced ambiguity
  - a language that is better processable by a machine.

Two Controlled Language Variants

- Formalist approach:
  - Attempto Controlled English (ACE)
  - Processable English (PENG)
  - Common Logic Controlled English (CLCE)
  - Computer-Processable Language Lite (CPL-Lite).
- Naturalist approach:
  - Computer-Processable Language (CPL)
  - Controlled English to Logic Translation (CELT).

Computer-Processable Language

- CPL falls into the naturalist CNL group.
- CPL attempts to simplify natural language processing.
- CPL translates sentences into KM assertions (~ situation calculus).
- CPL uses a variety of heuristics
  - to make disambiguation decisions
  - to produce a natural and "obvious" interpretation.
- CPL accepts
  - ground facts
  - questions
  - rules.

Computer-Processable Language

- Ground facts are basic sentences of the form:
  - There is | are *NP*
  - *NP verb* [*NP*] [*PP*]*
  - *NP* is | are *passive-verb* [by *NP*] [*PP*]*

  where
  - *verb* can include auxiliaries and particles
  - noun in NPs can be modified by other nouns, adjectives, PPs.

Computer-Processable Language

- CPL accepts questions of the form:
  - What is *NP*?
  - Is it true that *Sentence*?
- CPL accepts rules of the form:
  - IF *Sentence* [AND *Sentence*]* THEN *Sentence* [AND *Sentence*]*

Computer-Processable Language

- CPL's grammar supports:
  - simple sentences
  - prepositional phrases
  - compound nouns
  - ordinal modifiers
  - proper nouns
  - adjectives
  - pronouns
  - simple forms of definite references
  - limited use of conjunction (John met Sue and Mary).

Computer-Processable Language

- CPL's grammar does not allow:
  - modals
  - relative clauses
  - imperatives
  - disjunction and other forms of coordination.

Computer-Processable Language

- Definite descriptions are resolved to the most recent antecedent with the same description.
- If a referent is not found, CPL assumes that referent and creates an object.
- Pronouns resolve to the most recent object with the same gender as the pronoun.
- Existential quantification translates to Skolemized ground assertions.
- Universal quantification is expressed via IF-THEN rules.
- Application: Answering exam-style questions about physics, chemistry, biology from a knowledge base.
- User reconstructs the original text in CPL.

Computer-Processable Language

- Original:

  An alien measures the height of a cliff by dropping a boulder from rest and measuring the time it takes to hit the ground below. The boulder fell for 23 seconds on a planet with an acceleration of gravity of 7.9m/s^2. Assuming constant acceleration and ignoring air resistance, how high was the cliff.

- Reconstruction:

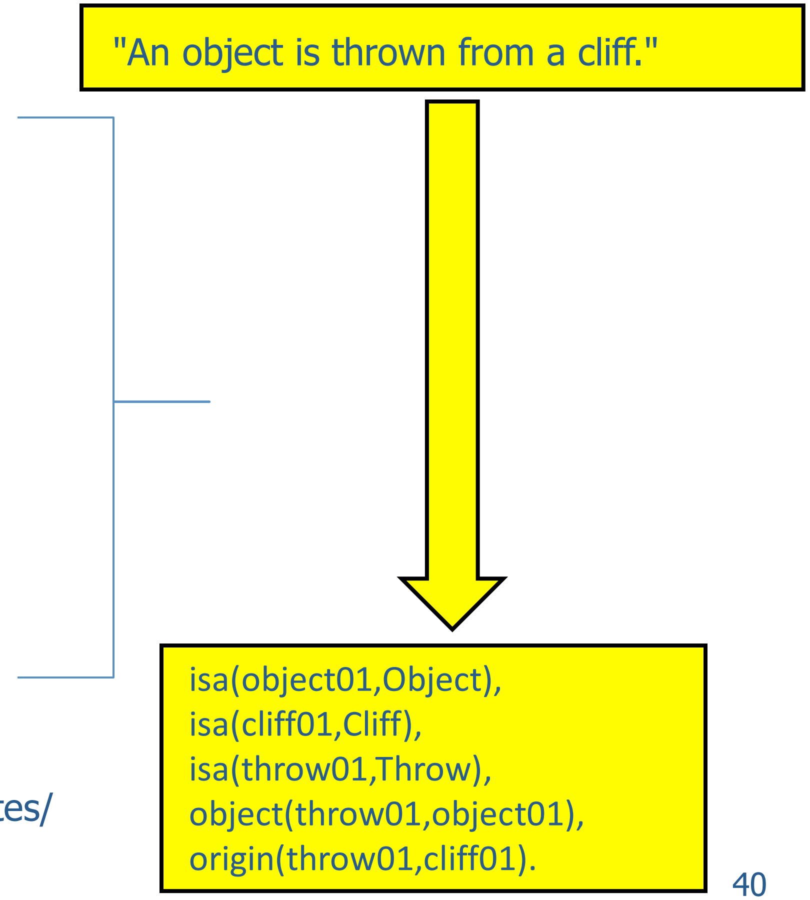
  An alien drops a boulder.
  The initial speed of the boulder is 0 m/s.
  The boulder drops for 23 seconds.
  The acceleration of the boulder is 7.9 m/s^2.
  The distance of the drop equals the height of the cliff.
  What is the height of the cliff?

Computer-Processable Language

- Reconstruction requires work of the user both conceptually and linguistically.
- CPL system provides online feedback.
- Users are trained beforehand (4 hours).
- Users make several attempts before finding a valid CPL formulation.

Computer-Processable Language

- CPL uses BLUE (Boeing Language Understanding Engine).
- Each paragraph is broken into sentences.
- CPL uses a pipeline architecture:
  1. Preprocessing
  2. Parsing
  3. Syntactic logic generation
  4. Reference resolution
  5. Transforming verbs to relations
  6. Word sense disambiguation
  7. Semantic role labeling
  8. Metonymy resolution
  9. Question annotation
  10. Additional processing



"An object is thrown from a cliff."

```
isa(object01,Object),
isa(cliff01,Cliff),
isa(throw01,Throw),
object(throw01,object01),
origin(throw01,cliff01).
```

http://userweb.cs.utexas.edu/users/pclark/working_notes/

Computer-Processable Language

1. Pre-processing
   - replaces math symbols by words
   - removes non-ASCII characters
2. Parsing
   - generates a syntax tree (GPSG-style grammar)
3. Syntactic logic generation
   - nouns, verbs, adjectives, adverbs become objects
   - prepositions, verb-argument positions become relations
4. Anaphora resolution
   - if more than one antecedent, warn user to pick the most recent
   - if none, assume a new object
   - the second red ball -> take 2nd matching object

Computer-Processable Language

5. Transform verbs to relations
   - Simple case: syntactic structure = semantic structure
   - A cell contains a nucleus -> encloses(cell01,nucleus01)
   - The explosion resulted in a fire -> causes(explosion01,fire01)
   - The cell has a nucleus -> "have"(cell01,nucleus01)
6. Word sense disambiguation
   - if word maps to ClLib ontology use that concept
   - if more than one mapping, use a preference table to pick best
   - if none then search WordNet for generalizations.

