Working for Two: a Bidirectional Grammar for a Controlled Natural Language

Rolf Schwitter

Centre for Language Technology
Macquarie University
Sydney NSW 2109, Australia
schwitt@ics.mq.edu.au

Abstract. This paper introduces the controlled natural language PENG Light together with a language processor that is based on a bidirectional grammar. The language processor has the following interesting properties: (a) it translates declarative sentences written in PENG Light into a first-order logic notation (TPTP); (b) it generates declarative sentences in PENG Light taking syntactically annotated TPTP formulas as input; and (c) it translates questions written in PENG Light into (conjunctive) queries in TPTP notation and uses the TPTP representation of the query as a starting point for generating answers in PENG Light. Moreover, the controlled natural language processor can be interfaced directly with an automated reasoner in order to resolve anaphoric references and to answer questions stated in PENG Light.

Key words: Controlled Natural Languages, Grammar Engineering, Human-Computer Interfaces, Knowledge Representation, Question Answering

1 Introduction

A controlled natural language is an engineered subset of a natural language with explicit constraints on grammar, lexicon, and style [1]. These constraints usually have the form of writing rules and help to reduce both ambiguity and complexity of full natural language. In general, controlled natural languages fall into two categories: human-oriented and machine-oriented controlled natural languages. Human-oriented controlled natural languages aim at improving text comprehension for human readers while machine-oriented controlled natural languages focus on improving text processability for machines.

During the last ten years, there has been substantial work in the area of machine-oriented controlled natural languages. These controlled natural languages have been designed and used for specification purposes, knowledge acquisition and knowledge representation, and as interface languages to the Semantic Web – among them Attempto Controlled English [2, 3], Boeing's Computer-Processable Language [4, 5], Common Logic Controlled English [6, 7], and PENG Processable English [8, 9]. Some machine-oriented controlled natural languages require the user to learn a small number of construction and interpretation rules

[3], while other controlled natural languages provide writing support that takes most of the burden of learning and remembering the language from the user [10]. The commercial success of the human-oriented controlled natural language ASD Simplified Technical English [11] suggests that people can learn to work with restricted English and that good authoring tools can drastically reduce the learning curve [12].

In this paper, we will introduce PENG Light and show that this controlled natural language can be processed by a **bidirectional** grammar in contrast to other controlled natural languages. For example, PENG Light's predecessor PENG translates declarative sentences and questions via discourse representation theory [13] into various first-order notations for automated reasoning but not backwards into controlled natural language [9]. We will demonstrate that in the case of PENG Light the same grammar can be used to translate sentences and questions into a first-order notation (that is augmented with syntactic information) and that this notation can be used as a starting point for generating answers to questions in controlled natural language.

2 PENG Light

PENG Light distinguishes between simple sentences and complex sentences. The writing process of these sentences can be supported with the help of a predictive authoring tool [10] that enforces the restrictions of the controlled language.

2.1 Simple Sentences

Simple PENG Light sentences have the following functional structure:

```
subject + predicator + [ complements ] + { adjuncts }
```

The subject has the form of a noun phrase that contains at least a nominal head in form of a noun (= common noun or proper noun). The predicator is realised by a verb that functions as the verbal head of a verb phrase. The existence of complements depends on the verb. An intransitive verb (*1*) does not take any complements, a transitive verb (*2*) takes one complement (a direct object), a ditransitive verb (*3*) takes two complements (a direct object and an indirect object), and a linking verb (*4*) takes a so-called subject complement:

1. *John Miller works.*
2. *John Miller teaches COMP249.*
3. *John Miller sends a letter to Mary.*
4. *John Miller is in the lecture hall.*

For example, in sentence (*2*), the direct object has the form of a noun phrase; in sentence (*3*), the direct object has the form of a noun phrase and the indirect object has the form of a prepositional phrase; and finally in (*4*), the subject complement has the form of a prepositional phrase. In contrast to complements, adjuncts are always optional and are used in PENG Light to describe the verb in more detail. They have the form of prepositional phrases (*5*) or adverbs (*6*):

