Common Logic

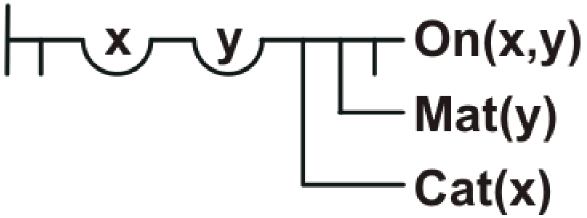
A Framework for a Family
Of Logic-Based Languages

John F. Sowa

19 May 2008

How to say “A cat is on a mat.”

Gottlob Frege (1879):



Charles Sanders Peirce (1885):

$$\Sigma_{x}\ \Sigma_{y}\ \mathbf{Cat}_{x} \bullet \mathbf{Mat}_{y} \bullet \mathbf{On}_{x,y}$$

Giuseppe Peano (1895):

$$\exists x\ \exists y\ \mathbf{Cat}(x) \wedge \mathbf{Mat}(y) \wedge \mathbf{On}(x,y)$$

Frege and Peirce developed their notations independently.

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

But all the 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
```

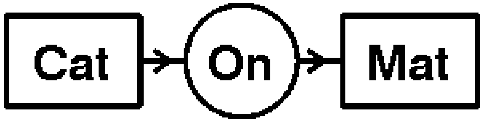
Common Logic Interchange Format (ISO 24707):

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

Conceptual Graph Interchange Format (ISO 24707):

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

Conceptual Graph Display Format:



Controlled English:

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

A Family of Languages