Computer-Processable Language

7. Semantic role labelling
   - via a hand-built database of about 100 rules
   - The man sang for an hour -> duration(sing01,x01)
   - The man sang for the woman -> beneficiary(sing01,woman01)
8. Metonymy resolution
   - five main types of metonymy are fixed
   - Example:

     The speed of the car is 10 km/h ->

     The speed of the movement of the car is 10 km/h

Computer-Processable Language

9. Question annotation
   - find value | definition | identity of | count | amount | subclasses
   - clausal questions: is it true | is it false | is it possible | why | how
10. Additional processing
    - Is it true that the reaction is an oxidation reaction?
    - equal(reaction01,oxidation-reaction01)
    - is-a(reaction01,oxidation-reaction01)

Computer-Processable Language

- CPL uses heuristic rules for:
  - prepositional phrase attachment
  - word sense disambiguation
  - semantic role labelling
  - compound noun interpretation
  - metonymy resolution
  - and other language processing activities.

Computer-Processable Language

- CPL output for:
  - A person throws a ball from the top of a building.

```
isa(person01,Person).
isa(ball01,Hollow-Ball).
isa(building01,Building).
isa(throw01,Throw).
has-region(building01,top01).
agent(throw01,person01).
object(throw01,ball01).
origin(thow01,top01).
```

CPL-Lite

- CPL-Lite uses the same interpreter as CPL.
- But CPL's heuristics are switched off.
- To avoid ambiguity:
  - grammar is restricted to a set of simple templates
  - vocabulary is restricted to words that map to WordNet concepts.
- CPL sentence corresponds to a binary relation between to entities.
- There are 113 sentences templates of three types.

CPL-Lite

- CPL-Lite sentence pattern:
  - 82 noun-like relations:
    - The age of an entity is a duration.
    - The agent of an event is an entity.
  - 10 verb-like relations:
    - An event_1 causes an event_2.
    - An entity_1 encloses an entity_2.
  - 21 preposition-like relations:
    - An entity_1 is above an entity_2.
    - An entity_1 is behind an entity_2.

CPL-Lite

- Complex noun phrases are not allowed.
- Sentences of the form *NP verb* [*NP*] are allowed.
- NP is one of 1000 simple nouns (incl. a few compound nouns).
- First NP is interpreted as: agent(x,y).
- Second NP is interpreted as: object(x,z).
- Verb maps to a concept.
- Definite reference is supported.
- IF-THEN rules are supported.
- Three forms of questions are supported.

CPL versus CPL-Lite

- NOTE: CPL and CPL-Lite have the same expressivity.
- CPL-Lite is grammatically more restricted and more verbose.
- CPL-Lite is more predictable.
- CPL-Lite does not guess meanings for unknown words.
- CPL graphs and paraphrases it interpretation back to the user.
- CPL paraphrases back in CPL-Lite; CPL "smarts" can go wrong.

CPL	**CPL-Lite**
A man drives a car along a road for 1 hour.	A person drives a vehicle.
The speed of the car is 30 km/h.	The path of the driving is a road.
	The duration of the driving is 1 hour.
	The speed of the driving is 30 km/h.

CPL versus CPL-Lite

- Again: CPL and CPL-Lite have the same expressivity:

```
isa(drive01,Drive)
isa(man01,Person)
isa(car01,Vehicle)
isa(road01,Road)
agent(drive01,man01)
object(drive01,car01)
path(drive01,road01)
duration(drive01,[1,hour])
speed(drive01,[30,km-per-hour])
```

Combining CPL and CPL-Lite

- CPL-Lite has been embedded as a deterministic core into CPL.
- The user can work within the core or beyond it.
- The user can fall back to the core if the "smarts" go wrong.
- Evaluation shows that users stay in the majority (70%) in CPL-Lite.
- Most common exceptions are:
  - complex noun phrases
  - using words outside the knowledge base’s lexicon
  - using metonymy with respect to the knowledge base
- But users were trained with CPL-Lite-style examples.
- In practice: degrees of non-determinisms on different levels.

Theoretical Considerations

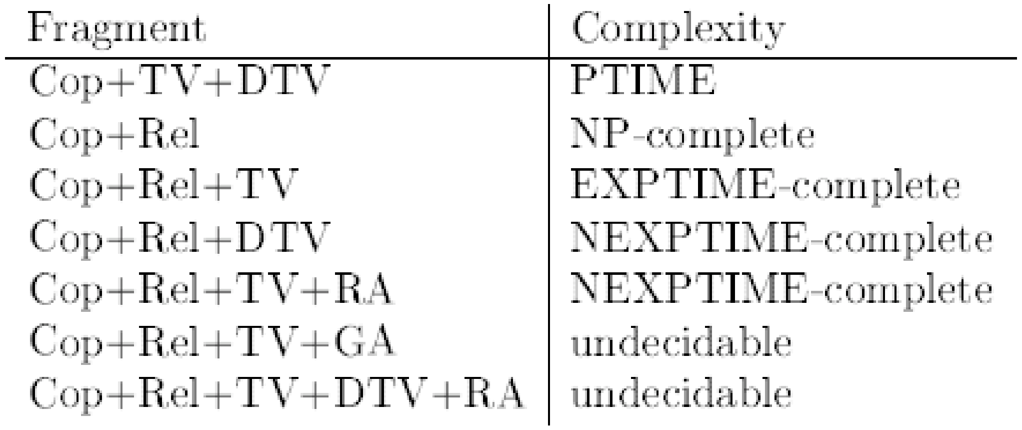
- Two important issues for the design of a CNL:
  - expressive power
  - computational complexity.
- E2V corresponds to the 2 variable fragment of first-order logic.
- Example:

  Every artist who employs a carpenter despises every beekeeper who admires *him*.
- Two interpretations:

  1. $\forall x_1\ (\mathtt{artist}(x_1)\ \&\ \exists x_2\ (\mathtt{carpenter}(x_2)\ \&\ \mathtt{employ}(x_1,x_2))\ \text{->}$
     $\forall x_3\ (\mathtt{beekeeper}(x_3)\ \&\ \mathtt{admire}(x_3,x_1)\ \text{->}\ \mathtt{despise}(x_1,x_3)))$
  2. $\forall x_1\ \forall x_2\ (\mathtt{artist}(x_1)\ \&\ \mathtt{carpenter}(x_2)\ \&\ \mathtt{employ}(x_1,x_2)\ \text{->}$
     $\forall x_3\ (\mathtt{beekeeper}(x_3)\ \&\ \mathtt{admire}(x_3,x_2)\ \text{->}\ \mathtt{despise}(x_1,x_3)))$
- Only the first one is in E2V: replace x_3 by x_2.