5. *John Miller teaches COMP249 on Monday.*
6. *John Miller works exactly.*

Not only verbs but also the nominal head of a noun phrase can be described in more detail in PENG Light: either by adding an adjective (*7*) or a possessive noun (*8*) as a pre-head modifier to the common noun or by adding a variable (*9*) or an *of*-construction (*10*) as a post-head modifier:

7. *The clever professor teaches COMP332 on Monday.*
8. *Mary's father teaches a computing unit in E6A.*
9. *The professor X teaches COMP348 on Monday.*
10. *The father of Mary teaches a computing unit in E6A.*

Note that the translation of the subject noun phrase in (*8*) and (*10*) will result in the same logical representation. Variables are used instead of personal pronouns in order to establish precise anaphoric references within and between sentences and to exclude potential ambiguities (see Section 2.3 for more details).

2.2 Complex Sentences

Complex PENG Light sentences are built from simpler sentences through quantification, negation, subordination and coordination. A special form of complex sentences are conditionals and definitions.

Quantification. PENG Light distinguishes between universal quantification and existential quantification ((*11*) and (*12*)), and qualified cardinality restriction (*13*):

11. *Every professor teaches a unit.*
12. *A professor teaches every unit.*
13. *John Miller teaches* [*exactly* | *at least* | *at most*] *two units.*

There is no scope ambiguity in PENG Light since the textual occurrence of a quantifier determines its scope that extends to the end of a sentence. For example, in sentence (*11*), the universal quantifier has scope over the existential quantifier, and in (*12*), the existential quantifier has scope over the universal one. Note that qualified cardinality restriction – as illustrated in (*13*) – can only occur in complement position in PENG Light but not in subject position.

Negation. PENG Light distinguishes three forms of negation: sentence negation (*14*), noun phrase negation (*15*), and verb phrase negation (*16*) and (*17*):

14. *It is false that a professor teaches COMP225.*
15. *No professor teaches COMP225.*
16. *Every professor does not teach COMP225.*
17. *John Miller is not a professor.*

Sentence negation is realised via the constructor phrase *it is false that*; noun phrase negation is introduced via the universal negative quantifier *no*; and verb phrase negation is realised via the negatives *does not* and *is not*. Note that (*15*) and (*16*) have the same meaning and will result in the same formal representation but will carry different syntactic annotations.

Subordination. Using a relative clause is another option to modify a noun in a noun phrase. In PENG Light, a relative clause is always introduced by a relative pronoun and modifies the immediately preceding noun. Relative clauses trigger so-called filler-gap dependencies since they have a structure where a phrase (or a partial phrase) is missing from its normal position (= gap) and another phrase (= filler) stands for the missing phrase outside the normal position, for example:

18. [*John Miller*]$_{filler}$ *who* [ ]$_{gap}$ *teaches a unit supervises Mary.*
19. [*Mary*]$_{filler}$ *who John Miller supervises* [ ]$_{gap}$ *takes COMP249.*

In the case of a subject relative clause such as in (*18*), the entire noun phrase in subject position is missing, and in the case of a complement relative clause such as in (*19*), the entire complement is missing from its normal position. Note that sentence (*18*) as well as sentence (*19*) can be expressed alternatively with the help of two simple PENG Light sentences as illustrated in (*20*) and (*21*):

20. *John Miller teaches a unit. John Miller supervises Mary.*
21. *John Miller supervises Mary. Mary takes COMP249.*

Relative clauses can also be used in combination with an existential *there*-construction to clarify the scope of the existential quantifier. Instead of (*12*), we can write (*22*):

22. *There is a professor who teaches every unit.*

Coordination. PENG Light distinguishes three forms of coordination: verb phrase coordination (*23*), relative clause coordination (*24*), and modifier coordination in adjunct position (*25*):

23. *Marc does not teach COMP249 and does not supervise Mary.*
24. *John Miller supervises Mary who works in the library and who lives in Epping.*
25. *John Miller works on Monday in E6A.*

Verb phrase coordination uses the two connectives *and* and *or* and coordinates complete verb phrases. In (*23*), the negative *does not* needs to be repeated in order to negate both conjuncts. As (*24*) shows, relative clause coordination requires that the relative pronoun (*who*) is repeated after the connective (*and*) in order to exclude potential ambiguity when a verb phrase and a relative clause coordination occur in the same sentence. Modifier coordination can only be realised with the conjunctive connective *and* but not with the disjunctive connective *or*, therefore the *and* can also be omitted in (*25*).