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 sample dialects are specified in ISO 24707:

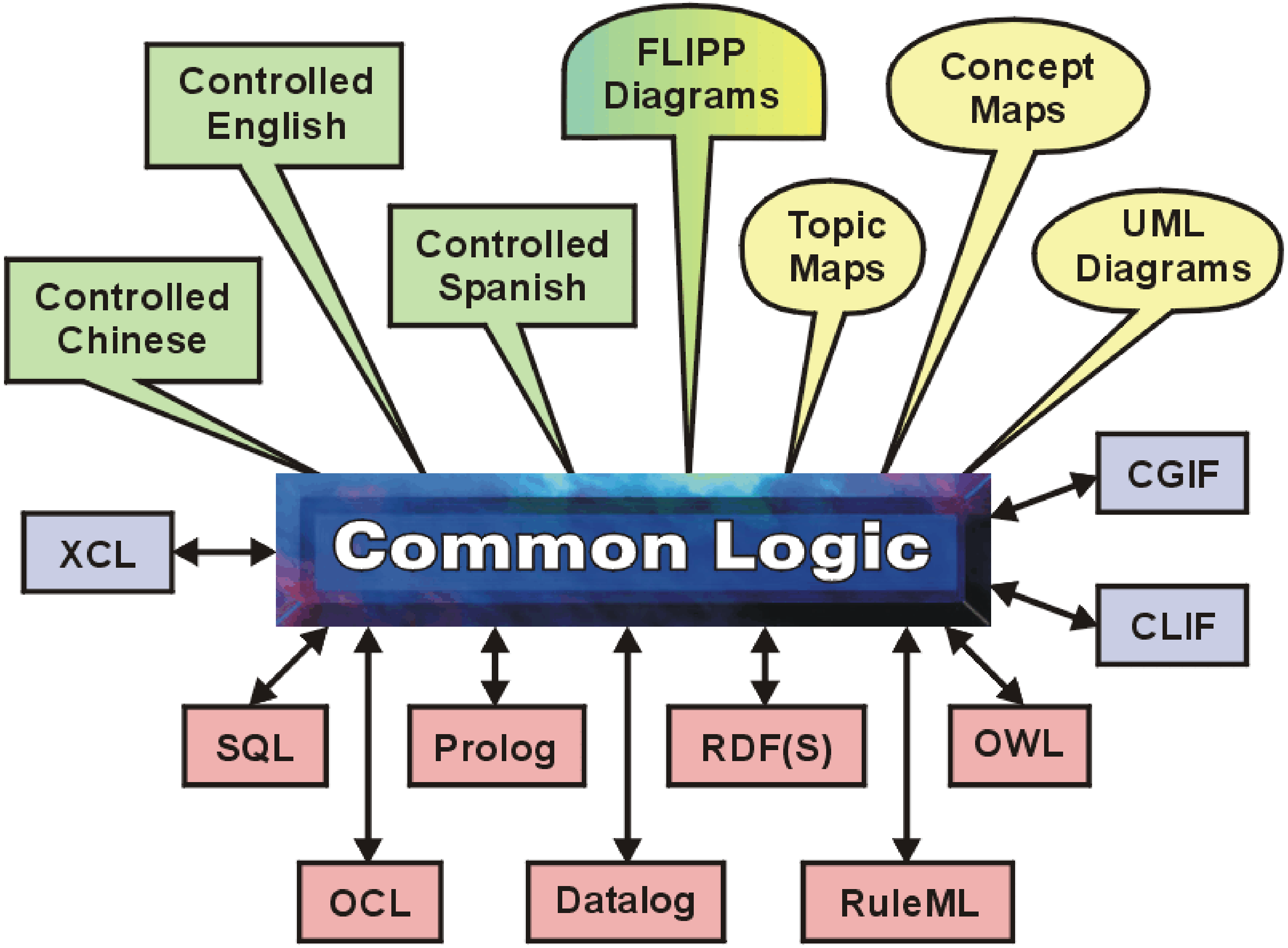
* CLIF — Common Logic Interchange Format

* CGIF — Conceptual Graph Interchange Format

* XCL — XML-based notation for Common Logic

But any dialect that adopts the common semantics can join the family.

Human Interfaces



Machine Interfaces

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 short-lived 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 the variable C 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

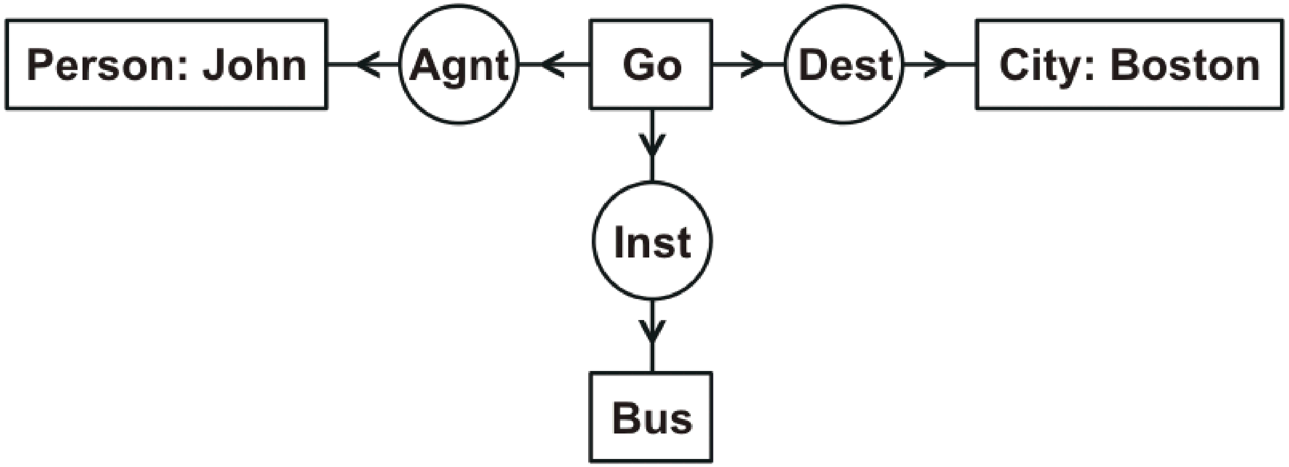
(b) some understanding of logical principles.

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 into a computable form.

CLCE: John goes to Boston by bus.

Conceptual graph display form:



Conceptual Graph Interchange Format (CGIF):

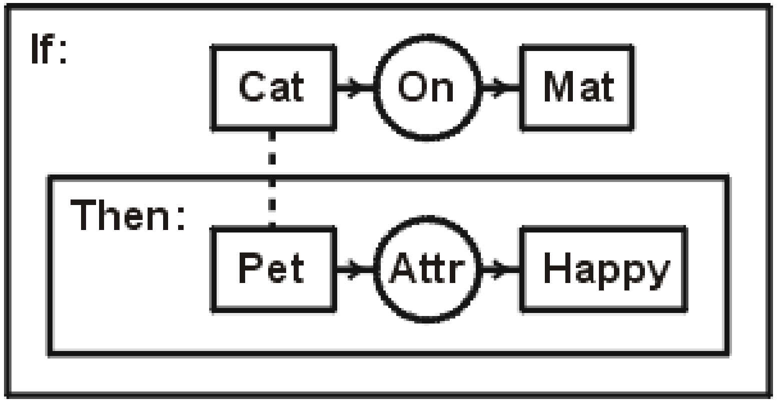
```
[Go *x]  [Person John]  [City Boston]  [Bus *y]
(Agnt ?x John)  (Dest ?x Boston)  (Inst ?x ?y)
```

Common Logic Interchange Format (CLIF):