Complexity of CNL Fragments

- Ian Pratt-Hartmann studies the computational complexity of determining satisfiability for a set of sentences in a specific fragment:



Fragment	Complexity
Cop+TV+DTV	PTIME
Cop+Rel	NP-complete
Cop+Rel+TV	EXPTIME-complete
Cop+Rel+DTV	NEXPTIME-complete
Cop+Rel+TV+RA	NEXPTIME-complete
Cop+Rel+TV+GA	undecidable
Cop+Rel+TV+DTV+RA	undecidable

Pratt-Hartmann and Third, 2006

- RA = restricted anaphora; pronouns take closest referent.
- GA = general anaphora.

CNL for the Semantic Web

- A number of CNLs have been used as interface languages to the Semantic Web: e.g.: Lite Natural Language (Bernardi et al. 2007).
- Lite Natural Language translates into DL-Lite (description logic).
- Examples:
  - Every student is a boy.

    $[\mathit{Student} \subseteq \mathit{Boy}]$
  - Everyone who drinks something and eats something leaves.

    $[\exists \mathit{Drinks} \cap \exists \mathit{Eats} \subseteq \mathit{Leaves}]$
- But not something like:
  - Everyone who is not a boy leaves.

    $[\neg \mathit{Boy} \subseteq \mathit{Leaves}]$

CNLs for the Semantic Web

- A number of CNLs have been used as interface languages to the Semantic Web: e.g.: Lite Natural Language (Bernardi et al. 2007).
- Lite Natural Language translates into DL-Lite (description logic).
- Examples:
  - Every student is a boy.

    `∀x.(student(x) -> boy(x))`
  - Everyone who drinks something and eats something leaves.

    `∀x.(∃y.drink(x,y) ∧ ∃z.eats(x,z)) -> leaves(x)`
- But not something like:
  - Everyone who is not a boy leaves.

    `∀x.(¬(∃y.boy(y) ∧ x=y)) -> leaves(x)`

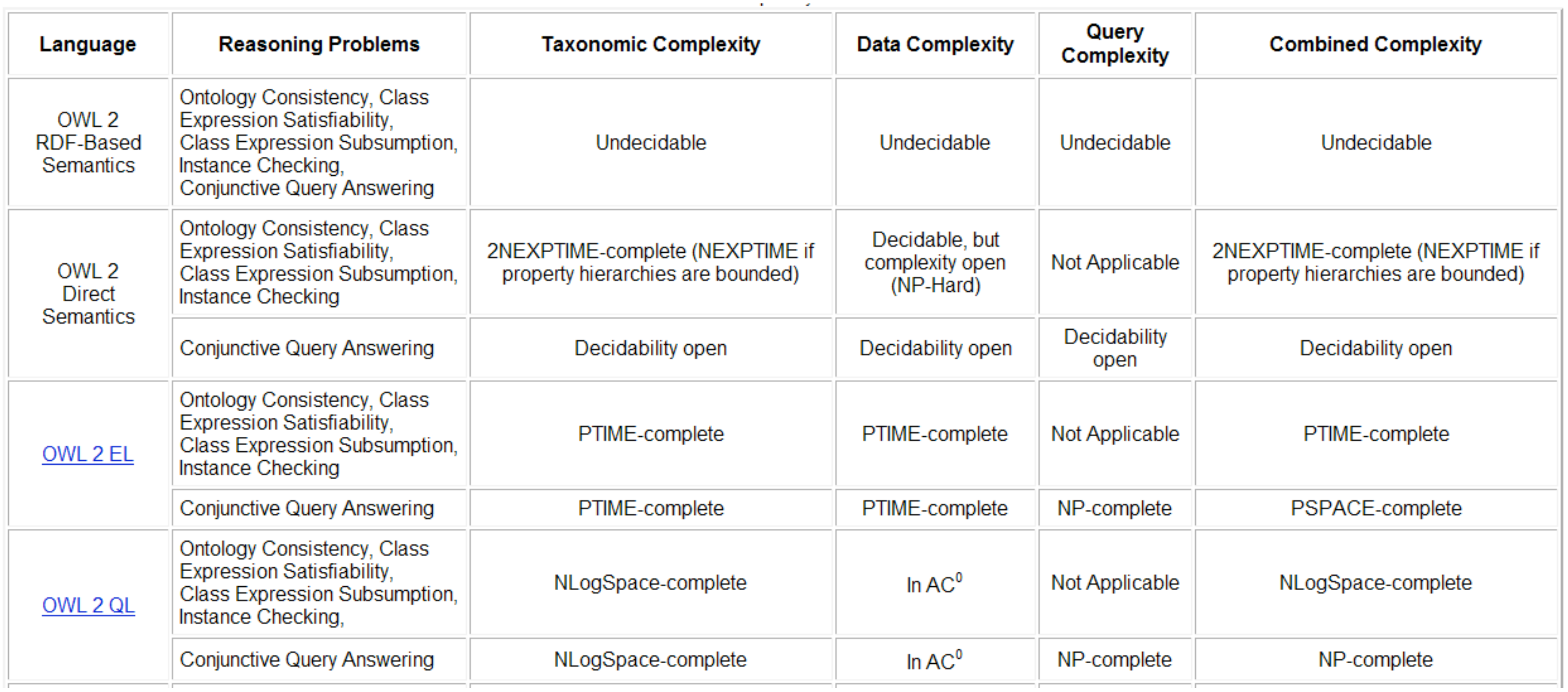
DL-Lite

- DL-Lite$_{\text{core}}$: $Cl \longrightarrow A \mid \exists R$ $Cr \longrightarrow A \mid \neg A \mid \exists R \mid \neg\exists R$
- *A* denotes an atomic concept.
- *R* denotes an atomic role.
- *¬∃R* denotes unqualified existential quantification.
  "those individuals (x) who know somebody (y)"

- DL-Lite$_{R\cap}$: $Cl \longrightarrow Cl_1 \sqcap Cl_2$ $Cr \longrightarrow \exists R.A$
- *¬∃R.A* denotes qualified existential quantification.
  "those individuals (x) who know somebody (y) who is a student"

- Assumption (Lite Natural Language): ABox is stored in a database.