Conditionals. PENG Light supports conditional sentences for describing hypothetical situations and their consequences. A conditional sentence consists of a subordinated clause that specifies a condition and a main clause that expresses the consequence, for example:

26. *If John Miller works on Monday then John is in the office.*

Note that the subordinated clause and the main clause in (*26*) have the same internal structure as other PENG Light sentences if we would drop the conditional connective *if-then*.

Definitions. PENG Light also supports the specification of definitions: a definition is a relation of equivalence that consists of a term to be defined and an expression that is used to define that term. The following two sentences use the key phrase *is defined as* and can be used alternatively to state a definition:

27. *A mother is defined as a woman who has at least one child.*
28. *Every mother is defined as every woman who has at least one child.*

The definition in (*27*) sounds more natural in English but the indefinite determiner (*a*) is interpreted as an universal quantifier (*every*). The language processor generates a paraphrase that clarifies the interpretation of (*27*) which looks similar to (*28*).

2.3 Anaphora Resolution

In PENG Light, proper nouns, definite noun phrases and variables (but not personal pronouns) can be used as anaphoric expressions. PENG Light resolves anaphoric references by replacing the anaphoric expression with the most recent accessible noun phrase (= noun phrase antecedent) and indicates this replacement in a paraphrase. That means a noun phrase antecedent must be accessible to be referred to by an anaphoric expression. Proper nouns always denote the same object and are accessible from everywhere. An indefinite noun phrase is not accessible from **outside**, if it occurs under negation (*29*), if it occurs in a conditional sentence (*30*) or in the scope of an universal quantifier (*31*), or if it occurs under a disjunction (*32*):

29. *John does not teach a tutorial. *The tutorial ...*
30. *If John teaches a tutorial then Sue teaches a practical. *The tutorial*
31. *Every professor teaches a tutorial. *The tutorial ...*
32. *John teaches a tutorial or teaches a practical. *The tutorial ...*

However, a noun phrase antecedent in the *if*-part of a conditional sentence such as (*33*) is accessible from the *then*-part of the same sentence:

33. *If John teaches a tutorial then Sue does not teach the tutorial.*

And a noun phrase antecedent under a disjunction – as for example in (*34*) – is accessible if the anaphoric expression occurs in one of the subsequent disjuncts:

34. *John sends a letter to Mary or brings the letter to Mary.*

An anaphoric expression can be syntactically less specific than its noun phrase antecedent as the following examples (*35-37*) illustrate:

35. *The clever professor teaches COMP332. The professor ...*
36. *The professor X teaches COMP348. X works ...*
37. *The computer scientist works at Macquarie. The scientist ...*

If we interface PENG Light with an existing ontology, then we can additionally resolve anaphoric references that rely on realisation (i.e. computing the most specific class for an individual) and on classification (i.e. computing the subclass hierarchy). For example, the ontology (or the background knowledge) might specify that *lecturer* in (*38*) is the most specific class that the individual *John Miller* belongs to and that the class *academic* in (*39*) subsumes *lecturer*:

38. *John Miller teaches COMP249 on Monday. The lecturer ...*
39. *John Miller teaches COMP249 on Monday. The academic ...*

Note that if a definite noun phrase can not be resolved by the anaphora resolution algorithm that is built into the grammar of PENG Light, then this definite noun phrase is interpreted as an indefinite noun phrase and introduces a new object into the universe of discourse. The presence of a relative clause in a noun phrase antecedent does not play a role in the determination of a suitable antecedent.

2.4 Questions

PENG Light distinguishes two types of questions: *yes/no*-questions and *wh*-questions. *Yes/no*-questions such as (*40*) investigate whether a specific situation is true or false. And *wh*-questions such as (*41*) interrogate a particular aspect of a situation:

40. *Does John Miller teach a tutorial on Monday?*
41. *When does John Miller who convenes COMP249 teach a tutorial?*

Questions are derived in a systematic way from simple PENG Light sentences: in the case of *yes/no*-questions by insertion of a *do*-operator or by inversion, and in the case of *wh*-questions with the help of a suitable query word (*who, what, which, when, where, etc.*) and insertion of a *do*-operator. Note that *wh*-questions also exhibit a filler-gap dependency similar to relative clauses and that they can be processed using the same type of gapping mechanism.