```
(exists ((x Go) (y Bus))
          (and (Person John) (City Boston)
                  (Agnt x John) (Dest x Boston) (Inst x y)))
```

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:

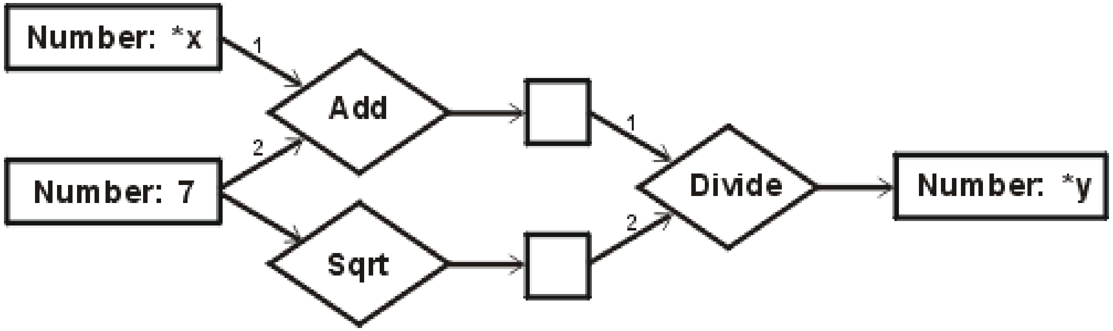
```
(forall ((x Cat) (y Mat))
   (if (On x y)
       (and (Pet x) (exists ((z Happy)) (Attr x z)))))
```

Note that most dialects of logic and natural languages permit many different ways of making 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:

Define "familiar 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.

An Example of Medical "English"

Patient 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.

Mapping English Denominator to CLCE to CLIF

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

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.

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".

CLIF generated from CLCE:

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

Choice of Representation

Two critical choice points:

1. End-user interface: Mapping informal English to CLCE.

2. IT interface: Mapping CLCE to a machine-oriented form.

Options at the first interface are determined by medical professionals and the terminology of the medical domain.

Options at the second interface are determined by IT professionals and the expressive power of the underlying logic.

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

Mapping English Numerator to CLCE to CLIF

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

CLCE: 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.

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).

CLIF generated from CLCE:

```
(Set N)
(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) ))
```

Mapping English Exclusions to CLCE to CLIF

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

CLCE: 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.

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).

CLIF generated from CLCE:

```
(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) ))
```

Defining the Ratio

CLCE: 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).

Define "x is y divided by z" as (= x (/ y z)).

Define "the count of x" as (Count x).

CLIF generated from CLCE:

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

Complete CLCE Definition

**The scope of variables extends beyond the semicolon,
but it ends at the first period.**

**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).**

Equivalent CLIF Definition

Names have global scope unless governed by a quantifier.

```
(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)) )
```

Equivalent CGIF Definition

Names have global scope; coreference labels are limited to context.

```
[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) ]
```

Readability and Comments

CLCE, CLIF, and CGIF allow comments.

But even without comments, anyone familiar with the subject matter who can read English can read CLCE.

With comments in CLCE, anyone with a minimal knowledge of the notation can read CLIF or CGIF.

Learning to write CLCE, however, requires some training.

Learning to write CLIF or CGIF requires more training.

Good tools can help users learn CLCE and can translate the CLCE to CLIF, CGIF, or XCL.

Good tools can also translate CLIF, CGIF, or XCL to and from other dialects of logic, including various kinds of diagrams.

Computational Complexity

Common Logic is very expressive because it has been designed as a superset of many different notations.

There is no standard way of using CL, but it can be used in the same manner and with the same efficiency as any of the more specialized notations. Translating any statement to CL does not in any way change its efficiency, either for better or for worse.

The medical example can be translated to the Horn clause subset of FOL and be used to answer questions in polynomial time in Prolog or many similar systems.

If the ground-level CL relations are defined by a mapping to SQL tables, then a typical implementation of CL or CLCE would invoke the SQL DB in order to do the basic data access.

Summary: The computational complexity of CL depends primarily on the nature of the problem, and it is no worse than the complexity of the same problem stated in any other language.

Mapping Common Logic to and from Other Notations

Common Logic was designed to be an intermediate language for translations among various logic-based languages.

Because Common Logic is so expressive, it is easy to map many other notations for logic into CL. (See the diagram in slide #2.)

But a CL specification can only be mapped to a less expressive notation if it observes the same limitations as the target notation.

However, it is possible to design tools that reorganize a specification in a very expressive version of logic, such as CL, and map it to an executable form in a less expressive language.

For a good example, see the following article, which describes a methodology for doing such translations:

Peterson, Brian J., William A. Andersen, & Joshua Engel (1998) "Knowledge bus: generating application-focused databases from large ontologies," *Proc. 5th KRDB Workshop*, Seattle, WA.

http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-10/

Technology for Controlled NLs

Many different controlled natural languages have been implemented. (See the suggested readings in the last slide.)

Nothing in the CLCE grammar is revolutionary, proprietary, or beyond the state of the art.

But there is a great deal of ongoing R & D to develop practical systems with good human factors.

One deterrent has been the “quirky” features of many languages that were designed for particular algorithms or data representations.

The absence of implementation-dependent features simplifies the mapping of CL to both natural languages and other varieties of logic.

The use of CL as a lingua franca for logic-based notations should facilitate interoperability among computer systems and the design of more usable human interfaces.

Related Readings

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

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

"Conceptual Graphs," by John F. Sowa
http://www.jfsowa.com/cg/cg_hbook.pdf

"A Guided Tour of Ontology," by John F. Sowa
http://www.jfsowa.com/ontology/guided.htm

Web site for Common Logic:
http://www.common-logic.org

Web site for Attempto Controlled English (ACE)
http://attempto.ifi.unizh.ch/site/

Web site for controlled natural languages:
http://www.ics.mq.edu.au/~rolfs/controlled-natural-languages/