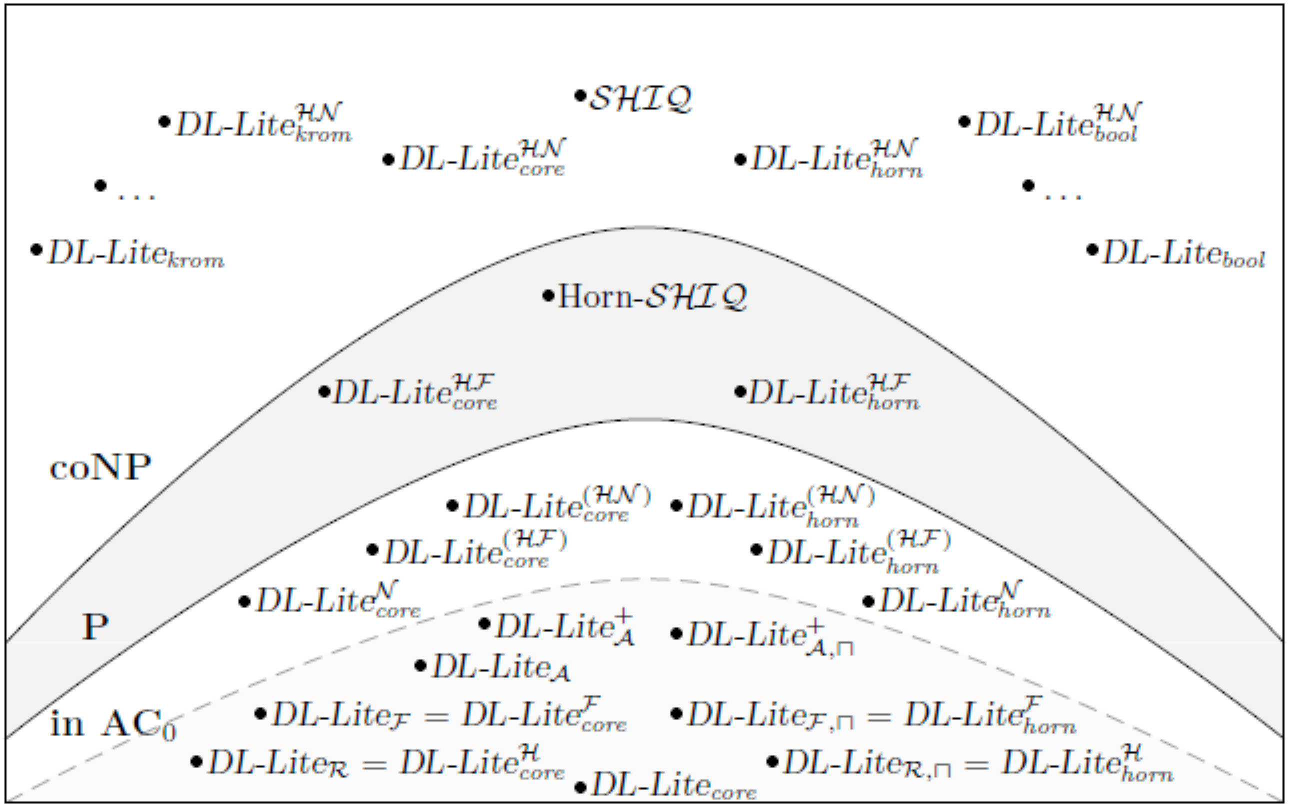
DL Complexity



Language	Reasoning Problems	Taxonomic Complexity	Data Complexity	Query Complexity	Combined Complexity
OWL 2 RDF-Based Semantics	Ontology Consistency, Class Expression Satisfiability, Class Expression Subsumption, Instance Checking, Conjunctive Query Answering	Undecidable	Undecidable	Undecidable	Undecidable
OWL 2 Direct Semantics	Ontology Consistency, Class Expression Satisfiability, Class Expression Subsumption, Instance Checking	2NEXPTIME-complete (NEXPTIME if property hierarchies are bounded)	Decidable, but complexity open (NP-Hard)	Not Applicable	2NEXPTIME-complete (NEXPTIME if property hierarchies are bounded)
	Conjunctive Query Answering	Decidability open	Decidability open	Decidability open	Decidability open
OWL 2 EL	Ontology Consistency, Class Expression Satisfiability, Class Expression Subsumption, Instance Checking	PTIME-complete	PTIME-complete	Not Applicable	PTIME-complete
	Conjunctive Query Answering	PTIME-complete	PTIME-complete	NP-complete	PSPACE-complete
OWL 2 QL	Ontology Consistency, Class Expression Satisfiability, Class Expression Subsumption, Instance Checking,	NLogSpace-complete	In AC^0	Not Applicable	NLogSpace-complete
	Conjunctive Query Answering	NLogSpace-complete	In AC^0	NP-complete	NP-complete

http://www.w3.org/TR/owl2-profiles/

DL-Lite Family and Relations



$\bullet \mathcal{SHIQ}$
$\bullet$ DL-Lite$^{\mathcal{HN}}_{krom}$
$\bullet$ DL-Lite$^{\mathcal{HN}}_{core}$
$\bullet$ DL-Lite$^{\mathcal{HN}}_{horn}$
$\bullet$ DL-Lite$^{\mathcal{HN}}_{bool}$
$\bullet \ldots$
$\bullet \ldots$
$\bullet$ DL-Lite$_{krom}$
$\bullet$ DL-Lite$_{bool}$
$\bullet$ Horn-$\mathcal{SHIQ}$
$\bullet$ DL-Lite$^{\mathcal{HF}}_{core}$
$\bullet$ DL-Lite$^{\mathcal{HF}}_{horn}$
coNP
$\bullet$ DL-Lite$^{(\mathcal{HN})}_{core}$
$\bullet$ DL-Lite$^{(\mathcal{HN})}_{horn}$
$\bullet$ DL-Lite$^{(\mathcal{HF})}_{core}$
$\bullet$ DL-Lite$^{(\mathcal{HF})}_{horn}$
$\bullet$ DL-Lite$^{\mathcal{N}}_{core}$
$\bullet$ DL-Lite$^{\mathcal{N}}_{horn}$
P
$\bullet$ DL-Lite$^{+}_{\mathcal{A}}$
$\bullet$ DL-Lite$^{+}_{\mathcal{A},\sqcap}$
$\bullet$ DL-Lite$_{\mathcal{A}}$
in AC$_0$
$\bullet$ DL-Lite$_{\mathcal{F}}$ = DL-Lite$^{\mathcal{F}}_{core}$
$\bullet$ DL-Lite$_{\mathcal{F},\sqcap}$ = DL-Lite$^{\mathcal{F}}_{horn}$
$\bullet$ DL-Lite$_{\mathcal{R}}$ = DL-Lite$^{\mathcal{H}}_{core}$
$\bullet$ DL-Lite$_{\mathcal{R},\sqcap}$ = DL-Lite$^{\mathcal{H}}_{horn}$
$\bullet$ DL-Lite$_{core}$