3 Implementation

The grammar of PENG Light is implemented as a definite clause grammar with feature structures and difference lists that occur as arguments in the grammar rules [14]. The language processor builds a logical formula in TPTP notation [15] for a given input sentence or generates an output string for a given TPTP formula. In order to achieve true bidirectionality, the TPTP formulas are annotated with syntactic information that supports the generation of declarative sentences in particular for question answering. PENG Light can serve as a high-level interface language to a first-order reasoner or a description logic reasoner or both, and we have experimented with all three options. In the following, we will show how sentences can be translated into TPTP notation and then into KRSS [16], the input language of the description logic reasoner RacerPro [17, 18]. Note that the focus here is not on the reasoning engine but on the bidirectional translation from PENG Light into TPTP and back into PENG Light.[1]

3.1 Analysis of PENG Light Sentences

The grammar of PENG Light does not simply interpret the verb of a sentence as a relation between a subject and a number of complements and adjuncts. Instead the event or state that is described by the verb is reified and the variable of the reified relation is linked via a number of thematic relations [19] to the objects and individuals that participate in a sentence. This allows for an uniform interpretation of obligatory complements and optional adjuncts (and provides a convenient way to deal with n-ary relations in description logics [20]). Note that we use a flat notation for representing objects, individuals, and properties in order to translate efficiently between different notations. For example, the complex PENG Light sentence:

42. *John Miller who supervises Mary teaches at least two computing units on Monday.*

results in the subsequent TPTP formula where the logical forms derived from the content words are annotated (#) with syntactic information:

```
input_formula(university,axiom,
  (? [A]: ((named(A,john_miller)#[‘John’,‘Miller’] &
    (?[B]: (named(B,mary)#[‘Mary’] &
     ?[C]: (property(C,has_agent,A) &
            event(C,supervise)#[fin,third,sg,pres,no,no] &
            property(C,has_theme,B)&contemp(C,u))))#[rc]) &
      (? [D]: (timex(D,monday)#[‘Monday’] &
                (property(E,has_time,D)#[on] &
```

[1] Only a subset of PENG Light sentences can be handled by the description logic $\mathcal{ALCQHIR}_{\mathcal{R}} + (\mathcal{D}^{-})$ that is supported by RacerPro but the TPTP to KRSS translator will reject PENG Light sentences that are not in this subset.

```
(? [F]: ((‘$min_cardinality’(F,2) &
          object(F,computing_unit)#[at,least,two]) &
  (? [E]: (property(E,has_agent,A) &
            (event(E,teach)#[fin,third,sg,pres,no,no] &
               (property(E,has_theme,F) &
                contemp(E,u)))))))))))).
```

These syntactic annotations can be used to support the generation process of sentences. For example, the annotation `[on]` indicates that the property `has_time` has been derived from a prepositional phrase with the preposition *on*, and the annotation `[fin,third,sg,pres,no,no]` signals that the event `supervise` has been derived from a finite verb in its third person singular. If we fed the above formula back to the grammar, **exactly** the input sentence (*42*) will be generated, and this is true for arbitrarily complex PENG Light sentences.

The lexicon of PENG Light contains apart from syntactic information partial logical forms for content words and function words. Here are two (incomplete) lexical entries for content words: the first one for a common noun and the second one for a transitive verb:

```
lex( wf:[computing,unit],... fol:X^object(X,computing_unit) ).
lex( wf:[teaches],... fol:E^X^Y^(property(E,has_agent,X) &
     event(E,teach) & property(E,has_theme,Y) & contemp(E,u)) ).
```

In the first example, the common noun *computing unit* is a compound noun that is stored as a list of tokens in the lexicon and its translation will result in an object in the logical form. The lexical entry for the transitive verb *teaches* illustrates that the logical form for this verb consists of a reified event that will connect the subject and the object of the sentence via thematic relations.