A = atomic concepts, $\mathcal{H}$ = role hierarchy, F = functionality, R = complex role inclusion, N = number restriction

Artale et al., 2009

Verbalising Ontologies

- Mapping between an ontology and a CNL (Power & Thrid, 2010):

```
<ClassAssertion>
  <Class IRI="http://www.example.org#admiral"/>
  <NamedIndividual IRI="www.example.org#HoratioNelson"/>
</ClassAssertion>

<ObjectPropertyAssertion>
  <ObjectProperty IRI="http://www.example.org#victorOf"/>
  <NamedIndividual IRI="http://www.example.org#HoratioNelson"/>
  <NamedIndividual IRI="http://www.example.org#BattleOfTrafalgar"/>
</ObjectPropertyAssertion>

<SubClassOf>
  <Class IRI="http://www.example.org#admiral"/>
  <ObjectSomeValuesFrom>
    <ObjectProperty IRI="http://www.example.org#commanderOf"/>
    <Class IRI="http://www.example.org#fleet"/>
  </ObjectSomeValuesFrom>
</SubClassOf>
```

Verbalising Ontologies

- Result:
  - Horatio Nelson is an admiral.
  - Horatio Nelson is the victor of the Battle of Trafalgar.
  - Every admiral is commander of a fleet.

Logical Sophistication

Functor	Example
SubClassOf	Every admiral is a sailor
EquivalentClasses	An admiral is defined as a person that commands a fleet
DisjointClasses	No sailor is a landlubber
ClassAssertion	Nelson is an admiral
ObjectPropertyAssertion	Nelson is victor of the Battle of Trafalgar
DataPropertyAssertion	The Battle of Trafalgar is dated 1805
ObjectPropertyDomain	If X commands Y, X must be a person
ObjectPropertyRange	If X commands Y, Y must be a fleet
SubObjectPropertyOf	If X is a child of Y, X must be related to Y
InverseObjectProperties	If X is a child of Y, Y must be a parent of X
TransitiveObjectProperty	If X contains Y and Y contains Z, X must contain Z
FunctionalObjectProperty	There can be only one Y such that X has as father Y
DataPropertyDomain	If X is dated Y, X must be an event
DataPropertyRange	If X is dated Y, Y must be an integer
SubDataPropertyOf	If X occurs during Y, X must be dated Y
FunctionalDataProperty	There can be only one Y such that X is dated Y

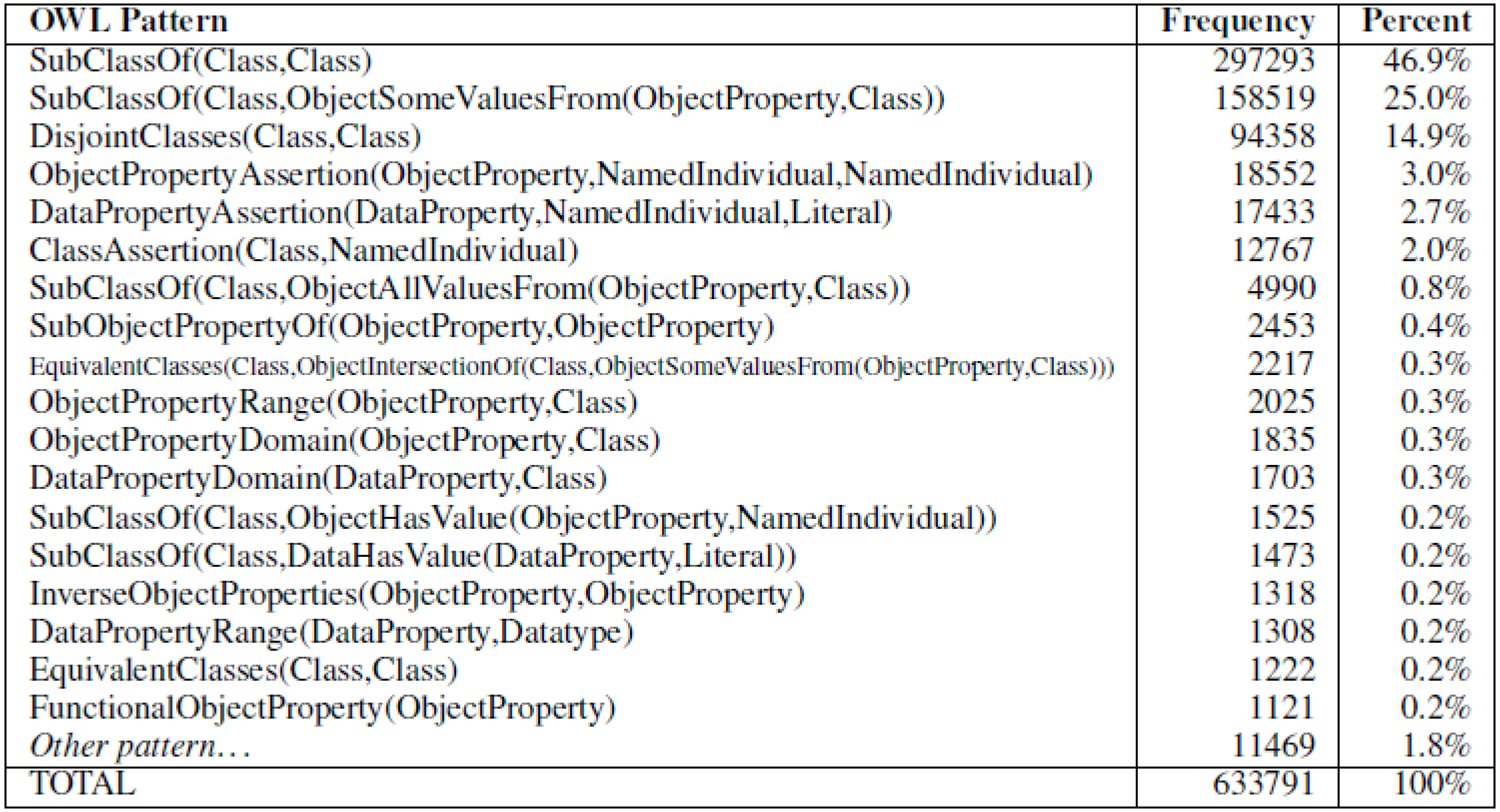
Simple pattern

Sophisticated pattern

http://aclweb.org/anthology-new/C/C10/C10-2116.pdf

Frequency of Pattern

- Study over the TONES Ontology Repository.



OWL Pattern	Frequency	Percent
SubClassOf(Class,Class)	297293	46.9%
SubClassOf(Class,ObjectSomeValuesFrom(ObjectProperty,Class))	158519	25.0%
DisjointClasses(Class,Class)	94358	14.9%
ObjectPropertyAssertion(ObjectProperty,NamedIndividual,NamedIndividual)	18552	3.0%
DataPropertyAssertion(DataProperty,NamedIndividual,Literal)	17433	2.7%
ClassAssertion(Class,NamedIndividual)	12767	2.0%
SubClassOf(Class,ObjectAllValuesFrom(ObjectProperty,Class))	4990	0.8%
SubObjectPropertyOf(ObjectProperty,ObjectProperty)	2453	0.4%
EquivalentClasses(Class,ObjectIntersectionOf(Class,ObjectSomeValuesFrom(ObjectProperty,Class)))	2217	0.3%
ObjectPropertyRange(ObjectProperty,Class)	2025	0.3%
ObjectPropertyDomain(ObjectProperty,Class)	1835	0.3%
DataPropertyDomain(DataProperty,Class)	1703	0.3%
SubClassOf(Class,ObjectHasValue(ObjectProperty,NamedIndividual))	1525	0.2%
SubClassOf(Class,DataHasValue(DataProperty,Literal))	1473	0.2%
InverseObjectProperties(ObjectProperty,ObjectProperty)	1318	0.2%
DataPropertyRange(DataProperty,Datatype)	1308	0.2%
EquivalentClasses(Class,Class)	1222	0.2%
FunctionalObjectProperty(ObjectProperty)	1121	0.2%
Other pattern…	11469	1.8%
TOTAL	633791	100%

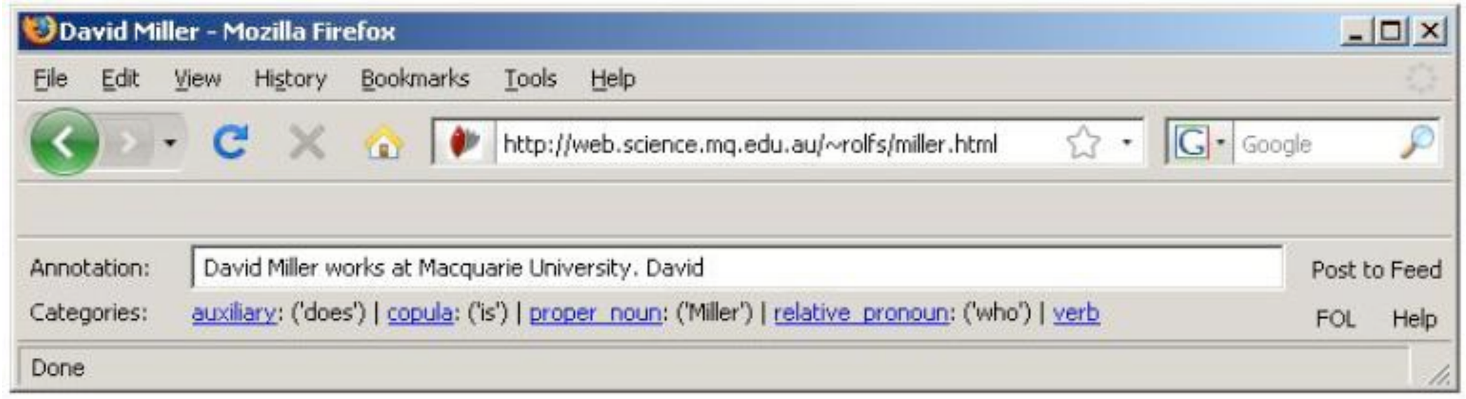
http://aclweb.org/anthology-new/C/C10/C10-2116.pdf

Findings

- A small number of patterns covers most of the axioms in the corpus (top five pattern cover 91.9%, top 20 cover 97.2%).
- All of the frequent pattern (top 20) can be expressed by a single sentence; most complex one is the equivalent class pattern, e.g.: An admiral is defined as a person that commands a fleet.
- None of the first 10 pattern belong to the sophisticated group.
- The simple argument invariable comes first (developers conceptualise statements in subject-predicate forms).
- Ontology developers treat OWL axioms by analogy with sentences.
- They assign a clear information structure: only 0.2% of first arguments are non-atomic.

Writing Support for CNLs

- Writing a specification in CNL is not easy and requires support.
- There exist three main techniques to support the writing process:
  - error messages
  - conceptual authoring
  - predictive feedback.
- Predictive feedback with lookahead categories:

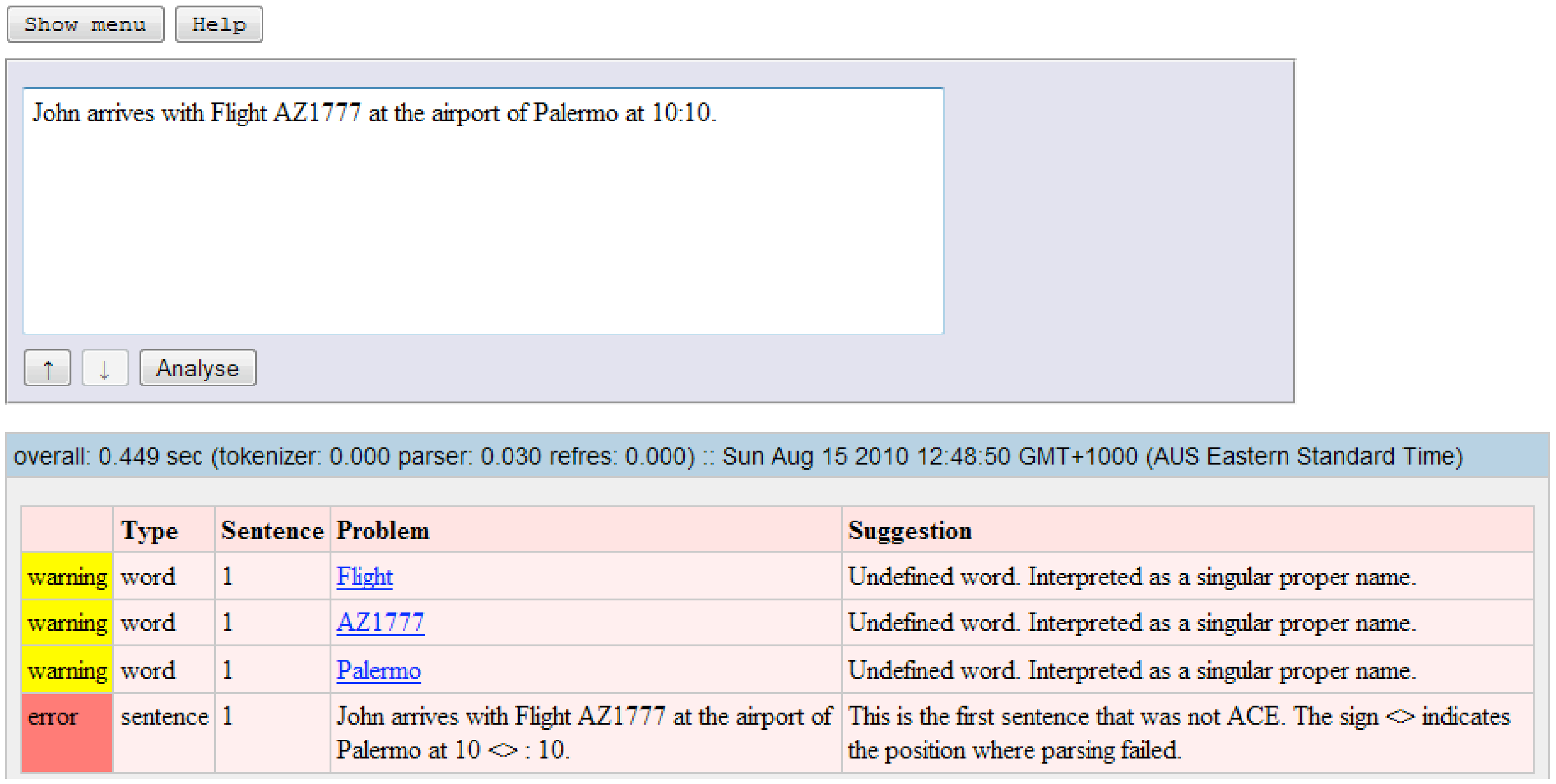

David Miller - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://web.science.mq.edu.au/~rolfs/miller.html
Google
Post to Feed
FOL Help
Done

Annotation: David Miller works at Macquarie University. David

Categories: auxiliary: ('does') | copula: ('is') | proper_noun: ('Miller') | relative_pronoun: ('who') | verb

White and Schwitter, 2009

Warning and Error Messages



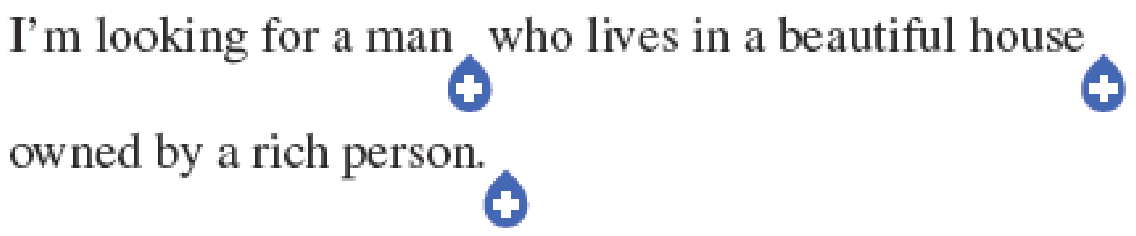
John arrives with Flight AZ1777 at the airport of Palermo at 10:10.

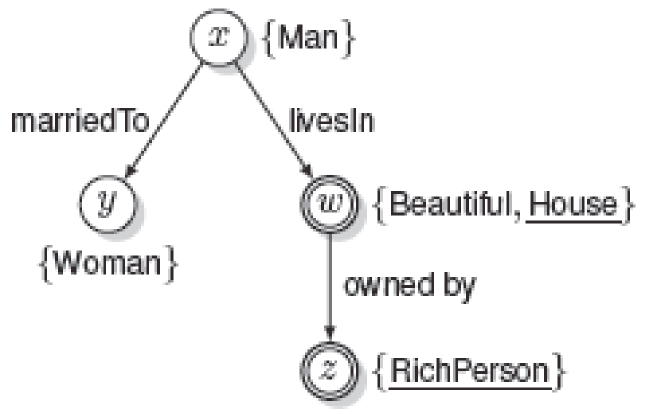
overall: 0.449 sec (tokenizer: 0.000 parser: 0.030 refres: 0.000) :: Sun Aug 15 2010 12:48:50 GMT+1000 (AUS Eastern Standard Time)

	Type	Sentence	Problem	Suggestion
warning	word	1	Flight	Undefined word. Interpreted as a singular proper name.
warning	word	1	AZ1777	Undefined word. Interpreted as a singular proper name.
warning	word	1	Palermo	Undefined word. Interpreted as a singular proper name.
error	sentence	1	John arrives with Flight AZ1777 at the airport of Palermo at 10 <> : 10.	This is the first sentence that was not ACE. The sign <> indicates the position where parsing failed.