The final semantic representation of a sentence will be crucially influenced by the function words that establish its logical structure. For example, the subsequent lexical entries show that quantifiers are treated as generalised quantifiers which provide a substantial part of the scaffolding for the formal representation:

```
lex( wf:[every], ... fol:(X^Res)^(X^Scope)^(! [X]: (Res => Scope)) ).
lex( wf:[a], ... fol:(X^Res)^(X^Scope)^(? [X]: (Res & Scope)) ).
lex( wf:[exactly,two], ... fol:(X^Res)^(X^Scope)^(? [X]:
     ((‘$min_cardinality’(X,2) & Res) & Scope)) ).
```

A generalised quantifier is a relation between two sets `Res` and `Scope` (where `Res` is the restrictor and `Scope` is the scope). During processing of an input sentence, the term `X^Res` collects the logical forms that can be derived from the noun phrase, and the term `X^Scope` collects the logical forms that can be derived from the entire verb phrase of the sentence. The following top-level grammar rule `s` combines the logical forms for a noun phrase in `n3` with the logical forms for a verb phrase in `v3` and returns the resulting logical form `LF` in `s`:

```
s( mode:M, ... fol:LF, ... ) -->
   n3( mode:M, ... fol:(X^Scope)^LF, ... ),
   v3( mode:M, ... fol:(X^Scope), ... ).
```

In the case of an existentially quantified noun phrase in subject position, the variable `LF` will eventually be instantiated by a term of the form: `(? [X]: (Res & Scope))`. The following grammar rule `n3` shows that this term is issued by the determiner `det` and that the restrictor `X^Res` is processed in the subsequent noun phrase `n2` (before the anaphora resolution algorithm is triggered):

```
n3( mode:M, ... fol:(X^Scope)^LF, ... ) -->
    det( ...  fol:(X^Res)^(X^Scope)^LF, ... ),
    n2( mode:M, ... fol:(X^Res), ... ),
    { anaphora_resolution( ... ) }.
```

The scope `X^Scope` "flows" from the noun phrase `n3` into the verb phrase `v3` and then into `v2`. Note that the following grammar rule `v2` is only used for analysing sentences (`mode:ana`) but not for generating sentences (we will discuss this issue in more detail in Section 3.3):

```
v2( mode:ana, ... fol:X^Scope, ... )  -->
    v1( mode:ana, ... fol:E^X^V1, ... ),
    p2( mode:ana, ... fol:E^V1^Scope, ... ).
```

The logical forms for the verb and its complement(s) are processed in `v1` and the result `E^V1` (with the event variable `E`) is pushed together with the scope `Scope` into the grammar rules for the prepositional modifier (first into `p2` and then `p1`) where this information is finally combined with the logical form for the modifier `Mod`:

```
p1( mode:M, ... fol:E^V1^Scope, ... ) -->
    prep( ..., fol:E^Y^Mod, ...),
    n3( mode:M, ... fol:(Y^(Mod & V1))^Scope, ... ).
```

To complete the picture, we present below the top-level grammar rule that is used for processing declarative PENG Light sentences:

```
s( mode:M, fol:LF, gap:G1-G3, para:P1-P3, ant:A1-A3 ) -->
   n3( mode:M, syn:[subj,Per,Num], sort:_, fol:(X^Scope)^LF,
       gap:[]-G2, para:P1-P2, ant:A1-A2 ),
   v3( mode:M, crd:_, syn:[fin,Per,Num,_,_,_], fol:E^(X^Scope),
       gap:G1-G3, para:P2-P3, ant:A2-A3 ).
```

Apart from the logical form, this grammar rule deals with syntactic constraints (`syn`) and uses three additional feature structures that are implemented as difference lists: the first one (`gap`) deals with filler-gap dependencies, the second one (`para`) builds up a paraphrase during parsing, and the third one (`ant`) maintains the accessible noun phrase antecedents for anaphora resolution. Anaphora resolution is done during the parsing process: whenever a definite noun phrase or a proper noun have been processed, the anaphora resolution algorithm is triggered. This algorithm can query the description logic knowledge base and check for class membership and subsumption, and it dynamically updates the paraphrase (`para`) which clarifies the interpretation of a sentence.