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

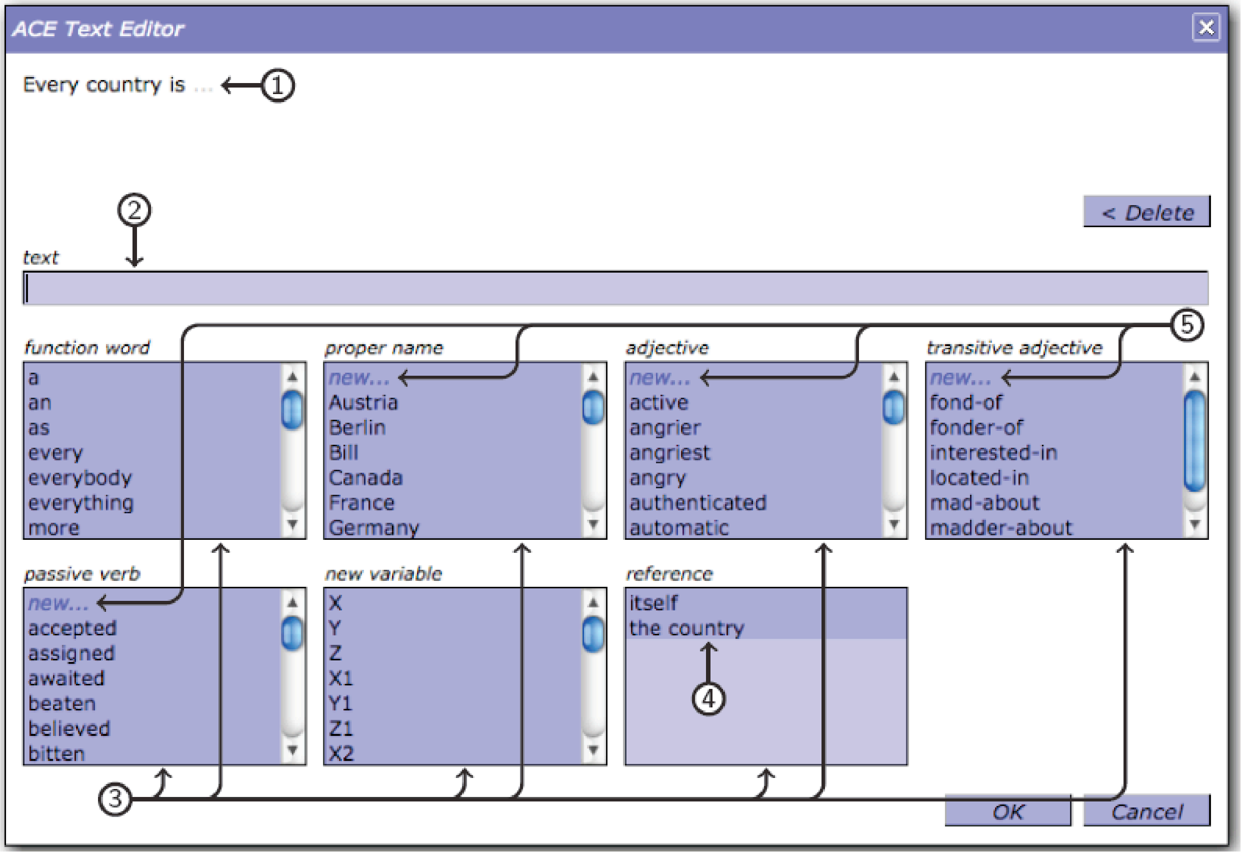
Conceptual Authoring


I'm looking for a man who lives in a beautiful house owned by a rich person.



Franconi et al. 2010

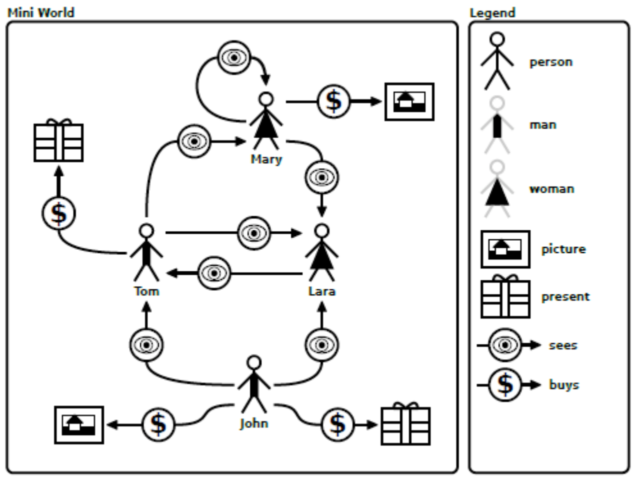
Predictive Editor

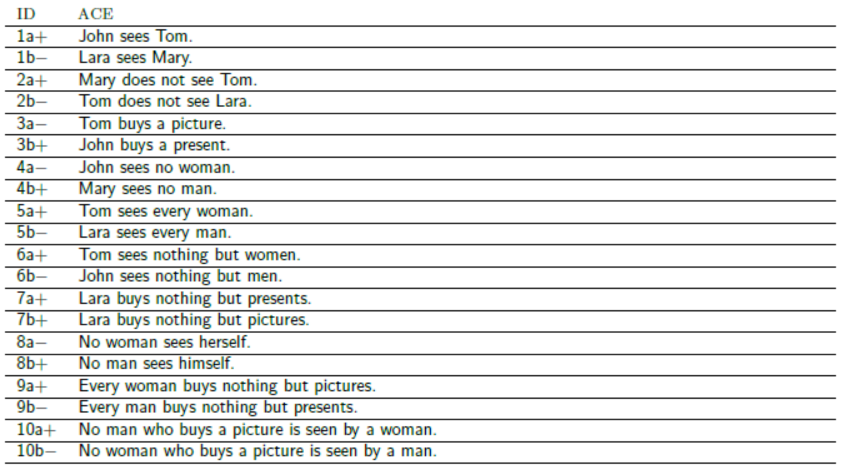


Kuhn 2010

Evaluating CNLs

- There exist different approaches to CNL evaluation:
  - task-based experiments test how easy it is to write a text in CNL but not how easy it is to understand the text;
  - paraphrase-based experiments aim to evaluate the understanding of a text in CNL in a tool-independent way;
  - graph-based experiments use a graph-based notation to describe a situation accompanied with statements in CNL.





ID	ACE
1a+	John sees Tom.
1b–	Lara sees Mary.
2a+	Mary does not see Tom.
2b–	Tom does not see Lara.
3a–	Tom buys a picture.
3b+	John buys a present.
4a–	John sees no woman.
4b+	Mary sees no man.
5a+	Tom sees every woman.
5b–	Lara sees every man.
6a+	Tom sees nothing but women.
6b–	John sees nothing but men.
7a+	Lara buys nothing but presents.
7b+	Lara buys nothing but pictures.
8a–	No woman sees herself.
8b+	No man sees himself.
9a+	Every woman buys nothing but pictures.
9b–	Every man buys nothing but presents.
10a+	No man who buys a picture is seen by a woman.
10b–	No woman who buys a picture is seen by a man.

Kuhn, 2010

Applications of CNLs

- Applications of CNLs include:
  - software and hardware specifications
  - specifications of legal contracts
  - medical regulations
  - agent control
  - business rule specifications
  - interfaces to formal ontologies
  - interfaces to situation awareness systems.

Future Research

- Many topics are promising for future research:
  - study of expressivity within CNL fragments
  - combination of deterministic and non-deterministic CNL fragments
  - improved and alternative interfaces for writing CNLs
  - visualisation of scenarios written in CNL
  - CNLs and temporal logics
  - CNLs and deontic logics
  - CNLs for legal reasoning
  - CNLs for specifying business rules.

Conclusions

- CNLs can be translated automatically into a formal target language.
- Writing process needs to be supported by an intelligent authoring tool.
- An ideal CNL should
  - have a well-defined syntax and a precise semantics;
  - look as natural as possible and be based on a natural language;
  - be easy to understand and easy to process;
  - be expressive enough to describe the problems at hand.
- CNLs can bridge the gap between natural and formal languages.
- CNLs allow for true collaboration between humans and machines.