The TPTP representation of a sentence (or an entire paragraph) can be further translated into KRSS notation; in our case, the translation of (*42*) results in the following assertions:

```
( related supervise_1 john_miller has_agent )
( instance supervise_1 supervise )
( related supervise_1 mary has_theme )
( relatated teach_2 monday has_time )
( instance sk_1 (at-least 2 computing_unit ))
( related teach_2 john_miller has_agent )
( instance teach_2 teach )
( related teach_2 sk_1 has_theme )
```

Note that `supervise_1` and `teach_2` are constants – without reification it would not be possible to represent the sentence (*42*) in description logic.

3.2 Analysis of PENG Light Questions

In PENG Light, questions are translated in a similar way as declarative sentences, and their processing uses most of the same grammar rules. Similar to relative clauses, *wh*-questions evoke filler gap-dependencies, for example:

43. [*When*]$_{filler}$ *does John Miller teach a computing unit* [ ]$_{gap}$*?*

This filler-gap dependency is handled in the grammar via a difference list (`gap`) that moves the filler term for the query word back into the position where the gap occurred. There are specific grammar rules that recognise gaps and remove filler terms from the difference list. The translation of (*43*) results in the following conjunctive TPTP query which contains the free variable `B` and the property `property(C,has_time,B)` that have both been derived from the query word *when* using information from the lexicon:

```
input_formula(university,conjunctive_query,
  ((? [A]: (named(A,john_miller)#['John','Miller'] &
            (free_variable(B) & (property(C,has_time,B) &
    (? [D]: (object(D,computing_unit)#[computing,unit] &
    (? [C]: (property(C,has_agent,A) &
              (event(C,teach)#[inf,third,sg,pres,no,no] &
               (property(C,has_theme,D) &
                contemp(C,u)))))))))) => answer(B))).
```

The language processor first stores this TPTP query that will later serve as a template for generating answers and then translates the TPTP query into an nRQL query, the query language of the description logic reasoner RacerPro [18]:

```
(retrieve (?2) (and (?1 ?2 has_time) (and (?3 computing_unit)
(and (?1 john_miller has_agent) (and (?1 teach) (?1 ?3 has_theme))))))
```

Here `(?2)` is the head of the nRQL query and the rest is the body of the query. The head specifies the format of the query result and the body specifies the retrieval conditions.

3.3 Generation of PENG Light Sentences

Let us assume that the description logic reasoner RacerPro finds the following answer: `(((?2 1400)))` for (*43*). The language processor takes the TPTP formula that has been stored for the question and transforms it into a formula for a declarative sentence. In our case, it replaces the term for the free variable `B` by an existentially quantified expression `(? [B]: (timex(B,1400)))` and updates the syntactic annotation of the verbal event since the answer must consist of a finite verb (`fin`) and not an infinite one (`inf`). The transformed TPTP formula is then sent to the grammar and a complete sentence is generated as an answer. Note that the annotated logical form drives this generation process. That means the logical form needs to be split up at various points into the relevant parts, in the case of a verbal modifier this splitting requires a specific grammar rule (`mode:gen`), otherwise generation would be blind for the obvious:

```
v2( mode:gen, ..., fol:E^X^(?[Y]: (Res & (Mod & V1))), ... ) -->
    v1( mode:gen, ..., fol:E^X^V1, ... ),
    p2( mode:gen, E^V1^(?[Y]: (Res & (Mod & V1))), ... ).
```

Here the variable `Res` contains the logical form for the temporal expression *2pm*, the variable `Mod` the logical form for the preposition *at* and the variable `V1` the logical form for the verb phrase *teaches a computing unit*. This will eventually results in the answer: *John Miller teaches a computing unit at 2pm.*

4 Conclusions

This paper presented the controlled natural language PENG Light and introduced a bidirectional grammar that can be used to translate PENG Light sentences and questions into syntactically annotated TPTP formulas and to generate answers to questions by transforming TPTP formulas for questions into TPTP formulas for declarative sentences. The novelty of this approach is that most of the same grammar rules can be used for the following three central tasks: analysing sentences, processing questions and generating answers to questions. The bidirectional grammar is written in a very modular way and only a small number of the grammar rules are task-specific. PENG Light can be interfaced directly with a reasoning engine and can serve as a high-level knowledge specification and query language. Note that the writing process of PENG Light sentences can be supported by a predictive text editor. Such a predictive text editor enforces the restrictions of the controlled natural language and guides the user of the controlled natural language via lookahead information. This lookahead information can be harvested directly from the grammar rules of the controlled natural language while a sentence is written. There are many potential applications that can benefit from a high-level interface language like PENG Light. We are currently investigating the usefulness of controlled natural language as an interface language to the Semantic Web and as a language for writing business rules.

References

1. Kittredge, R.I.: Sublanguages and controlled languages. In: Mitkov (ed.), The Oxford Handbook of Computational Linguistics, Oxford University press, pp. 430–447 (2003)
2. Fuchs, N.E., Schwertel, U., Schwitter, R.: Attempto Controlled English – not just another logic specification language. In: Proceedings of LOPSTR'98, pp. 1–20 (1998)
3. Fuchs, N.E., Kaljurand, K., Kuhn, T.: Attempto Controlled English for Knowledge Representation. In: Reasoning Web, Fourth International Summer School 2008, LNCS 5224, pp. 104–124 (2008)
4. Clark, P., Harrison, P., Jenkins, T., Thompson, T., Wojcik, R.: Acquiring and Using World Knowledge Using a Restricted Subset of English. In: Proceedings of FLAIRS05, pp. 506–511 (2005)
5. Clark, P., Harrison, P., Thompson, J., Wojcik, R., Jenkins, T., Israel, D.: Reading to Learn: An Investigation into Language Understanding. In: Proceedings of AAAI 2007 Spring Symposium on Machine Reading, pp. 29–35, 2007.
6. Sowa, J.F.: Common Logic Controlled English. Draft, 24 February 2004, `http://www.jfsowa.com/clce/specs.htm` (2004)
7. Sowa, J.F.: Common Logic Controlled English. Draft, 15 March 2007, `http://www.jfsowa.com/clce/clce07.htm` (2007)
8. Schwitter, R.: English as a Formal Specification Language. In: Proceedings of DEXA 2002, 2-6 September 2002, Aix-en-Provence, France, pp. 228–232 (2002)
9. Schwitter, R., Tilbrook, M.: Meaningful Web Annotations for Humans and Machines using Controlled Natural Language. In: Expert Systems, 25(3), pp. 253–267, (2008)
10. Schwitter, R., Ljungberg, A., Hood, D.: ECOLE – A Look-ahead Editor for a Controlled Language. In: Proceedings of EAMT-CLAW03, May 15-17, Dublin City University, Ireland, pp. 141–150 (2003).
11. ASD STMG: ASD Simplified Technical English. Specification ASD-STE100, International specification for the preparation of maintenance documentation in a controlled language, Issue 4, January (2007)
12. Thompson, C.W., Pazandak, P., Tennant, H.R.: Talk to Your Semantic Web. In: IEEE Internet Computing, 9 (6), pp. 75–79 (2005)
13. Kamp, H., Reyle, U.: From Discourse to Logic. Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory Kluwer, Dordrecht (1993)
14. Pereira, F.C.N., Shieber, S.M.: Prolog and Natural-Language Analysis, CSLI Lecture Notes, Number 10 (1987)
15. Sutcliffe, G., Suttner, C.B.: The TPTP Problem Library: CNF Release v1.2.1. In: Journal of Automated Reasoning, 21(2), pp. 177–203 (1998)
16. Patel-Schneider, P.F., Swartout, B.: Description-Logic Knowledge Representation System Specification from the KRSS Group of the ARPA Knowledge Sharing Effort, 1 November (1993)
17. Haarslev, V., Möller, R.: Racer: A Core Inference Engine for the Semantic Web, in: Proceedings of EON2003, pp. 27–36, (2003)
18. Racer Systems: RacerPro User's Guide, Version 1.9.2, Technical report, `http://www.racersystems.com` (2007)
19. Parsons, T.: Events in the Semantics of English: A Study in Subatomic Semantics, MIT Press (1994)
20. Noy, N., Rector, A.: Defining N-ary Relations on the Semantic Web, W3C Working Group Note 12 April 2006, http://www.w3.org/TR/swbp-n-aryRelations/ (2006)